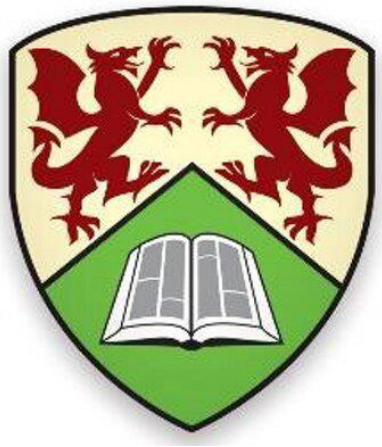
Methods for 3D Shape Description, Indexing, Matching and Retrieval

Ekpo Otu

A thesis presented for the degree of
Doctor of Philosophy



Department of Computer Science
Aberystwyth University
Wales, United Kingdom
September, 2021

Methods for 3D Shape Description, Indexing, Matching and Retrieval

Ekpo Otu

Abstract

3D Models are essential to modern-day Computer Vision (CV) applications, as 3D data representations are now being widely used instead of the already popular 2D image representation, for tasks such as object detection, classification, retrieval, and modeling, etc. However, many different problems have been associated with the representation, description, indexing, matching, retrieval, and classification of 3D models found in rapidly emerging domain-specific and 3D benchmarks datasets. One of such problems is developing a robust, compact, yet computationally efficient 3D shape descriptor. Although simple to complicated knowledge-based 3D shape representation and matching methods have been proposed, the simple methods usually lack efficiency and needed robustness (i.e. discriminating power), while the complicated methods, with remarkably high retrieval and classification accuracies, are either computationally prohibitive or rely on remarkably large surface points for optimum performance, which negatively impact processing speed and storage. Recently, the Deep Learning (DL) methods have been used to develop highly robust and compact 3D shape descriptors which also produce remarkably high retrieval and classification accuracies. However, the DL approaches are highly data-driven, which requires a lot of training data and high-powered GPUs to run successfully. Another important research problem is developing a 3D shape descriptor that can generalise across a wider range of datasets, each of which presents unique retrieval and classification challenges to the shape descriptor.

This thesis focuses on the knowledge-based approach to propose three novel, robust, and computationally efficient methods for 3D shape retrieval and classification. Our first novel research contribution is a local 3D shape descriptor called the Augmented Point Pair Features Descriptor (APPFD). Our second novel contribution is the Hybrid Augmented Point Pair Signature (HAPPS), developed to further improve the overall robustness of the APPFD, while providing invariance to rigid 3D objects. Finally, we propose an improved method called the Agglomeration of local APPFDs with Fisher Kernel and Gaussian Mixture Model (APPFD-FK-GMM), which aggregates d-dimensional local descriptors into a single more compact vector representation, with improved performances. The proposed methods are statistically based, and able to effectively describe the local - global geometry of 3D mesh or point cloud surfaces, using as low as 3500 points samples from each surface, and capable of generalising across a wider range of datasets containing rigid and non-rigid 3D objects. The latter method produces robust, compact, and concise 3D shape signature that support more-efficient indexing and matching, for retrieval and classification tasks.

The accuracies and robustness of our methods have thoroughly been examined

and compared to several other state-of-the-art (data-driven and knowledge-based) methods, using nine different standard SHape REtrieval Contests (SHREC) 3D benchmark datasets, including the most recent. This thesis provides exhaustive (quantitative and qualitative) comparative analyses of performance evaluation results, for each dataset and retrieval challenge. The following Information Retrieval (IR) evaluation metrics were adopted to assess the performances (accuracies and robustness) of all shape descriptors: Nearest Neighbour (NN), Firs- Tier (FT), Second-Tierm(ST), e-Measurem(E), Discounted Cumulative Gain (DCG), mean-Average Precision (mAP), normalised-Discounted Cumulative Gain (nDCG), Area Under Curve (AUC), and Precision-Recall Curve (PRC) plots. Results of experimental evaluations reveal outstanding retrieval performances by our proposed methods, compared with several other state-of-the-art methods.

We demonstrate the superiority of the HAPPS method over several other state-of-the-art methods on the SHREC'18 protein dataset. In several other experimental evaluations, our method still outperforms many of the state-of-the-art methods, including ranking top 2 or 3 position in most cases, competing very closely with the overall best performing methods for each of those retrieval challenges and datasets. Generally, the APPFD method is robust to objects with holes (i.e. with large missing surface parts) and noise, and we demonstrate that both the APPFD and HAPPS methods are highly discriminative, efficient, and capable of effectively representing 3D point clouds and triangular meshes. In addition, we demonstrate the high performance of the APPFD-FK-GMM method, which rivals both the APPFD and HAPPS methods, even with about 98% reduction of the final fv from these methods, thus providing both robustness and compact representation of 3D objects for easier indexing and faster matching.

Declaration of Authorship

DECLARATION
This thesis has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

13.09.2021
Date

Signature (Ekpo Ekpo Otu)

STATEMENT 1
This thesis is the result of my own investigations, except where otherwise stated. Where correction services have been used, the extent and nature of the correction is clearly marked in a footnote(s). Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

13.09.2021
Date

Signature (Ekpo Ekpo Otu)

STATEMENT 2
I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

13.09.2021
Date

Signature (Ekpo Ekpo Otu)

Dedication

I dedicate this thesis, first to the almighty God for His grace, strength, and ability to finish my Ph.D., despite the many challenges. Next, to my beloved child, Emmanuella Ekpo Otu, for her understanding, endurance, patience, and cooperation during my entire Ph.D. To my parents, siblings, and the Nigerian government including others, too numerous to mention, my Ph.D. work and effort is also dedicated to you.

Acknowledgement

It has now been possible for me to finish my Ph.D. study/research work, following what seemed like an endless journey. This success story would not have been possible without my academic supervisors: Prof. Reyer Zwiggelaar, Prof. Yonguai Liu, and Dr. David Hunter. You have been exceptional, selfless, kind, highly supportive, encouraging, and generous in providing me with the needed guidance, advise, feedback, encouragements and much more. Besides supervising my Ph.D. research to a successful completion, you all have inspired me to excellence in many aspects, including that I have learned from your exceptional leadership styles, among other qualities. I still struggled to find suitable vocabularies to express my gratitude and describe my appreciation to you, most especially Prof. Yonguai Liu and Prof. Reyer Zwiggelaar, but words would never be enough to appreciate and acknowledge your support for me throughout this journey, and saying "Thank you" would never do.

Special acknowledgement also goes to other academic and non-academic staff of the computer science department, Aberystwyth University, who have been immensely assisting and supportive to me during my Ph.D. Thanks to Dr. Myra Wilson, Dr. Richard Jensen, Prof. Bernie Tiddeman, Ms. Michelle Schemes, Mrs. Margaret Walker, Mr. Andy Spence, Prof. Chris Price, Dr. Edel Sherratt, Meinir Davies, Dr. Hannah Dee, Dr. Amanda Clare, Mr. Dave Price, and all VGVG members where one point or the other I have had the privilege of either getting your support or learning from you. I also acknowledge Rosa Soto of the international office, including Kim Broom, Paul Gatehouse, and the compliance team for all their support. To all my friends and colleagues in the VGVG lab and department, too numerous to mention, working alongside and sharing ideas with you was fun, engaging, and inspiring.

Beyond the university, I greatly acknowledge the church family and leadership at St. Mikes, Aberystwyth, where I have found deep sense of belonging, moral, mental, and spiritual support. Special thanks to the leadership and members of the music/worship team, and FLOW group, for your prayers, encouragements and giving me the opportunity to serve. I specially recognise Dr. Richard Jensen and Dr. (Mrs.) Elaine Jensen for all your support and prayers.

This acknowledgement would be incomplete without specially recognising Prof. Ndem Ayara, Prof. Emmanuel Ikpeme, High-Chief, (Eld.) Prof. Eyo E. Nyong, Amb.& Hon. (Mrs.) Nkoyo Toyo, Eld. Barr. Efefiom Ekong (SAN) and your dear wife, Princess Uzodinma Ekong (of blessed memory), Dr. Solomon O. Ita, Prof. (Mrs.) Susan Etim, Dr. (Mrs.) Caroline Aboh, and Dr. Enoima Umoh, for your supports and encouragements regarding my Ph.D. programme. I also acknowledge

Ms. Yasmin Pemberton-Brown, Alh. Dr. Aminu Anas, and other friends for standing by me at the right time.

Finally, I specially acknowledge the Tertiary Education Trust Fund (TETFund), Nigeria and the management of the Cross River University of Technology (CRUTECH), Calabar, Nigeria, for some financial support, including the department of Computer Science, Aberystwyth University for awarding me the CSD01 Scholarship. The generosity of the Aberystwyth University hardship fund is also greatly acknowledged, without which it would have been impossible to complete my Ph.D.

Contents

List of Figures

List of Tables

List of Acronyms

ST Second Tier 3

VFH Viewpoint Feature Histogram 36

VTK Visualization ToolKit 16

WKS Wave Kernel Signature 39

Chapter 1

INTRODUCTION

1.1 Background

Presently, 3D data formats or representations are preferable to the already very popular 2D image representation, in several CV related projects/tasks, such as object detection, retrieval, classification, detection, modeling, and robot navigation. Computational techniques in 3D shape analyses and processing are highly relevant and applicable in medicine, robotics, industrial applications, 3D games development, and virtual reality, etc. Consequently, it has now become easier to practically acquire 3D models of any object due to freely available 3D modelling software, low-cost 3D scanning/acquisition devices, and the ability to scan 3D shapes using mobile devices equipped with photogrammetry [143] technology. As direct consequences of these: (i) there is an exponential rise in the available number of 3D models on the internet and domain-specific databases (e.g. the Protein Data Bank (PDB) [15] with biological macromolecules [182, 118], the AIM@SHAPE shape repository [247], the national design repository for Computer-aided Design (CAD) models [238], and CAESAR for Anthropometry [5], etc.), (ii) several 3D benchmark datasets are rapidly-emerging, such as the Princeton Shape Benchmark (PSB) [216] and the SHape REtrieval Contest (SHREC) benchmark datasets, and (iii) 3D models now have numerous application areas (e.g. video games character and virtual reality 3D models). However, an important aspect of research remains the development of concise, robust, and efficient 3D shape representation and methods to facilitate matching and retrieval of desired 3D objects from these repositories. The advantages of developing such 3D shape representations or retrieval methods are numerous. For example, if we consider the scenario in Section 1.2 where an existing 3D model/design for an additive manufacturing [264, 69] company needs to be obtained from a database of millions of other 3D objects, a compact, concise, and robust 3D representation (3D descriptor), including an efficient shape retrieval method is needed to obtain the correct results in a time-effective manner. A key component of such retrieval methods is estimating a shape descriptor that can robustly represent shape information, which should represent the underlying surface structure. Features, i.e. geometric properties, or physical measurements, are extracted from the surface of a given 3D object locally (for local shape descriptors), globally (for global shape descriptors), or both for hybrid shape descriptors on which this study concentrates.

The process of developing appropriate shape descriptors for efficient matching,

retrieval and classifiction basically constitute a *'3D shape search'* problem. Also, in CV, Computer Graphics (CG), Pattern Recognition (PR), and other related disciplines, measuring the (dis)similarity between 3D objects, including developing a shape descriptor that is generalizable across a wider range of datasets, are some common research issues regarding objects classification, recognition, detection, and retrieval tasks, etc. However, searching for a particular 3D object from an enormous collection of 3D objects (3D repository) can be an incredibly challenging and painstaking process, especially when the collection is un-organized into classes or categories. Even if classification or categorization of objects are enabled for 3D repositories, it is still difficult to find a query object within a category, especially where the number of 3D objects within that category is large. A number of open problems associated with 3D shape retrieval have not been satisfactorily addressed. One of such problems is finding a concise representation of 3D shapes [54], but majority of the existing 3D retrieval algorithms and matching techniques are too complex, computationally expensive, or not sophisticated enough to accurately describe 3D shapes. A widespread problem remains that of semantic gap [79, 116], which is concerned about the difference between representation of extracted features from 3D shapes and their actual visual perception. The semantic problem involves a collection of smaller problems, and ongoing research effort has been towards resolving these. To address the *3D shape search* or shape retrieval problems, the problem of creating a concise representation of 3D shapes must first be solved, which involves developing a concise, robust (i.e. highly discriminating) and efficient shape descriptor to accurately represent 3D objects. A shape descriptor, however, is a compact mathematical description of a given 3D object, often represented by a vector, a graph, or real numbers, in such a way that its complexity is much less than its corresponding original 3D representation (see Section 2.3 for further details regarding 3D shape descriptors).

The motivation for this research is summarised in Section 1.2, and our main goal is to build upon exiting techniques to propose a new set of compact, concise, robust, and computationally efficient descriptor for 3D meshes and point clouds retrieval. We begin by examining two broad approaches to 3D shape retrieval (data-driven and knowledge-based, described in Section 2.5), and adopt the knowledge-based approach considering that it does not require large training dataset or high computing power to succeed. Next, we reviewed a wide range of knowledge-based 3D shape descriptors/retrieval methods, where the statically-based approach is found to be the most popular and convenient to implement considering its ability to reduce 3D objects (dis)similarity into merely histograms comparison. Statistical descriptors also produce highly impressive results in 2D/3D objects detection, classification, and retrieval tasks.

In this thesis, we propose a number of statistically-based 3D shape descriptors that support efficient indexing and matching of rigid and non-rigid 3D objects (for rigid shapes, the distances between points in different coordinate systems will not change, while non-rigid shapes are subject to affine transformations from one coordinate system to another). First, as part of our research contributions, a novel statistically-based local 3D shape descriptor called the Augmented Point Pair Feature Descriptor (APPFD) is proposed, which is robust and computationally efficient. It describes the geometry of local surface patches around key points for 3D mesh and

point cloud data using minimal number of sample points from their surfaces. Secondly, considering that the APPFD is a local descriptor, which is most suitable for non-rigid 3D objects, assuming the local patterns extracted will not change significantly from the query shape to the database shape (i.e the shape in the database), we propose two variants of the Hybrid Augmented Point Pair Signature (HAPPS), to further improve the overall performances and robustness of the APPFD, including providing invariance to rigid objects. The first variant, HAPPS-1 involves the combination of the APPFD and a global Histogram of Global Distances (HoGD) descriptors, while the second variant, HAPPS-2 combines the APPFD with a global Multi-view 2D Projection (M2DP) [80] descriptor. Through extensive experimental evaluations, we demonstrate that the APPFD and the HAPPS retrieval methods have excellent retrieval accuracies on several benchmark datasets. However, although the APPFD and HAPPS methods produces shape signatures that are concise to store, easy to index, and straightforward to match, the lengths of their final feature-vectors (fv) are very high-dimensional, which takes longer time to match. Consequently, further investigation is carried out, into a potential technique that could further shorten the length of the APPFD and/or the HAPPS fv to produce a more compact final descriptor without losing retrieval accuracies. Finally, we introduce an improved method called the Agglomeration of local APPFD with Fisher Kernel and Gaussian Mixture Model (APPFD-FK-GMM), with the goal of aggregating d-dimensional local descriptors into a single vector representation.

Currently, to the best of our knowledge, there is hardly a knowledge-based 3D shape descriptor that has been tested across a wide variety of 3D datasets (each of which presents a unique retrieval challenge to the shape descriptor), while recording excellent retrieval performances across board. Besides, many of the existing knowledge-based 3D shape descriptors (defined in Section 2.5.2) require 3D meshes and point cloud with remarkably high Level-of-Details (LoD) for optimum performance, which negatively impacts processing speed and storage. The knowledge-based, statistical 3D shape retrieval methods we propose in this thesis are robust across several 3D benchmark datasets, including having the ability to distinguish between similar and dissimilar 3D objects. Our methods use comparatively incredibly low number of surface points sample (i.e. low LoD), unlike other methods which require objects with high LoD.

Evaluating the performances of one or more 3D shape descriptors across different datasets is a challenging task. Thankfully, the annual 3D SHREC provides a framework to systematically evaluate the performances of 3D shape retrieval methods (i.e. 3D shape descriptors) on a variety of 3D benchmark datasets, each of which present unique retrieval complexities, and some consisting a mixture of rigid, non-rigid, watertight as well as non-watertight 3D models. The SHREC framework allows for unbiased comparison of a given shape descriptor against several others (e.g. state-of-the-art descriptors) for a particular retrieval challenge or task. Following extensive retrieval experiments, we adopt several standard Information Retrieval (IR) performance evaluation metrics, such as the Nearest Neighbour (NN), First Tier (FT), Second Tier (ST), E-measure (E), Discounted Cumulative Gain (DCG), mean Average Precision (mAP), normalised Discounted Cumulative Gain (nDCG), Area Under Curve (AUC) and Precision-Recall Curve (PRC) plots, to compare the performance

results (i.e. retrieval accuracies) of each of our proposed methods against several other state-of-the-art methods, using several different 3D benchmark datasets provided by SHREC retrieval challenges, up to the most recent ones. The APPFD method was evaluated against two different datasets: SHREC'12 and SHREC'14 datasets, which contains 1200 and 8987 3D objects, respectively. Next, we evaluate the HAPPS method against six different datasets: the SHREC'10, SHREC'11, SHREC'17, SHREC'18, SHREC'19, and SHREC'20 protein datasets, containing 200, 600, 100, 2267, 5298, and 588 3D objects, respectively. Finally, we adopt four of the most recent SHREC datasets, which are the SHREC'17, SHREC'18, SHREC'19 and SHREC'20 relief (with 220 3D objects) datasets to evaluate our APPFD-FK-GMM method. In each evaluation phase, we provide qualitative and quantitative comparisons of our methods' retrieval performances against several other state-of-the-art methods for that particular retrieval challenge/dataset. Additionally, we investigate effects of different (dis)similarity metrics on the overall performance of shape descriptors by comparing the results of five distance metrics (Cosine, Euclidean, Earth Mover's Distance (EMD), Squared-Euclidean Distance (SED) and Kullback-Liebner Divergence (KLD)) with each of our proposed 3D shape descriptors and finally adopts the metric with the best overall performance accuracies for that descriptor.

The results of all experimental evaluations are presented in this thesis, which reveals outstanding retrieval performances of all our proposed methods, compared with several other state-of-the-art methods for each 3D shape retrieval challenges/task that the performances of our methods are evaluated against. We demonstrate the superiority of the HAPPS-1 method over several other state-of-the-art methods on the SHREC'18 protein dataset. In several other experimental evaluations, our method still outperforms many of the state-of-the-art methods, including ranking top 2-3 in most cases, competing very closely with the overall best performing methods for each of those retrieval challenges and datasets. Intuitively, 3D surface with holes or other defects would result in the features or underlying geometry of that portion of the local surface not to be accurately captured, represented,or described, thereby adversely affecting the robustness/descriptiveness of a typical retrieval method (descriptor) applied to such defective surface. Generally, however, the APPFD method is robust to objects with holes (i.e. with large missing surface parts) and noise, and we demonstrate that both the APPFD and HAPPS methods are highly discriminating, efficient, and capable of effectively representing 3D point clouds and triangular meshes. In addition, we demonstrate the high performance of the APPFD-FK-GMM method, which rivals both the APPFD and HAPPS methods, even with about 98% reduction of the final *fv* from these methods, thus providing both robustness and compact representation of 3D objects for easier indexing and faster matching. A typical approach to 3D shape retrieval is to compute a compact representation (descriptors) for all database 3D objects, which are then matched to determine the (dis)similarity between any two objects. Our approach is capable of approximating the performance of a standard metric for 3D objects matching, including providing the needed robustness and efficiency in a real-time 3D shape retrieval system. Overall, the standard approach to the *3D shape search* problem is the adoption of a 3D-Content-based Retrieval System (3D-CBRS), fully described in Section 2.2.4.

1.2 Motivation

Originally, this research is motivated by the following real-life scenario and practical need. When we consider the time and cost of designing a 3D model for any given product by a parts manufacturing company who is into product design, for instance, there is need to ascertain whether the product had previously been designed by searching existing archive for the model or create a new one. To search, there must be some sort of database that stores previous models for the company and a search engine, which provides the functionality to find and re-use existing models to accelerate new products design and overall production processes. By implication, the company would be able to save a huge amount of time and cost. However, assuming the 3D model designs of some products already exists and the company has no effective means of checking or searching for previously designed/stored models but have to re-design or re-model such product(s), the result could be a huge financial cost and time for the company if every product design must follow such trend, considering the expenses for designing a single model. For example, if it costs £10 each to develop a single model, a database including 1 million models which 10% of them are redesigned would amount to a total cost of £1m, which could be saved through 3D shape retrieval and search engine implementation. Alternatively, suppose there is a large database of 3D models containing the company's product design, and a 3D search engine already in place, what effective 3D search strategy, methods, and techniques exists or could be implemented to retrieve desired 3D models? The scenario described here became our primary motivation for embarking on this research, and this thesis addresses these concerns by leveraging on existing techniques to propose efficient 3D shape retrieval methods and techniques for feature extraction, and robust signatures for 3D representation which rivals state-of-the-art methods.

In addition, considering the wide application areas of 3D shape retrieval methods and associated techniques (see Section 1.6), including the numerous attention attracted by 3D content-based shape retrieval (3D-CBSR) [152, 169, 48], following the successes of 2D Content-based Image Retrieval (CBIR) [246, 124, 189] over the past two to three decades, we found that each sub-process of the overall 3D-CBRS, such as data pre-processing, feature extraction, shape descriptor computation, indexing and matching (see Section 3.1) has need or chances for improvements. Essentially, the overall outcome of any Content-based Shape Retrieval (CBSR) system depends on the computational accuracies and efficiency in each sub-stage of the system. For example, while colour may be a particularly good feature for 2D images, extracting such feature to represent a 3D surface would result to an incredibly low discriminating feature to represent the 3D object, which would adversely affect the overall descriptiveness of the final shape signature for such object. Consequently, any CBSR system adopting such retrieval algorithm would have poor overall performances. In this thesis, we are further motivated to learn about the most effective methods and techniques that could address these challenges we have identified, and to propose improved alternatives to addressing them.

1.3 Research Aim and Objectives

The primary goal of this research is to further investigate the *3D Shape search* problem explained in Section 1.1 (which basically involves the process of developing an appropriate 3D shape descriptor and matching method for efficient retrieval and classification of 3D objects), with some of the open challenges associated with 3D shape description, indexing, matching and retrieval, and be able to develop a new set of compact/concise, accurate, robust and computationally efficient 3D shape retrieval methods, with the aim of providing improvements (or alternatives) to existing methods, for efficient 3D CBSR framework, given a selection (i.e. *'benchmark'*) of rigid or non-rigid 3D triangular meshes and point clouds.

The most important aspect of any CBSR framework is the development of an efficient and accurate shape signature (descriptor), which is an abstraction of the shape used to semantically represent it. Shape descriptors are sometimes also referred to as feature-vectors (*fv*), and are needed for matching, retrieval, and classification of similar objects, given a query object. However, computing a shape descriptor typically follows a number of steps. The most crucial step is the extraction of features from the surface of the 3D shape beside the pre-processing steps, such as smoothing, normalization, etc. Efficient and accurate 3D descriptors (i.e. signatures) are those which are (i) *robust* against noise, clutter, occlusion, redundant parts, holes/defects, (ii)concise/compact, (iii) accurate, and (iv) efficient/easy to compute - see also the characteristics of appropriate shape descriptors in Section 2.3.1. Such signatures are also *invariant* under rigid, non-rigid, and affine transformation (rotation, translation, scaling). Intuitively, the accuracy (i.e. matching and retrieval results) of a 3D-CBRS is directly proportional to the accuracy of and robustness of the shape signature used, including the matching technique adopted. However, shape signature accuracy and robustness depend on the expressiveness of the extracted features from 3D surfaces being described. In view of these, this thesis considers the following research objectives:

- Development of concise/compact, accurate, robust, and efficient 3D shape representation (descriptor) to address the *3D Shape search* problem and facilitate 3D objects matching, retrieval and classification.
- Development of a 3D shape descriptor or retrieval method that is widely applicable, and generalisable across diverse and wider range of datasets and retrieval problems.
- Addressing (i.e. improving upon) some of the research gaps in existing research work, which include computational complexity, ambiguity, and accuracy of shape retrieval methods.
- Propose a retrieval method that is capable of effectively and mathematically represent 3D mesh and point cloud with the lowest number of vertices and points as possible, while still meeting the robustness and descriptiveness criteria of an effective 3D shape descriptor.
- Investigate theoretical aspects of practical importance in 3D shape representation, pre-processing, feature extraction, indexing, matching, retrieval, and classification.

- Finally, provide a simple and focused review of the knowledge-based 3D shape descriptors, its various categorisation and computational techniques, including highlighting the major difference between these kinds of descriptors and the data-driven shape descriptors, with the goal that the reader (i.e. future researchers) would be adequately informed to make a decision regarding the choice of algorithms for their available dataset and computer vision problems.

1.4 Research Questions

Considering our primary research goal and the need to tackle the *3D Shape search* problem defined in Section 1.3, other numerous concerns highlighted in Section 1.1 (Background), Section 1.2 (Motivation), and Section 1.3 (Research Aim and Objectives) also need to be addressed. This research, therefore, considers the following research questions:

i. Several existing 3D shape descriptors are known for their robustness (i.e. better performance) on specific retrieval or classification problem. What are the chances and computing cost of developing a robust, compact, and computationally-efficient 3D shape descriptor, that can generalise well (i.e. with incredible performance accuracy) across a diverse range of 3D datasets/retrieval challenges?

ii. Considering the overly popular deep learning (i.e. data-driven) approaches and the highly robust 3D shape retrieval methods they propose, for solving 2D/3D computer vision related problems, to what extent are the retrieval and classification performances of the traditionally hand-crafted (i.e. knowledge-based) methods relevant, compared to those obtained by deep learning methods?

iii. Given that there are many useful approaches to 3D shape descriptors under the knowledge-based category, which is the most applicable/appropriate approach for developing a concise, accurate, robust, and computationally efficient signature that is widely applicable?

iv. Given a 3D surface with high LoD (having incredibly large number of surface points), the intricate local surface, including the overall global structure and topology of the object would be better represented than a surface with low LoD, which, on the other hand, enhances storage and processing speed. However, what is the possibility of robustly and effectively describing a 3D surface using extremely low number of points, and still achieve high overall retrieval and classification performances?

v. Does the overall performance of a given shape descriptor depend on the choice of shape similarity metric used, and if true, how much bias does the metric have on the robustness or performance of a 3D shape descriptor?

1.5 Thesis Originality and Contributions

Essentially, this section highlights the originality of this thesis and our research work and presents a summarised overview of the contributions that we have made, considering our motivation (see Section 1.2) and primary goal (see Section 1.3) for this

research. However, the reader is referred to Section 6.2 for additional information regarding a review of our thesis contributions.

i We introduce a novel 6-dimensional local Augmented Point Pair Feature (APPF), which consists of a new 2-dimensional local feature and a 4-dimensional *local* point pair (Surflet-pair) feature [252] (see Section 4.2.1). The APPF encodes the physical characteristics (i.e. geometrical properties) of a local surface region, or Local Surface Patch (LSP), and is used to improve the overall performance of the surflet-pair feature, for a more accurate and robust representation/description of 3D meshes and point cloud surfaces. Unlike the latter, our 2-dimensional LSP feature, does not require surface normals, making its computation extremely fast, efficient, and straightforward. In addition, extra computational steps are not required for our 2-dimensional feature extraction from that of the surflet-pair feature extraction. Detailed description of this contribution (i.e. APPF) is presented in Section 4.2.1 and Figure 4.4.

ii The feature extraction sub-process for our methods rely on key points, rather than all the points from the 3D surface. We adopt the voxel-grid downsampling approach, instead of the conventional methods. After the voxel-grid down-sampling algorithm has been applied to each 3D point cloud object, we observe that for some model, the down-sampled points are not located directly on the surface of the point cloud (see Figure 3.19a). Using k-NN algorithm (where k=1) we develop a simple and quick method that searches each original point cloud surface for all the 1-closest points to the *down-sampled* points and return these as the actual down-sampled points. The outcome is visualised in Figure 3.19b, where all the down-sampled points are now properly located on the surface of the point cloud. This process is computationally very efficient considering that the k-NN algorithm is only searching for one point per iteration, and has contributed to the overall high performance of our final retrieval and classification method.

iii Following the APPF in (i), we propose and apply a novel statistically-based 3D shape representation called the APPFD (see Section 4.2.1), which involves bucketing each feature dimension of the locally extracted APPF into a multi-dimensional histogram, using up to 8 bins (i.e. $b_{(APPFD)} = 8$) in each feature dimension. The APPFD have been widely applied to 3D shape retrieval tasks (see Section 5.4), where it records some great overall retrieval performances, which in most cases, outperforms or performs just as well as several other state-of-the-art methods for the retrieval challenges or tasks.

iv We also introduce another novel statistically-based *global* shape descriptor called the HoGD, which is very intuitive, computationally very efficient, and easy to compute, thus capturing the global structure of 3D objects into a 1-dimensional histogram (see Section 4.2.2). The goal of proposing such descriptor is to combine it with the APPFD in order to complement the later and produce a resultant hybrid 3D retrieval method capable of describing both local and global properties of 3D objects, for overall improved performances.

v We propose and apply a novel hybrid 3D shape descriptor called the HAPPS (see Section 4.2.3), which combines the descriptive powers of both the *local*

APPFD and the *global* HoGD 3D descriptors to provide a more effective representation of 3D surfaces. An immensely popular real-world *3D Shape search* problem is the protein shape retrieval and classification challenges [157, 122, 32, 33, 123, 65]. We then applied the HAPPS method for the SHREC'18 dataset and SHREC'20 retrieval tasks. Experimental results demonstrated the superiority of the HAPPS-1 method over every other state-of-the-art methods for the SHREC'18 dataset [175]. In the SHREC'20 retrieval task, the HAPPS-1 and HAPPS-2 methods also demonstrated exceptional overall retrieval performances, outperforming several other state-of-the-art methods, as well as rival the overall best performing state-of-the-art method for that retrieval challenge [122]. In addition, the HAPPS method also performs optimally on several other benchmark datasets (see Section 5.3).

vi We develop a new highly efficient method for 3D shape representation, called APPFD-FK-GMM, where we harnessed the benefits of the Fisher Kernel (FK) and Gaussian Mixture Model (GMM) to agglomerate our locally computed APPFD into a more compact/concise, robust, and accurate single *global* descriptor. Experimental results with the APPFD-FK-GMM retrieval method is first contributed towards shape retrieval of 3D surfaces with similar geometric relief [231]. Although poor overall performance is recorded for this contribution, it did however outperform at least one other state-of-the-art method for that retrieval challenge. Subsequent experimental results (Section 5.5) of the APPFD-FK-GMM method on 3 other most recent benchmark datasets demonstrates its exceptional retrieval performances, where it outperforms several other state-of-the-art methods, including the HAPPS method.

vii To the best of our knowledge, our work is the first to consistently evaluate its shape retrieval methods on several standardised 3D shape retrieval benchmark datasets by SHREC (see Section 5.2.1), which is developed with the primary goal of testing the robustness and performances of different 3D shape retrieval algorithms. It is important to mention that each of the SHREC retrieval challenges pose distinct levels of retrieval challenges to shape retrieval algorithms. The respective challenges for each SHREC dataset is summarised in their associated sub-sections described under Section 5.2. This means that a retrieval method or algorithm which can maintain great evaluation performance across different datasets is proven to be exceptionally reliable and robust against several shape retrieval issues. We evaluated our retrieval method across several SHREC dataset and it recorded very high-performance evaluation. We therefore conclude that, with very minimal tweaks/improvements, the methods we propose in this thesis are suitable for several other computer vision tasks, such as detection, classification, etc. In summary, the 3D shape descriptors (retrieval methods) we have proposed in this thesis have been applied to address diverse range of 3D shape retrieval problems and task, thus contributing the most to several global SHREC retrieval challenges and tasks for 3D objects (see Chapter 4). Through series of thorough experimental evaluation and analyses, the retrieval performances of our methods have been compared against a substantial number of other state-of-the-art 3D shape retrieval methods, for each of the datasets we tested our methods against.

viii In Section 2.3 through to Section 2.6, we provide some useful and updated review of 3D shape descriptors, highlighting the major differences between data-driven and knowledge-based 3D shape descriptors, their advantages and disadvantages in computer vision applications, and the choice of the knowledge-based over the data-driven approach as in this thesis. The aim of this review is to provide a clear roadmap for new researchers in this field to make quick decision regarding retrieval methods to adopt for their retrieval experiment.

1.6 Research Relevance and Application Areas

Presently, the ability to retrieve existing 3D objects from a 3D object database helps to facilitate tasks for professionals in several application contexts, allowing them to quickly obtain desired shapes without spending much time re-modeling existing ones. In addition, several content-based 3D shape retrieval techniques and algorithms are widely applicable to other computer vision tasks as well as in computer graphics, digital geometry processing, and pattern recognition communities. For example, extracted 3D features or shape descriptor computed for 3D shape retrieval task can as well be employed for 3D shape classification, 3D object recognition (i.e. 3D-face recognition), 3D object detection, or 3D registration and alignment tasks.

Generally, the application contexts for which the techniques and methods implemented in this research are relevant include:

- **Bioinformatics (Biology):** Proteins can be described as non-rigid surfaces representing their solvent-excluded surface [157]. Since 3D structures of proteins are better preserved than their sequences, it is inadequate to compare amino acid sequences of the proteins by sequence comparison alone, because it would be impossible to detect the similarities between any two homologous proteins by only comparing their sequences, therefore detection of partial dis(similarities) between related multi-domains protein surfaces is of main importance in drug discovery pipelines, adverse drug event prediction and in the characterization of molecular processes and diseases.

- **Entertainment, Games and Augmented Reality:** There has been rapid development and interest in 3D games lately. The video game industry uses 3D models as assets for computer and video games. The movie industry uses 3D models as characters and objects for animated and real-life motion pictures. Augmented Reality (AR) overlays digital content and information onto the physical world – as if they exist in the physical world. In AR, 3D models are integrated in real time, with actual or scaled size in a particular surrounding using the camera. Similarity search for existing 3D models databases enable reuse and adaptation of 3D models for game development, movie, and entertainment industry in general, thus reducing production costs, while speeding up its production delivery.

- **3D Digital Catalogue and Cultural Heritage:** It is now possible to take a virtual tour of several museums in the world and get a 3D view of thousands of artefacts displayed at these museums. For example, the Isabella Stewart Gardner Museum [165] has a virtual 3D tour called "Thirteen Works: Explore

the Gardner's Stolen Art". The tour allows for virtual tour of the museum while learning about the thirteen pieces that were stolen in the early 1990s. MyMiniFactory [167] platform, for example, also has the largest collection of 3D scanned statues and artifacts from around the world. The digitized scans of these 3D models are freely available and widely used by educators, artists, and the 3D printing community. Many other digital catalogues exist also, such as Command module Columbia (CM-107) [9], etc. However, effective 3D searching and retrieval techniques, and methods are therefore required to be able to sieve through these large collections of 3D models.

- **3D Medical Image Analysis:** Detailed 3D models of glands and organs, which may be created with multiple 2D image slices from Magnetic Resonance Imaging (MRI) [248] or Computed Tomography (CT) [30] scan, are now widely used in the medical industry for disease detection, prevention, and control. Furthermore, 3D volume data are often generated in medical imaging applications using MRI scans, and a possible application lies in automatic diagnosis support by analysis of organ deformations, by matching actual images with medical database of known deformations [28].

- **Engineering and Architecture:** In architecture, it has now become more convenient to demonstrate proposed buildings and landscapes using 3D models, instead of traditional, physical architectural models. In computer-aided design (CAD), manufacturing (Computer-aided Manufacturing (CAM)), and engineering (CAE), 3D models are used as designs of new components, such as mechanical parts, automobile, and other structures from original vehicle parts. Consequently, Architects, Technicians and Engineers in architecture industry and manufacturing companies need to be able to exploit the large CAD, CAM and CAE model databases during their design and manufacturing processes. It is possible with human intervention to inspect contents of these databases manually and physically, but the processes are awfully expensive and time consuming. Development of retrieval methods and search algorithms to automate some of the process is therefore important, in order to improve productivity.

Other areas of application include: Agriculture, Security, Industrial inspection [183], Autonomous Driving and Robot Navigation (i.e. loop closure detection [80] among several others).

1.7 Research Outcome

In this thesis, we develop and apply new methods to address the *3D Shape search* problem by separately considering the challenges of descriptor robustness versus compactness, and ease of computation. Our methods are capable of accurately representing 3D surfaces in a concise (compact) and descriptive (robust) manner, by using comparatively incredibly small number of points sample (between $3,500$ to $4,500$ points), including a non-complicated shape matching technique and distance metrics. In addition, results of extensive experimental evaluation demonstrates that our methods generalise very well across several different *'benchmark'* datasets and domains. Specifically, the contributions of this thesis to the area of 3D shape matching, retrieval and classification are manifold as already outlined in Section 1.5.

1.7.1 Overview of Methods and Results

To avoid unnecessary repetition, we refer the reader to Section 6.3, where we present a detailed review of the retrieval results (i.e. performances accuracies) for each of the 3D shape retrieval method we propose in this thesis.

1.8 Organization of Thesis

This thesis presents our entire research work, which is organised into six chapters. This chapter already provides a general introduction regarding the work involved in this research with adequate background and overview of our research, including aim, objectives, as well as a summary of our research contributions. This chapter also explain real life scenarios where our research outcome could be applied, which forms part of our research motivation. The remaining parts of this thesis is organised as follows:

Chapter 2: In this chapter, we begin by providing definition to some of the terms, techniques and functionalities which are most relevant to this thesis and research work. Next, we give an insight into a few of the available formats that 3D objects are presented including types of 3D objects representation, followed by classifications of 3D objects. This chapter also provides detailed review of 3D shape descriptors (i.e. retrieval methods), including the types, classification, their advantages, and disadvantages. Two broad approaches to 3D shape descriptor computation: the data-driven approach and the knowledge-based approach are discussed also, and we reveal how the knowledge-based approach plays a significant role to the success of the now extremely popular data-driven approach, which largely depends on hand-crafted low-level features and to some extent, experts' knowledge. The characteristics of appropriate shape descriptors are also highlighted here. Finally, we conclude this chapter by providing a concise review of statistically-based 3D shape descriptors, under the knowledge-based category, which is the basis for the methods we propose in this thesis. However, we also discuss some of the challenges that different dataset presents to 3D shape retrieval algorithms and insights into how defective data can be handled.

Chapter 3: We dedicate this chapter to most of the implementation techniques needed for our proposed 3D shape retrieval and classification methods. Here, we provide in-depth description for our research strategy, where essential techniques and issues, such as data pre-processing, point cloud sampling, feature extraction, 3D key points detection/determination, as well as shape descriptor construction, indexing, matching and the different types of (dis)similarity metrics implemented in our work, are addressed. We adopt and implement about five different (dis)similarity metrics for matching of our proposed 3D shape descriptors - overview of these metrics are provided in this chapter.

Chapter 4: This chapter essentially focuses on our research methodology, which involves the major contributions of the work propose in this thesis. Our proposed methods for describing 3D objects (meshes and point clouds) are presented, which include all the three broad classifications of shape descriptors (global, local, and

hybrid) described in Section 2.4. We provide detailed background to each of the 3D shape descriptors (i.e. retrieval methods) we propose in this thesis. Performance evaluation is an essential part of any IR system or algorithm, which allows us (i.e. researchers) to assess how well a particular algorithm or retrieval method performs on a particular dataset and retrieval problem. In this chapter, we provide adequate information regarding the evaluation techniques we have adopted in this thesis, where we describe the tools, terminologies and processes involved, as well as the different performance evaluation metrics used to measure the overall performance of different methods.

Chapter 5: This is another especially important chapter of this thesis, where we present experimental evaluation results and discussions, regarding all our proposed methods described in the previous chapter. Datasets are needed to evaluate the performances of these retrieval methods. This chapter first presents and describes up to ten different benchmark datasets used in this thesis to evaluate the performances of our methods, and highlight a few principal issues regarding these datasets, including the retrieval challenges that each of these datasets present to shape retrieval algorithms. The evaluation strategy we use is by applying each of the methods we propose on at least two different datasets. For a given retrieval method and a dataset, we run series of experiments using different parameters and/or distance metrics, where the qualitative and quantitative results of these different parameter settings are presented and analysed. We then compare the retrieval performances of this method to the performances of several other state-of-the-art methods for that dataset and further provide comparative analyses. These processes are repeated for all other datasets and methods described in this thesis. We conclude the chapter with a review/summary of all the experimental evaluations performed in the various sub-sections of the chapter.

Chapter 6: This chapter provides a summary to the entire work presented in this thesis. We begin the chapter by providing an overall review of the entire thesis, followed by a concise review of the contributions we have made, which include contributions from each of the 3D shape retrieval methods we propose in this work. Next, we provide the key summary of the retrieval performances of each of our proposed methods. We then conclude this chapter by providing some of the findings of this research and points the reader to potential future research directions.

Chapter 2

LITERATURE REVIEW

2.1 Introduction

This chapter begins by introducing the concepts, techniques, and processes which are most relevant to this thesis and our research work, such as 3D object representations, 3D-CBRS, 3D shape search engine, 3D shape descriptors, distance/(dis)similarity metrics, etc. 3D objects are available in many different representations and file formats, each having different data structure which constitutes various challenges to shape retrieval algorithms. This chapter provides overview of 3D object representations, including detailed review of the two most popular ones: 3D triangular meshes and point cloud representations, including their respective suitability for certain CV and PR tasks. Their advantages and disadvantages are also considered in the review, including the reasons why we fully adopt these representations for our experimental evaluations.

Two broad classifications of 3D objects are presented, which are: rigid and non-rigid 3D models, including the two major deformations or defects which are present in any of these classes of 3D objects (i.e. water-tight 3D objects and non-water-tight 3D objects). We discuss the effects of each of these classes and defects on shape retrieval algorithms because they play a key role in feature extraction and shape descriptor performances of 3D objects. For example, the algorithm developed to process rigid, water-tight 3D object would certainly fail to produce the same results when applied to non-rigid, non-water-tight 3D objects, etc. In addition, we provide a concise and detailed review of 3D shape descriptors (retrieval methods) in the literature of the past two to three decades, including an overview of each of the three broad classifications of these descriptors (global, local and hybrid), their advantages, and disadvantages. Alternatively, two broad perspectives (approaches) regarding 3D shape descriptors computation are thoroughly examined: (i) Data-driven approach and (ii) Knowledge-based approach, to 3D shape descriptors or retrieval methods. However, our research work completely adopts the knowledge-based approach, and we provide a more detailed review of this approach, including justification for adopting this approach.

Further in-depth analyses of several other sub-categories of 3D shape descriptors within the knowledge-based approach, including the statically-based 3D shape descriptors (which is the basis for our thesis and research work) are provided in

this section. We reveal how the knowledge-based approach plays a significant role to the success of the now extremely popular data-driven approach, which largely depends on hand-crafted low-level features and to some extent, experts' knowledge. Finally, we conclude this chapter with an overview of 3D retrieval challenges (i.e. highlights of how various abnormalities in datasets, such as holes, degeneracies, and duplicates in vertices, faces, and edges, etc.) affect shape retrieval algorithms due to certain factors: dataset representations, object's benchmarks design, data sources or capturing devices used to acquire the 3D models.

2.2 Definition of Terms

There are core terminologies, techniques, naming conventions and processes, etc., which are most relevant to this thesis and our research work. In this section, we would try our best to provide detailed introduction, definitions, and description of these key aspects.

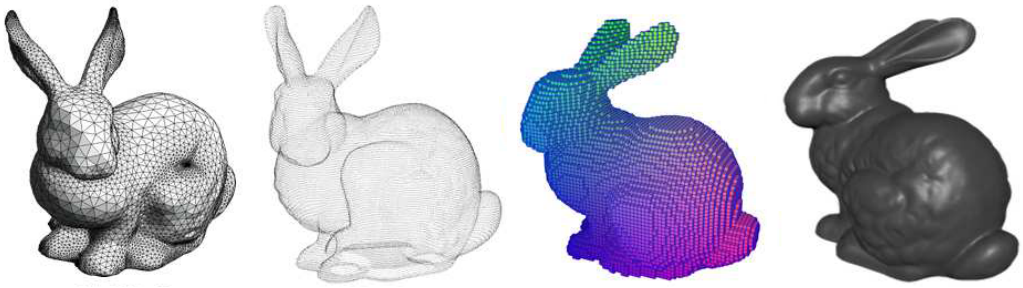
2.2.1 3D Object Representations

It is particularly important to consider objects representation when developing solutions for 3D shape retrieval and other CV applications involving three dimensional objects. This is because the design of shape analysis algorithms strictly depends on the input data-type or representation, whereby the algorithm designed to process 3D triangular meshes, for example, would fail to work for another type of representation, such as voxel representation of the same 3D object. Besides, 3D objects can be represented in several different ways, and finding an appropriate representation for shape that is amenable to its surface matching is still an open research issue in computer vision [95]. In this section, we provide an overview of two most common types of surface representations for 3D objects.

Unlike 2D images which have a rather unique representation of (n x $n = n^2$) 2D grid of picture elements (i.e. pixels) containing grey scale or colour values, many different representations exist for 3D shapes making them more difficult to process. It is therefore important to discuss the various forms by which 3D shapes are commonly represented in CG; highlight the advantages and disadvantages of each form of 3D representation for different CV/CG applications; and discuss the preferred choice of 3D representation for this research study.

3D objects are basically represented as polygonal meshes (triangular meshes) or point clouds. Variety of other possible representations exist, such as parametrized surface patches (Non-uniform Rational B-spline (NURBS) surfaces [89, 194]), and Constructive Solid Geometry (CSG) (voxel-grids [262]), etc. However, polygon surfaces [259, 222, 233, 12] are the most common surface representation. Figure 2.1 depicts the 3D model of the famous Stanford bunny in four possible representations: mesh, point cloud, voxels and NURBS. Many other representations exist [247], but we mention only these four, for brevity. Because certain CV applications require specific kind of 3D format, it is possible to convert between these forms of object representations with the help of freely available software tools, such as Trimesh [47],

Meshio [207], Open3D [279], Visualization ToolKit (VTK) [108] libraries, etc. However, our thesis implementation relies only on the polygon mesh and point cloud object representation. We provide further details regarding 3D mesh, point cloud and voxel representations in the next sub-sections. Since NURBS surfaces are parameterised, outside the scope of our research implementations, and not commonly used, we do not include this in further discussions. However, the reader is referred to [89, 194], for more details regarding the NURBS surface representation.



(a) Mesh (b) Point Cloud (c) Voxel (d) NURBS

Figure 2.1: Four possible surface representations of the Stanford bunny 3D model. (2.1a): 3D triangular mesh, consisting of *vertices* and *edges* (connectivity). (2.1b): 3D point cloud, consisting of unstructured points in $[x, y, z]$ Euclidean space. (2.1c): 3D voxels grids, which is made up of volumetric pixels (cubes). (2.1d): Non-uniform Rational B-spline (NURBS) 3D surface, which is exceptionally smooth.

3D Mesh Representation

Surface meshes are the general form of object representations because, given a sufficient number of vertices, they can represent almost any object [97]. 3D meshes are built up from simple primitives like *points*, *lines* and *planes* (i.e. faces). Representing 3D objects with a mesh is possible using either of the following functions:

- Implicit functions: $f(x, y, z) = 0$,
- Parametric functions: $(x(u, v), y(u, v), z(u, v))$, or
- Simple mathematical functions: $z = f(x, y)$

For any given 3D mesh, a variety of file formats, such as *.obj*, *.off*, *.std*, *.ply*, *.stl*, *.vtk* and *.wrl* etc., exist for storing polygon mesh's data structure, comprising of vertices, faces, and edges (connectivity) information [259]. Other properties of the mesh, like the unit normal normals, colours, and texture-coordinates for each vertex can also be part of the data. The vertices property in a 3D mesh file includes coordinates of points in 3-dimensional space i.e. $[v_x, v_y, v_z]$ coordinates. The faces property represents triplets of vertices that makes up each face of a triangular mesh, and quadruplets of vertices for a quadrilateral mesh (quad mesh). However, triangular mesh is the most suitable mesh representation for most computer vision and graphics applications. The edge property is used to store connectivity that exist between any two vertex coordinates that makes up the mesh's surface. The vertices, faces, and edges properties of a given 3D mesh can be dubbed as features,

but the vertex coordinate is the most informative because it contains information on topology and overall shape structure, and is the basis for which all other properties/features, like vertex normals, face normals, faces, edges, texture coordinates, are derived.

Most 3D shape descriptors like Spin Images [96], Surflets-Pair relation Histogram (SPRH) [252], etc. are designed to work with triangular meshes. Point cloud data, on the other hand, mostly always contain only the $[p_x, p_y, p_z]$ coordinates of points that forms the topology and structure of the 3D shape they represent. Besides the points coordinates, a typical point cloud file, like the .pcd [141], may contain other properties like points normals $[n_x, n_y, n_z]$, RGB colours, light intensity details, moment invariants $(j1, j2, j3)$ and viewpoint information, etc.

3D Point Cloud Representation

Point clouds are the raw output of many 3D scanning devices and sensors, such as Microsoft Kinect [258], LiDAR [257]. They could also be output from camera and photogrammetry software [208, 209]. Beside these three main sources, point clouds can also be generated from triangular meshes using a process called mesh sampling, thus point clouds and polygonal (triangular) meshes are the two main representations for 3D data, and they are closely interlinked. For example, 3D objects are represented using polygonal meshes with vertices and faces, but the vertices alone could be interpreted as a point cloud and represents the underlying surface geometry. Similarly, the points in a point cloud are likened to the vertices of a mesh, through triangulation and surface extraction [14], and represent the underlying surface of the 3D object.

3D Voxel Representation

Voxel (i.e. volumetric pixels) is a term used to represent a volume in 3D space, similar to the word, *pixel*, which represents a picture element in 2D space. The following explanation clarifies the concept of voxels. Basically, 2D and 3D computer graphics are represented as *vector* or *raster* graphics. Considering 2D image, for example, while mathematical equations are used to describe a 2D image represented as vector graphics, array of color values are used to describe raster graphics, instead. In 2D vector graphics, a vector with two components, $p = [x, y]$ is used to described each point of a line or a polygon, while three components vector, $p = [x, y, z]$ is used in the case of 3D vector graphics. Similar to 2D raster graphics, where 2D images are divided into a number of evenly sized rows and columns, the volume in a 3D raster graphics is divided into evenly spaced rows and columns in three different directions (up-down, in-out, and left-right) which divides the 3D space into cubes known as *voxels* (i.e. volume elements), each of which has a 3D coordinate within the volume (see Figure 2.1c). Voxels are commonly used 3D representation for the output data from an Ultrasound, CT Scan, and MRI, etc. However, one of the disadvantages of this kind of 3D object representation is that they contain a great deal more data and exponentially larger than 3D polygonal surfaces, which adds up to a lot of memory. According to [260], a direct consequence of this difference is that polygons can efficiently represent simple 3D structures with lots of empty or

homogeneously filled space, while voxels are good at representing regularly sampled spaces that are non-homogeneously filled.

Why Mesh and Point Cloud Representations

3D modelling software, such as Blender, Autodesk, SketchUp, FreeCAD, QGIS, etc., are typically used for generating triangular meshes. In addition, triangular meshes can as well be generated from a point cloud via surface reconstruction process described in details in [14]. Basically, it is quite difficult to visually distinguish triangular mesh *vertices only* from the point cloud, identification of neighboring points is challenging and time consuming, visualization, extraction of surface and estimation of its properties such as curvature is difficult, etc. However, while triangular meshes are quite convenient for many computer graphic tasks, point clouds are ideal for processing and extracting information from 3D objects. In line with this, our implementations consider the extra steps of generating point clouds from triangular meshes.

A few CV tasks, such as 3D shape registration [16] can succeed by utilising only raw point cloud or mesh vertices as input to the Iterative Closest Point (ICP) algorithm, various other tasks, such as shape retrieval relies upon a more compact and descriptive form of input (signature or descriptors) to its matching algorithm, rather than utilizing the 3D objects (raw meshes or point clouds) themselves. In this thesis, we only depend upon two properties of a given 3D shape: (i) the $[x, y, z]$ coordinate of mesh vertices, $v_{(x,y,z)}$ or point cloud points, $p_{(x,y,z)}$ and (ii) their associated normal vectors or surface normals, $n_{(x,y,z)}$. Although, for mesh input, we rely upon the vertices and faces properties, from which a point cloud representation of the mesh would be generated, including estimation of their corresponding associated normals. Therefore, the input to our retrieval algorithm is primarily $[x, y, z]$ coordinates of points (point cloud) and their associated normals. Basically, if the input 3D objects are raw point cloud data, their normals property are not usually given by default. We however derive these using the techniques in Section 3.3.2 (see Figures 3.9, 3.10 and 3.11), depending on the representation of the input shape. For instance, the feature extraction technique for our proposed APPFD (see Section 4.2.1) strictly relies on 3D coordinates of points on the surface, and their corresponding normal vectors. i.e. $\{P_s, N_s\}$, where $\{P_s \subset p_i : (p_x, p_y, p_z)_i, i = 1...N\}$, and $\{N_s \subset n_i : (n_x, n_y, n_z)_i, i = 1...N\}$. N is the total number of points in the point cloud, or vertices in the triangular mesh. Details of technique(s) for generating (sampling) point clouds from meshes are provided in Sections 3.2.3 and 3.2.3.

2.2.2 Classification of 3D Objects

In the previous section, we discussed four common data structures used to represent 3D objects - mesh, point cloud, voxel, and surface representations. Naturally, every 3D object appears in a rigid or a non-rigid form. Similarly, considering any of the above-mentioned representations of 3D objects, two aspects are important, which are: (i) rigidity and (ii) fluidity. This means that 3D objects can either be rigid or non-rigid. It is important to consider this classification for 3D objects when developing shape retrieval methods that would describe them, because performances of 3D shape descriptors (retrieval methods) are different for each of these classes of

dataset. For example, the Global D2 and A3 shape distribution histogram descriptors by [174] are reported to perform better on rigid 3D models, yielding excellent retrieval results but not on non-rigid models.

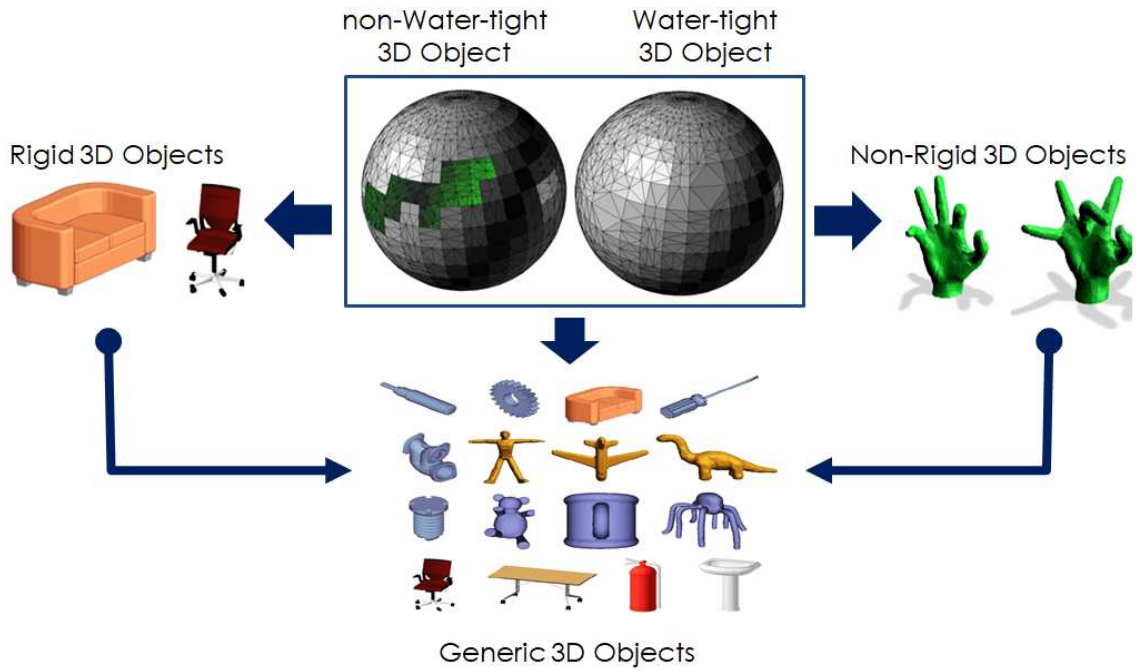


Figure 2.2: Two classes of 3D objects: Rigid and Non-Rigid, showing defective (non-water-tight) and non-defective (water-tight) surface meshes. The Rigid and Non-Rigid objects can be combined in a dataset to form the Generic 3D objects dataset as shown.

Our preliminary studies and experimental findings reveal that shape descriptors which perform well with rigid objects did not do so with non-rigid objects and vice versa. The experimental evaluations of the shape distribution histogram descriptors [174] also confirms this. 3D shape retrieval tasks are therefore considered in two broad perspectives: (i) the rigid CAD model retrieval, and (ii) the non-rigid shape retrieval, The publications in [138, 227, 23] and [241] survey most of the work in non-rigid 3D shape matching and retrieval, while the publications in [166, 145, 217, 274] and [37] reveal research efforts on rigid 3D CAD/CAM shapes retrieval. In addition, the third broad categorisation also exists, which is the generic 3D models retrieval. This involves the concept of developing 3D shape descriptors and matching methods that are suitable for both rigid and non-rigid 3D models aims to solve 3D retrieval challenges for the generic class of 3D models for either or both. The SHREC 2012 [131] and SHREC 2014 [130] tracks were developed with this in mind, for instance. These broad categorisations of shape retrieval problems allow researchers the ability to evaluate how best their retrieval method performs on the different broad classes of 3D models (rigid, non-rigid, and generic datasets), which is very important for practical implementations in real-life scenarios where 3D objects contain a wide combination of both rigid and non-rigid 3D objects.

As illustrated in Figure 2.2, rigid and non-rigid 3D objects represented particu-

larly as triangular meshes can be water-tight or non-water-tight. Water-tight meshes are known to have minimal or no issues during shape analysis, data processing, and feature extraction, etc. On the other hand non-water-tight objects pose lots of challenges to shape analysis algorithm and adversely affects the future extraction and shape descriptor computation process. Non-water-tight 3D object simply includes an object with missing surface parts (i.e. with small or large holes as shown in Figure 2.2), which does not enclose a volume, making it difficult for retrieval methods to determine the complete structure or topology of the object. The larger the missing parts the more difficult it gets to accurately describe such an object. In most cases, all database 3D objects would first need to be manually repaired to make them water-tight before subsequent feature extraction and shape descriptor algorithms are applied to them. If that happens, then real-time application is impossible to achieve with these kind of defective objects and very few known methods exist which are capable of dealing successfully with non-water-tight 3D objects, such as all the methods discussed in [142] and one of our research contributions in [175], which is also suitable for real-time application, because the HAPPS method does not need to manually repair the objects prior to feature extraction and subsequent retrieval functions.

We stress here that the techniques we have implemented in this thesis are not restricted to water-tight 3D objects alone. Since our retrieval algorithm primarily processes point clouds by default, and given a polygonal mesh as input, we first convert it to point set representation. We also do not introduce a constraint that the input mesh needs to be water-tight or orientable, in order to extract any of our proposed descriptors. Secondly, among the two-broad classification of 3D objects or datasets (i.e. rigid and non-rigid), and the third class (i.e. generic), discussed in this section, we have been able to evaluate our retrieval algorithms mostly with the non-rigid and generic datasets only, due to the wide availability and ease of access to these datasets and their respective ground truths, unlike the rigid 3D objects datasets, which is not common and not widely available in the public domain. However, we could have scanned a sizable number of rigid 3D objects to develop our own 3D CAD models dataset(s) and ground truth file(s) in order to enable us independently test the performance of our retrieval algorithms *strictly* on the rigid 3D datasets and analyse these methods for their suitability in non-rigid objects retrieval problems. We chose not to go this route due to the scope of this thesis and for the following reasons: (i) recent research directions in 3D shape retrieval are mainly focused on the generic 3D objects and non-rigid 3D objects datasets, (ii) the generic dataset already contains reasonable number of rigid 3D objects, and (iii) the ground truth for the rigid 3D dataset or benchmark we create would need to be verified with a number of other state-of-the-art retrieval methods in comparison with the results returned by our retrieval methods before an unbiased conclusion can be drawn to ascertain the performance of our method on the rigid-dataset we would have created. This, we fear might take an exceedingly long time and further broaden the scope of this thesis. We instead hope to take this route as part of our future research direction on this topic.

2.2.3 Distance or (Dis)similarity Metrics

Distance metrics are measures used to quantitatively define the (dis)similarity between any two shapes. They simply compare how much alike two different data objects are. Defining two objects' similarity largely depends on the descriptor and/or signature used to represent the objects (3D in our case). There are many different distance or (dis)similarity metrics in literature and so far, there is no one metric that suits all shape matching and retrieval cases. For example, the $1D$ Earth Mover's Distance metric is only expected to return good matching results for 1D histograms, such as shape distributions (i.e. A3, D1, and D2 descriptors etc.) [173], whereas it would perform poorly on other types of signatures like our proposed HAPPS descriptor as demonstrated by our experimental results (see Table 5.12, Figure 5.20, Section 5.3.3), for example. When two objects are matched using a similarity metric, a floating-point value (within the range of 0.0 and 1.0, i.e. $[0, 1]$) is returned. This value is known as the "similarity score", where 0 is the lowest and 1 the highest score. Two objects are said to have a high degree of dissimilarity if the distance between their signatures (i.e. similarity score) is small, whereas a high similarity score will translate to a low degree of similarity between the objects. Generally, where d is a distance metric and two 3D objects are given by their respective feature-vector (fv), as fv_1 and fv_2, then:

- $Similarity(fv_1,\ fv_2) = 1$, if $d(fv_1, fv_2) = 0$
- $Similarity(fv_1,\ fv_2) = 0$, if $d(fv_1, fv_2) = \infty$

In summary, different metrics perform differently for comparing different shape signatures. We demonstrate this in our results presented in Figure 5.20, Section 5.3.3. Since a similarity measure must be selected to determine how close one signature is to another, in our experimental evaluations therefore, we applied five different metrics in order to test and finally select the metric which returns better overall matching results for our respective final shape signatures - represented as feature vectors (fv_s). The problem of shape matching is therefore converted to computing the distances between two d-dimensional fv_s: $h_Q,\ h_D \in R^d$, where h_Q and h_D are feature vectors of the *query* and *database* shapes, respectively. Details of these distance metrics are presented as follows.

Euclidean Distance

This is the most used distance metric, which can provide the best proximity measure for dense or continuous data. Generally, the Euclidean distance between two points a and b is the length of the path connecting them, and a generalized term for the Euclidean norm is the L_2 norm or L_2 distance [255]. Mathematically, the Euclidean distance between $h_Q,\ h_D \in R^d$ is computed according to Equation 2.1, where the subscript, i is used to denote the i'th component of the shape descriptor.

$$\delta_{L_2}(h_Q, h_D) = \|h_Q - h_D\|_2 = \sqrt{\sum_{i=1}^{d}(h_{Q_i} - h_{Di})^2} \tag{2.1}$$

Cosine Distance

Unlike other distance metric which measures the straight-line distance between data objects, the Cosine similarity metric measures the *angle* between two objects, or their L_2-normalized dot product. Determining cosine similarity between two objects is the same as finding the cosine of the angle between the objects. Considering that $Cos(0°) = 1.0$, $Cos(90°) = 0.0$, $Cos(180°) = -1.0$, and $Cos(360°) = 1.0$, then the cosine similarity score between two objects would certainly be less than 1.0 for any other angle (between these objects) greater than 0°. The Cosine similarity metric is therefore a measure of orientation rather than magnitude. For example, two vectors with the same orientation would have a cosine similarity score of 1.0, whereas two vectors at an angle of 90° to each other would have a cosine similarity score of 0.0. Similarly, two vectors diametrically opposed to each other would have a cosine similarity score of −1.0, irrespective of their magnitudes. Mathematically, the Cosine distance between two feature vectors: h_Q, $h_D \in R^d$ is given by Equation 2.2.

$$\cos(\mathbf{h_Q}, \mathbf{h_D}) = \frac{\mathbf{h_Q} \cdot \mathbf{h_D}}{\|\mathbf{h_Q}\| * \|\mathbf{h_D}\|} = \frac{\sum_{i=1}^{d} \mathbf{h_Q}_i \mathbf{h_D}_i}{(\sqrt{\sum_{i=1}^{d} (\mathbf{h_Q}_i)^2}) * (\sqrt{\sum_{i=1}^{d} (\mathbf{h_D}_i)^2})} \tag{2.2}$$

Earth Movers' Distance (EMD)

The EMD is a metric used to evaluate (dis)similarity between two multi-dimensional distributions in some feature space where a distance measure between single features (ground distance) is given. Mathematically, EMD is also known as Wasserstein metric [261]. Intuitively, given two distributions, one can be seen as a mass of earth properly spread in space, the other as a collection of holes in that same space. Then, the EMD measures the least amount of work needed to fill the holes with earth [43]. We decided to implement this distance metric in our work owing to its numerous successes in 2D image processing [196]. More details regarding this metric can be found in [196, 197].

The Wasserstein metric can be used to compute the distance between two 1D distributions (i.e. shape descriptors in our case). Given two descriptors, say h_Q and h_D, the Wasserstein distance is considered as the minimum amount of *work* required to transform h_Q into h_D, where *work* is measured as the amount of distribution weight that must be moved, multiplied by the distance it has to be moved. If we consider our shape signatures as a Cumulative Distribution Function (CDF), then for a single signature, h_Q with d-dimension, its CDF, h_Q^C is defines by Equation 2.3.

$$h_Q^C[d] = \sum_{i=0}^{d} h_Q^C[i] \tag{2.3}$$

We therefore compute the EMD between our signatures, h_Q and h_D with d-dimensions according to Equation 2.4.

$$EMD(h_Q, h_D) = \sum_{i=1}^{d} |h_Q^C[i] - {h_D}^C[i]| \tag{2.4}$$

.

Kullback-Leibler Divergence (KLD)

In mathematical statistics, the Kullback–Leibler divergence (also called relative entropy) is a measure of how one probability distribution is different from a second, reference probability distribution [114, 113, 184]. Generally, when we consider two distributions of probability P and Q. Usually, P represents the data, the observations, or a probability distribution precisely measured. Distribution Q represents instead a theory, a model, a description, or an approximation of P. The Kullback–Leibler divergence is then interpreted as the average difference of the number of bits required for encoding samples of P using a code optimized for Q rather than one optimized for P [256]. We imagine our final 3D shape signatures: h_Q and h_D (for which we want to compare) to be two probability distributions, and assume that the KLD similarity score between them would be a measure of how either h_Q or h_D diverges from the other. Intuitively, if the KLD divergence is low, then the two objects represented by h_Q and h_D are similar to the degree of the KLD similarity score, else, they are not similar. Hence, for discrete probability distributions h_Q and h_D having the same dimension, d (i.e. defined on the same probability space, d), the KLD from Q to P is defined by Equation 2.5.

$$\begin{aligned} D_{KL}(h_Q \| h_D) &= \sum_{i \in d} h_Q(i) \log \left(\frac{h_Q(i)}{h_D(i)} \right) \\ &= -\sum_{i \in d} h_Q(i) \log \left(\frac{h_D(i)}{h_Q(i)} \right) \end{aligned} \tag{2.5}$$

Since $\log(\frac{a}{b}) = \log(a) - \log(b)$, Equation 2.5 can then be re-written (i.e. re-implemented) as shown in Equation 2.6

$$D_{KL}(h_Q \| h_D) = \sum_{i \in d} h_Q(i) \log(h_Q(i)) - \sum_{i \in d} h_Q(i) \log(h_D(i)) \tag{2.6}$$

Practically, is asymmetric and we found that $D_{KL}(h_Q \| h_D) \neq D_{KL}(h_D \| h_Q)$. It is therefore important to state that the results we obtained with the KLD similarity metric (see Section 4) was with $D_{KL}(h_Q \| h_D)$, and not $D_{KL}(h_D \| h_Q)$.

Jensen-Shannon Divergence (JSD): Alternatively, another divergence metric, called the Jensen-Shannon Divergence (JSD) [153, 63], exist that can also quantify the statistical difference (distance or similarity) between two probability distributions. The JSD metric is an extension of the KLD, where it uses the later to compute a normalised score, between 0 (identical) and 1 (maximally different), for base-2 log. Such score is smooth and symmetrical (i.e. $D_{JS}(P||Q) == D_{JS}(Q||P)$) as opposed to that of the KLD metric. Essentially, the JSD can be calculated based on Equation 2.7.

$$D_{JS}(P||Q) = 0.5 * D_{KL}(P||M) + 0.5 * D_{KL}(Q||M) \tag{2.7}$$

Where M is given as: $M = 0.5 * (P + Q)$.

Considering these desirable characteristics of the JSD the choice of the KLD metric, instead, for our 3D shape matching tasks is mainly due to its popularity and success. The KLD has been very widely used in machine learning, statistics, signal processing, and content-based shape matching/retrieval. Secondly, the JSD is significantly more computationally challenging to implement, not very suitable for real data, and behaves well only when both distributions (i.e. $P(x)$ and $Q(x)$) being matched are small. The KLD metric, on the other hand, is easier to compute and performs well on large distributions, such as the final feature-vectors of our APPFD method which is very high-dimensional (i.e. up to $200k$ dimensions), whereas the JSD failed to match such distributions.

Squared Euclidean Distance (SED)

The SED is simply the square of the standard Euclidean distance, and the SED between two 1-D arrays (i.e. shape descriptors), h_Q and h_D are given by Equation (2.8), which can be simplified to be:

$$\delta^2_{L_2}(h_Q, h_D) = (h_{Q_1} - h_{D1})^2 + (h_{Q_2} - h_{D2})^2 + (h_{Q_3} - h_{D3})^2 + \cdots + (h_{Q_d} - h_{Dd})^2$$

where h_Q and h_D has dimension, d.

$$\delta^2_{L_2}(h_Q, h_D) = \|h_Q - h_D\|^2_2 \tag{2.8}$$

According to [255], the Squared Euclidean distance is not a metric, as it does not satisfy the triangle inequality [105], nor does removing the q parameter and its associated terms (see Equation (2.9)) render the SED function into a norm or semi norm for the same reason. However, it is a more general notion of distance, namely a divergence (specifically a Bregman divergence [22]), and can be used as a statistical distance. The Pythagorean theorem is simpler in terms of squared distance (since there is no square root); if $pq \perp qr$; then:

$$\delta^2_{L_2}(p, r) = \delta^2_{L_2}(p, q) + \delta^2_{L_2}(q, r) \tag{2.9}$$

However, in computer science (and especially computer graphics, simulations, and video game development), the usage of traditional Euclidean distance and norm functions (in certain contexts) can be sometimes considered non-optimal due to its dependence on the square root operation, which in many cases can be prohibitively slow. Algorithms based on comparisons between multiple distances or magnitudes can forgo the Euclidean metric and can instead utilize SED as an optimization, as relations between arbitrary non-negative values (in our case, distances) in a tuple should remain preserved after all values become squared (the SED values) [255].

In Section 3.6, we provide further details on how each of these distance metrics are used to compare the similarity between two 3D objects (i.e. query model and database models) given 3D objects that are represented by different retrieval methods (descriptors).

2.2.4 3D Content-based Retrieval System (3D-CBRS) and 3D Search Engine

Definition 2.2.1 *A 3D-CBRS (also known as 3D shape search engine) is an application that accepts a 3D model as input query, then retrieves other 3D models from a specified 3D models database, and ranks these retrieved models by their degree of similarity to the query model. This system/application enhances the process of searching a 3-dimensional collection (i.e. database of 3D objects) to retrieve similar objects to a query 3D object.*

The concept of 3D-CBRS involves the application of CV techniques to *3D shape search* problems (i.e. the problems of representation, pre-processing, description, indexing and matching of 3D objects) in order to retrieve similar objects from a database of 3D objects. Much of the recent work on 3D-CBRS can be found in [228] and [268]. The *content-based* concept in 3D-CBRS refers to a search approach that analyses the physical properties of surfaces (measurements), which is the mathematical contents or measurements of 3D objects, such as shape descriptor or feature vectors, rather than their abstract contextual contents (meta-data), such as keywords, tags, colours or appearances. In Figure 2.3, we present a clarified and detailed overview of 3D-CBRS, where the entire process is broken down into three stages, thus:

- **DB 3D Input:** Involves using a particular/target database of 3D objects (e.g., the SHREC'10 dataset described in Section 5.2.2) and performing the following CV processes on the dataset: pre-processing, feature extraction, shape descriptor construction, and indexing the descriptors for all 3D objects in the database.
- **Query 3D Input:** Given a single 3D object for which we want to find other similar 3D objects from the target database, exact CV processes performed for the target database objects are performed on the query object, such as pre-processing, feature extraction, and shape descriptor construction, except indexing.
- **3D Output:** Also known as matching and retrieval stage, involves loading up the indexed database-object descriptors, matching each of these descriptors with the query-object descriptor to find matches (similarities). i.e. those indexed database-object descriptors which are similar/close to the query-object descriptor. These matches (i.e. similar database 3D objects to the query 3D object) may also be ranked based on pre-determined threshold and top ones can be returned as output to the user of the search engine.

Essentially, the "DB 3D Input" stage is generally referred to as the *off-line* phase, while the last two stages ("Query 3D Input" and "3D Output") are generally referred to as the *on-line* phase by most content-based shape retrieval researchers, for example [64]. It is possible to design the 3D-CBRS using several approaches. For example, a user of the system can query the 3D database using a query 3D object, a 2D sketch representation of the query 3D object or a text-based approach which involves the meta-data of the query 3D object. If the sketch-based or text-based approaches are adopted during the on-line phase, then, the retrieval system would have

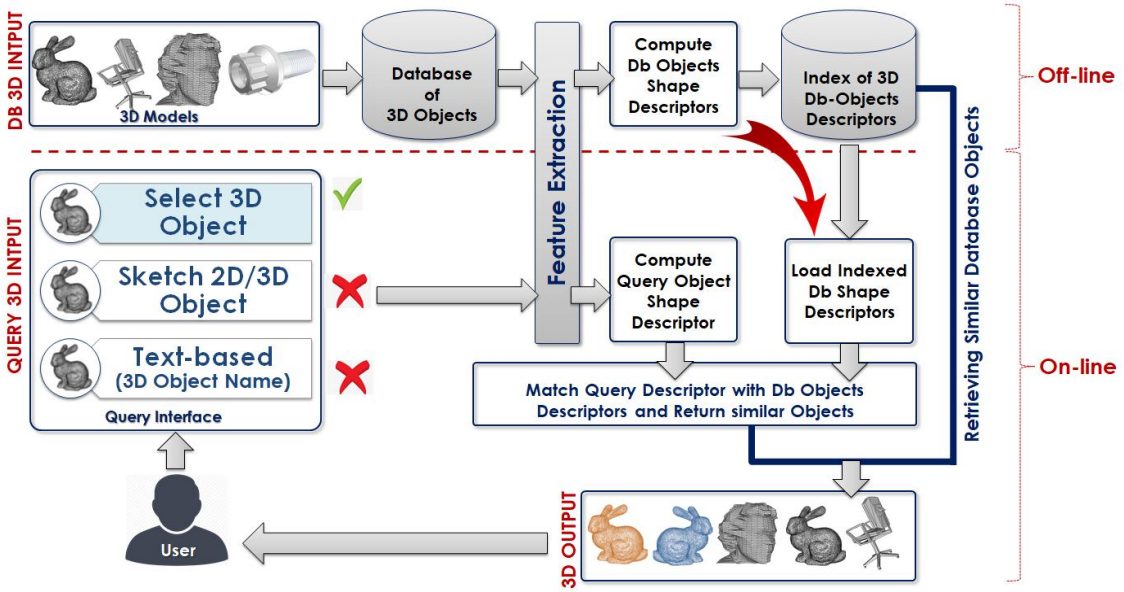


Figure 2.3: General overview of 3D Content-based Retrieval System.

been designed to support database indexing of 3D objects using exact approaches, respectively. The literature in [64] provides more details regarding these. However, this thesis and our research implementations strictly adopts the 3D query object approach due to its suitability and robustness over the other approaches.

The following six key aspects are considered in 3D-CBRS: (i) Object representation, (ii) data pre-processing, (iii) feature extraction and optional feature selection, (iv) shape description, (v) indexing, (vi) matching and retrieval. In this section, we only provide detailed description of the first aspect (i.e. 3D object representation), which is an important aspect to consider for any shape analysis task, including shape retrieval, while the other five aspects are thoroughly described as processes (i.e. as part of our research strategies and techniques) in Section 3.1. However, the 3D-CBRS is incomplete without an effective performances evaluation mechanism for shape descriptors (or retrieval algorithms) prior to deployment. In Section 4.3, we provide details of the performance evaluation approach and metrics we have adopted to validate our 3D shape retrieval methods or descriptors.

2.2.5 3D Search Searching

The final stage of Content-based Shape Retrieval (CBSR) pipeline is to search for 3D object(s), where the user is expected to submit a query object to the CBSR system (3D search engine) and be able to retrieve similar ones already in the object's database. During user search, the query object is first described (i.e. we apply exactly the same features extraction and shape descriptor functions that were applied to all other shapes in the database from which we want to retrieve similar objects) to obtain a query signature, which is then compared to the signatures of database objects already indexed, using a similarity function. Finally, based upon similarity scores and matching results, the most relevant results, sorted according to our similarity function are returned. It is important to mention that the *relevant results* that would be returned are different for different similarity metrics or func-

tions, because they (results) strictly depend on the similarity function or similarity metric (see Section 2.2.3) used.

Prior to searching, shape descriptors are first computed, and used to represent the input query and the database objects. During shape matching, the degree of similarity/(dis)similarity between two descriptors (query object descriptor and each of the database objects' descriptor) is compared using a suitable (dis)similarity or distance metric $d : \mathbb{M} \times \mathbb{M} \longrightarrow \mathbb{R}$, where $\mathbb{M}$ represents space of descriptors. Let us denote the query object's descriptor as $Q \in \mathbb{M}$, and a collection of other descriptors associated to the database objects as $D_i \subset \mathbb{M} : i = 1...N$, where D_i are objects of the same type as the query object. Then, the 3D objects in the collection are represented by a set of indices $1, ..., N$. The distance metric, d used, depends on the type of descriptor computed, and we adopt about five choices of distance metrics, d (described in Section 2.2.3) for performance evaluation purposes. Suppose d is a measure of (dis)similarity between the query object and the database objects, then the database object with descriptor D_{s_1} would be the descriptor of object with the *"best match"* to the query object, having an identifier, s_1 (regarded as the i-th match, where $i = 1$ in this example) which is the *"highest ranked"* similarity *"score"*. This is expressed in Equation (2.10)

$$d(Q, D_{s_1}) = \min_{1 \leq i \leq N} d(Q, D_i), \quad s_1 \in \{1, ..., N\} \tag{2.10}$$

In information retrieval and pattern recognition, finding the *"best match"* to a query object is usually a problem. However, we choose to retrieve only $K \leq N$ ranked objects for the user as the retrieval results of his/her query, where for our retrieval performance evaluation purpose, $K = 32$, as described in Section 4.3.3. However, depending on the retrieval application, a user can specify how many objects, K, should be retrieved. Then, the K matched results (retrieved objects), which are the most similar to the query object, are ranked so that their descriptors $D_{s_1}, ..., D_{s_K}$ $(s_1, ..., s_K \in \{1, ..., N\}, \ i \neq j \Longrightarrow s_i \neq s_j)$ satisfy Equation (2.11).

$$d(Q, D_{s_i}) \leq d(Q, D_{s_{i+1}}), \quad 1 \leq i \leq K - 1 \tag{2.11}$$

Several 3D query and retrieval systems (i.e. 3D search engines) have been developed, such as 3D retrieval engine at Utrecht University [243], Princeton Shape Benchmark [239], the University of Konstanz 3D model similarity search engine [29] and the National Taiwan University 3D model retrieval system [36, 35], among others. 3D retrieval system employs a number of methods and software tools [7] as well as shape matching techniques such as the ones described in [91] (such as the ones we describe in Section 2.2.6). The main idea behind content-based retrieval systems is to develop a platform that uses these techniques to enable users to perform object query by similarity of contents. Compared to traditional 2D image techniques or textual keyword searching, 3D objects retrieval task is not as easy. However, 3D objects searching usually perform better when using 3D descriptors than using there textual keywords or 2D image representations. This is because it may be more difficult to adequately describe a 3D object or properly label (name) all the 3D objects in a dataset in such a way as to provide uniqueness and search efficiency when conducting a 3D search with textual keyword, for example. On the other hand, 2D image-based technique involves the projection of 3D objects into 2D

space [230], where some useful 3D information are lost in the process. Besides, some useful properties of the 3D objects, such as size, height, volumes, etc, are difficult to capture in the 2D space. However, the effectiveness of this approach somehow depends on the application or problem we are trying to solve, and there are a lot of exiting techniques which involves the projection of a single 3D object to multiple 2D images, and proves to be useful in some application.

3D retrieval system (search engines) uses either local or global shape descriptors for its similarity measure. When a query is made to the system, the retrieval engine calculates the query object's descriptors and compares the result to all the stored descriptors from the database, using a distance function. The distance function then measures the similarity between the query object and those of all the objects in retrieval system's database. When this has been done, the search engine sorts database objects in terms of increasing distance values. Shape descriptors are crucial in 3D object retrieval, especially when dealing with a large repository of 3D objects. In Section 2.3, we provide a detailed overview of 3D shape descriptors, including their classifications (see Section 2.4), thus provide a reference for constructing or adopting a shape descriptor for 3D object retrieval system project.

2.2.6 Shape Descriptors Matching Approaches

Searching through a database of 3D objects for an object of interest (query object) can be achieved by either of two broad approaches to shape descriptors matching. They are:

i *local descriptor matching (see Section 2.2.6):* which involves aligning points (or vertices) from the query object to the points of each object in the database to find correspondences (i.e. how well the points or vertices fit, or are close to, each other). A technique similar to the popular ICP algorithm [52], such as the signature of histograms of orientations [235] is typically adopted for this kind of matching.

ii *global descriptor matching (see Section 2.2.6)*: which involves reducing each 3D object in the database, including the query object into a feature-vector, also known as shape signature (i.e. a concise mathematical description of its shape characteristics), then compare the descriptor of the query object to the descriptor of each of the database objects to find a match. For this approach to be achievable, the database shape descriptors must first be indexed (see Section 3.5).

Unlike the global approach (see Section 3.6 and Figure 3.22 for clarity), the local approach to 3D shape matching is very expensive because it involves matching multiple points or key points descriptors of the query object with multiple points or key points descriptors of each of the database object. The local approach is computationally expensive and time consuming, assuming there are exceptionally a substantial number of models in the database, as it would require all individual items in the database to be evaluated every time a query is made. The global approach, however, generally involves dimensionality reduction of all database and query models. Each database 3D object consists of a set of 3D points connected by edges (in the case of triangular meshes) or raw points (in the case of point clouds),

which are transformed to a *feature-vector* (i.e. a set of floating-point numbers) using some mapping or function for each of the 3D objects, thus reducing the problem of finding similar shapes to finding vectors of numbers that are similar to each other between query and database objects.

There are various kinds of data structure for addressing this situation (Graphs, K-d Tree, B-trees, Heaps, hash table, etc.), which turns the problem into a proximity search in a vector space. Some of these data structures are fast and others slow. The goal behind reducing 3D objects to a concise vector of features is to enable them to be comparable and to use the fast data structures, which would result in faster query. For this reason, the choice of the global approach becomes a preference over the local approach when considering 3D querying and retrieval. In this thesis, we adopt the global shape descriptor matching approach (see Section 3.6 and Figure 3.22 for more details), instead, due to its simplicity and ease of computation, and in order to obey some of the needs expressed in Section 2.3.1 (i.e. characteristics of an appropriate shape descriptor). In the sub-sections that follow, we provide an overview of these two approaches to shape descriptor matching, and describe our actual implementation of global descriptors matching in Section 3.6.

Local Descriptors Matching

Here, several key point descriptors, $[K \times D]$ (which describes the local geometrical properties of the underlying 3D surface) would have been computed and indexed for a single 3D object, where K is the number of detected key points per 3D object; D is the dimension or size of each local descriptor, K_i; and the value of K differs for each 3D object. Then, local descriptors matching technique for this type of shape signatures representation generally involves finding correspondence between the local descriptors of $X = [K_x \times D]$ and $Y = [K_y \times D]$, where X and Y are the two 3D objects we want to match, for instance. Alternatively, k-NN (which involves using any of the distance metrics, like Cosine to find the nearest Cosine neighbours between X and Y local descriptors), brute-force, etc., are some of the most common approaches in literature for local descriptor matching. The Bag-of-Words (BoW) technique is also a common approach used to combine and match locally-extracted descriptors. However, as already stated, we do not implement any of these matching techniques for this thesis.

Global Descriptors Matching

With this approach to shape descriptor matching, we ensure that for a single 3D object, only one final signature/descriptor is computed, which is a global shape descriptor as described in Section 2.4.1. During global matching, we compare any two 3D objects as follows. The distance (see Section 2.2.3) between their global descriptors (which are d-dimensional *feature-vectors*) are computed. The normalised distance value, which lies between $[0, 1]$ determines how similar the two shapes are. Our global and hybrid descriptors (described in Section 4.2) were first computed and stored (indexed) for all the 3D models present in each dataset. Then, we considered each descriptor in turn, as a query descriptor, for matching/comparison with every other descriptor in the dataset to obtain their respective similarity score, which is recorded during each match. We then sort a collection of these similarity

scores for all pairs of matching, to create a ranked list, which is sorted based on the similarity scores. With the ranked list, we finally compute a distance/(dis)similarity matrix, DM. The DM contains retrieval scores for the shape retrieval algorithm and distance metric used for that particular dataset.

2.3 3D Shape Descriptors

Definition 2.3.1 *A 3D Shape Descriptor is simply a concise mathematical representation (i.e. a d-dimensional feature-vector) of a given 3D object. It is commonly also referred to as 3D shape signature. According to [149], "3D Shape Descriptor can be viewed as a 'mapping' from 3D object space to some high-dimensional vector space", and the primary goal of research in this field is to produce such "mappings" that can preserve as much information as possible about a 3D object while producing a resulting feature-vector representation in a possibly low-dimensional space.*

Basically, a shape descriptor is a compact mathematical description of a given 3D object, often represented by a vector, a graph, or real numbers, in such a way that its complexity is much less than its corresponding original 3D representation. The primary purpose of this study is to propose such a compact, yet computationally efficient and highly discriminating signature for 3D mesh and point cloud retrieval, and the above definitions are important in order to improve the efficiency and effectiveness of 3D shape recognition and retrieval engines. 3D shape descriptors are derived by first extracting (identify and compute) salient and discriminating features from 3D surfaces, which are then combined to form a compact mathematical representation (vector) for the 3D object. 3D shape descriptors are indispensable for a variety of computer vision, graphics, and pattern recognition tasks, etc. They play a key role in 3D object retrieval tasks, which involves the process of querying a 3D object against a database of 3D objects in order to find similar objects that closely matches the query object. Therefore, it would be almost impossible or inconvenient to directly match a raw 3D query object (i.e. triangular mesh or point cloud data structure) with several other 3D objects in the database, irrespective of whether, or not, the 3D database is small, without first describing the object, that is: converting the objects to shape descriptors. Describing 3D objects makes it easier to index or store a large number of them and to quickly retrieve closest matches during query.

Several methods have been proposed for computing appropriate shape descriptors for 3D shapes. A good 3D shape descriptor must be able to represent the original 3D shape or data such that the descriptor (new representation) is invariant to rotation, translation, and scaling, while being robust to noise, tessellation, and occlusion. We give further details regarding the characteristics of an appropriate shape descriptor in Section 2.3.1. The literature by [64] however noted that, unfortunately, no existing shape descriptor has all these properties, and provided reasonable arguments to support this claim. In their work, they proposed the Spherical Harmonics Transform (SHT) 3D shape descriptor (see Section 2.5.3), stating that the statistically-based shape descriptors such as the shape distributions [173] (see our reviews in Sections 2.6) and feature vectors [53] are usually not discriminating enough to distinguish between similar classes of objects.

Until now, an exceptionally large number of different categories and approaches to 3D shape descriptors have been developed and evaluated against several different 3D benchmark datasets by computer vision/graphics researchers, educators, and experts. However, research contributions within the last two-to-three decades contain the most relevant and up-to-date research findings on 3D shape analysis, retrieval, and pattern recognition. Moreover, it is somehow impractical to address all the existing 3D shape descriptors in literature within the scope of our thesis. Therefore, in the following sections, we provide a detailed and well-informed review of 3D shape descriptors focusing on two broad perspectives (i) broad classifications of 3D shape descriptors (see Section 2.4), and (ii) broad approaches to 3D shape descriptors (see Section 2.5) as illustrated in Figure 2.4, where we review the various types of 3D shape descriptors available in literature within the above-mentioned period. Approaches and techniques we adopt to describe (i.e. compute descriptors for) 3D meshes and point clouds for efficient 3D shape retrieval tasks are presented in Section 3.

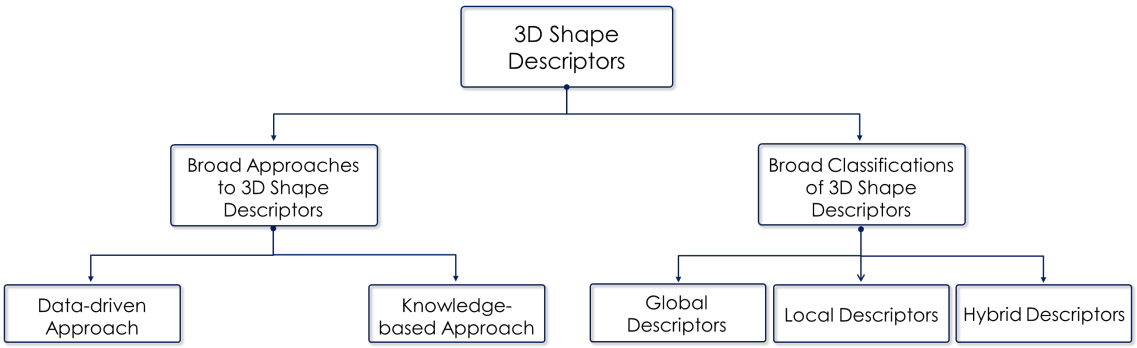


Figure 2.4: Broad overview of 3D shape descriptors in terms of classification and computational approaches.

3D shape descriptors or retrieval methods deal with various forms of 3D objects representation (see Section 2.2.1). For example, some descriptors are designed specifically for 3D triangular meshes [3, 249, 135], others are most suitable for raw 3D point cloud data [200, 263, 142], and many others are suitable for RGB-D, 2.5D range scans, LiDAR data [73, 257], etc. However, the voxel data representations are the most suitable kind of data for 3D shape descriptors utilizing the machine/deep learning computational approach (for data-driven 3D shape descriptors, discussed in Section 2.5.1). In our work and in this review, we focus primarily on those 3D shape descriptors that are mostly suitable for 3D triangular meshes and raw 3D point cloud data (for knowledge-based 3D shape descriptors, discussed in Section 2.5.2).

2.3.1 Characteristics of Appropriate Shape Descriptors

According to [54], six algorithmic criteria are needed to determine the invariance of shape descriptors, thus: *(i) Size of Vector:* the size of the final shape descriptor or *feature-vector, fv*, corresponding to a particular 3D object. *(ii) Feature Extraction Complexity:* the complexity of the feature extraction and shape descriptor construction algorithms that produces the final 3D shape descriptor, *(iii) Matching Complexity:* the complexity of the similarity measure between two shape descriptors being

compared, *(iv) Generality:* specifies whether a specified 3D object descriptor can be applied, for example, to topologically ill-defined 3D object as well as polygon soup, *(v) Geometric Invariance:* specifies if the descriptor is invariant to affine/geometric transformations, as illustrated in Figure 3.4, and *(vi) Topological Invariance:* which specifies if the descriptor is independent of the polygonal representation. In addition, eight desirable properties that a good 2D/3D shape descriptor must have, were mentioned by [64], as follows. A good shape descriptor must be: (i) quick to compute, (ii) concise to store, (iii) easy to index, (iv) invariant under (dis)similarity transformations (such as translation, rotation and scaling), (v) insensitive to noise, i.e. robust or insensitive to sampling errors and some sorts of noises (such as topology or geometric noises), (vi) independent of 3D object representation, tessellation, or genus, (vii) robust to arbitrary topological degeneracies, and (viii) discriminating of shape differences at many scales.

Developing a single shape descriptor that would satisfy all the above conditions/criteria would be great, but remains an open research problem to the best of our knowledge. For example, a given shape descriptor may only be robust to noise, invariant under rigid (similarity) transformation (which is much more challenging to describe and represent), and possibly discriminating of objects' differences at many scales, while another descriptor may be better in terms of being quick to compute, concise to store, easy to index and probably robust to arbitrary topological degeneracies, including being invariance under (dis)similarity transformations. The choice of descriptor for a particular computer vision task would therefore depend on the users' preference and the task involved. In this thesis, we present statistically-based 3D shape retrieval methods which have been able to satisfy most of the above-mentioned criteria, including descriptor size/compactness with good descriptive power. We summarise (list) our original methods in Table 4.1. More details regarding these descriptors are provided in Section 4.2, while experimental evaluations are detailed in Section 5.1.

2.4 Classification of 3D Shape Descriptors

Broadly, depending on the process of computation and usage, 3D shape descriptors (retrieval methods) are classified into two main categories: local and global descriptors. It has been possible to derive a third category, the hybrid descriptor, by a combination of different variants of local descriptors, global descriptors or local and global descriptors as illustrated in Figure 2.5. The process of computing each of the three classes of descriptors is different. The hybrid descriptor or retrieval approach, however, is intended to further improve upon the overall retrieval performances of the resultant combined methods as opposed the individual performances of each of the single descriptors. Detailed review of the types of 3D shape descriptors that falls within each category are presented below.

2.4.1 Global 3D Shape Descriptors

Essentially, global 3D descriptors deal with the global nature of 3D objects. These types of descriptors are more interested in the general aspect of a given object rather than its details. Most 3D shape retrieval tasks make use of global shape descriptors,

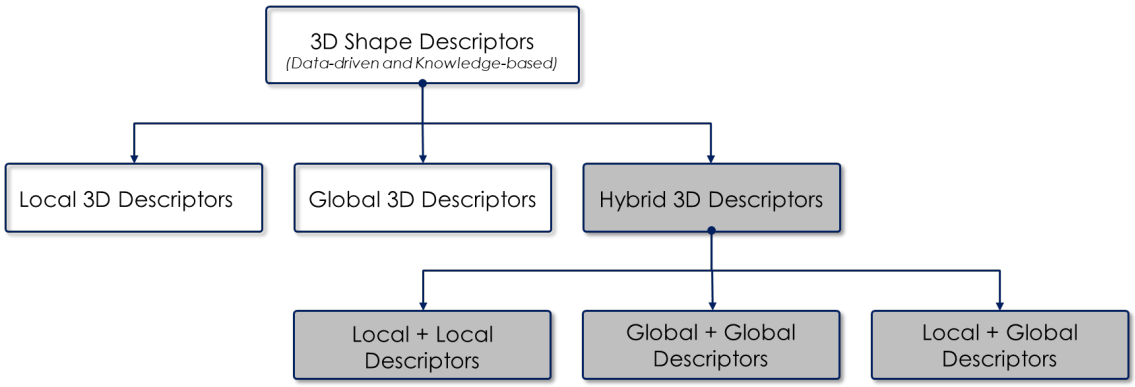


Figure 2.5: 3D shape descriptors, broadly classified into 3 groups: Local, Global and Hybrid descriptors (i.e. *local-local*, *global-global*, or *local-global* descriptors).

such as the SHT [251], Shape distributions [174], Lightfield Descriptors (LFD) [214], etc., which require the complete geometry of a 3D object. Over the years, several global 3D descriptors have been studied and proposed in the literature, with the most popular ones being the shape distributions which are based on statistical distributions of shape functions measuring geometrical properties of 3D objects. These measurements are then binned into histograms. While there are wider range of possibilities to computing shape functions, the work by [174] proposed only five shape functions listed below, which were chosen for their computational simplicity and invariance.

- A3: represents the angle between three random points sample on the surface of a 3D object.
- D1: represents the distance between a fixed point and one random point sample on the surface of a 3D object.
- D2: represents the distance between two random points sample on the surface of a 3D object.
- D3: represents the square root of the area of the triangle formed by three randomly chosen points sample on the surface of a 3D object.
- D4: represents the cube root of the volume of the tetrahedron formed by four randomly chosen points sample on the surface of a 3D object.

Global Features: Global descriptors are computed from global features (i.e. measurements that characterises the global shape of a 3D object), and the five shape functions listed above (i.e. A3, D1, D2, D3 and D4) are typical examples of global features. According to [174], the distribution measuring distances between pairs of random points (D2) is most effective compared to other retrieval methods. Other examples of these type of features are the area, volume, statistical moments, and Fourier transform coefficients [272], statistical moments of the volume/boundary of 3D objects [180], measure of reflective symmetry (specified by two parameters) to every plane through the object's mid-point [101], and the Fourier transform of the volume or the boundary of 3D object (i.e. the ratio of volume to surface of the object), such as: (i) ratio of the cubed surface area of the hull and the squared volume

of the convex hull, tagged "hull compactness", (ii) ratio of object's surface area and the surface area of its convex hull, tagged "hull crumpliness", and (iii) percentage of the convex hull volume not occupied by the object, tagged "hull packing", proposed by [42].

Multi-view 2D Projection (M2DP)

One other typical example of a global 3D shape descriptors is the M2DP [80], which is developed for 3D point cloud and applied to the task of loop closure detection. M2DP involves the projection of 3D cloud to multiple 2D planes from which density signature of points in each plane are computed and combined to produce 196-dimensional *feature-vector*, *fv*. In Chapter 4, Section 4.2.3, we adopt the M2DP descriptor to further improve the performance of our APPFD, due to its success and computational efficiency. For more details regarding the M2DP global descriptor, we refer the reader to [80].

Generally, global feature-based descriptors (global descriptors) analyse 3D objects as a whole unit, but fails to capture the specific details of a 3D shape. Since only global features are used to characterize the overall shape of objects these methods have straightforward implementation, but are not very discriminating about the object's details. Therefore, they can be used in combination with other methods to improve retrieval performances and/or results. Secondly, global shape descriptors are computationally efficient [91] and have proven to be effective methods for describing rigid 3D models. Some global shape descriptors such as the shape distributions can be used to represent objects from different broad categories [275]. They are robust to noise and can distinguish wide categories of 3D objects.

The Key limitations of the global feature-based methods (global descriptors) are that they fail to capture the specific details of a shape and overall, are not very robust They fail to discriminate among locally dissimilar shapes [91], nor efficient at discriminating objects that are globally similar but with different minute details in their shapes [54]. In other words, global shape descriptors are capable of comparing entire surface of 3D objects but unable to locally compare surface points for the purpose of point matching. For this reason, local descriptors are essential. For a broader overview regarding global features and shape descriptors, we refer the reader to [275, 91, 228, 29].

2.4.2 Local 3D Shape Descriptors

The characterisation of local surface property is an open research problem that is gaining more popularity over the years [231], and 3D shape descriptors have been proposed, which are based on local features of object's surface. Interestingly, local shape descriptors have several key advantages, and matching 3D surfaces with local shape descriptors has become a popular research trend within this period, basically because local descriptors are able to efficiently describe surfaces with missing data, clutter, occlusion, and they are affine-invariant. It is important to note that local shape descriptors are computed from locally extracted features (i.e. local features) from the surface on a 3D object. Local feature-based method describes a 3D object using selected number of surface points and provide different approaches that

considers the *surface's shape* within a local neighbourhood of points on the entire object's surface. Consequently, a descriptor for each surface point is used and several local descriptors are derived for a single 3D object instead of a single descriptor for the entire 3D object.

Recently, researchers have also proposed methods using the BoW paradigm which provides a framework to compare two 2D/3D objects using local descriptors. The BoW approach has been successful in both text and 2D image retrieval, and has shown promising results in 3D shape retrieval. In the BoW approach, local descriptors are first computed for selected local surface patches or region (see Section 4.2.1) around estimated key points for a given 3D object, then each of these descriptors is assigned a frequency/probability value from a pre-constructed dictionary (i.e. vocabulary or visual word). The vocabulary becomes the final feature-vector for the object, which has dimensionality equal to the size of the vocabulary. Finally, a histogram of visual words or vocabulary is computed for the object, and two objects can be matched by comparing their BoW histograms, similar to matching global descriptors. The work by [170] used uniformly distributed depth-buffer views of normalized 3D objects as features to construct local Scale-Invariant Feature Transform (SIFT) descriptors and the BoW model was used to produce final histogram of visual words. Then, the Kullback-Leibler divergence metric (see Section 2.2.3) was used to determine the (dis)similarity between two BoW histograms. However, while the BoW paradigm has been a common approach for combining local shape descriptors, it is not the only way out for the complexity in local shape descriptors matching. Several other local shape descriptors have been proposed independent of the bag-of-words model.

The most popular approach to local shape descriptors remains the statistical approach (see detailed review in Section 2.6) which involves computing histograms of local (also global) features, where in this instance, local feature measurements on the surface representing a given 3D object are binned into histograms and the normalised histograms are used as the final local shape descriptors. The spin images have been used as local descriptors (2D histograms counting the number of surface points at various locations) by [213] and [144]. The work by [110] used curvature based local descriptors (i.e. the mean and Gaussian curvatures), including shape index, and curvedness. Alternatively, the Heat Kernel Signature (HKS) is another type of popular local shape descriptor, which was introduced by [226]. Also, random set of vertices were chosen from 3D triangular meshes as seeds by [117], who then applied Lloyd relaxation iterations to propose a local shape descriptor for 3D matching.

In addition to the characteristics of appropriate shape descriptors mentioned in Section 2.3.1, a good local descriptor is one that must be able to account for the local shape of the surface surrounding a given point. Local descriptor usually requires a previous key point detection step, which complicates its adaptation to recognize non-rigid objects [148]. Secondly, because local descriptors are computed from surface regions (i.e. LSP) around key points, the challenge with these types of descriptors therefore remains that of detecting appropriate and repeatable key points. Key points detection for 3D objects is more complicated than for 2D images due to images having richer set of distinct features. However, [220] extended

the Harris 2D key point detection to 3D objects, i.e. 3D-Harris. The literature in [234] also reviewed several other techniques which have been proposed to perform detection of repeatable and distinctive key points in 3D surfaces. In our work, we implemented a number of popular state-of-the-art key points detectors for 3D models, but recorded several other issues aside from repeatability (see Section 3.3.5). Part of our research goal and efforts has been to develop robust and computationally efficient local 3D shape descriptor that satisfy most of the conditions mentioned for appropriate shape descriptors (see Section 2.3.1). We therefore adopted an alternative voxel-grid downsampling technique to select a set of points (used as key points) for every 3D objects in the dataset. We describe this process in Section 3.3.5 and 3.3.5.

2.4.3 Hybrid 3D Shape Descriptors

The primary goal of the hybrid descriptor approach is to further improve the overall retrieval performances of the global and local descriptors by combining either of these descriptors as illustrated in Figure 2.5. For example, the work by [66] combined two local descriptors: (i) the eccentricity function, which produced spatial information about a 3D object and (ii) the local-diameter function describing local surface of 3D model, to produce a global 2D histogram as the final shape descriptor for a given 3D object. The literature in [263] formed a hybrid 3D descriptor known as the Ensemble of Shape Function (ESF). In order to combine the descriptive powers of both local and global descriptors, the work by [4] combined the Viewpoint Feature Histogram (VFH) global descriptor [201], and the Fast Point Feature Histogram (FPFH) local descriptor to form a hybrid (FPFH + VFH) descriptor. In the same manner, [242]combined three distributions (a distance histogram, a curvature histogram, and an elementary volume histogram) to construct a hybrid descriptor for 3D object indexing and retrieval. Two separate studies were performed by [178] on 3D shape matching, where hybrid global descriptor was constructed by combining 2D and 3D descriptors. In one of their studies, they modelled the property of an object using Fourier Transform descriptor technique, then combined 2D image view of 3D object (panoramic views for unsupervised 3D object) [179]. In another study, they combined 2D descriptors which captures the distance to surface points from 6 sides of a cube with global 3D spherical harmonics, computed over the entire 3D object.

Essentially, the hybrid technique can be used to combine different statistically-based 3D descriptors aimed at enhancing the overall performance of 3D shape retrieval. This approach is known for its exceptional performances, and literature reviews have revealed its popularity due to several work utilising this approach to 3D shape description. However, the practicability of some hybrid-based techniques is limited due to the high computational cost. To improve the retrieval efficiency of 3D shapes, we propose a number of highly discriminative, yet computationally efficient hybrid 3D shape descriptors for 3D meshes and point cloud shapes. In order to address the practicability concern and fulfil most of the criteria specified in Section 2.3.1, we compute our 3D shape retrieval methods with incredibly small number of point samples, say between $N = 3,500$ points to $N = 10,000$ points, while still recording impressive retrieval performance scores (see Section 4) compared with

state-of-the-art methods for a given benchmark dataset and retrieval challenge. Detailed description of our hybrid descriptor methods is provided in Section 4.2.

In our work however, we identified and considered the gaps presented by the local, global, and hybrid approaches to shape descriptors computation while proposing improved solutions to shape retrieval, in each of these approaches, that accounts for existing research gaps. We have mainly contributed to the local (see Section 2.4.2) and hybrid (see Section 2.4.3) shape descriptors and evaluated their retrieval performances using several performance evaluation metrics. We implement these descriptors in Section 4.2. Therefore, our local and hybrid descriptors, which combines the descriptive powers of both global and local descriptors, produce retrieval results with better performances across a wider range of 3D benchmark datasets - see experimental evaluation sections, Chapter 4.

2.5 Approaches to 3D Shape Descriptors

According to [195], and [20], all methods involving analysis, feature extraction and shape descriptor computations for 2D/3D shapes retrieval, classification, detection, and recognition adopts either one of two broad approaches depending on the computational technique adopted or the size of dataset. The first is the Data-driven approach (see Section 2.5.1), and the second, Knowledge-based approach (see Section 2.5.2). In Section 2.4, we discuss three broad classifications involving 3D shape descriptors (global, local and hybrid), depending on mode of feature extraction and final descriptors combinations. However, irrespective of their classifications, the general process of 3D shape descriptors computation involves either of the two broad approaches mentioned above. In this section, we provide detailed overview of these two broad approaches to 3D shape descriptors computation, including reviewing several different state-of-the-art 3D shape descriptors/retrieval methods that are within each of the sub-categories of shape descriptors under each approach broad approach, particularly, the knowledge-based approach, which forms the basis for our research work and thesis contributions.

2.5.1 Data-driven 3D Shape Descriptors

Definition 2.5.1 ***Data-driven Approach:*** *The data-driven approach to 3D shape descriptors involves the application of Machine Learning (ML) or Deep Learning (DL) techniques to learn representations from given dataset, thus yielding precise outcome on even larger datasets.*

The data-driven approach, which involves machine learning algorithms and techniques have since been used in developing 3D shape descriptor [85], but have recently gained popularity and attracted more attention with the emergence of the deep learning algorithms [84, 83]. Briefly, the general pipeline for the data-driven approach to shape descriptor constructions is as follows: (i) The entire database or dataset would have to first be divided into two sets (training and validation/testing set), and training dataset (75% - 85% of original dataset) is presented to the machine-learning (ML) or deep-learning (DL) algorithm that learns the patterns in the input data. Typically, the original data itself could be used as input or set(s) of extracted

features from the data. (ii) The input data is trained (i.e. the algorithm learns patterns from the input data) until the algorithm converges to a potentially meaningful descriptive model. This happens when some criteria are met, depending on the parameter settings. The learning stage can either be supervised or unsupervised learning. (iii) Following the training phase, the system is able to generate shape descriptors for unseen data samples. In the case of supervised learning, the model (ML/DL) needs to first be validated for accuracy using training and testing labels which serves as ground truth, while unsupervised learning involves a different approach.

Unlike the knowledge-based approach (see Section 2.5.2), a key advantage of the data-driven approach to computing 3D shape descriptors is that 3D features are automatically learned from training data with little (in the case of supervised or semi-supervised learning) or no (unsupervised learning) expert's knowledge [20]. Secondly, this approach has the advantage of analysing and extracting meaningful knowledge from large volumes of data. Unfortunately, despite its many benefits, the data-driven approach to 3D shape descriptor computation is not without its complications.

First, ordinary ML algorithms has not been robust enough to produce desired accuracy or descriptiveness. Instead, the DL technique has been shown to record better retrieval performances. However, the limitations with either of these approaches or techniques are: (i) they are highly parametrised, and (ii) the deep learning method depending on remarkably high computer resources (hardware and software) to achieve optimal/desired performance, therefore its suitability for practical real-time computer vision/graphics application remain an issue. Secondly, the data-driven techniques can be computationally prohibitive due to the large volumes of data they analyse, whereby they perform poorly when the size of data is small, and raises plenty of research concerns. For example, PartNet: A Large-scale Benchmark for Fine-grained and Hierarchical Part-level 3D Object Understanding [160] and ShapeNet: A Large-scale Dataset of 3D Shapes [34] contains over 26,000 and 63,000 3D shapes, respectively. A few other large repositories like these exist which only the data-driven technique is suitable for constructing their 3D shape descriptors. Besides the remarkably high computational costs to construct shape descriptors for these kinds of datasets using the data-driven approach, large amount of storage space is also required. In summary, one of the key disadvantages of the Machine Learning (ML)/Deep Learning (DL) methods or the data-driven approach is the requirement of large amount of training data which limits their applicability and effectiveness. Therefore, the knowledge-based approach (see Section 2.5.2) is essential and expert's knowledge becomes important. This is because the knowledge-based approach does not require a large volume of data to function.

Finally, the number of small-sized and domain-specific 3D shape repositories which are suitable for the knowledge-based techniques (see Section 2.5.2) to 3D shape descriptors (retrieval methods) greatly outnumbers the large-sized repositories used by data-driven techniques. As an example, detailed descriptions of 3D shape repositories suitable for the knowledge-based techniques are provided in Chapter 5.2. Alternatively, as a further proof, over 200 uniquely-related literatures have

been cited and/or referenced in this thesis, and the research work by each of these literatures utilises small-sized domain-specific databases for their experimental evaluations. Therefore, in this thesis we exclusively focus on the knowledge-based approach to 3D shape retrieval methods, as described in the next sub-sections, instead of the data-driven approach. In this thesis, our primary focus is on the knowledge-based approach to 3D shape descriptor computation due to the reasons explained above, but in this section, we first provide a brief overview of the data-driven approach for the sake of clarity and comparison. However, for an in-depth analysis and review of the data-driven approach to 3D shape descriptor computation, we refer the reader to [128, 195].

2.5.2 Knowledge-based 3D Shape Descriptors

Definition 2.5.2 ***Knowledge-based Approach:*** *The knowledge-based approach to 3D shape descriptors involves the manual extraction of hand-crafted features from 3D objects for shape descriptors computation, by experienced and knowledgeable computer vision/graphics experts and researchers.*

In knowledge-based approach, features are extracted from data using hand-crafted techniques. This approach involves human supervision or hard-coded rules whereby experienced experts are needed to extract hand-crafted features from 3D shapes for the construction of 3D shape descriptors. Unlike the data-driven approach which builds more generalizable shape descriptors (through large training dataset), the traditional approach using knowledge-based techniques is task-specific, less generic and does not require a large dataset in order to succeed. In this section, we provide a comprehensive literature survey of local, global as well as hybrid 3D shape descriptors which falls within the knowledge-based approach.

Reasons we adopt the Knowledge-based approach: Basically, the justification for adopting the knowledge-based approach is mainly due to the limited data available, computational efficiency, limited resources available, such as the Graphics Processing Unit (GPU) and memory for storage. Unlike the knowledge-based approach, the data-driven approach further complicates the process of 3D descriptor computation by depending on already extracted hand-crafted features and in most cases, already computed shape descriptors from expert's knowledge. For example, low-level features such as spin images, curvature (Gaussian curvatures, mean curvatures, principal curvatures etc.), and Average Geodesic Distance (AGD) have been used in the literature by machine learning algorithms to build shape descriptors [75, 219, 280]. Also, other spectral descriptors such as the HKS [25], Wave Kernel Signature (WKS) [8] are local 3D descriptors which have become the building blocks of many data-driven 3D shape descriptor approaches in the literature [155, 24, 18], which inspired researchers to construct data-driven shape descriptors in the spectral domain. All these features and descriptors are derived using the knowledge-based hand-crafted feature technique in the first place. What is even more interesting is that researchers who adopt purely the data-driven approach to 3D shape descriptors uses the HKS and WKS as an evaluation standard for the performance of their descriptors [265, 20]. Above reveals that the success of the data-driven approach, to a great extent, depends on the knowledge-based approach in the first place. To this

end, the need and importance of the knowledge-based approach cannot be overemphasised.

2.5.3 Categories of Knowledge-based 3D Shape Descriptors

Essentially, depending on the different 3D models data type utilised, and the corresponding 3D shape retrieval method, existing knowledge-based approach to 3D shape descriptors can further be sub-divided into four categories: (i) View-based, (ii) Transform-based, (iii) Graph-based (Structural), and (iv) Statistically-based approaches to 3D shape retrieval methods. The literature in [54] equally overviewed these four categories (but referred to Graph-based as Structural-based), including a fifth category, the *normative aspect*. In contrast, the literatures in [228, 229] and [275] categorised 3D shape descriptor into three broad categories, as follows: (i) Feature-based, (ii) Graph-based, and (iii) Geometry-based and/or Others, respectively. In the Feature-based category, three other sub-categories of 3D shape descriptors were reported by [228, 229], which are: Global-features (Global-feature Distribution), Spatial-map, and Local-features; while [275] expanded descriptors in this category to four other sub-categories, thus: Global-features, Spatial-map, Local-features, and Distribution-based. In the second category (i.e. Graph-based), both [228, 229] and [275] agreed on the following three sub-categories: Model-graph, Skeletal, and Reeb-graph. However, there was a contradiction in their mention of the third category. Here, four other sub-categories: View-based, Volumetric Error-based, Weighted Point set-based, and Deformation-based were mentioned by [228, 229]. The literature in [275] did not however name their third category, but mentioned three individual 3D shape descriptors: The Extended Gaussian Images (EGI), Complex Extended Gaussian Images (CEGI), and 3D Zernike moments in their *"others"* categories, whereas in our classification (see Section 2.5.3), these three descriptors are categorised under the *Transform-based* group. We provide a side-by-side comparison of the classification of 3D shape descriptors by both [228, 229] and [275] in Figure 2.7.

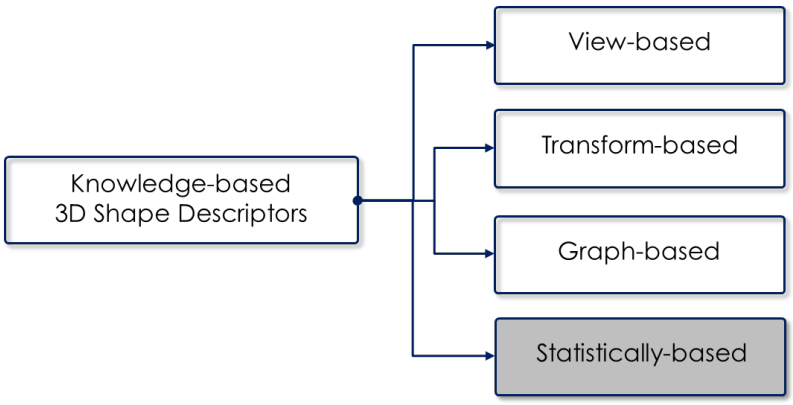
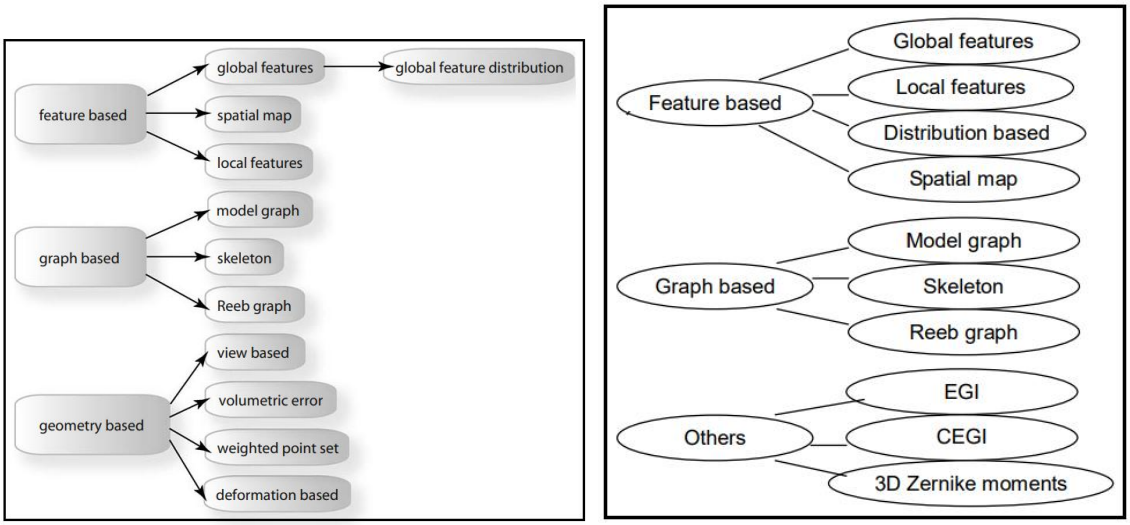


Figure 2.6: Classification of Knowledge-based 3D shape descriptors

Recently, the literature survey by [103] provided five categorisations for 2D and 3D shape descriptors, thus: (i) View-based, (ii) Histogram-based, (iii) Transform-based, (iv) Graph-based, and (v) Hybrid 3D Descriptors. We do not consider their fifth categorisation (i.e. Hybrid 3D Descriptors) to be suitable in this context of

shape descriptors categorisation, primarily because the hybrid shape descriptors fall within a broader classification of 3D shape descriptors, encompassing both the data-driven and knowledge-based, as described in Section 2.4 - see Figure 2.5. Secondly, based on the context of this thesis, the hybrid 3D shape descriptor does not exclusively belong to the knowledge-based approach to 3D shape descriptors. In addition, the hybrid descriptors mentioned by [103] are simply a combination of individual descriptors that may already belong to either or all the other categories, which are classified mainly based upon their method of feature extraction and shape descriptor computation, rather than just final combination.



(a) 3D descriptors grouping by [228, 229] (b) 3D descriptors grouping by [275]

Figure 2.7: Comparison of 3D shape descriptors classification by two different literatures.

Considering all the reviews of 3D shape descriptors classification in the literatures we have indicated in this section, such as those in [228, 229, 275] and [103], it becomes obvious that classifying existing 3D shape descriptors to commonly agrees with several different research hypothesis is a non-trivial task, due to the overwhelming number of available descriptors over the past decades. However, in this thesis, we have been able to clearly provide 3D shape descriptors classification which, to a great extent, agrees with the categorisations of other literatures we have mentioned [228, 229, 275, 103], but different in context and scope. As previously stated, in this thesis, we only briefly review knowledge-based 3D shape descriptors within the four sub-categories in Figure 2.6, as outlined in the following sub-sections. Although [54] provided detailed analysis of 3D shape descriptors within these sub-categories, their review, however, was limited to just a few mention and additional 3D retrieval methods have since been developed which are not mentioned in [54].

View-based Approach:

Generally, view-based descriptor technique deals with the similarity of two 3D objects which appears similar from all viewing angles. This technique uses a single view or multiple views of a given 3D object for its representation. Digital or virtual cameras may be used to obtain a collection of these views. For example, Figure 2.8

illustrates a single view of several 3D objects, while Figure 2.9 show a single 3D object represented by three 2D views around the 3D object. In order to obtain different views of the same object, the camera angle would have to change each time a shot is taken or each object is rotated at different angles to a fixed camera for each shot. Human beings have the ability to visually recognize any object from single view. The work by [192] explored the issue of whether 3D model recognition should rely on internal representations that are inherently 3D or on collections of 2D views. They showed that in a human visual system, a 3D model is represented by a set of 2D views [54].

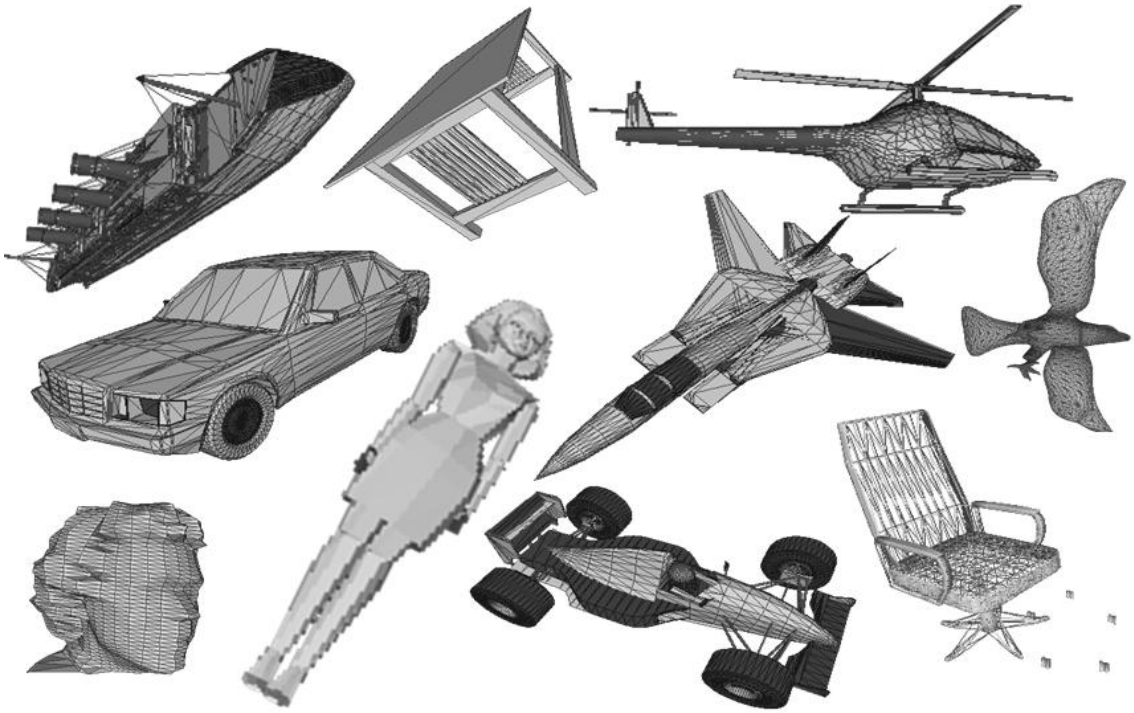


Figure 2.8: Example single view of different 3D objects. View-based 3D object descriptor approach uses a single view or multiple views of 3D object representation instead of the actual 3D model.

The view-based 3D shape descriptors have been widely used in CAD applications. According to [36, 6], the Fourier descriptor and the Zernike moments are the most widely used features to represent 2D views of 3D objects. Popular descriptors that fall within this category are the Lightfield Descriptor (LFD) [214], the depth buffer-based descriptor (BBD) [181], aspect graph [127], and adaptive view clustering (AVC) [134], etc. A 3D descriptor from [273] that is made up of rotationally invariant 2D images Fourier descriptor, was introduced by [171], using a set of depth-buffer images viewed from 42 viewpoints as the final feature-vectors representing a given 3D object. A total of 42^2 possible combinations of two sets of feature-vectors were derived from matching two 3D objects by minimizing the distance between these combinations. Alternatively, a single view of a 3D object was used as a query by [44], where, for each 3D model several 2D views/images were derived as signature. Clustering algorithm was then applied to these 2D views to select a single view as a representative of each cluster using a shock-graph, thereby

keeping the total number of views for each 3D object small. 3D object recognition was then performed by matching a view associated with the query 3D object with those of other database 3D items using shock-graph matching. In this implementation, the shock-graph had indexing problem which resulted to linear search of all database 2D views during retrieval phase.

The view-based descriptor technique was used in [64], by computing 13 thumbnail images representing the silhouette (outer shape boundary) of the 3D object as seen from 13 orthographic view directions. They implemented a 2D sketch query interface for retrieval, where a descriptor comprising a number of binary images is acquired during the pre-processing phase, for every 3D object. Then, during the query phase, a 2D image sketch or a 2D image representation of 3D objects in the database can be used as a query to retrieve 3D database objects whose binary images matches the query sketch/image. Similarly, the work by [146] applies view-based technique to 3D shape retrieval employing a query user interface that is 2D. Essentially, 3D shape retrieval is done by matching the descriptors of the query 2D sketches with those of the 3D objects in the database by simply using 2D image matching technique. Additionally, the view-based strategy by [250] described a 3D shape descriptor using 3 silhouette images perpendicular to the $[x, y, z]$-axes of a canonical coordinate system, and [214, 36] applied same, for 3D shape retrieval, using a light field descriptor (Zernike moments and Fourier descriptors) which matches 10 silhouettes (i.e. 2D images), derived from 10 evenly-distributed viewing angles on the viewing sphere of a given 3D object. This concept is illustrated in Figure 2.10. Matching two 3D objects is therefore reduced to the minimal dissimilarity gotten by rotating the viewing sphere of one light field descriptor relative to the other. Although the Lightfield descriptor has the advantage of being highly discriminating among several other 3D shape descriptors, it utilises a higher storage and computational costs than most other descriptors. Various other view-based methods exist in literature [115].

However, 3D shape retrieval using the view-based approach involved four main stages: (i) *view capture*, where images of different views of a single 3D object is being captured with the help of a fixed or moving camera; (ii) *view selection*, where only the best representative 2D views of the 3D object are filtered out from the many views that were previously captured in order to avoid redundancy and high computational costs. (iii) *feature extraction*, where 2D features are extracted from the respective multi 2D views of images representing a particular 3D shape, and (iv) *object matching*, requires many-to-many matching of multiple 2D views of two 3D objects, unlike the existing 2D image retrieval tasks that are based on one-to-one matching of images between two images.

The downside of the view-based 3D shape descriptors approach is manifold: First, considering the feature extraction stage of the view-based approach, mentioned above, the spatial correlation among different 2D views need to be strictly considered, and this is still an open research problem which requires further investigation. Secondly, due to the special characteristics of 3D data, it is still difficult to extract features for multiple 2D views of a single 3D object with concavity and self-occlusion. Secondly, due to the many-to-many matching of the multiple 2D views of 3D objects, it becomes incredibly challenging to determine how best to

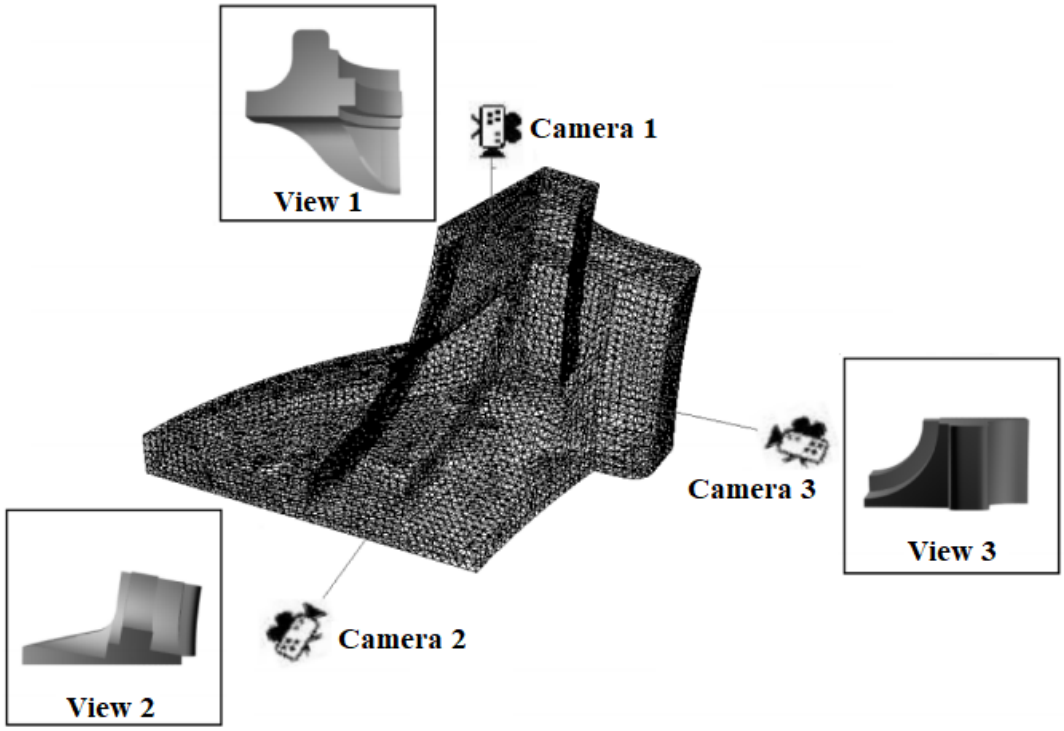


Figure 2.9: A single 3D object (middle) represented by three 2D views obtained from three different camera positions around the object. Image Source, courtesy [54].

conduct many-to-many view matching and estimate the relevance among different 3D objects. However, for more information regarding the view-based approach to 3D shape retrieval, we refer the reader to [54, 67, 276].

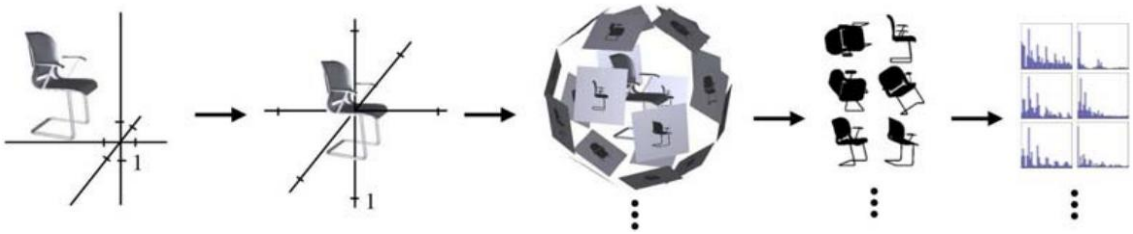


Figure 2.10: The Lightfield Descriptors extraction for a 3D chair model, courtesy [214].

Transform-based Approach:

Transform-based approach to 3D shape descriptors involves the transformation of the 3D shape from 3D Euclidean space representation to another space representation [54]. Popular 3D shape descriptors that fall within this category are: Spherical Harmonics Transform (SHT), Spherical Extent Function, Radial Spherical Extent Function, Extended Gaussian Images (EGI) [88], Complex Extended Gaussian Image (CEGI) [100], Spin Images [96], 3D Zernike Moments [168], etc. Another type of 3D descriptor, the Reflective Symmetries [101] also falls within this category. Reflective Symmetry of 2D and 3D shapes is a problem of finding the main axes of symmetry (i.e. reflective symmetry for an arbitrary 3D model for all planes through

the model's center of mass, even if they are not planes of symmetry), or determining that none exist. We provide a brief review of only the popular descriptors within this category below. The reader is referred to [64, 101, 168, 88, 96, 100], for more in-depth reviews of these descriptors.

The SHT is a rotational invariant 3D shape descriptor whose main idea is to decompose a 3D model into a collection of functions defined on concentric spheres and to use spherical harmonics to discard orientation information (phase) for each one, thus yielding a shape descriptor that is both orientation invariant and descriptive [64]. The literature in [251] described a 3D surface by computing spherical harmonics for spherical extent function, a method which needed pose normalization in order to be rotational invariance. Comparison between two 3D objects was achieved using the Euclidean distance between the Fourier transforms of the object's descriptors. A general Spherical Harmonics approach (only applies to function on a voxel grid or spherical functions) was used to transform rotational-dependent shape descriptors into rotational-independent ones by [270], where they computed a rotational invariant spherical harmonics descriptor through spherical function decomposition. Their method took care of the need for pose normalisation due to the final descriptor being rotation invariant. However, because the SHT uses a Voxel grid-based representation, many fine details are lost.

The EGI method characterises a 3D model in terms of its distribution of surface normal vectors. It is designed mainly for pose normalization/determination and used histogram to record the variation of surface area with surface orientation [88]. In their work, [64] aligned the EGI for each model based on its principal axes, and compared two aligned EGIs by computing their L_2 difference. Several other works has been done using the EGI descriptor, including a variant of the EGI, the CEGI mentioned above. The main advantages of the EGI descriptors include having unique representation for convex objects without occlusions, avoiding the more difficult problem of local feature matching, and because it does not contain any direct distance information, it is considered translation invariant [275]. Also, they are good at discriminating between fabricated and natural objects, but not that good at making detailed class distinctions [229, 228]. However, it does not contain any direct distance information and presents confusion in non-convex objects [275].

The 3D Zernike moments is a rotation invariant shape descriptor that captures object coherence in the radial direction as well as in the direction along a sphere [228]. It is a wonderful descriptor for 3D shapes dissimilar in local parts, has the advantages of capturing global information about a 3D object, and does not require closed boundaries [168]. Zernike moments can be described as a projection of the function defining an object onto a set of orthonormal functions within a unit sphere. They can be considered as the magnitudes of a set of orthogonal complex moments of the 3D shape and the natural extensions of spherical harmonics-based descriptors [275]. In summary, the 3D Zernike moments has an advantage of superiority over others in this category with regards to discrimination power, noise sensitivity, and data redundancy. Also, allows for easy 3D shape reconstruction, but sadly, it always produces high order moments due to the high instability of its geometrical moments [275]. Considering the above analysis, the Zernike moments fits within two categories: the

view-based and transformed-based.

Another type of 3D shape descriptor within the transform-based category is the Spin Images proposed by [96], which was developed to capture both local and global features of 3D objects and supports a view-independent object recognition. The Spin Images on the other hand, is essentially 2D histograms of the surface locations around a point. They are invariant to rigid transformation [97], rotation, scale, and pose invariant [50]. As a result, have been particularly successful in the registration of range images, object recognition, and shape matching [26, 31, 39]. By adjustment of its support parameters, spin-images can be smoothly transformed from global to local representations [97]. Describing 3D objects based on above definition of spin images makes the spin images appear to be a part of either view-based or statistically-based approaches. However, based on the above definition, we can see that spin images shares similar characteristic with view-based approaches. For detailed description and more in-depth analysis of the spin images 3D shape descriptor, we refer the reader to the following literatures: [97, 26, 31, 39, 54, 50]. However, according to [101], The main characteristic of most transform-based 3D shape descriptors is their ability to describe the global properties of a 3D object, and they are defined over a canonical parametrisation (i.e. sphere). Unfortunately, also, these classes of shape descriptors require a priori registration into a canonical coordinate system, which is difficult to achieve robustly [64].

Graph-based (Structural) Approach:

The structural approach to 3D shape descriptors involves the use of high-level information (e.g. a skeleton or a graph) to describe the structure of 3D objects. A typical example of 3D shape descriptor that falls within this category is the graph-based shape retrieval method, which describes the structural components of 3D objects, that are attributed with geometric characteristics and their relational connections with each other) [2]. In the graph-based approach, the structural description of 3D object is created using the Attributed Relational Graph (ARG or Extended Reeb Graph (ERG) [11] concept, where meaningful components of the object can be extracted using a segmentation algorithm, which are then represented as the nodes of a graph, and the relationship of the components with each other are represented as the edges of the graph [2]. Following this approach, the problem of shape matching is therefore transformed into the problem of matching the ARG/ERG of a query 3D object with that of the objects stored in the database [151, 107]. The graph-based approach is illustrated in Figure 2.11, which is the 3D model of a dog (left) with its corresponding skeletal graph (ERG) representation (right), based on 3D segmentation of its body parts (middle). The general idea is to derive 1D skeletal curves from a 3D object such that each curve represents a significant part of the object. These curves are then converted to an attributed graph representation (a skeletal graph), which can be used for indexing, matching, segmentation, correspondence finding, etc.

The structural approach to 3D descriptors is also termed the graph-based approach, and is further sub-divided into three categories: (i) Reeb Graphs (ii) Skeletal Graphs and (iii) Model graphs [229]. First, the Reeb Graph is a topological structure that encodes the connectivity relations of the critical points of a Morse function [112] defined on an input surface. It is also seen as a schematic way of

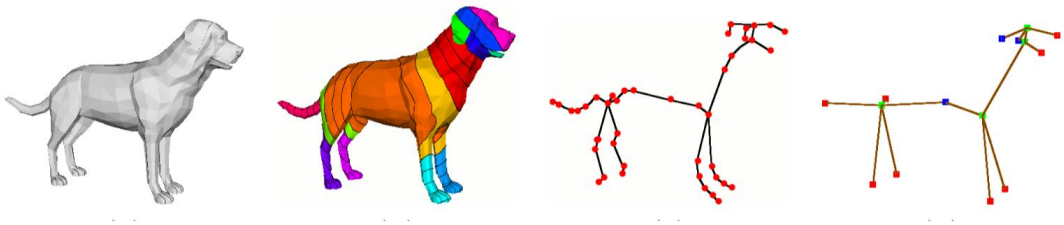


Figure 2.11: 3D shape descriptors based on Extended Reeb Graph - Courtesy: S. Biasotti [11].

presenting a Morse function [54]. For a detailed explanation regarding the RG approach refer the reader to [54]. The skeletal approach, on the other hand, uses skeletal graph to encode geometric and topological information about a 3D object. With each node of the skeletal graph a topological signature vector is associated by encoding the topology of the sub-trees rooted at the node. Thereafter, indexing of the skeletal graph is achieved by storing the topological signature vector for each node. Overall, the model graph-based approaches are mainly suitable for describing the rigid CAD/CAM 3D models, but are difficult to apply to non-rigid models such as humans and animals, although [281] tried implementing a similar approach to model graphs for the retrieval of non-rigid shapes.

In summary, the reeb graph and skeletal graph approaches are applicable to both rigid and non-rigid 3D models represented as voxels, while the model graph-based approaches are only applicable to rigid models. According to [86] model graphs are extracted from solid model representations used by most CAD systems, with an exception of the model graph-based approach by [281], which is also applicable to non-rigid shapes. However, for structural/graph-based 3D descriptors, the complexity of the exact computation of a metric obeying the triangle inequality prevents practical application. Hence, the efficient implementation of approximate matching methods is a current research issue. In addition, a pure graph-based method is overly sensitive to noise and details, including having a limited discriminating power, because only topology is considered and minor changes in topology may result in significant differences in similarity. As a result, the graph-based method is often combined with other methods in order to improve overall discriminative abilities [229].

Knowledge-based Approach Summary and Statistically-based Approach

Shape retrieval methods in each of these categories are not completely disjoint, which implies that there are commonalities between them. For example, it is possible to convert the spin images shape descriptor from a view-based to a transform-based descriptor and vice-versa in terms of computational approach, by adjusting its parameters. Alternatively, the Spin Images shape descriptor can be categorised either as a global or a local descriptor (see Sections 2.4) based on its parameters settings. Considering all the approaches to 3D shape descriptors, we found the statistical approach to be the most dominating (i.e. popular) in literature and convenient in terms of implementation. The statistical approach to shape descriptors computation also accounts for several recorded successes in 2D and 3D content-based shape re-

trieval tasks over the decades according to literature reviews (see Section 2.6), where they have recorded highly impressive performance evaluation results for detection, classification, and retrieval tasks, etc. Motivated by its successes and popularity, this thesis focuses on the statistically-based approach to 3D shape descriptors (retrieval methods), which we extensively review in the next section.

Technically, the statistical approach to 3D shape descriptors is based on the distribution of local and/or global measurements of surface properties (i.e. features) of a 3D object, instead of directly using these measurements. This approach has the advantage of reducing the 3D shape similarity measurements to simple histograms' comparison. Several shape functions (i.e. surface properties measurement) can be utilized to calculate histograms. However, most of these functions are typically invariant to rotation and to translation. For example, considering the local and global approaches, the local approach, such as the curvature histogram, are able to identify various classes of objects, but are however not robust to noise, although the global approaches are inefficient in discriminating globally similar objects having minor fine details. In Section 2.6, we provide a detailed review of statistically-based 3D shape descriptors, which are directly related to our work in this thesis.

2.6 Statistically-based 3D Descriptors Review

As we already mentioned in Section 2.5.3, the statistically-based 3D shape descriptors are descriptors based on the distribution of local, global, or both measurements of surface properties (features) of a 3D object, instead of directly using these measurements. They may use single or multi-dimensional indexing (i.e. histogram), including tree-based approach, which is helpful for searching for similar objects from a large database. Extracted features from 3D surfaces are mostly presented as a 1-dimensional or multi-dimensional feature vectors. Observably, the statistically-based approach to shape descriptors have become the most popular in literature for describing 2D/3D objects, perhaps, due to their relative ease of computation and efficient indexing characteristics. For instance, five global signatures were computed directly from 3D meshes in [173, 174]. These signatures were expressed as shape distributions sampled from shape functions that measures the global geometric properties of 3D objects. The shape description approach by [173, 174] reduced the problem of 3D objects matching to the comparisons of their respective histogram distributions. Matching histogram distributions is easier than traditional shape matching methods which deal with issues such as pose registration, model estimation and finding feature correspondences [173]. Regarding what type of measurements can be derived from 3D surfaces for distribution, features, such as moments, Fourier coefficients, volume, etc., have been extracted from the surface of a mesh [272]. Another work by [66] combined two local features, the eccentricity function, which gives spatial shape information, and local-diameter function describing local patches around a shape, into a 2D histogram.

A statistical representation of 3-dimensional shape is introduced in [252]. In their work, they first defined 3D local reference frame (LRF) for each pair of oriented points (surflet-pairs) on the surface of a 3D mesh, and derived a 4-dimensional feature that parameterizes the intrinsic geometrical relation of these surflets. The

4-dimensional feature is binned into a multi-dimensional histogram, using 5-bins in each feature dimension to return a 625-dimensional feature vector as a final descriptor (signature) for a given 3D object. Inspired by this, the literature in [136] extended the 4-dimensional feature in [252] to 6-dimensions and obtained an even higher dimension of 15625-dimensional signature. This approach to 3D shape descriptor can be applied directly on 3D mesh representation according to [252] and point cloud [136]. However, their computation utilises all the vertices (mesh) and points (point cloud) models that they describe. This approach raises serious concerns about efficiency as a total of $k(k-1)/2$ features are returned from a surface of k surflets, resulting in remarkably high and redundant values for each iteration. In addition, the surflet-pairs based techniques in [252] and [136] return features which are invariant to rotation, scale, translation and robust to noisy data, but fail to deal with local surface dissimilarities and large-scale problems, while the Adaptive Hybrid Descriptor in [136] produces signature with excessively large dimensions, and employs all points on the surface of the shape it describes.

Two other local shape descriptors, the Point Feature Histogram (PFH) [201], and FPFH, developed by [198], also employs similar feature extraction techniques to that of [252]. Three common aspects between PFH and FPFH are: (i) Both methods present local shape descriptors, (ii) Both descriptors are computed for the entire surface points for point clouds only, (iii) Both PFH and FPFH descriptors are only suitable for 2.5D scans (Pseudo single position range images) [202]. The computational complexity of the PFH is $O(nk^2)$, where (nk^2) is the number of key points and neighbours. However, the FPFH extends PFH with some optimization steps, such that only direct surflet-pairs (p_i, n_i) between the query point and its neighbourhood are taken into consideration, negating the fully connected graph approach, i.e. $k(k-1)/2$ features used by other similar descriptors mentioned in this section [198]. This took the complexity of the PFH down to $O(nk)$, because while the PFH uses a fully-connected graph of $k(k-1)/2$ between all points, k in the neighbourhood of the key point, the FPFH considers only the direct connections between a given key point and its neighbouring points. In addition, the FPFH adopts only three of the 4-dimensional features proposed by [252] and used also in PFH. Generally, the feature extraction technique employed by [252, 136, 198], and [203] strictly depends on both $[x, y, z]$ coordinates of 3D surface and their corresponding normal vectors. For surface with noisy data, it is difficult to accurately estimate surface normals for its points. Although some robust methods have been proposed for accurate surface normals estimation for meshes and point cloud, the results of these methods vary with datasets. Intuitively, the smoother the surface of any given shape, the more accurate its normal vector estimation is likely to be.

2.6.1 Point Pair-based Statistical 3D Shape Descriptors

As already indicated in Sections 2.5 and 2.4, several approaches have been proposed in the literature for local and global 3D surface description. Although the literatures in [13] and [78] already provides comprehensive and up to date reviews of 3D shape descriptors, we would briefly review literatures regarding the 3D shape descriptors summarized in Table 2.1, in line with this thesis and our research focus.

S/N.	Descriptors (Year)	Size	Category	Use Case
1	SPRH (2003 [252])	625	Global	Mesh classification
2	PFH (2008 [200])	625or125	Local	Point cloud alignment
3	FPFH (2009 [198])	33	Local	2.5D Scans registration
4	PPF (2010 [51])	4	Hybrid	Point cloud recognition
5	ADH (2017 [136])	15,625	Local	Point cloud classification
6	PPFH (2018 [27])	512	Local	Point cloud matching

Table 2.1: Point Pair Features-Based 3D Shape Descriptors.

A global shape descriptor for 3D mesh classification is introduced by [252], where they computed the histogram of oriented point-pair relations, called the Surflet-Pairs Relation Histogram (SPRH), which describes the relative geometrical properties between two-points by a 4-dimensional feature. Similarly, another 3D local shape descriptor called the Adaptive Hybrid Shape Descriptor (ADH) was proposed by [136], which extends the 4-dimensional features of [252] to six dimensions. Similar to the features in [252], given two oriented points (p_i, n_i) and (p_j, n_j), the Point Pair Feature (PPF) [51] extracts four different simple relations from that of [252], thus: $F(p_i, p_j) = (\alpha, \beta, \gamma, \delta)$, where $\alpha = \angle(n_i, n_j)$, $\beta = \angle(n_i, d)$, $\gamma = \angle(n_j, d)$, $\delta = \|p_i - p_j\|$ and $d = p_i - p_j$. These features are extracted globally for all points in the point cloud and matched locally, thereafter for two shapes (object and scene) comparison, hence we consider the PPF a hybrid descriptor. In a more recent work by [27] only two features $F(p_i, p_j) = (\delta, \gamma)$ were adopted from PPF for their proposed Points Pair Feature Histogram (PPFH). A hybrid 3D descriptor known as the Ensemble of Shape Function (ESF) was formed by [263]. In order to combine the descriptive powers of both local and global descriptors, the work in [4] combined the Viewpoint Feature Histogram (VFH) global descriptor [201], and the FPFH local descriptor to form a hybrid (FPFH + VFH) descriptor. The above reviews certifies to the popularity and relevance of the point pair-based 3D retrieval method. Although, most of the work reported here are tested on a different kind of 3D data, such as the range scans or RGB-D data, the methods in this thesis completely dwells on the standard 3D triangular mesh and point cloud data.

The statistically-based methods to 3D descriptor consider pure geometry of 3D surface to represent extracted features of a given 3D object by a single descriptor, which is a fixed-sized or fixed-length n-dimensional feature-vector of values for all the 3D objects in the dataset. However, the size or length of the feature-vector (descriptor) could be as low as 1,024-dimension as in D2 global 3D descriptor [174] or very as high as 262,209-dimension as in HAPPS hybrid descriptor, described in Section 4.2.3), depending on the method/algorithm used to compute the descriptor. It is important to note however, that there is a trade-off between robustness and compactness considering the length of the final descriptor. We explain this in Chapter 4. In summary, we consider the 3D shape descriptor of any given object as a point in a high dimensional (i.e. n-dimensional) feature space. Therefore, any two objects are considered to be similar if they are close to each other in this space, using distance metric (see Section 2.2.3) for their comparison.

2.7 3D Shape Retrieval Challenges

In this section, we provide an overview of 3D shape retrieval challenges while in Sections 3.2.1 and 3.2.2, we further discussed how defective data or various abnormalities in datasets (such as holes, degeneracies, and duplicates in vertices, faces, and edges, etc.) affects shape retrieval algorithms due to certain factors: dataset representations, object's benchmarks design, data sources or capturing devices used to acquire the 3D models.

Deriving a shape signature that can capture the geometry and physical properties of a defective 3D object accurately and effectively is difficult. For example, how can the volume of a hollow 3D object (i.e. a 3D object with large missing parts), such as the objects found in PRoNTo dataset [142] be computed? considering that such a signature must be invariant to affine transformations, robust to noise, missing parts, and occlusion, etc. In addition, most 3D shapes in large databases, such as SHREC'14 [132] with 8987 3D objects, or PSB [191] with 1814 3D objects, are represented by un-organized sets of degenerate polygons, which are non-manifold. Other datasets, such as the SHREC'17 PRoNTo dataset [142] contains 3D shapes that are represented by point cloud with large part of missing surfaces and noise. Several of the available 3D objects are represented in a complex or rather challenging manner, often containing noise, holes and/or missing parts, disjointed, wrongly-oriented, self-intersecting and overlapping polygons. *Automatically*, repairing the degeneracies in these poorly represented 3D objects during shape analysis/processing (in order to produce manifold surfaces) remains a difficult research problem and sometime require *manual* human intervention to solve ambiguity [74, 10]. While recent work [45] has proposed automatic solution to address the holes on 3D surfaces from range scans (e.g., SHREC'17 Point cloud Retrieval of Non-rigid Toys (PRoNTo) dataset), however, their method does not generalize well for all 3D objects, especially, objects with different and overly complex topological representations, such as those type of 3D objects in [132].

Chapter 3

RESEARCH STRATEGY, TECHNIQUES, AND TOOLS

3.1 Introduction

This chapter provides in-depth research strategy, tools, and essential techniques (applicable to our proposed methodology and major contributions in Chapter 4, listed in Table 4.1, and described in Section 4.2). These techniques include: data pre-processing, point cloud sampling, feature extraction, 3D key points detection/determination. Strategies, such as shape descriptor construction, indexing, matching are also described, including the different types of dissimilarity metrics implemented in our work.

In line with the last paragraph of Section 2.2.4, although six aspects were mentioned, there are primarily four central (core) aspects to Content-Based Shape Retrieval (CBSR), which include: (i) Feature extraction, (ii) Shape description construction, (iii) Database indexing, and (iv) Shape Matching and retrieval. Preceding these aspects is another important (though optional) aspect, "Data Pre-processing", which is usually implemented to refine and present the input 3D shape(s) in a suitable format before the above-mentioned techniques are implemented. Each of these research strategies and techniques are explained in the sections that follow.

3.2 Data Pre-processing

In order to bring all database objects into a concise form and deal with issues (i.e. scales, translation, noise, varying surface details, degeneracies, etc.) that may occur with different objects, it is often necessary to apply data pre-processing functions to the objects. Determining which pre-processing function is applicable depends on the type and/or format of the input data presented to the shape analysis algorithms (such as those described in Section 2.2.1). For example, the input 3D objects may either be a polygonal mesh or point cloud, and could also be represented in different scale, translation, noisy, faulty, or with varying surface details. Therefore, we have developed and implemented several pre-processing algorithms, including adopting some from existing research for our shape retrieval methods, including: (i) point cloud sampling, (ii) affine transformations (scaling, translation, rotation), and (iii)

faulty 3D models handling (noise removal, surface smoothing, hole filling, degenerate, and duplicate vertices and/or faces removal, etc.) for mesh and point cloud data, whenever the need arises.

3.2.1 Defective Data

Intuitively, defects in the data structure of 3D object representation would most likely lead to inaccurate shape description of the object. In addition, key processes such as surface points sampling (see Section 3.2.3), and surface normal estimation (see Section 3.3.1) are likely to be affected by defective/faulty 3D objects, resulting in undesirable outcomes. The fundamental computational principle of GIGO (*garbage-in, garbage-out*) explains this hypothesis. Algorithms and methods designed to operate on vertices, faces, edges, or other properties of a 3D mesh expects consistent ordering in the structures of the input data they receive. When the orderings are inconsistent, altered or have data with incomplete details, duplicates, etc., several issues listed below erupts, thus:

i. The algorithm may fail and return error,

ii. The algorithm may run and return incorrect, undesirable output,

iii. Subsequent methods using the undesirable output eventually produces undesirable final output, etc.

3.2.2 Defective Data Handling

In view of the challenges posed by defective data, the need for data pre-processing prior to features extraction, therefore, cannot be overemphasized, especially for 3D surfaces which are more complicated to deal with compared to 2D images. Several different data pre-processing functions can be applied on 3D surfaces, such as surface triangulation, surface smoothing, hole filling, re-meshing, surface re-sampling (up-sampling or down-sampling), artefacts removal (removal of degeneracies in vertices or faces of mesh), etc. Depending on the application, some or all these pre-processing functions may be needed. In Section 2.2.2, we described two categories of 3D surfaces: the water-tight and the non-water-tight 3D surfaces. Usually, the latter is considered to be defective. Some 3D benchmark dataset, such as SHREC'12 (see Section 5.2.4, Figure 5.3) and PSB [191] contains a combination of defective (non-water-tight) and non-defective(watertight) 3D objects. Therefore, considering such datasets, not all the above-mentioned pre-processing functions may be needed for all the data it contains.

Developing an intelligent application that automatically detects which objects in the dataset are defective or not is incredibly challenging. For example, consider a heterogeneous dataset (i.e. benchmark dataset containing mixed variety of 3D shapes, each having different levels of defects or no defect), some of the data may require just one of the pre-processing steps to be performed on them while others may require two or more pre-processing steps. In our implementation, we develop function(s) that includes the implementation of some particularly useful operations, such as smoothing, hole filing, etc. Unfortunately, the smoothing function would

over-smooth an already smooth 3D object, for example. In addition, especially for larger heterogeneous datasets like the SHREC'12 [131] and SHREC'14 [132] datasets, which contain variety of mixed models with different pre-processing needs, the challenge remains that, it is exceedingly difficult, frustrating, and tedious task to loop through each of these data in turn to correct their respective defects.

Considering the above, a one-solution-fits-all technique that solves this kind of open research problem is yet to be developed. However, in this thesis, the approach we adopt to address the pre-processing needs of large heterogeneous dataset, such as the ones mentioned here above is outlined as follows:

i. first apply our features extraction method automatically on all the raw datasets. Defective files were identified as they returned errors. They were either manually corrected, repaired, or noted in the case of minor defects,

ii. following the above steps, all 3D objects with common pre-processing needs were identified for the entire dataset, and preprocessing algorithms, like smoothing and removal of degeneracies, applied automatically on the whole objects in the dataset,

iii. those shapes whose defects needed some complicated pre-processing steps that could not be resolved via the automatic process, were selectively dealt with, in order to correct the defects. For instance, a particular 3D mesh file in the SHREC'14 dataset contains two separate objects (a 3D object and disjointed 3D text) as a single object. As a result, point sampling feature extraction completely failed to apply on this file because the algorithm expects every file to include data structure for a single object, not multiple,

iv. in another investigation, we decided to completely spot and remove all defective 3D meshes from the SHREC'12 dataset, re-configure its corresponding ground-truth file accordingly and re-evaluated our retrieval method on the remaining supposedly better meshes. The results (not included in this thesis) reveal tremendous performance improvements, evaluating our method on only those 3D objects without defects i.e. from 1,200 models to about 1,068 models. Unfortunately, we did not investigate further with this approach due to the need to also implement other state-of-the-art methods (whose code was not given) using the new Ground Truth (GT) of 1,068 models, and be able to compare our method.

Unfortunately, there is still a problem with approaches (i) to (iii) to addressing defects in a benchmark data, because the pre-processing needs for a particular set of data in the dataset may adversely affect other set of data without such need, and vice-versa. For example, if a dataset contains a 3D shape - A, with *smooth surface*, and another shape - B, with very *"rough surface"*, automatically applying smoothing filter on the entire database (in attempt to correct those shapes whose surfaces are rough), would fix shape B, while completely altering the overall topology and structure of shape A due to unnecessary smoothing of shape A, which eventually results to over-smoothing of shape A.

The only unambiguous way to resolve the above problem would be to selectively apply smoothing to B instead of doing so automatically on all database models, but the challenge, again, is that for large datasets of say 1000^+ models, this approach would make the pre-processing task nearly impossible. Therefore, this remains one key research challenge in 3D shape retrieval for large and heterogeneous datasets.

The overall goal of data pre-processing is to ensure that different 3D shapes are fairly comparable, by bringing them into a concise form through the application of similar transformations across all database objects. Considering that 3D shape retrieval involves two phases (i.e. online and offline [228]), during features extraction and shape descriptor construction, our overall shape descriptor method is developed to automatically take care of above-mentioned pre-processing steps both for the offline (database indexing phase, described in Section 3.5) and online (query shape) feature extraction and descriptor construction phase. Secondly, we understand that our retrieval method can be applied to various 3D datasets, and each of these datasets may contain 3D models with different representations: polygon mesh or point cloud representation, with polygon mesh being the most common (see Section 2.2.1), we design our retrieval method to handle pre-processing for these two data representations accordingly. Therefore, we would consider the process of point cloud sampling from 3D triangular mesh next.

3.2.3 Point Cloud Sampling

The shape retrieval algorithms we implement in this research are applicable to both 3D triangular meshes and 3D point clouds, but not other formats of 3D object representations (discussed in Section 2.2.1). However, it is possible to convert from one representation to another depending on the need. Given a triangular 3D mesh as input, vertices of the mesh could be used directly as point cloud without the connectivity information (edges) that makes up the faces of the mesh. Alternatively, points can be sampled from every triangular faces of the input 3D mesh to form point cloud. The later approach is better illustrated in Figures 3.1 and 3.8. While triangular meshes are quite convenient for many computer vision and/or graphic tasks, there are a number of reasons why extracting features and constructing descriptors from 3D point clouds is rather preferable to their mesh representation, including that point clouds are ideal for processing and extracting information from 3D objects. Secondly, while the complicated objects may be less well represented than the simple ones, we agree that the number of points sampled should be determined by the complexity of the objects, the number of triangles for example. However, for a given dataset, we are interested in the same number of points for all its available objects. We assume that having fixed number of points sample (evenly dispersed over shape's surfaces) for every database shape has the advantage of a stable probability distribution for all the shapes.

For the majority of our research implementations and experimental evaluations in this thesis, given a triangular mesh as input, we first sample N points from the mesh to form a point cloud P, using the points sampling technique described by [173], and implemented in the Trimesh library [47]. We used $N = 4200$, for example, in the SHREC'18 and SHREC'10 datasets, while the PRoNTo dataset is

originally presented as an un-organised point cloud data with $[x, y, z]$ coordinates that contains approximately $N3000_{min}$ to 4200_{max} points. Further details regarding these datasets and several others used to evaluate our shape retrieval methods in this thesis are adequately described in Section 5.2.

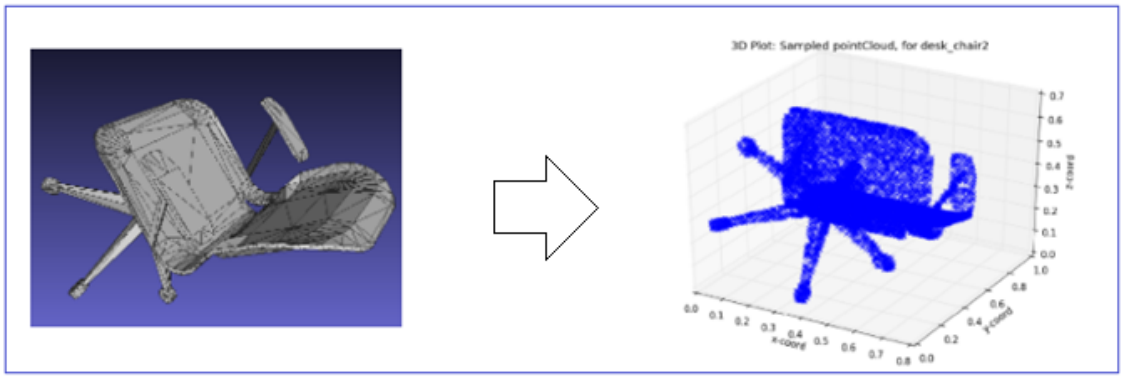


Figure 3.1: 3D mesh (left) to 3D Points Cloud (right).

The research contribution by [173] described a Monte Carlo sampling method that loops through all triangular faces for a given 3D mesh, and for each triangle that makes up a face, random points, S_f are generated *w.r.t.* the area of the triangle. Collectively, the points, P_f for all triangles making up the mesh's surface are returned as point cloud, P generated for the input 3D shape. This technique can be successfully implemented using the open-source utility in the Trimesh library [47]. The surface points sampling techniques by [173] has the advantage of generating unbiased random points, sampled evenly according to the area of the individual faces of the triangular mesh from which the points are sampled – therefore larger faces are sampled more often and hold more point samples along their surface than smaller faces. The pseudo-code for this random point sampling techniques as in [173] is outlined below:

Step 1: Given a triangular face with vertices A, B and C, compute the area for each face in a mesh and store into an array.

Step 2: Store the cumulative area of triangular faces visited, into an array.

Step 3: Select a face (at random), with probability proportional to its area. To do this, a random number between 0 and the total area sum (cumulative area) is generated, and a binary search is performed on the array of cumulative areas, to select a number (value), which corresponds to one in the stored (accumulated) area.

Step 4: For each selected triangle (face) in Step 3., with vertices A, B and C, two random numbers r_1 and r_2, between 0 and 1, are generated. Finally, point-coordinates are constructed on the surface of this selected face by evaluating equation 3.1. Overall, uniform random points are produced on the surface of a mesh, *w.r.t.* the surface area of each faces, by taking the square-root of r_1, see Figure 3.2.

$$P = (1 - \sqrt{r_1})A + \sqrt{r_1}(1 - r_2)B + \sqrt{r_1}r_2C \tag{3.1}$$

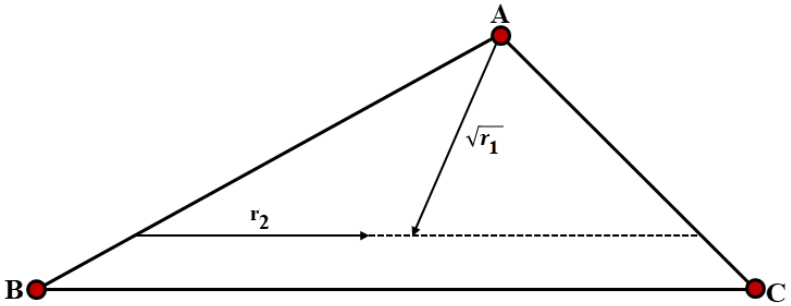


Figure 3.2: Random Points sampling in triangle, [173].

Mesh to Point Cloud Sampling: Barycentric Interpolation Technique

Assuming that the complex surface of a 3D model is represented by a triangulated approximation defined by three non-collinear vertices: A, B, and C (see Figure 3.3a), any point, $p \in \triangle A, B, C$ is on the line segment between one vertex and some other point p' on the opposite edge (see Figure 3.3b). The ability to represent the position of any such point, p (located within the bounds of $\triangle A, B, C$) with three real numbers (i.e. scalars), α, β, and γ is particularly important in Computer Graphics. Barycentric coordinates can be used to express this position according to Equation (3.2), where α, β, and γ are the three scalars representing the barycentric coordinates of p, such that $\alpha + \beta + \gamma = 1$, for $p \in \triangle A, B, C$ (i.e. normalized barycentric coordinates). Alternatively, $\gamma = 1 - \alpha - \beta$ and $\alpha + \beta \leq 1$ [211].

$$p = \alpha A + \beta B + \gamma C. \tag{3.2}$$

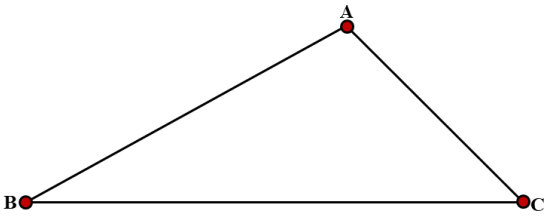
- p is within the bounds of the triangular face, $\langle A, B, C \rangle$ if and only if:
  $0 \leq \alpha \leq 1$,
  $0 \leq \beta \leq 1$,
  $0 \leq \gamma \leq 1$.

- p is outside the bounds of $\langle A, B, C \rangle$ if any of the coordinates is less than zero or greater than one:
  $\alpha < 0$ or $\alpha > 1$,
  $\beta < 0$ or $\beta > 1$,
  $\gamma < 0$ or $\gamma > 1$.

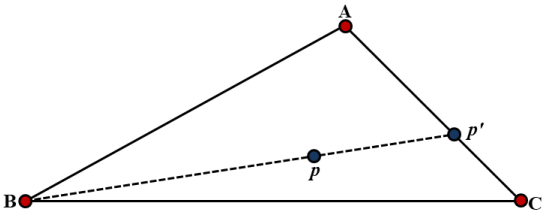
- p is on an edge (i.e. on one of the lines joining the vertices of the triangle) if any of the coordinates is zero. Which line/edge it is depends on which one of the coordinates is zero.

- p is on either of vertices, A, B, or C if any two of the coordinates are zero. Which vertex exactly depends on which two coordinates are zero.

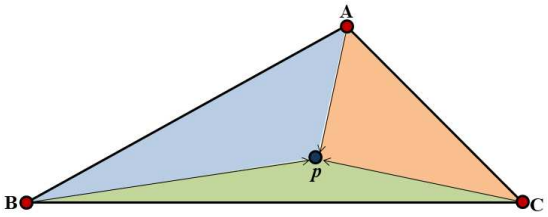
The barycentric coordinates, $[\alpha, \beta, \gamma]$ of a point, say p, can be used as a weighting factor for properties (i.e. normals, colours, and texture coordinates) of the vertices, A, B, and C of the triangle bounding p [210]. Similarly, $[\alpha, \beta, \gamma]$ allows p to be expressed as a weighted average of the vertices of $\triangle A, B, C$. That is, they can be expressed as the area of sub-triangles $\triangle BCp$ (denoted by α), $\triangle CAp$ (denoted by β)



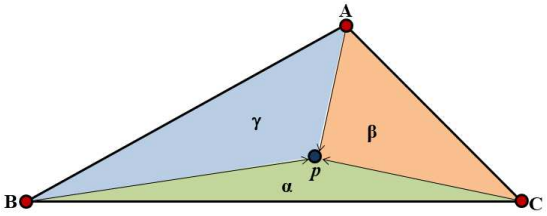
(a) Typical triangular face ($\triangle A, B, C$) defined by three non-collinear vertices: A, B, and C.



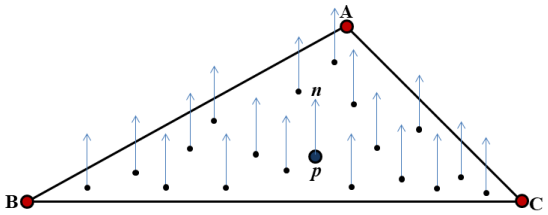
(b) Random point, p within $\triangle A, B, C$ is on a line segment between point B and point p' on the edge, $\bar{CA}$.



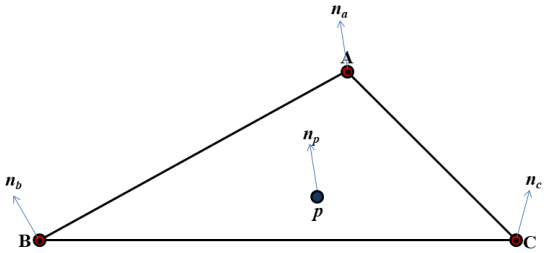
(c) Point, p expressed as a weighted average of 3 sub faces, $\triangle BCp$ (green), $\triangle CAp$ (red), and $\triangle ABp$ (blue) within $\triangle ABC$.



(d) Each sub-triangular face is represented by barycentric coordinate (i.e. areal coordinate), α (green), β (red), and γ (blue)



(e) Associating the same normal, n obtained at point, p within a triangular face, $\triangle ABC$ to every other points on the same surface/face.



(f) Linearly interpolated normal, n_p at random point, p using normals, n_a, n_b, and n_c of the three vertices, A, B, and C of $\triangle A, B, C$.

Figure 3.3: Barycentric coordinate system technique to points, P_s and normals, N_s sampling from a triangular face, $\triangle A, B, C$.

and $\triangle ABp$ (denoted by γ) divided by the overall area of $\triangle ABC$ (see Figure 3.3d). For this reason, they are called areal coordinates. In computer graphics, barycentric coordinates are used for *Ray-Tracing* (to test for intersection), as well as *Rendering* (to interpolate triangular face information). However, in 3D models, triangular face information (i.e. normals, color, and texture coordinates) is often associated with vertices rather than the triangular face itself. In our work, we are particularly interested in only the normals, n_p to every point, p sampled from within $\triangle ABC$.

The same normal, n could be associated to every point on the face of $\triangle ABC$ using the formula in Equation (3.3) - see also Figure 3.3e, but that is a bad idea and would give rise to what is known as flat shading [17] in computer graphics. Theoretically, if the barycentric coordinates are used to compute the position of a point located on the triangle using the triangle vertices, then any other data defined at the triangle's vertices (such as normals, colours etc.) can also be interpolated in exactly the same way [211]. Therefore, the normal, n_p at point, p within $\triangle ABC$ could be expressed as the linear interpolation of the normals at the vertices of $\triangle ABC$. This interpolation is given by Equation (3.4) and visualised in Figure 3.3f.

$$n = \frac{(B - A) \times (C - A)}{\|(B - A) \times (C - A)\|} \tag{3.3}$$

$$n_p = \frac{\alpha(p \times n_a) + \beta(p \times n_b) + \gamma(p \times n_c)}{\|\alpha(p \times n_a) + \beta(p \times n_b) + \gamma(p \times n_c)\|} \tag{3.4}$$

$$\frac{1}{2}\|(\mathbf{A} - \mathbf{C}) \times (\mathbf{B} - \mathbf{C})\|_2 \tag{3.5}$$

$$\begin{aligned} \alpha &= \frac{Area\ of\ \triangle BCp}{Area\ of\ \triangle ABC} \\ \beta &= \frac{Area\ of\ \triangle CAp}{Area\ of\ \triangle ABC} \\ \gamma &= \frac{Area\ of\ \triangle ABp}{Area\ of\ \triangle ABC} \end{aligned} \tag{3.6}$$

The area of the triangular face, $\triangle ABC$ can be computed as given by Equation (3.5), while the area of each of the sub-triangles (represented by α, β, and γ) within $\triangle ABC$ can be computed as given in Equation (3.6). Finally, for every new point, p_i that is randomly generated using the barycentric coordinate system technique, the areas of three sub-triangles recomputed and a corresponding normal vector, n_i is instantly derived. In order to obtain the number of samples for each triangular face (i.e. $\triangle ABC$), all the triangle areas are first summed up and converted to a probability distribution. Multiplying the number of points sample, N by the distribution gives the number of samples per face.

Defective 3D Objects Processing

Generating (sampling) points from 3D mesh to form point cloud produces a "cloud" of points that depicts the input 3D mesh - see Figures (3.1) and (3.8). Intuitively, the robustness of any final shape descriptor or feature computed directly depends

on the surface quality of input objects. For example, when the mesh is deformed (i.e. have holes (non-watertight), duplicate faces, edges, and vertices, etc.), the defects would also be reflected in the sampled point cloud thereby affecting subsequent features extracted from such mesh or point cloud. To deal with the deformations in the input data, we first apply some pre-processing functions such as hole filling, smoothing, and cleaning to remove degeneracies and duplicates, etc., to clean and/or repair the faulty triangular meshes, using existing software library, codes or packages [223]. Essentially, the reader is referred to Section 3.2.2, for a more detailed approach regarding how defective 3D mesh/data have been handled for all the implementations in this thesis.

3.2.4 Affine Transformation (Scaling, Rotation, and Translation)

The shape of an object is the geometric information that remains after the effects of affine transformations (i.e. rotation, scaling/rescaling, and translation) has been removed (factored out) from the object [104, 275]. We illustrate this concept in Figure 3.4. Therefore, for two shape descriptors to be comparable, affine transformations must be applied to each of their objects to ensure that the final descriptors constructed from these objects are invariant to affine 3D transformations. Scaling and translation are easy to deal with as described below, but rotation invariance is difficult. However, we ensure that rotation invariance is wrapped into our respective shape descriptor implementation for each 3D object (see Section 4.2).

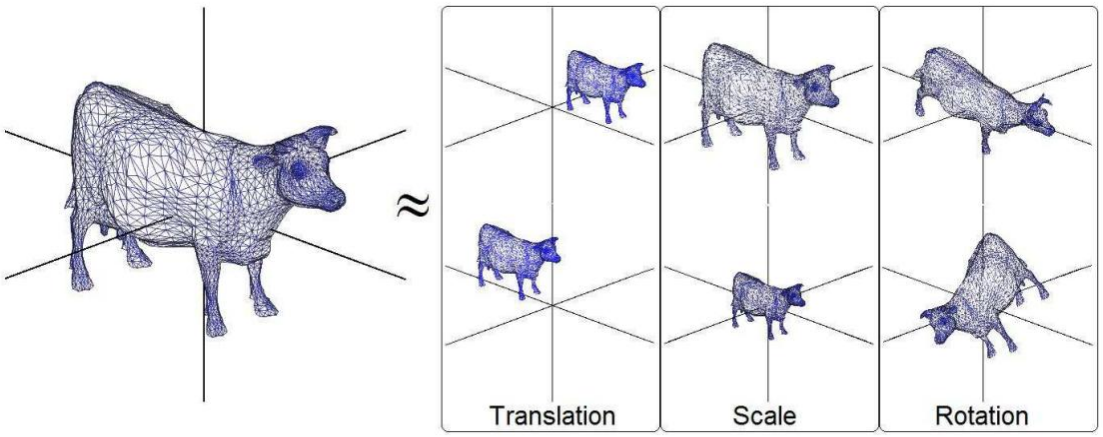


Figure 3.4: 3D model of a cow with different Euclidean similarity transformations (locations, scales, and rotations) but the same shape. Image source: [102]

In order to ensure that all final descriptors computed from shapes are translation invariant, each input shape (point cloud) is centred on its centroid. That is, for a point cloud $P = p_i(i = 1 : N)$, with N points and centroid,

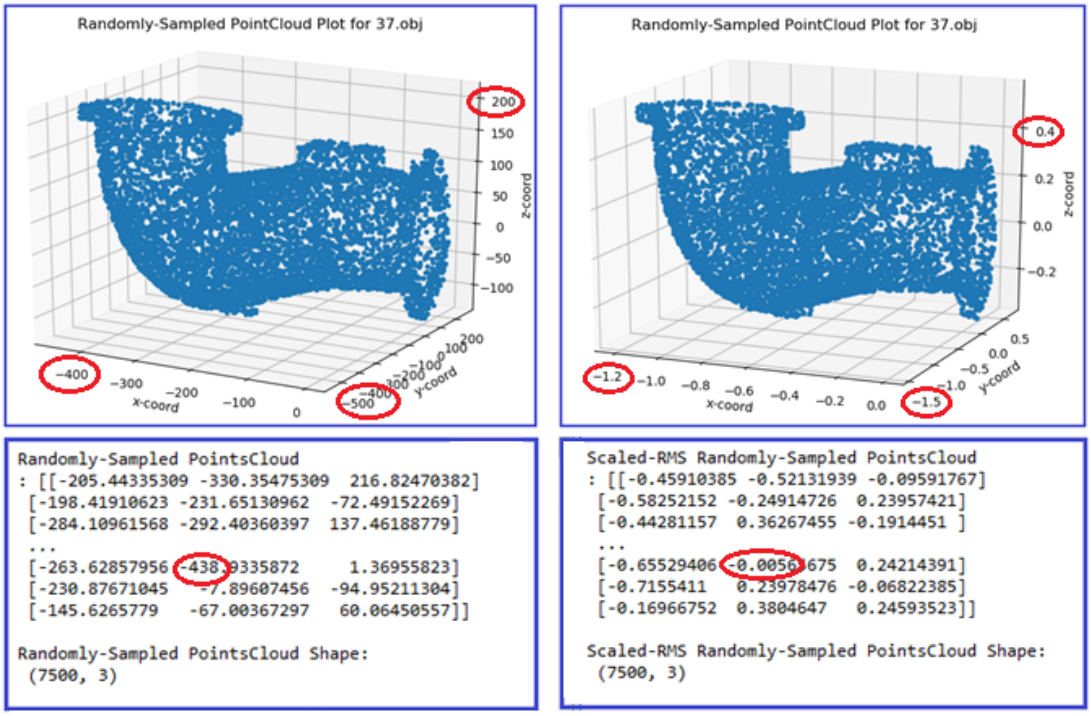
$$p_c = \frac{1}{N}\sum_{i=1}^{N} p_i$$

we apply a translation-invariant transform to P thus: $P = P - p_c$. Likewise for scale invariance, a uniform scale, S is applied to each point (of the point cloud)

in all directions such that the Root Mean Square (RMS) distances of each of these points to its centroid is 1. Mathematically, for a point cloud $P = p_i(i = 1 : N)$ with N points, S is applied such that

$$\sqrt{\frac{1}{N}\sum_{i=1}^{N}||S_{p_i}||} = 1$$

For point cloud with normals, the estimated *normals* are not affected by the scaling and translation transforms. We show practical implementations (visualizations) of the above-mentioned affine transformations on 3D rigid point cloud shape in Figure 3.5. In Figure 3.5a, we observe the coordinates of the object without affine scaling ranged between $[0, -400]$, $[200, -500]$, and $[200, -100]$ for the $x-coordinate$, $y-coordinate$, and $z-coordinate$, respectively, while in Figure 3.5b, all the three coordinates ranged between $[-1, 1]$ after RMS-scaling transform has been applied to it.



(a) Unscaled point cloud. (b) Scaled point cloud.

Figure 3.5: Visualisation of the 3D point cloud representation of a rigid 3D pipe object. Figure 3.5a shows the coordinates of this object with very large values, while Figure 3.5b shows the same object after being scaled using the scaling factor mentioned above.

3.3 Feature Extraction

Features must first be extracted from 2D/3D objects to compute shape signatures or descriptors for such objects. The extracted features and computed descriptors

have greater impact on the overall performance outcome of application (like retrieval, classification, recognition, detection etc.) involving them. For this reason, the process of features extraction and shape descriptors computation are considerably very important for any application involving 2D or 3D data. Following features extraction processes is shape descriptor computation, where a compact mathematical representation (descriptor or signature) of extracted features is computed for a given shape/object. Details about this are presented in Section 3.4.

Secondly, in computing descriptors for 3D shapes, algorithms do not use the input 3D data as they are; instead, low, or middle-level features are first extracted from the input data and used for constructing the shape descriptors. For example, geometric features such as Gaussian curvature, mean curvature [137], and Shape Index (SI) [110], etc., which constitute local feature extraction have been used in the literature. Experiments have also shown that constructing shape descriptors from extracted features leads to robust shape retrieval methods compared with using the original 3D data features as input. This is because, unlike 2D image data which could directly be used by machine learning algorithms [24] in the case of data-driven approach, for example, the raw 3D data are not rich in features. Feature extraction, therefore, is an important process and one of the key preliminary aspects of content-based retrieval.

Finally, most 3D shape descriptors, including the ones we implement in this research work requires that some sort of features (such as surface normals, local surface characteristics (deformations), or surface geometry measurements) are extracted first. In this section, we provide details of useful features and their extraction techniques - which are most relevant and are implemented in our research work.

Definition 3.3.1 ***k*-Nearest Neighbours (*k*-NN) Algorithm :** *This is the most-commonly used technique for several applications, including classification and clustering. Basically, it is a technique used to get all k closest/nearest points or nodes to a given point of interest, by finding the spatial distance between these points in the most efficient way. The number of nearest neighbours (i.e. optimal value of k) highly depends on the data used. The k-NN algorithm returns the "index" and "distance" of each of the derived neighbouring points to the interest point, where (for the implementations in this thesis) these "indices" are used to derive the actual k points, for the next phase of our implementation (performing Covariance analysis or fitting a plane to derived k points, for Principal Component Analysis (PCA)-based surface normal estimation in point cloud).*

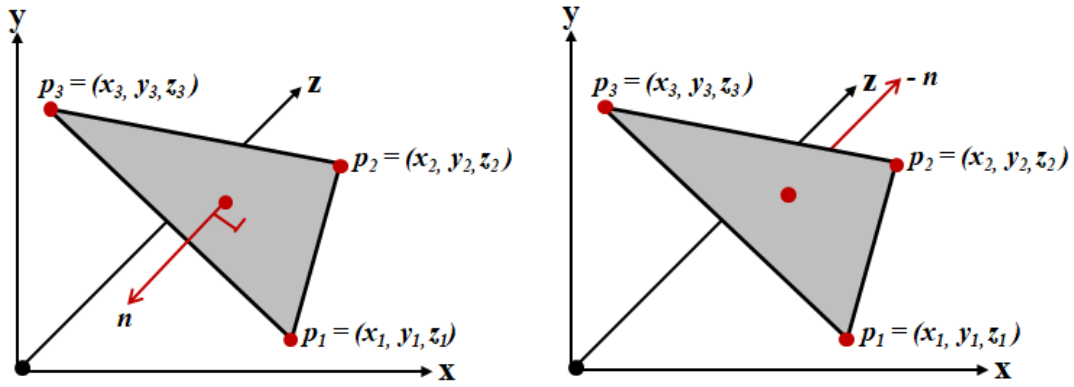
Definition 3.3.2 ***r*-Nearest Neighbours (*r*-NN) Algorithm :** *This algorithm is used to find all neighbouring points or nodes within a sphere of radius, r to a given point or point of interest, and returns the "index" and "distance" of each of the derived neighbouring points to the interest point, where (for the implementations in this thesis) these "indices" are used to derive the actual k points, for the next phase of our implementation (performing Covariance analysis or fitting a plane to derived k points, for PCA-based surface normal estimation in point cloud and selection of LSP for our proposed APPFD method).*

In line with the above definitions for the k/r-NN algorithms, Scikit-Learn [125], a scientific Python library implements two different nearest neighbors classifiers:

"KNeighborsClassifier", which implements learning based on the k nearest neighbours of each query point, where k is an integer value specified by the user, and *"RadiusNeighboursClassifier"*, which implements learning based on the number of neighbours within a fixed radius, r of each training point, where r is a floating-point value specified by the user. We adopt these implementations for this thesis.

3.3.1 3D Surface Normals Estimation

One fundamental feature of any 3D surface is the unit normal vector (i.e. surface normals), from which many other geometric features, physical surface measurements and final shape descriptors are possibly computed. Surface normals can be estimated for mesh, as well as point cloud representations of any given 3D object. Each surface normal (normal vector), n corresponds to a respective surface point, n of a given 3D mesh or point cloud object. In addition, several other features or algorithms for feature extraction, object recognition, segmentation, and registration, etc., strictly rely on the surface normals. However, the sampling method in [173, 47] only generates point cloud from mesh, and requires additional steps to estimate the "normal" to each point in the generated point cloud, which is computationally very expensive and slow. In order to address this concern in our implementations, we modified the code in [47] to compute the surface normal simultaneously also, n_i for each sampled point, p_i during the sampling phase, using a computer graphic Barycentric Interpolation (BI) technique fully described by [190] and NumPy broadcasting capability for faster computation. Further details regarding our implementation can be found in Section 3.3.4 - "Improved Surface Normal Estimation Technique".



(a) Outward-pointing Face Normal (b) Inward-pointing Face Normal

Figure 3.6: 3D triangular face with vertices i.e. points (p_1, p_2, p_3) and normal n. In (a), the *normal vector* is pointing in the right direction (outward), while in (b), the *normal vector* is pointing in the opposite direction (inward), which is incorrect.

Surface Normals Estimation for 3D Mesh

Depending on the shape descriptor to be computed, surface normals can be estimated for either vertices or faces of a triangular mesh. Given a 3D triangular mesh with vertices and faces data, the "unit normal vectors" pointing outwards from each of the vertices (i.e. vertex normals) or faces (i.e. face normals) of the shape can be

derived in a number of ways. However, the techniques to normal vector estimation for vertices and faces of a triangular mesh are different. For example, if we consider three points (p_1, p_2, p_3) that makes up a triangular face as illustrated in Figure 3.6, taking the cross product of the edges (i.e. vectors), $[p_1 - p_3]$ and $[p_1 - p_2]$ returns a vector for that face, which can be normalized to form its unit normal vector (i.e. face normal).

Alternatively, to derive a normal vector for any given vertex, say p_1, as illustrated in Figure 3.7, a common method would be to take the average of all normals to the faces associated with p_1. For example, Figure 3.7 shows a typical triangular mesh patch with five faces ($N_{face1}, N_{face2}, N_{face3}, N_{face4}$, and N_{face5}). The *normal* vector at vertex p_1 (N_{vertex}), can be derived as shown in Equation (3.7). For more details on vertex normal estimation from triangular meshes, refer to [232], and [94].

$$N_{vertex} = \frac{N_{face1} + N_{face2} + N_{face3} + N_{face4} + N_{face5}}{5.0}. \tag{3.7}$$

In summary, consider any triangular face, say in Figure 3.7, with vertices (p_1, p_2, p_3) as in Figure 3.6, computing its unit normal vector n is remarkably simple and straightforward. n would be the vector cross product of vectors $(p_1 - p_3)$ and $(p_1 - p_2)$, where p_1, p_2 and p_3 are points in R^3 space . The result is then normalized to unit vector. However, if the triangle has zero area, the result is invalid due to an improperly defined normal. This formulation is illustrated in Figure 3.6, and in order to achieve the result in Figure 3.6a, with outward-pointing *normal vector*, the vertices (p_1, p_2, p_3) of the triangle must be ordered anti-clockwise, else, the result in Figure 3.6b is attained.

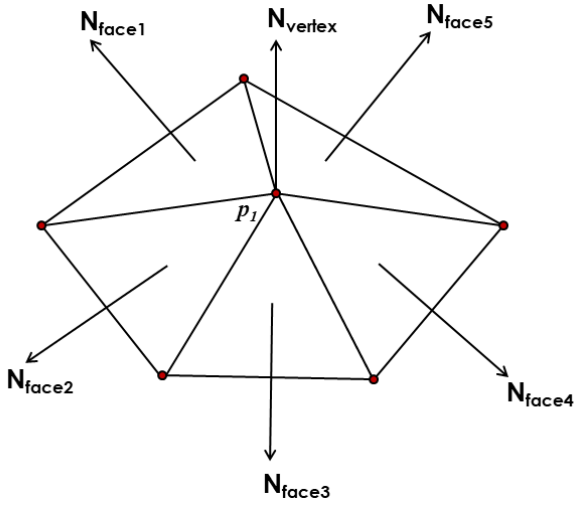


Figure 3.7: Normal Vector for Vertex and Faces of a triangular mesh.

Surface Normals Estimation for 3D Point Cloud

Point cloud has become the most preferred type of data for algorithms involving surface normals, as opposed to just using mesh vertices and their associated normal vectors [27, 82, 147, 200, 225, 106]. Most datasets are presented as raw point

cloud data without their corresponding surface normals information. In most instances where the datasets are given as triangular meshes, point clouds could also be estimated using surface points sampling technique described in Section 3.2, as illustrated in Figure 3.8. In any of these cases, there is the need to estimate surface normals for these point clouds, using existing techniques, since the majority of the feature extraction techniques in our research implementations completely rely on the unit normal vectors (surface normals) of the given shapes.

Similar to 3D mesh vertices, unit normal vectors can also be estimated for all points of a 3D point cloud using any of various existing techniques/methods for point cloud normals estimation, such as the commonly used methods described in [133, 159, 87]. However, it is largely unclear which of these methods are preferable for which application as there is the trade-off between quality and speed [109].

The most common approach to normal vector estimation for point cloud is using principal component analysis (PCA) on the covariance matrix obtained from $k - neighbourhood$ or $r - neighbourhood$ of points to the interest point p_i. The same approach is presented in [199, 212]. Generally, the PCA-based normal vectors estimation is faced with ambiguous orientation of the estimated normals as shown in see Figure 3.11b, and this remains an open problem till date. According to [212], "a key issue is determining the size of the region over which a normal vector is computed. A common technique involves taking all the points lying within a sphere of radius r, centred on the query point p_i. Unfortunately, such heuristic would require CPU expensive queries on a KD-Tree to determine the points inside the sphere".

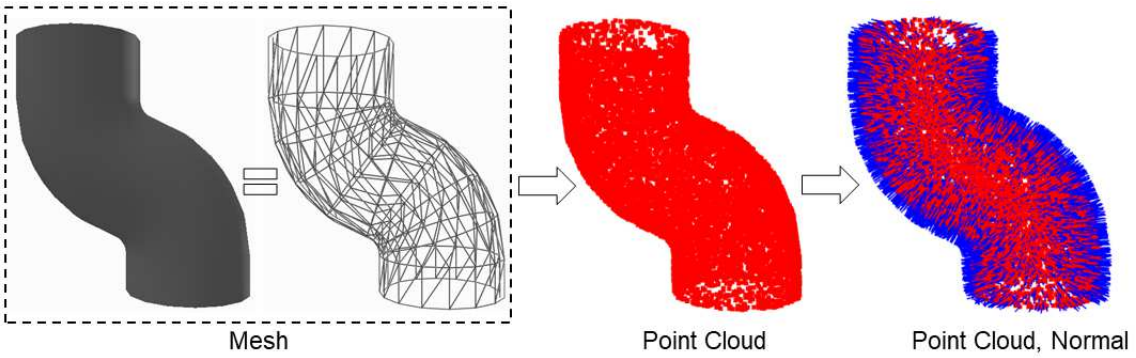


Figure 3.8: Mesh to point cloud and normals.

In most cases, the estimated surface normals using PCA-based approach produces results with inaccuracies. For example, in Figure 3.11b, i-iii, we visualize the point cloud models of a Pipe with inconsistent and incorrectly oriented normal vectors. Figure 3.11b, ii shows a point cloud model of a Human-head with almost all of its normal vectors pointing inwards. However, one solution to the orientation problem, as suggested by [204] is to redirect each surface normal to be consistent with majority of normals' orientation within the specified $k - neighbourhood$. Another suggestion by [201], is using viewpoint information, if it is known. Unfortunately, nearly all point cloud dataset are without viewpoint information and the point cloud sampled from 3D meshes lacks same as well, although, if $p\dot{n} < 0$, n is pointing outward, and If $p\dot{n} > 0$, n should be flipped, i.e. $n = -n$. On a manifold 3D surface,

which encloses a complete volume, there seems to be little or no issue with this orientation check, but on a non-manifold surface, this final checking for the orientation or direction of the normal vector seems ambiguous, resulting to inconsistent normal orientation across the whole 3D surface.

3.3.2 Improved Surface Normals Estimation for 3D Point Cloud

Unfortunately, correct and computationally efficient way to estimate surface normals from raw point cloud data is still an open research challenge. For retrieval algorithms that depend on point cloud data and their corresponding surface normals but are provided with 3D meshes, we described the possibility, ease, and accuracy of sampling points and their corresponding normals from 3D mesh for such algorithm. However, some datasets, such as the SHREC'17 PRoNTo dataset (see Section 5.2) present only raw point cloud data as input and the retrieval algorithm would have to estimate corresponding surface normals for such point cloud objects. The conventional approach to achieving this is the use of a PCA-based technique mentioned previously (also, see Algorithm 1), which is computationally expensive, among other issues. An alternative approach would be to first convert the input point cloud data to triangular mesh and then apply the Barycentric approach to point sampling on the 3D mesh (see Section 3.2.3), but converting point cloud to mesh and back to point cloud and normals is not without cost.

We note here, that Algorithm 1 is adapted from [201]. However, considering that the dataset source in [201], which provides their algorithm with view point coordinate, is different from ours, which does not provide view point coordinate, we implement a slightly different technique which checks if the direction of the *"eigen-vector"* with the least *"eigen-value"* is less/greater than zero to negate the result, as seen in *line 10* of our Algorithm 1, for outward pointing normal.

Fortunately, in this research, the majority of our datasets (see Section 5.2) were presented as triangular mesh and not point cloud, except for SHREC'17 PRoNTo dataset. Therefore, we did not have to deal much with surface normals estimation directly from point cloud. However, since the number of points for each 3D object in the SHREC'17 dataset were below 5,000 (i.e. between $3,000$ to $4,500$), we simply implemented the PCA-based technique to estimate their surface normals for our algorithm. In order to ensure an efficient computation and accurate point cloud normal estimation (as shown in Figure 3.11a), we selected a reasonable parameter for the size of r or k neighbours around each point for which normal we are estimating. The process to accomplish this are outlined in Algorithm 1 and summarised below.

Let p_i be a point in point cloud, P for which we want to compute a unit normal vector. Let p_k be the local neighbourhood of K points selected around p_i using k/r-NN (nearest neighbour) algorithm. We then fit a plane to p_k using Covariance analysis (i.e. PCA-based technique), and return the eigen-vector with the least eigen-value as the "normal vector" to the point of interest, p_i. Also see [176, 205]. These steps are repeated for all $p_i \in P$, according to Algorithm 1. Finally, a check is performed to ensure that the vectors are pointing outward (mostly needed for

Algorithm 1: Normal Estimation for 3D Point Cloud (adapted from [201])

1: **INPUT:** Point Cloud, $P_s \subset \{p_i,\ i = 1...N\}$; Parameters, $r = 0.04$ or $k = 11$ which determines the size of p_i's neighbourhood, P_i for Covariance analysis on p_i. Where $p_i \in P_s$ is the interest point for which normal vector is to be computed, and $N \approx 4,500$, the total number of points in P_s.

2: **OUTPUT:** Normal vector, $N_s \subset \{n_i,\ i = 1...N\}$ for point cloud, P_s

Estimate normals.

3: $[Ps, Ns]$ = normalEstrimation$(P_s, r = 0.15, k = 15)$

Initialise empty array for $[P_s,\ N_s]$.

4: $[P_s, N_s] = [\ ]$

Loop through all $p_i \in P_s$.

5: **for all** p_i in P_s **do**

6: extract patch, P_i, using k/r-NN

Compute Covariance Matrix (M) for P_i.

7: $M_{P_i} = [P_i \cdot P_i^T]$

Compute the $eigVals$ and $eigVecs$ of M_{P_i}.

8: $[eigVals,\ eigVecs] = linalg.eigh(M_{P_i})$

Sort all $eigVecs$ in decreasing ORDER of $eigVals$.

9: $eigVec_{[3]}$ = ArgSort$(eigVecs,\ eigVals)$

10: Direction check: if $p_i \dot{eigVecs} > 0$, then eigVecs = -eigVecs

Return least $eigVecs = eigVecs_{[3]}$ as n_i of p_i.

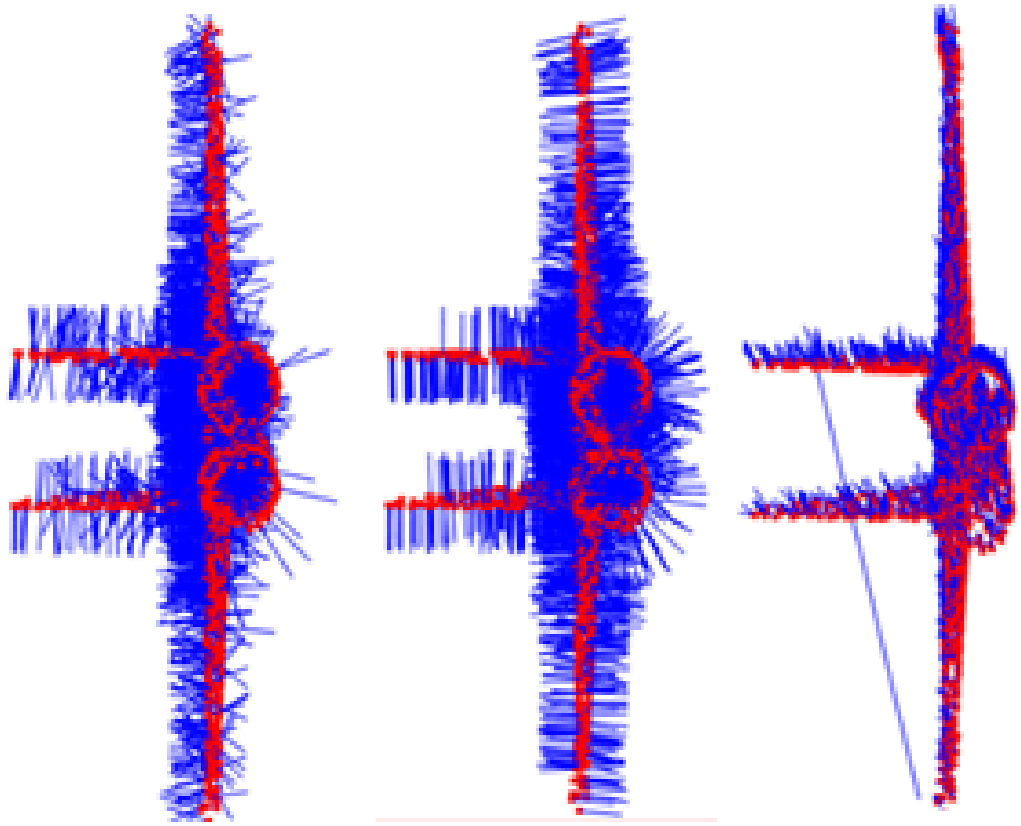
11: $[P_s,\ N_s]$.append$(p_i,\ eigVecs_{[3]})$

12: **end for**

Repeat loop $\forall\, p_i \in P_s$.

13: return $[P_s,\ N_s]$

points from 3D scanners). This technique has one main parameter, k or r in k-NN or r-NN algorithm, that influences the result of the estimated "normal vector" for each points cloud, and depends on how dense or sparse the input point cloud is (i.e. number of points sampled from mesh). In our implementations, we however, investigated both k-NN and r-NN on rigid, non-rigid, water-tight, and non-water-tight models and it was found that the implementation with k-NN search gave better and more desirable "normal vector" estimation results.

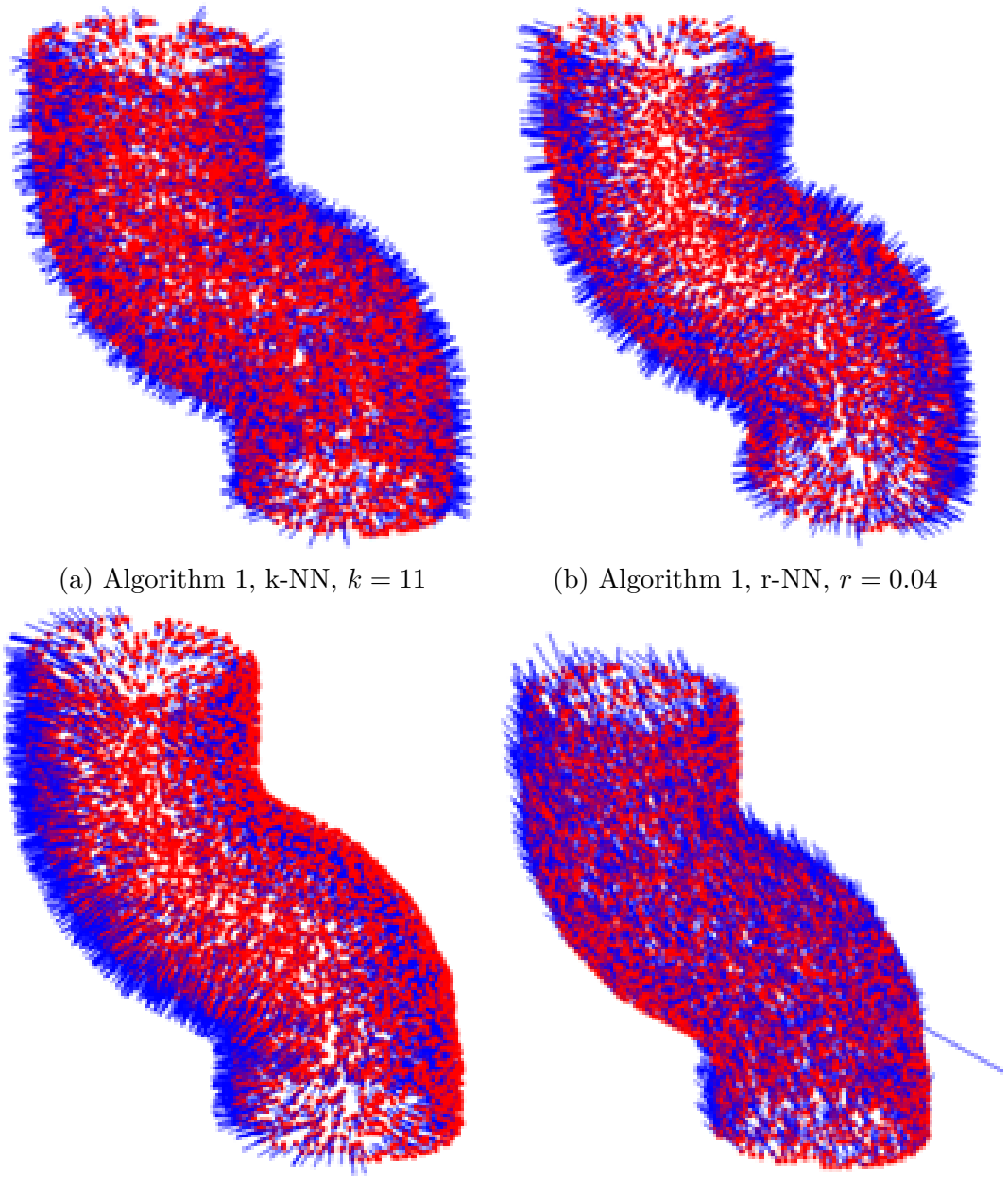


(a) Open3D tool [278] (b) Algorithm 1 (c) PCL tool [199, 4]

Figure 3.9: Visualization of Airplane 3D point cloud with associated surface normals estimated with three different algorithms/tools. Accurately estimated normals points outward in all direction to the surface as seen in Figure 3.9b, while inaccurately estimated normals are shown in Figure 3.9a and Figure 3.9c.

Established research by Open3D [278] and Point Cloud Library (PCL) [199, 4], for example have applied similar PCA-based technique to point cloud normals estimation and recorded remarkable success. We visually compared the results obtained with our PCA-based point cloud surface normals estimation algorithm with those returned by Open3D and PCL as shown in Figure 3.9. The basis for this comparison is to check that our algorithm was retuning acceptable results (estimated surface normals) and if not, we adopt the best tool/technique from existing tools, since inaccurately estimated normals would adversely affect our final shape descriptor. As we see in Figure 3.9b, our implementation (Algorithm 1) appears to estimate more desirable surface normals for the airplane model than methods with Open3D (see Figure 3.9a) and PCL (see Figure 3.9c). The results of our method were also good

across different range of datasets (rigid, non-rigid, water-tight, and non-water-tight models).



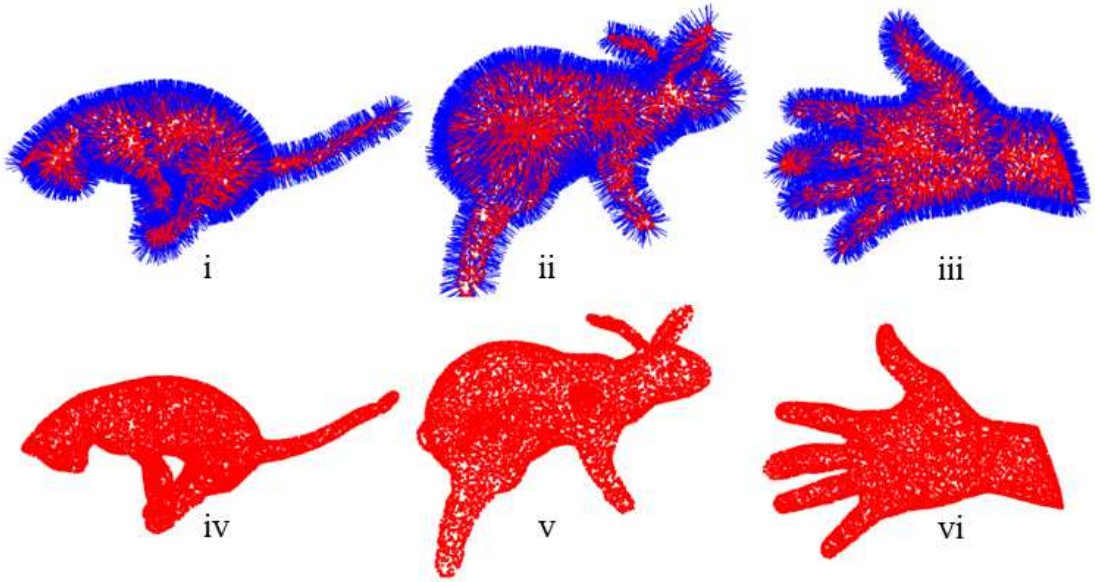
(a) Algorithm 1, k-NN, $k = 11$

(b) Algorithm 1, r-NN, $r = 0.04$

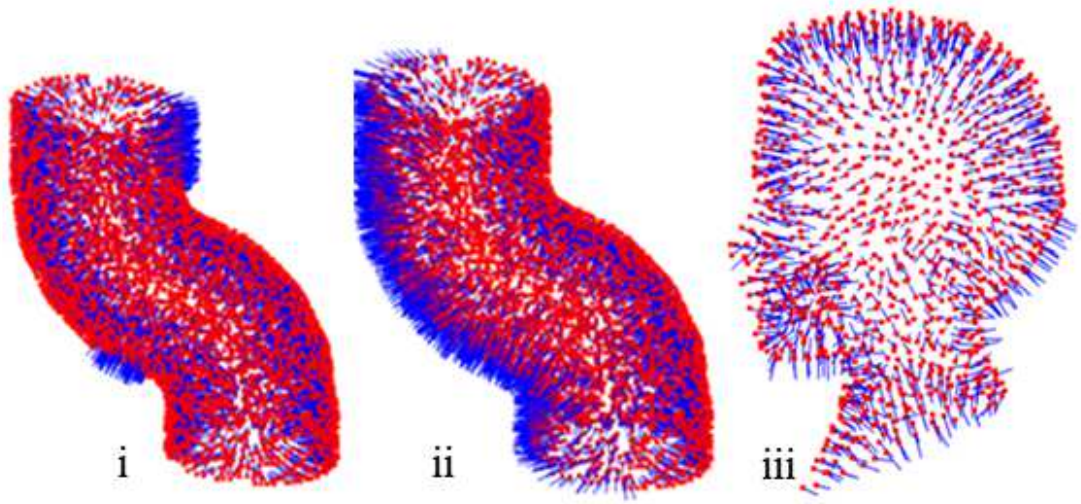
(c) Open3D tool

(d) PCL tool [199, 4]

Figure 3.10: Visualisation of rigid open pipe 3D point cloud with associated surface normals estimated with three different algorithms/tools. Figure 3.10a with k-NN search, where $k = 11$ and Figure 3.10b with r-NN search, where $r = 0.04$ shows implementation of Algorithm 1 with accurately estimated normals points outward in all direction to the surface, while Open3D and PCL tools/libraries produced poor normals as shown in Figure 3.10c and Figure 3.10d.



(a) Accurate Normals

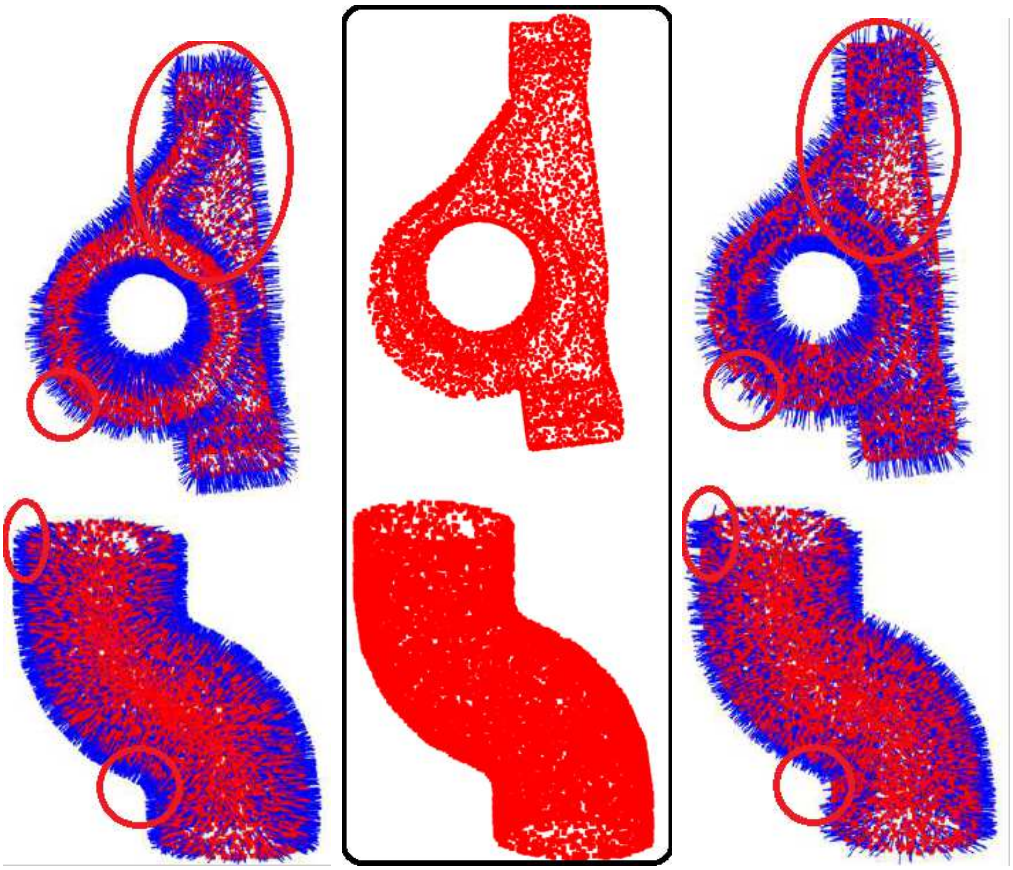


(b) Inaccurate Normals

Figure 3.11: 3D point cloud models with their estimated normal vectors. Figures 3.11a, iv-vi represent Cat, Standford Bunny, and Human-arm 3D models with their repective accurately-estimated surface normals in Figures 3.11a, i-iii. Alternatively, Figures 3.11b, i-iii represents the 3D models of a Pipe model (with inconsistent normal orientation), Pipe model (with one-directional normal orientation), and Human-head model (with inward pointing normal orientation).

3.3.3 3D Surface Normals Accuracy/Descriptor Robustness

Accurately estimated surface normals must have consistent orientations, pointing inward or preferably outward, as illustrated in Figure 3.11a, i-iii. To a considerable extent, however, the descriptiveness and robustness of a *normal vectors*-dependent shape descriptors (such as our proposed APPFD (see Section 4.2.1) and other points pair feature-based shape descriptors) depend on the accuracy of the estimated surface normals of the underlying 3D shape. Therefore, inconsistencies in normal vector orientations would adversely affect the descriptiveness and robustness of any shape descriptor or signature that depends on such surface normals (normal vectors) feature.



(a) Normals, BI sampling (b) Rigid point cloud (c) PCA-based normals

Figure 3.12: Rigid point clouds and their corresponding normal vectors, estimated with two different approaches, (a) normal vectors are estimated simultaneously using the Barycentric Interpolation (BI) approach described in Section 3.2.3, and (c) normal vectors are estimated using the PCA or Covariance technique in Algorithm 1, after point sampling technique explained in Section 3.2.3. *Left:* Point clouds of mechanical parts, and their corresponding normals estimated during sampling phase. *Middle:* Rigid point clouds of mechanical parts. *Right:* Point clouds of mechanical parts, and their corresponding estimated normals using PCA.

In our research implementations which requires the *surface normals* feature, we instead adopt BI technique (described in Section 3.2.3) for surface points sampling and normal vectors estimation that is computationally very efficient and reliable,

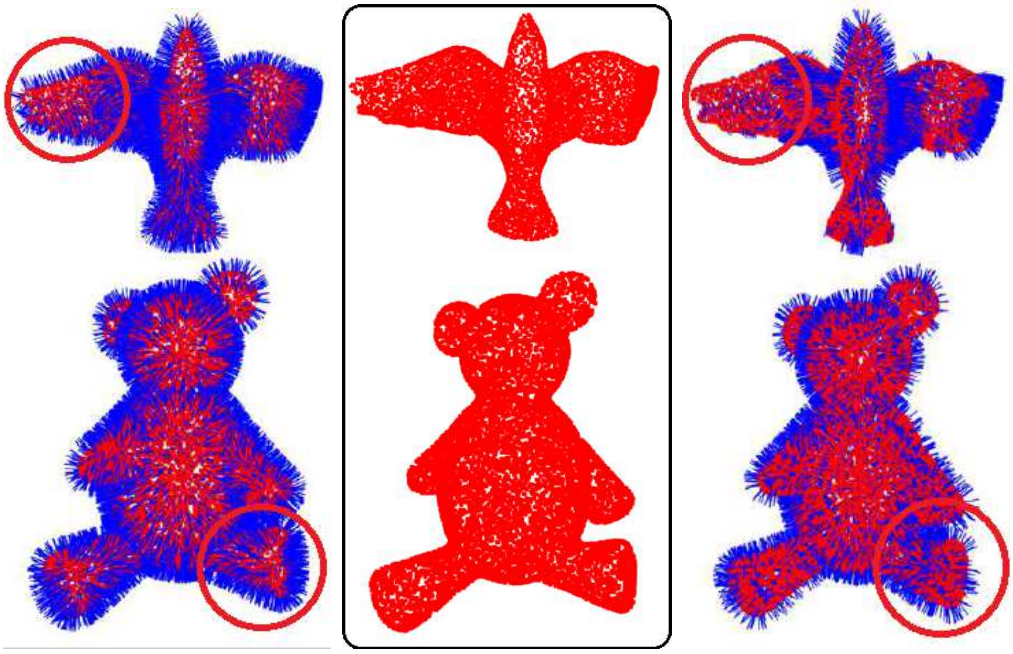
with remarkable results similar to that in [190]. The BI approach involves simultaneously estimating corresponding surface normals for all randomly generated points during mesh to point cloud conversion (described in Section 3.2.3), as opposed to only sampling the points and subsequently implementing the PCA-based approach to normal estimation. In addition, this approach is capable of producing consistently oriented, outward pointing, and accurately estimated normal vectors for the sampled (generated) point cloud, as shown in Figure 3.11b, i-iii, where the point cloud models of a Cat, Bunny, and Human-hand, respectively have their estimated surface normals positioned accurately and pointing outwards from their respective surfaces.

Unfortunately, there is no known computational technique specially developed to judge the accuracy (i.e. how *"good"* or *""*bad) of estimated surface normals other than mere inspection (visualisation). Empirically, the expectation is that a normal vector at any point on a flat surface (i.e. triangular face, as in Figure 3.6) must be perpendicular to the surface plane containing the point, to be adjudged as accurate. Alternatively, an accurately-estimated surface normal at any point on a curved surface is expected to be perpendicular to the tangent touching the surface and that point. Open source software and codes, such as MeshLab, Blender, VTK, Open3D, PCL, Matplotlib, etc., provide functions/tools to visualize 3D surfaces (mesh and point cloud) and inspect their normal vectors. Such visualizations are shown in Figures 3.8 and 3.9.

Further examples show several point clouds of rigid (see Figure 3.12) and non-rigid (see Figure 3.13) 3D shapes respectively (middle), and their corresponding surface normals estimated using two different approaches - one to the left and the other right of Figures 3.12 and 3.13. The surface normals visualised to the left are those from the improved BI approach (see the section that follows), while those visualised on the right-hand side are from the PCA-based approach. The visualisations on the left hand side of Figures 3.12 and 3.13, i.e. Figures 3.12a and 3.13a, depicts the simultaneously estimated surface normals during points sampling stage, for both rigid and non-rigid 3D point cloud models, respectively, while on the right, i.e. Figures 3.12c and 3.13c, the estimated surface normals are derived from further computation using PCA-based approach. Besides having computational speed and memory efficiency advantage, it is also very obvious in Figures 3.12a and 3.13a, that improved BI approach to point cloud normals estimation is much better (accurate) than the PCA-based approach.

3.3.4 Improved Surface Normals Estimation Technique for Triangular Mesh

As previously described, given a 3D triangular mesh, points can be sampled from every triangular face that make up the mesh to form point cloud as illustrated in Figures 3.1 and 3.8. The points sampling method described in Section 3.2.3 for randomly generating N points from the surface of a mesh is limited because additional implementation is needed to estimate surface normals for the sampled point cloud which adds to the complexity of the overall retrieval system. The ideal solution would be to estimate corresponding unit normal vector, $n_i = \{n_x, n_y, n_z\}_i \in N_s$, for



(a) Normals, BI sampling (b) Non-rigid point cloud (c) PCA-based normals

Figure 3.13: Non-Rigid point clouds and their corresponding normals, estimated with two different approaches, (a) normal vectors are estimated simultaneously using the Barycentric Interpolation (BI) approach described in Section 3.2.3, and (c) normal vectors are estimated using the PCA or Covariance technique in Algorithm 1, after point sampling technique explained in Section 3.2.3. *Left:* Point clouds of Bird and Teddy with their corresponding normals estimated during sampling phase. *Middle:* Non-rigid point clouds of Bird and Teddy. *Right:* Point clouds of Bird and Teddy with their corresponding estimated normals using the Covariance analysis (PCA).

every sampled point, $p_i = \{p_x, p_y, p_z\}_i \in P_s$ and simultaneously return point cloud and normal vectors, i.e. (P_s, N_s) in one run, during the sampling stage.

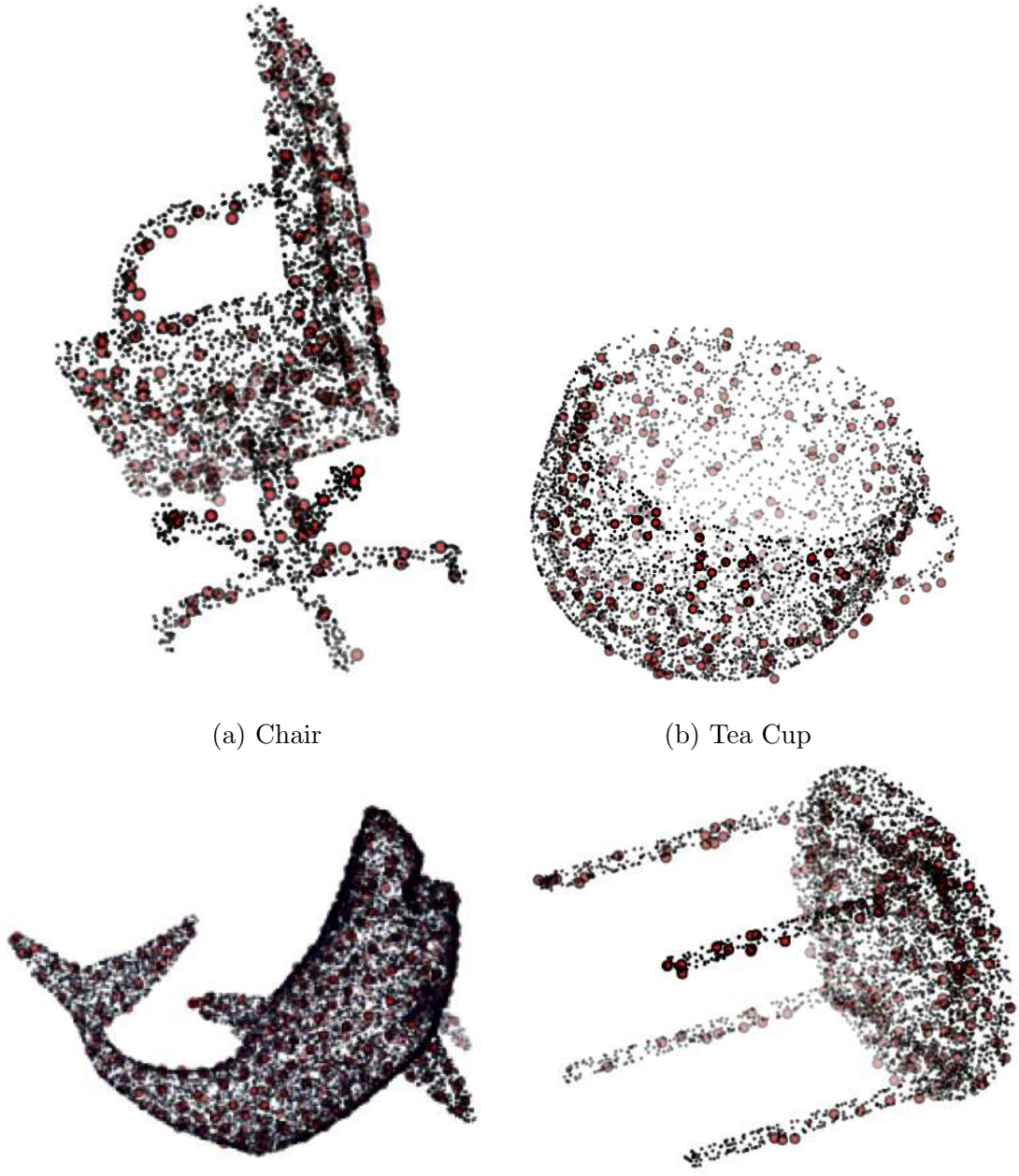
Using the traditional PCA-based approach described in [212] and [199] to estimate the surface normals (as in Figure 3.12c) for an already sampled point cloud would require looping over all points in the cloud. For 3D shapes with arbitrarily large number of points (point cloud), this process of looping over all points (including selecting surface regions for each point) for PCA would be very expensive, thus may adversely impact the overall performance and complexity of the retrieval system. To ameliorate this, the code in [47] can be modified to simultaneously also compute normals n_i for each sampled point p_i during points sampling phase using a computer graphic BI technique and NumPy broadcasting capability for faster computation.

In this section, we describe a point cloud sampling technique similar to the one described in Section 3.2.3 [173], for sampling (generating) point cloud P_s, with N points from a mesh, using the concept of barycentric coordinate system [21]. However, this approach is an improvement over the point cloud sampling technique proposed by [173], because corresponding unit normal vector, $n_i \in N_s$ is simultaneously estimated for each sampled point, $p_i \in P_s$ where $P_s \subset p_i$ and $i = 1...N$, during random points sampling phase. The barycentric approach presents an overall faster and efficient solution to point cloud surface normals estimation, which also produces accurate and outward-pointing (this is because the points in the original points have been put in a certain anti-clockwise order, leading the normal vectors computed to always pointing outward), unlike the previous techniques in [173] (Section 3.2.3) or PCA-based approach that only returns point cloud, P and expects further computational steps for surface normals estimation, which are prone to issues such as ambiguity and inconsistent normals orientation. However, in future work (see Section 6.5), it may be useful to do an ablation study about which method is better for shape retrieval between using the BI based approach or the one estimated from the sampled points vis PCA approach, rather than just criticising the latter.

3.3.5 3D Key Points

When we consider a typical 3D surface of, say, a rectangular-shaped table, for example, which is represented as points (or point cloud), the most noticeable or important points on this surface would be the four points at each corner of the table top, including each of the four points at the base of each of the legs of the table. Similarly, when we consider the point cloud of a human head, the points each of the eyes, ears, nose, mouth, cheeks, chin, and forehead would be considered as noticeable and important points. These type of points are known as *salient points* and must be capable of uniquely representing the topology of the surface it represents. For example, each of the red points for Chair, Tea Cup, Fish, and Round Table in Figure 3.14 can be considered as salient points.

When describing a 3D surface, it is possible to extract features from or within every point which makes up the geometry of that surface. However, while this approach



(a) Chair (b) Tea Cup

(c) Fish (d) Round Table

Figure 3.14: Four different 3D point cloud surfaces (*black points*), taken from the SHREC'12 dataset, showing sub-sampled points (bold, *red points*) which we consider as key points. Figures 3.14(a), 3.14(b) and 3.14(d) represent rigid-shapes, while Figure 3.14(c) represent non-rigid-shape. Note: Some of the red colors are showing as pink, due to being on the back, rather than front side of the 3D visualisation, hence are partially occluded.

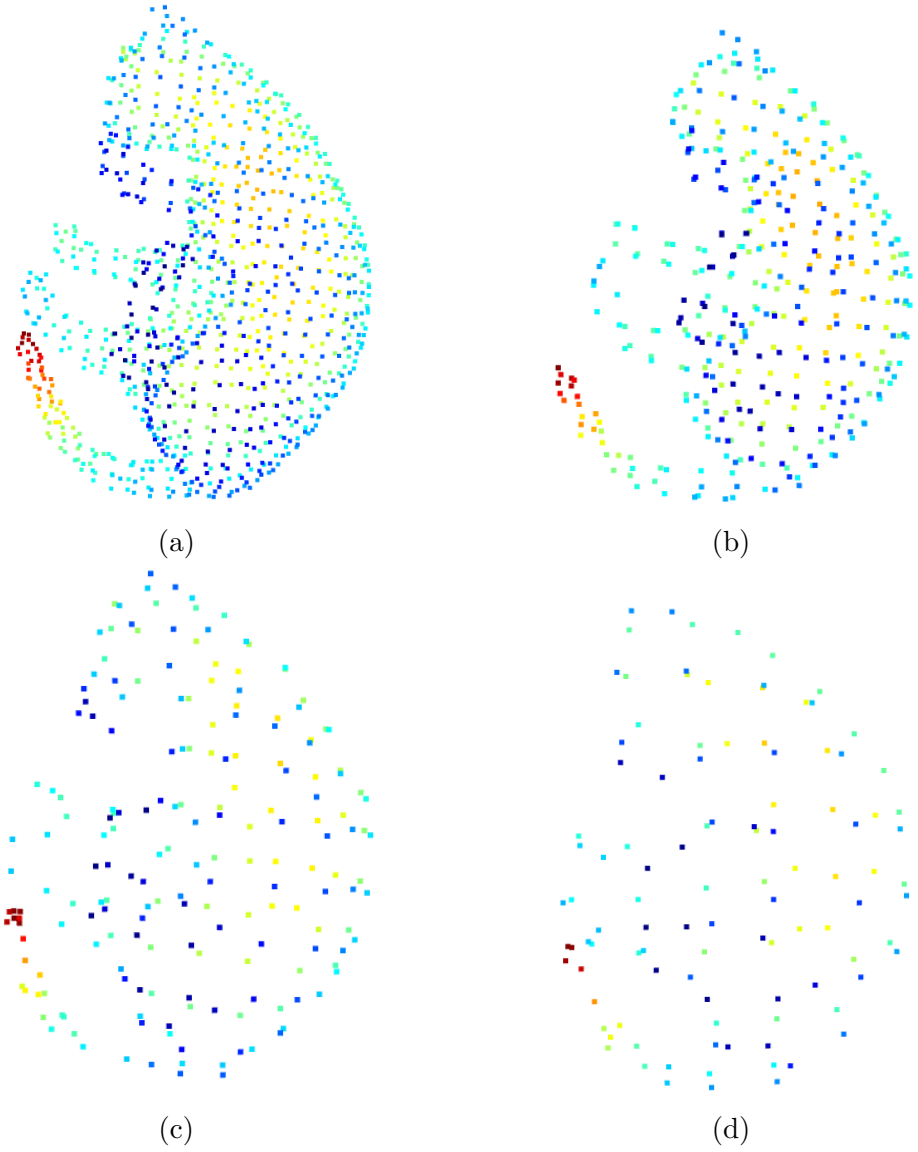
would be computationally inefficient, most of the surface points are not unique, thus extracting features from such no-important points would be unnecessary (i.e. insignificant for describing such surface). Salient points are therefore needed, to save computational resources (time and memory) and provide useful features and more representative surface details for any given surface. Although random selection or sub-sampling of original points set might also be useful during feature extraction and surface description, but the overall outcome would not be without deficiency, since some salient points may be omitted during random sampling and more insignificant points may be included. In the following sub-sections, we provide more details regarding 3D key points and the techniques adopted for this research.

Key points (also known as interest points) are salient points on a given 3D surface, which represents a sub-set of the entire surface points. Key points can be used to identify dominant or salient regions of the surface. Figure 3.14 shows four different 3D point cloud surfaces (black points) with derived key points (red points). Detecting salient points on a 3D surface (like point cloud) data not only save computational cost, but helps to improve the quality of locally extracted features from the surface. However, the process of detecting such salient points on 3D surfaces is more difficult than in 2D images, because images have richer set of distinct features. Generally, irrespective of whether 2D, 2.5D images, or 3D surfaces, the issues of key points saliency and repeatability (as explained by [221], Boroson and Ayanian [19]) plays a significant role during key points detection.

3D Key Points Determination

Developing a suitable 3D shape feature representation has become vital in content-based retrieval system (CBRS). These features are directly extracted from the vertices or points that make up the surface of 3D models, and must represent the intrinsic physical properties of the surface. To deal with the large number of vertices and faces (as in 3D mesh) or points (as in 3D point cloud) on the surface of a given 3D model, local or global descriptors are typically computed around a selection of salient points (key points) on the surface of the model. Similarly, in order to avoid the computational complexity that is required when extracting features from all 3D surface points or vertices, such features must be extracted from a minimal set of points, commonly referred to as key points. Although key points detection and/or extraction presents great challenge and remains a key research problem, especially for 3D shapes which requires that the detected surface points (key points) must meet certain repeatability criteria [19, 58], in this research, however, we are mainly interested in the use of existing 3D key points detector or key points selection technique as part of our data pre-processing pipeline, rather than proposing new key points detection method(s).

Determination Technique: Several key points detection techniques for 3D shapes (mesh or point cloud data) exists in literature [58], but only few techniques are suitable for 3D point cloud data. For example, the Harris 3D key points detector [220] and intrinsic shape signature – ISS [277]. Others, like a variant of Harris 3D key point detector [221], are designed to operate on 3D triangular meshes; NARF 2.5D [225] is



(a) (b)

(c) (d)

Figure 3.15: Visualization of down-sampled 3D point cloud shape, with $10,500$ points, using voxel-grid down-sampling techniques with varying voxel-sizes. In Figure 3.15(a), Voxel Size $= 0.06$, sub-sampled points returned $= 1,050$. Figure 3.15(b) had Voxel Size $= 0.10$, sub-sampled points returned $= 397$. Figure 3.15(c) had Voxel Size $= 0.15$, sub-sampled points returned $= 190$, while Figure 3.15(d) had Voxel Size $= 0.20$, sub-sampled points returned $= 105$.

suitable for sensor or depth-images, while THRIFT [59], generalized as SIFT 3D [58], are designed to work with range scans, for example.

3D Point Cloud Key points: In order not to reinvent the wheel, we implemented and tested the suitability of the Harris 3D and ISS 3D key points detectors across a wide variety of dataset types, such as the SHREC'11 [139] watertight dataset (see Section 5.2.3), which contains non-rigid 3D shapes and SHREC'12 [131] generic dataset (see Section 5.2.4), containing both rigid and non-rigid shapes, using publicly available code from Point Cloud Library (PCL) [199]. Only the Harris-3D technique seem to return meaningful results for rigid and non-rigid watertight shapes as visualised in Figure 3.17. Figures (3.17a)-(3.17c) showed reliable results of the Harris-3D key points detector on non-rigid watertight shapes, while Figures (3.17d)-(3.17f) showed poor results for some rigid 3D point cloud shapes. Overall, the Harris 3D key points detector technique failed completely on non-watertight shapes (rigid and non-rigid).

Problem with Existing 3D Key Points Detectors

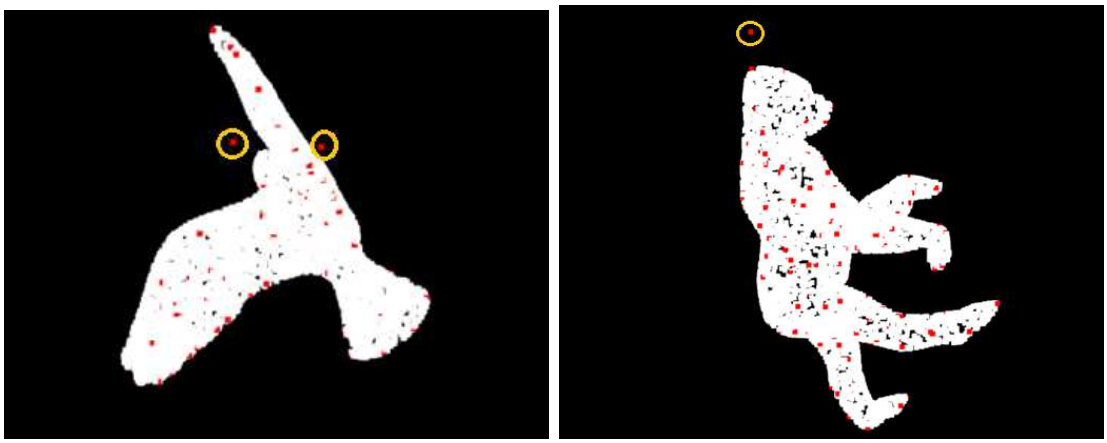
After several implementation attempts with existing 3D key points detection methods, we found out that for a given 3D shape (point cloud), the traditional 3D key points detectors like Harris-3D produced inconsistent key points at different runs (with all parameters remaining unchanged), which violates key points repeatability criteria. The resultant key points also contain outliers (see Figures 3.17(d)-(f)), even for the non-rigid watertight shapes in most instances, as seen in Figures 3.16(a) and (b). Accepting these results (i.e. key points with outliers) for the next phase - "key points feature extraction" and/or "shape descriptor construction" phase (see Section 3.4) would require additional pre-processing steps to remove outliers, which could impose further computational cost to our overall feature extraction pipeline.

Furthermore, for all the datasets we adopted for this study (see Section 5.2), none of the above-mentioned 3D key points detectors met the repeatability criterion expected for a good key points detector according to [58, 221, 19]. Therefore, there was need for a better and more stable or reliable approach that can ameliorate the 3D key points detection uncertainties experienced with the above-mentioned and/or existing techniques.

Alternative to 3D Key Points Detectors

As a better alternative to 3D key points detection for our datasets, we adopt the voxel-grid down-sampling technique by [278] to select sub-set of sub-sampled points (key points) around which features are extracted for each input 3D shape. These down-sampled points (i.e. key points) are capable of correctly representing the entire geometry of the sampled surface. We provide more details regarding the voxel-grid down-sampling technique in the following sub-section.

Voxel-grid Down-sampling: Intuitively, voxel can be understood as a **vo**lumetric pi**xel** as with pixels in a 2D bitmap images. It represents a value on a regular grid in 3-dimensional space. Voxel grid on the other hand is a structure obtained by subdividing the minimum bounding box of a point cloud into voxels. Figure 3.18



(a) Bird.off (b) Gorilla.off

Figure 3.16: Harris-3D key points with outliers detected for non-rigid, watertight, 3D Bird Figure 3.16(a) and Gorilla Figure 3.16(b) point cloud shapes, taken from the SHREC'11 watertight dataset. The point clouds are coloured in *White*, key points are coloured in *Red*, and the outliers (ill-defined blobs) are circled with *Yellow* marker.

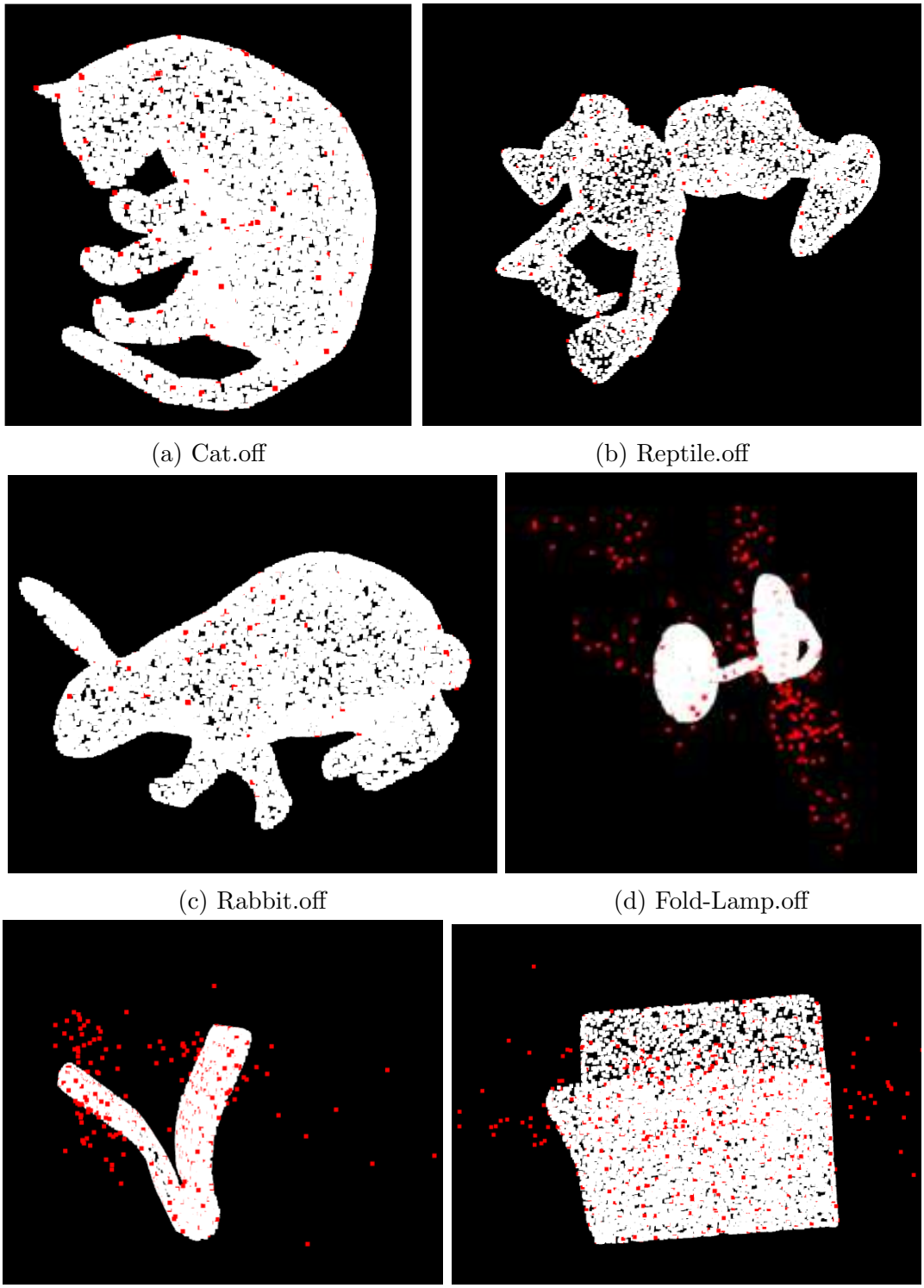
gives a clever illustration of voxel and voxel grid. As shown in Figure 3.18, an initial voxel bounding all point cloud data in 3D Euclidean space R^3 is sub-divided into subset voxels by grids along x, y, and z coordinates in a Cartesian coordinate system. Each voxel in the subset is represented by an index $V(i, j, k)$, where i $\in [0; N_x - 1]$, j $\in [0; N_y - 1]$, and k $\in [0; N_z - 1]$. Further description of voxels, voxel grid and voxelization is available in [81].

Practically, the sub-set of points returned from any 3D key points detection technique mentioned above are points that enclose the entire surface and accurately represent the topology of the 3D object. Likewise, down-sampling a 3D point cloud object produces a sub-set of the original point cloud which also accurately represents the overall topology of the the original 3D object, as we show in Figure 3.15.

Voxel-grid Down-sampling (3D Point Cloud)

The voxel grid down-sampling technique for 3D point cloud objects employ a two-step approach: (i) Bucketing a collection of points from the input point cloud into voxels. (ii) Computing the average of all points contained in each occupied voxel-grid, to generate one representative point for that grid.

Figure 3.15 shows a point cloud shape of a Cat with $10,500$ points, that has been down-sampled using four different voxel size parameters. Figure 3.15(a) returned $1,050$ sub-sampled points with a voxel size of 0.06, Figure 3.15(b) returned 397 sub-sampled points with a voxel size of 0.10, Figure 3.15(c) returned 190 sub-sampled points with a voxel size of 0.15, while Figure 3.15(d) returned 105 sub-sampled points with a voxel size of 0.20. From the above, we see that the *voxel size* parameter influences the number of sub-sampled points returned and the greater its value, the fewer the number of down-sampled points returned, and vice-versa. As illustrated in Figure 3.15(d), even the least set of sub-sampled points (i.e. 105 points)



(a) Cat.off (b) Reptile.off

(c) Rabbit.off (d) Fold-Lamp.off

(e) Book.off (view1) (f) Book.off (view2)

Figure 3.17: Harris-3D keypoints detected for rigid and non-rigid watertight 3D point cloud shapes. The point clouds are coloured in *white*, while key points are coloured in *red*. Figures (3.17a)-(3.17f) are point clouds of non-rigid shapes and their detected key points, except Figure 3.17d, which is the point cloud of rigid shape and its detected key points. All shapes are taken from the SHREC'11 watertight dataset described in Section 5.2.3.

with voxel size of 0.20 accurately represent the topology of the original Cat shape and properly captures the intrinsic geometry of the shape. In addition, unlike all the other 3D key points detector techniques we implemented, the down-sampling technique produced a stable solution by returning better *key points* result across different types of datasets (i.e. rigid, non-rigid, watertight, and non-watertight). Therefore, we adopted the voxel-grid down-sampling technique as an alternative to traditional 3D key points detectors.

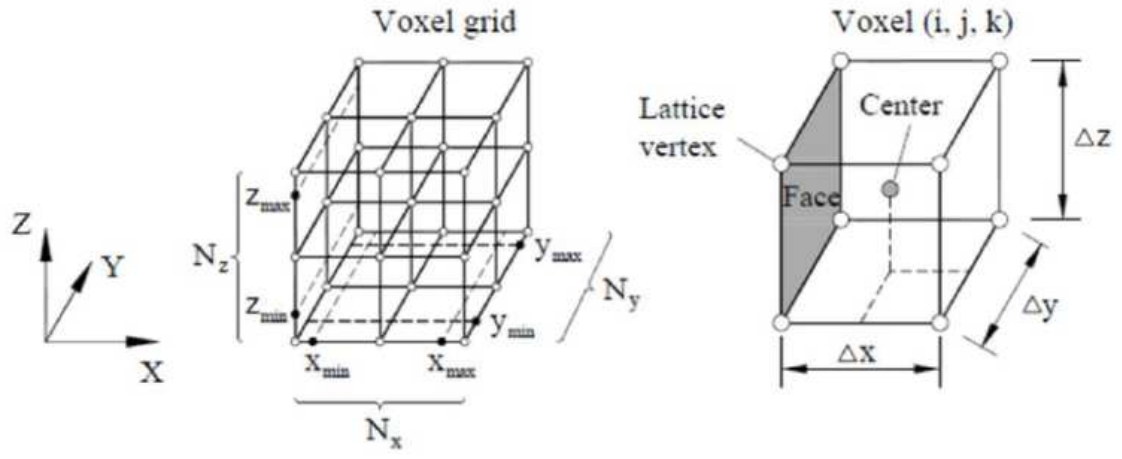
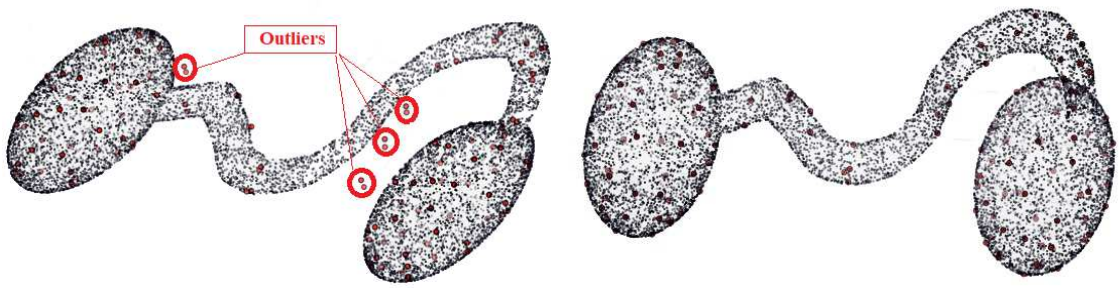


Figure 3.18: Voxel grid spanning a volume in a 3D space bounded by (x_{min}, x_{max}), (y_{min}, y_{max}), and (z_{min}, z_{max}). (Δx, Δy, Δz) represent voxel size, while (N_x, N_y, N_z) are the number of voxels in each direction [81].

In summary, we expect the outcome of voxel-grid down-sampling technique on 3D point cloud data to be a sub-sample of the original data, with reasonably smaller number of points representing the original point cloud. However, this is never the case with this approach, as the average of the points in each occupied voxel is calculated as its representative. Also, the position of the voxel would affect its representation. This means that the method may be sensitive to the noise, occlusion and appearance, including disappearance of points (see Figure 3.19). We applied a simple computational trick to resolve this, as illustrated in Figure 3.19b. The number of resultant sub-sampled points is directly proportional to the size of voxel-grid (i.e. voxel size) used during down-sampling, as clearly illustrated in Figure 3.15(a)-(d). Like the key points detection technique, the choice of points cloud voxel down-sampling technique, as pre-processing step has the advantage of further reducing the size of input data, thereby reducing computational time on future processing of the data (i.e. feature extraction).

After the voxel-grid down-sampling algorithm has been applied to each 3D point cloud object, as part of the preprocessing steps, we observe that for some model, the down-sampled points are not located directly on the surface of the point cloud (see Figure 3.19a). Using k-NN algorithm (where k=1) we develop a simple and quick method that searches each original point cloud surface for all the 1-closest points to the *down-sampled* points and return these as the actual down-sampled points. The outcome is visualised in Figure 3.19b, where all the down-sampled points are now properly located on the surface of the point cloud. This process is computationally



(a) Down-sampled points returns outlier points for some 3D objects.

(b) Outlier points replaced by their respective 1-NN on the surface.

Figure 3.19: Visualisation of a point cloud of a non-rigid 3D object (black points) with its down-sampled points (bold red points), showing some outliers after down-sampling in Figure 3.19a. We applied k-NN algorithm (where $k = 1$) to replace each of the outliers with its closest point on the surface of the point cloud, resulting in the outcome shown in Figure 3.19b.

very efficient considering that the k-NN algorithm is only searching for one point per iteration. However, we adopt the Numeric Python (*NumPy*) broadcasting technique as the best alternative to *FOR-loop* which slows down iterative processes.

3.4 Shape Descriptor Construction

Having extracted some features from 3D objects, the next action is to compute and index a compact mathematical representation of extracted features (i.e. descriptor or signature) for each 3D shape or object. Essentially, a shape descriptor defines the respective algorithms we have utilised to describe our database objects, and the outcome of a shape descriptor is a *feature-vector* (*fv*). For example, when a 3D object is used as input to a descriptor algorithm (i.e. shape retrieval method), the output would be a *fv*. This concept is illustrated in Figure 3.20.

However, throughout this thesis, we have used the terms: shape descriptors, shape retrieval algorithms and shape retrieval methods interchangeably to refer to the same thing. Consequently, we refer to the outputs of shape retrieval methods as shape descriptors, or feature-vector (*fv*).

In Section 2.3, we defined and provided detailed overview of 3D shape descriptors, including the different types and categories of 3D descriptors that are available in literature over the last few decades. We described in details the two broad approaches and/or techniques (i.e. data-driven and knowledge-based) commonly used to compute 3D shape descriptors and identified the knowledge-based approach to fall within the scope of our research. Further classification of the knowledge-based approach reveals other sub-categories of 3D shape descriptors based upon their computational approach, such as the normative aspect, structural, transformed-based, view-based, and statistical approaches to 3D shape descriptor construction.

In order to have proper understanding of our research work, we initially re-implemented a suitable number of existing local, global, and hybrid 3D shape de-

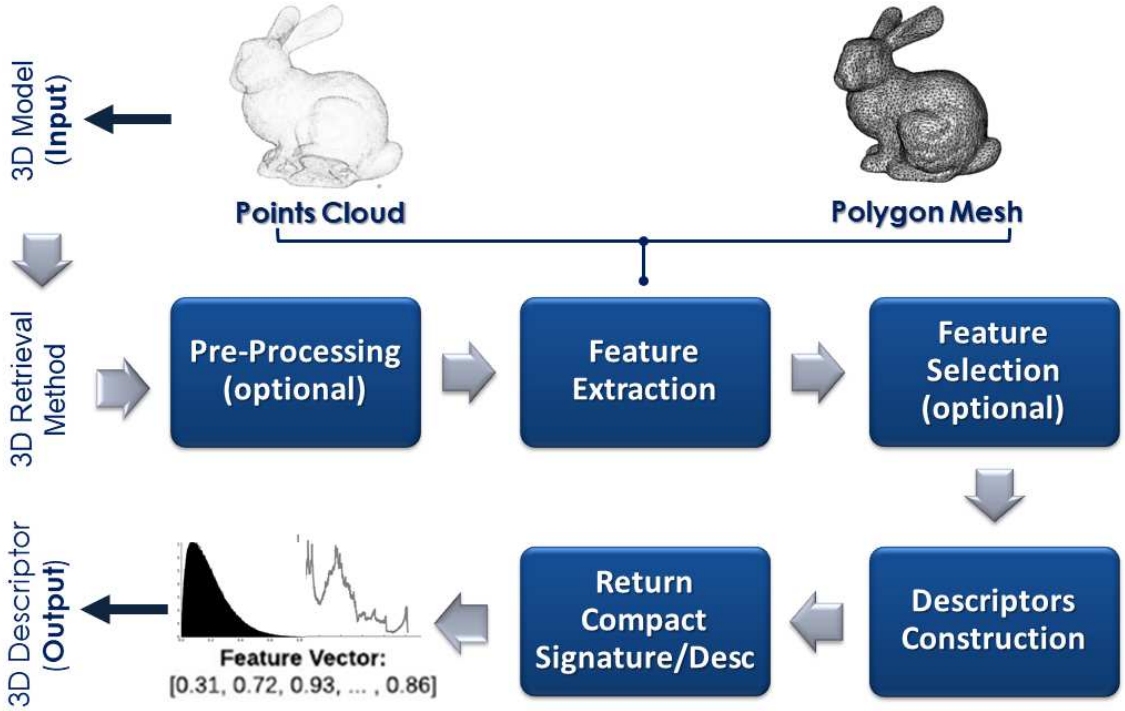


Figure 3.20: 3D descriptor (shape retrieval method) accepts 3D object as input and returns feature-vector, (fv) as output used to quantify the 3D object.

scriptors methods, such as the Shape Distribution (A3, D1, D2, D3) [174], Surflet Pair Relation Histogram [252], Spin Images [95], M2DP [80], and many others, using the dataset described in Section 5.2.11. We also performed several experiments (not included in this thesis) on varied sizes and types of datasets, such as rigid, non-rigid, watertight, and non-watertight 3D models, described in Section 2.2.2, using different parameter settings for each experimental run. Several observations were made and qualitative results were noted, some of which are recorded in Section 4. Following this, we implemented and proposed a number of novel local, global and hybrid 3D shape descriptors as major contributions of our research. They are: The APPFD, HAPPS, HoGD, and Agglomeration of Local Augmented Point Pair Features Descriptor with Fisher Kennels and Gaussian Mixture Model (APPFD-FK-GMM). Details of these proposed descriptors implementation are provided in Section 4.2.

3.5 Database Indexing

Indexing a dataset basically involves the process of quantifying a given dataset using a shape descriptor, signature or feature-vector representing each 3D object in the database. After deciding on which features to extract and which shape descriptor algorithm to implement, the next step is to implement (define) our shape retrieval method. Then, to index the 3D shapes in a given dataset, we apply the defined shape descriptor to each 3D object in the database, extract intended features, compute defined descriptor, and write the output signature (feature-vector) to a file, which are retrieved for later similarity comparison. Generally, the goal of features extraction and shape descriptor computations is to produce a set of shape signature or feature-vectors that effectively represent database objects without having to

use all the complex original data. The concept of database indexing is depicted in Figure 3.21, where 3D features are extracted and shape descriptors computed for each of the 3D objects in the database (left), with $size = M$, where M is the total number of shapes in the database. Then another database (right) is created, which contains respective signatures (feature-vectors) for each of the 3D objects in the first database.

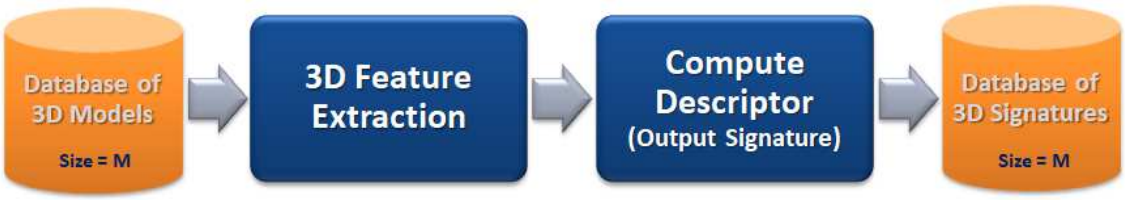


Figure 3.21: 3D Objects Database Indexing: We take a given dataset of 3D objects, extract features and apply our descriptor algorithm to each 3D object to obtain feature-vectors, and then storing these signatures back in a new database.

Compact representation of data allows for effective indexing and enables much quicker comparison of two 3D objects than would otherwise be possible, because rather than indexing and matching the complex 3D structures (represented as mesh, voxels, or unorganised points), simple and compact mathematical representations (feature vectors) are indexed and compared instead. During indexing, the idea is to efficiently store the computed compact representations (descriptors/signatures) for all database objects in a way to allow easy access in the future during shape matching and retrieval. Therefore, shape descriptors govern how the database objects are quantified.

3.6 Shape Matching and Retrieval

Having completed all the previous shape analysis procedures (i.e. data pre-processing, feature extraction and shape descriptor construction, including database indexing) for a given dataset of 3D objects, such as SHREC'2017 PRoNTo dataset for example (see Section 5.2.6), the final task is to compare the indexed signatures for any two 3D shapes (i.e. the query shape and each of the database shape) using a distance metric or similarity function and produce a (dis)similarity score. The distance metrics and/or similarity functions take two *feature-vectors* (i.e. the output of the process in Section 3.4) as inputs and then output a (dis)similarity score (a number that represents how *"similar"* the two *feature-vectors* are). This concept is illustrated in Figure 3.22, whereby given two feature vectors (Signature #1 and Signature #2), representing the query 3D objects and a database 3D object, a similarity function (say Euclidean distance) is used to determine how similar the two 3D objects are. Therefore, the similarity between two 3D objects is defined as the distance between their respective signatures or descriptors. The smaller their distance value, d, the more identical the two objects are and vice versa.

At this point, it may be necessary to evaluate the *"qualitative"* and *"quantitative"* performances of our signature(s) or shape retrieval algorithm(s). To achieve

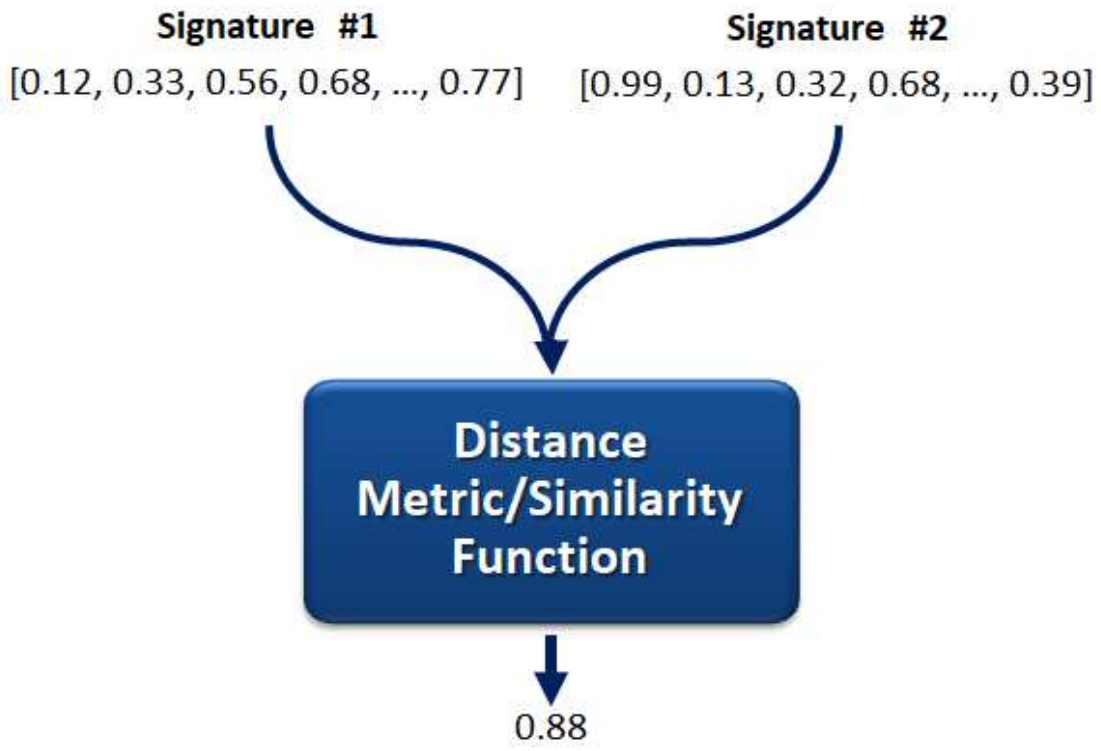


Figure 3.22: Comparing two 3D objects represented by their signatures (feature-vectors, *fv*), using a distance metric and/or similarity function to return a similarity score.

these, the information retrieval (IR) performance evaluation metrics described in Section 4.3.3 are needed. Prior to performances evaluation, we first produce a (dis)similarity or distance *Matrix*, DM, described in Section 4.3.2, for a given benchmark dataset (say SHREC'17 PRoNTo dataset) by computing the distances between all pairs of signatures describing all 3D objects in the dataset. Any of the distance/(dis)similarity *measures/metrics* described in Section 2.2.3 can be used. For each distance metric, we compute a single DM for evaluation purpose. Details of this computation is provided in Section 3.6.2. The DM, together with the classification file or ground truth are needed by the performance evaluation mechanism or software tools in order to compute qualitative and quantitative evaluation results. We describe these performance evaluation processes in details in Section 4.3.

3.6.1 Shape Descriptor Comparison

The similarity or (dis)similarity (i.e. distance) between any two 3D objects can primarily be obtained using two major approaches: (i) local matching approach and (ii) global matching approach, depending on whether the final shape descriptors representing the 3D objects are *local* or *global* descriptors. For example, if several key points descriptors, $[K \times D]$, are returned per 3D object, where K is the number of key points detected and D, the size or dimension of each key point descriptor, then the local descriptor matching approach is required, but if a single $[1 \times D]$ descriptor is returned per 3D object, them the global descriptor matching approach would be

useful, instead. An overview of each of these shape descriptors matching approach is provided in the sub-sections that follow. However, regardless of whether local or global matching is involved, distance/(dis)similarity metrics is required to match any two descriptors or signatures. There are several standard metrics for computing distances (i.e. comparing shape similarity) between two descriptors and we had provided detailed description of five of these in Section 2.2.3. Before (dis)similarity comparison, we normalise each signature (*feature-vector*), S for every 3D object, such that their absolute sum is equal to 1, as expressed in Equation (3.8), where d is the dimension of the *feature-vector* or number of bins in the histogram if the final signature is a $1D$ histogram. The primary purpose of normalisation is so that the method is robust to the variation in the number of sampled points and variations in scaling between different 3D objects.

$$\sum_{i=1}^{d} S[d] = 1 \tag{3.8}$$

After shape descriptor normalisation, local or global matching is done using any of the distance metrics described in Section 2.2.3. In this thesis, we implement the global matching approach/technique described in Section 2.2.6, using five standard (dis)similarity metrics (see Section 2.2.3). The choice of the global technique is due to its simplicity (ease of implementation), computational speed and efficiency, unlike the local descriptor matching, which is associated with prohibitive cost of computation. For our 3D shape retrieval method implementations, however, descriptors are computed locally, but concatenated and matched globally.

3.6.2 Distance or (Dis)similarity Matrices From Metrics

In Section 2.2.3, we provided a general overview and detailed description of the distance/(dis)similarity metric, and provided further details on each of the distance metrics we have used in our implementations and reported in this thesis. In Section 4.3.2, we also provide adequate details regarding distance or (dis)similarity matrix, Distance Matrix (DM). In this section, we will briefly describe how we used each of the five distance metrics described in Section 2.2.3 to produce a distance matrix, DM data structure (see Section 4.3.2 for details). DM acts as a preliminary retrieval result which is needed for final performance evaluation of our shape retrieval algorithm (descriptor). Among the many different distance or (dis)similarity metrics that exist in literature, the Euclidean distance metric (see Equation (2.1)) has become one of the most popular metric for similarity matching. However, we applied the following five metrics for the similarity measurements of our various descriptors in order to finally adopt the ones with better results for our shape matching/retrieval algorithms.

- **Euclidean Distance Matrix DM:** For a given 3D database where shape signatures have been computed and indexed for all the 3D objects in the database, we defined a DM function that computed the Euclidean distance between all pairs of signatures after normalisation (as in Equation (3.8)) and return a pairwise distance matrix: $D = M \times M$, where D_{ij} is the distance

between the shape descriptor for 3D object i and descriptor for object j, and M is the number of 3D objects in the database. The distance formula used in this case is given in Equation (2.1). That is:

$$ED(h_Q, h_D) = \delta_{L_2}(h_Q, h_D) = \sqrt{\sum_{i=1}^{d}(h_{Q_i} - h_{Di})^2}$$

.

- **Other Distance Matrices DMs:** Following the above computation of DM using the Euclidean distance metric for pair-wise descriptor matching, in order to perform matching with any other distance metrics described in Section 2.2.3, we simply replace the Euclidean metric formula in the DM function with such metric. For example, if we consider the Cosine distance metric for shape/descriptor matching, we would simply replace Equation (2.1) with Equation (2.2), and return a matrix: $D = M \times M$, where D_{ij} is (1 - Cosine distance) between the shape descriptor for 3D object i and descriptor for object j.

Besides experimental evaluations approach where we compared our retrieval methods with those of several other state-of-the-art, part of our evaluation approach/strategy was to investigate the performances of different (dis)similarity metrics on each of our retrieval method. We observed that the best performing distance metric for one method may not perform so well for another methods or dataset. However, for each retrieval problem/task and dataset, we selected the best performing metric - see results sections in Chapter 4.

Chapter 4

METHODOLOGY

4.1 Introduction

This chapter essentially focuses on the major contributions of the work in this thesis. Our proposed methods for describing 3D objects (meshes and point clouds) are presented, which includes all the three broad classifications of shape descriptors (global, local, and hybrid) described in Section 2.4, and are listed in Table 4.1. Next, we provide detailed background to each of the proposed 3D shape descriptors (i.e. retrieval methods) in this thesis. In our shape descriptor implementation approach, we consider an *oriented* raw point cloud data structure, with $[x, y, z]^T$ coordinates as the de-facto input to our shape descriptor algorithm. By *'oriented'*, we refer to point cloud and their associated or corresponding normal vectors. Datasets are needed to evaluate the performances of these retrieval method. We present and describe the datasets for which our proposed methods have been evaluated against in Chapter 5 (see Section 5.2).

Performance evaluation is an essential part of any IR system or algorithm, which provide researchers with tools to assess how well a particular algorithm or retrieval method performs on a particular dataset and retrieval problem. In this chapter (precisely Section 4.3), we provide adequate information regarding the evaluation techniques we have adopted in this thesis, where we describe the tools, terminologies and processes involved, as well as the different performance evaluation metrics used to measure the overall performance of different retrieval methods. However, the retrieval performances of each of the shape descriptors, which are presented in this chapter, are provided in Chapter 5. There, the experimental results and testing for each of our retrieval methods on various 3D benchmark datasets are compared with several other state-of-the-art methods.

Earlier (see Section 3.1), we already laid strong foundations toward all the proposed 3D shape retrieval methods we present in this chapter, by providing comprehensive details regarding our research strategies, tools, and techniques, and explaining important concepts which are applicable to our retrieval methods and algorithms implementation. Such techniques and concepts include data pre-processing, point cloud sampling from meshes, affine transformation, feature extraction, surface normal estimation, 3D key point detection and point cloud down-sampling using voxel grid, etc. For instance, the primary data input to our algorithms is the raw point

S/N	Descriptor Name	Approach	Category	Abbrev.	Size
1	Histogram of Global Distances	Statistical	Global	HoGD	65
2	Augmented Point Pair Feature Descriptor	Statistical	Local	APPFD	15625, 262144
3	Hybrid Augmented Point Pair Signature	Statistical	Hybrid	HAPPS	117845, 262209
4	APPFD Agglomeration with Fisher Kernel and GMM	Statistical	Local - Global	APPFD-FK-GMM	4,210 / 162, 312510 / 186

Table 4.1: List of our proposed 3D shape retrieval methods (descriptors).

cloud data as mentioned previously. However, given a triangular mesh, instead as input to our algorithm, we implement a mesh to point cloud random sampling method based on barycentric interpolation technique, described in Section 3.2.3, which simultaneously determines corresponding normals for the samples point cloud. With the raw point cloud data, our method adopts a simple Covariance analysis (see Section 3.3.2), with carefully selected k-NN parameters, to estimate fairly accurate point cloud normals if needed by the shape descriptor algorithm. Because the accuracy of the estimated surface normals is critical to the retrieval methods we propose, we provide further analysis regarding surface normal estimation from 3D mesh and point cloud, and visually examine the results we get in comparison to those from existing methods.

Another critical aspect of our base retrieval method, the APPFD local descriptor, is key point detection for which local features are extracted. In Section 3.3.5, we overview the different 3D key points detection technique in literature and reveal how unsuitable they are to most of the datasets we evaluate in this thesis. We therefore adopt the voxel-grid downsampling approach as the best alternative to traditional 3D key points detectors, which are not exceptionally reliable. In order to determine the (dis)similarity between two 3D objects, their respective shape descriptors are compared using a suitable distance metric. In this chapter, we further explain the implementation of the five different shape similarity (distance) metrics, such as the Euclidean metric, Cosine metric, Earth Mover's Distance metric, Kullback-Leibler Divergence metric, and Squared Euclidean Distance metric (see Section 3.6.2), which we have tested with our retrieval methods (descriptors). In Section 5.2 we introduce several 3D benchmark datasets for which our retrieval methods were evaluated upon. The 3D objects in these datasets include rigid, non-rigid, watertight and non-watertight models, and majority of the datasets contain 3D triangular meshes, except one (SHREC'17 PRoNTo dataset), which contain raw 3D point cloud models. Standardised tools and methods are required to evaluate algorithm performances on the respective datasets. In Section 4.3, we describe several commonly used performance evaluation tools and metrics in Information retrieval.

4.2 Methods and Algorithms

In Sections 2.3, 2.4, and Section 2.6, we provide details of most relevant 3D shape descriptors, including the different types and categories of 3D descriptors that are available in literature. We also examined the two broad approaches and/or techniques to 3D shape descriptors construction (data-driven and knowledge-based) in

Section 2.5. As previously mentioned, the knowledge-based approach reveals other sub-categories of 3D shape descriptors based upon their computational strategies. These are the normative aspect, structural, transformed-based, view-based, and statistical approaches to 3D shape descriptor construction. In this section, we specifically provide the implementation details of our proposed 3D shape descriptors which are essentially within the knowledge-based category and the statistical-based 3D shape descriptors sub-category.

4.2.1 Augmented Point-pair Feature Descriptor (APPFD)

The APPFD is a local 3D object descriptor made of local features that capture the geometric characteristics or properties of surface patches, each centred at a point (i.e. *key point*), $p_{k_i} = [x, y, z]$ or vertex, $v_{k_i} = [v_x, v_y, v_z]$ of 3D point clouds or meshes, respectively. The APPFD incorporates the geometric relation between p_{k_i} and its r-nearest neighbours (i.e. surface patch at p_{k_i}). Implementing the APPFD algorithm involves the following stages: (i) point cloud sampling and surface normals estimation, (ii) key point, p_{k_i} determination, (iii) local surface patch (LSP) or region, P_i selection, (iv) PPF and APPF extraction per LSP, and (v) final key points descriptor (i.e. APPFD) computation for each LSP for *"local"* APPFD. It is also very possible to compute of final APPFD using all the locally-extracted and vertically-stacked APPFs from each 3D model for *"global"* APPFD. We have provided detailed description of stages (i) and (ii) in Sections 3.3. In this section, we would describe the remaining three stages - (iii) LSP selection, (iv) PPF and APPF extraction, and (v) APPFD computation for each LSP. This last step is very important and reveals exactly how our locally-extracted novel 6-dimensional PPF arrays are converted to a multi-dimensional histogram, which is flattened to produce the novel APPFD.

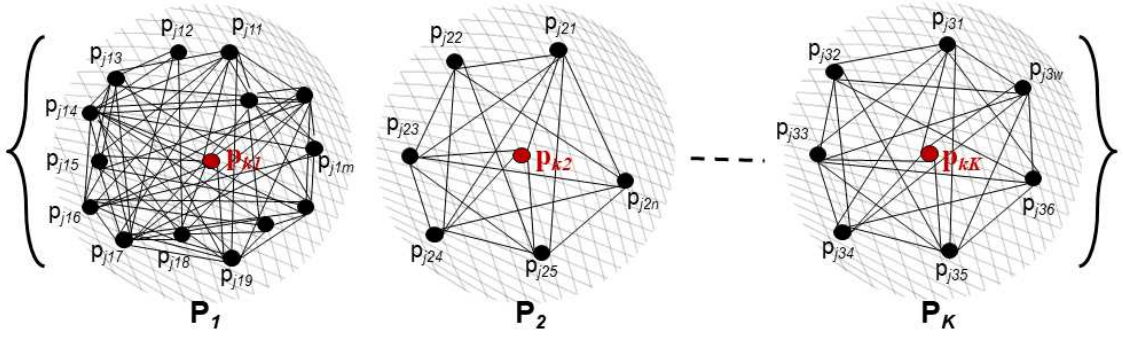
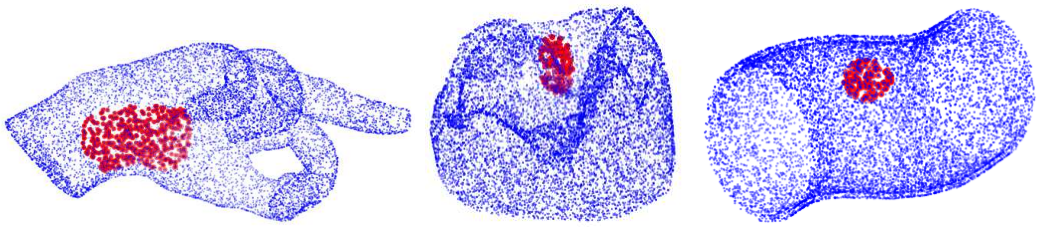


Figure 4.1: Illustration of Local Surface Patches (LSPs) or Regions, $P_i = \{P_i, i = 1...K\}$, which are extracted around their corresponding key points $p_{ki} = \{p_{ki}, i = 1...K\}$ (red), from which local features, $APPF$ are extracted to compute our final descriptors, $APPFD$ for a given 3D shape.

Local Surface Patch (LSP) Selection

For a given 3D object represented as a point cloud P, with K *key points*, $\{p_{k_i}, i = 1 : K\}$, we extract K LSPs or regions $\{P_i, i = 1 : K\}$, by using the fast KD-Tree algorithm, *cKDTree* in Scientific Python (SciPy) package [186] to select neighbourhood of points, P_i within a specified radius ($r \approx 0.20$min, 0.50max) around each p_{k_i}. We illustrate the LSP extraction concept in Figure 4.1, where for each LSP (i.e. P_i), the actual number of points in the spherical r-neighbourhood varies, hence the size of the selected surface region also varies. This number of key point neighbourhood that makes up the LSP is controlled by the r or k parameters, for r-NN or k-NN algorithm, respectively and depends on the density of points in the point cloud data. Figure 4.1 illustrates the LSPs $P_i = \{P_i, i = 1...K\}$, for a given 3D object, centred around each key point, $p_{ki} = \{p_{ki}, i = 1...K\}$ (in red colour), where K here, represents the number of LSP and/or key point for a single 3D object. We show the variation in the number of r-neighbourhood or k-neighbourhood points p_j, extracted for each LSP, P_i about a key point, p_{ki}. This variation depends on the topology of the surface region and/or points density in that region. Similarly, the sizes of the LSP differs as we have also illustrated in Figure 4.2, showing different 3D point cloud models (in blue colour), each having a single LSP, P_i (in red colour) with different patch (LSP) sizes.



(a) Hand model (b) Rock model (c) Pipe model

Figure 4.2: Each of the blue points are three different point cloud representations of human hand, rock and pipe 3D shapes. The red points in each of them depicts a single LSP, $\{p_n = (x_n, y_n, z_n)^T \in P_i\}$.

PPF and APPF Extraction for each LSP

In our work, the PPF and the APPF are extracted locally for every LSP, P_i ($\overline{P_i}$ if normalised) and/or key point, p_{k_i} derived from a 3D surface. After point cloud sampling and surface normal estimation, the next step to APPFD is to compute key points, $\{p_{k_i}, i = 1, 2, 3, \cdots, K\}$ and locally extract 4-dimensional PPF, $f_1 = (\alpha, \beta, \gamma, \delta)$ as in [252] from points combination in $\{P_i, i = 1 : K\}$ around each key point $\{p_{k_i}, i = 1 : K\}$, where K is the number of key points for a given 3D object. For every pair of points, $[p_i, p_j]$ and their estimated normals, $[n_i, n_j]$ i.e. oriented points, $[(p_i, n_i), (p_j, n_j)]$ in P_i, where $i \neq j$ and p_i is the ***origin*** with respect to the constraint in Equation (4.1) holding TRUE, a transformation-independent Darboux frame U, V, W is defined thus: $U = n_i$, $V = U \times ((p_j - p_i)/\delta)$, $W = U \times V$. This formulation is clearly depicted in Figure 4.3.

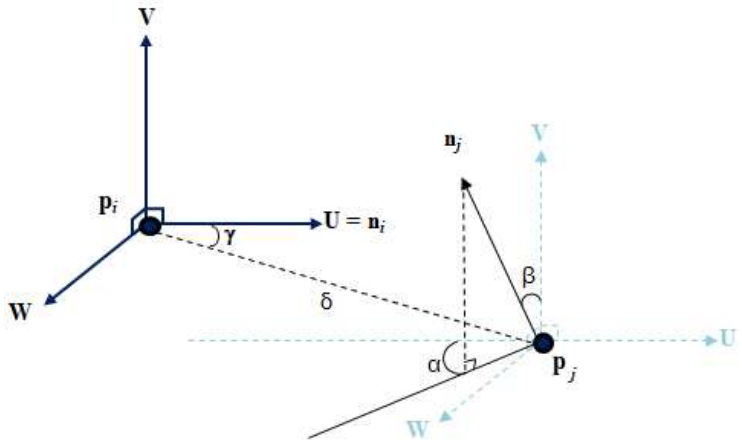


Figure 4.3: An instance of a single surflet-pair, i.e. $\{(p_i, n_i), (p_j, n_j)\}$, which is a pair of points (p_i, p_j) and their corresponding normal vectors (n_i, n_j). In this example, a fixed or Local Reference Frame (LRF), U, V, W is defined at point p_i, from which the angular differences α, β, γ, between normals n_i and n_j, and the distance δ between points p_i and p_j are extracted.

$$|n_i \cdot (p_j - p_i)| \leq |n_j \cdot (p_j - p_i)| \tag{4.1}$$

Alternatively, p_j becomes the ***origin*** (i.e. point with the larger angle between its associated normal and the line connecting the two points) if the constraint in Equation (4.1) is FALSE, and the variables in Equation (4.1) are reversed. f_1 is then derived for the source point as follows:

$$\alpha = \arctan(W \cdot n_j, U \cdot n_j), \;\; \alpha \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right] \tag{4.2}$$

$$\beta = V \cdot n_j, \;\; \beta \in [-1, 1] \tag{4.3}$$

$$\gamma = U \cdot \frac{p_j - p_i}{\|p_j - p_i\|}, \;\; \gamma \in [-1, 1] \tag{4.4}$$

$$\delta = \|p_j - p_i\| \tag{4.5}$$

Secondly, $f_2(p_i, p_j) = (\phi, \theta)$ is extracted for every possible combination of points pair, $[p_i, p_j]$ in P_i, because f_1 is not robust enough to capture the entire geometric information for a given LSP. In addition, the PPF approach opens up possibilities for additional feature space. Therefore, as illustrated in Figure 4.4, ϕ is the angle of the projection of the vector $\overrightarrow{S}$ onto the unit vector $\overrightarrow{V_2}$, while θ is geometrically the angle of the projection of the vector, $\overrightarrow{S}$ onto the unit vector $\overrightarrow{V_1}$, where $\overrightarrow{V_1} = p_i - p_c$, $\overrightarrow{V_2} = p_i - l$, and $\overrightarrow{S} = p_i - p_j$, with $p_c = \frac{1}{n_i}\sum_{i=1}^{n_i} p_{k_i}$ (i.e. LSP centroid), and $l = (p_{k_i} - p_c)$, the vector location of p_{k_i} with respect to its LSP. Note that p_i, p_j, p_c, and l are all points in $\mathbb{R}^3$ space, although l is a vector. In summary, a step-by-step procedure for our PPFs and/or APPF extraction has been provided in Algorithm 2 for more clarity. Algorithm 3 and Algorithm 5 is called from within Algorithm.

Algorithm 2: 6D APPF Extraction Procedures for LSPs (needed by Algorithm 3 or Algorithm 4)

INPUT: Oriented Local Surface Patch, (P_i, N_i) from normalised and scaled point cloud, P_s computed in Algorithm 4. Where $P_i = lspExtraction(P_s, r)$, $i = 1...K$, and K is the number of key points.

OUTPUT: 6-dimensional APPF *feature*, $f_{3(P_i)} = (\phi, \theta, \alpha, \beta, \gamma, \delta)$ for input LSP, P_i.

\# Extract 6D APPF for every oriented point-pair combination in LSP, P_i around a key point, p_k.

for all $[(p_i, n_i), (p_j, n_j)]$ in P_i **do**

  \# Compute Darboux Frame (U, V, W)

  \# First, evaluate constraint in Equation (4.1), for p_i as ***"Origin"***.

  if $|n_i \cdot (p_j - p_i)| \leq |n_j \cdot (p_j - p_i)|$ **then**

    $U = n_i$, $V = U \times ((p_j - p_i)/\delta)$, $W = U \times V$.

    \# Compute 4D PPF, $f_1 = (\alpha, \beta, \gamma, \delta)$ for (p_i, p_j) in P_i [252]

    Extract $(\alpha, \beta, \gamma, \delta)$ according to Equations (4.2), (4.3), (4.4), and (4.5), respectively.

    \# Compute 2D PPF, $f_2 = (\theta, \phi)$ for (p_i, p_j) in P_i

    Compute $p_c = \frac{1}{n_i} \sum_{i=1}^{n_i} p_{k_i}$, $l = (p_{k_i} - p_c)$, $\overrightarrow{V_1} = p_i - p_c$, $\overrightarrow{V_2} = p_i - l$, and $\overrightarrow{S} = p_i - p_j$

    Derive α and ϕ based upon Figure 4.4.

    \# Get 6D APPF for (p_i, p_j) - concatenate f_1 and f_1

    $f_{3[(p_i,n_i),(p_j,n_j)]} = (\phi, \theta, \alpha, \beta, \gamma, \delta)$

  else

    Perform steps in ***if*** block, but for p_j as ***"Origin"***.

  end if

  \# Return $f_{3[(p_i,n_i),(p_j,n_j)]}$ to $f_{3(P_i)}$

end for

Basically, α, β, γ are the angular variations between (n_i, n_j), while δ is the spatial distance between p_i and p_j. In Euclidean geometry, each of the projections ϕ and θ is considered angle between two vectors. For example, $\angle_1\langle \overrightarrow{S}, \overrightarrow{V_1}\rangle$ and $\angle_2\langle \overrightarrow{S}, \overrightarrow{V_2}\rangle$ are equivalent to θ and ϕ respectively. These angles are derived by taking the scalar products of $(\overrightarrow{S} \cdot \overrightarrow{V_1})$ for $\angle_1$, and $(\overrightarrow{S} \cdot \overrightarrow{V_2})$ for $\angle_2$ about a point p_i in a given LSP. Mathematically, scalar products defined in this manner are homogeneous (i.e. invariant) under scaling [254] and rotation [156]. For this reason, our 2-dimensional local geometric features, ϕ, θ are considered rotation and scale invariant for 3D shapes under rigid and non-rigid transformations.

Interestingly, the pairwise points, p_i and p_j in the fully-connected graph for each LSP are not selected by a separate computational steps, because they are already part of the surflet-pair (p_i, n_i and p_j, n_j). Additionally, as a major improvement over the feature extraction techniques mentioned in Section 4.2.1, our proposed

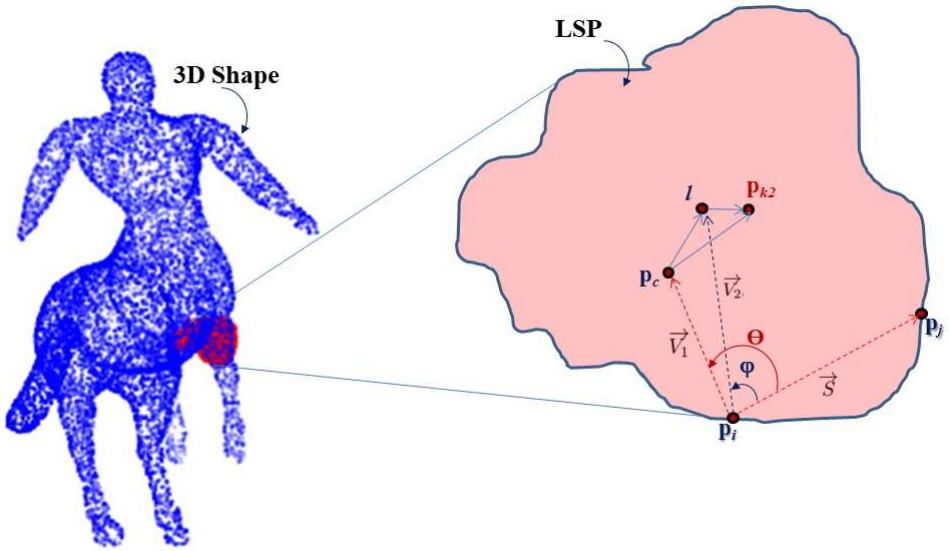


Figure 4.4: Local Surface Patch (LSP), P_i with pairwise points (p_i, p_j) as part of a surflet-pair relation for (p_i, n_i) and (p_j, n_j), with p_i being the origin. θ and ϕ are the angles of vectors projection about the origin, p_i. θ is the projection angle from vector $\langle p_i - p_j \rangle$ to vector $\langle p_i - p_c \rangle$ while ϕ is the projection angle from vector $\langle p_i - p_j \rangle$ to vector $\langle p_i - l \rangle$. The LSP centre is given by p_c, key point is given as p_{k_i} where $i = 2$. Finally, l is the vector position of $p_{k_i} - p_c$.

2-dimensional geometric features (ϕ and θ - see Figure 4.4) does not rely on estimated surface normals, making them faster and very efficient to compute. Thus, the robustness of $f_2(p_i, p_j) = (\phi, \theta)$ is unaffected by inaccuracies in estimated surface normals (if there exist any). Overall, ϕ and θ provides a unique signature for every surface patch (LSP), as they represent an extraction of unique physical (angular) measurements within each local surface region, and offers additional descriptiveness to the signature of P_i.

Final Local APPF Descriptor for Each LSP

Lastly, assuming we want to adopt the *"local"* matching approach explained in Section 3.6.1 and Section 6.5 for our shape retrieval or classification task, then a *"local"* APPFD can be computed (according to Algorithm 3 or 4) as follows: For every possible combination, q of oriented point pair, $p_i, p_j = [(p_i, n_i), (p_j, n_j)]$ in an LSP, (P_i, N_i), $q(q-1)/2$ 6-dimensional APPF: $f_3 = (f_2 + f_1)$ are locally obtained thus: $f_3(p_i, p_j) = (f_2(p_i, p_j), f_1(p_i, p_j)) = (\phi, \theta, \alpha, \beta, \gamma, \delta)$. As stated, an LSP characterised by 250 points would have $250(250-1)/2 = 31125$ APPF array which are then vertically stacked, resulting in $f_3 = [31125 \times 6]$ array. To obtain a *"local"* APPFD for the LSP, this f_3 is therefore discretized into a multi-dimensional histogram using $4, 5, 6, 7$ or 8 bins in each feature-dimension of the multi-dimensional histogram. If the number of bins (i.e. $b_{APPFD} = 7$), then the flattened histogram would give $7^6 = 117649$-dimensional single local descriptor (APPFD) per LSP, which is *normalized*. With this approach, if there are K key points (i.e. K LSPs), then there would be K *"local"* APPFDs.

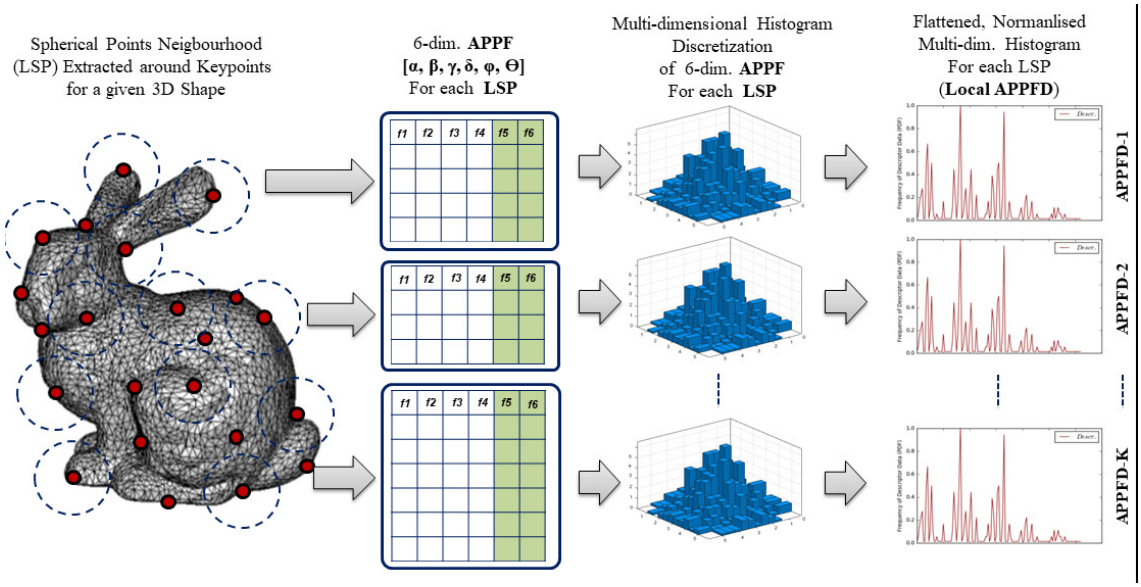


Figure 4.5: Overview of our novel 3D APPFD framework (for *"local"* matching). Rather than extract features from all surface points, we first down-sampled all surface points and used as *key points* (red), instead. 6-dimensional features are then extracted from local neighborhood (LSP) of each *key point* and binned into a multi-dimensional histogram, which is then flattened and normalized to produce final key point feature-vector tagged *"local"* APPFD. For a single 3D object, $[K \times D]$ *"local"* APPFDs, where K is the number of key points and D, the descriptor/fv dimension.

A scientific Python (i.e. SciPy) package called "binned_statistic_dd(args...)" implements the multi-dimensional histogram function, which we adopt during the binning stage (to avoid re-inventing the wheel). In addition, we optionally implement a simple function that replaces all zero bins with a common value, lower than the lowest non-zero value occurring in the histogram. The essence of this is to prevent error during descriptors' matching with metrics such as the KLD or JSD. The percentage of the non-zero values can be selected as $25\%, 50\%, 75\%, 98\%$, or any other value depending on the dataset. For all our implementations, we used $98\% \times lowestNoneZero\_val$.

For an implementation where the resulting combination of the *"local"* APPF are binned into a multi-dimensional histogram, using $b_{APPFD} = 8$ in each feature dimension of the multi-dimensional histogram, the result would then be a $8^6 = 262,144$-dimensional feature vector as the final shape descriptor. Because $S(p_i, p_j)$ as well as $A_{ug}(p_i, p_j)$ are rigid transformation invariant, then the resultant *"local"* APPFD would be transformation invariant as well. Overview of the *"local"* APPFD is shown in Figure 4.5.

Final Global APPF Descriptor for All LSP

Alternatively, if we rather chose to adopt the *"global"* matching approach explained in Section 3.6.1 and Section 6.5 (as is the case in this research) for our shape retrieval or classification task, then a *"global"* APPFD can be computed (accord-

ing to Algorithm 3 or 4) as follows: For every possible combination, q of oriented point pair, $p_i, p_j = [(p_i, n_i), (p_j, n_j)]$ in an LSP, (P_i, N_i), $q(q-1)/2$ 6-dimensional APPF: $f_3 = (f_2 + f_1)$ are locally obtained thus: $f_3(p_i, p_j) = (f_2(p_i, p_j), f_1(p_i, p_j)) = (\phi, \theta, \alpha, \beta, \gamma, \delta)$, then vertically stacked together and discretized into a multi-dimensional histogram with $bins = 7$ in each feature-dimension, flattened and normalized to give $7^6 = 117649$-dimensional single local descriptor (APPFD) per 3D shape. In another experimental implementation, the resulting combination of our local APPF were binned into a multi-dimensional histogram, using 8 bins in each feature dimension. This results in a $8^6 = 262,144$ dimensional feature vector as our final shape descriptor. Since $S(p_i, p_j)$ as well as $A_{ug}(p_i, p_j)$ are rigid transformation invariant descriptors, $APPF(p_i, p_j)$ or $APPFD(\overline{P})$ is therefore transformation invariant, where $\overline{P}$ is a normalized input point cloud. Overview of our novel APPFD is shown in Figure 4.6.

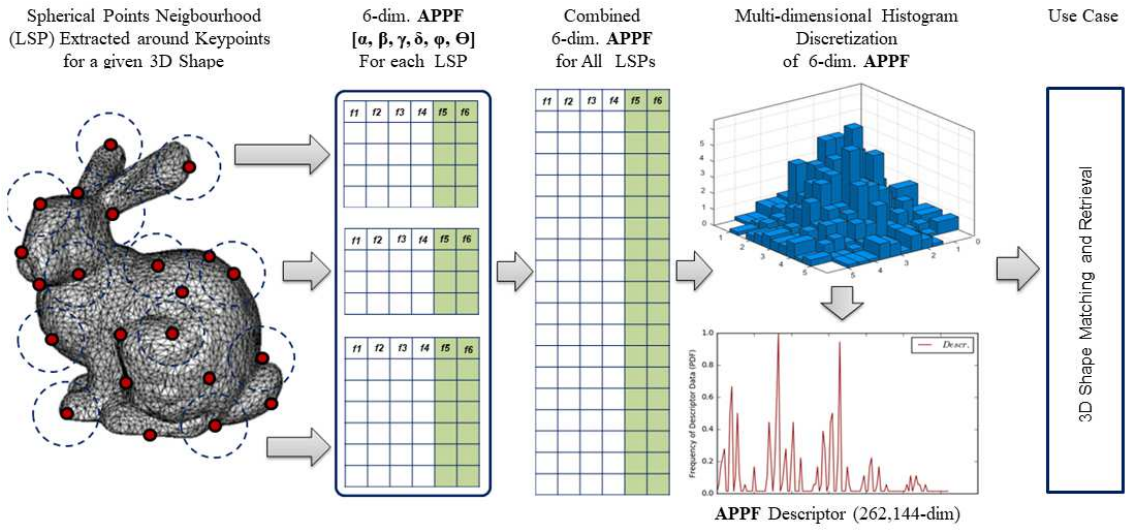


Figure 4.6: Overview of our novel 3D APPFD framework (for *"global"* matching). Rather than extract features from all surface points, we first down-sampled all surface points and used as *key points* (red), instead. 6-dimensional features are then extracted from local neighborhood (LSP) of each *key point*, and stacked together for all patches. Finally, we bin these features into a multi-dimensional histogram, which is then flattened and normalized to produce the final feature-vector tagged *"global"* APPFD. This method is effective for both full and partial 3D objects, such as those in PRoNTo (see Section 5.2.6). We show full object here only for illustration purpose.

APPF Binning Technique

Following the successful extraction of our APPF for each LSP around a key point, which are represented by six parameters $(\phi, \theta, \alpha, \beta, \gamma, \delta)$, we attempted several binning strategies in order to derive a robust and more compact representation of these features (APPF) as our final shape descriptor. For example, we tried the 2-dimensional binning strategy similar to those in [27, 77, 267, 76] for pairs of feature dimension, i.e. (ϕ, θ), (α, β), and (γ, δ), then subsequently concatenate the resultant

flattened histogram as the final descriptor. We tried 4 and 5 bins in each feature-dimension as in [252]. In addition, we tried six individual 1-dimensional bins, similar to [198], after which we concatenated the six resulting histograms to one. However, similar to [252], using 6-dimensional binning strategy yielded the best overall results. This binning technique involves using a multi-dimensional histogram approach to bucket our extracted 6-dimensional APPF, whereby, for the $b_{(APPFD)} = 8$ parameter, a final descriptor of $8^6 = 262,144$-dimensional *fv* would be produced.

Review of The APPFD Retrieval Method

The Augmented Point Pair Feature Descriptor (APPFD) is a 3D object descriptor made of local features that capture the geometric characteristics or properties of surface patches, each centred at a point (*key point*), $p_{k_i} = [x, y, z]$, which incorporates the geometrical relation between p_{k_i} and its r-nearest neighbours (i.e. surface patch, P_i at p_{k_i}). The APPFD is derived by three sub-steps for each LSP: (i) extracting 4-dimensional local PPF, $f_1 = (\alpha, \beta, \gamma, \delta)$, (ii) Augmenting f_1 to a 6-dimensional feature - the Augmented PPF, using additional 2-dimensional local angular feature, $f_2 = (\theta, \phi)$, depicted in Figure 4.4, and (iii) Binning the 6-dimensional augmented feature $f_3 = (\theta, \phi, \alpha, \beta, \gamma, \delta)$ into one or multi-dimensional histograms to yield the final local APPFD.

The PPF and our APPF involve two sets of oriented points, (p_1, n_1) and (p_2, n_2), on a 3D surface, which are typically employed to encode the underlying surface geometry. Essentially, the APPFD augments the 4-dimensional feature of [252] into a 6-dimensional feature using additional 2-dimensional feature, ϕ and θ, where ϕ and θ are geometrically the angles formed by projecting vector $\overrightarrow{S}$ onto the unit vector $\overrightarrow{V_2}$, and $\overrightarrow{S}$ to $\overrightarrow{V_1}$, respectively (see Figure 4.4). The 4-dimensional PPF used by [252] were extracted globally for all surface points. We extracted the same features, but locally for each LSP centred on their respective *key points*.

In computing the APPFD for our experimental evaluation and shape retrieval tasks, we sampled points and their normals, (P_s, N_s), where $N = 3500$ and $N = 4200$ represents the number of points, N from each 3D shape. Then, we compute K key points, $\{p_{k_i}, i = 1 : K\}$, around which LSPs, $\{P_i, i = 1 : K\}$ and their corresponding normals, $\{N_i, i = 1 : K\}$ are extracted within a specified radius, $r = 0.40 - 0.50$ for each p_{k_i}, where $P_i \in P_s$ and $N_i \in N_s$. For every possible combination, q of (p_i, n_i) and (p_j, n_j) in an LSP, (P_i, N_i), we extract the 4-dimensional surflet-pair feature, $S((p_1, n_1), (p_2, n_2)) = (\alpha, \beta, \gamma, \delta)$ defined in Equations (4.2 - 4.5), and another 2-dimensional angular feature, $A_{ug}((p_1, n_1), (p_2, n_2)) = (\phi, \theta)$. Finally, for each p_{k_i} with q combination, S and A_{ug} features are combined to obtain $q(q-1)/2$ 6-dimensional aAPPF, thus: $APPF(p_1, p_2) = (S_{P_i}(p_1, p_2), A_{ug}(p_1, p_2)) = (\alpha, \beta, \gamma, \delta, \phi, \theta)$, which are binned into a multi-dimensional histogram with $5 \leq bin \leq 8$ in each feature-dimension (for example, if $bin = 7$ and the number of locally extracted feature is 6, then $7^6 = 117,649$-dimensional *feature-vector*), flattened and normalized to give bin^6-dimensional local descriptor per 3D object. Note that $bin = b_{(APPFD)}$ parameter, which we further discussed in Chapter 4. In conclusion, our APPFD algorithm presents a shape descriptor technique where features are extracted locally

Algorithm 3: APPFD Algorithm for 3D Mesh

INPUT: Mesh M, voxel down-sampling radius: vs, number of bins for each feature dimension: b, key point's neighbourhood radius: r, moving-least-square smoothing radius: r_{mls}, number of points: N, to sample from M.

OUTPUT: D-dimensional *feature-vector* (APPFD) for point cloud $\overline{P}$. Where $D = b^f$, $b = b_{(APPFD)}$, and $f = f_{3(P_i)}$.

.

Sample N points from M to get Point Cloud, $\overline{P}$.

$\overline{P} = sampleMesh(M,\ N)$.

Apply Moving Least Square (MLS) smoothing method by [199] on $\overline{P}$.

$P = msl(\overline{P},\ r_{mls})$.

Apply affine Transform (Scale Point cloud), Section 3.2.4.

Apply uniform scale, S to p_i in P, $s.t.$ $\sqrt{\frac{1}{N}\sum_{i=1}^{N}||S_{p_i}||} = 1$.

Estimate normals, N_s for P.

$P_s, N_s = normalEstimation(P)$.

Determine 3D Key points (Voxel-grid Downsampling by [278]), Section 3.3.5.

$p_k = P_i = voxelDownsample(P_s,\ vs)$.

Initialise empty array of $f_{3(P_i)}$ for P_i

$f_{3(P_i)} = [0, 0, 0, 0, 0, 0]$.

Extract LSP (P_i) for P_s, as in Figure 4.1

for all i in $size(p_k)$ **do**

  $P_{i(i=1...K)} = lspExtraction(P_s, r)$

  compute $f_3 = [\theta, \phi, \alpha, \beta, \gamma, \delta]$ - using: ***Algorithm 2***.

  # Append f_3 to $f_{3(P_i)}$

  $f_{3(P_i)} \leftarrow f_3$

end for

$h_{DD} = histogramDD(f_{3(P_i)},\ b)$.

Replace all zero bins with 98% of lowest non-zero value in Hist.

$h_{DD} = replaceZeroBins(flattened(h_{DD}), weight = 98\%)$

$APPFD = normalize(h_{DD})$

and combined to compute a single feature vector (fv) per 3D object. We summarise the computational steps for the APPFD retrieval method for 3D mesh and point cloud input in Algorithm 3 and Algorithm 4, respectively.

- **3D Triangular Mesh Retrieval with APPFD:** Algorithm 3 is invoked to compute the APPFD for 3D triangular mesh dataset. This algorithm first produced point cloud representation of each input 3D mesh, by randomly sampling N points, P_s from the triangles that constitute facets of the mesh, and simultaneously estimating corresponding normals, N_s using Barycentric coordinate system sampling approach described in 3.2.3. Next, a MLS smoothing method is optionally applied, depending on the dataset and nature of the surfaces of 3D objects it contains. In order to ensure that two 3D models are comparable during matching, we apply uniform scale, S in all direction to all points in the input cloud, such that the RMS distance of each point to the

origin is 1. Subsequently, key points are derived and LSP selected around each key point, then local APPF are extracted to compute the final local APPFD.

- **3D Point Cloud Retrieval with APPFD:** Algorithm 4 outlines the steps we take to compute the APPFD considering point cloud as input data. Similar to the steps in Algorithm 3, we adopt the Voxel-grid down-sampling approach, which is found to return a sub-sample of the entire point cloud that accurately represent the entire topology and geometry of our shape, to determine our interest points (key points), instead. The rest of the implementation steps for this algorithm are the same with those in Algorithm 3.

Algorithm 4: APPFD Algorithm for 3D Point Cloud

INPUT: Point Cloud: $\overline{P}$, voxel downsampling radius: vs, number of bins for each feature dimension: b, key point's neighbourhood radius: r, moving-least-square smoothing radius r_{mls}.

OUTPUT: D-dimensional *feature-vector* (APPFD) for point cloud $\overline{P}$. Where $D = b^f$, $b = b_{(APPFD)}$, and $f = f_{3(P_i)}$.

.

Apply MLS smoothing method by [199] on $\overline{P}$.

$P = msl(\overline{P},\ r_{mls})$.

Apply affine Transform (Scale Point cloud), Section 3.2.4.

Apply uniform scale, S to p_i in P, $s.t.$ $\sqrt{\frac{1}{N}\sum_{i=1}^{N}||S_{p_i}||} = 1$.

Estimate normals, N_s for P.

$P_s, N_s = normalEstimation(P)$.

Determine 3D Key points (Voxel-grid Downsampling by [278]), Section 3.3.5.

$p_k = P_i = voxelDownsample(P_s,\ vs)$.

Initialise empty array of $f_{3(P_i)}$ for P_i

$f_{3(P_i)} = [0, 0, 0, 0, 0, 0]$.

Extract LSP (P_i) for P_s, as in Figure 4.1

for all i in $size(p_k)$ **do**

  $P_{i(i=1...K)} = lspExtraction(P_s, r)$

  compute $f_3 = [\theta, \phi, \alpha, \beta, \gamma, \delta]$ - using: ***Algorithm 2***.

  # Append f_3 to $f_{3(P_i)}$

  $f_{3(P_i)} \leftarrow f_3$

end for

$h_{DD} = histogramDD(f_{3(P_i)},\ b)$.

Replace all zero bins with 98% of lowest non-zero value in Hist.

$h_{DD} = replaceZeroBins(flattened(h_{DD}), weight = 98\%)$

$APPFD = normalize(h_{DD})$

4.2.2 Histogram of Global Distances (HoGD)

For this implementation, we adopt a simple approach for the HoGD, inspired by the notion that a shape is represented by a discrete set of points, sampled on its surface,

which forms the external, as well as internal, contour of the shape. We then consider a set of normalized vectors $\delta_i = \|P_c - p_i\|_i$ between the centroid $P_c = \frac{1}{N}\sum_{i=1}^{N} p_i$ of a given 3D object to all other points, $p_i \in P$, on its surface, where P and N are the entire points and number of points, respectively, that makes up the point cloud of a given shape. Such normalized vectors δ_i are regarded as global features whose distribution (histogram) is capable of expressing the configuration of the entire shape relative to its centroid, and is a rich description of the global structure of the shape. We bin these global features into a 1-dimensional histogram, with number of bins $\approx \sqrt{N}$, integer, i.e. $\sqrt{P} \approx 65$ bins, normalized to give HoGD, which is very fast and straightforward to compute, with $P = 3500$ and 4200, as in APPFD. We provide an overview of the HoGD in Figure 4.7.

The computation steps for the HoGD global descriptor for either 3D triangular mesh or point cloud is provided in Algorithm 5. It is important to note that the robustness of this algorithm is directly proportional to the number, N of vertices or points contained in the input data, whereby, for 3D objects with fewer number of points, such as those in the PRoNTo dataset with an average of 3,500 - 4,000 points per 3D scan object (see Section 5.2.6), the effect of the HoGD would be negligible. Alternatively, for 3D objects with very high number of points or vertices, such as the SHREC'18 Protein models with an average of 100k vertices/points (see Section 5.2.7), the HoGD descriptor robustness is noticeable. However, in line with the objectives of this thesis and the characteristics of a good 3D shape descriptor provided in Section 2.3.1, we limit our implementations of the HoGD method to a very minimal set of at most $N = 4,500$ 3D surface points.

The primary goal of the HoGD method is to provide additional capability and *"global"* surface characteristics to our *"local"* APPFD method, which is most suitable for describing local surface geometry. Practically and theoretically, we expect that the resultant outcome of combining these two retrieval methods would provide a more robust *"hybrid"* 3D shape retrieval method that is most suitable for describing complex 3D surfaces.

4.2.3 Hybrid Augmented Point Pair Signature (HAPPS)

As discussed in Section 2.4, shape descriptors can be categorised into two main groups: *local* and *global*. Combining two or more descriptors (e.g. local-local, local-global, or global-global) yields a third category, the *hybrid* descriptor, which is aimed at improving the resultant performance of the combined descriptors. The Hybrid Augmented Point Pair Signature (HAPPS) is a 3D shape descriptor in the third category, computed from a combination of two separate descriptors: (i) *local* Augmented Point Pair Feature Descriptor (APPFD), and (ii) *global* Histogram of Global Distances (HoGD) and/or Multi-view 2D Projection (M2DP) [80] descriptors, each of which are computed using hand-crafted features extracted from 3D surface. Details of the APPFD are provided in Section 4.2.1, while details of the HoGD can be found in Section 4.2.2. We already provided a brief overview of the M2DP descriptor in Section 2.4.1. During experimental evaluations, we use the parameter, $N = 3,500$ and $N = 4,200$ as in previous cases with APPFD and/or HAPPS-1.

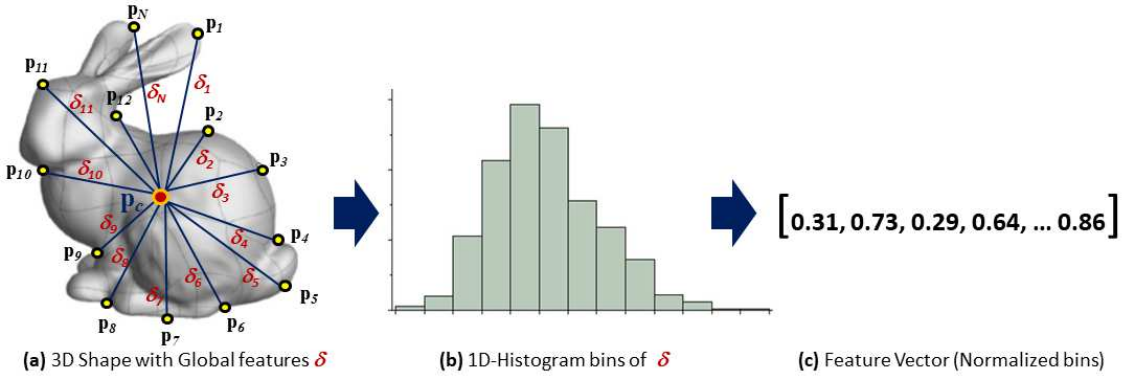


Figure 4.7: Overview of Histogram of Global Distances (HoGD), which involves binning the distances, $\delta = \|P_c - p_i\|$ between the centroid P_c and all other points p_i on the surface of a given 3D object. **(a)** shows a 3D Stanford Bunny with its centroid P_c and N points $\{p_i, i \in 1...N\}$ on its surface, with normalized distances, δ as global features. **(b)** represents a 1D-histogram of these features, while **(c)** represents the final feature vector, which is the normalized histogram bin counts, and used as the global descriptor for the Bunny.

The HAPPS is simply an improvement over the APPFD, aimed at achieving better retrieval performances. Although the APPFD is capable of robustly representing 3D shapes, we found that for some retrieval problems, such as the Protein shape retrieval challenges (SHREC'18 Protein dataset in Section 5.2.7, SHREC'19 Protein dataset in Section 5.2.8, and SHREC'20 Protein dataset in Section 5.2.9) a closer inspection of protein shapes for these respective retrieval challenges reveal identical local surface characteristics and somewhat uniqueness in the global structures of these Protein models, hence the need to extend the capability of the APPFD and effectively capture both local and global characteristics of the these types of 3D objects (i.e. Protein models in this case). Consequently, we separately combined two global 3D descriptors: The HoGD and the M2DP with the APPFD to derive two variants of hybrid descriptor, which we call the Hybrid Augmented Point Pair Signatures (HAPPS), referred to as HAPPS-1 and HAPPS-2, i.e. hybrid descriptors formed by combining local APPFD with global HoGD and M2DP, respectively. Alongside the APPFD, the HAPPS algorithm was first introduced in our publication [175] and recorded extremely high-performance scores across several 3D benchmark datasets. Figure 4.8 presents an overview of the HAPPS algorithms.

The HAPPS-1 Descriptor Implementation

Our novel hybrid descriptor (the HAPPS-1) is derived by combining the final flattened and normalised multi-dimensional histogram of the *local* APPFD with the normalized 1-dimensional histogram of the *global* HoGD. The resultant descriptor is therefore a $15,625 + 65 = 15,690$-dimensional, or a $117,649 + 65 = 117,717$-dimensional, or a $262,144 + 65 = 262,209$-dimensional *feature-vector*, depending on which value is used as the APPFD bin size, b_{APPFD}. For example, using $b_{APPFD} = 5$, we obtain a final APPFD descriptor of $5^6 = 15,625$-dimensional feature-vector, and using $b_{APPFD} = 8$, we obtain a final APPFD descriptor of $8^6 = 262,144$-dimensional feature-vector. The bin size for the HoGD global descriptor is approx-

Algorithm 5: HoGD Algorithm for 3D Triangular Mesh or Point Cloud

INPUT: Triangular Mesh: M, or Point Cloud: $\overline{P}$, with N number of vertices or points in M or $\overline{P}$, respectively.

OUTPUT: 65-dimensional *feature-vector* (HoGD) for M or $\overline{P}$. Where $65 \approx \sqrt{N}$.

if $InputData = M$ **then**
  //Sample N points from M to get Point Cloud, $\overline{P}$.
  $\overline{P} = sampleMesh(M,\ N)$
  Apply uniform scale, S to p_i in $\overline{P}$, $s.t.$ $\sqrt{\frac{1}{N}\sum_{i=1}^{N}||S_{p_i}||} = 1$
else
  if $InputData = \overline{P}$ **then**
    Apply uniform scale, S to p_i in $\overline{P}$, $s.t.$ $\sqrt{\frac{1}{N}\sum_{i=1}^{N}||S_{p_i}||} = 1$
  end if
end if
//Compute Point Cloud Centroid: P_c.
$P_c = \frac{1}{N}\sum_{i=1}^{N} p_i$, where $p_i \in \overline{P}$.
//Compute normalised vector: δ_i.
$features = [...]$
for all p_i in $\overline{P}$ **do**
  $\delta_i = \|P_c - p_i\|_i$
  $features \leftarrow \delta_i$
end for
$hist = histogram(features,\ b = \sqrt{N})$
$HoGD = normalize(hist)$

imately $b_{HoGD} = 65$. Essentially, the five parameters: N, vs, r, $b_{(APPFD)}$, and $b_{(HoGD)}$ affects the overall the performance of the HAPPS retrieval method, with the $b_{(APPFD)}$ and a bit of $b_{(HoGD)}$ parameters having the most influence, including that they determine the size (compactness) of the final descriptor. These parameters are adequately explained in Section 5.3.1. Also, see Table 5.11 for a typical example of the parameter settings (i.e. how these parameter values are combined) for the HAPPS retrieval method.

Matching between two 3D objects is done simply by returning the spatial distance between *feature-vectors* of the two objects, using any of the distance metrics described in Section 2.2.3. In Section 5.3, we have provided experimental evaluations and retrieval results using the HAPPS-1 retrieval method, and first compared these results with those of several state-of-the-art retrieval methods in one of our publications [175]. In that work, we first tested the robustness and descriptiveness of the HAPPS-1 retrieval method by evaluating its retrieval performances on three publicly available datasets from various standard SHREC benchmarks datasets of 3D objects. Table 4.2 identifies some basic properties of these datasets. However, we refer the reader to Section 5.2 for more details regarding a comprehensive list of 3D object datasets we evaluate our retrieval method against in this thesis. The

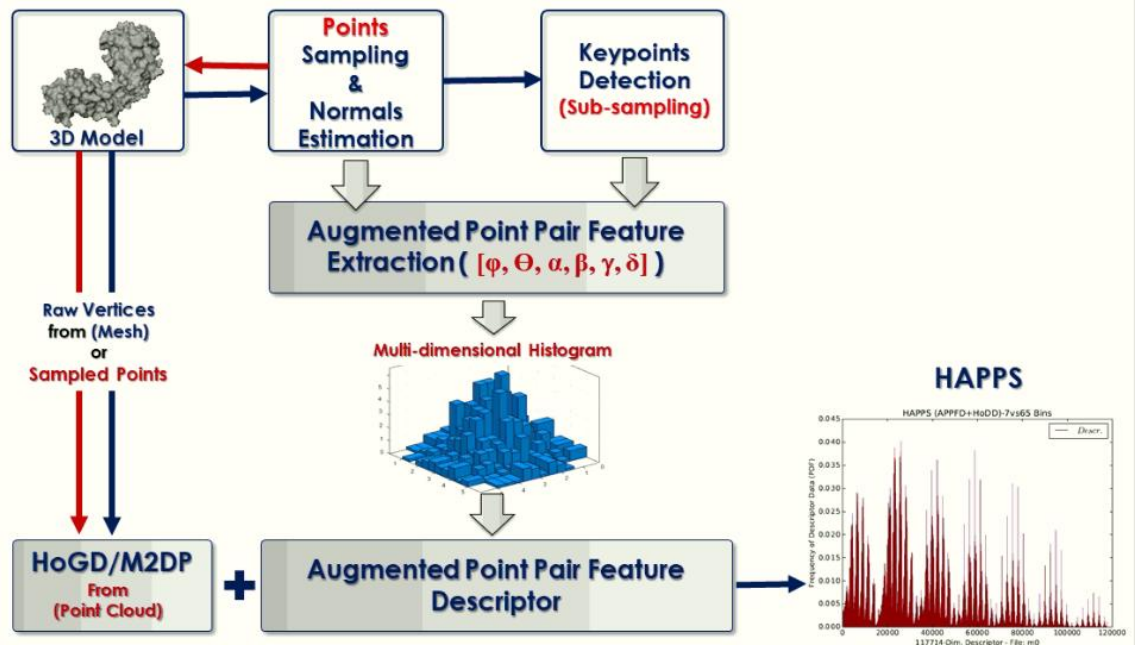


Figure 4.8: Overview of HAPPS algorithm.

choice of these datasets for testing the HAPPS-1 retrieval method was due to their diverse acquisition technique, diverse retrieval challenges, diverse surface representation, shape quality and possible application scenarios.

No.	Dataset (Year)	Quality	Size	Model	Type	Use Case
1	SHREC-PSR (2018) [121]	Very High	2,267	3D	Polygon Mesh	Retrieval
2	PRoNTo (2017) [269]	Poor	100	3D	Point Cloud	Retrieval
3	SHREC-10 (2010) [140]	High	200	3D	Polygon Mesh	Retrieval

Table 4.2: Three different 3D shape retrieval benchmark datasets used to evaluate HAPPS method.

The HAPPS-2 Descriptor Implementation

Basically, the HoGD *"global"* descriptor already has that characteristics (computational efficiency, simplicity, speed, etc.) needed to form a *"hybrid"* descriptor in conjunction with APPFD. This is confirmed by the successes recorded with the HAPPS method on several different 3D benchmark datasets, such as the SHREC'18 Protein dataset. However, the choice of another global descriptor (preferably the M2DP, due to its success in loop-closure detection [80], compactness and high computational efficiency - speed) to combine with our APPFD to form another variant of a hybrid descriptor, the HAPPS-2, was to be able to determine, with further evidence, the influence and robustness of the APPFD local descriptor, or to what

extent these global descriptors contributed to the overall HAPPS retrieval method. The M2DP descriptor is extremely fast to compute and returns a 196-dimensional feature-vector, while the APPFD descriptor returns $117,649$-dimensional *fv*, depending on the parameter value for the $b_{(APPFD)}$ bins. Therefore, combining these two descriptors, the resultant HAPPS-2 descriptor becomes a $117,649 + 196 = 117845$-dimensional *fv*, considering that $b_{(APPFD)} = 7$. See Figure 4.8 for an overview of the general HAPPS descriptor algorithm or retrieval method. Each of these final descriptors (*local* and *global*) are first normalised within the range $[0,\ 1]$, before concatenation. This ensures that their resultant *hybrid* descriptor is within the same range, before matching. However, it would be interesting to further investigate the impact of combining two or more descriptors with variable lengths (i.e. $117,649$ and 196), using different similarity metrics and selected datasets.

HAPPS Descriptor Experimental Settings and Running Time

All the parameters used for the implementation of our retrieval methods (APPFD, HoGD, and HAPPS) are presented in Table 4.3. In sample experimental run-A and run-B, we show the parameter settings for the HAPPS-1 and the HAPPS-2 retrieval methods, respectively. Note that the $b_{(HoGD)}$ parameter does not apply to the HAPPS-2 method. These two parameters of the APPFD: (i) vs, a parameter that determines the size of an occupied voxel grid [279], thus, the number of returned *key points*, during point cloud down-sampling (see Section 3.3), and (ii) r, which controls the size of LSP for each *key point*, influences the overall performances of the HAPPS retrieval algorithms. For example, r is directly proportional to the sizes of the LSP selected from the input 3D surface. This implies that the greater the value of r, the larger the sizes or number of local point neighbourhood of each LSP selection, and vice versa. Alternatively, vs is inversely proportional to the number of sub-sampled points (*key points*), which means increasing the value of vs reduces the number of key points, and vice versa. During our experimental implementations, several N, r and vs parameter value combinations were tested in order to find best possible combinations that yields optimum results. Depending on the size of dataset, the computational time and memory can be affected by these parameter settings. Therefore, we carefully selected the configurations summarized in the following tables, for the respective experimental runs with several different datasets: Table 5.4, 5.8, 5.11, 5.13, 5.16, 5.17, 5.19, and 5.25. Further analyses regarding these parameter settings are summarised in Section 5.3.3. These experimental runs, their results for respective benchmark datasets/retrieval challenges, and discussions are presented in Chapter 4.

The HAPPS algorithms are implemented in Python 3.60 and depending on the size of the dataset, all experiments are carried out on either of our: **(i)** HP Pavilion, Windows 10 Home, x64-based PC, with the following configurations. Processor: Intel(R) Core(TM)i3-5157U CPU @ 2.50GHz 2.49GHz. Installed memory(RAM): 8.00GB, and **(ii)** Windows 7 desktop PC with Intel Core i7-4790 CPU @ 3.60GHz, 32GB RAM. On the first machine, it took an average of 25 *Seconds* to compute the HAPPS-1 for a single 3D object, using the parameters mentioned in the preprocessing steps (see Sections 3.2 to 3.3.5). Alternatively, it took an average of 23 *secs* and

Dataset:	SHREC	Parameter Settings					
Experiment(s)	**Algorithms**	N	r	vs	$b_{(APPFD)}$	$b_{(HoGD)}$	**Dist.Metric**
run-A	HAPPS-1	3500	0.50	0.32	7	65	Cosine
run-B	HAPPS-2	3500	0.50	0.32	7	n/a	Cosine

Table 4.3: Typical parameter settings for the APPFD, including the HAPPS-1&2 methods having two experimental runs: run-A and run-B, having exactly the same parameter values. The parameters: P, r, vs, $b_{(APPFD)}$, and $b_{(HoGD)}$ are explained in Section 5.3.

45 *secs* to compute HAPPS-1 and HAPPS-2, respectively, about 1 *Sec* to extract (P_s, N_s), and roughly 0.3 *secs* each, to compute the HoGD and M2DP per 3D object on the second machine.

4.2.4 Agglomeration of Local APPFD with Fisher Kernel and Gaussian Mixture Model (APPFD-FK-GMM)

The Gaussian Mixture Model (GMM)

The GMM is a type of clustering algorithm, where each cluster is modelled according to a different Gaussian distribution, hence the name, Gaussian Mixture Model. The GMM provides a flexible and probabilistic approach to data modelling using soft assignment of data points into clusters, as opposed to the hard assignments approach used by the k-means algorithm. The probabilistic approach of this model entails that a single data point could be generated by any of the Gaussian distributions with a corresponding probability. Generating a GMM-based model is possible by first introducing a latent variable, ξ for each data point (that defines which Gaussian generated a particular data point), with the assumption that each data point was generated using some information about ξ. To achieve this, the Expectation Maximisation (EM) algorithm is involved. We refer the reader to [161] for an in-dept information regarding the EM algorithm as it is beyond the scope of this thesis. In a nutshell, the EM algorithm is an efficient density estimation technique for missing data, that is most-commonly used with clustering algorithms, such as the GMM. In addition, EM is a technique for performing a challenging maximum likelihood estimation on data, given its ξ variables, by first estimating the values for the latent variables, then optimising the model, and repeating these two processes until the model converges.

The Fisher Kernel (FK) Framework

The FK, defined as $K(M_i, M_j) = V_{M_i}^T \mathcal{I}^1 V_{M_j}$ (where $\mathcal{I}$ is the Fisher information matrix), is a function used in statistical classification to compare the similarity between two objects, M_i and M_j based on a set of measurement for each object and a statistical model (i.e. Gaussian Mixture Model, for example). Fisher Vector (FV) can be used to combine the power of generative statistical model, such as the hidden Markov model and discriminating method, such as the support vector machine. The Fisher information measures the amount of details in an observed random variable, M about an unknown parameter, ρ of a distribution that models M, and used to calculate the covariance matrices associated with maximum-likelihood estimates.

The FK uses the Fisher score, which is defined as: $V_M = \nabla\rho \log P(M|\rho)$, where ρ is a set of parameters, and the function converting ρ to $P(M|\rho)$ is a log-likelihood of the probabilistic model. In IR the FK is commonly applied to image descriptors for classification or retrieval problems. It has the advantage of producing a compact and dense representation, which is desirable in shape classification and retrieval problems. In our implementation, we adopt the FK framework described in [93].

Improving The APPFD or HAPPS Descriptor

Through experimental evaluations, we demonstrate that the APPFD and the HAPPS retrieval methods have excellent retrieval accuracies on several benchmark datasets (see Sections 5.3 and 5.4). However, the lengths of their respective final *fv* is very high-dimensional, as result of the $b_{(APPFD)}$ parameter and the number of feature-dimension in the APPF, which is 6-dimension. That is, the higher the value of both the $b_{(APPFD)}$ parameter and the APPF, the better the retrieval performance as proven by our experimental evaluations in Chapter 4. For a 6-dimensional APPF and $b_{(APPFD)} = 8$, we get $8^6 = 262,144$-dimensional *fv*, which is remarkably high dimensional, although very descriptive. Similarly, for a 5-dimensional APPF and say $b_{(APPFD)} = 5$, we get $5^5 = 3,125$-dimensional *fv*, which is comparatively incredibly low dimensional, but very poorly descriptive for a given 3D object, using this method. Considering the former case, where the final desirable descriptor (*fv*) from the APPFD or HAPPS retrieval method is remarkably high, it takes longer to match the shape descriptors of two 3D objects, and for a larger dataset, such as the SHREC'14, SHREC'18, SHREC'19 datasets, etc., computational time and memory would most likely be affected, depending also on the machine's configuration. This motivated further investigation into a technique that could further shorten the final APPFD and/or HAPPS *fv*, to produce a more compact final descriptor without losing retrieval accuracies, which led to the agglomeration of local APPFD with FK and GMM (APPFD-FK-GMM) approach: a method which aims to aggregate *d*-dimensional local descriptors into a single vector representation.

Considering the success and contributions of the FK and GMM approach in the 2D domain [93, 218], we adopted the APPFD as the basis for the new 3D shape retrieval method: Agglomeration of local APPFD with Fisher Kernel and Gaussian Mixture Model (APPFD-FK-GMM). The two main stages of the APPFD-FK-GMM algorithm are as follows: (i) Computing local APPFD for selected key points, and (ii) Key points APPFD aggregation with FV and GMM. We have already provided sufficient details regarding feature extraction technique and shape descriptor computation steps for the APPFD in Section 4.2.1. Therefore, in this section we only focus on the second stage, which is the agglomeration techniques for locally extracted APPFD using the fisher kernel and Gaussian mixture model. Precisely, the FV computation approach involves an initial step of training a GMM, as a generative probabilistic model, given aggregated key points local APPFDs for all database 3D objects, including the application of FK technique with the help of this trained model and local APPFDs for a given 3D object to derive a single signature, the APPFD-FK-GMM for that 3D object. The final stage of this process, therefore, consists of computing a global FV for each input 3D object given their key points APPFDs. Figure 4.9 shows the processing pipeline of the APPFD-FK-GMM algorithm.

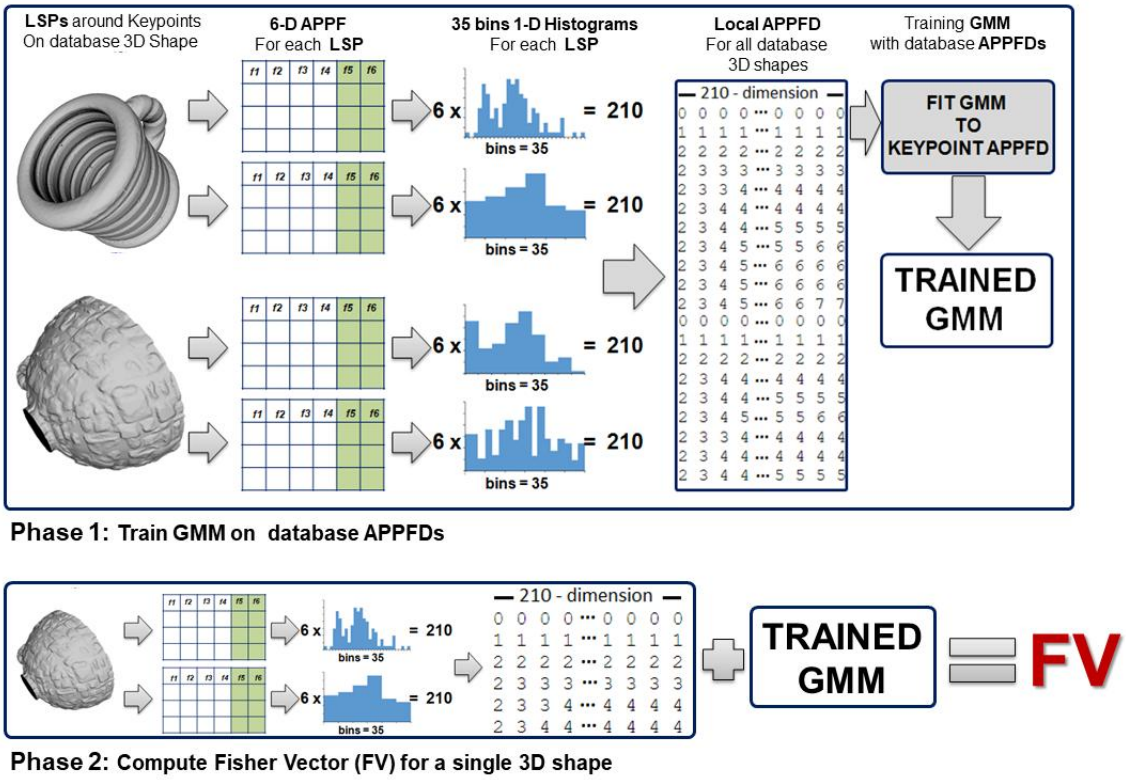


Figure 4.9: Overview of APPFD-FK framework.

APPF Binning Technique for The APPFD-FK-GMM Method

Unlike the initial binning technique for the APPFD algorithm (see Section 4.2.1), where we compute a multi-dimensional histogram of the extracted 6-dimensional APPF to produce a final signature per 3D object, which is very high in dimension, and consumes additional computational time during descriptor matching, the binning technique adopted for the APPFD algorithm used in the APPFD-FK-GMM method is very efficient and involves binning each feature dimension of the extracted 6-dimensional APPF into 1D histogram of between 8 to 40 bins, and concatenating the final $1D$ histograms, which results in a very low dimensional final local fv (say 210-dimension, if $b_{(APPFD)} = 35$) for each key point and/or LSP. These local LSP descriptors are combined for each 3D object and for all the objects in the dataset and used to train a GMM.

Fitting Our Data To Gaussian Mixture Model (GMM)

As already stated in "APPF Binning Technique" section, for each 3D object and for all the objects in the dataset, the local LSP descriptors are combined together to train a GMM (i.e. *"fit"* a GMM to our data). The SkLearn library in Python provides a class to train a GMM on sample distribution in order to estimate the parameters of the distribution with the help of the EM algorithm. This training relies on a parameter, G which specifies the number of underlying processes used to generate the data when defining the model. Assuming the number of processes are unknown, (such as in our case where we are using an unsupervised approach and a variable number of local APPFD for each 3D object) a range of different values of

G could be tested, using either of Akaike or Bayesian Information Criterion (AIC or BIC), to determine which of the values would converge the model, then the model with the best fit is chosen. In our implementation, we tested the following values for $G = [8, 10, 12, 15, 20, 24, 40]$ and found $G = 10$ to give good approximation. Finally, we trained our GMM with 10 Gaussian, using diagonal covariances for all experimental runs.

The Fisher Vector (FV): APPFD-FK-GMM

The FV encoding on the other hand, is an approximate and improved case of the FK that stores the mean and the covariance deviation vectors per component k in a GMM and each element of the local shape descriptors together. Following the example in [218], let $I = (bx_1, \ldots, bx_N)$ be a set of d-dimensional feature vectors (e.g. APPFD local descriptors) extracted from a 3D surface. Let $\rho = (\mu_k, \Sigma_k, \pi_k : k = 1, \ldots, K)$ be the parameters of a GMM fitting the distribution of descriptors. The GMM associates each vector bx_i to a mode k in the mixture with a strength given by the posterior probability:

$$q_{ik} = \frac{\exp\left[-\frac{1}{2}(bx_i - \mu_k)^T \Sigma_k^{-1}(bx_i - \mu_k)\right]}{\sum_{t=1}^{K} \exp\left[-\frac{1}{2}(bx_i - \mu_t)^T \Sigma_k^{-1}(bx_i - \mu_t)\right]}.$$

For each mode k, consider the mean and covariance deviation vectors

$$u_{jk} = \frac{1}{N\sqrt{\pi_k}} \sum_{i=1}^{N} q_{ik} \frac{x_{ji} - \mu_{jk}}{\sigma_{jk}},$$
$$v_{jk} = \frac{1}{N\sqrt{2\pi_k}} \sum_{i=1}^{N} q_{ik} \left[\left(\frac{x_{ji} - \mu_{jk}}{\sigma_{jk}} \right)^2 - 1 \right].$$

where $(j = 1, 2, \ldots, d)$ spans the vector dimensions. The FV of 3D object I is the stacking of the vectors bu_k and then of the vectors bv_k for each of the K modes in the Gaussian mixtures:

$$\Phi(I) = \begin{bmatrix} \vdots \\ bu_k \\ \vdots \\ bv_k \\ \vdots \end{bmatrix}.$$

According to [93], the FV describes how the set of descriptors deviates from an average distribution of descriptors, modeled by a parametric generative model. In summary, the FV encoding aggregates a large set of local shape descriptors into a high-dimensional vector representation by fitting a generative probabilistic model (i.e. the GMM) to the descriptors and encoding the derivatives of the log-likelihood of the model with respect to its parameters, ρ.

Finally, given the trained GMM-based model from the previous step (see Section 4.2.4) and local key points APPFDs for a given 3D object, the Fisher Kernel

described in Section 4.2.4 is used to compute a final global FV for the 3D object, which is L_2 and power-normalized. We achieved this with the help of [111]. However, the FV derived with the FK are high-dimensional, which can impact computational speed during matching of the FVs of any two 3D objects, including that such high-dimensional vectors cannot be efficiently indexed. Then application of a linear dimensionality reduction technique, such as the PCA significantly reduced the dimensionality of the final descriptor, making it more compact, while still retaining its robustness. For example, given local APPFDs with 210 and 15,625 dimensions, a FV with 4,210 and 312,510 dimensions, respectively are returned, which represent a single 3D shape, and applying the PCA to either of the 4,210 and 312,510-dimensional FVs while retaining 99% of their information reduces the dimensions to 162 and 186 respectively, and still yield close matching results/retrieval accuracies. The primary goal of applying a dimensionality reduction on the final global descriptor returned by the APPFD-FK-GMM algorithm was to further produce a final shape descriptor that is more compact, including to investigate the descriptor robustness thereafter, in line with the characteristics of an appropriate shape descriptor outlined in Section 2.3.1.

4.3 Evaluation Techniques

Performance evaluation is an essential part of information retrieval (IR) [206], such as 3D-CBSR. It deals with processes that objectively assess the ability of retrieval systems to meet users' needs. Evaluating the performances of shape retrieval methods is of significant importance in 3D-CBSR to judge the similarity measurements between 3D shapes. As described in Section 2.2.4, a complete 3D-CBSR system involves four main stages, thus: feature extraction, shape descriptor construction, indexing, matching and retrieval. Ideally, human judgement (user-based) can be used to evaluate the accuracy and performance of the retrieval system by manually comparing how similar the retrieved shapes are to the query shape. However, the user-based performance evaluation approach is highly ineffective for several reasons, including: (i) When the retrieval results are undesirable, due to, perhaps a bad (i.e. non-robust) shape descriptor, the entire CBSR system may have to be re-implemented following descriptor re-implementation. (ii) It would be practically ineffective to deploy reliable real-time applications for retrieval or classification tasks if there is no way to measure performance before deploying the final CBSR system. Therefore, there is the need for an effective performances' evaluation mechanism.

Alternatively, system-based performance evaluation approach [99] involves the use of software tools to quantify how well a retrieval method ranks retrieval results that are relevant to the query shape with the abstraction that good evaluation performance is subject to good ranking results. In this thesis, we adopt the system-based performance evaluation approach for performance evaluation measurements of our retrieval methods.

4.3.1 Performance Evaluation Algorithm (Tools)

In this thesis, we employ commonly used IR tools, which are described in Section 4.3.3, to evaluate retrieval performance of our shape descriptors. The main

goal of performance evaluation in IR system is to determine how different aspects of the system (3D shape retrieval method) can retrieve shapes that are relevant to the query shape. To effectively achieve this, system-based evaluation approach, which involves the use of software tools, is needed. Granted, most of these evaluation tools have already been developed by researchers. Although many of the available software tools are only suitable for performance evaluation of other areas of IR, such as texts, audios, and 2D images retrieval, RETRIEVAL (http://retrieval.ceti.gr) [90], a web-based integrated information retrieval performance evaluation platform, and other publicly available source codes by the Princeton 3D Shape Benchmark (PSB) [216], are a few examples of existing software tools for performance evaluation of 3D retrieval methods. The PSB, for example, provides source codes for several performance evaluation metrics such as Nearest Neighbour (NN), Precision-recall (PR), First Tier (FT), Second Tier (ST), Discounted Cumulative Gains (DCG), etc. See Section 4.3.3 for more on evaluation metrics. These codes can also be customized to suit specific needs. In addition, the annual competition for 3D SHape REtrieval Contest, SHREC [140, 132, 142, 269, 121, 157], also provides different source codes for each of the retrieval challenges the competition addresses. Essentially, SHREC organises several annual retrieval tracks, since 2006 [244] till present [122, 163], with the goal of quantifying the performances of 3D shape retrieval methods over a variety of application domains.

Unlike the user-based performance evaluation approach which is more challenging and costly due to the need for methods to be equally developed, trained and followed by a user interface of similar level of functionality [90], adopting system-based approach (which involves the use of already existing software tools) for performance evaluation takes away computational burden from the user (i.e. researcher). Rather than re-invent the wheel, we adopted the respective software evaluation tools provided by the different benchmark datasets and SHREC for which our retrieval methods are evaluated upon.

4.3.2 Performance Evaluation Algorithm Inputs

Following the successful implementation of shape retrieval method (see Section 4.2) for a given benchmark dataset, retrieval results such as (dis)similarity matrix (DM), Ranked List (RL), and Binary Ranked Lists (BRL) data structures, are computed. A combination of these results with a *Classification Index* (CI) or *Ground Truth* (GT) data are needed as inputs to an evaluation system (software tool, see Section 4.3.1) for performance evaluation of the retrieval method. Details about these retrieval results and/or data structures are presented in the following sections.

Distance/(Dis)similarity Matrix (DM)

Distance/(Dis)similarity Matrix, also known as self-similarity or distance matrix (DM), represents the dissimilarity of all pairs of shapes in the database. For N shapes, the DM is a sequence of N by N floating point values (between 0.00 and 1.00), where the value at position $i*N+j$ represents the distance, i.e. (dis)similarity between shapes i and j. A distance value of 0.00 returned from matching any two shape descriptors (query descriptor and database descriptor) indicates that the descriptors (matched shapes) are identical, while larger values, say 0.82, indicate

greater dissimilarity between the two matched descriptors representing two different 3D shapes. However, all distance values must be positive and can be arbitrarily large(for un-normalised descriptors). Typical examples of self-similarity matrices are presented in Figures 5.15 and 5.17.

It is helpful to examine the general structure produced by each type of shape descriptor and its corresponding *Distance Matrix*, besides generating other plots like the PRC. For any given DM, the cooler (blue) colour indicates *Closeness* or greater similarity, while the hotter (red) colour indicates *Furthest* or greater (dis)similarity. For all the 3D benchmark datasets we experimentally evaluated our retrieval methods against in this thesis, shapes are categorized into groups known as *Classes*, therefore, in all the DM plots (see Section 5.3.1, Figures 5.18, 5.15), for example, we expect to see a contiguous sets of C-rows (and also C-Columns), each corresponding to a shape class, such as Cars, fishes, chairs, human, etc., where C is the number of classes or groups for a given dataset. For example, $C = 107$ for the SHREC'18 protein shapes dataset [121] and $C = 10$ for the PRoNTo dataset [269]. The fifth column in Table 5.1 reveals the value of C for all the other datasets evaluated in this study.

In the distance/(dis)similarity matrix plot for each retrieval method and benchmark dataset evaluated, we expect to see a $C \times C$ pixel blue square for each class/group along the diagonal, with hotter colors everywhere else in the same row. This shows that a particular group is close to itself and far from others when the shape descriptor signatures are compared using spatial distance metric, like Euclidean-distance, cosine-distance, etc.

Ranked Lists (RL)

Given a set of users' queries, a shape retrieval algorithm searches through the benchmark database of 3D shapes and returns an ordered list of responses called the ranked list(s), RL. The greater the ranked position (value) of a relevant item (shape) in the RL, the less valuable such item (shape) is to the user, because it is less likely that the user will examine such item (shape) due to time, effort, and cumulated information from objects already seen (refer to E-measure in Section 4.3.3). The evaluation of the algorithm then is transformed to the evaluation of the quality of the ranked list(s).

Binary Ranked Lists (BRL)

In binary ranked lists, the binary relevance values of 1 and 0 are used instead, to replace the retrieved item (shape) ID with the relevance value, where 1 represents relevance of the ranked item (shape) to the query item (shape), and 0 irrelevant.

Classification Index (CI)/Ground Truth (GT)

For all the benchmark datasets evaluated in this thesis except ShapeGoogle (see Section 5.2 and Table 5.1), a Classification Index (CI)/GT file (i.e. **.cla*) is provided. The CI/GT file specifies two important pieces of information for the performance

evaluation algorithm that returns PR scores and other performance evaluation statistics like NN, FT, ST, E, and DCG, etc. First, the classification file indicates the order that shapes appear in the rows and columns of the (dis)similarity matrix, DM. Second, it provides a grouping of the shapes so that the performance evaluation algorithm can determine relevant matches (shapes belonging to the same class as the query shape) from irrelevant ones. For more details on classification index or Ground Truth file format for 3D shapes retrieval, refer to the Princeton Shape Benchmark (PSB) classification file format documentation in [1].

The *"performance evaluation algorithm"* uses a CI and a DM computed with any shape retrieval method to produce statistics and visualisations that facilitate evaluation of the retrieval results. The retrieval results determine how many of the top-ranked shapes are from the same class as the query shape.

4.3.3 Performance Evaluation Metrics

In information retrieval (IR), evaluation metrics are used to measure how well the retrieved results satisfy the user's query. Although the performance evaluation of a shape retrieval method depends on the application domain, in order to make a thorough and unbiased evaluation of a 3D shape retrieval method with high confidence (due to the subjectivity of human similarity judgement [164]) different evaluation metrics are needed to measure different aspects of shape retrieval behaviour. To this end, several commonly used metrics (in information retrieval) are adopted to measure the performances of our *3D shape retrieval algorithms* on different datasets. The performances of several other retrieval methods are also compared using these metrics.

In this thesis, at least 8 IR performance evaluation metrics: NN, FT, ST, E, DCG, normalised Discounted Cumulative Gain (nDCG), mAP, and the PRC plots, have been considered to evaluate our retrieval methods. However, while none of these evaluation measures are new, their detailed descriptions (i.e. how each metric computes/evaluates the performance of a retrieval method) are included in this section for the reader to understand the results presented in Section 4. The first three statistics: NN, FT, and ST indicate the percentage of the top K matches that belong to the same class as the query shape. However, in all three cases, an ideal matching result (where all the other shapes within the query's class appear as the top matches) gives a score of 100%.

Nearest Neighbour (NN)

For the NN metric, $K = 1$, and provides an indication to how well a nearest neighbour classifier would perform.

First Tier (FT)

Considering the percentage of shapes in the query's class that appear within the top K matches, where K depends on the size of the query's class, the FT metric indicates the recall for the smallest K that could possibly include 100% of the shapes in the query class [215]. For this metric, $K = C_q - 1$.

Second Tier (ST)

The ST metric produces similar result to FT, but is a little less stringent with K being $2(C_q - 1)$ (i.e. K is twice as big). It is similar to the "Bull's Eye Percentage Score" [215], where $K = 2C_q$.

E-Measure (E)

The idea behind E-measure is that a user of a search engine is more interested in the first page of query results than in later pages. Therefore, this metric only considers the first 32 retrieved items (shapes) for every query for its calculation. It is regarded as a composite measure of precision and recall for a fixed number of retrieved results, and mathematically defined according to Equation (4.6). Essentially, E-measure combines precision and recall into a single value for the performance evaluation of the entire system [98].

$$E = \frac{2}{(\frac{1}{P} + \frac{1}{R})} \tag{4.6}$$

Alternatively, the value of E-measure (see Equation (4.9)) can also be derived by first computing the F-measure, which is the weighted harmonic mean of precision and recall, given by the definition in Equation (4.7)

$$F_\alpha = \frac{(1+\alpha) \times precision \times recall}{\alpha \times precision \times recall} \tag{4.7}$$

where α is the weight. Assuming $\alpha = 1$, the weight of the precision and recall values will be the same, hence:

$$F = 2 \times \frac{precision \times recall}{precision + recall} \tag{4.8}$$

$$E = 1 - F \tag{4.9}$$

The E-measure has a maximum value of 1.0 and the higher its value, the better the matching results (for two 3D shapes).

Discounted Cumulative Gain (DCG)

The DCG provides a clue of how well the overall retrieval results would be viewed by the user. Correctly retrieved shapes near the front of the ranked list are more likely to be seen than correctly retrieved shapes near the end of the list. Mathematically, if the i^{th} shape is in the correct query class, DCG is given as:

$$1 + \sum(\frac{1}{log(i)}) \tag{4.10}$$

This sum is then normalized by the maximum possible value if the first $C-1$ items (shapes) were all in the correct class where $C-1$ is the size of the class without the query item (shape).

It is helpful to review Ranked Lists (RL) and **BRL!** (**BRL!**) (see Section 4.3.2) in order to understand the formulation for the DCG metric. Consider the **BRL!**

results, where item (shape) IDs are replaced with relevance values, the cumulative gain vector, G:

$$G = [1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1...]$$

at ranked position i is derived by summing from first position, i.e. 1 to i when i ranges from 1 to the length, l of the ranked list. Assuming the position i in the gain vector, G is $G[i]$, the cumulative gain vector CG is then recursively defined thus:

$$CG_i = \begin{cases} G_1 & if\ i = 1 \\ CG_{(i-1)} + G_i & otherwise \end{cases}$$

The size of the gain vector, $G = l$ is equal to the number of 3D shapes, N for each respective benchmark dataset we evaluate our retrieval method against. For example, G is a 100, 200, 600, and 2,267-sized vectors for SHREC'17, SHREC'10, SHREC'11, and SHREC'18 datasets, respectively. If $G_i = 1$, then the i^{th} retrieved item (shape) is in the same class of the query item (shape), and otherwise, if $G_i = 0$.

The comparison of matching algorithms is then equal to compare the cumulative gain, and the greater the rank, the smaller the share of the item (shape) value added to the cumulative gain [71]. A discounting function is needed which progressively reduces the item (shape) weight as its rank increases but not too steeply, thus:

$$DCG_i = \begin{cases} G_1 & if\ i = 1 \\ DCG_{(i-1)} + \frac{G_i}{log_2(i)} & otherwise \end{cases}$$

Normalised Discounted Cumulative Gain (nDCG)

According to [121], the value, DCG_i is divided by the maximal value possible (i.e. the value obtained by the ground truth) as follows:

$$DCG = \frac{DCG_N}{1 + \left(\sum_{j=2}^{|C|}\right) \frac{1}{log_2(j)}} \tag{4.13}$$

where N is the total number of items (shapes) in the database and C the classes size. This value is a good summary of the comparative evaluation of the performance of different retrieval methods, and a normalized value, $nDCG$ of the DCG is therefore computed over all methods, and compared to the average value, $avgDCG$, as in Equation (4.14).

$$nDCG = \frac{DCG_N}{avgDCG} - 1 \tag{4.14}$$

where negative value indicates that the performance of the method is under the average, and a positive value indicates that the performance of the retrieval method is above average [121]. In all our performance evaluation results where we applied the $nDCG$ metric, the values returned are positive (see Section 4). This indicates

that those our retrieval methods on the respective benchmark datasets where those methods are experimented performed well, even in cases where our results did not outperform state-of-the-art methods (see results in Section 5.5.5 and Table 5.46, for example).

Precision (P)

Precision metric is defined as the ratio of retrieved items(shapes) which are relevant to a query item (shape), thus revealing the probability (P) of the shape retrieval method to retrieve relevant shapes [49, 46, 154]. The *precision* metric can be seen as the percentage of retrieved items that are correct, and is formulated as:

$$Precision = P(relevant|retrieved) = \frac{\#(relevant\ items\ retrieved)}{(\#relevant\ items)} \tag{4.15}$$

Recall (R)

Recall measures the ratio of retrieved items(shapes) that are relevant, given a query item (shape). It reveals the probability (P) that a relevant shape is retrieved by the query or the capability that only relevant shapes are retrieved [49]. The *recall* metric can be seen as the percentage of correct items that are retrieved, and is formulated as:

$$Recall = P(retrieved|relevant) = \frac{\#(relevant\ items\ retrieved)}{(\#relevant\ items)} \tag{4.16}$$

Precision-Recall (PR) Curve

Whereas precision and recall are two fundamental measures often used to evaluate the performance of information retrieval method (shape matching algorithm), *Precision-Recall Curve* is the combination of the two metrics (Precision and Recall) into a single plot that describes the relationship between precision and recall in a ranked list of matches.

In principle, precision is the percentage of retrieved items(shapes) that are relevant, while recall is the percentage of relevant items(shapes) that are retrieved. These *PR* concepts and notations are clarified in Figure 4.10 and the PR contingency table, Table 4.4 that follows.

	Relevant	**Not Relevant**
Retrieved	true positive (**tp**)	false positive (**fp**)
Not Retrieved	false negative (**fn**)	true negative (**tn**)

Table 4.4: Precision-Recall contingency table.

Key PR Metrics:

$$Precision = \frac{tp}{tp + fp} = \frac{tp}{Retrieved} \tag{4.17}$$

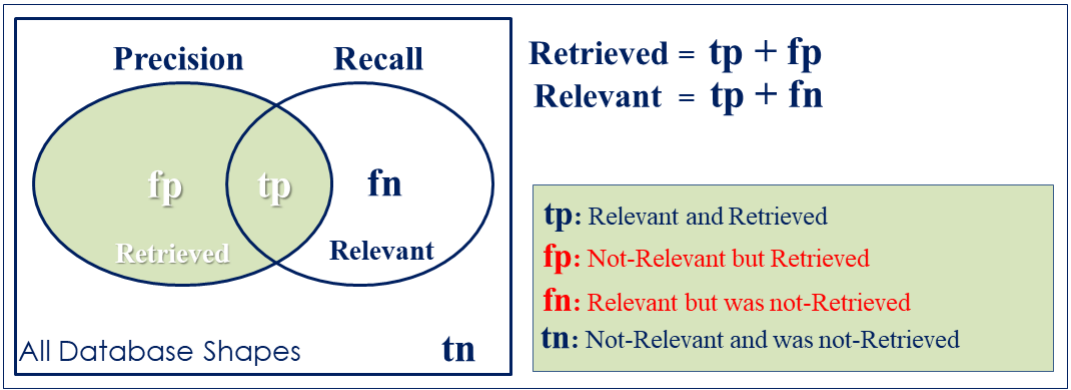


Figure 4.10: Precision-Recall concepts and notations.

$$Recall = \frac{tp}{tp + fn} = \frac{tp}{Relevant} \tag{4.18}$$

$$Accuracy = \frac{(tp + tn)}{(tp + tn + fp + fn)} = \frac{Correct}{All\ Database\ Shapes} \tag{4.19}$$

$$Error = 1 - Accuracy \tag{4.20}$$

The precision-recall curve (see Figure 5.13 for example) is used in IR to show the tradeoff between precision and recall for different values. A high value for both precision and recall shows that the retrieval method returned accurate (i.e. high precision) and positive (i.e. high recall) results. Likewise, a high area under the PR curve represents high precision, as well as high recall, and high precision represents a low fp rate, while high recall represents a low fn rate.

According to [154], an obvious alternative that may occur to the reader is to judge an information retrieval system by its accuracy, that is, the fraction of its classifications that are correct. In terms of the contingency table (see Table 4.4), shape retrieval accuracy can be formulated as shown in Equation (4.19), while the error of the retrieval method can be derived as given in Equation (4.20). However, it's a bad idea to depend on accuracy measure, and possibly the error measure, in an IR system because accuracy is, more commonly, a description of systematic errors (i.e. a measure of statistical bias), thus a measurement system can be accurate but not precise, precise but not accurate, neither, or both. For example, if an experiment contains a systematic error, then increasing the sample size generally increases precision but does not improve accuracy [126].

AP or mAP

As the name suggests, average precision is a scalar metric that averages all precision values where a relevant item (shape) have been retrieved, and represents the average precision performance of a shape retrieval method over all relevant items(shapes) up to a ranking position [90]. According to [129], AP measure is a single-value that evaluates the performance over all relevant items(shapes). It is not an average of the precision at standard recall levels, but the average of precision values at each relevant item (shape) retrieved. For example, considering a query with five

relevant items which are retrieved at ranks $1, 2, 4, 7$ and 10, the actual precision value obtained when each relevant item is retrieved would be $1, 1, 0, 0.75, 0.57$ and 0.50, respectively. Therefore, the average precision would be 0.76 [71]. Finally, AP summarises the PR curve or plot as the weighted mean of precisions achieved at each value, with the increase in recall from the previous value used as the weight:

$$AP = \sum_{n} (R_n - R_{n-1}) P_n \tag{4.21}$$

where P_n and R_n are the precision and recall at the nth threshold. A pair (R_k, P_k) is referred to as an *operating point* [126]. The mean average precision (mAP) presents an overall performance evaluation of the shape retrieval method, by finding the average precision for each query item (shape) and computing the mean average precision over all query items(shape). It is also considered to be robust in quantifying the overall performance of a retrieval method [154].

Chapter 5

EXPERIMENTAL EVALUATION RESULTS AND DISCUSSIONS

5.1 Introduction

This chapter presents the result of our experimental evaluations. First, we present and describe up to ten different benchmark datasets used in this thesis to evaluate the performances of our proposed methods, and highlight some critical issues regarding these datasets, including the retrieval challenges that each dataset present to 3D shape retrieval algorithms (i.e. descriptors). Although we provide some results of experimental evaluation for the performance of our proposed HoGD *"global"* method, using the SHREC 2021 [188] 3D protein benchmark dataset (not described in this thesis), detailed description of all other 3D benchmark datasets upon which the retrieval accuracies and robustness of our proposed methods have been tested are provided in Section 5.2.

For all the different 3D shape retrieval methods proposed in this thesis (see Section 4.2), their retrieval accuracies (i.e. performances) on several different standardised 3D benchmark datasets (mentioned above) are presented and discussed, following thorough experimental evaluations. Essentially, for each of these retrieval methods (our proposed methods and state-of-the-art methods), we present both qualitative and quantitative performance evaluations, reported against *some* specific 3D SHREC benchmark datasets to which the method has been applied. In addition, we compare and analyse the overall performances (results) for each of our proposed methods on a given dataset with the performances of several other state-of-the-art retrieval methods for that particular dataset, using at least six standard and commonly used IR performance evaluation metrics described in Section 4.3.3, which are: NN, FT, ST, E, DCG, and PRC, including nDCG, mAP, and AUC.

It is important to mention here, that as part of our experimental evaluation strategy, rather than testing the robustness of our shape retrieval methods (descriptors) using a single dataset that have different variations such as noise/distortions, holes, occlusion, deformations, level-of-details, etc., we instead choose to evaluate the retrieval performances of our methods and descriptors, using several different standardized 3D shape retrieval benchmark datasets described in Section 5.2. We consider this strategy to provide a more trusted and unbiased approach to testing

the effectiveness of retrieval methods, descriptors robustness, and scalability, because each of these datasets presents varieties of complicated and unique retrieval challenges to any retrieval algorithm applied to it. Unlike earlier work such as [174, 64] and several others, whose strategy was to use a single 3D database where each 3D model has different variations of noise, cracks/holes, tessellation, LoDs, etc., our experimental evaluation strategy to robustness testing (i.e. retrieval results) reveals how our solutions (retrieval methods) perform on a wider range of 3D datasets and against a variety of retrieval challenges that each of these datasets present.

In addition, rather than use synthetic dataset containing fewer number of 3D objects for our robustness testing and performance evaluation, the choice of SHREC'12 (having 1,200 3D shapes), SHREC'18 (having 2,267 3D protein conformers) and particularly SHREC'14 (having 8,987 3D shapes), each withan exceptionally large number of diverse and complex 3D models, for example, is basically for us to determine the suitability of our proposed solution in real-world applications. As previously stated in Section 4.2.3, for most of the shape retrieval methods we present in this thesis, our algorithms are implemented in Python 3.6.0 and all experiments are carried out on either of our: **(i)** HP Pavilion, Windows 10 Home, x64-based PC, with the following configurations. Processor: Intel(R) Core(TM)i3-5157U CPU @ 2.50GHz 2.49GHz. Installed memory (RAM): 8.00GB, and **(ii)** Windows 7 desktop PC with Intel Core i7-4790 CPU @ 3.60GHz, 32GB RAM, depending on the size of the dataset.

Note that for each experimental run reported in this chapter, we first apply each retrieval method described in Section 4.2 (with a given parameter setting for that method) on a single 3D benchmark dataset, and compute 3D shape descriptors (local, global, or hybrid, depending on the retrieval method used) for all the 3D objects in that dataset. Next, we compute a single distance/(dis)similarity matrix (DM) per method/experimental run on the whole dataset. The DM represents the (dis)similarity of all pairs of 3D objects in the given dataset. For N objects, the matrix is a sequence of N by N floating point numbers, where the numerical value at position $i*N+j$ represents the (dis)similarity between objects i and j. A value of 0.00 indicates that the descriptors for objects i and j are identical, and larger values indicate greater (dis)similarity between the two 3D objects. All of our (dis)similarity measures have positive values. We refer the reader to Section 4.3.2 for additional details regarding this. Finally, this DM and the GT data/information are plugged into an evaluation software tool described in Section 4.3.1 and Section 4.3.3 to produce quantitative evaluation results, including the PRC plots.

The idea or reason behind measuring (dis)similarity among the objects in the database to produce N by N matrix (DM) instead of just between the objects in the database and the query objects is this: Computing a DM is needed by the evaluation algorithm. This computation considers each item (3D object descriptor) in the database as a query object, Q_i which is then matches to the remaining items in the dataset. For example, assuming there are 10 items $(i_0, i_1, i_2, i_3 \cdots i_9)$ in the database, $Q_1 = i_0$: i_0 is matched with itself and the other nine objects, which produces 10 (dis)similarity scores for line 1, using Q_1. Similarly, $Q_2 = i_1$: i_1 is matched with i_0, itself, and the remaining eight objects, which produces 10 (dis)similarity scores

for line 2, using Q_2. Also, $Q_1 0 = i_9$: i_9 is matched with the remaining nine objects and itself, which produces 10 (dis)similarity scores for line 3, using $Q_1 0$, etc. The overall results of this is a 10 by 10 DM of (dis)similarity scores for the dataset, which is needed by the evaluation code, designed to accept such format of measurements output to produce the results we get. Finally, however, for production purposes, the matching technique or algorithm does not need to output a DM. Direct querying of indexed database descriptors are rather used to match with newly extracted query object descriptors and the final results are ranked/sorted according to similarity values. The evaluation algorithm is designed to compute or derive RLs from DM - see Section 4.3.2.

5.2 Datasets

This section introduces a variety of 3D shape benchmark datasets which have been used for testing our shape retrieval methods. Most of these datasets are made of 3D triangular meshes (i.e. contains vertices and connectivity information) while others contain unstructured data types, such as point cloud objects (refer to Section 2.2.1 for more details on 3D object representations). The datasets we have adopted for this thesis covers a very wide range and classes (categories) of rigid and non-rigid 3D models such as human, vehicles (cars, semi-trucks, etc.), animals, furniture, artefacts, buildings, mechanical parts, toys, biological structures, and insects, etc. A summary of essential information regarding each of the datasets described in this section is provided in Table 5.1. However, the choice of all these datasets presented in this thesis for testing/evaluating the performances of our retrieval methods is largely due to their diverse acquisition technique, diverse retrieval challenges, diverse surface representation, shape quality and possible application scenarios.

In Table 5.1, the "Database Size" column reveals the total number 3D objects that each benchmark dataset contains, while "Shapes/Class" lists the number of 3D objects contained in each class. Also as shown in Table 5.1, some benchmarks (i.e. SHREC'10, SHREC'11, SHREC'12, SHREC'17, SHREC'20, and ShapeGoogle) have fixed number of objects per class, while others, such as the SHREC'14, SHREC'18 *protein*, SHREC'19 *protein*, and SHREC'20 *protein*, have variable number of objects per class. Finally, it is worth mentioning that in addition to the 3D objects data, each of these shape retrieval benchmarks (databases) also provide a CI or GT data (i.e. *.cla* file) for performance evaluation of shape retrieval methods and/or algorithms, except the ShapeGoogle dataset, which was only used for raw experimental testing of our techniques and never used to evaluate our methods against any state-of-the-art ones. Some benchmarks also include evaluation code which allows the retrieval performances of several different retrieval methods (including the methods we propose) to be measured with the exact same criteria, for quantitative, as well as qualitative evaluations and comparisons.

In summary, several 3D retrieval benchmarks exist, such as the ones in [139, 245, 68, 271, 131, 132, 142, 139, 140] and [119] among others. Each of these datasets presents a unique challenge to shape retrieval algorithms. Some benchmarks contains only 2.5D range data [240], or a combination of 3D mesh and scenes objects for recognition, such as the datasets in [106, 150] and [193]. Other benchmarks provide point

S/N	Database	Shape Representation	Database Size	Categories (Classes)	Shapes /Class	File Format	Year	Data Provider	Data Source(s)
1	SHREC'10 [140]	Polygon meshes	200	10	20	ASCII .off	2010	McGill University	[237]
2	SHREC'11 [139]	Polygon meshes	600	30	20	ASCII .off	2011	NIST,Gaithersburg, USA	[139, 216, 23, 139]
3	SHREC'12 [131]	Polygon meshes	1,200	60	20	ASCII .off	2012	NIST,Gaithersburg, USA	various [131]
4	SHREC'14 [132]	Polygon meshes	8,987	171	varies	ASCII .off	2014	NIST (Generic3D@nist.gov)	various [132]
5	SHREC'17 [142]	Point Cloud	100	10	10	ASCII .off	2017	University of York	[269, 142]
6	SHREC'18 *protein* [121]	Polygon meshes	2,267	107	varies	ASCII .off, .pdb	2018	L.Florent, M. Matthieu [121]	[60]
7	SHREC'19 *protein* [119]	Polygon meshes	5,298	Table 5.2	Table 5.2	ASCII .off, .pdb	2019	L.Florent, M. Matthieu [119]	[61]
8	SHREC'20 *protein* [122]	Polygon meshes	588	Table 5.3	Table 5.3	ASCII .off, .pdb	2020	L.Florent, M. Matthieu [122]	[157]
9	SHREC'20 *reliefs* [163]	Polygon meshes	220	11	20	ASCII .off	2020	IMATI [40]	[231]
10	SHREC'21 *protein* [188]	Polygon meshes	1,543	144	varies	ASCII .off	2021	IMATI [40]	[187]
11	ShapeGoogle	Polygon meshes	200	20	10	ASCII .off	n/a	Duke University	[38]

Table 5.1: Outline of 3D shape benchmark datasets used in this thesis.

cloud data containing only the geometry of points, i.e. $[x, y, z]$ coordinates, for example, the PRoNTo (Point Cloud Retrieval of Non-rigid Toys) dataset [142]. However, many other benchmarks, like the Princeton Shape Benchmark (PSB) [245], McGill dataset [57], SHREC'10 [140], SHREC'11 [139], SHREC'14 [132], SHREC'16 [68], SHREC'19 [119, 56], etc. provides their data as triangular meshes.

5.2.1 SHape REtrieval Contests (SHREC) 3D Benchmarks

In 2016, the Network of Excellence AIM@SHAPE [247] began to organize SHREC [244], which at the time only used 3D polygonal models from the Princeton Shape Benchmark (PSB) [191]. In 2017, AIM@SHAPE expanded the SHREC retrieval to multiple tracks, including watertight 3D shapes, protein shapes and 3D face models, etc. Since SHREC 2009 [72] till presently, the SHREC competition has been organized in collaboration with Eurographics Workshop on 3D Object Retrieval, 3D Object Retrieval (3DOR) [56, 55]. 3DOR is the dedicated workshop series for methods, applications and benchmark-based evaluation of 3D object retrieval, classification, and similarity-based object processing [55].

Since 2006, SHREC has received huge attention within and outside the research community, including the current year, 2020, where it provides multiple tracks of retrieval and classification challenges. However, the ultimate goal of SHREC is to provide a platform for researchers working in 3D object retrieval field to present and evaluate the effectiveness of state-of-the-art 3D-shape retrieval methods, using a common performance evaluation criteria and tools - see Section 4.3. Following the efforts of previous track organizers, SHREC provides many resources to compare and evaluate 3D retrieval methods.

Choice of 3D Shape Retrieval Benchmarks

In conjunction with the above goal of SHREC, the robustness (i.e. performances) of our proposed 3D shape retrieval methods or descriptors in this thesis are tested against several different SHREC 3D datasets listed in Table 5.1. We described each of these datasets in the sub-sections that follow. It is important to note that each of these datasets (databases) has its own uniqueness and presents different challenges to shape retrieval algorithms. For example, the shape retrieval method for the SHREC'17 [142] dataset would have to worry about common scanning problems, mainly caused by object's self-occlusions, for non-rigid point cloud data. This dataset is characterized by missing surface parts (see Figure 5.5) and the primary objective of the track was to evaluate shape retrieval methods that can be computed directly and efficiently from point clouds. A brief description of the SHREC'17 dataset, and the motivation for using it, is presented in Section 5.2.6.

All evaluated datasets in this thesis contain greater number of 3D shapes than in databases used for recognition [204, 96, 58, 150], registration [73, 198, 199, 227], and segmentation [199] research. Experimental results (see Section 5.1) show that our proposed retrieval methods perform very well on these variety of data sets, each of which presents different challenges associated with the 3D data. Therefore, we expect that our feature extraction method would produce better performance outcome

on other computer vision tasks such as 3D detection, recognition, segmentation, and registration etc.

5.2.2 SHREC 2010 Non-rigid 3D Dataset

SHREC'10 - Shape Retrieval Contest of Non-rigid 3D Models: Basically, this is a McGill Articulated Shape Benchmark database [237] that consists of 255 non-rigid, watertight (manifold) 3D models represented as watertight triangle meshes. They are classified into 10 categories. The maximum number of the objects in a class is 31, while the minimum number is 20. 200 models are selected (or modified) to generate the test database for SHREC 2010 track [140] in order to ensure that every class contains equal number of models. Note that similar to all other SHREC retrieval tracks, the SHREC'10 track provides source code and ground truth data for evaluating quantitative retrieval statistics of shape retrieval algorithms, which then enables results from new methods to be fairly comparable to the original benchmark result.

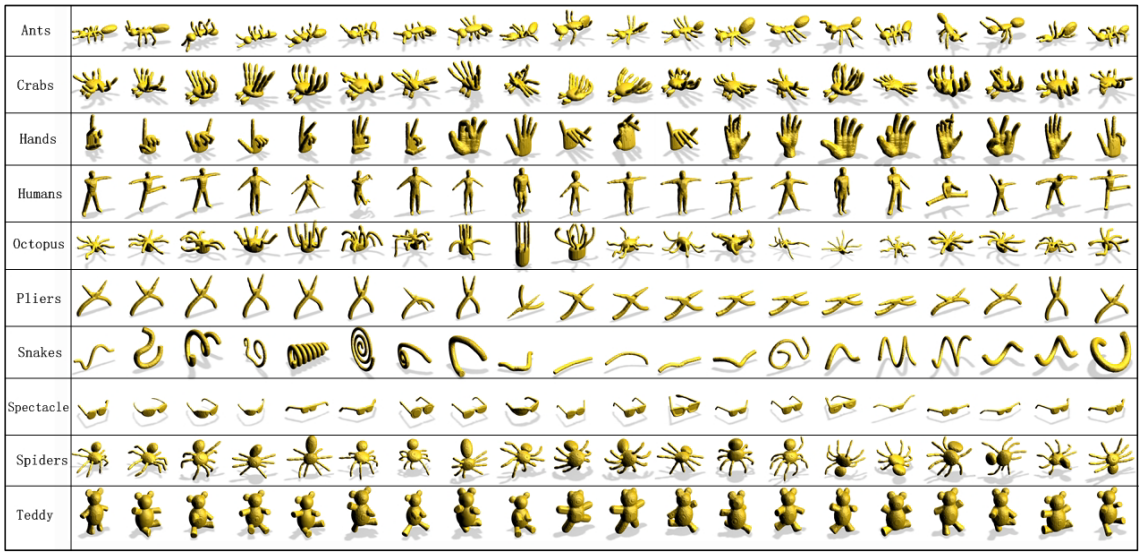


Figure 5.1: Sample of 3D shapes in SHREC'10 database, grouped into 10 classes and each class contains 20 shapes [140].

Motivation for Using SHREC'10 Dataset/Retrieval Challenge: Previous efforts have mainly been devoted to the retrieval of rigid 3D models, and thus comparing non-rigid 3D shapes is still a challenging problem in content-based 3D object retrieval. Therefore, this track (SHREC'10) is devoted to promote the development of non-rigid 3D shape retrieval [140].

5.2.3 SHREC 2011 Non-rigid 3D Watertight Dataset

SHREC'11 - Shape Retrieval Contest of Non-rigid 3D Watertight Meshes: To augment the number of 3D models in the SHREC'10 track, SHREC'11 [139, 139] employed a large-scale database consisting of 600 non-rigid 3D objects that are created by their group using some modelling software and codes. They classified the models properly to make sure that every class holds equal number of models. The models

are represented as watertight triangle meshes. Some of the models were recreated with permission and are originally from different publicly available databases: such as McGill database, TOSCA shapes, PSB, etc.

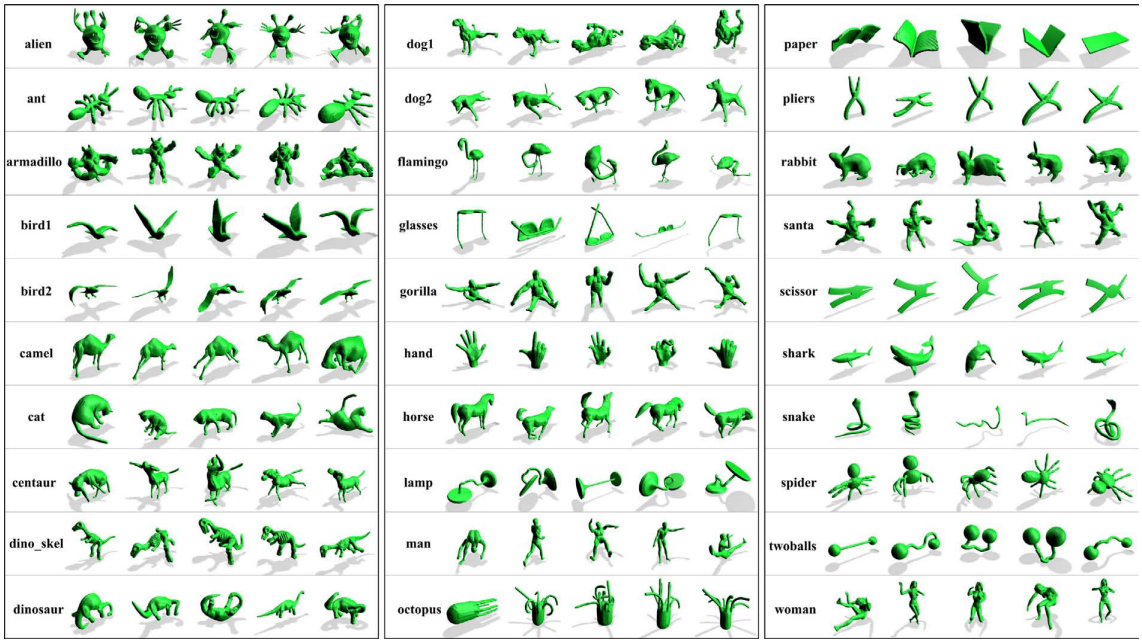


Figure 5.2: Sample of 3D shapes in SHREC 2011 database, classified into 30 categories [139].

Motivation for Using SHREC 2011 Dataset/Retrieval Challenge: Similar to, and in addition to that of SHREC'10, the motivation for the SHREC'11 track was to evaluate the performance of 3D shape retrieval approaches on a large-scale database of non-rigid 3D watertight meshes generated by the group [139, 139]. As shown in Figure 5.2, this dataset contains a set of models which have similar overall appearances but belong to various categories because they are different in the details of local regions or/and topological structures. This makes the new benchmark more challenging than other non-rigid 3D databases [139].

5.2.4 SHREC 2012 Generic 3D Shape Dataset

SHREC'12 Track - Generic 3D Shape Retrieval: Generic 3D shape retrieval is a fundamental research area in the field of content-based 3D shape retrieval [131]. The SHREC'12 track contains 1,200 triangular meshes that are equally classified into 60 categories, with each category having 20 different shapes. They are comprised of rigid models, non-rigid models, including mechanical parts as shown in Figure 5.3. For more details about this dataset, we refer the reader to [131].

Motivation for Using SHREC 2012 Dataset/Retrieval Challenge: Our motivation for using this dataset is that generic models have more variations compared to professional models, for example, a chair can have diverse shapes and it may have wheels or not; a table can be round or rectangular. These variations make it more difficult for a retrieval method to be able to identify and retrieve models in

a particular class, thus posing some level of challenge. Additionally, generic models include the 3D objects that we see often in everyday life, hence the need to develop a retrieval solution for such models.

Figure 5.3: Example 3D models in Generic 3D Dataset. [131].

5.2.5 SHREC 2014 Large Scale Comprehensive 3D Shape Dataset

The *SHREC'14 Track: Large Scale Comprehensive 3D Shape Retrieval:*, presents 8,987 triangle meshes that are classified into 171 categories, is made up of a collection of relevant models in major previously proposed 3D objects retrieval benchmarks such as the Engineering Shape Benchmark (ESB) [92], McGill 3D Shape Benchmark (MSB) [217], Watertight Model Benchmark (WMB) [70], Bonn Architecture Benchmark (BAB) [253], and others. This dataset contains different types of models, such as Computer-Aided-Design/Manufacturing (CAD/CAM), generic, architecture, and articulated models. Figure 5.4 show example models in ESB, MSB, WMB, and BAB datasets.

Motivation for Using SHREC 2014 Dataset/Retrieval Challenge: We are motivated to test the robustness of our algorithm and retrieval method on this dataset because it is the most exhaustive dataset in terms of the number of semantic query categories covered, as well as the variations of model types for hand-crafted features. It also combines generic and domain-dependent model types and therefore rates the retrieval performance with respect to cross-domain retrieval tasks [130]. Refer to [130], for more information regarding this dataset.

5.2.6 SHREC 2017 Non-rigid Point Cloud (PRoNTo) Dataset

SHREC 2017 - Point-Cloud Shape Retrieval of Non- Rigid Toys: According to [142], the dataset in SHREC 2017 contains 100 different models that are derived from 10 different real objects. Each object has been scanned in 10 distinct poses by articulating them around their joints. Objects were scanned using the Head and Face Color 3D Scanner (http://cyberware.com/products/scanners/ps.html). The point clouds acquired by these scans suffer from missing parts resulted from self-occlusions of the objects. The scanned point clouds were randomly rotated across XYZ axis and they contain in average 4K uncoloured points each. The file format was chosen as the ASCII Object File Format (**.off*) [185], where they are only vertex information. The 3D points-cloud with missing parts are intended to test signatures robustness against scanning problems since real-time interaction with 3D

Figure 5.4: Example 3D models in ESB, MSB, WMB and BAB datasets, combined to produce SHREC 2014 dataset. [132].

scanners/objects are expected to be part of our everyday lives in the near future as suggested by the growth of Virtual and Augmented Reality and the new 3D platform from Microsoft [177], which will include many 3D features in its products [142].

Motivation for Using SHREC 2017 Dataset/Retrieval Challenge: The tasks for this track present a 3D shape retrieval problem where a shape retrieval method has to worry about common scanning problems, mainly caused by object self-occlusions, however, it cannot be classified as a part-based shape retrieval task because the gross structure of the shape is always presented (only fine details are missing). According to [142], scanning problems occur as a result of occlusion, "*when the laser hits first one part of the model leaving another part of the model unseen (in the same direction pointed by the scan head). These objects with missing parts are intended to test signatures robustness against scanning problems since real-time interaction with 3D scanners/objects are expected to be part of our everyday lives in the near future as suggested by the growth of Virtual and Augmented Reality and the new 3D platform from Microsoft, which will include many 3D features in its products [269]*".

In addition, we have been interested in evaluating the robustness of our descriptors and/or shape retrieval methods using a different type of surface representation from the popular 3D mesh, hence our primary motivation for using the PRoNTo dataset. Additionally, we expect that our proposed solution would be robust against scanning problems for real-time interaction with 3D scanners/objects that are part of everyday life.

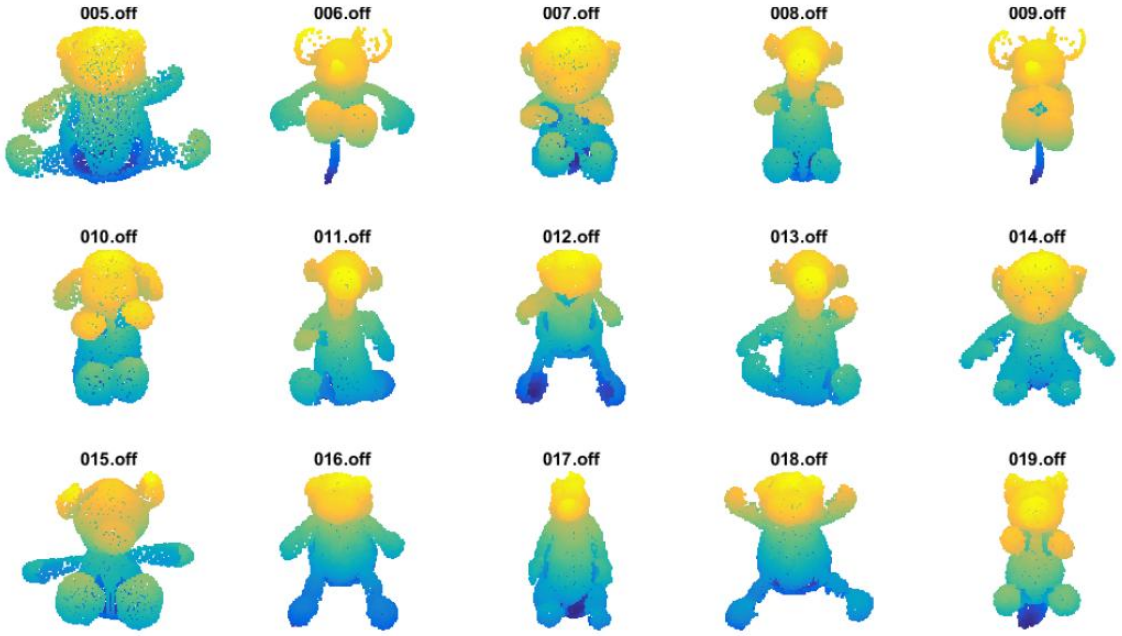


Figure 5.5: Samples of 15 Non-rigid point cloud 3D scans from SHREC'17 Track: Point Cloud Shape Retrieval of Non-Rigid Toys. "Point clouds coloured by coordinates $Y * Z$". [142].

5.2.7 SHREC 2018 Protein Shape Dataset

SHREC 2018 - Protein Shape Retrieval: This dataset contains $2,267$ unique protein structures/shapes called conformers. These shapes were distributed into 107 categories with an average class size of 21.18. Out of the 107 classes, 18 classes had only one conformer each, and the largest class has a size of 110 conformers. In Figure 5.6 we visualize 8 sample shapes from this dataset, taken from 8 different classes. For additional details regarding this dataset, the reader is referred to [121]. Typically, a single 3D shape in this dataset is the Solvent Excluded Surface (SES) of a protein molecule, which has been created from the molecule's tertiary structure (PDB format) using the EDTSurf software that produces a high-resolution watertight mesh. The mesh, which is triangulated, is simplified and used as input to our shape retrieval algorithms - see Section 4.2.3.

Motivation for Using SHREC 2018 Protein Dataset/Retrieval Challenge: Proteins are macromolecules that display dynamic and complex surface structure (composed of hundreds of thousands of atoms), and central to biological processes [121]. They display multiple conformations differing by local (residue side-chain) or global (loop or domain) structural changes at the atomic scale which can drastically impact their global and local shape. Because proteins display multiple possible conformations in solution, the detection of shape similarity and/or identity is of biological relevance in drug discovery processes and molecular characterization of diseases. Also, protein shape retrieval [121, 119] has recently become an import research focus, hence the motivation for testing our retrieval methods on this dataset.

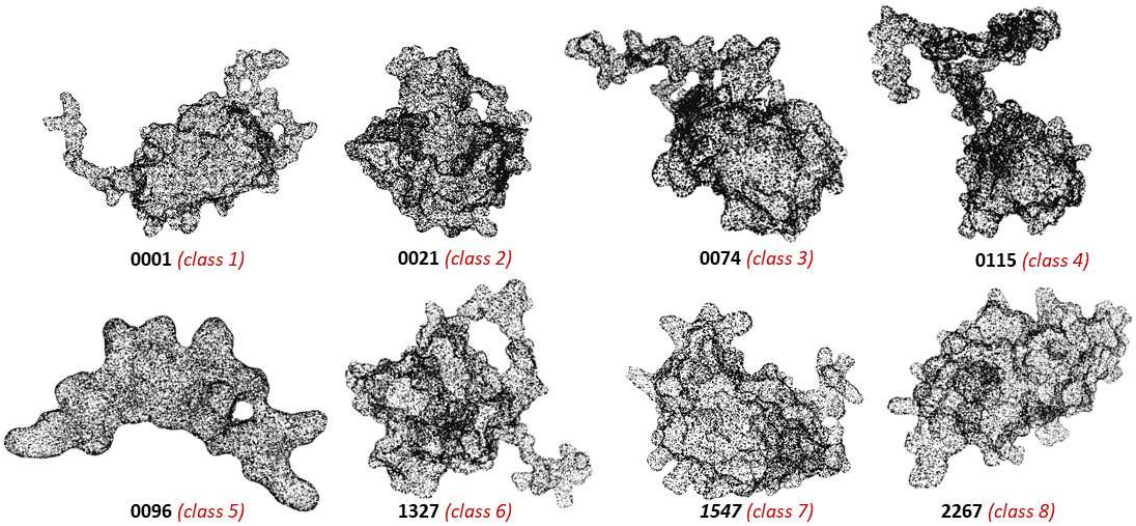


Figure 5.6: Point cloud representation of sample shapes (i.e. conformers) in the SHREC'18 protein shapes retrieval track, showing one conformer each from 8 out of 107 different classes. Conformers 0001.off, 0021.off, 0074.off, 0115.off, 0096.off, 1327.off, 1547.off, and 2267.off are respectively from classes P16INK4A, CRABPII, EIF1A, EIF1A-HisTag, HisTag, Bax, IFNA2A, and UIMC1-humanUbiquitin.

5.2.8 SHREC 2019 Protein Shape Dataset

SHREC'19 - protein shape retrieval challenge: Given that proteins are dynamic, non-rigid objects and that evolution tends to conserve patterns related to their activity and function, the SHREC'19 protein retrieval track aimed at retrieving protein evolutionary classification based only on their surfaces meshes, and offers a challenging issue using biologically relevant molecules [119]. In addition, we decide to also test the retrieval performances of our shape retrieval methods on a much larger Protein shape dataset: SHREC'19, considering the successes recorded with our HAPPS-1 method on the SHREC'18 Protein dataset as reported in [175].

Dataset Creation and Ground Truth (GT) Generation: A total of 5,298 protein structures representing 241 SCOPe [33, 32, 62] database entries extracted from 211 PDB entries, were randomly selected from SCOPe entries containing: i) NMR structures (whose conformers display the same number of atoms), ii) Three classes of proteins (i.e. α, $\alpha + \beta$, and $\frac{\alpha}{\beta}$), and iii) Entries with at least four ortholog proteins [119]. More details on the selection criteria and final 3D protein shapes for this dataset can be obtained from [119]. At the *species* classification level, this data set is composed of 54 classes while at the *proteins* classification level, it is composed of 17 classes. A summary of these classifications and class sizes are presented in Table 5.2, while Figure 5.7 presents a visual representation of some of the shapes in this database.

Motivation for Using SHREC 2019 Protein Dataset/Retrieval Challenge: The retrieval challenge for the SHREC'19 protein retrieval track was to retrieve various conformations of identical proteins and various conformations of ortholog proteins (proteins from different organisms and showing the same activ-

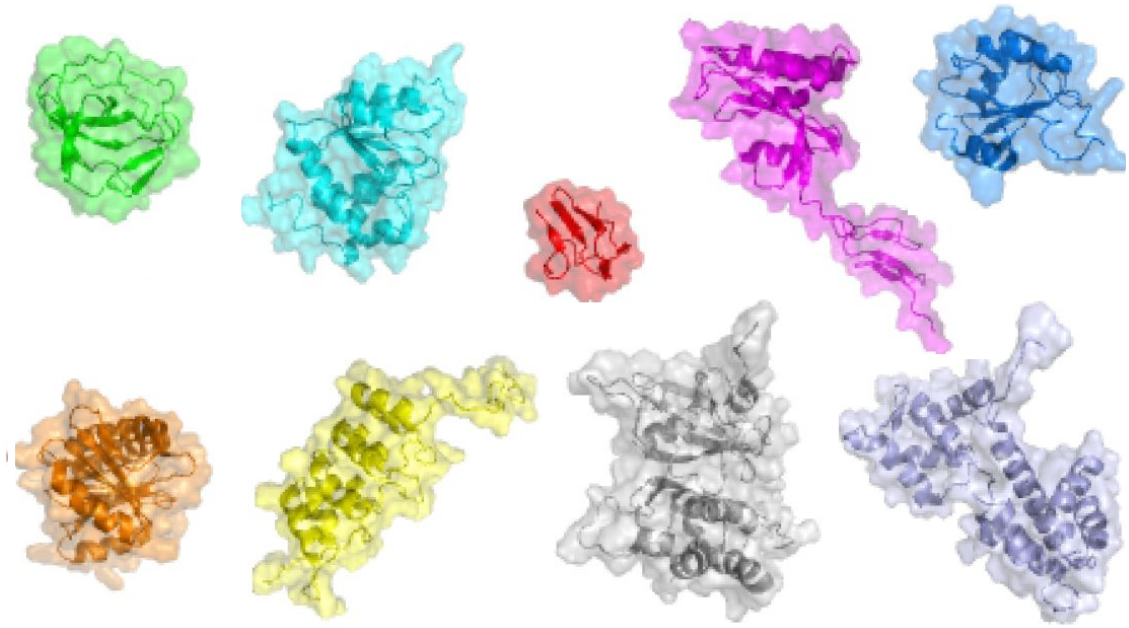


Figure 5.7: Selected 3D shape samples from the SHREC'19 protein retrieval track. Visualisation by: [61, 60]

ity). Compared to the SHREC'18 protein shape retrieval track [121], the 2019 track was focused on the evolutionary relationships between proteins shapes. According to [119]: "*As a consequence of the selection process of surfaces to be included in the dataset, the selected proteins are similar in terms of size (from 32 to 161 amino-acids) with most of the structures being 66 to 111-amino-acids long. The corresponding meshes are also of equivalent size, ranging from 54,000 to ˜270,000 points. Furthermore, because of the evolutionary relationships between them, the shapes from the same class at the proteins level (reflecting the variety of shapes from ortholog proteins) are expected to have a high level of similarity in their overall shape as they share the same activities and functions in organisms that co-evolved from the same ancestor. For these reasons, the discrimination between shapes at the species level is expected to be more difficult compared to the proteins level*".

Level	Average class size	Largest class size	Number of classes
Protein	311.65	1160	17
Species	98.11	1049	54

Table 5.2: Summary of 3D shapes and their grouping at the species and proteins levels in the SHREC'19 protein benchmark dataset [119].

5.2.9 SHREC 2020 Protein Shape Dataset

SHREC 2020 - Multi-domain protein shape retrieval challenge: The dataset includes 588 proteins consisting of two domains (i.e. functional units of the proteins), where only the corresponding triangulated meshes of their solvent-excluded surfaces

(SES) [41] were provided as input to shape retrieval algorithms to retrieve the evolutionary relationships between orthologous proteins (proteins that have the same function in different organisms), and to retrieve the different conformations of an individual protein. Although the dataset is limited in size, where only 588 proteins were considered for this retrieval challenge, out of more than 160,000 protein structures that are experimentally solved. We think that this work provides useful insights into the current shape comparison methods performance and highlights possible limitations to large-scale applications due to the computational cost.

Dataset creation: The SCOPe database [62, 33, 32] organizes the protein domains according to their structural (2 top levels of the SCOPe tree) and evolutionary (four bottom levels) relationships. Protein domains in the SCOPe database originate from Protein Data Bank (PDB) experimental structures [15] and are characterized by their *PDBId* and *chainId*, allowing for filtering based on these parameters. From all entries implemented in the SCOPe tree (excluding entries from the *"Artifacts"* and *"Low resolution protein structures"* classes), we kept only the entries from X-ray crystallography PDB structures which composed of two domains. When multiple copies of the same protein chain were present in the same PDB structure, the track organisers only kept one of those copies to limit redundancies. Finally, all proteins were required to have at least one orthologous protein, and classes with less than 10 members were discarded.

Motivation for Using SHREC 2020 Protein Dataset/Retrieval Challenge: Proteins are linear polymers (protein chains) made of several hundred amino-acid residues, which fold into a specific, well-defined 3D structure. Many proteins need to form a complex of several chains to become functional. For instance, the human haemoglobin requires two α-globin and two β-globin chains to be fully functional. Domains define the functional units of the proteins and are usually associated with a specific function and/or interaction; it is thus commonplace for two proteins to share one domain while their other respective domains differ. This characteristic led to the development of databases classifying proteins according to both their structure and the functions of their domains [122].

The SHREC2020 track on multi-domain protein shapes dataset [122] is devoted to the analysis of protein shapes generated from protein chains that comprise two domains and evaluates the current ability of shape comparison (shape matching/retrieval) methods proposed by six separate groups to tackle the protein surface comparison problem. Compared to other known protein shapes datasets, this dataset is exclusively composed by two-domains proteins while only one- domain proteins were included in [121, 120]. As multi-domain proteins are commonplace at the cellular level, the impact of additional domains on the protein shape retrieval performances need to be evaluated. Recently, another dataset of protein surface patches was published [65], encompassing both geometric and chemical features of proteins surfaces. That dataset gathers partially overlapping patches rather than complete proteins surfaces and is currently limited to structures that display specific functionalities, namely the ability to bind selected small molecules or to form a protein-protein complex [122].

Level	Shapes by class (min / max)	Number of classes
Protein	19 / 168	7
Species	12 / 63	26

Table 5.3: Number of shapes and number of classes in the SHREC'20 protein benchmark dataset, at the protein and the species levels. Source: [122].

Ground Truth (GT) Generation: The ground truth was generated using the resulting SCOPe tree of two-domains proteins. Only the biggest domain (highest number of amino-acid residues) was used to define two ground truth classifications, namely the *protein* and *species* levels, which reproduce the SCOPe tree classifications at the protein and species levels, respectively. By using this protocol, 588 protein chains were retrieved, from 26 orthologs (proteins having the same activity in different organisms such as the human and murine haemoglobin proteins) and 7 proteins (see Table 5.3). The solvent-excluded surfaces [41] were computed for all the entries using EDTSurf [266] (non-protein atoms were discarded) after protonation of the structure using propka [224, 172], and only the corresponding *.off* files were provided on the track web- site (http://shrec2020.drugdesign.fr).

5.2.10 SHREC 2020 3D Geometric Relief Dataset

SHREC 2020 - Retrieval of Digital Surfaces with Similar Geometric Reliefs: The dataset proposed for this challenge consists of 220 triangulated surfaces. Each one of them is characterized by one of 11 different geometric reliefs. 20 base models already used in [162] were selected to create the models. These models represent pots, goblets, and mugs. The surfaces of these models are properly oriented, and they are made of a single connected component. The topology of some models is non-trivial (they may contain handles or tunnels) and may present a boundary, depending on the object represented. Then, a set of 11 textures is selected from the free dataset of textures available online from the site Texture Haven [236] that contains a set of natural, high quality texture images made from scanned maps. Most of these textures represent real bricks, floors, roofs surfaces and rock or wood materials.

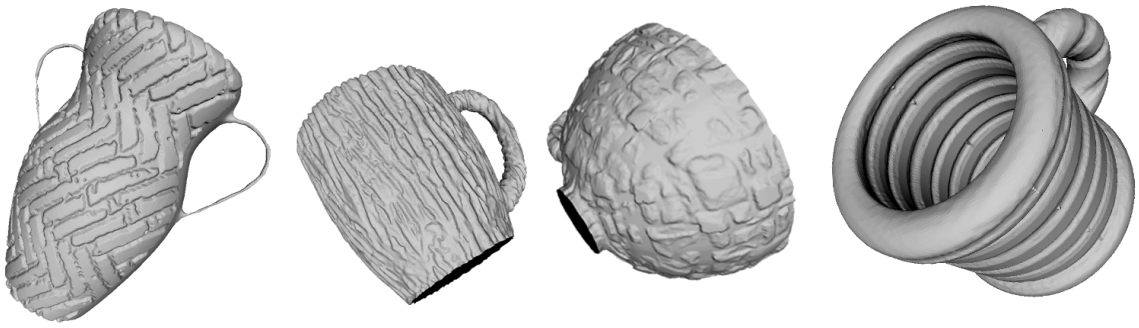
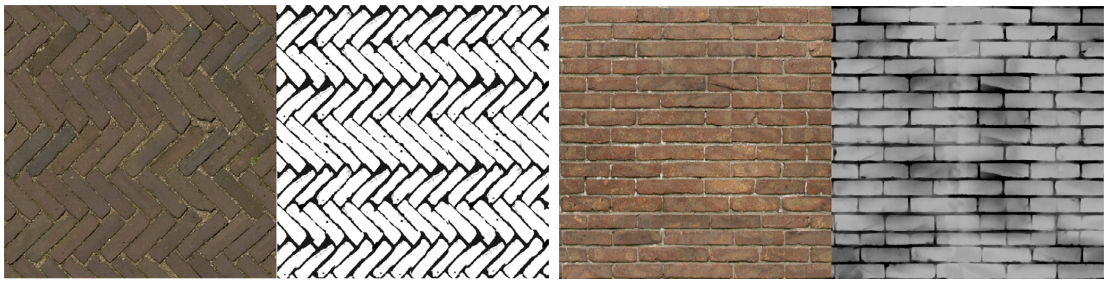


Figure 5.8: Selected Samples of 3D Shapes in the SHREC'20 Retrieval of Surfaces with Similar Geometric Relief Track. Image source: [163].

The organisers of this retrieval challenge transformed each texture into height values suitable to create a geometric relief by converting each texture into a gray-

scale image. The brightness and the contrast values of each image were tuned for each image, based on the values that better enhance the details of the respective color texture. The obtained height field map is applied to the models: initially, the texture is projected onto the target model. Depending on the surface bending, this procedure deforms the texture. To limit this effect, each model is particularly fixed by hand in correspondence of significant distortions and parts of the surface with complex geometry (like tight handles). Next, the vertices of the triangle mesh were raised based on the gray-scale value of the previously processed image along the normal vectors of the models. The same process is repeated for all the textures. A couple of examples of the conversion of a texture into a height map are depicted in Figure 5.9.

Finally, the models are slightly smoothed to minimize the perturbations in the color derived from the gray-scale conversion of the textures and the models are sampled with 50,000 vertices. Base models, height fields and examples of the final 3D models are shown in Figure 5.8. The dataset creation process is summarised in Figure 5.12.



(a) Texture 1(*left*), Height Map 1(*right*) (b) Texture 2(*left*), Height Map 2(*right*)

Figure 5.9: An example of the transformation process from texture to height map. On the *left* and *right* for both Figures (5.9a) and (5.9b), the original textures are shown, and the final height-map obtained, respectively, with the process explained in Section 5.2.10. This process can end with a binary image (i.e. black and white, as in Figure 5.9a, *right*) or a gray-scale one (like that in Figures (5.9b), *right*). Image source: [163].

Motivation for Using SHREC 2020 Geometric Reliefs Dataset/Retrieval Challenge: Given the nature of the textures selected , we think that in 3D pattern retrieval the most challenging issue is to deal with free-form models, possibly with more complex bending and non-trivial topology. Also, given the heterogeneity of the textures selected and the geometry of the base models, methods that perform well in this contest have a high chance of being equally valid in other contexts, with little to no changes.

The challenge proposed in this shape retrieval contest is to group the models only according to the geometric relief impressed on them, rather than their shape. In other words, a perfect score is obtained if a method is able to define 11 groups of 20 models each, each group with the models characterized by one of the 11 different geometric reliefs. See example of this expectation in Figure 5.10.

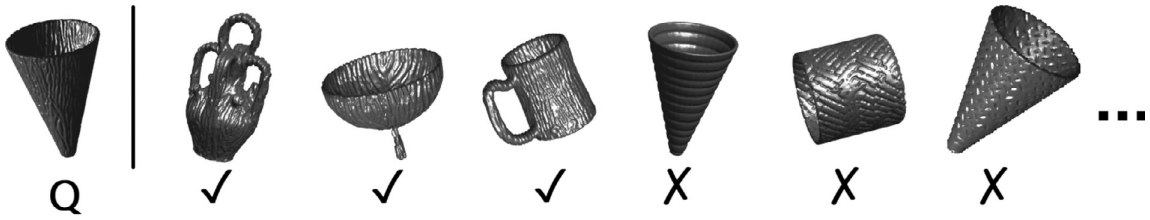
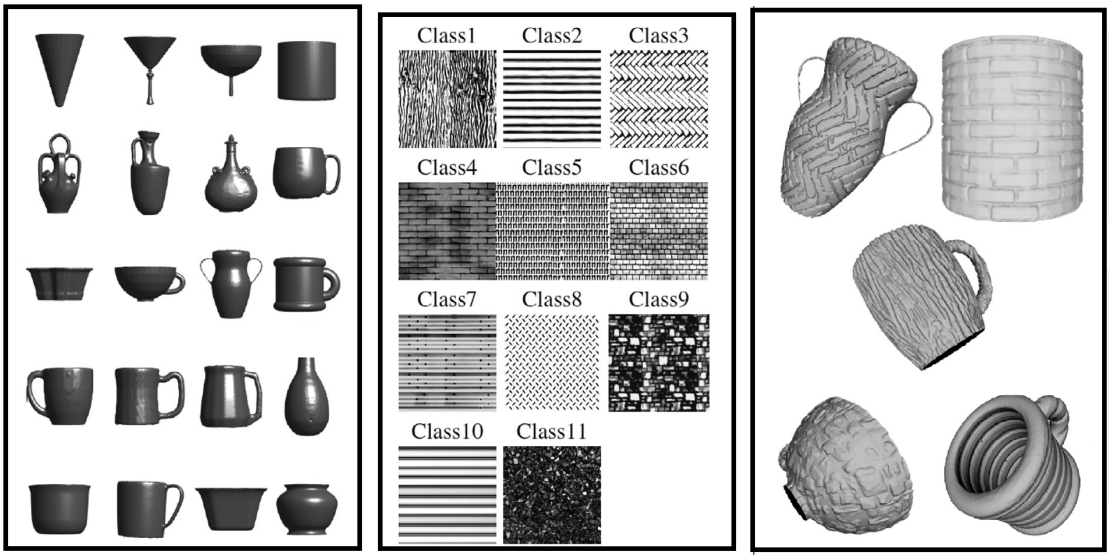


Figure 5.10: A visual representation of the challenge proposed in *SHREC'20 retrieval of 3D shapes with similar geometric reliefs* contest. A query model Q with a bark-like relief impressed on its surface is selected. In the ideal case, models with a bark-like relief are retrieved before than models with different reliefs, independently of the global geometry of the models. The "check" and "cross" marks highlight models that are relevant or non-relevant to the query. Image source: [163].



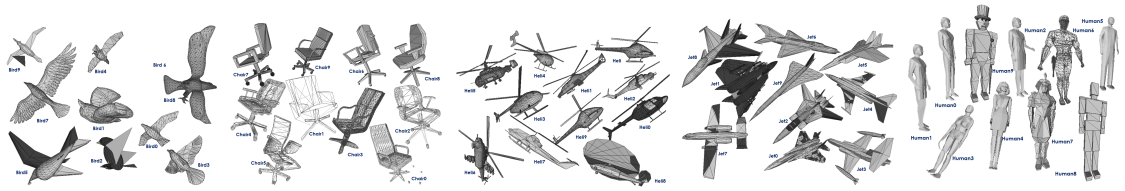
(a) Base Models (b) Textures (height maps) (c) Final Models (example)

Figure 5.11: (a): the 20 base models on which the reliefs are applied. (b): the 11 transformed textures used as height-fields on the base models (the brighter the color, the higher is the value of the field in that point). (c): a sample of the final models of the dataset of the contest. Image source: [163].

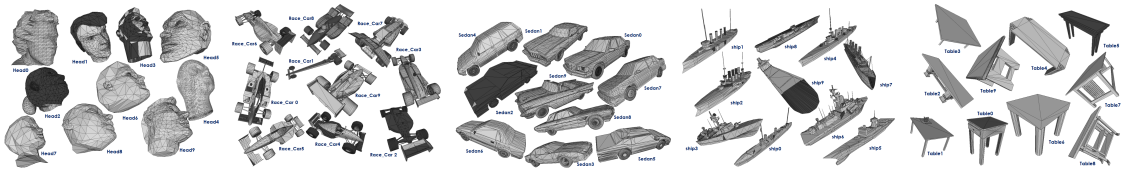
5.2.11 ShapeGoogle: Random, Generic 3D Shape Dataset

This group of datasets [38] are a collection of 200 triangular meshes made up of 20 classes of 3D shapes from different categories of commonly available objects such as Biplanes, Tables, Chairs (Dining-chairs, Desk-chairs), Flying-birds, Fishes, Human-heads, Human-arms, Humans, Handguns, Guitars, Helicopter, Jets, Swords, Cars (Sedan, Racing-cars), Potted-plants, Shelves, and Ships, all randomly selected from 3D shapes in the Princeton Shape Benchmark (PSB) [216, 191]. Each class holds 10 shapes. As a result of the smaller size of this dataset allowed us to conduct experiments to quickly test our initial set retrieval methods.

Motivation for Using the Random, Rigid/non-Rigid 3D Shape Dataset: As outlined in Table 5.1, most of the publicly available 3D shapes are contained in benchmarks datasets which are either rigid or non-rigid 3D shapes, but not generic



(a) Birds (b) Chairs (c) Helicopters (d) Jets (e) Humans



(f) Heads (g) Race Cars (h) Sedan Cars (i) Ships (j) Tables

Figure 5.12: Database of 200 3D shapes (ShapeGoogle) grouped into 10 classes from (5.12a) to (5.12j). Each class contains 20 shapes in the same category. This dataset consists of generic (i.e. it contains a mixture of rigid and non-rigid) 3D shapes.

(i.e. rigid and non-rigid) such as the SHREC'12 [131]. In the early stage of this research work, we needed some type of freely available dataset that would allow us to quickly evaluate the implementation of existing techniques for 3D shape matching and retrieval algorithms and assess the suitability of our earlier shape descriptors implementation for both rigid and non-rigid 3D shapes. This dataset, which was adapted from [38] became handy and particularly useful for these purposes.

5.3 Experimental Evaluations of The HAPPS Retrieval Method

In this section, we provide the experimental results of all experimental evaluations using our HAPPS retrieval method (predominantly HAPPS-1) with selected 3D datasets described in Section 5.2, and compare its retrieval accuracies to those of other state-of-the-art retrieval methods evaluated with the respective datasets, using about seven standard IR performance metrics described in Section 4.3.3, which are: NN, FT, ST, E, DCG, mAP, and PRC (see Section 4.3.3). In summary, for these experimental evaluations, we provide the results and overall retrieval performances of testing the HAPPS method on eight different standardized benchmark datasets for 3D shape retrieval: (i) SHREC'10 Non-rigid shape retrieval having 200 triangular meshes, (ii) SHREC'11 - shape retrieval contest of non-rigid 3D water-tight meshes with 600 3D objects, (iii) SHREC'17 PRoNTo, with 100 3D point clouds, (iv) SHREC'18 protein shapes benchmark for Protein Shape Retrieval Contest, with 2,267 protein conformers, (v) SHREC'19 protein shape retrieval challenge, having a dataset with 5,298 protein models (vi) SHREC'20 multi-domain protein shape retrieval challenge, having a dataset with 588 protein models, (vii) SHREC'12 - generic 3D shape retrieval with 1,200 rigid and non-rigid triangular meshes, and finally, (viii) SHREC'14 track - large scale comprehensive 3D shape retrieval, having 8,987 rigid and non-rigid 3D objects. Detailed description of all these datasets are presented in Section 5.2. Additionally, for each experimental run with our retrieval

method on some of the datasets, we present visualisation for the (dis)similarity matrix values returned by the retrieval algorithm (i.e. descriptor) and matching method (i.e. distance metric) for that experiment, in order to further examine how the algorithm performs across each shapes and classes for a given dataset. Details about the (dis)similarity matrix is described in Section 4.3.2.

5.3.1 Experiment 1: Evaluating The Retrieval Performances of The HAPPS-1 Method On SHREC'18 Protein Shape Dataset

For this experimental evaluation, we adopt six performance evaluation metrics, thus: the NN, FT, ST, E, DCG, as well as the PRC graph to evaluate the retrieval performances of the HAPPS-1 method on the SHREC'18 Protein shapes database. We refer the reader to Section 4.3.3 for more details regarding these performance metrics. At first, we compare our retrieval performance results of the HAPPS-1 method (run-1a), side-by-side, with the results of six other state-of-the-art retrieval methods reported in [121], which competed for the SHREC'18 Protein shape retrieval challenge.

Experiment 1: Parameter Settings and Configurations

The APPFD is the basis for the HAPPS method. In Section 4.2.1 we described the implementation processes involved in constructing the APPFD, which involves point cloud sampling, key points determination (via voxel-grid down-sampling), LSP selection around each key point, and multi-dimensional binning of the locally extracted APPF to form the APPFD. Each of these processes are characterised by parameters. For example, the parameters: N, vs, r, $b_{(APPFD)}$, respectively, represents: (i) the number of points, N to sample from 3D triangular mesh to form point cloud (we explain this in Section 3.2.3), (ii) the size of voxel grid, vs in which a collection of points are bucketed to form an occupied voxel grid where an average point is derived. These averaged points produce the final down-sampled point, which is a representative of all the points collected into the grid (we explain this in Section 3.3.5), (iii) the specified radius, r around a key point in which neighbouring points to the key point are selected to form a LSP (as explained in Section 4.2.1), (iv) the number of bins, $b_{(APPFD)}$ in each dimension of the multi-dimensional histogram used to collect the local APPF to form APPFD (we provide more details regarding this in Sections 4.2.1 and 4.2.1).

Basically, three key parameters of the APPFD: (vs, r, b) greatly affects the robustness performance of the final HAPPS retrieval method. However, the fourth parameter, N controls the outcome of vs and r. That is, the size of the selected LSP and the number of down-sampled points (key points) are directly proportional to the number of points samples in the point cloud input data. In our implementations, the r parameter is used in k-NN algorithm to control the radius around an interest point (key point, p_{k_i}) for which neighbouring points to p_{k_i} are selected to form a LSP for p_{k_i}. Secondly, the vs parameter is used by the voxel-grid down-sampling algorithm to determine the size of voxel for which points are collected and averaged during the down-sampling process explained in Section 3.3.5.

Dataset:	SHREC'18 Protein DB	**Parameter Settings**				
Experiment 1	**Algorithm(s)**	vs	r	$b_{(APPFD)}$	$b_{(HoGD)}$	**Metric**
run-1a	HAPPS-1	0.10	0.20	7	50	Cosine
run-1b	HAPPS-1	0.30	0.50	7	65	Cosine
run-1c	**HAPPS-1**	**0.20**	**0.50**	**8**	**65**	**Cosine**
run-1d	HAPPS-1	0.30	0.60	7	65	Cosine

Table 5.4: Parameter settings for four different experimental runs (run-1a to run-1d) with the HAPPS retrieval method on the SHREC'18 Protein dataset. Number of points samples, $N = 4,500$.

Considering that the overall performances of the APPFD and subsequently, the HAPPS retrieval methods strictly depends on these three parameter combinations: (vs, r, b), it is not immediately obvious which parameter settings/configurations would produce optimal results for the descriptors (i.e. APPFD or HAPPS) on a given dataset of 3D objects or retrieval problem. In order to determine this possibility (i.e. best parameters value combination), we had to perform extensive experiments on every dataset we tested our retrieval methods upon, each time, with a different parameters setting. Finally, we would have to evaluate the returned retrieval results as indicated in Figures 5.13 and Table 5.6, for example, before it becomes obvious which parameters value combination or configuration are best for a given dataset or retrieval challenge. It is important, however, to note that different *"best parameters setting"* are obtained for different 3D retrieval benchmark dataset or retrieval challenge. That is, the parameters setting or configuration which yields best retrieval performance results for 3D benchmark dataset A, for example, would not necessarily produce the same best performance result for, say, dataset B, using the same retrieval method. In our experimental evaluations with the HAPPS-1 retrieval method for the SHREC'18 Protein dataset, we examined the outcome of different settings (parameter values) for each of the above-mentioned three parameters, vs, r and b. We summarise these parameter settings in Table 5.4, and present results of these experimental runs in Table 5.6.

Experiment 1: Results and Discussion

We carried out several experiments to test the performances of the HAPPS-1 retrieval method on the SHREC'18 Protein shapes dataset and compare our results with existing ones, using various (dis)similarity metrics mentioned in Section 2.2.3 to perform spatial matching between two shape descriptors for every experiment, but presents only four results of such experiments with the Cosine distance metric. The SHREC'18 protein retrieval track defines four types of classes: (i) *Small* classes, (ii) *Medium* classes, (iii) *Large* classes, and (iv) *All* classes. We refer the reader to [121] for more details regarding this dataset classification. The *All* classes is the most important classification and for which ground-truth information (i.e. classification file) is publicly available, hence our evaluations are based only upon the *All* classification ground-truth. Overall, the HAPPS-1 method yields the best results in all

Method	Class	NN	FT	ST	E	DCG
3D-FusionNet [121]	All	0.6890	0.4040	0.4590	0.3660	0.6810
HAPT4 [121]	All	0.7700	0.4930	0.5840	0.4620	0.7550
SIWKS [121]	All	0.1990	0.1090	0.1890	0.1140	0.4520
DEM [121]	All	0.4210	0.2380	0.3190	0.2310	0.5550
WKS [121]	All	0.7170	0.4100	0.4900	0.3770	0.7010
GSGW [121]	All	0.5140	0.2610	0.3500	0.2470	0.5810
HAPPS-1 Run-1c	**All**	**0.8525**	**0.6669**	**0.7815**	**0.5783**	**0.8818**

Table 5.5: Quantitative performance evaluation results of seven different shape retrieval methods: only the best of the different variants of (i) all six different state-of-the-art retrieval methods submitted for the SHREC'18 protein retrieval contest, and (ii) our HAPPS-1 (run-1c), tested on the SHREC'18 protein shapes dataset using five standard IR metrics.

six performance evaluation criteria: NN, FT, ST, E, DCG, and PRC, over six other state-of-the-art methods submitted for the SHREC'18 protein retrieval contest. We summarise the results of this comparative analyses in Table 5.5 and Fig. 5.13. The HAPT method (i.e. HAPT4) followed ours (i.e. HAPPS-1) in performance rankings, while WKS and 3D-FusionNet became next, with similar performances. The GSGW, DEM, and SIWKS recorded lower performances. We demonstrate these results in the PRC plot shown in Figure 5.13.

According to [121], a total of six state-or-the-art retrieval methods (3D-FusionNet, HAPT, SIWKS, DEM, WKS, and GSGW) were evaluated for the SHREC'18 Protein shapes retrieval dataset using seven standard IR performance evaluation metrics: The nDCG and six others: NN, FT, ST, E, DCG, and PRC, described in Section 4.3.3. They reported a total of nine experimental results: Five from different algorithms and four variants of the HAPT algorithm (HAPT1, HAPT2, HAPT3, and HAPT4). Similarly, we carried out several experiments with our HAPPS-1 retrieval method on the SHREC'18 Protein dataset using all the performance metrics in [121], except the nDCG. However, we present the performance evaluation results of only four different most significant experimental runs (i.e. run-1a to run-1d, with each run combining different parameter values as presented in Table 5.4 and Figure 5.14). The parameters are $r = (0.20, 0.50, 0.60)$: the "radius-neighbouhood" floating value used to determine LSP size; $vs = (0.10, 0.20, 0.30)$: the "voxel-size" parameter used by the voxel-grid down-sampling technique described in Section 3.3.5; $b_{(APPFD)} = bm = 7, 8)$: the "bin size" used for the local-based APPFD "multi-dimensional" histogram descriptor; and finally $b_{(HoGD)} = b = (50, 65)$: the "bin size" used for the global-based "1-dimensional" histogram descriptor (see Section 4.2.2). These comprehensive evaluation results are presented in Table 5.6.

As indicated in Table 5.4, the four different experimental runs for the HAPPS-1 retrieval method combined different parameter values, thus:

- HAPPS-1run-1a [r:0.20, vs:0.10, $b_{(APPFD)}$:7, $b_{(HoGD)}$:50],
- HAPPS-1run-1b [r:0.50, vs:0.30, $b_{(APPFD)}$:7, $b_{(HoGD)}$:65],

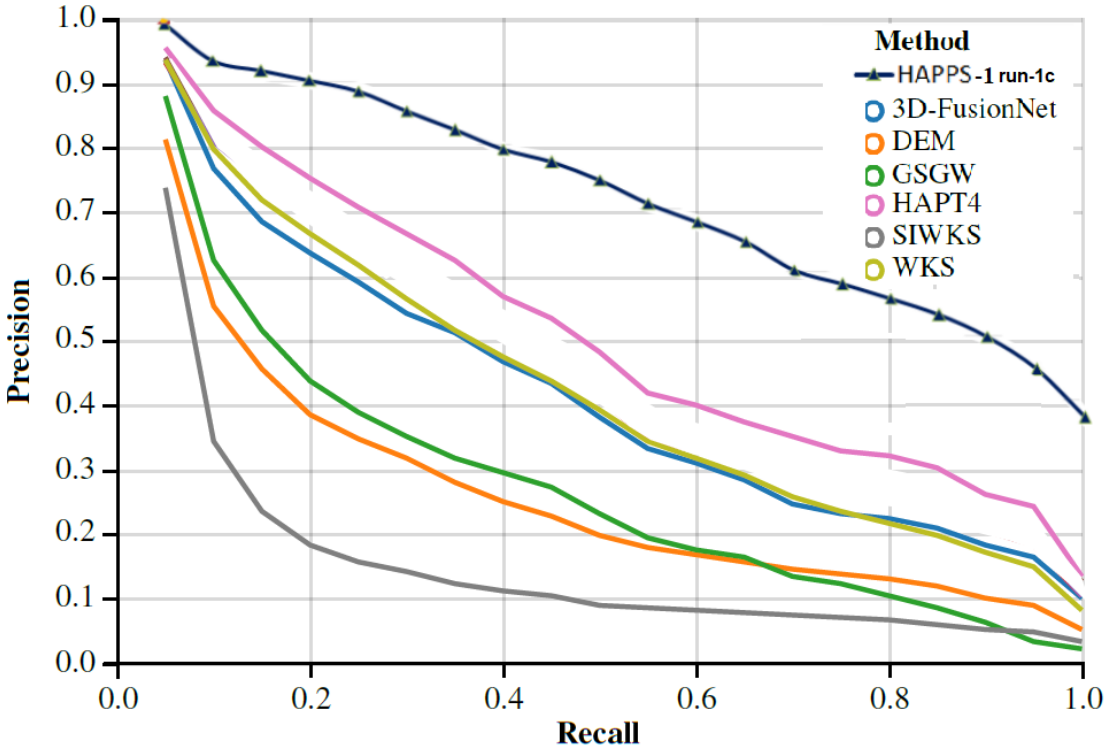


Figure 5.13: Precision-Recall Curve (PRC) plot of seven retrieval methods for SHREC'18 protein shapes retrieval dataset according to Table 5.5. Top-most PRC plot (Blue color) is that of our HAPPS-1 method (run-1c), while the other six plots are those of existing state-of-the-art methods in [121]. Here, we superimpose the PRC plot of HAPPS-1 method on only the best variants of the six existing methods in [121].

- HAPPS-1run-1c [r:0.50, vs:0.20, $b_{(APPFD)}$:8, $b_{(HoGD)}$:65],
- HAPPS-1run-1d [r:0.60, vs:0.30, $b_{(APPFD)}$:7, $b_{(HoGD)}$:65].

We compare these results (from HAPPS-1 retrieval method) side-by-side with all the other results in [121] having nine experimental runs. Therefore, we present a total of thirteen performance evaluation results for all experimental runs (nine from [121] and four from our work), for the SHREC'18 protein shapes retrieval dataset as shown in Table 5.6. In Table 5.6 and Figure 5.13, we show a summary of all results, for previously submitted methods in [121], side-by-side with those of our proposed method (the HAPPS-1) using the *All* classes classification ground-truth.

The four plots in Figure 5.15 shows the DM results of the four experimental runs of our HAPPS-1 retrieval method mentioned above and presented in Table 5.4. Figures 5.15b- 5.15d shows the DM of the three best performing experimental runs which are also revealed in the PRC plot in Figure 5.14 and the results in Table 5.6, while Figures 5.15a show worst performing run based on parameter combination (see the yellow coloured PRC in Figure 5.14). In these DM plots, the cooler (i.e. Blue) colours signifies more similarity, while hotter (Red) colours signifies less similarity, between pairs of 3D shapes in the SHREC'18 database. The abbreviations: *r*, *vs*

Method	Class	NN	FT	ST	E	DCG
3D-FusionNet [121]	All	0.6890	0.4040	0.4590	0.3660	0.6810
HAPT1 [121]	All	0.7130	0.4130	0.5340	0.4090	0.7190
HAPT2 [121]	All	0.7030	0.4390	0.5410	0.4150	0.7200
HAPT3 [121]	All	0.7120	0.4590	0.5600	0.4330	0.7340
HAPT4 [121]	All	0.7700	0.4930	0.5840	0.4620	0.7550
SIWKS [121]	All	0.1990	0.1090	0.1890	0.1140	0.4520
DEM [121]	All	0.4210	0.2380	0.3190	0.2310	0.5550
WKS [121]	All	0.7170	0.4100	0.4900	0.3770	0.7010
GSGW [121]	All	0.5140	0.2610	0.3500	0.2470	0.5810
HAPPS-1run-1a	All	0.4958	0.3563	0.4995	0.3249	0.6642
HAPPS-1run-1b	All	0.8507	0.6609	0.7807	0.5746	0.8797
HAPPS-1run-1c	All	0.8525	**0.6669**	**0.7815**	**0.5783**	**0.8818**
HAPPS-1run-1d	All	**0.8729**	0.6648	0.7699	0.5746	0.8812

Table 5.6: Quantitative evaluation results with five standard retrieval performance metrics, for seven distinct retrieval methods (6 state-of-the-art and our proposed method) evaluated on the SHREC'18 protein shapes dataset. Our HAPPS-1 and the HAPT methods submits four different variant of results, each with different parameter combinations. We used the Cosine distance metric to produce the DM for this experiment.

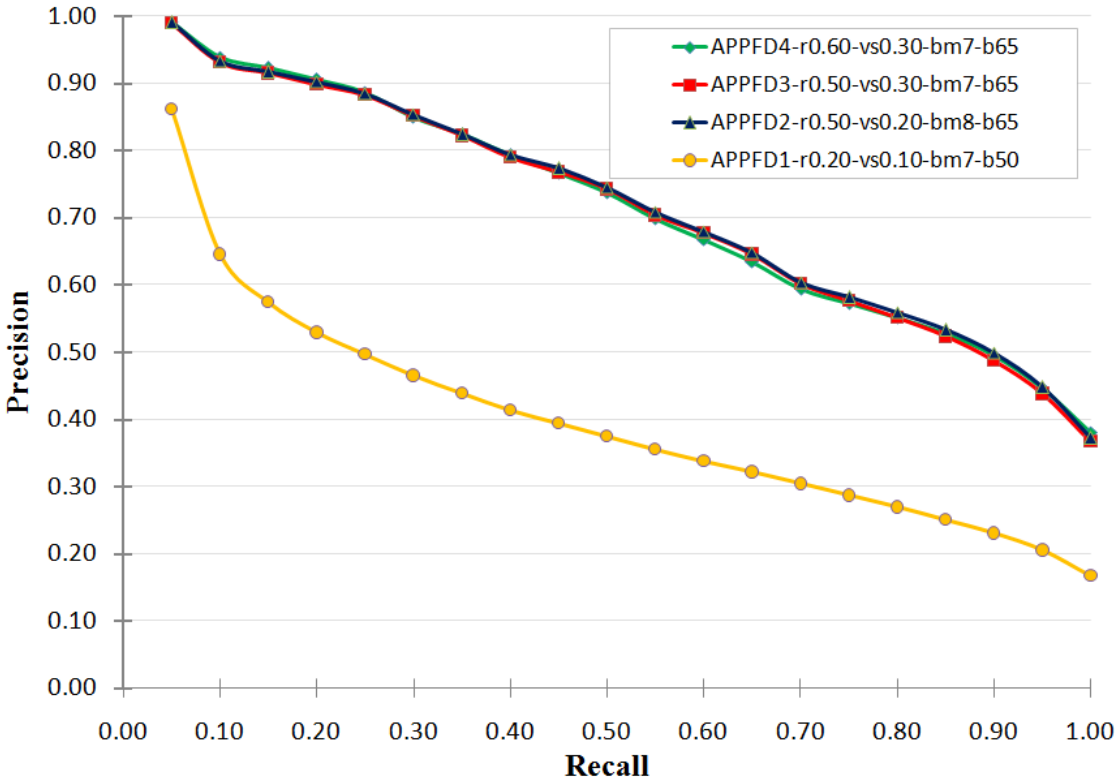
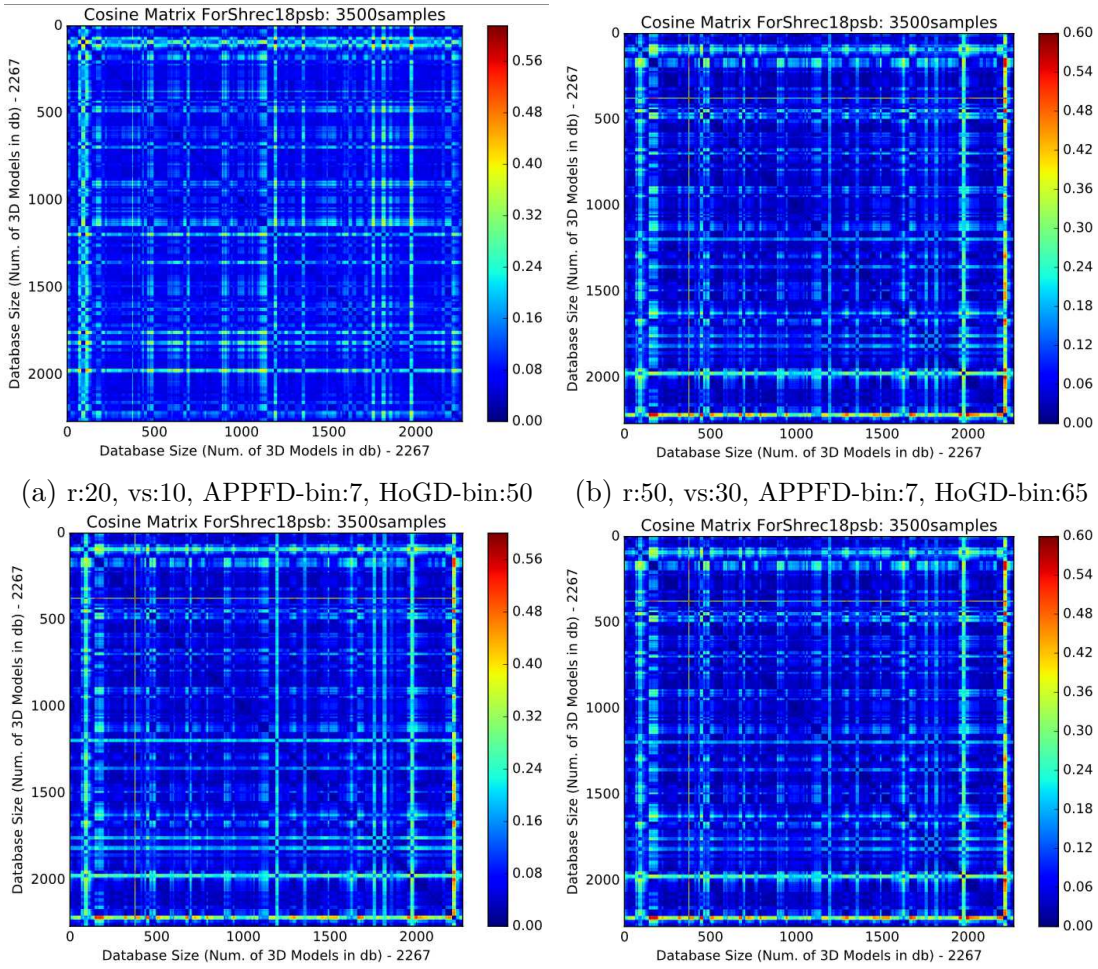


Figure 5.14: PRC plot of the HAPPS-1 method (run-1a to run-1d), on the SHREC'18 protein shapes retrieval.

APPFD-bin, and HoGD-bin are adequately explained in Section 5.3.1.



(a) r:20, vs:10, APPFD-bin:7, HoGD-bin:50

(b) r:50, vs:30, APPFD-bin:7, HoGD-bin:65

(c) r:50, vs:20, APPFD-bin:8, HoGD-bin:65

(d) r:60, vs:30, APPFD-bin:7, HoGD-bin:65

Figure 5.15: Distance matrix (2,267 x 2,267) plots of four different experimental runs (run-1a to run-1d) with our Hybrid: APPFD+HoGD (HAPPS) method on SHREC'18 Protein shapes dataset, using Cosine distance metric (mentioned in Section 2.2.3) for spatial matching, and the different parameter settings presented in Table 5.4. Note the difference in colour intensity between Figure 5.15a, which produced relatively poorer results, and the other three: Figures 5.15b - 5.15d, which produced higher and better results.

The PRC plot in Figure 5.14 clearly shows this interesting pattern of how the different parameter settings we used in Table 5.4 affects the retrieval performances of the HAPPS-1 method on the SHREC'18 Protein shapes dataset. It is interesting to see how the different parameters combination affects the retrieval performance of the APPFD and eventually, the HAPPS-1 methods. As the size of local surface region (i.e. LSP) around an interest point increases from $r = 0.20$ to $r = 0.60$, while the number of overall interest point (i.e. key points) decreased, i.e. voxel-sizes from $vs = 0.10$ to $vs = 0.30$, all performance statistics: NN, FT, ST, E, and DCG, increases in the positive direction (see Table 5.6). There is a performance gain for the FT, ST, E and DCG measures and drop for NN, as voxel-size, vs changes from $vs = 0.30$ to $vs = 0.20$ (as indicated in the third experimental run: HAPPS-1run-1c,

Table 5.6), while the size of LSP remains unchanged at $r = 0.50$, refer to Table 5.4.

In addition, we see from the PRC in Figure 5.14 that the APPFD-bin (i.e. $b_{(APPFD)}$) parameter has the highest influence on the overall results. Setting this parameter to 7 (Table 5.4) in conjunction with smaller number of key points and LSP controlled by the parameters: *vs* and *r*, respectively, produces a final *feature-vector* of size: $7^6 = 117,649$ and the result is HAPPS-1run1 (Table 5.6) which is comparatively very low compared to results of the other three experimental runs, where the sizes of LSPs and number of key points are larger. Another interesting observation in this experiment is the fact that setting the the APPFD-bin parameter to 8 instead produces a *feature-vector* of size: $8^6 = 262,144$, which is extremely high and defies the compactness characteristics of a suitable shape descriptor (see Section 2.3.1) while also impacting on computational time. However, the results returned (as shown in the PRC plot) reveals that with $b_{(APPFD)} = 7$ and fine-tuning the r and *vs* upward produces the same results, with a lower *feature-vector* size. Although it is somehow difficult to see (in Figure 5.15) how different 3D shapes (objects) relate with each other in terms of similarity, as explained in Section 4.3.2 because fines details are lost in the plot due to the size of the SHREC'18 protein shapes database, which has 2,267 3D objects. However, a careful observation of the four different DM plots would reveal that Figures 5.15b to 5.15d are generally very similar (with darker blue colours) than Figure 5.15a as confirmed by Table 5.6 and Figure 5.14.

Finally, as shown by HAPPS-1run-1d in Table 5.6, increasing the size of LSP from $r = 0.50$ to $r = 0.60$ regains back the confidence of the NN statistic. Although we did not have time to run further experiments, but from the trend revealed in these experimental runs and several other not reported here, we conjecture that combining the parameters $r = 0.60$, $vs = 0.20$, $bm = 8$, and $b = 65$ for the APPFD retrieval method would produce best results for all performance statistics on the SHREC'18 protein shapes dataset. Also, following the trend reported by [121] and the results we have achieved, our proposed HAPPS-1 is capable of maintaining overall best performances for the other three *Classes*: Small, Medium, and Large.

Experiment 1: Comparative Analysis of The HAPPS-1 Retrieval Method with SHREC'18 Protein Shapes Dataset

Following performances ranking of our APPFD/HAPPS-1 method on the SHREC'18 dataset is the Histograms of Area Projection Transform (HAPT) method, which also showed the impressive results for all IR performance statistics compared to the other state-of-the-art methods reported by [121]. Unlike every other state-of-the-art methods evaluated with the SHREC'18 dataset, our points sampling, and features extraction approaches returns a final shape descriptor that is computationally efficient, yet robust enough to accurately distinguish between different protein conformers. For example, a typical protein conformer in this dataset contains an average of 100,000 *vertices* (i.e. points) and 300,000 *faces*, which takes up to 10GB of storage capacity, and our proposed Hybrid method (HAPPS-1 used only 3,500 points (as indicated in Figure 5.15), with a further down-sampling to an average of 195 key points, p_{K_i}, while being very capable of providing results that outperforms several of these existing state-of-the-art methods on all retrieval performance

statistics, as presented and discussed above.

5.3.2 Experiment 2: Evaluating The Retrieval Performances of The HAPPS-1 Method On SHREC'17 PRoNTo Dataset

The aim of the SHREC'17 PRoNTo track was to create a fair benchmark to evaluate the performances of shape retrieval methods (descriptors) on the non-rigid point-cloud shape retrieval problem and to test shape descriptors which are able to capture the main characteristics of 3D objects. In this experimental evaluation, we demonstrate the retrieval performances of our proposed HAPPS-1 retrieval method on the SHREC'17 dataset and compare our best results with results from the best performing methods submitted for the SHREC'17 retrieval task in [142]. For this retrieval challenge, the performances of eight different state-of-the-art retrieval methods were submitted for evaluation by eight participants, and each participant submitted a minimum of 1 and a maximum of six different retrieval results (where each result is an outcome of different experimental run of the same method, but with different parameter settings). Therefore, a total of 31 experimental runs were submitted for evaluation by the retrieval task in [142].

Experiment 2: Parameter Settings and Configurations

The parameter settings and combinations for this particular experimental evaluation are already outlined in Table 5.8 and fully explained in Section 5.3.1. However, the maximum number of points, N for each 3D scan model in the PRoNTo dataset is between 3,500 - 4,200. This number of points is relatively small, which means that the contribution for the global HoGD descriptor to the final hybrid HAPPS retrieval method is very minimal, as explained in Section 4.2.2. In Table 5.8, we report our experimental results with experimental run-2a, having the parameter settings: $vs = 0.30$, $r = 0.50$, $b_{(APPFD)} = 8$, and $b_{(HoGD)} = 50$ (see Table 5.8), because these parameter settings are found to return better results compared to other values, based on our experimental findings (not reported in this thesis). Finally, setting the $b_{(HoGD)}$ between 45 and 65 bins, as in experimental runs-2b, run-2c and run-2d, as in Table 5.8, had negligible impact on the overall performance of the HAPPS-1, as already explained, but increasing the $b_{(APPFD)}$ parameter from 7 to 8 bins improved retrieval results of the HAPPS-1 algorithm for the PRoNTo dataset.

While computing the APPFD for the HAPPS-1 retrieval method used with the SHREC'17 PRoNTo dataset, we adopt the steps specified in Algorithm 4, since the PRoNTo dataset contains raw 3D point cloud, $\overline{P}$ as input. Therefore, the point cloud sampling algorithm was not needed for this dataset. Considering that this dataset is characterised by noisy point cloud, we first applied smoothing filter using the MLS algorithm by [158], with radius parameter: $r_{mls} = 0.05$. Arguably, we believe that smooth surfaces offer a better representation for 3D surfaces, thereby increasing the accuracy of features extracted from them. Next, we estimated the surface normals, N_s for $\overline{P_s}$, using Algorithm 1, and computed K key points, $\{p_{k_i}, i = 1 : K\}$) with the voxel down-sampling technique described in Section 3.3.5, around which LSPs, $\{P_i, i = 1 : K\}$) and their corresponding normals, $\{N_i, i = 1 : K\}$) are extracted,

Author [142]	Method	NN	FT	ST	E	DCG
Boulch	SnapNet	0.8800	0.6633	0.8011	0.3985	0.8663
Giachetti	POHAPT	0.9400	0.8300	0.9144	0.4156	0.9419
	BPHAPT	0.9800	0.9111	0.9544	0.4273	0.9743
Li	m3DSH-1	0.4000	0.1656	0.2778	0.1824	0.4802
	m3DSH-2	0.4400	0.1867	0.2856	0.1932	0.4997
	m3DSH-3	0.4400	0.1767	0.2878	0.1917	0.5039
	m3DSH-4	0.4000	0.1511	0.2511	0.1712	0.4659
	m3DSH-5	0.4200	0.1722	0.2767	0.1815	0.4930
	m3DSH-6	0.4100	0.1700	0.2678	0.1712	0.4848
This Thesis	HAPPS-1run-2a-kld	0.9900	0.7867	0.8744	0.4132	0.9367
	HAPPS-1run-2b-cos	0.9900	0.7767	0.8844	0.4171	0.9325
	HAPPS-1run-2c-cos	0.9900	0.7767	0.8844	0.4171	0.9325
	HAPPS-1run-2d-cos	0.9900	0.7767	0.8844	0.4171	0.9325
Limberger	GL-FV-IWKS	0.8200	0.5756	0.7244	0.3595	0.8046
	GL-SV-IWKS	0.7000	0.5267	0.6678	0.3327	0.7562
	MFLO-FV-IWKS	0.8900	0.7911	0.8589	0.4024	0.9038
	MFLO-SV-IWKS	0.9000	0.7100	0.7933	0.3702	0.8765
	PCDL-FV-IWKS	0.8200	0.6656	0.7978	0.3976	0.8447
	PCDL-SV-IWKS	0.8900	0.6656	0.7911	0.3732	0.8613
Sipiran	SQFD(HKS)	0.2900	0.2244	0.3322	0.2176	0.5226
	SQFD(WKS)	0.5400	0.3111	0.4467	0.2507	0.6032
	SQFD(SIHKS)	0.2900	0.2533	0.4133	0.2590	0.5441
	SQFD(WKS-SIHKS)	0.5000	0.3100	0.4500	0.2634	0.6000
	SQFD(HKS-WKS-SIHKS)	0.3900	0.2844	0.4389	0.2624	0.5722
Tatsuma	CDSPF	0.9200	0.6744	0.8156	0.4005	0.8851
Tran	BoW-RoPS-1	1.0000	0.9744	0.9967	0.4390	0.9979
	BoW-RoPS-2	1.0000	0.9778	0.9933	0.4385	0.9973
	BoW-RoPS-DMF-3	**1.0000**	**0.9778**	**0.9978**	**0.4390**	**0.9979**
	BoW-RoPS-DMF-4	1.0000	0.9778	0.9978	0.4390	0.9979
	BoW-RoPS-DMF-5	1.0000	0.9733	0.9978	0.4390	0.9979
	BoW-RoPS-DMF-6	1.0000	0.9733	0.9978	0.4390	0.9979
Velasco	AlphaVol1	0.7900	0.5878	0.7578	0.3980	0.8145
	AlphaVol2	0.7800	0.5122	0.6844	0.3751	0.7673
	AlphaVol3	0.7700	0.4567	0.6467	0.3629	0.7364
	AlphaVol4	0.7000	0.4356	0.6111	0.3454	0.7148

Table 5.7: Quantitative evaluation results of five standard retrieval performance measures of thirty-five methods computed for the SHREC'17 PRoNTo dataset. Thirty-one different runs from eight authors, and four experimental runs of our proposed HAPPS-1 method.

within a specified radius, $r = 0.40 - 0.50$ for each p_{k_i}.

Dataset:	SHREC'17 [269]	**Parameter Settings**				
Experiment 2	**Algorithm(s)**	vs	r	$b_{(APPFD)}$	$b_{(HoGD)}$	**Dist.Metric**
run-2a	HAPPS-1	0.30	0.50	8	50	KLD
run-2b	HAPPS-1	0.30	0.50	7	45	Cosine
run-2c	HAPPS-1	0.30	0.50	7	50	Cosine
run-2d	HAPPS-1	0.30	0.50	7	65	Cosine

Table 5.8: Parameter settings for four different experimental runs with the HAPPS-1 retrieval method on the SHREC'17 PRoNTo dataset. Number of points for each point cloud, $N \approx 3,000 \rightarrow 4,200$.

Experiment 2: Results and Discussion

We carried out several experiments with the HAPPS-1 retrieval method using different parameter settings in each experimental run on the SHREC'17 PRoNTo dataset, and compared four different experimental runs of our retrieval results: run-2a HAPPS-1, run-2b HAPPS-1, run-2c HAPPS-1, and run-2d HAPPS-1 (see Table 5.8), side-by-side with those in [142]. To be able to see how the results of the HAPPS-1 retrieval method compares with all the different results in [142] we present all the results of this comparison in Table 5.7, which shows a total of 35 quantitative evaluation results: 31 different runs from eight authors (participants), and four runs from our proposed method. Note that the best of the results for each participants are highlighted in light-gray colour and the overall best results of all the methods and participants are additionally presented in bold fonts.

Author [142]	Method	NN	FT	ST	E	DCG
Boulch	SnapNet	0.8800	0.6633	0.8011	0.3985	0.8663
Giachetti	BPHAPT	0.9800	0.9111	0.9544	0.4273	0.9743
Li	m3DSH-3	0.4400	0.1767	0.2878	0.1917	0.5039
This Thesis	HAPPS-1run-2a-kld	0.9900	0.7867	0.8744	0.4132	0.9367
Limberger	MFLO-FV-IWKS	0.8900	0.7911	0.8589	0.4024	0.9038
Sipiran	SQFD(WKS)	0.5400	0.3111	0.4467	0.2507	0.6032
Tatsuma	CDSPF	0.9200	0.6744	0.8156	0.4005	0.8851
Tran	**BoW-RoPS-DMF-3**	**1.0000**	**0.9778**	**0.9978**	**0.4390**	**0.9979**
Velasco	AlphaVol1	0.7900	0.5878	0.7578	0.3980	0.8145

Table 5.9: Quantitative evaluation measures of best eight of thirty-one state-of-the-art retrieval methods, including our proposed HAPPS-1 retrieval method (run-2a), using five standard performance measures on the whole of SHREC'17 PRoNTo dataset.

A summary of only the best performing retrieval methods (i.e. best experimental runs) from all the submission to the SHREC'17 PRoNTo tasks, including

the best result of the HAPPS-1 experimental run are presented in Table 5.9 for clarity. Note, the PRC plot in Figure 5.16 confirms the summarised results in Table 5.9. Here, the retrieval performances of the HAPPS-1 retrieval method are compared with eight (out of thirty-one) best-performing state-of-the-art methods for the PRoNTo dataset. For this experimental evaluation, we observe that the best performance evaluation result of our proposed HAPPS-1 method outperforms the results of six existing state-of-the-art methods in all five quantitative performance evaluation measure (NN, FT, ST, E, and DCG), and ranks third in overall performances, next to the results presented by the BPHAPT and BoW-RoPS-DMF-3 retrieval methods [142] (see Table 5.9 and the PR plot in Fig. 5.16). We see that the BoW (BoW-RoPS-DMF-3) method outperforms every other method, followed by BPHAPT, then our HAPPS-1 method for all the quantitative evaluation statistics. However, for the NN statistic the HAPPS-1 method ranked second overall, higher than the BPHAPT, while the MFLO-FV-IWKS method recorded higher result than ours in the FT statistic.

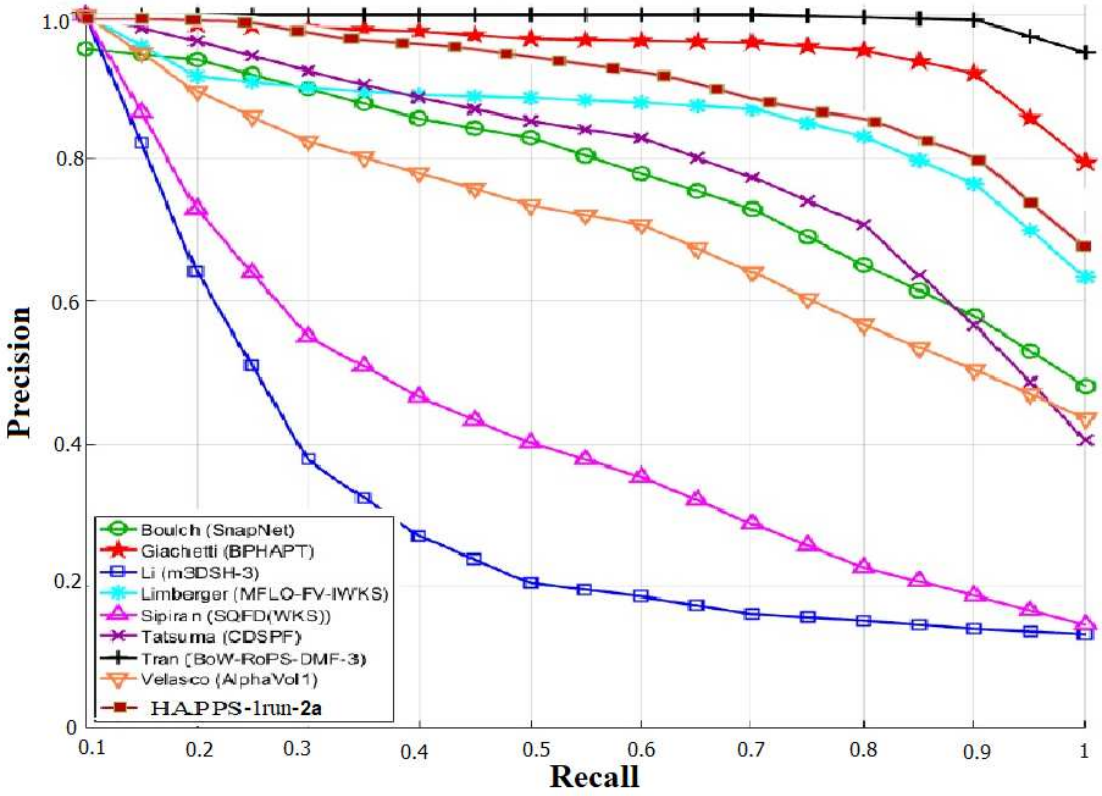
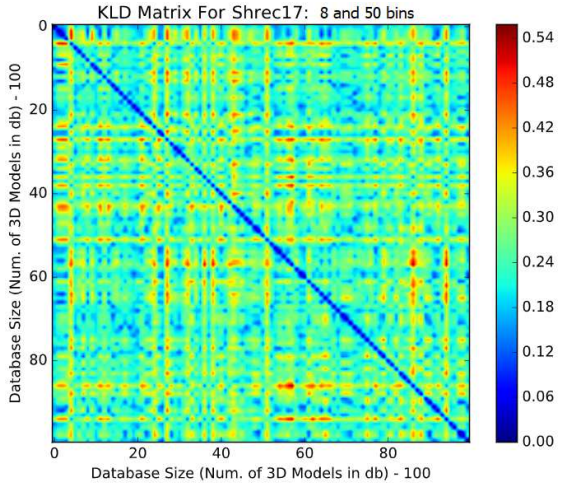


Figure 5.16: PRC plot of HAPPS-1 retrieval method *vs* PRC plots of only the best retrieval methods (out of 31 submitted methods) from each of the eight authors (participants), for the SHREC'17 PRoNTo dataset [142], according to Table 5.9). We super-imposed the PRC plot of the HAPPS-1 method (run-2a) on the original PRC plots presented in [142].

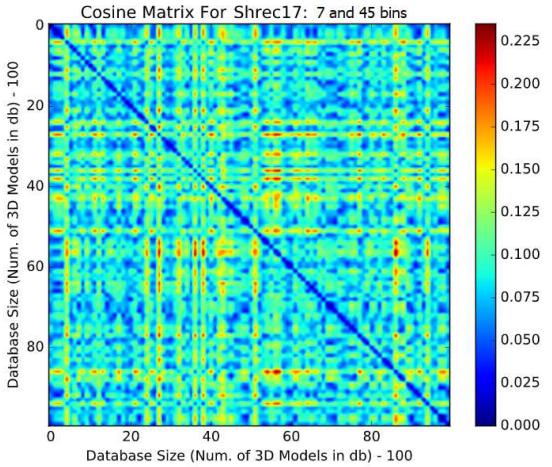
In the SHREC'17 shape retrieval track, four methods: (i) MFLO-FV-IWKS, (ii) SQFD(WKS), (iii) CDSPF and (iv) BoW-RoPS-DMF-3 computed local features, while the other half: (v) BPHAPT, (vi) SnapNet, (vii) m3DSH-3 and (viii) Al-

phaVol1 computed global features according to [142]. We instead adopted a hybrid (local and global) approach: the HAPPS method. From Table 5.7 and Table 5.9, we see that Tran's method ranked first place with *"local features"*, Giachetti's method with *"global features"*, second place and our HAPPS-1 method with *"hybrid features"*, third overall. According to [140], the DCG is an incredibly good and stable quantitative statistic for evaluating shape retrieval methods, and we can see that only four (out of 8) methods (HAPPS-1, BoW-RoPSDMF-3, BPHAPT and MFLO-FV-IWKS) has DCG above 90%, with Tran's methods surprisingly having DCG values greater than 99%. The first deduction from this observation is that all three features extraction approach (local, global and hybrid) are capable of accurately representing non-rigid shapes. However, further qualitative analysis of the computational approaches of each of these three retrieval methods (i.e. HAPPS-1, BoW-RoPS-DMF-3 and BPHAPT) reveals that our HAPPS-1 retrieval method could as well rank in the first position instead of third if we implement some preprocessing functions, such as increasing the density of the point cloud data in the PRoNTo dataset (as is the case with the BoW-RoPS-DMF-3 method) or meshing the point cloud to provide a better surface representation devoid of holes and noise present in the original PRoNTo dataset (as is the case with the BHAPT method), before feature extraction step is considered. According to [142], the BoW-RoPS-DMF-3 method densified each point cloud to reduce significant difference in the density between different parts before feature extraction, which enabled each point cloud to have better surface representation as a result of high LoD, and hence more accurate feature extraction for the final descriptor. Similarly, the BHAPT retrieval method utilised an approach where all 3D point cloud in the PRoNTo dataset are first meshed in the preprocessing step to produce closed mesh surfaces, making the BHAPT descriptor robust against inaccuracies due to holes that existed in the original point cloud. Meshing these objects also provides better surface representation for effective feature extraction.

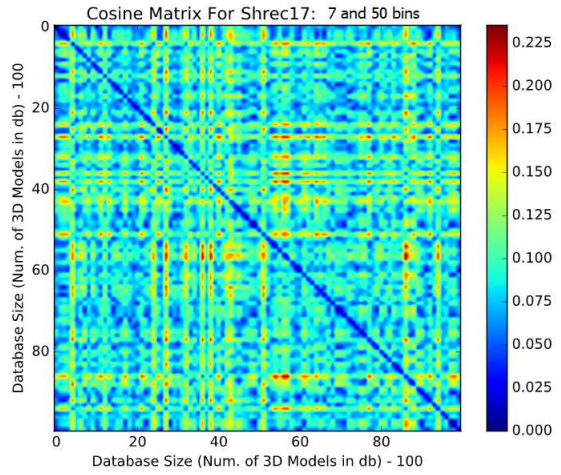
On the other hand, our HAPPS-1 method was applied directly on the PRoNTo dataset without up-sampling (to densify or increase its LoD) or meshing to provide a better surface representation devoid of holes and noise. Although we applied only the smoothing filter in our preprocessing step, we do not see this step making much difference that could affect the accuracy of our feature extraction process. In addition, the BoW paradigm has proven to be one of the most effective technique for combining local features. Our APPFD algorithm, which is the basis of the HAPPS-1 retrieval method, however, did not do a good job in combining its local feature compared to the BoW approach. We also notice from Table 5.8 that experimenting with different number of $b_{(HoGD)}$ bins (45 - 65) for the HoGD algorithm made no difference on the performance of the HAPPS-1 method as shown by our experimental runs (HAPPS-1run-2b-cos, HAPPS-1run-2c-cos, HAPPS-1run-2d-cos) in Table 5.7, but changing the $b_{(APPFD)}$ bin from seven to eight witnessed slight performance gain with the NN, FT, and DCG as seen in HAPPS-1run-2a-kld. Considering all the above analysis, we therefore conclude that our HAPPS-1 method performed the best in terms of computational efficiency, with the understanding that it takes extra computational resources and time for other methods to pre-process (up-sample/densify and meshing of the point cloud data) each point cloud object. Finally, given that we include the meshing or up-sampling preprocessing steps for



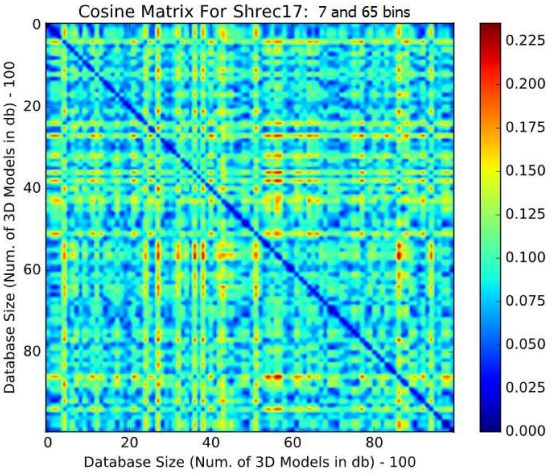
(a) r:50, vs:30, APPFD-bin:8, HoGD-bin:50



(b) r:50, vs:30, APPFD-bin:7, HoGD-bin:45



(c) r:50, vs:30, APPFD-bin:7, HoGD-bin:50



(d) r:50, vs:30, APPFD-bin:7, HoGD-bin:65

Figure 5.17: Distance matrix (100 x 100), plots of four different experimental runs (run-2a to run-2d) of our HAPPS-1 method on SHREC'17 PRoNTo dataset. Figure 5.17a shows distance matrix results for KLD (dis)similarity metric, while Figures 5.17b - 5.17d shows the DM plots using Cosine (dis)similarity metric. Cooler colours (Blue) signifies more similarity, while hotter colours (Red) signifies less similarity, between pairs of 3D shapes in the database. The parameters are explained in Section 5.3.1.

our APPFD computation and adopt a more effective local feature combination approach to combining our locally extracted APPF, the HAPPS-1 retrieval method has a very great chance of outperforming all other state-of-the-art retrieval methods for the SHREC'17 PRoNTo dataset. However, this is a consideration for further work on this kind of retrieval challenge with the HAPPS-1 method.

In Figure 5.17 we show the DM plot of the four experimental runs (HAPPS-1run-2a-kld, HAPPS-1run-2b-cos, HAPPS-1run-2c-cos, and HAPPS-1run-2d-cos) of our HAPPS-1 method on the SHREC'17 PRoNTo dataset, as reported in Table 5.7. In this experiment evaluation, we performed matching with different distance metrics to see how the different metric described in Section 2.2.3 affects performs with our descriptor. We report results for the Cosine metric (see Section 2.2.3) and Kullback-Leibler Divergence (see Section 2.2.3). We see how the plot for HAPPS-1run-2a-kld in Figure 5.17a is different from the rest (Figures 5.17b - 5.17d), which produces comparable results according to Table 5.7. In this section, we have been able to present the results of our experimental runs for the SHREC'17 PRoNTo dataset and compare them with several other state-of-the-art retrieval methods. However, we focus our evaluation analysis on the four best performing methods, which includes our method. For performance evaluation analysis of the other methods whose results we present in Table 5.7, we refer the reader to [142].

5.3.3 Experiment 3: Evaluating The Retrieval Performances of The HAPPS-1 Method On SHREC'10 Dataset

In this section, we present the results of our HAPPS-1 retrieval method alongside the results of three other state-of-the-art methods, from three participants, who competed for the SHREC'10 retrieval of Non-rigid 3D shape challenge, which involves 200 3D objects. A total of six different experimental runs (MRBF-DSIFT-E, BF-DSIFT-E, DMEVD_run1, DMEVD_run2, DMEVD_run3, and CF, respectively) were submitted for the SHREC'10 retrieval track, where two experimental runs (MRBF-DSIFT-E, BF-DSIFT-E), three experimental runs (DMEVD_run2, DMEVD_run3), and one experimental run, were submitted by Ohbuchi, Smeets, Wuhrer, respectively, according to Table 1 in [140]. Five quantitative performance evaluation statistics (measures) were adopted for this dataset, which are: NN, FT, ST, E, and DCG, and the PRC plot described in Section 4.3.3. Essentially, the HAPPS-1 method is evaluated against the above-mentioned six methods for the SHREC'10 benchmark dataset and retrieval challenge [140]. First, we are comparing the retrieval performances of our HAPPS-1 method with the best experimental run (HAPPS-1,run-3a) with the retrieval performance of the best experimental run for each of the methods from the participants of the SHREC'10 retrieval challenge.

As a generalised concept, the performance results presented for all our experimental evaluations (including those from other authors who submitted their methods for evaluation in the respective shape retrieval challenges) are obtained by applying each retrieval method, such as the HAPPS method for example, to calculate the (dis)similarity between every two objects in a given dataset and then generating corresponding (dis)similarity matrices. We have provided detailed explanations regarding this concept in Section 4.3.2. For the SHREC'10 dataset which contains

200 3D objects, the matrix is composed of 200×200 floating point numbers, where the number at position (i, j) represents the (dis)similarity between models i and j. In Figure 5.18, we visualize the distance/(dis)similarity matrix for the results of our experimental run (run-3a) with the HAPPS-1 method (presented in Table 5.10). In Section 4.3.2, we already provide adequate explanation about what the colours in the DM plot represents. Given the (dis)similarity matrix for each retrieval method and experimental run, the retrieval performance of the method can be quantitatively evaluated based on the IR performance evaluation statistics mentioned in Section 4.3.3.

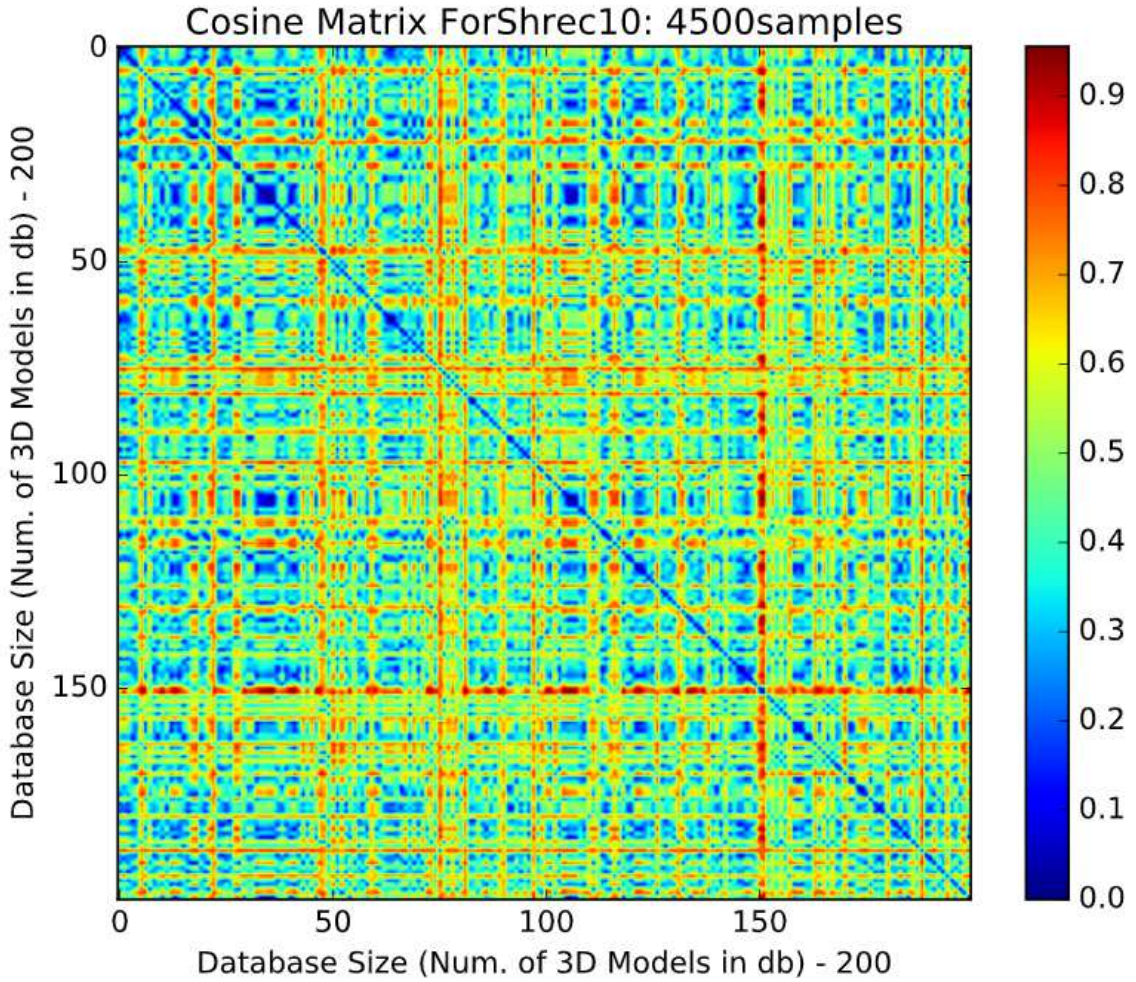


Figure 5.18: Distance Matrix for experimental run-3a with the Cosine Distance metric, whose results are presented in Table 5.10, using the HAPPS-1 retrieval method on the SHREC'10 dataset, according to the parameters in Table 5.11.

Experiment 3: Results and Discussion

The quantitative retrieval results reported in Table 5.10 recorded high overall performances with both the HAPPS-1 and DMEVD_Run1 methods for the NN statistic, while the MR-BF-DSIFT-E method by Ohbuchi outperforms all other methods for the FT, ST and E statistics. Considering the DCG statistic, the HAPPS-1, DMEVD_Run1, and MR-BF-DSIFT-E measured higher than 90% while Ohbuchi

Author	Method	NN	FT	ST	E	DCG
Ohbuchi [140]	MR-BF-DSIFT-E	0.9850	**0.9092**	**0.9632**	**0.7055**	0.9763
Smeets [140]	DMEVD_Run1	**1.0000**	0.86117	0.9571	0.7012	**0.9773**
Wuhrer [140]	SQFD(WKS)	0.9200	0.6347	0.7800	0.5527	0.8781
This Thesis	**HAPPS-1 run-3a**	**1.0000**	0.8063	0.8837	0.6463	0.9454

Table 5.10: Retrieval results of five quantitative performances statistics revealing the retrieval performances of four methods: result of our HAPPS-1 method (experimental run-3a) and best result from each of the three participants who's methods were evaluated for the SHREC'10 dataset.

and Smeets methods are remarkably close, with Smeets' method taking the lead. Overall, the MR-BF-DSIFT-E method by Ohbuchi recorded the highest performance, followed by DMEVD_Run1 method by Smeets, then our HAPPS method. However, it is interesting to see that our HAPPS-1 method and DMEVD_Run1 method by Smeets recorded the highest performances of 100% with the NN statistic, outperforming the MR-BF-DSIFT-E method by Ohbuchi and all others. Similar observation can be seen from Figure 5.19 where the PR curves by the first two methods by Ohbuchi and Smeets are remarkably close, followed by ours. It is however difficult to say from this plot which of the two methods obtained the best overall performance, as they seem robust to most of the non-rigid deformations present in this dataset, including the HAPPS-1 method with nearly the same performances as these two. Interestingly, also, we find that Ohbuchi used the BoW approach to represent its features, which again confirms the suitability of the BoW model for non-rigid shape retrieval, including being one of the best approaches for combining local features and/or local shape descriptors.

Considering the two methods (MRBF-DSIFT-E and DMEVD_Run1) whose performances slightly outranked our HAPPS-1, we note from [140] that unlike the MR-BF-DSIFT-E method which uses global isometric-invariant features, the MRBF-DSIFT-E and our HAPPS-1 method uses local features. However, the MR-BF-DSIFT-E method apply an unsupervised machine learning algorithm (i.e. manifold ranking) which can be applied to enhance the results of any other approaches, including features extracted for the APPFD method. In addition, this method employs the BoW technique for its local features' combination, which like the Bow-RoPSDMF-3 method analysed in Section 5.3.2, contributed to it being the best overall retrieval method for the PRoNTo dataset. Note that the manifold ranking used by the MR-BF-DSIFT-E method is time consuming [140] unlike our techniques which are not. On the other hand, the DMEVD_Run1 method adopted geodesic distance measure between pair of points on the 3D mesh surfaces, which involves solving the complex Eikonal equation and fast marching algorithm, including their use of the heat equation to calculate the diffusion distance matrix. While this approach is most effective for describing non-rigid surfaces (unlike spatial Euclidean distance as with our HoGD algorithm), it is computationally expensive. In order to have an idea of how expensive, we provide further analyses regarding the computational times of for our proposed method in Sections 5.4.1 and 5.4.2. Whereas, given very low number of points sample for each 3D objects in the SHREC'10 dataset, which we

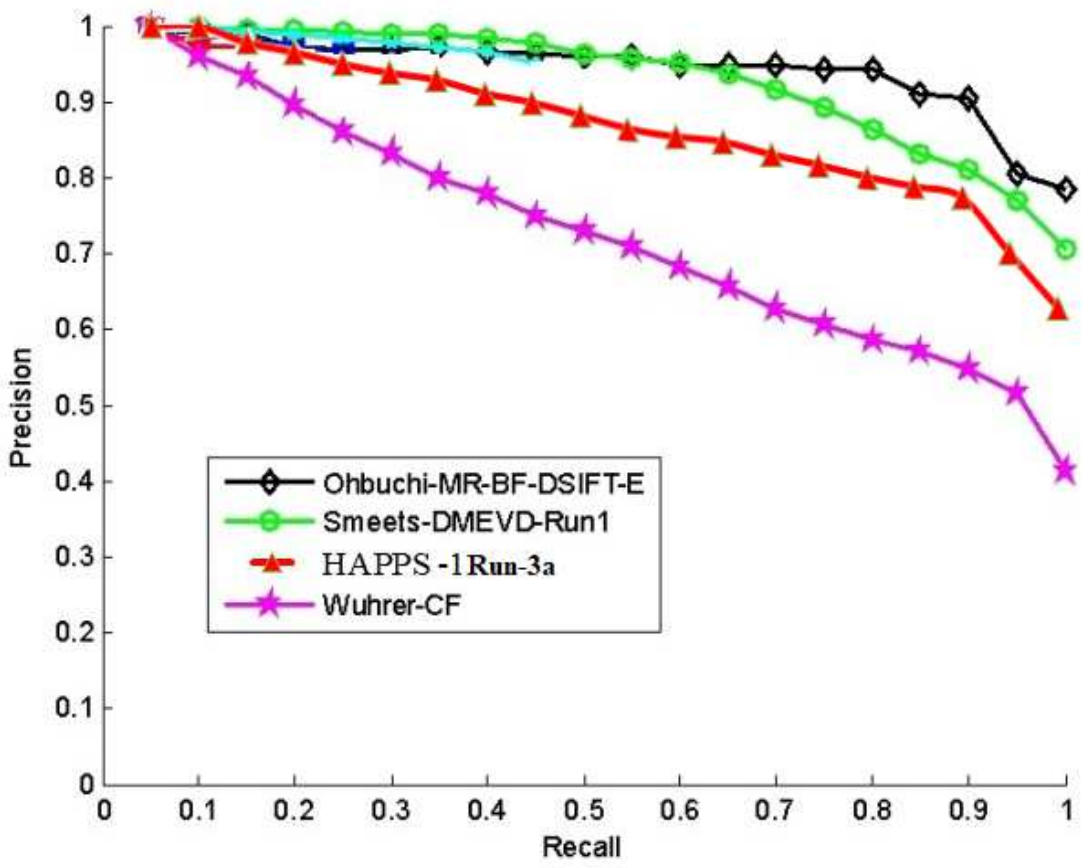


Figure 5.19: PRC plot showing the retrieval results of our HAPPS-1 retrieval method (run-3a), including only the best methods from all three authors (participants) for the SHREC'10 retrieval challenge, according to Table 5.10. The PRC plot for the HAPPS-1 method is shown in Red colour, and super-imposed on the original PRC plot in [140].

also used to compute the HoGD algorithm (whose robustness is proportional to the number of points) and minimal computational efforts, our HAPPS-1 method is able to compete favourably with these other methods such as MR-BF-DSIFT-E which employs well-established techniques (SIFT, BoW, manifold ranking, and multi-view 2D projections). We therefore assume that our HAPPS-1 method would experience performance boost if, for instance: (i) higher number of points sample is used for the HoGD algorithm, (ii) a better local features combination approach, such as the BoW approach is adopted instead, for finalising the local APPFD features, and (iii) the APPFD as it is were to be combined with a more robust global descriptor to form the final hybrid HAPPS method.

Experiment 3: Parameter Settings and Configurations

The experimental results presented in Table 5.10 for our HAPPS-1 retrieval method are derived from the parameters provided in Table 5.11. Here, we only provide results of experimental runs whose parameter combination/settings gave the best overall

Dataset:	SHREC'10 [140]	**Parameter Settings**					
Experiment 3	**Algorithms**	N	r	vs	$b_{(APPFD)}$	$b_{(HoGD)}$	**Dist.Metric**
run-3a	HAPPS-1	4500	0.50	0.20	8	120	Cosine

Table 5.11: Parameter settings for a single experimental run (run-3a) with the HAPPS-1 retrieval method on the SHREC'10 dataset for retrieval of non-rigid 3D objects challenge. The parameters: N, r, vs, $b_{(APPFD)}$, $b_{(HoGD)}$ are explained in Section 5.3.

evaluation performance. Our experimental outcomes followed a series of preliminary testing, where we found that sampling 4,500 points for the SHREC'10 dataset was enough to give good evaluation results. We selected different values for each of the other four parameters (r, vs, $b_{(APPFD)}$,) and $b_{(HoGD)}$ through separate experimental runs as discussed in Section 5.3.3, but the parameter values reported in Table 5.11 gave the best overall performance scores as already discussed in Section 5.3.3 above. We present the other results of our different experimental runs (showing different parameter settings) in Table 5.13, and plot their differences in the PRC plot shown in Figure 5.21 - see Section 5.3.3.

Experiment 3: Performance Analysis of HAPPS-1 Method With Different Distance Metrics (Fixed Parameter Settings) On The SHREC'10 Dataset

Considering that each of the distance/(dis)similarity metrics, described in Section 2.2.3 performs differently on a given 3D shape descriptor, as already explained, in all our experimental runs and evaluations (including this one), we adopt and implement at least five different (dis)similarity metrics (Cosine Distance, Euclidean Distance, SED, KLD and the EMD) for the final shape matching of 3D object descriptors. The goal, ultimately, is to be able to understand how the performance results of each of our retrieval methods (shape descriptors) are being affected by certain sub-processes of the entire shape retrieval pipeline (in this case, the matching sub-process), and to be able to analyse and finally select the results of the matching method with the best overall performance outcome. For this testing/analysis, we found that, for the HAPPS-1 retrieval method and the parameter settings in Table 5.11, both the Cosine Distance and KLD metrics performed better than other metrics for the SHREC'10 dataset and retrieval task. In Table 5.12 and the PRC plot in Figure 5.20, we present these quantitative results with different (dis)similarity metrics. Note that these are for experimenatl run-3a.

Overall, we can see in Table 5.12 that the Cosine Distance metric returns the best performance than all the other four metrics, using the HAPPS-1 retrieval method and the same parameter settings reported in Table 5.11. Following the Cosine metric in terms of performance rating is the KLD metric, then the Euclidean Distance metric. However, these three metrics essentially competes favourably for performance ranking from a global perspective, as there is no much differences between the area under their respective curves. Notice how poor the results with EMD are, which is a strong indication that this metric is not suitable for the HAPPS-1 method, as earlier mentioned in Section 2.2.3, regarding the suitability of certain

Experiment5	Dist.Metric	Algorithms	NN	FT	ST	E	DCG
run-3a	**Cosine**	**HAPPS-1**	**1.0000**	**0.8063**	**0.8837**	**0.6463**	**0.9454**
✓	EMD	HAPPS-1	0.1950	0.2682	0.5155	0.3367	0.5744
✓	Euclidean	HAPPS-1	0.9850	0.7726	0.8474	0.6214	0.9289
✓	KLD	HAPPS-1	0.9850	0.7766	0.8729	0.6365	0.9349
✓	SED	HAPPS-1	0.9850	0.7703	0.8468	0.6206	0.9283

Table 5.12: Quantitative results of five evaluation statistics revealing the retrieval performances of our HAPPS-1 retrieval method with single experimental run (run-3a), using five different distance metrics and the parameter settings in Table 5.11, evaluated for the SHREC'10 dataset, with the Cosine Distance metric having the best overall performance.

distance/(dis)similarity metrics with certain shape descriptors. It is however, interesting to see that the performances of the Euclidean Distance, KLD, and SED metrics are exactly the same with the NN statistics. Tweaking the r and vs parameters is likely to have much influence on these statistics as we would show in results in Table 5.14. Note that the parameter settings which yield these results are respectively presented in Table 5.13 The PRC plot in Figure 5.20 also confirms the results presented in Table 5.12.

Experiment 3: Performance Analysis of Additional Results With Different Parameter Settings/Different Experimental Runs of The HAPPS-1 Method for SHREC'10 Dataset

Considering that the overall outcome of every algorithm/method, including the methods we implemented for our 3D shape retrieval tasks, are affected by their parameter settings (i.e. by choice of parameters value), and recognising in our work that certain parameters (mentioned in Section 5.3.1 and Section 5.3.2, for example) have the potential to influence the outcome of the APPFD and HAPPS retrieval algorithms, we carried out several experiments with different parameter settings for each experimental run, for every shape retrieval challenge and/or dataset. The goal here is to find appropriate parameter settings that would return the best overall performance results for a given dataset and retrieval method. In Table 5.13, we present several other experimental runs (different from the one discussed in Section 5.3.3), each with different parameter settings for the SHREC'10 dataset, using our HAPPS-1 retrieval method. For each experimental run with its unique parameter settings, five different distance/(dis)similarity metrics (see Section 2.2.3) were used for matching the shape descriptors of two 3D objects in the SHREC'10 dataset. However, only the metric that gave best overall results for that experimental run is indicated in the *"Dist.Metric"*7// column. In addition, Table 5.14 shows the quantitative retrieval results of five evaluation metrics/statistics (NN, FT, ST, E, and DCG) for the respective experimental runs outlined in Table 5.13, according to the respective distance metrics indicated, which is a similar approach to that explained in Section 5.3.3. Following these results, the PRC plots in Figure 5.21 is used to further examine how they compare with one another.

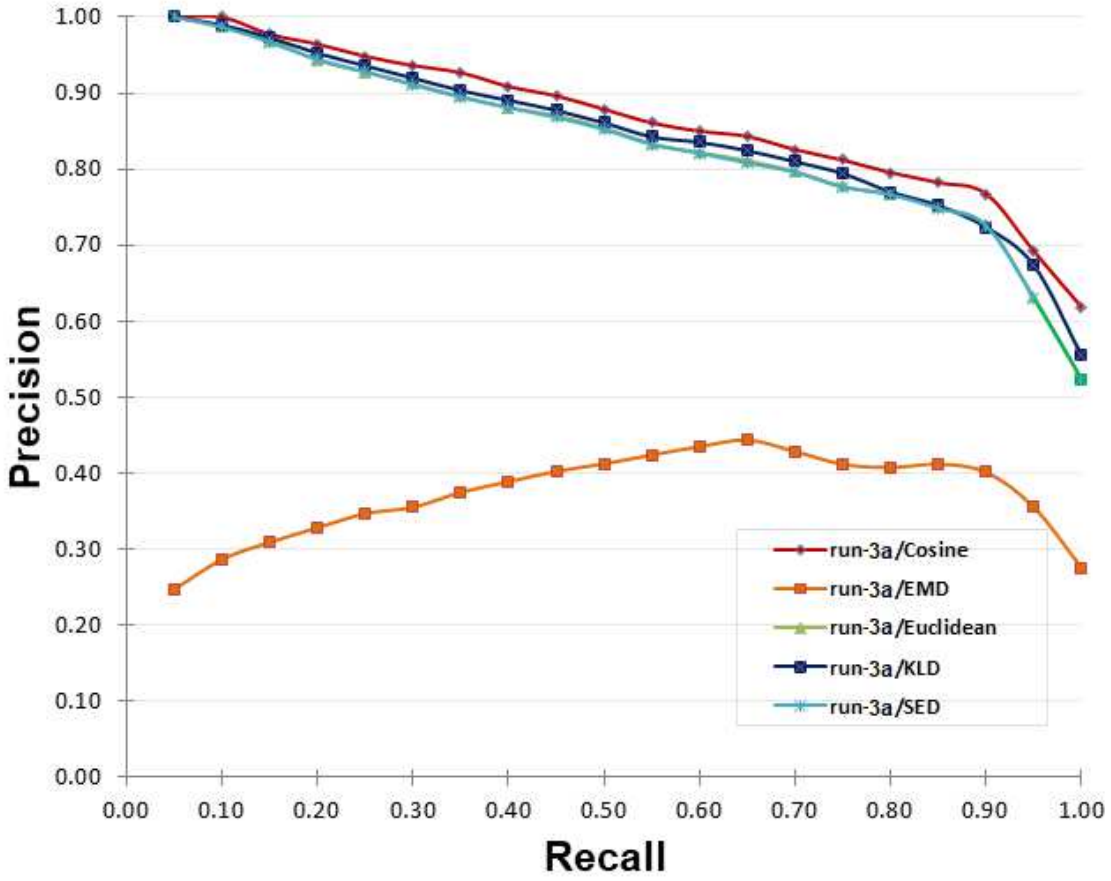


Figure 5.20: PRC plots of the HAPPS-1 experimental run-3a‘, showing the retrieval results of five different distance metrics with the same parameter settings as in Table 5.11, tested on the SHREC'10 dataset. Cosine Distance metric (red) shows the best performance while EMD metric (orange) shows the worst performance.

Essentially, for this performance/experimental analysis, our intention is to explore how the size of the LSP (controlled by the r parameter) during local APPFD computation or the size of key points, (controlled by the voxel-grid down-sampling parameter, vs), including the number of bins (for both the multi-dimensional APPF histogram and the 1-dimensional HoGD histogram) would affect the overall performance of our final HAPPS retrieval method. Among the five different distance/(dis)similarity metrics used to match our HAPPS descriptors, including the HAPPS-1 method, we see from this analysis that the Cosine and KLD distance metrics are the most suitable metrics for this retrieval method. In Table 5.13 and its corresponding PRC plots in Figure 5.21, we show that these two metrics closely compete with each other as the best metric for the different experimental runs that we have presented. As explained in Section 4.2.3, increasing the r and vs parameters increases the LSP size and reduces the number/size of key points, respectively, for the local APPFD computation, and vice-versa. We first examined the effects of tweaking/tuning the r parameter while leaving the other parameters (vs, $b_{(AFFD)}$, $b_{(HoGD)}$ and Dist.Metric) unchanged as shown in Table 5.13, Experiment 3: run-3b,

Dataset:	SHREC'10 [140]	**Parameter Settings**					
Experiment 3	**Algorithms**	N	r	vs	$b_{(APPFD)}$	$b_{(HoGD)}$	**Dist.Metric**
run-3a	**HAPPS-1**	**4500**	**0.50**	**0.20**	**8**	**120**	**Cosine**
run-3b	HAPPS-1	4500	0.40	0.20	8	120	Cosine
run-3c	HAPPS-1	4500	0.40	0.20	7	120	Cosine
run-3d	HAPPS-1	4500	0.30	0.30	8	120	KLD
run-3e	HAPPS-1	4500	0.30	0.20	8	120	KLD
run-3f	HAPPS-1	4500	0.20	0.10	8	187	KLD
run-3f	HAPPS-1	4500	0.20	0.10	8	187	Cosine
run-3g	HAPPS-1	4500	0.20	0.10	8	65	Cosine

Table 5.13: Parameter settings for several different experimental runs with the HAPPS-1 retrieval method on the SHREC'10 dataset for retrieval of non-rigid 3D objects challenge. The parameters: N, r, vs, $b_{(APPFD)}$, $b_{(HoGD)}$ are explained in Section 5.3. For each experimental run with the parameter settings shown, five different distance/(dis)similarity metrics are used for descriptors matching, but here, only the metric that gave best overall result for that experimental run is indicated in the *"Dist.Metric"* column.

where we reduce r by 0.10 (i.e. from $r = 0.50$ to $r = 0.40$). Table 5.13 and its corresponding PRC plots in Figure 5.21 show a performance drop.

Secondly, we understand that the higher the $b_{(APPFD)}$ parameter, the larger the size of the local APPFD signature/*feature-vector* as explained in Section 4.2.1, therefore, in experimental run-3b, we decided to reduce this parameter from 8 to 7 as in experimental run-3c (which drastically reduced the final HAPPS-1 *fv* from 262,264-dimension to 117,769-dimension), and got interesting results where the performance (run-3c) was close to that of run-3b as shown in Table 5.14 and its corresponding PRC plots in Figure 5.21. In terms of overall performance rating, we would up-vote the parameter settings for experimental run-3c instead of that for run-3b regarding the characteristics for an appropriate (i.e. a good) shape descriptor (see Section 2.3.1).

Thirdly, in experimental run-3d and run-3e, we decided to exclusively investigate the effect of only the vs parameter on the overall HAPPS-1 retrieval method, by changing it from $vs = 0.30$ to $vs = 0.20$, which increases the number/side of key points for each 3D object (as previously explained) and leaving all other parameters (r, $b_{(AFFD)}$, $b_{(HoGD)}$ and Dist.Metric) unchanged. Note that by this experiment: run-3e, the LSP size is already kept reduced (i.e. having poorer performance), by reducing the r parameter from $r = 0.50$ to $r = 0.30$. Again, it is interesting to witness a slight performance gain for experimental run-3e from run-3d.

We observe here (Table 5.14 and Figure 5.21) that experimental run-3b and run-3c have close performance rating, while experimental run-3d and run-3e also have close rating, irrespective of the different parameter tuning. However, looking carefully at the analyses for all these four experimental runs: run-3b to run-3f, we can agree that increasing LSP size and number of key points would positively impact the final performance of the HAPPS method, including that better retrieval performances can still be achieved by reducing the $b_{(APPFD)}$ parameter from 8 to

Experiment4	Dist.Metric	Algorithms	NN	FT	ST	E	DCG
run-3a	**Cosine**	**HAPPS-1**	**1.0000**	**0.8063**	**0.8837**	**0.6463**	**0.9454**
run-3b	Cosine	HAPPS-1	0.9950	0.7808	0.8734	0.6388	0.9381
run-3c	Cosine	HAPPS-1	0.9750	0.7750	0.8716	0.6339	0.9359
run-3d	KLD	HAPPS-1	0.9700	0.7221	0.8405	0.6104	0.9128
run-3e	KLD	HAPPS-1	0.9700	0.7358	0.8466	0.6155	0.9182
run-3f	KLD	HAPPS-1	0.9600	0.6745	0.8213	0.5865	0.8995
run-3f	Cosine	HAPPS-1	0.9400	0.6303	0.7808	0.5531	0.8755
run-3g	Cosine	HAPPS-1	0.9400	0.6303	0.7808	0.5531	0.8755

Table 5.14: Quantitative results of five evaluation statistics revealing the retrieval performances of our HAPPS-1 retrieval method with several (8) different experimental runs (run-3a to run-3g), each with different parameter settings. We indicate only results of the distance metric with overall best results for each parameter settings as in Table 5.13. These experiments are for the SHREC'10 dataset. Note that for run-3f, we show two results, each with different distance metric.

7 resulting in a more compact shape descriptor. Nevertheless, computational efficiency is a critical issue that must be considered when selecting these parameters, and setting the $b_{(APPFD)}$ parameter lower than 7 bins would produce poor results as we would see in Experiment 7, Section 5.4.1. Table 5.27 and Table 5.28 confirms.

Finally, the results of experimental run-3f and run-3g reveal something remarkably interesting about the influence of the $b_{(HoGD)}$ bin parameter and choice of distance metric used to match the final HAPPS-1 descriptor. Here, keeping other parameters (r, vs, $b_{(AFFD)}$, and Dist.Metric) the same, we decided to increase and decrease the $b_{(HoGD)}$ parameter from 120 to 187 and 65 bins, respectively, to examine its influence/contribution to our final HAPPS-1 retrieval method. We see from Table 5.14 and Figure 5.21 that the results of both experimental run-3f and run-3g remain exactly unaffected by the $b_{(HoGD)}$ parameter tuning using the Cosine distance metric for matching. But with the KLD distance metric, a noticeable improvement is recorded for the higher number of $b_{(HoGD)}$ bin of 187 (see Yellow PRC curve for experimental run-3f/KLD). This further confirms that, depending on the parameter settings, both the Cosine and KLD distance metrics would most definitely be suitable for the HAPPS-1 retrieval method. Overall, experimental run-3f and run-3g produced poorer retrieval performances, which is expected, due to the smaller LSP size caused by reducing the r parameter from $r = 0.50$ to $r = 0.20$ and increasing the number of key points via vs parameter decrease from $vs = 0.20$ to $vs = 0.10$.

5.3.4 Experiment 4: Evaluating The Retrieval Performances of The HAPPS-1 Method On SHREC'11 Dataset

In this section, we present the retrieval performances of our HAPPS-1 retrieval method on the SHREC'11 dataset. This dataset contains 600 water-tight 3D triangular meshes grouped into 30 classes/categories, with each category containing 20 3D models (see Section 5.2.3). In order to evaluate how best our method performs on this dataset, we also compare our results side-by-side with the results of

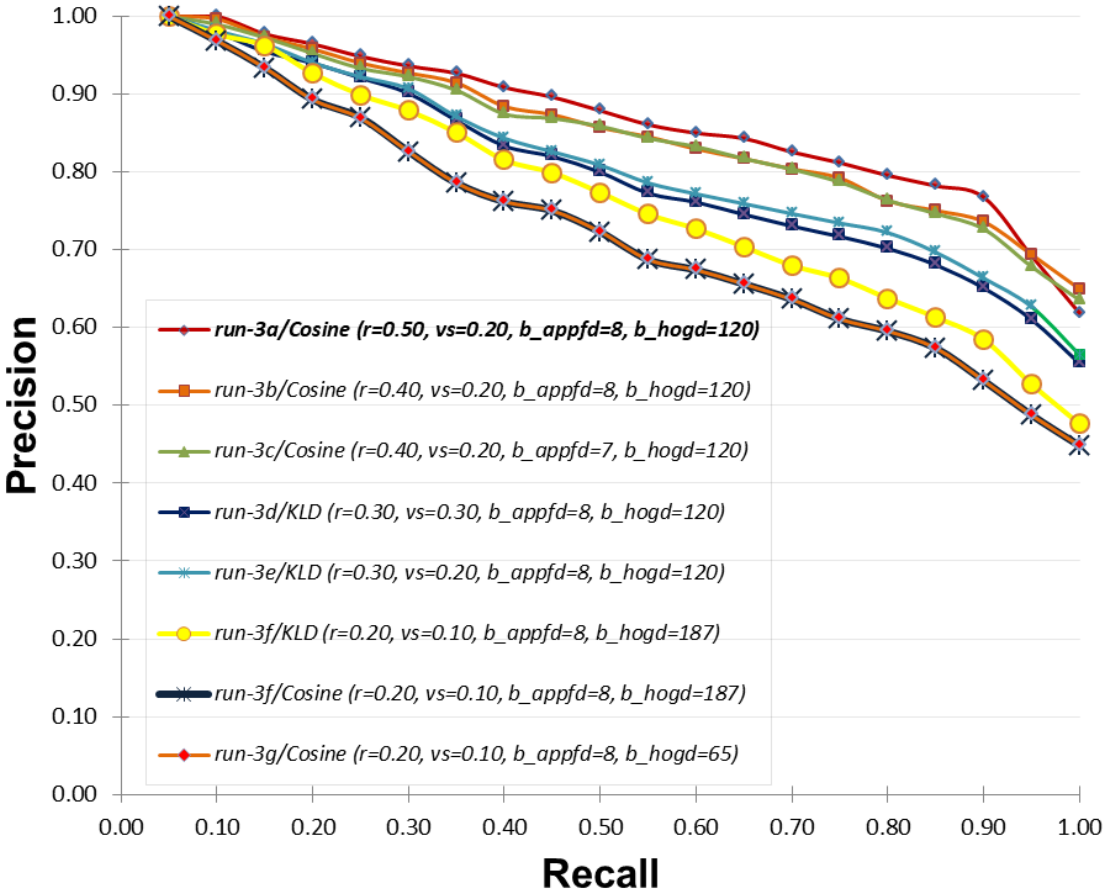
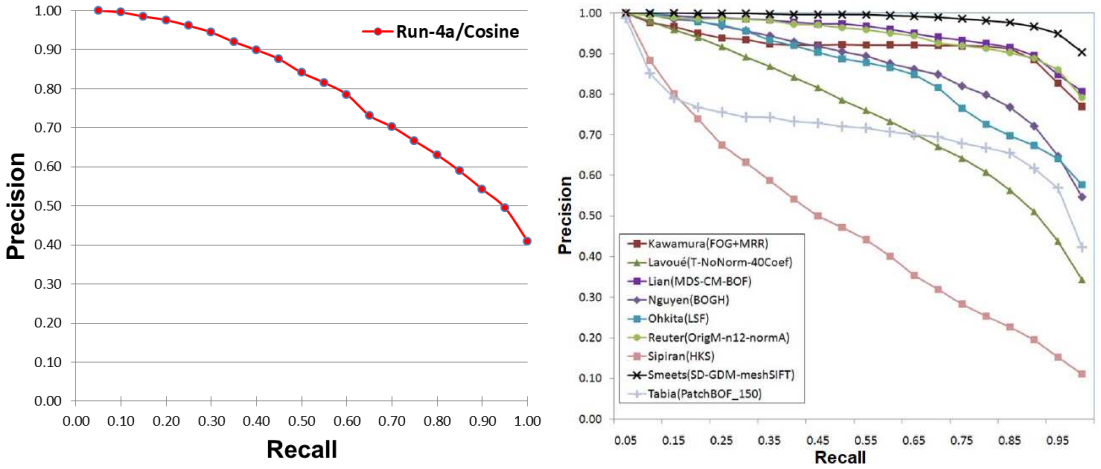


Figure 5.21: PRC plots of the HAPPS-1 on the SHREC'10 dataset, showing the retrieval results of eight different experimental runs, each with a different parameter settings as in Table 5.13. For each plot, the distance metric and parameter values involved are shown. Note that for run-3f, we show two plots, each with different distance metric.

25 other state-of-the-art retrieval methods, from nine participants, who competed for the SHREC'11 - Shape Retrieval Contest of Non-rigid 3D Watertight Meshes challenge. The other nine retrieval methods and their respective experimental runs, making a total of 25 runs are listed as follows: **(i)** Features on Geodesics (FoG, FOG+MR, and FOG+MRR), **(ii)** Bag of Words with Local Spectral Descriptors (T-NoNorm-40Coef, T-r01-40Coef, T-r01-50Coef, T-r015-40Coef and T-r015-50Coef), **(iii)** Visual Similarity based Non-rigid 3D Shape Retrieval Using MDS (MDS-CM-BOF), (iv) Bag of Geodesic Histograms (BOGH), **(v)** Localized Statistical Features (LSF and MLSF), **(vi)** ShapeDNA: Laplace Spectra for Non-Rigid Shape Analysis (ShapeDNA: OrigM-n10-norm1, OrigM-n12-norm1, OrigM-n12-normA, OrigM-n15-norm1 and ReM-n12-norm1), **(vii)** Harris 3D Geodesic Map and HKS based Point-to-point Matching (Harris3DGeoMap16, Harris3DGeoMap32 and HKS), **(viii)** Fusion of SD-GDM and meshSIFT (MeshSIFT, SD-GDM and SD-GDM-meshSIFT), **(ix)** Bag-of Densely-Sampled Local Visual Features (PatchBOF_100 and PatchBOF_150). More details regarding these methods are found in [139]. First, using our shape retrieval method, HAPPS-1, we evaluate the (dis)similarity between every two objects in this dataset and output the (dis)similarity matrix as described in the

last paragraph of Section 5.1. Then for evaluation method, we employ the following six standard evaluation measures: NN, FT, ST, E, DCG, and PRC plots (described in Section 4.3.3), in line with the evaluation in [139].



(a) PRC Plot: HAPPS-1, Run-4a, Cosine (b) PRC Plots: SHREC'11 Paper [139]

Figure 5.22: PRC plots of the retrieval results presented in Table 5.15 for the SHREC'11 retrieval challenge/dataset. Figure 5.22a: (Left) is the PRC plot of our HAPPS-1, run-4a, Cosine method, Figure 5.22b: (Right) is the PRC plots of nine of the best experimental runs of each retrieval of the retrieval methods in [139], evaluated for the whole database, while Figure 5.23 presents the PRC plots of all 10 different methods/experimental runs, which is a combination of HAPPS-1 method and those from [139]. Note, we superimpose Figure 5.22a on Figure 5.22b to have Figure 5.23.

The literature for the SHREC'11 retrieval challenge [139] compares the results of 25 experimental runs submitted by a total of nine groups/authors, where each group submitted at least one retrieval method, with some having different parameter settings, thus, different experimental runs. Given 25 (dis)similarity matrices, one for each experimental run, the work in [139] carried out evaluations for the submitted retrieval methods not only on the average performance of the whole database, but also on the result corresponding to each specific class. However, a total of nine unique state-of-the-art retrieval methods (3D shape descriptors) were involved for the SHREC'11 retrieval challenge. Each retrieval method had more than 1 experimental run, and for each method, the experimental run having the highest performance results (i.e. retrieval accuracies) in all five quantitative statistics (NN, FT, ST, E, and DCG) are provided, including their respective PRC plots. Similarly, we are interested in comparing and analysing the retrieval accuracies of our HAPPS-1 method with only the best performing experimental run for each of the nine retrieval methods submitted for the SHREC'11 retrieval challenge, and not on the entire 25 experimental runs. In addition, our evaluations are carried out only on the average performance of the whole SHREC'11 dataset, and not on the results corresponding to each specific class/category, unlike the evaluation steps followed by [139]. For further analysis of all the 25 runs and specific class evaluation, we refer the reader to [139].

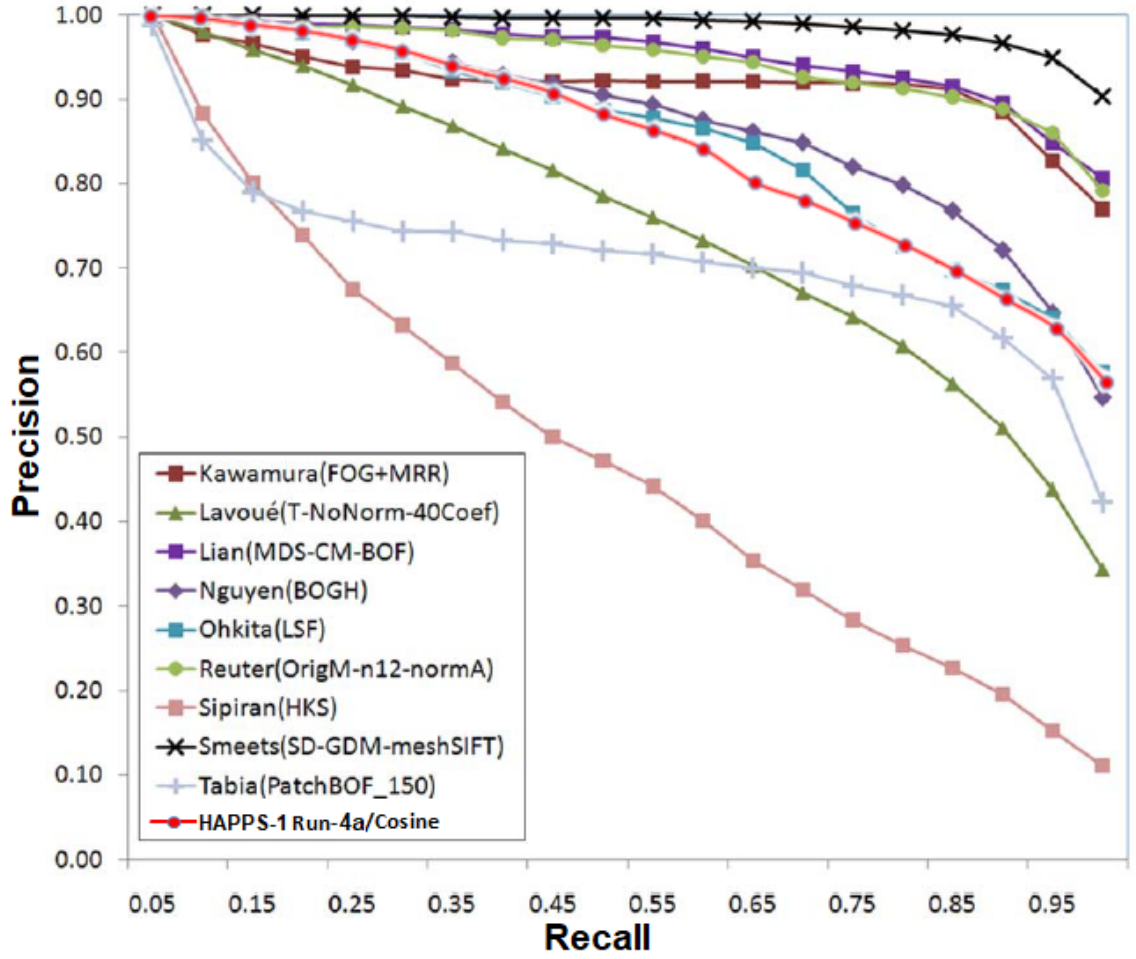


Figure 5.23: PRC Plots: of all 10 different methods/experimental runs, which is a combination of HAPPS-1, run-4a method and those from [139], which is a combination of Figure 5.22a with Figure 5.22b.

Authors [139]	Method	NN	FT	ST	E	DCG
Kawamura	FOG+MRR	0.960	0.881	0.946	0.696	0.959
Lavoue	T-NoNorm-40Coef	0.955	0.672	0.803	0.579	0.897
Lian	MDS-CM-BOF	0.995	0.913	0.969	0.717	0.982
Nguyen	BOGH	0.993	0.811	0.884	0.647	0.949
Ohkita	MLSF	0.987	0.809	0.879	0.643	0.948
Reuter	OrigM-n12-normA	0.992	0.915	0.957	0.705	0.978
Sipiran	HKS	0.837	0.406	0.497	0.353	0.730
Smeets	**SD-GDM-meshSIFT**	**1.000**	**0.972**	**0.990**	**0.736**	**0.996**
Tabia	PatchBOF_150	0.748	0.642	0.833	0.588	0.837
This Thesis	HAPPS-1 run-4a	0.993	0.720	0.801	0.582	0.920

Table 5.15: Quantitative evaluation measures of best nine of twenty-five state-of-the-art retrieval methods, including our proposed HAPPS-1 retrieval method using five standard performance measures on the whole of SHREC'11 benchmark dataset.

Experiment 4: Parameter Settings and Configurations

Table 5.16 presents the parameter settings of the best overall experimental run (run-4a) using the HAPPS-1 retrieval method, regarding the experimental evaluation of the SHREC'11 dataset. These parameter settings are responsive for the HAPPS-1,run-4a quantitative results in Table 5.15. Our experimental approach here is very similar to the approach and parameter settings used with the SHREC'10 dataset (see Section 5.3.3). For all the experimental runs in this performance evaluation, a total of 4500 random points are sampled from each 3D surface to compute our HAPPS-1 descriptor.

Dataset:	SHREC'11 [139]	**Parameter Settings**					
Experiment 4	**Algorithms**	N	r	vs	$b_{(APPFD)}$	$b_{(HoGD)}$	**Dist.Metric**
run-4a	HAPPS-1	4500	0.20	0.10	8	65	Cosine

Table 5.16: Parameter settings for an experimental run with the HAPPS-1 retrieval method on the SHREC'11 dataset. The parameters: N, r, vs, $b_{(APPFD)}$, and $b_{(HoGD)}$ are explained in Section 5.3.

Experiment 4: Results and Discussion

Table 5.15 lists the retrieval accuracies of 10 shape retrieval methods for 3D objects: our HAPPS-1 method and nine other methods (i.e. overall best performing method/experimental run from each of the nine groups submitted for the SHREC'11 retrieval challenge) evaluated on the whole SHREC'11 dataset. We also present a corresponding PRC plots in Figure 5.23 for the results presented in Table 5.15. The quantitative results show the SD-GDM-meshSIFT method by Smeets to have the highest overall retrieval accuracy in all five quantitative performance measures. The second-best performing method is the MDS-CM-BOF method by Lian, although the OrigM-n12-normA method by Reuter performed slightly better than MDS-CM-BOF in terms of the FT measure. Interesting to see that closely following the best overall performing method (SD-GDM-meshSIFT) of 100%, only four other methods: our HAPPS-1, OrigM-n12-normA, BOGH and MDS-CM-BOF methods recorded over 99% score for the NN quantitative measure. In addition, seven (out of ten) methods, including our HAPPS-1 method shows performance score of over 90% for the E-measure, which is another important metric for performance evaluation.

Generally, for this 3D shape retrieval task and the retrieval methods presented in Table 5.15, we can clearly see from the PRC plots in Figure 5.23 that **(i)** these three methods (SD-GDM-meshSIFT, MDS-CM-BOF, and OrigM-n12-normA) performed exceptionally well, considering their respective AUC, **(ii)** these four methods (HAPPS-1, BOGH, LSF, and FOG+MRR) compete in the middle-class performance ranking, while **(iii)** these three methods (T-NoNorm-40Coef, PatchBOF_150, and HKS) struggles in the lower-class ranking, with the HKS method by Sipiran having the worst overall performance or retrieval accuracies. Although, the HAPPS-1 methods completely outperforms the PatchBOF_150 method by Tabia, HKS method

by Sipiran, and T-NoNorm-40Coef method by Lovoue, in most quantitative performance measures such as the NN, FT, and DCG, we are interested to know why this is the case with the PatchBOF_150 for example, which uses the BoW technique found to be very efficient for local feature combinations. According to [139], the PatchBOF_150 uses local geodesic distance as its feature, which might either not be robust enough to capture the properties of non-rigid 3D objects or not suitable for the BoW algorithm. Another factor might be the choice of vocabulary size for training the BoW vocabulary, while not disregarding the L_2 distance metric they used for their matching, which might not be suitable for this descriptor. Secondly, the HKS approach by Sipiran performed poorly and we find that this method was used for key points detection, which might not be as effective compared to the approach we adopted instead (see Section 3.3.5). In addition, they returned descriptors having 100-dimension, which might not be able to adequately capture local features for each 3D object.

Interestingly, the SD-GDM-meshSIFT extracted salient local features and match them directly to compare 3D objects and is insensitive against various isometric transformations mainly due to the use of 3D Canonical Forms. From these results, it is obvious that the combination of several different kinds of retrieval methods, such as the SD-GDM-meshSIFT and our HAPPS-1, can result in better retrieval accuracies. We are also interested in knowing why the fusion of SD-GDM and meshSIFT performed so well, therefore we observe that this approach combines a global feature method (SD-GDM) with a local features method (meshSIFT) for non-rigid 3D shape retrieval. The SD-GDM approach presents shape descriptors that are represented by a geodesic distance matrix (GDM), which is isometric deformation invariant matrix, having the normalised geodesic distance between each pair of about 2500 sampled points on the surface of each 3D object in the dataset. The extra computational step: spectral decomposition (SD) of the GDM provides a sampling order invariant global feature (shape descriptor, SD-GDM), which makes the GDM more robust. On a final note, the SIFT descriptor is generally known to be very robust for 2D/3D shapes and the combination of SD-GDM with meshSIFT provide an even more robust hybrid 3D shape descriptor: SD-GDM-meshSIFT, which is likened to averaging the retrieval results over two separate descriptors, since their respective DM were fused together in this approach. However, 15 (out of 25) of the methods/experimental runs mentioned in Section 5.3.4 for the SHREC'11 retrieval challenge adopts the BoW approach to quantize a 3D object's local features into a word histogram. This is because of the efficacy of this approach for local feature aggregation and confirmed from analysis of previous retrieval methods (see Section 5.3.3).

Experiment 4: Performance Analysis of Additional Results With Different Parameter Settings/Different Experimental Runs of The HAPPS-1 Method for SHREC'11 Dataset

In addition to experimental run-4a (overall best HAPPS-1 experimental run for the SHREC'11 dataset) with its given parameter settings, where we compare its quantitative results with the results of several other state-of-the-art methods for this retrieval challenge (see Section 5.3.4), we also perform several other experiments with the HAPPS-1 method, with each experimental run having different parameter set-

tings, as outlined in Table 5.17. Similar to every other retrieval experiment presented in this thesis, we first compute the (dis)similarity of pairs of 3D descriptors representing objects in the SHREC'11 dataset using five different distance/(dis)similarity metrics, as previously mentioned. Unlike the performance analysis of the retrieval results for Experiment 3 (see Section 5.3.3), where we presented the results of the distance metric with the best quantitative scores for each experimental run, in this section, and for the SHREC'11 retrieval task, we instead analyse how the different parameter settings affects the retrieval accuracies/performance of the HAPPS-1 retrieval method on the SHREC'11 dataset, using a single distance metric: the Cosine metric. This is because we are interested to see how different parameter settings affect the final HAPPS-1 retrieval method under a fixed circumstance. Interestingly, we found that for each of these different experimental runs with the HAPPS-1 method, the retrieval results obtained using the Cosine distance metric for descriptors matching outperforms all the other results obtained with other distance metrics.

Dataset:	SHREC'11 [139]	**Parameter Settings**					
Experiment 4	**Algorithms**	N	r	vs	$b_{(APPFD)}$	$b_{(HoGD)}$	**Dist.Metric**
run-4a	**HAPPS-1**	**4500**	**0.20**	**0.10**	**8**	**65**	**Cosine**
run-4b	HAPPS-1	4500	0.20	0.10	7	65	Cosine
run-4c	HAPPS-1	4500	0.25	0.10	7	65	Cosine
run-4d	HAPPS-1	4500	0.25	0.15	7	65	Cosine
run-4e	HAPPS-1	4500	0.16	0.10	7	65	Cosine
run-4f	HAPPS-1	4500	0.18	0.10	7	65	Cosine
run-4g	HAPPS-1	4500	0.27	0.17	7	65	Cosine

Table 5.17: Parameter settings for several different experimental runs with the HAPPS-1 retrieval method on the SHREC'11 dataset. The parameters: N, r, vs, $b_{(APPFD)}$, $b_{(HoGD)}$ are explained in Section 5.3. For all experimental runs with the different parameter settings shown, only results with the Cosine distance/(dis)similarity metric are provided, instead.

For each of the experimental run/parameter settings listed in Table 5.17, we present its respective quantitative retrieval performances/results in Table 5.18, and plot these outcome in the PRC shown in Figure 5.24. From these results, especially, looking at the PRC plots, we can see that experimental run-4a takes the lead, followed very closely by run-4e, where $7^6(+65) = 117,714$ instead of $8^6(+65) = 262,209$-dimensional final fv was used. Although it is difficult to analyse individual differences of retrieval accuracies resulting from the different parameter settings, using the PRC plots, the general observation/conclusion in this is that the range of parameter values chosen for $N = 4,500$ point samples from the surface of every 3D object in the SHREC-11 dataset were in order. Interestingly, through these experimental runs, we discover that it is possible to capture local properties of the 3D surface and obtain higher retrieval accuracies with considerably low dimensional fv, by using $b_{(APPFD)} = 7$ instead of $b_{(APPFD)} = 8$ (compare run-4f to run4a, in Figure 5.24). Finally, we see in this analysis that slightly altering the r and vs

parameters did not have much impacts on the overall performance accuracies of our HAPPS-1 retrieval method on the SHREC'11 dataset. Unfortunately, due to time and in trying to avoid repetition, we did not analyse the effect of $b_{(HoGD)}$ parameter for these experiments. However, we refer the reader to Section 5.3.3 for analysis on the effects of the $b_{(HoGD)}$ and HoGD algorithm on the HAPPS-1 retrieval method.

Experiment4	Dist.Metric	Algorithms	NN	FT	ST	E	DCG
run-4a	**Cosine**	**HAPPS-1**	**0.9933**	**0.7204**	**0.8005**	**0.5820**	**0.9201**
run-4b	Cosine	HAPPS-1	0.9917	0.7145	0.7978	0.5822	0.9183
run-4c	Cosine	HAPPS-1	0.9917	0.7004	0.7791	0.5687	0.9114
run-4d	Cosine	HAPPS-1	0.9933	0.7010	0.7797	0.5686	0.9113
run-4e	Cosine	HAPPS-1	0.9950	0.6984	0.7942	0.5772	0.9158
run-4f	Cosine	HAPPS-1	0.9917	0.7098	0.7981	0.5818	0.9186
run-4g	Cosine	HAPPS-1	0.9850	0.6953	0.7749	0.5649	0.9080

Table 5.18: Quantitative results of five evaluation statistics revealing the retrieval performances of our HAPPS-1 retrieval method on the SHREC'11 dataset, with several (7) different experimental runs (run-4a to run-4g), where each has a different parameter setting. For each of these parameter settings/experimental runs, we find that results of matching descriptors with the Cosine distance metric returned the highest accuracies.

5.3.5 Experiment 5: Evaluating The Retrieval Performances of The HAPPS-1 Method On SHREC'19 Protein Dataset

Motivated by the success of the HAPPS-1 retrieval method on the SHREC'18 Protein shape retrieval dataset, which produced the best overall retrieval accuracies, outperforming several other state-of-the-art methods for the SHREC'18 Protein shape retrieval challenge [175], we decided to also evaluate the retrieval performances of the HAPPS-1 retrieval method on a similar dataset with different configuration and classification arrangements: the SHREC'19 Protein dataset. As indicated in Section 5.2.8, two levels of classification: *Protein* and *Species* are adopted for the SHREC'19 Protein dataset (see Table 5.2). The 3D shapes from ortholog proteins, i.e. *Protein* level, are expected to have a high level of similarity in their overall shape because they share similar activities and functions in organisms that co-evolved from the same ancestor. The implication is that the discrimination between shapes at the *Species* level is expected to be more difficult compared to those at the *Protein* level. In this section, the results of different experimental runs for our HAPPS-1 retrieval method (for both *Proteins* and *Species* classification levels) are presented and compared, side-by-side, with the results of several other state of the art retrieval methods which competed for the SHREC'19 Protein shape retrieval challenge.

Experiment 5: Parameter Settings and Configurations

Similar to the results presented in previous sections, Table 5.19 indicates the parameter settings containing different values for different experimental runs of the

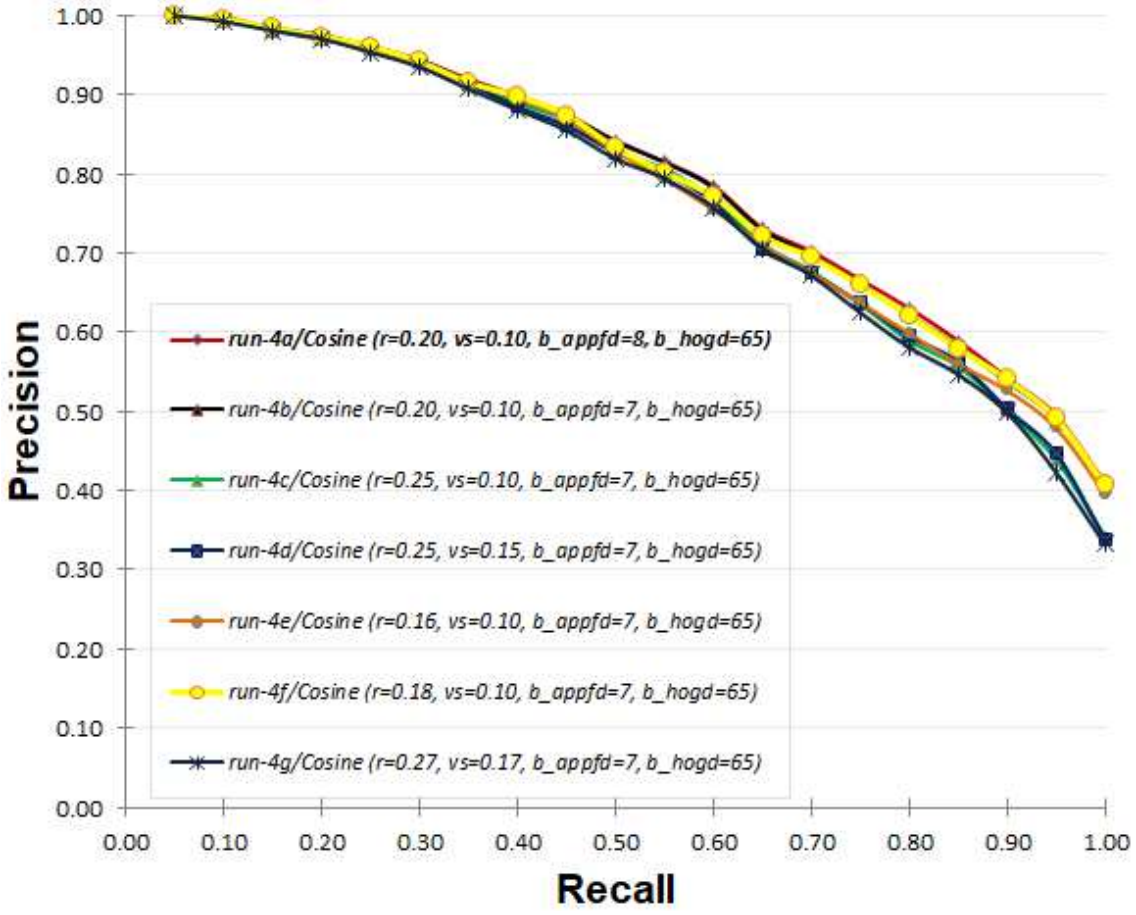


Figure 5.24: PRC plots of the HAPPS-1 on the SHREC'11 dataset, showing the retrieval results in Table 5.18, of seven different experimental runs, each with a different parameter settings listed in Table 5.17. For each plot, the distance metric and parameter values involved are shown.

HAPPS-1 retrieval method on the SHREC'19 protein dataset. However, unlike with the other benchmark datasets, including the SHREC'18 protein dataset where we sampled 4,500 points for our shape retrieval algorithm, for each 3D object in the SHREC'19 protein dataset (with approximately 54,000 to 270,000 points/vertices), we only sampled 3,500 points instead, in order to effectively deal with storage complexity and we still record high performance accuracies. Essentially, the larger the number of points sample, multiplied by the total number of 3D objects (5,298 in this case), the larger the storage requirement and the more computational resources used up, and vice-versa. Therefore, one of the objectives for us in this task was to test the accuracy of our retrieval method while maintaining high level of efficiency, in consideration of the characteristics discussed in Section 2.3.1.

In line with the dual classifications of the SHREC'19 dataset, for experimental runs (i.e. run-5a and run-5b) with the parameter settings shown in Table 5.19, two sets of evaluation results are shown: (i) results of evaluation using the *Protein* classification level, and (ii) results of evaluation using the *Species* classification level.

Dataset:	SHREC'19 [123]	**Parameter Settings**					
Experiment 5	**Algorithms**	N	r	vs	$b_{(APPFD)}$	$b_{(HoGD)}$	**Dist.Metric**
run-5a	**HAPPS-1**	**3500**	**0.50**	**0.20**	**7**	**65**	–
run-5b	HAPPS-1	3500	0.50	0.30	6x15	65	–

Table 5.19: Parameter settings for the first experimental run with the HAPPS-1 retrieval method on the SHREC'19 dataset. The parameters: N, r, vs, $b_{(APPFD)}$, and $b_{(HoGD)}$ are explained in Section 5.3.

First, we present the retrieval accuracies (results) of our HAPPS-1 method on this dataset, showing the effect of four different distance metrics (Cosine, Euclidean, EMD, and SED), using the parameter settings in experimental run-5a and only for the *Protein* level classification. These quantitative performance results are presented in Table 5.20 and their corresponding PRC plots are shown in Figure 5.25. This test was done to enable us ascertain which distance metric is most suitable for the different parameter settings we adopted for this experiment.

Experiment5	**Dist.Metric**	**Algorithms**	**NN**	**FT**	**ST**	**E**	**DCG**
run-5a	**Cosine**	**HAPPS-1**	**0.9853**	**0.5545**	**0.6976**	**0.1287**	**0.9075**
run-5a	Euclidean	HAPPS-1	0.9851	0.5453	0.6906	0.1272	0.9045
run-5a	EMD	HAPPS-1	0.6321	0.3968	0.5808	0.0703	0.8259
run-5a	SED	HAPPS-1	0.7282	0.5209	0.6602	0.1054	0.8780

Table 5.20: Quantitative results of five standard evaluation metrics revealing the retrieval performances of our HAPPS-1 retrieval method on the SHREC'19 protein dataset, with experimental run-5a. Here, performance accuracies are presented for four different distance metrics, using the same parameter settings. Note, these results are obtained using the *Protein* classification level.

Secondly, for the two parameter settings in Table 5.19, we used the *Species* classification level to compare their results using the two experimental runs (run-5a to run-5b), in order to further investigate how the different parameter settings affects the overall retrieval accuracies of the HAPPS-1 retrieval method. We show these results in Table 5.21 and their corresponding PRC plots in Figure 5.27. Finally, we adopt the best overall results for each of the two classification levels and compare with the results of several other state-of-the-art retrieval methods who competed for the SHREC'19 protein shape retrieval challenge. This is to enable us ascertain the global impact of our retrieval method in an unbiased way. In Table 5.22, the best overall retrieval results of our HAPPS-1 method is compared with those in [123], considering the *Protein* classification level. Similarly, in Table 5.23, the best overall retrieval results of our HAPPS-1 method is compared with those in [123], considering the *Species* classification level.

Experiment 5: Results and Discussion

First, looking at the performance accuracies of the HAPPS-1 method using four different distance metrics (Cosine, Euclidean, EMD, and SED) on the SHREC'19

dataset, as presented in Table 5.20 and their corresponding PRC plots in Figure 5.25, we can tell that both the Cosine and Euclidean metrics returned very close results with the Cosine metric quantitatively taking the lead, and followed by the SED, while the EMD, again performed poorly. Secondly, comparing the quantitative performance results of two different experimental runs: run-5a and run-5b, each employing different parameter settings, as presented in Table 5.21 using the *Species* classification level, we can still see that the Cosine metric produces the best overall results for this particular evaluation. Notice how we used a 1-dimensional histogram for the APPFD method with 15 bins in each feature-dimension, resulting to a concatenation of $6x15 = 90$ and finally joining this with the 65 bins of the HoGD descriptor to have a final hybrid HAPPS-1 signature of 155-dimensional *fv*. With this variant of the HAPPS-1 method, with extremely low-dimensional *fv*, we could still record impressive retrieval accuracies. This reveals some more interesting aspect of the APPFD algorithm. Further work is required to further investigate more efficient ways of representing the APPFD.

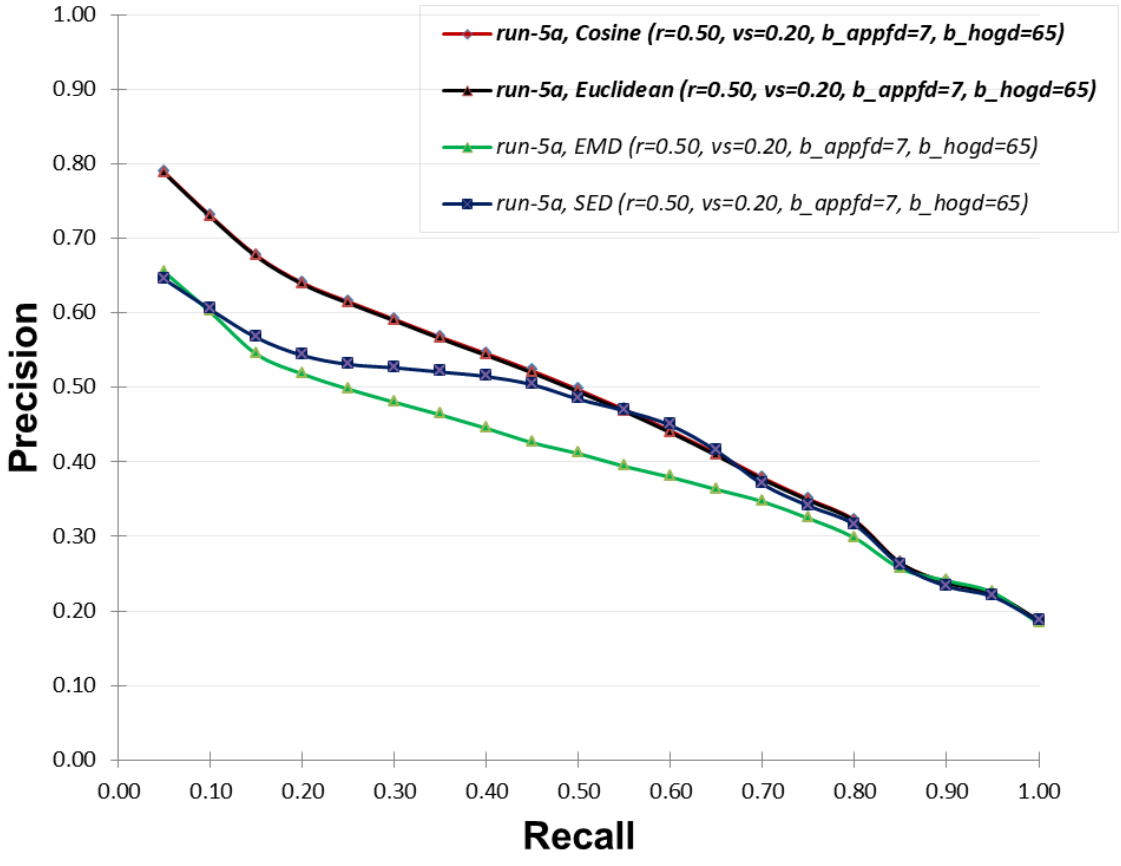


Figure 5.25: PRC plots of the HAPPS-1 on the SHREC'19 protein dataset, depicting the retrieval results of Table 5.20 for a single experimental run: run-5a, with four different distance metrics. Note, the results obtained for these plots were obtained using the *Protein* classification level.

```
...Computing Shape-Descriptor for PointCloud 5298 of 5298

OUTPUT:        Downsampled Cloud Size:   324
Processing Time for PointCloud 5298:     25.350035429000854secs.


Traceback (most recent call last):
  File "C:\ekpoPhD2018\shrec19protein_experiments\officePC\shrec19ProteinRetrieval.py", line 303, in <module>
    allDb_Descriptors = joblib.load(path_to_load_descr)
  File "C:\Python36\lib\site-packages\sklearn\externals\joblib\numpy_pickle.py", line 598, in load
    obj = _unpickle(fobj, filename, mmap_mode)
  File "C:\Python36\lib\site-packages\sklearn\externals\joblib\numpy_pickle.py", line 526, in _unpickle
    obj = unpickler.load()
  File "C:\Python36\lib\pickle.py", line 1050, in load
    dispatch[key[0]](self)
  File "C:\Python36\lib\site-packages\sklearn\externals\joblib\numpy_pickle.py", line 352, in load_build
    self.stack.append(array_wrapper.read(self))
  File "C:\Python36\lib\site-packages\sklearn\externals\joblib\numpy_pickle.py", line 195, in read
    array = self.read_array(unpickler)
  File "C:\Python36\lib\site-packages\sklearn\externals\joblib\numpy_pickle.py", line 141, in read_array
    array = unpickler.np.empty(count, dtype=self.dtype)
MemoryError
>>>
```

Figure 5.26: Screen-shot showing memory error during serialization of all computed shape descriptor (HAPPS-1) for SHREC'19 protein dataset with a very large number of 3D objects: 5,298.

Experiment5	Dist.Metric	Algorithms	NN	FT	ST	E	DCG
run-5a	**Cosine**	**HAPPS-1**	**0.9405**	**0.5071**	**0.6454**	**0.2241**	**0.8561**
run-5b	Cosine	HAPPS-1	0.9268	0.4783	0.6482	0.1073	0.8761

Table 5.21: Quantitative results of five standard evaluation metrics revealing the retrieval performances of our HAPPS-1 retrieval method on the SHREC'19 protein dataset, with two experimental runs: run-5a and run-5b. Both results are from the Cosine distance metric, being the highest performing metric for the respective parameter settings. Note, these results are obtained using the *Species* classification level.

Experiment 5: Comparing Best Results of The HAPPS-1 Method With Other State-Of-The-Art Methods for SHREC'19 Protein Dataset Using Protein and Species Levels

In Table 5.22, we present the best overall result of our experimental evaluation (run-5a in Table 5.20) with the results of only the best experimental runs from each of the state-of-the-art retrieval methods in [123], using the *Protein* classification level. This comparison shows the 3DZD3 method having the best overall performances in all five quantitative performance metrics, followed closely by the HAPT2 and the HAPPS-1 methods, outperforming two other state-of-the-art methods: GASD and ConvLDSNet2. Alternatively, in Table 5.23 we provide similar results comparisons of the HAPPS-1 retrieval method with four other state-of-the-art methods, using the *Species* classification level, instead. Basically, in this thesis, we are only concerned about how our method compares with other well-established retrieval method on this dataset/retrieval challenge. For further in-depth analysis of the results of the other state-of-the-art methods for the SHREC'19 retrieval challenge, we refer the reader to [123]. In these comparisons, we can see that for both the *Protein* and *Species* classification levels, the HAPPS-1 retrieval method continues to rank very

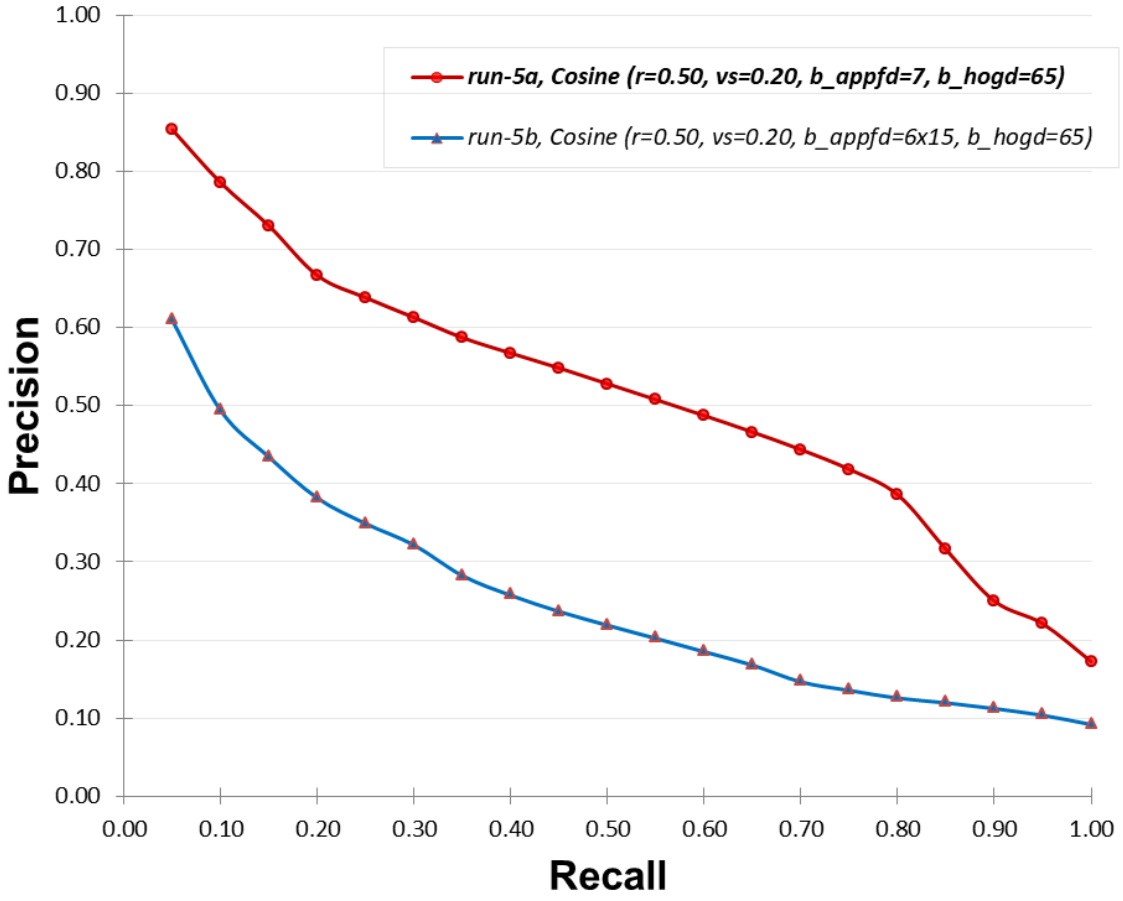


Figure 5.27: The PRC plots of the HAPPS-1 method for the quantitative results in Table 5.21 and experimental run-5a and run-5b, using the SHREC'19 protein dataset. Here, we see that the parameter settings for experimental run-5a returns better retrieval accuracies than the other: run-5b.

high and very close to the highest performing methods, while also outperforming some other state-of-the-art methods.

In conclusion, an important observation in this experimental evaluation is that the value of the $b_{(APPFD)} = 7$ bin parameter chosen in order to efficiently compute the HAPPS-1 *fv* for the SHREC'19 dataset, which is exceptionally large: 5,298 3D objects. For a non-GPU computation, using $b_{(APPFD)} = 8 + b_{(HoGD)} = 65$ would produce a $8^6 = 262,144 + 65 = 262,209$-dimensional *fv*. During implementation, memory issues were experienced while trying to serialise this extremely high dimensional descriptor for a total of 5,298 3D objects (see Figure 5.26). As previously explained, 7 bins yielded a final HAPPS-1 descriptor of 117,649-dimensional *fv*, which was possible to serialize and evaluate. In addition, considering the exceptionally large number of 3D objects in this dataset and in order to avoid memory issues, we rather sampled only 3,500 points from each 3D surface, instead of 4,500 which (from earlier experimental evaluations) has been found to yield optimal results.

Although the final retrieval accuracies with these configurations (see Table 5.19)

Author	Method	NN	FT	ST	E	DCG
Xusi Han [123]	**3DZD2**	**0.995**	**0.662**	**0.794**	**0.131**	**0.935**
Stelios M. [123]	ConvLDSNet2	0.806	0.267	0.354	0.248	0.710
Halim B. [123]	GASD	0.982	0.365	0.500	0.113	0.845
Andrea Gichetti [123]	HAPT2	0.993	0.631	0.741	0.126	0.920
This Thesis	HAPPS-1 run-5a	0.985	0.555	0.698	0.129	0.908

Table 5.22: Performance comparison of our HAPPS-1 method results with the results of four other state-of-the-art retrieval methods [123] submitted for the SHREC'19 protein shape retrieval challenge. All results are macro-average values computed over each *Proteins* classification level.

Author	Method	NN	FT	ST	E	DCG
Xusi Han [123]	**3DZD2**	**0.965**	**0.581**	**0.694**	**0.250**	**0.884**
Stelios M. [123]	ConvLDSNet2	0.604	0.308	0.427	0.335	0.646
Halim B. [123]	GASD	0.945	0.377	0.461	0.213	0.800
Andrea Gichetti [123]	HAPT2	0.954	0.556	0.695	0.241	0.877
This Thesis	HAPPS-1 run-5a	0.940	0.507	0.645	0.224	0.856

Table 5.23: Performance comparison of our HAPPS-1 method results with the results of four other state-of-the-art retrieval methods [123] submitted for the SHREC'19 protein shape retrieval challenge. All results are macro-average values computed over each *Species* classification level.

competes very closely with several other state-of-the-art retrieval methods for this dataset and retrieval challenge (see Table 5.22 for *Protein* level classification and Table 5.23 for *Species* level classification), we conclude that given a larger number of points sample and increasing the value of the $b_{(APPFD)}$ bin parameter from 7 to 8, in addition to fine-tuning other related parameters, our HAPPS-1 retrieval method is capable returning best overall retrieval accuracies that outperforms even all other state-of-the-art methods for this particular retrieval challenge. While this is subject to further investigation, our assumptions can be confirmed in the recent work with the HAPPS-1 method on the SHREC'20 protein dataset, published in [122] and discussed in Section 5.3.6.

5.3.6 Experiment 6: Evaluating The Retrieval Performances of The HAPPS-1 And HAPPS-2 Methods On SHREC 2020 Protein Dataset

In this section, we would examine the retrieval accuracies of both the HAPPS-1 and HAPPS-2 retrieval methods and compare them side-by-side with several other retrieval methods on the SHREC'20 protein dataset and retrieval challenge. For this retrieval challenge [122], a total of six groups (including us) from five different countries participated for the track and submitted, for performance evaluation, a total of fifteen sets of (dis)similarity matrices (including three from the HAPPS

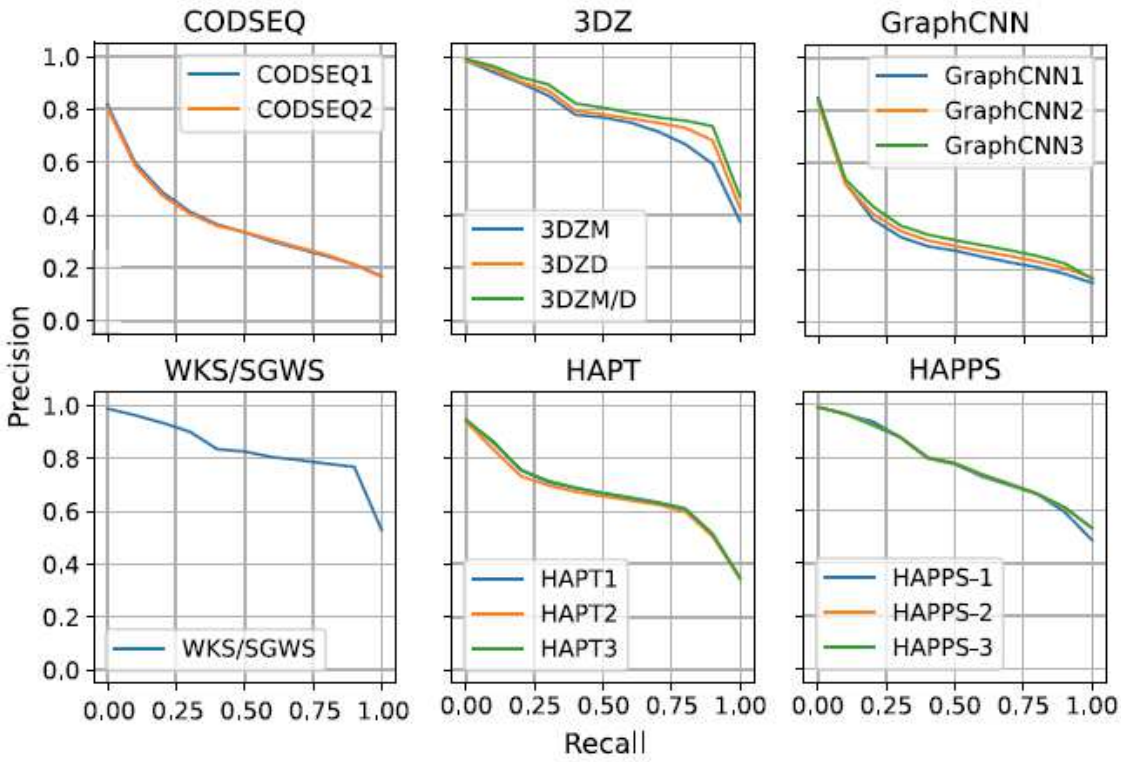


Figure 5.28: The PRC plots for the quantitative results in Table 5.24, regarding the *Protein* classification label, independently showing the plots of all six different retrieval methods with a total of 15 experimental runs, including three runs from HAPPS. Plots source: [122].

retrieval method), each of which represents the pairwise matching results between each shape taken as a query shape, in turn, and matched with other shapes in the dataset. We submitted results of two experimental runs using the HAPPS-1 (see Section 4.2.3) with different parameter settings in each run, and one run using the HAPPS-2 (see Section 4.2.3) methods. In this section, the quantitative performance of our HAPPS methods (described in Section 4.2.3) including each of the other five methods described in [122] are presented and assessed. The performance at the *Protein* (Figure 5.28 and Table 5.24) and the *Species* (Figure 5.29 and Table 5.26) classification levels as described in Section 5.2.9 and Table 5.3 are analysed.

Experiment 6: Parameter Settings and Configurations

The parameter settings for the three experimental runs we submitted for the SHREC'20 protein retrieval challenge are listed in Table 5.25. As we can see, the distance metric (Cosine) and the $b_{(APPFD)}$ parameter are kept constant for all three experimental runs. We decided to test the outcome of varying the number of points sample, N, including the r and vs parameters in experimental run-6b and run-6c. In the third experimental run, we further tried to probe the effect of using an alternative global descriptor (the M2DP), which is robust and successful in earlier computer vision application (i.e. loop closure detection), other than the one we proposed (the HoGD).

Method [122]	NN	FT	ST	mAP
CODSEQ1	0.697	0.350	0.266	0.358
CODSEQ2	0.666	0.345	0.264	0.356
3DZD	0.978	0.753	0.428	0.797
3DZM	0.975	0.719	0.422	0.766
3DZD/3DZM average	0.980	0.789	0.436	0.823
WKS/SGWS	**0.985**	**0.818**	**0.438**	**0.840**
HAPT1	0.898	0.617	0.407	0.658
HAPT2	0.875	0.602	0.402	0.646
HAPT3	0.892	0.620	0.406	0.659
GraphCNN1	0.773	0.278	0.218	0.301
GraphCNN2	0.734	0.295	0.235	0.317
GraphCNN3	0.770	0.310	0.243	0.339
HAPPS-1	0.982	0.738	0.416	0.774
HAPPS-2	0.983	0.746	0.420	0.779
HAPPS-3	0.983	0.746	0.420	0.779

Table 5.24: Quantitative performance evaluation of the six different retrieval methods for the SHREC'20 protein retrieval, using four statistical metrics at the *Protein* classification level. Each method submitted at least one experimental run: For each metric, the highest value is in bold. Results source: [122].

Dataset:	SHREC'20 [122]	**Parameter Settings**					
Experiment 6	**Algorithms**	N	r	vs	$b_{(APPFD)}$	$b_{(HoGD)}$	**Dist.Metric**
run-6a	HAPPS-1	4200	0.40	0.20	7	65	Cosine
run-6b	HAPPS-1	3500	0.50	0.20	7	65	Cosine
run-6c	HAPPS-2	3500	0.50	0.20	7	–	Cosine

Table 5.25: Parameter settings for the HAPPS method on the SHREC'20 protein retrieval dataset, where we presented three experimental runs: run-6a (HAPPS-1), run-6b (HAPPS-1), and run-6c (HAPPS-2). The parameters: N, r, vs, $b_{(APPFD)}$, $b_{(HoGD)}$ are explained in Section 5.3.

The goal was twofold: (i) to further investigate how much impact/contribution the HoGD global descriptor has on the overall HAPPS retrieval method, and (ii) to maximize the possibility of further improving the overall retrieval accuracies of the HAPPS method with a possibly more robust global descriptor, beyond the capabilities of the HAPPS-1 method with HoGD. Therefore, we adopted exactly the same parameter settings for experimental run-6b (with HoGD global descriptor) and run-6c (with M2DP global descriptor), for fair comparison.

For the experimental evaluations in this retrieval challenge, only four quantitative performance evaluation metrics, including the PRC plots, were used, thus: NN,

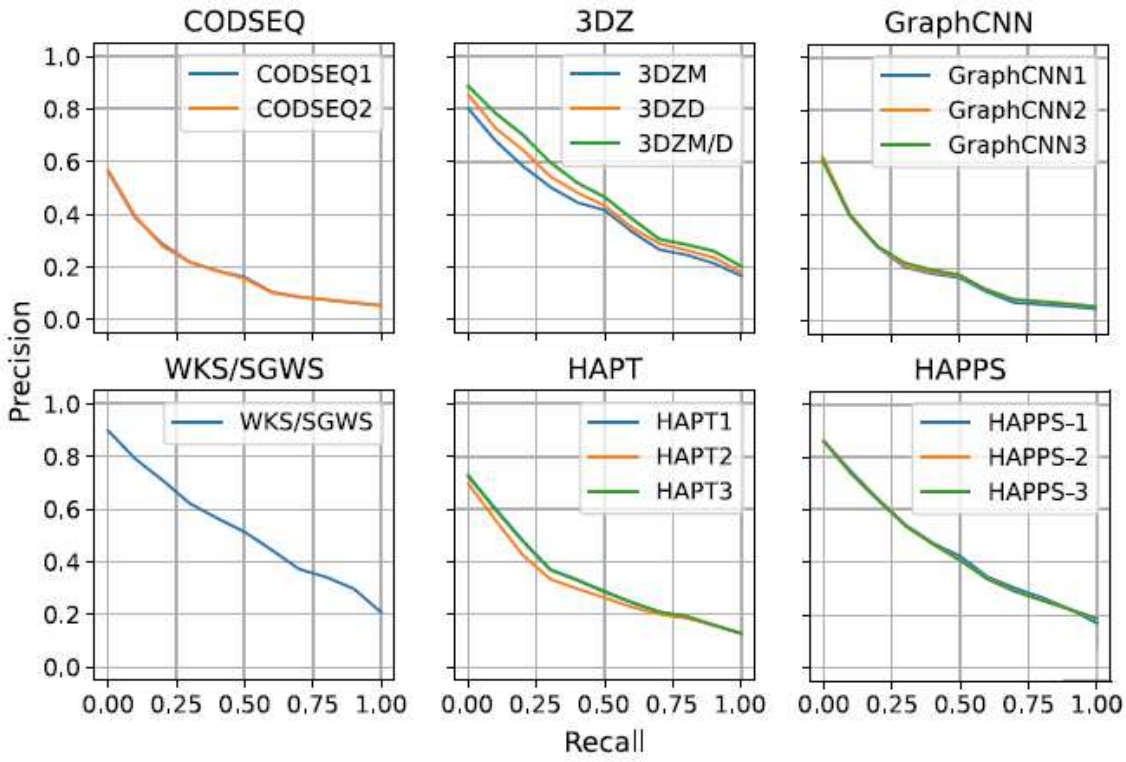


Figure 5.29: The PRC plots for the quantitative results in Table 5.26, regarding the *Species* classification label, independently showing the plots of all six different retrieval methods with a total of 15 experimental runs, including three runs from HAPPS. Plots source: [122].

FT, ST, mAP, to evaluate the retrieval performances of all the competing retrieval methods mentioned above. We refer the reader to Section 4.3.3 for more details regarding these performance metrics. Please note that both HAPPS-1 and HAPPS-2 in [122] are the same as the HAPPS-1 described in this thesis (see Section 4.2.3), while the HAPPS-3 in [122] is the same as the HAPPS-2 described in this thesis (see Section 4.2.3). In this thesis, the experimental runs: run-6a, run-6b, and run-6c, respectively refers to HAPPS-1, HAPPS-2, and HAPPS-3 mentioned in Table 5.24, Table 5.26, Figure 5.28 and Figure 5.29, where experimental runs: run-6a and run6-b are from the same method (the HAPPS-1, described in this thesis) with different parameter settings described in Table 5.25, while run-6c is from a different method (the HAPPS-2 described in this thesis).

Experiment 6: Results and Discussion

The quantitative results of all the three experimental runs of our HAPPS method, including results of five other state-of-the-art methods for the SHREC'20 protein dataset have been presented for both the *Protein* and *Species* classification levels. As mentioned in Section 5.2.9 and Table 5.3, two classification levels are used to evaluate the accuracies of 3D shape retrieval methods for the SHREC'20 protein retrieval challenge. Therefore, the quantitative retrieval performances of all six retrieval methods (see Section 5.3.6 and [122]) at the *Protein* classification level,

Method [122]	NN	FT	ST	mAP
CODSEQ1	0.438	0.173	0.125	0.180
CODSEQ2	0.447	0.172	0.124	0.179
3DZD	0.783	0.391	0.262	0.435
3DZM	0.722	0.369	0.256	0.402
3DZD/3DZM average	0.825	0.419	0.277	0.470
WKS/SGWS	**0.844**	**0.460**	**0.298**	**0.508**
HAPT1	0.595	0.286	0.209	0.313
HAPT2	0.572	0.264	0.200	0.289
HAPT3	0.608	0.286	0.209	0.313
GraphCNN1	0.513	0.177	0.117	0.178
GraphCNN2	0.533	0.175	0.120	0.186
GraphCNN3	0.499	0.181	0.122	0.186
HAPPS-1	0.757	0.407	0.272	0.432
HAPPS-2	0.772	0.400	0.269	0.430
HAPPS-3	0.768	0.400	0.269	0.430

Table 5.26: Quantitative retrieval performances measures of the six different retrieval methods for the SHREC'20 protein retrieval, using four statistical metrics at the *Species* classification level. Each method submitted at least one experimental run: For each metric, the highest value is in bold. Results source: [122].

and their corresponding PRC plots are presented in Table 5.24 and Figure 5.28, respectively. Similarly, the quantitative retrieval performances of these methods at the *Species* classification level, and their corresponding PRC plots are presented in Table 5.26 and Figure 5.29, respectively. From these results, on both classification levels, we see that three methods (3DZM/D, WKS/SGWS, and HAPPS) achieved successful NN retrieval with more than 97.5%, and more than 75.5% for the *Protein* and *Species* classification level, respectively. Overall, out of the six methods for this retrieval challenge our method (HAPPS) ranked top three and very closely struggles second position with the 3DZD/3DZM method. We refer the reader to [122] for further analyses regarding the results of this retrieval challenge.

5.4 Experimental Evaluations of The Retrieval Accuracies of The APPFD Method

In this section, we provide the results of all experimental evaluations using our APPFD retrieval method on some of the 3D datasets described in Section 5.2 and compare its retrieval accuracies to those of other state-of-the-art retrieval methods evaluated with the respective datasets. Most of the SHREC retrieval tracks adopt up to seven standard IR performance metrics described in Section 4.3.3, which

are: NN, FT, ST, E, DCG, mAP, and PRC (see Section 4.3.3). Similarly, we compare the retrieval accuracies of the APPFD method with the accuracies of other method for the respective datasets and retrieval challenges, using the exact metric from the track(s). The methodology for this retrieval method has been thoroughly described in Section 4.2. A thorough description of its core influential parameters and their effects on the final descriptors (APPFD and HAPPS) have already been presented in Section 5.3. Therefore, in this section, we would only present the quantitative results of this method, including (where possible) their respective PRC plots on two standardised SHREC benchmark datasets (SHREC'12 and SHREC'14), and analyse its retrieval accuracies side-by-side with several other state-of-the-art retrieval methods for the respective retrieval challenge on these datasets. Unlike in Section 5.3 where up to six different datasets are evaluated with the HAPPS method, only two datasets are reported for the APPFD method in order to keep the thesis compact, considering the amount of details involved in each investigation.

5.4.1 Experiment 7: Evaluating The Retrieval Accuracies of The APPFD Method On SHREC'12 Dataset

In this section, we present and evaluate the retrieval performances of our APPFD retrieval method using the SHREC'12 dataset (see Section 5.2.4). This dataset contains 1,200 triangular meshes in the generic categories, having both rigid and non-rigid 3D objects, most of which are non-watertight. First, we compare the quantitative retrieval performance of two experimental runs (run-7a and run-7b) of the APPFD method with different parameter settings for each run, and the complete 6-dimensional features (APPF). This is to allow us adopt results of the experimental settings with the highest performance for later comparison with other state-of-the-art methods. Secondly, we provide further analysis regarding the effects of the APPF and demonstrate how the number of extracted features affects the final results (i.e. retrieval accuracies) of the APPFD retrieval method, by showing the retrieval results of adopting only 5 of 6 APPF described in Section 4.2.1: $f_1 = (\phi, \alpha, \beta, \gamma, \delta)$ and compare its results with those obtained with the complete 6-dimensional APPF: $f_2 = (\phi, \theta, \alpha, \beta, \gamma, \delta)$. Several different experimental runs were conducted, but we analyse only four runs: run-7c to run-7f. We show these comparative results analyses in Table 5.29. Finally, we compare our best overall quantitative results (run-7b) with the best results of each of the five other state-of-the-art retrieval methods, which competed for the SHREC 2012 generic shape retrieval challenge. For all the performances evaluation in this section, the following six standard metrics described in Section 4.3.3 are used, in line with [131]: NN, FT, ST, E, DCG, including the PR plots.

Experiment 7: Parameter Settings, Results and Discussion

The experimental evaluation approach we have adopted regarding the SHREC'12 dataset/retrieval challenge is as mentioned in Section 5.4.1 above, with the parameter setting outlined in Table 5.27. In Section 5.3, we demonstrated through experimental evaluations, how the $b_{(APPFD)}$ parameter affects both the retrieval accuracies and compactness (size) of the final APPFD/HAPPS signature. For this experimen-

tal evaluation, we consider the importance of indicating the computational time of the APPFD on a single 3D object and showing how the different parameter settings affects the computational time. First, we want to show the effects of increased LSP size and decreased number of bins (i.e. $b_{(APPFD)}$) on the APPFD retrieval method. Surprisingly, the quantitative results for experimental run-7b (see Table 5.28) shows better retrieval performances with $b_{(APPFD)} = 7$ and $r = 0.50$, unlike the former run: run-7a, with $b_{(APPFD)} = 8$ and $r = 0.40$. It therefore becomes a trade-off between *fv* dimensionality and computational time, as it takes approximately 2 minutes to compute APPFD experimental run-7b parameter settings, opposed to only about 28 seconds. Essentially, the outcome (i.e. descriptors) of experimental run-7b with 117,649-dimensional *fv*, will impose lesser computational time during matching than run-7a with 262,144-dimensions. Therefore, in this regard, irrespective of high-dimensional *fv* or LSP size, the experimental runs whose parameter settings return the highest retrieval accuracies would be preferable for a given retrieval task.

Dataset:	SHREC'12 [131]	**Parameter Settings**					
Experiment 7	**Algorithms**	N	r	vs	$b_{(APPFD)}$	**Metric**	**Run Time/3D**
run-7a	APPFD-6d	4200	0.40	0.30	8	Cosine	26 secs
run-7b	APPFD-6d	4200	**0.50**	**0.30**	**7**	**Cosine**	**98 secs**
run-7c	APPFD-5d	4200	0.40	0.30	8	Cosine	26 secs
run-7d	APPFD-5d	4200	0.40	0.30	5	Cosine	26 secs
run-7e	APPFD-5d	4200	0.30	0.30	5	Cosine	12 secs
run-7f	APPFD-5d	4200	0.27	0.30	5	Cosine	3 secs

Table 5.27: Parameter settings for the APPFD method on the SHREC'12 generic shapes dataset, where we first present two experimental runs: run-7a, and run-6b with different r and $b_{(APPFD)}$ parameters using complete 6-dimensional features. Next: run-7c to run-7f, we present parameter settings for other experimental runs with 5-dimensional features. Here, we show how the $r, /vs$, parameters affects the average computational time of the APPFD per 3D object. All parameters are explained in Section 5.3.

Secondly, using this experimental evaluation, we want to be able to specifically show the contribution of our novel 2-dimensional local geometric features: θ, ϕ (see Section 4.2.1, Figure 4.4), which we use to augment the 4-dimensional PPF in [252]. Therefore, in experimental runs: run-7c to run-7f (Table 5.27), we decide to drop one of our 2-dimensional feature, ϕ, leaving us with a 5-dimensional APPF, $f1 = (\phi, \alpha, \beta, \gamma, \delta)$. The corresponding results (Table 5.28) of these experimental runs in Table 5.27 reveals significant decrease in performance using 5-dimensional features, opposed to 6-dimension, while keeping the parameters the same (see parameter settings and their corresponding results for experimental run-7a and run-7c).

Finally, in Table 5.29, we compare the best results of the APPFD experimental runs (run-7b) with the best of each of the five methods in [131]. While six

Experiment7	**Dist.Metric**	**Algorithms**	**NN**	**FT**	**ST**	**E**	**DCG**
run-7a	Cosine	APPFD-6d	0.551	0.280	0.368	0.256	0.585
run-7b	Cosine	APPFD-6d	**0.580**	**0.298**	**0.395**	**0.272**	**0.607**
run-7c	Cosine	APPFD-5d	0.527	0.256	0.339	0.238	0.566
run-7d	Cosine	APPFD-5d	0.514	0.256	0.345	0.239	0.572
run-7e	Cosine	APPFD-5d	0.496	0.244	0.328	0.228	0.555
run-7f	Cosine	APPFD-5d	0.448	0.218	0.302	0.209	0.529

Table 5.28: Quantitative results of five standard evaluation metrics revealing the retrieval performances of our APPFD retrieval method on the SHREC'12 dataset, for the respective parameters in experimental runs: run-7a to run-7f. All results are from the Cosine distance metric.

quantitative performance evaluation metrics (NN, FT, ST, E, DCG and Average Precision (AP)) were used in [131], including the PRC plots, we compare our results for this dataset using only five metrics (NN, FT, ST, E, and DCG), for simplicity. Unfortunately, our method recorded relatively low retrieval accuracies for the SHREC'12 dataset. This is due to the reasons discussed in Section 3.2.1, regarding defective data, and majorly some of the reasons summarised in Section 6.3.2. The SHREC'12 is characterised with lots of defective data (3D triangular meshes). Out of the 1200 3D shapes in this dataset, we identified 131 faulty models, having degenerate faces and vertices, inconsistent face windings, unreferenced vertices and faces, missing surface parts, other kinds of defects, artefacts, etc. Besides the steps mentioned in Section 3.2, our retrieval methods did less in the pre-processing stage to automatically handle defective data. Alternatively, manual pre-processing of data before feeding the data into our retrieval algorithm could further improve the results we obtain with our method for this kind of dataset.

Author	Method	NN	FT	ST	E	DCG
Yanagimachi [131]	**DG1SIFT**	**0.879**	**0.661**	**0.799**	**0.576**	**0.871**
Tatsuma [131]	DVD+DB+GMR	0.828	0.613	0.739	0.527	0.833
Li [131]	ZFDR	0.818	0.491	0.621	0.442	0.776
Redondo [131]	3DSP-L2-1000-hik	0.685	0.376	0.502	0.351	0.685
This Thesis	APPFD run-7b	0.580	0.298	0.395	0.272	0.607
Bai [131]	LSD-sum	0.517	0.232	0.327	0.224	0.565

Table 5.29: Quantitative evaluation measures of only the best results of each methods presented in [131], including our APPFD method for SHREC'12 benchmark dataset.

As we can see from Table 5.29, although the DG1SIFT method had the overall best performance in the SHREC'12 retrieval challenge, followed by DVD+DB+GMR and the rest, none of these state-of-the-art methods recorded up to 90% retrieval accuracy in any of the performance statistics, unlike with several other datasets and retrieval challenges we have evaluated in this thesis so far (see Section 5.3).

This further confirms the difficulties, in terms of retrieval challenge, posed by the SHREC'12 dataset. Nevertheless, our retrieval method did not perform too poorly on for this evaluation, considering that its results were not too far from the best overall and that they outperform at least one state-of-the-art method, the LSD-sum. We refer the reader to [131] for a detailed description of each of the retrieval methods that competed for this retrieval challenge, and further analysis regarding the results and evaluation aspects not mentioned in this thesis.

5.4.2 Experiment 8: Evaluating The Retrieval Accuracies of The APPFD Method On SHREC'14 Dataset

The SHREC 2014 track on Large Scale Comprehensive 3D Shape Retrieval featured five groups of participants who submitted a total of 14 experimental runs. The objective of this track is to evaluate the performance of 3D shape retrieval approaches on a large-sale comprehensive 3D shape database which contains different types of models, such as generic, articulated, CAD and architecture models [132], making a total of 8,987 3D objects, categorized into 171 classes. In this section, we show how the results of our APPFD method compares with other state-of-the-art methods for this retrieval challenge. Seven commonly adopted performance metrics in 3D model retrieval technique such as the NN, FT, ST, E, DCG, PRC plot, and AP were employed by [132] to have a comprehensive evaluation of the retrieval algorithms submitted to the track. In addition to the common definitions, they also developed weighted variation for each benchmark by incorporating the popularity of each class in real work and life, based on the number of available 3D objects, where they assume there is a linear correlation between the number of available models in one class and the degree of popularity of the class. Therefore, they adopt a weight of reciprocal of the number of models to define each weighted performance metric. In line with this, they defined two metrics: the proportionally and reciprocally weighted metrics, $mr(m = NN/FT/ST/E/DCG)$ as follows:

$$m_p = \sum_{i=1}^{N} \frac{n_i}{N} \cdot m_i$$

,

$$m_r = \sum_{i=1}^{N} \frac{\frac{1}{n_i}}{\sum_{j=1}^{N} \frac{1}{n_j}} \cdot m_i$$

where N is the total number of models, n_i is the size of class to which the i^{th} model belongs, m_i is the non-weighted NN/FT/ST/E/DCG metric value for the i^{th} model. m_p assigns bigger weights to the classes with more variations. On the contrary, m_r highlights the performance in retrieving classes with few models/variations. For these experimental evaluations, we also employ all performance metrics mentioned above, except the AP, including computing evaluation scores for the two weighted performance metrics: the proportionally weighted (m_p) and the reciprocally weighted (m_r) metrics, using the evaluation codes provided by [132].

Experiment 8: Parameter Settings and Configurations

Several experimental testing were done on the SHREC'14 dataset using the APPFD algorithm and for each experimental run, we selected different parameter settings

like in the previous instances. However, unlike the experimental evaluations we have presented with the other benchmark datasets, where our 6-dimensional features (APPF) were adopted for both the APPFD and HAPPS algorithm, we decided to adopt only 5-dimensional APPF: $f3 = [\phi, \theta, \alpha, \beta, \gamma]$, where we excluded the fourth feature dimension (δ) of the 4-dimensional features in [252], and embraced the following 3-dimensional features: (α, β, γ), which is also used in PFH [200], because, according to [200], this feature does not present an extreme significance for some datasets, such as 2.5D datasets - see Section 2.6.1. Nevertheless, for 3D dataset, δ represents the distance between the source point, p_i and the target point, p_j in the point pair combination (see Figure 4.4), and takes care of scaling transformation, but in this experimental evaluation, we want to see how our results would be affected with a minimal combination of the extracted PPF. For all the parameter settings in Table 5.30, we also present the quantitative performance results considering both the comprehensive proportionally-weighted (see Table 5.32) and the comprehensive reciprocally-weighted (see Table 5.33) evaluation criteria.

Dataset:	SHREC'14 [132]	**Parameter Settings**					
Experiment 8	**Algorithms**	N	r	vs	$b_{(APPFD)}$	**Metric**	**Run Time/3D**
run-8a	APPFD-6d	4200	0.40	0.30	7	Cosine	≈ 26 *Sec.*
run-8b	APPFD-5d	4200	0.40	0.30	5	Cosine	≈ 26 *Sec.*
run-8c	**APPFD-5d**	**4200**	**0.30**	**0.30**	**5**	Cosine	$\approx$ **22 Sec.**
run-8d	APPFD-5d	4200	0.27	0.30	5	Cosine	≈ 4 *Sec.*
run-8e	APPFD-5d	4200	0.17	0.30	5	Cosine	≈ 1.6 *Sec.*

Table 5.30: Parameter settings for the APPFD method on the SHREC'14 large-scale, generic dataset. The following 5-dimensional APPFD features were used, instead: $f3 = [\phi, \theta, \alpha, \beta, \gamma]$. However, in run-8a, we first present parameter setting with 6-dimensional APPF in order to further compare the results with the other runs: run-8b to run-8e, which used $f3$. Again, we show how the $r, /vs$, parameters affects the average computational time of the APPFD per 3D object. All parameters are explained in Section 5.3.

Experiment 8: Results and Discussion

In Table 5.30, different parameter settings for five experimental runs (run-8a to run-8e) are presented and results of these runs are examined in Table 5.31. In the first experimental run (run-8a), we used all 6-dimensional APPF with $b_{(APPFD)} = 7$ and the Cosine distance metric. In subsequent runs (run-8b to run-8e), we decided to investigate the outcome of adopting a minimal set of APPF, $f3 = [\phi, \theta, \alpha, \beta, \gamma]$, with only five of six dimensions by dropping the δ feature in [252], while including our novel 2-dimensional feature, $[\theta, \phi]$, to have $f3$. Next, instead of 7 bins, we set the parameter $b_{(APPFD)} = 5$ for the other experimental runs. Having a minimal set of feature-dimensions and a lower number of bins has several advantages of computational speed, final descriptor compactness, but low descriptive power. This analysis

Experiment8	Dist.Metric	Algorithms	NN	FT	ST	E	DCG
run-8a	Cosine	APPFD-6d	0.650	0.158	0.216	0.102	0.589
run-8b	Cosine	APPFD-5d	0.647	0.172	0.239	0.104	0.602
run-8c	Cosine	APPFD-5d	**0.666**	**0.185**	**0.257**	**0.110**	**0.614**
run-8d	Cosine	APPFD-5d	0.615	0.157	0.223	0.095	0.589
run-8e	Cosine	APPFD-5d	0.570	0.138	0.198	0.084	0.571

Table 5.31: Quantitative results of five standard evaluation metrics revealing the retrieval performances of our APPFD retrieval method on the SHREC'14 dataset, for the respective parameters in experimental runs: run-8a to run-8e, considering only the comprehensive evaluation criteria. All results are from the Cosine distance metric.

enables us to see to what extent the APPFD retrieval method is affected by these parameter settings.

Experiment8	Dist.Metric	Algorithms	NN	FT	ST	E	DCG
run-8a	Cosine	APPFD-6d	134.788	30.059	44.090	14.787	130.358
run-8b	Cosine	APPFD-5d	136.496	35.441	53.245	15.565	134.815
run-8c	Cosine	APPFD-5d	**140.226**	**38.391**	**57.816**	**16.322**	**137.125**
run-8d	Cosine	APPFD-5d	130.511	32.592	49.742	14.375	132.688
run-8e	Cosine	APPFD-5d	121.111	29.045	44.774	12.822	129.595

Table 5.32: Quantitative results of five standard evaluation metrics revealing the retrieval performances of our APPFD retrieval method on the SHREC'14 dataset, for the respective parameters in experimental runs: run-8a to run-8e, considering only the comprehensive proportionally-weighted evaluation criteria given in [132].

The PRC plots of the quantitative evaluation in Table 5.31 are shown in Figure 5.30, which allows us to see clearly how the different parameter settings affects the retrieval accuracies of the APPFD method. Surprisingly from these plots, we see that the PRC plot of experimental run-8c, with lower parameter values ($r = 0.30$, $vs = 0.30$, $b_{(APPFD)} = 5$) and a final fv size of $5^5 = 3,125$-dimension produces performance accuracies which is better than experimental run-8a, which has $r = 0.40$, $vs = 0.30$, $b_{(APPFD)} = 4$ parameter values, and final fv of $7^6 = 117,649$-dimension - anticipated to have better performance results. Furthermore, there is a slight difference with regards to the performance of experimental run-8a and run-8b. Basically, the observation is that the parameter values usually reduce with the performance accuracies and vice versa, while overly increasing the parameter values would also not return satisfactory results for the APPFD retrieval method.

Next, we show the proportionally and the reciprocally weighted metrics evaluation results in Table 5.32 and Table 5.33, respectively for the parameter settings in Table 5.31. Note that all of these evaluations are carried out using the Cosine

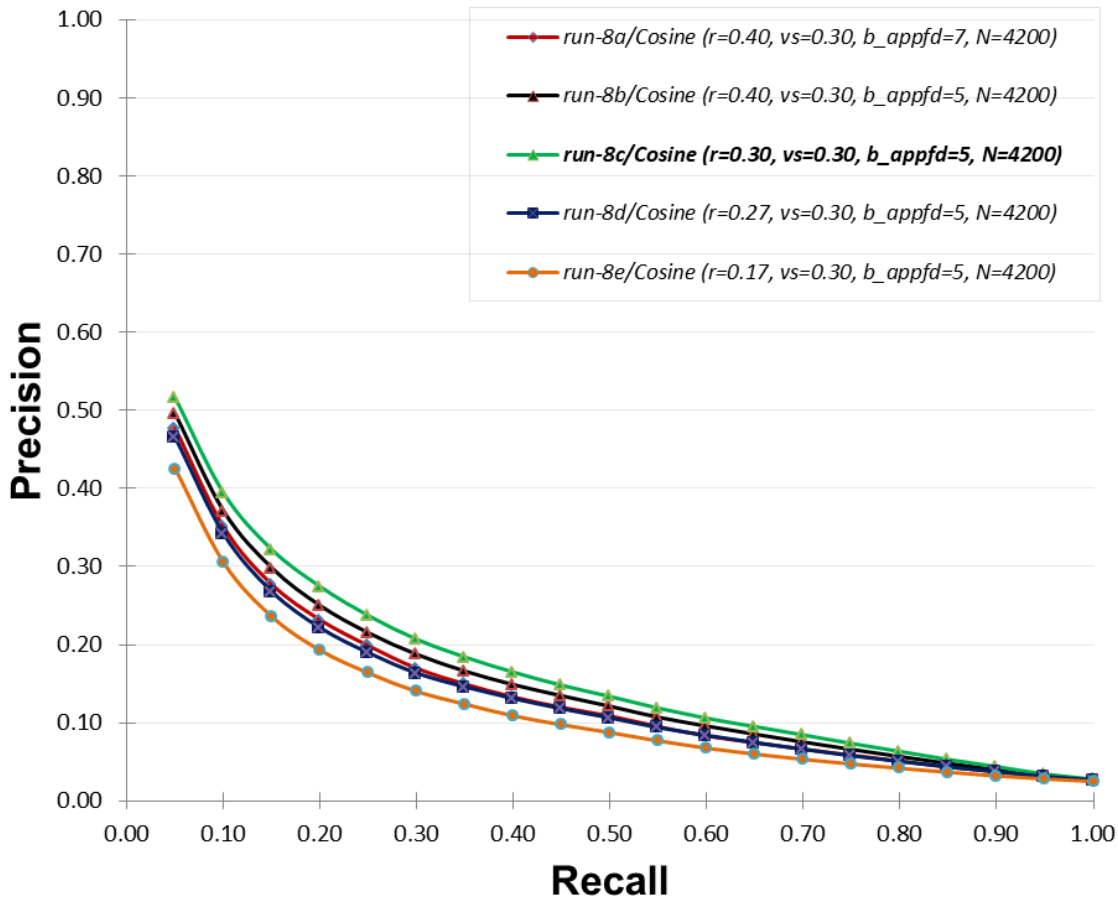


Figure 5.30: The PRC plots of the APPFD retrieval method for the quantitative results in Table 5.31, having experimental run-8a to run-8e, and considering the comprehensive evaluation criteria only. The plot for run-8c performs slightly better than the rest.

distance metric. From these results, we can also see a similar performance trend to those reported for the comprehensive quantitative evaluation in Table 5.31, where the results of experimental run-8c remains the highest.

Finally, we compare the quantitative results of run-8c (overall best results of the APPFD method) to only the best of each of the five state-of-the-art retrieval methods in [132], considering all three evaluation criteria, which are: (i) comprehensive evaluation, (ii) comprehensive proportionally-weighted evaluation, and (iii) comprehensive reciprocally-weighted evaluation. We show each of these comparative results in Table 5.34, Table 5.35, and Table 5.36, for comprehensive, comprehensive proportionally-weighted, and comprehensive reciprocally-weighted evaluations, respectively. According to these results, our APPFD retrieval method ranks the lowest, while the LCDR-DBSVC method [132] ranked the highest in all three different evaluation criteria. This low performance is expected due to the choice of parameter values, such as the $b_{(APPFD)}$ and the minimal number of the APPF we adopted. Secondly, extremely complicated models, with varied quality, high degree of rigidity, and small number of sampling points, etc., could also be responsible.

Experiment8	**Dist.Metric**	**Algorithms**	**NN**	**FT**	**ST**	**E**	**DCG**
run-8a	Cosine	APPFD-6d	3.423	1.072	1.297	0.635	2.924
run-8b	Cosine	APPFD-5d	3.397	1.118	1.353	0.643	2.978
run-8c	Cosine	**APPFD-5d**	**3.498**	**1.179**	**1.414**	**0.685**	**3.036**
run-8d	Cosine	APPFD-5d	3.199	1.046	1.269	0.598	2.889
run-8e	Cosine	APPFD-5d	2.932	0.935	1.159	0.522	2.769

Table 5.33: Quantitative results of five standard evaluation metrics revealing the retrieval performances of our APPFD retrieval method on the SHREC'14 dataset, for the respective parameters in experimental runs: run-8a to run-8e, considering only the comprehensive reciprocally-weighted evaluation criteria given in [132].

Author	Method	NN	FT	ST	E	DCG
Aono [132]	KVLAD	0.605	0.413	0.546	0.214	0.746
Chen [132]	DBNAA_DERE	0.817	0.355	0.464	0.188	0.731
Furuya [132]	MR-D1SIFT	0.856	0.465	0.578	0.234	0.792
Li [132]	ZFDR	0.838	0.386	0.501	0.209	0.757
Tatsuma [132]	**LCDR-DBSVC**	**0.864**	**0.528**	**0.661**	**0.255**	**0.823**
This Thesis	APPFD run-8c	0.666	0.185	0.257	0.110	0.614

Table 5.34: Quantitative evaluation measures of only the best results of each methods presented in [132], including our APPFD method for SHREC'14 benchmark dataset, considering only the comprehensive evaluation criteria.

Especially, more points could be used for the extraction of features at the cost of computational efficiency. However, the performance of the APPFD is not too far from that of the other methods, especially for the NN and DCG statistics.

Author	Method	NN	FT	ST	E	DCG
Aono [132]	KVLAD	123.059	83.382	114.400	28.756	160.724
Chen [132]	DBNAA_DERE	171.149	79.380	108.438	27.193	159.316
Furuya [132]	MR-D1SIFT	178.497	94.309	121.762	31.804	167.318
Li [132]	ZFDR	175.142	79.407	106.578	29.422	161.351
Tatsuma [132]	**LCDR-DBSVC**	**177.863**	**107.851**	**144.179**	**33.691**	**173.773**
This Thesis	APPFD run-8c	140.226	38.391	57.816	16.322	137.125

Table 5.35: Quantitative evaluation measures of only the best results of each methods presented in [132], including our APPFD method for SHREC'14 benchmark dataset, considering only the comprehensive proportionally-weighted evaluation criteria.

As previously indicated, the idea behind the choice of parameter values for the different experimental runs, using the APPFD retrieval method, in this (SHREC'14) evaluation are: (i) to mitigate the use of computational resources, such as memory and time, as a result of the exceptionally large size of the SHREC'14 dataset, and (ii)

to further investigate the impact of our 2-dimensional feature (θ, ϕ), to the APPF augmentation, after dropping the δ feature. As we can see, the quantitative results of experimental run-8a (where 6-dimensional APPF and $b_{(APPFD)} = 7$ were used) is lower than the results of run-8b and run-8c (where 5-dimensional APPF and $b_{(APPFD)} = 5$ were used), although all these results are with slight differences. This confirms the robustness of the θ and ϕ local features we proposed (see Section 4.2.1). In future implementation, we are interested in only applying these 2-dimensional local features, in a suitable manner to further investigate its discriminating abilities, besides other features.

Author	Method	NN	FT	ST	E	DCG
Aono [132]	KVLAD	3.261	2.196	2.945	1.445	3.830
Chen [132]	DBNAA_DERE	4.252	1.911	2.306	1.146	3.747
Furuya [132]	MR-D1SIFT	4.678	2.604	3.090	1.542	4.263
Li [132]	ZFDR	4.476	2.216	2.661	1.321	3.994
Tatsuma [132]	**LCDR-DBSVC**	**4.881**	**2.905**	**3.435**	**1.731**	**4.470**
This Thesis	APPFD run-8c	3.498	1.179	1.414	0.685	3.036

Table 5.36: Quantitative evaluation measures of only the best results of each methods presented in [132], including our APPFD method for SHREC'14 benchmark dataset, considering only the comprehensive reciprocally-weighted evaluation criteria.

In conclusion, it is very interesting to find, from this experimental evaluation, how the computational time of the APPFD retrieval method drastically improved from approximately 26 seconds to about 2 seconds as the LSP size further reduced from $r = 0.40$ to $r = 0.17$, while maintaining constant values for the vs and $b_{(APPFD)}$ parameters (at $vs = 0.30$ and $b_{(APPFD)} = 5$, respectively). With all of these, we have demonstrated that our APPFD retrieval method is capable of yielding far better retrieval accuracies in all evaluation criteria and metric for the SHREC'14 large scale comprehensive benchmark dataset, given higher parameter values for this method. By indication, also, our HAPPS retrieval method is believed to perform very well for this dataset and rank among top state-of-the-art methods. We refer the reader to [132] for more details regarding the methods mentioned in Table 5.34, Table 5.35, and Table 5.36, including further analyses of the results presented in [132].

5.5 Experimental Evaluations of The APPFD-FK-GMM Retrieval Method

The implementation goal and detailed description of our proposed APPFD-FK-GMM retrieval method is presented in Section 4.2.4. We applied this method to several benchmark datasets and found, through extensive experimental evaluations, that it returns near retrieval performance accuracies to the HAPPS retrieval methods, while producing a more compact final shape descriptor. In this thesis we present the results of the APPFD-FK-GMM retrieval method on four of the most resent SHREC datasets, which are: SHREC'17, SHREC'18, SHREC'19 and a differ-

ent type of the SHREC'20 datasets (see Section 5.2). Next, we compare the retrieval performances of this method to that of the HAPPS-1 retrieval method, respectively for each of these four datasets. Finally, for each dataset, a comparative analysis of the retrieval performance of our APPFD-FK-GMM method is compared to those of several recent state-of-the-art methods on the respective datasets.

5.5.1 Parameter Settings and Configurations for The APPFD-FK-GMM Retrieval Method

During experimental testing and evaluation for the APPFD-FK-GMM method, only the Cosine and Euclidean distance metrics (see Section 2.2.3) were found to provide the best overall retrieval accuracies, therefore all the quantitative performance reports presented in this section are based upon these two metrics. Considering the APPF, which relies on surface point sampling, and is the base feature extraction algorithm technique for the APPFD-FK-GMM retrieval method, the number of point sample for all the experimental evaluation using this method are between 3,500 to 4,500 samples as we would see in the following sections. We have already described the $b_{(APPFD)}$ binning technique for this retrieval approach with the FK in Section 4.2.4, where $b_{(APPFD)} = 35 \times 6$, using a 1-dimensional histogram to collect each of the 6-dimensional APPF and finally concatenate them to yield final descriptor. Considering the training of GMM with local APPF, we conducted several different experiments with different values of the G parameter, to investigate which of the values would converge for different datasets and the APPFD, and found that $G = 10$ returned high performance accuracies.

5.5.2 Experiment 9: Evaluating The Retrieval Performances of The APPFD-FK-GMM Method On The SHREC'17 Dataset

Table 5.37 presents the retrieval accuracies of four experimental runs for this dataset evaluation with the APPFD-FK-GMM retrieval method, which are: (i) run-9a-Cosine (APPFD-FK-GMM): this involves using the final global descriptor produced by the APPFD-FK-GMM algorithm, as it is, which has a dimension of 4,210 and the Cosine distance metric for descriptor matching, (ii) run-9a-Cosine (APPFD-FK-GMM$_{(PCA)}$): prior to shape matching, this involves the application of traditional linear dimensionality reduction technique, the PCA in this case, to further reduce the dimension of the final global descriptor returned by experimental "run-9a-Cosine", which results in a final 92-dimensional *fv* that is more compact and most nearly as descriptive, (iii) run-9c-Euclidean (APPFD-FK-GMM): this involves using the final 4,210-dimensional global descriptor produced by the APPFD-FK-GMM algorithm, as it is, while descriptor matching is done with the Euclidean distance metric, instead, and finally (iv) run-9d-Euclidean (APPFD-FK-GMM$_{(PCA)}$): again, prior to shape matching, we applied the PCA technique to further reduce the dimension of the final global descriptor returned by experimental "run-9c-Euclidean" to 92-dimensional *fv* that is more compact and most nearly as descriptive.

According to the results in Table 5.37 and the corresponding PRC plots in Figure 5.31, it is interesting to see that: (i) both the Cosine and Euclidean metrics

Experiment 9	Algorithms	NN	FT	ST	E	DCG
run-9a-Cosine	**APPFD-FK-GMM**	**0.9900**	**0.8356**	**0.9433**	**0.4283**	**0.9620**
run-9b-Cosine	APPFD-FK-GMM$_{(PCA)}$	0.9900	0.8300	0.9422	0.4278	0.9603
run-9c-Euclidean	**APPFD-FK-GMM**	**0.9900**	**0.8356**	**0.9433**	**0.4283**	**0.9620**
run-9d-Euclidean	APPFD-FK-GMM$_{(PCA)}$	0.9900	0.8278	0.9422	0.4278	0.9601

Table 5.37: Quantitative retrieval performance results using standard evaluation metrics to evaluate APPFD-FK-GMM retrieval method on the SHREC'17 PRoNTo dataset, for experimental runs: run-9a to run-9d. Here, both the Cosine and Euclidean metric returns the same results.

returned exactly the same performance scores in all five quantitative statistics (NN, FT, ST, E, and DCG) as indicated by experimental run-9a and run-9c, and (ii) both of the PCA-reduced final global descriptors, with 92-dimensional *fv* each (run-9b and run-9d), produce exactly the same NN accuracies and negligible differences in the other four statistics. Overall, there is no visible differences in the retrieval performances obtained in all experimental runs (run-9a to run-9d) as revealed by the PRC plots. For this experimental evaluation, we tested several different parameter settings, but for simplicity, we report the following configurations for the APPFD and FK-GMM technique: $r = 0.40$, $vs = 0.20$ and the number of Gaussian, $G = 10$. Table 5.38 presents these details more clearly.

Dataset:	SHREC'17 [142]	**Parameter Settings**					
Experiment 9	**Algorithm(s)**	N	r	vs	$b_{(APPFD)}$	G	**Metric**
run-9a	APPFD-FK-GMM	3,000-4,500	0.40	0.20	35×6	10	Cosine

Table 5.38: Parameter settings for experimental run-9a with the APPFD-FK-GMM retrieval method on the SHREC'17 PRoNTo dataset.

Experiment 9: Results and Discussion, Comparing The APPFD-FK-GMM Method With The HAPPS-1 Method and Other State of The Art Methods On The SHREC'17 Dataset

In this section, we compare the outcome of experimental runs/methods having the best retrieval performance accuracies for the SHREC'17 PRoNTo dataset. For this comparison, we consider the APPFD-FK-GMM and HAPPS algorithms, including comparing these with two of the overall best state-of-the-art retrieval methods for the SHREC'17 PRoNTo retrieval track. The goal is to evaluate how best each of our methods has performed for this particular dataset. We show the quantitative results of such comparison in Table 5.39. Considering the HAPPS-1 retrieval method on the PRoNTo dataset, experimental run-2a with the KLD distance metric yielded the best overall results, while the BoW-RoPS-DMF-3 method in [142] ranked overall highest for that retrieval competition, followed closely by the BPHAPT method. This comparison, however, reveals that our APPFD-FK-GMM approach outperforms the

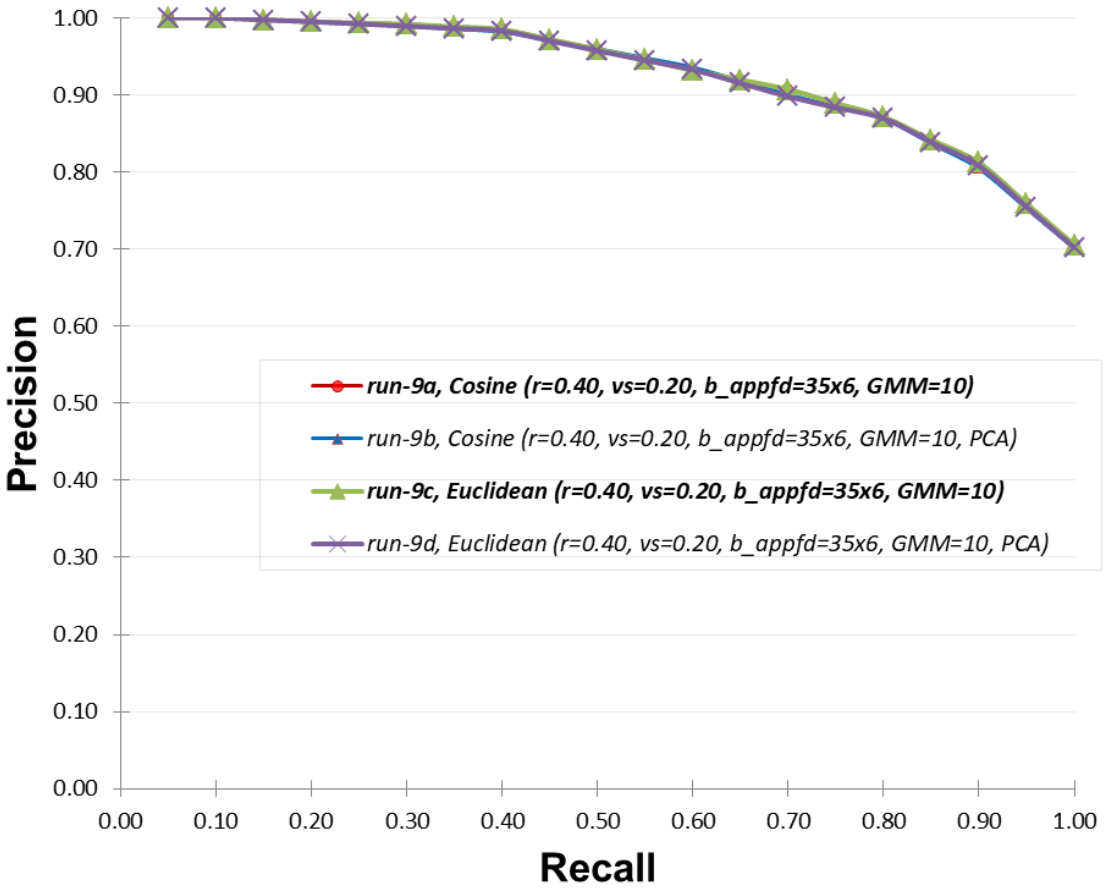


Figure 5.31: The PRC plots showing the quantitative retrieval performances of experimental run-9a to run-9d (see Table 5.37), using the APPFD-FK-GMM retrieval method on the SHREC'17 PRoNTo dataset. The plots show that there is no visible differences in the results of these four experimental runs, although quantitatively, the results of experimental run-9a and run-9c are a bit higher in values than those of run-9b and run-9d.

HAPPS-1 based method and BPHAPT method in some performance statistics, while also ranking closely next to the BoW-RoPS-DMF-3 method. In both the NN and E metrics, the APPFD-FK-GMM scores higher than the BPHAPT method which happened to be the second highest ranking method after BoW-RoPS-DMF-3 for the SHREC'17 retrieval challenge. In conclusion, all four methods in this comparison (see Table 5.39) has performed exceptionally well, and it is difficult to tell which of the BPHAPT or APPFD-FK-GMM methods ranks second place for this retrieval challenge, following the BoW-RoPS-DMF-3 method.

In terms of computational cost and robustness combined, our new APPFD-FK-GMM method has advantages over the HAPPS-1 method and the others, regarding the SHREC'17 retrieval tasks. For example, considering the HAPPS-1 method, experimental run-2a-kld adopts higher values for its $r = 0.50$, $vs = 0.30$, and $b_{(APPFD)} = 8$ parameters (see Table 5.8), unlike experimental run-9a/run-9c, with $r = 0.40$, $vs = 0.20$, $b_{(APPFD)} = 35$. The $b_{(APPFD)} = 8$ parameter in experimental run-2a would produce a final descriptor of $8^6 = 262,144$-dimensional fv, while the $b_{(APPFD)} = 35$ parameter in experimental run-9a would produce a final descriptor

Experiments	Algorithms	NN	FT	ST	E	DCG
run-9a-Cosine	APPFD-FK-GMM	0.9900	0.8356	0.9433	0.4283	0.9620
run-2a-kld	HAPPS-1	0.9900	0.7867	0.8744	0.4132	0.9367
SHREC'17 [142]	BPHAPT [142]	0.9800	0.9111	0.9544	0.4273	0.9743
SHREC'17 [142]	**BoW-RoPS-DMF-3 [142]**	**1.0000**	**0.9778**	**0.9978**	**0.4390**	**0.9979**

Table 5.39: Comparing the retrieval accuracies of two state-of-the-art retrieval methods (BoW-RoPS-DMF-3 and BPHAPT) on the SHREC'17 PRoNTo dataset, with the results accuracies of two of our proposed methods (APPFD-FK-GMM and HAPPS-1), using five standard quantitative evaluation metrics.

of $35 \times 1 = 210$-dimensional *fv*. In addition, run-2a combines the global HoGD, while the technique in run-9a relies only on the base APPFD algorithm. The computational cost associated with the method in run-2a is far greater compared to the method in run-9a, which makes the APPFD-FK-GMM method most preferable for a typical 3D-CBRS, than HAPPS-1 or any others mentioned. With the APPFD-FK-GMM implementation, we have been able to provide further improvement to our already robust local APPFD and hybrid HAPPS retrieval methods, including introducing a new robust and efficient global descriptor, the APPFD-FK-GMM. The comparative results of this method (see Table 5.39) provides a strong indication that adjusting the APPFD parameters of the APPFD-FK-GMM method to suitable settings would further improve its retrieval accuracies. For example, assuming the same parameter values in experimental run-2a are applied in run-9a/run-9c, the final APPFD-FK-GMM results from run-9a/run-9c would definitely outperform the results of experimental run-2a, with additional advantage of reduced overall computational cost.

5.5.3 Experiment 10: Evaluating The Retrieval Performances of The APPFD-FK-GMM Method On The SHREC'18 Protein Dataset

It became necessary to apply the APPFD-FK-GMM retrieval method on a different dataset having a larger number of 3D objects (in this case 2,267 protein surfaces), which also present a unique retrieval challenge to shape retrieval algorithm. In addition, we consider the need to compare the performance of the APPFD-FK-GMM method with the performance of other state-of-the-art methods, including the results of our earlier evaluation with the HAPPS-1 retrieval method on this dataset/retrieval task. Therefore, in this section, we follow exactly the same evaluation pattern for Experiment 9 and present, first the quantitative retrieval performances of the APPFD-FK-GMM retrieval method on the SHREC'18 protein dataset, using five evaluation metrics (NN, FT, ST, E, and DCG) and two distance metrics (the Cosine and Euclidean distances). We populate these quantitative results of four experimental runs: run-10a to run-10d in Table 5.41 and plot their corresponding PRC as shown in Figure 5.32. Finally, the bests of these results are compared side-by-side with results of the HAPPS-1 method and those of the best overall state-of-the-art retrieval method in the SHREC'18 retrieval challenge [121].

It is important to note that the parameter settings used for all the experimental runs in Experiment 10 are exactly the same as those used in Experiment 9 (see Section 5.5.2). The parameter settings of these experimental runs (run-10 to run-10d) for this evaluation are presented in Table 5.40.

Dataset:	SHREC'18 [121]	**Parameter Settings**					
Experiment 10	**Algorithm(s)**	N	r	vs	$b_{(APPFD)}$	G	**Metric**
run-10a	APPFD-FK-GMM	4,500	0.40	0.20	35×6	10	Cosine

Table 5.40: Parameter settings for experimental run-10a with the APPFD-FK-GMM retrieval method on the SHREC'18 protein dataset. Note that similar settings also apply to run-10b to run-10d.

Experiment 10	**Algorithms**	**NN**	**FT**	**ST**	**E**	**DCG**
run-10a-Cosine	**APPFD-FK-GMM**	**0.8334**	**0.6410**	**0.7785**	**0.5546**	**0.8655**
run-10b-Cosine	APPFD-FK-GMM$_{(PCA)}$	0.8254	0.6368	0.7755	0.5519	0.8632
run-10c-Euclidean	**APPFD-FK-GMM**	**0.8334**	**0.6410**	**0.7785**	**0.5546**	**0.8655**
run-10d-Euclidean	APPFD-FK-GMM$_{(PCA)}$	0.8330	0.6404	0.7779	0.5541	0.8651

Table 5.41: Quantitative retrieval performance results using standard evaluation metrics to evaluate APPFD-FK-GMM retrieval method on the SHREC'18 protein dataset, for experimental runs: run-10a to run-10d. Again, both the Cosine and Euclidean metric returns the same results.

From the above results and plots, we can see how the Cosine and Euclidean distance metric has remained the most preferred metrics for the APPFD-FK-GMM retrieval method. In addition, where results returned by these metrics are exactly the same in all five evaluation metrics. In addition, we also see that although applying the PCA technique for dimensionality reduction on the final descriptor (FV) returned by FK, as in experimental run-10b and run-10d, the results are still remarkably close quantitatively and qualitatively, which is obvious in Figure 5.32.

Experiment 10: Results and Discussion, Comparing The APPFD-FK-GMM Method With The HAPPS-1 Method and Other State of The Art Methods On The SHREC'18 Protein Dataset

In this section, we compare and analyse the quantitative and qualitative results of the APPFD-FK-GMM: experimental run-10a (it could as well be run-10c, see Table 5.41) with the best overall results of the HAPPS-1 retrieval method (experimental run-1c, see Table 5.5), and HAPT4 [121] (which was the best overall state-of-the-art retrieval method for the SHREC'18 protein retrieval challenge). These comparative results are presented in Table 5.42. We remind the reader that all evaluation results presented in this thesis for the SHREC'18 protein dataset are done using the GT

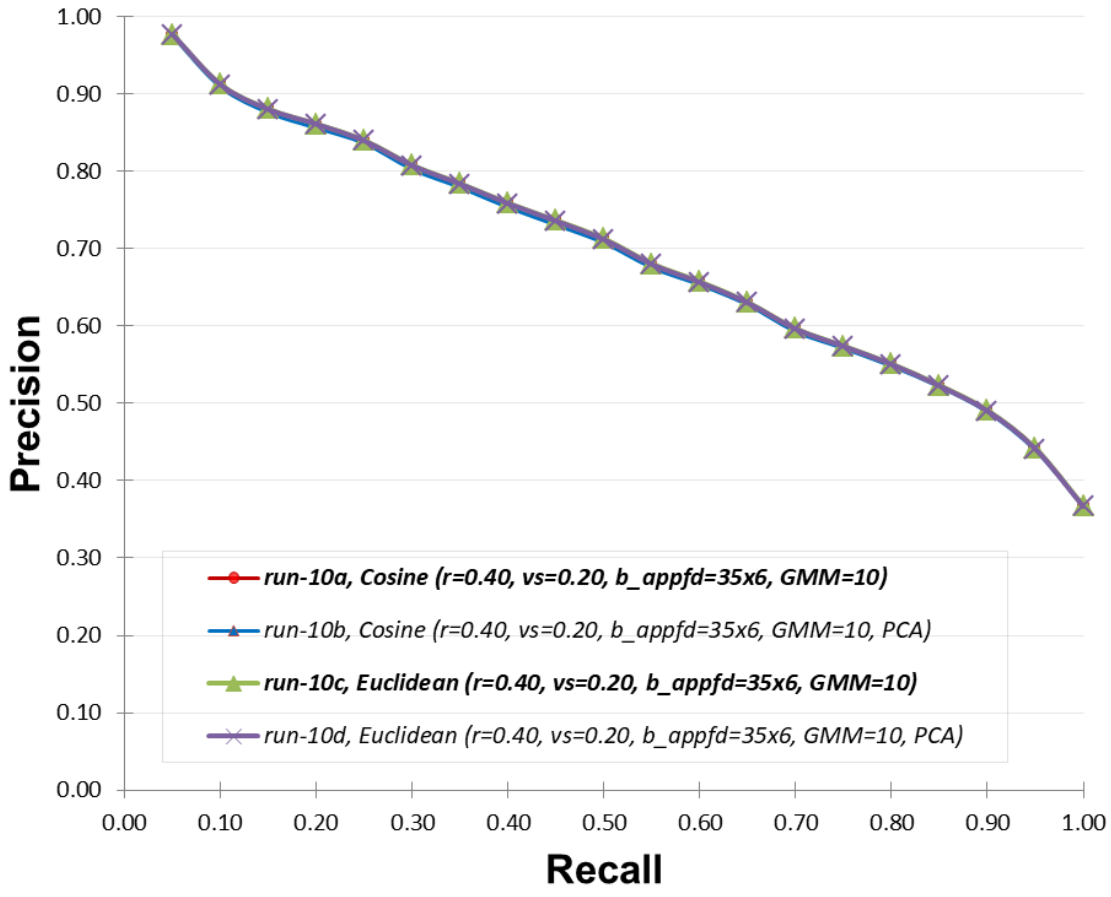


Figure 5.32: The PRC plots showing the quantitative retrieval performances of experimental run-10a to run-10d (see Table 5.41), using the APPFD-FK-GMM retrieval method on the SHREC'18 protein dataset. The plots show that there is no visible differences in the results of these four experimental runs, although quantitatively, the results of experimental run-10a and run-10c are a bit higher in values than those of run-10b and run-10d.

Experiments	Algorithms	NN	FT	ST	E	DCG
run-10a-Cosine	APPFD-FK-GMM	0.8334	0.6410	0.7785	0.5546	0.8655
run-1c-Cosine	**HAPPS-1**	**0.8525**	**0.6669**	**0.7815**	**0.5783**	**0.8818**
SHREC'18 [121]	HAPT4 [121]	0.7700	0.4930	0.5840	0.4620	0.7550

Table 5.42: Comparing the retrieval accuracies of the APPFD-FK-GMM method (experimental run-10a), HAPPS-1 (experimental run-1c), and HAPT4 [121] (the best state-of-the-art retrieval method for the SHREC'18 protein retrieval challenge/dataset). In this comparison, five standard quantitative evaluation metrics are used.

file provided by [121], for "All" classification level. Additional information is provided in [121], regarding all the methods who competed for the SHREC'18 protein retrieval task and their retrieval performances.

From the results in Table 5.42, although the HAPPS-1 retrieval method (experimental run-1c-Cosine) still takes the lead in terms of retrieval accuracies for all the five evaluation measures, the APPFD-FK-GMM method (experimental run10a-Cosine) still performs better in the second place and outperforms the overall best method (HAPT4) in [121]. Both experimental run-1c and run-10a uses the Cosine distance metric. Considering the parameter settings for these two experimental runs, we see that the parameter settings for experimental run-1c are $r = 0.50$, $vs = 0.20$, and $b_{(APPFD)} = 8$ (see Table 5.4), while those for experimental run-10a are $r = 0.40$, $vs = 0.20$, and $b_{(APPFD)} = 35$, including $G = 10$ (see Table 5.32). This reveals why the HAPPS-1 method outperforms the APPFD-FK-GMM method for the same dataset and retrieval challenge. By indication, given the same parameter settings for both experimental run-1c and run-10a, we are confident that the APPFD-FK-GMM method would certainly improve in overall retrieval accuracies, and outperform the HAPPS-1 method for the same retrieval task. We further investigate these assumptions in Section 5.5.4, using an even larger sized protein dataset from the SHREC'19 protein retrieval challenge.

5.5.4 Experiment 11: Evaluating The Retrieval Performances of The APPFD-FK-GMM Method On The SHREC'19 Protein Dataset

The SHREC'19 protein database, with 5,298 protein models is recent, and contains greater number of 3D protein models than those from previous protein shape retrieval challenges, including being twice as big as the number of 3D protein models in the SHREC'18 dataset. It therefore became necessary for us to also investigate the performances of our APPFD-FK-GMM retrieval method on such large dataset, given its retrieval performances on previous datasets. Similar to the approaches adopted for previous dataset evaluation with this method, several experimental runs are carried out, each with different parameter settings in order to determine which parameter combination of the APPFD-FK-GMM method is most suitable for this dataset. However, in this evaluation, we provide further analysis regarding two different experimental runs, instead of one. Considering the performance outcomes of previous experimental evaluations using this retrieval method, where we found that results of applying the PCA technique for dimensionality reduction of the final *fv* are negligibly lower, with overall similar retrieval performances to the results of instances where the PCA technique is not applied (see the results in Table 5.37 and Table 5.41, for example), we chose not to present all the results comparing the PCA-reduced *fv* with the other non-PCA approach in this analysis. However, in this experimental evaluation, both the Cosine and Euclidean metric returns nearly similar results in all five evaluation measures, but interestingly, the Euclidean metric returns overall best retrieval accuracies when the PCA is applied to reducing the final *fv* dimension. This evaluation is performed for both the *Protein* and *Species* classification levels (see Table 5.44).

Dataset:	SHREC'19 [123]	**Parameter Settings**					
Experiment 11	**Algorithms**	N	r	vs	$b_{(APPFD)}$	G	**Metric**
run-11a	APPFD-FK-GMM$_{(PCA)}$	3500	0.50	0.20	35×6	10	Euclidean
run-11b	APPFD-FK-GMM$_{(PCA)}$	3500	0.50	0.30	35×6	10	Euclidean

Table 5.43: Parameter settings for two different experimental runs with the APPFD-FK-GMM retrieval method on the SHREC'19 protein dataset.

In the first run (run-11a), we set the parameters to: $r = 0.50$, $vs = 0.20$, and $b_{(APPFD)} = 35$, including $G = 10$, while in the second run (run-11b) we set the parameters to $r = 0.50$, $vs = 0.30$, and $b_{(APPFD)} = 35$, including $G = 10$. These parameters settings are clearly presented in Table 5.43. Unlike experimental evaluations for the APPFD-FK-GMM retrieval method in Experiment 9 (see Section 5.5.2) and Experiment 10 (see Section 5.5.3), the motivation for increasing the r parameter from $r = 0.40$ to $r = 0.50$ in run-11a, and subsequently both the r and vs parameters in run-11b in this experimental evaluation is based on the observation from the comparative analysis recorded in Table 5.42, where the HAPPS-1 method (run-1c) with the parameter settings: $r = 0.50$ and $vs = 0.20$ outperformed the new APPFD-FK-GMM method with the parameter settings: $r = 0.40$ and $vs = 0.20$. The results of these experimental runs (Experiment 11) considering both the Protein and Species classification levels are provided in Table 5.44.

Experiment 11	**Algorithm(s)**	**NN**	**FT**	**ST**	**E**	**DCG**
PROTEINS	**LEVEL**					
run-11a$_{(Protein)}$	**APPFD-FK-GMM**	**0.9549**	**0.5346**	**0.6980**	**0.1242**	**0.8980**
run-11b$_{(Protein)}$	APPFD-FK-GMM	0.8766	0.4451	0.6260	0.1002	0.8622
SPECIES	**LEVEL**					
run-11a$_{(Species)}$	**APPFD-FK-GMM**	**0.9013**	**0.4832**	**0.6222**	**0.2110**	**0.8422**
run-11b$_{(Species)}$	APPFD-FK-GMM	0.7335	0.3721	0.5234	0.1493	0.7771

Table 5.44: Quantitative retrieval performance results using five standard evaluation metrics to evaluate APPFD-FK-GMM retrieval method on the SHREC'19 protein dataset, for experimental runs: run-11a and run-11b, for the *Protein* and *Species* classification levels. The results presented here are only from the Euclidean distance metric for both levels.

From the results in Table 5.44, there are some improved retrieval performances/accuracies for the APPFD-FK-GMM method, considering the large size of this dataset than in previous evaluations. The parameter settings for experimental run-11a produces the best overall results for both *Proteins* and *Species* classification levels. We notice that increasing the r and vs parameters (as in run-11b) causes performance drop of about 8% to 10% in all evaluation metrics, except the E measure. The PRC plots in Figure 5.33 and 5.34, also confirm these results. Notice that unlike the evaluations in experimental run-10a, where 4,500 samples were used for the APPFD-FK-GMM retrieval method, 3,500 samples are used for this evaluation with the SHREC'19

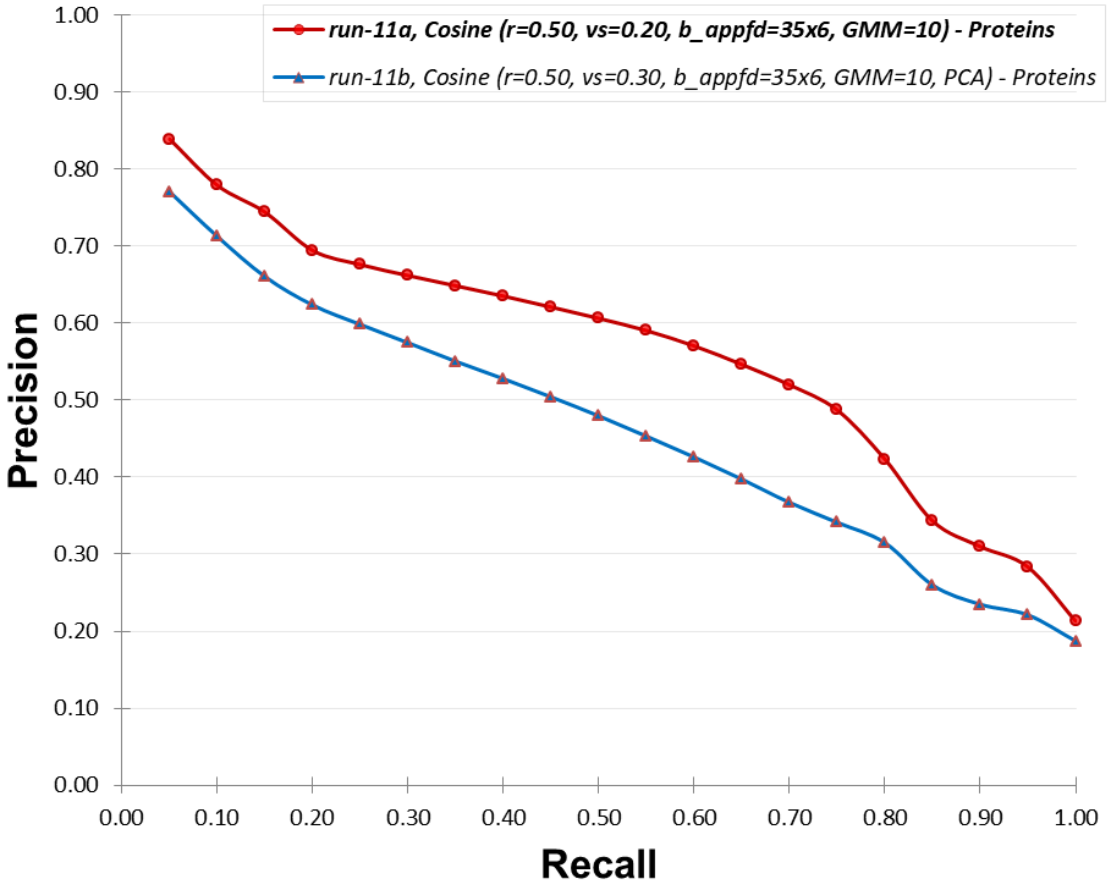


Figure 5.33: PRC plots of the retrieval results presented in Table 5.44 (run-11a and run-11b, *Proteins*) for the SHREC'19 protein shape retrieval challenge/dataset, using the APPFD-FK-GMM method.

dataset instead. This is as a result of the large size of this benchmark dataset, whereby sampling more points from the surface is capable of impacting computational cost (memory and time). However, it would be useful to keep the parameters consistent for a more direct comparison, since accuracy is more important and interesting to determine. In the next section, we would further compare the best overall retrieval results of the APPFD-FK-GMM retrieval method (run-11a) with that of the HAPPS-1 method and state-of-the-art method for the SHREC'19 protein dataset.

Experiment 11: Results and Discussion, Comparing The APPFD-FK-GMM Method With The HAPPS-1 Method and Other State of The Art Methods On The SHREC'19 Protein Dataset

Table 5.45 compares the retrieval results of three different retrieval methods (the APPFD-FK-GMM, HAPPS-1, and the 3DZD2 [120]) for the SHREC'19 protein dataset. We have already analysed the comparison between the HAPPS-1 and the 3DZD2 [120] retrieval methods in Section 5.3.5. Here, we are interested in seeing how the performances of the APPFD-FK-GMM compares with both the HAPPS-1 and state-of-the-art retrieval methods. Although that 3DZD2 method still maintains

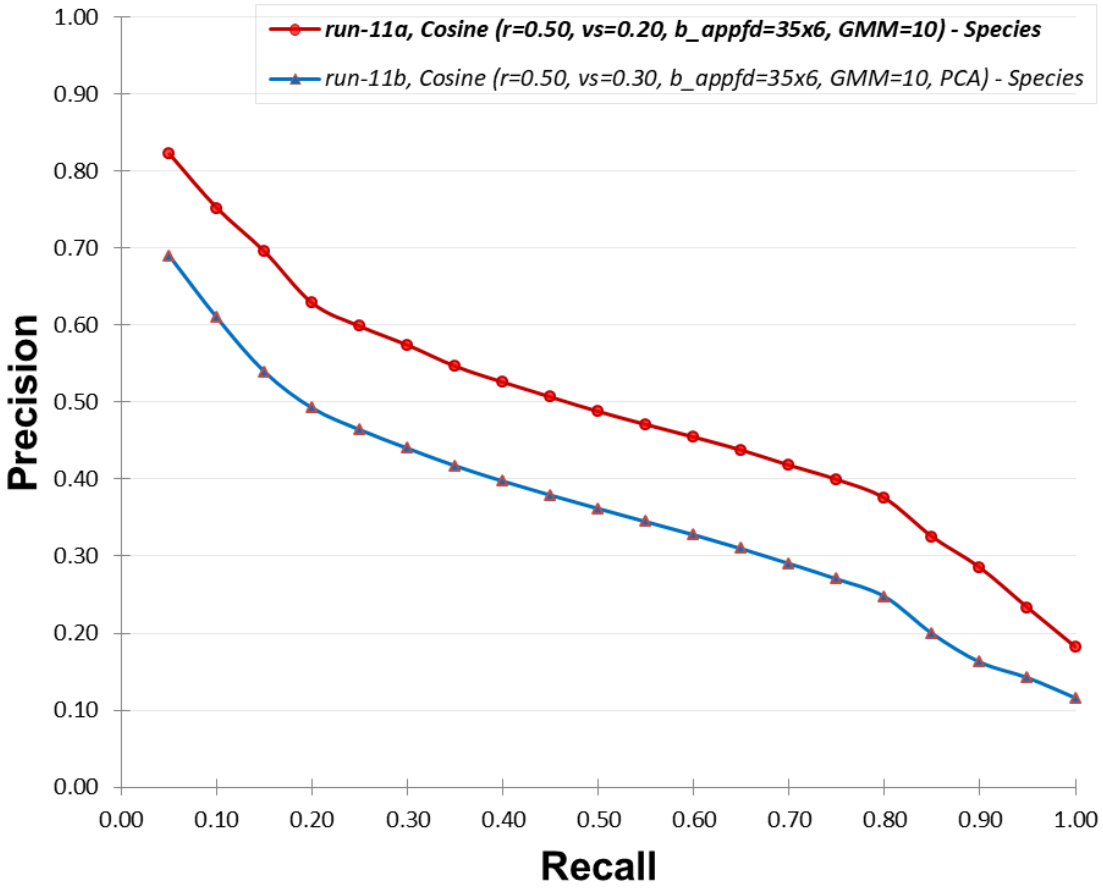


Figure 5.34: PRC plots of the retrieval results presented in Table 5.44 (run-11a and run-11b, *Species*) for the SHREC'19 protein shape retrieval challenge/dataset, using the APPFD-FK-GMM method.

the highest retrieval accuracies for this experimental evaluation, followed again, by the HAPPS-1 method, the performances of the current method is not far from that of the HAPPS-1 and the overall best state-of-the-art method (3DZD2) for this dataset (SHREC'19). This slight difference between the two of our methods is expected, given that a lower number of point samples (3,500) were used for the current method (APPFD-FK-GMM) as opposed to 4,500 which is the case with the HAPPS-1 method, and we think that these performances can further be improved at the expense of computational resources (memory and time), by sampling more points for feature extraction.

5.5.5 Experiment 12: Evaluating The Retrieval Performances of The APPFD-FK-GMM Method On The SHREC'20 Relief Dataset

Following the many success of our proposed retrieval methods (APPFD, HAPPS, and APPFD-FK-GMM) on raw 3D objects retrieval from several different benchmark datasets as presented in the previous sections of this thesis, we became interested in investigating the robustness of one of these methods on entirely different set

Experiments	Algorithm(s)	NN	FT	ST	E	DCG
PROTEINS	**LEVEL**					
run-11a$_{(Protein)}$	APPFD-FK-GMM	0.9549	0.5346	0.6980	0.1242	0.8980
run-5a$_{(Protein)}$	HAPPS-1	0.9853	0.5545	0.6976	0.1287	0.9075
SHREC'19 [120]	**3DZD2 [120]**	**0.9950**	**0.6620**	**0.7940**	**0.1310**	**0.9350**
SPECIES	**LEVEL**					
run-11a$_{(Species)}$	APPFD-FK-GMM	0.9013	0.4832	0.6222	0.2110	0.8422
run-5a$_{(Species)}$	HAPPS-1	0.9405	0.5071	0.6454	0.2241	0.8561
SHREC'19 [120]	**3DZD2 [120]**	**0.9650**	**0.5810**	**0.6940**	**0.2500**	**0.8840**

Table 5.45: Comparing the retrieval performances of three methods: the APPFD-FK-GMM, HAPPS-1 and best overall state-of-the-art method (3DZD2 in [120]), for the SHREC'19 protein dataset, using five quantitative evaluation metrics. These comparisons are done with the *Protein* and *Species* classification levels.

of retrieval challenges. As part of our research contributions to 3D Object Retrieval 2020 (3DOR'20), we submitted three experimental runs (the APPFD-FK(run1), APPFD-FK (run2) and APPFD-FK (run3), see [163]) of the APPFD-FK-GMM method for the retrieval of digital surfaces with similar geometric relief [163], where the retrieval performances of this method were evaluated and compared with several other state-of-the-art methods for this retrieval challenge. The dataset for this evaluation is very well described in Section 5.2.10 and in [163]. The goal of the contest is to verify the possibility of retrieving 3D models only based on the relief that are present on their surface and to compare methods that are suitable for this task [163]. Seven groups competed for this retrieval challenge where at least one retrieval method was submitted by each group. In general, a total of eight different methods and twenty experimental runs were submitted for evaluation.

Experiment 12: Results and Discussion, Comparing The APPFD-FK-GMM Method With Seven Other State of The Art Methods On The SHREC'20 Relief Dataset

A summary of the quantitative retrieval results comparing each of the best experimental runs of all the retrieval methods which competed for the SHREC'20 track on retrieval of digital surfaces with similar geometric relief [163] are shown in Table 5.46. The performance evaluation for this dataset uses seven evaluation metrics NN, FT, ST, mAP, nDCG, E, and AUC. Although the retrieval performance of our method (the APPFD-FK-GMM) for this retrieval challenge outperforms one of the state-of-the-art method (PointNet+SQFD [163]), it is disappointing to see, for the first time, that it recorded overall very low performances compared to other state-of-the-art methods. However, this outcome, in addition to several other performance results obtained for other 3D datasets, confirms part of our investigation regarding the suitability of our underlying method (APPFD) for certain shape retrieval tasks. This task, regarding experimentally evaluating our method in comparison with other state-of-the-art methods using the relief patterns dataset confirms the suitability of

the APPFD mainly for 3D objects with topological deformations as opposed to flat surfaces or surface patterns, due to the extremely poor performance accuracies recorded by the APPFD-FK-GMM approach. We refer the reader to [163], for an in-depth analyses of all methods and experimental runs for this retrieval challenge, including computational approach for each of the methods involved.

Methods	NN	FT	ST	mAP	nDCG	E	AUC
AFFFD-FK(run1) [163]	0.186	0.204	0.332	0.235	0.523	0.211	0.672
OH(run3) [163]	0.714	0.405	0.575	0.469	0.732	0.382	0.818
DFE(run1) [163]	**0.982**	**0.920**	**1.000**	**0.930**	**0.974**	**0.715**	**0.987**
DPML(run2) [163]	0.982	0.887	0.992	0.912	0.968	0.690	0.978
PointNet+SQFD(run3) [163]	0.173	0.119	0.225	0.190	0.470	0.137	0.605
SRNA(run2) [163]	0.923	0.494	0.683	0.563	0.811	0.453	0.882
MeshLBP [163]	0.905	0.671	0.832	0.726	0.884	0.570	0.909
kd-tree FLANN [163]	0.686	0.312	0.424	0.359	0.656	0.283	0.690

Table 5.46: Quantitative evaluation measures of only the best results of each methods presented in [163] (Experiment 12), including our APPFD-FK-GMM method for SHREC'20 geometric relief dataset. Seven evaluation metrics are used, and the two results highlighted in Gray performed poorly for this track.

From the results in Table 5.46 and the PRC plots in Figure 5.35, we see that the DFE, followed by the DPML methods produced the best overall performances for this retrieval task. These methods are based on a machine learning (i.e. data-driven) approach to pre-train a neural network and are characterised by sampling one representative surface patch during the pre-processing stage. Alternatively, methods, such as the MeshLBP and SRNA, including OH, which adopts the knowledge-based approach ranked better for this challenge. The MeshLBP and SRNA sampled multiple patches but seem to outperform other methods, such as the APPFD-FK and PointNet+SQDF which also applied the same sampling approach during processing. However, from these results, it seems that methods, such as the APPFD-FK which converts the 3D object into point clouds or that are based on convolutional neural networks trained on point clouds (PointNet) seem to be sub-optimal for this task. Probably these methods lose information on local details (for instance, the sampling process in the APPFD-FK focuses on the representation of the global geometry) and do not capture the subtle geometry and structure variations of local patterns and reliefs [163]. Essentially, these methods did not distinguish the model from the relief on it but treat them altogether. This means that the relief and models should be separated first and then the relief should be used for feature extraction and retrieval.

5.6 Experimental Evaluations of The HoGD Retrieval Method

Considering that the HoGD has contributed to improving the overall performances of the APPFD method resulting in the HAPPS method (see Sections 4.2.3 and 5.3), it is therefore important to also provide a separate experimental evaluation of the re-

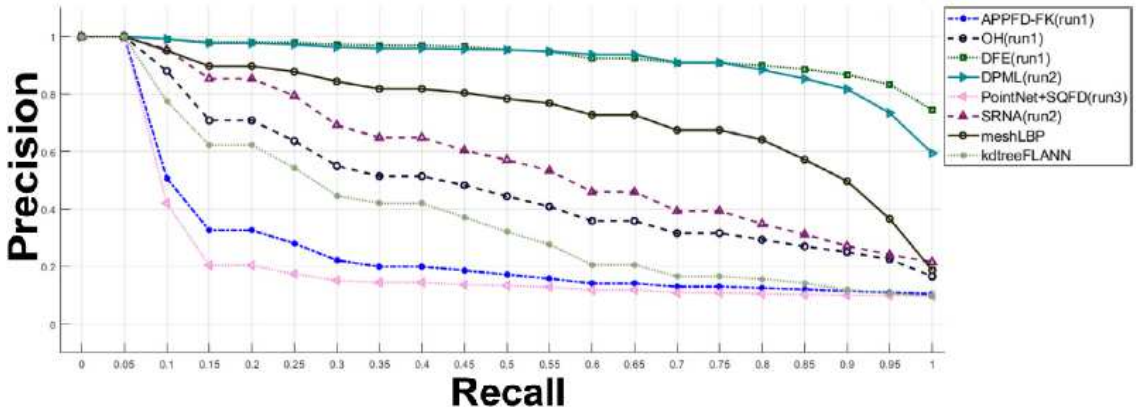


Figure 5.35: The PRC plots showing the quantitative retrieval performances of the best methods for the SHREC'20 surface relief dataset, as in Table 5.46) [163].

trieval performances of the HoGD method in conjunction with the HAPPS-1 method to assess HoGD's level of contributions via the performance differences between these two methods. In Section 4.2.2, we provided a succinct description of the HoGD *"global"* 3D retrieval method and its computation steps, including its parameterisation, such as the number of bins, b_{HoGD} and points sample, N used. Again, the implementation goal of the HoGD method is to further contribute to the overall retrieval performances of the APPFD *"local"* method, making the resultant *"hybrid"* descriptor more robust and suitable for a variety of 3D surface structure.

In this section, we provide some useful experimental evaluation results for the retrieval performances of our proposed *"global"* HoGD method and compare these results to those of the *"hybrid"* HAPPS-1 method. For these experiments, we consider the most recent SHREC 2021 benchmark dataset of 3D protein surfaces. However, the results in this section are not regarded as part of our major thesis contributions, but are presented to reveal the contributions of our proposed HoGD retrieval method to the APPFD method. These results comparison also helps to adjudge the HoGD's performance levels (or that of any other global 3D descriptor, such as the M2DP).

5.6.1 Experiment 13: Evaluating The Retrieval Performances of The HoGD Method On The SHREC'21 Protein Dataset

The SHREC'21 retrieval challenge for the retrieval and classification of protein surfaces equipped with physical and chemical properties [188] provide two variants of datasets (*"geometry"* and *"physichochemical"* data) for its retrieval track. Each of these data variants are further divided into the *"training"* and *"testing"* sets. The training data consists of 3,585 3D protein surfaces while the testing set consists of 1,543 3D protein surfaces.

While our research contributions to the retrieval and classification challenges in the SHREC'21 track [188] adopted the HAPPS-1 method for the *"geometry"* variant

of the datasets involving only the *"testing"* set, the HoGD and HAPPS-1 experimental evaluations in this section involve the *"training"* set of the *"geometry"* variant instead.

In line with our evaluation approach for this thesis, where several (dis)similarity metrics are applied during final descriptors matching, we present the experimental results of our HoGD retrieval method (evaluated against the SHREC'21 protein dataset) in Table 5.47. The number of points sample, $N = 4,500$ and the number of bins, $b_{HoGD} = 65$ for the quantitative results in Table 5.47, using the Cosine, Euclidean, and KLD metrics.

Experiment 13	Algorithms	NN	FT	ST	E	DCG
run-13a-Euclid	**HoGD**	**0.6483**	**0.2939**	**0.3845**	**0.2585**	**0.6176**
run-13b-Cosine	**HoGD**	**0.6469**	**0.2916**	**0.3823**	**0.2573**	**0.6165**
run-13c-KLD	**HoGD**	**0.6731**	**0.3100**	**0.3959**	**0.2700**	**0.6293**

Table 5.47: Quantitative retrieval performance results using five standard evaluation metrics to evaluate the HoGD retrieval method on the SHREC'21 protein dataset (*"Training"* set).

Table 5.47 present the quantitative retrieval performances of the HoGD method on the SHREC'21 protein dataset. The results in this table reveals near performances for both the Cosine and Euclidean metrics in all five quantitative statistics (NN, FT, ST, E, and DCG) as indicated by experimental run-10a and run-10b, while KLD metric returned the best overall performances for this experimental evaluation. Overall, poor retrieval performances (below 40%) have been recorded for the FT, ST, and E statistics with all three (dis)similarity metrics. Also, although the NN and DCG statistics record performance improvements over 60% in all three metrics, the overall performance of the HoGD method with the SHREC'21 protein dataset still leaves much to be desired. This is because this method only deals with the global surface structure of each protein surface and unable to characterise the local surface properties. In the next section, we would also perform similar experiment, this time, using the HAPPS-1 retrieval method which combines the HoGD with our local APPFD to investigate performance improvement.

5.6.2 Experiment 14: Evaluating The Retrieval Performances of The HAPP-1 Method On The SHREC'21 Protein Dataset

In this section, we present the quantitative performance results of the HAPPS-1 method on the SHREC'21 protein dataset (*"Training"* set) in order to examine the impact of combining the *"local"* APPFD with the *"global"* HoGD methods. The parameter settings adopted by the APPFD in HAPPS-1 method for Experiments 13 are as follows: $b_{APPFD} = 8$, $b_{HoGD} = 65$, $r = 0.40$, $vs = 0.20$. The quantitative performance evaluations for this experiment is summarised in Table 5.48.

Experiment 14	Algorithms	NN	FT	ST	E	DCG
run-14a-Euclid	**HAPPS-1**	**0.8611**	**0.6073**	**0.7520**	**0.5235**	**0.8510**
run-14b-Cosine	**HAPPS-1**	**0.8636**	**0.6133**	**0.7579**	**0.5282**	**0.8530**
run-14c-KLD	**HAPPS-1**	**0.8683**	**0.5729**	**0.6818**	**0.4874**	**0.8163**

Table 5.48: Quantitative retrieval performance results using standard evaluation metrics to evaluate the HAPPS-1 retrieval method on the SHREC'21 protein dataset (*"Training"* set).

Table 5.48 presents the quantitative performance evaluation of the HAPPS-1 method on the SHREC'21 protein dataset. For this experiment, it is obvious the the Cosine (dis)similarity metric returns the best overall retrieval scores on all evaluation statistics (FT, ST, E, and DCG), except with the NN statistic, where the KLD metric takes the lead. The Euclidean metric, which ranks second place, also performs better than the KLD in other evaluation statistics, but the NN. As usual, the evaluation performances of the NN and DCG statistics are better than the other three (i.e. FT, ST, and E). Interestingly, both the NN and DCG evaluation measure reveals of 85% performances for all metrics, except the KLD, and over 68% with the ST, over 60% with the FT, except with the KLD metric, and finally, over 50% performances with the E measure, except with the KLD metric. Considering the complexity and size of the SHREC'21 protein *"Training"* set, these results are impressive, but could further be improved by parameter adjustments. In the next section, we would provide succinct discussion which compares the retrieval performance evaluation of the HAPPS-1 method according to Table 5.48 with those of the HoGD method according to Table 5.47.

Experiment 14: Results & Discussion, Comparing Retrieval Performances Of The HoGD and HAPPS-1 Methods On SHREC'21 Protein Dataset

In this section, we provide succinct comparisons between the quantitative retrieval performances of Experiment 13 (using the HoGD method) to those of Experiment 14 (using the HAPPS-1 method). Both methods have separately been evaluated using the most recent SHREC'21 3D protein benchmark *"Training"* set.

As shown in Table 5.48 for each of the three different (dis)similarity metrics evaluated and the five evaluation statistics (NN, FT, ST, E, and DCG), the retrieval performances of the HAPPS-1 method are far greater than their counterparts from the HoGD method in Table 5.47. Since the HAPPS-1 method combines the APPFD and HoGD methods, comparing the results in Table 5.47 and Table 5.48 clearly reveals and demonstrate performance improvements to the HoGD method, when combined with the APPFD on a given dataset (in this case, the SHREC'21 protein dataset). These outcomes (performance evaluation comparisons) further support our position in this thesis (also see Section 4.2.3) that the overall retrieval performances of the local APPFD can further be improved by combining the global HoGD. In summary, the purpose of evaluating the HAPPS-1 retrieval method again in this section is only to provide fair comparisons of retrieval performances for the HoGD method.

5.7 Summary

In this chapter, we have presented the results of several experimental evaluations from all of our proposed methods (APPFD, HAPPS-1, HAPPS-2, APPFD-FK-GMM, and HoGD) on 3D shape retrieval, using at least ten different 3D benchmark datasets. We describe each of these retrieval methods in Section 4.2. The various retrieval challenged and/or datasets evaluated in this thesis are: (i) SHREC 2021: Retrieval and classification of protein surfaces equipped with physical and chemical properties [188, 187], (ii) SHREC 2020: Retrieval of digital surfaces with similar geometric reliefs [163, 231], (iii) SHREC 2020: Multi-domain protein shape retrieval challenge [122], (iv) SHREC 2019 Protein Shape Retrieval Contest [123, 61], (v) SHREC 2018 protein models [60], (vi) SHREC 2017: Point-cloud Shape Retrieval of Non-rigid Toys [142, 269], (vii) SHREC 2014 Track: Large Scale Comprehensive 3D Shape Retrieval (comprising of rigid and non-rigid 3D objects) [132], (viii) SHREC 2012 Track: Generic 3D Shape Retrieval (comprising of rigid and non-rigid 3D objects) [131], (ix) SHREC 2011 Track: Shape Retrieval on Non-rigid 3D Watertight Meshes [139], and finally, (x) the SHREC 2010 Track: Non-rigid 3D shape retrieval [140]. Each of these 3D benchmark dataset presents different levels of challenges to shape retrieval algorithms and further details regarding them are provided in Section 5.2.

The performance evaluation strategy used in this thesis involves testing the robustness of each of our proposed retrieval methods on at least two benchmark datasets, including comparing their retrieval accuracies (i.e. their overall performance results) with several other state-of-the-art methods applicable to the respective dataset and retrieval challenge. We achieve these by mainly using the following standard IR quantitative evaluation metrics: NN, FT, ST, E, DCG, and PRC, including the mAP, nDCG, and AUC, for example, to evaluate the performance of our methods in comparison with others (see Table 5.46).

First, in Section 5.3, we evaluate the retrieval performances of two variants of the HAPPS retrieval method tagged: the HAPPS-1 and HAPPS-2 (see Section 4.2.3) on six different 3D retrieval problems, each associated with a unique collection of 3D benchmark data. In Experiment 1 (section 5.3.1), the performance accuracies and robustness of the HAPPS-1 method were evaluated using the SHREC'18 protein shape database which contains 2,267 models. In Experiment 2 (section 5.3.2), we evaluate this method on the SHREC'17 dataset, consisting 100 raw point cloud of non-rigid toys, followed by Experiment 3 (section 5.3.3), where 200 non-rigid and watertight 3D triangular meshes from the SHREC'10 benchmark were used to evaluate the performances of the HAPPS-1 method. The SHREC'11 benchmark dataset contains three times the number of 3D models in the previous year (i.e. 600 watertight and non-rigid triangular meshes), and we evaluated our HAPPS-1 method on this dataset in Experiment 4 (section 5.3.4). In Experiment 5 (section 5.3.5), we adopted the SHREC'19 protein dataset, with 5,298 protein models (which is much more than those in the previous protein shape retrieval tracks) for our experimental evaluation of the HAPPS-1 method, following its success in the SHREC'18 dataset. Finally, Experiment 6 (section 5.3.6) further investigates the performance abilities of the HAPPS retrieval methods by using the HAPPS-1 method, including intro-

ducing another variant, the (HAPPS-2), evaluated on the SHREC'20 protein shape dataset. This dataset has 588 models. A review of the overall retrieval performance summary of the HAPPS method is presented in Section 6.3.1.

Secondly, in separate experimental investigations discussed in Section 5.4, we examined the robustness and retrieval performances of our baseline retrieval method, the APPFD using the SHREC'12 generic 3D shape dataset, which consists of 1,200 mixed (rigid and non-rigid) 3D triangular meshes characterised by lots of defects. The evaluation results are reported in Experiment 7 (Section 5.4.1). In addition, Experiment 8 (Section 5.4.2) presents and analyses the evaluation results of the APPFD method with the SHREC'14 large scale comprehensive shape benchmark dataset, which contains a total of 8,987 triangular meshes. A review of the overall retrieval performance summary of the APPFD method is presented in Section 6.3.2.

Finally, we consider the importance of validating the retrieval performances of some of our proposed methods against the most recent datasets including a comparative analysis of their performances with the most recent state-of-the-art methods for 3D shape retrieval. Therefore, in Section 5.5 we adopted four of the most recent 3D shape benchmark datasets from SHREC, for such performance evaluation, using our most recent retrieval method, the APPFD-FK-GMM, where we examined performances, robustness and compactness of the final shape descriptor returned, in line with the objectives of this research (see Section 1.3 and Section 2.3.1). The outcomes of the APPFD-FK-GMM method are then compared side-by-side with the earlier method (HAPPS-1), including several other state-of-the-art methods which were also evaluated on these four datasets. In Experiment 9 (Section 5.5.2) the SHREC'17 PRoNTo dataset was considered for this evaluation. In Experiment 10 (Section 5.5.3), we adopted the SHREC'18 protein dataset, and the SHREC'19 protein dataset for Experiment 11 (Section 5.5.4). We decided to try a completely unique 3D retrieval challenge in Experiment 12 (Section 5.5.5), where the SHREC'20 3D surface relief dataset was used instead. A review of the overall retrieval performance summary of the APPFD-FK-GMM method is presented in Section 6.3.3. In addition to the above, we have also actively contributed to the latest 3D protein surface retrieval challenges (the SHREC'21 Track: Retrieval and classification of protein surfaces equipped with physical and chemical properties) where the comparative performance of its *"Testing"* dataset is published in [188], while the performances with its *"Training"* dataset has been experimentally evaluated in Section 5.6.

Chapter 6

CONCLUSION AND FUTURE WORK

The development of a concise, robust and computationally efficient 3D representation, including a reliable technique for determining the (dis)similarity between 3D objects remain common problems in 3D objects recognition, classification, detection and retrieval tasks, especially when we consider the fast rate at which 3D models and domain-specific 3D databases are emerging. Although several methods have been developed for 3D shape representation and matching, a number of open problems associated with 3D shape matching and retrieval are yet to be satisfactorily addressed, such as finding concise 3D shape representation. In addition, majority of the existing 3D shape retrieval algorithms and matching techniques are either too complicated, computationally prohibitive, or not robust enough to accurately describe 3D objects, and most of these methods also are not widely applicable nor able to generalise across diverse range of retrieval challenges. These constitutes a *"3D shape search"* problem.

This thesis explored different issues associated with the challenges of developing a highly effective (i.e. concise, robust, and computationally efficient) 3D objects representation to facilitate 3D shape description, indexing, matching and retrieval. In this chapter, overall summaries of all the research work in this thesis are provided, the most important results are emphasised, followed by a summary of our research contributions, overview of findings, as well as limitations and strength of this thesis. In addition, a concise summary of retrieval performances of each of our contributions is presented. Finally, we conclude the thesis by highlighting what has been learned from this thesis and provide recommendations for further research direction.

6.1 Overall Thesis Review

The *"3D Shape search"* problem for additive manufacturing (3D repository and the rapidly emerging domain-specific 3D benchmark datasets) involves computing highly effective mathematical representations of 3D objects to facilitate the searching and retrieval of these objects from their respective database or repository. Motivated by the need to address the *"3D Shape search"* problem, in addition to the wide application areas of 3D shape retrieval methods and associated techniques, our main goal in this research was to learn from existing techniques and develop a new set

of improved 3D shape retrieval methods (descriptors) for rigid and non-rigid 3D meshes and point clouds retrieval. To achieve these, we first examined two broad approaches to 3D shape retrieval: the data-driven and knowledge-based approaches (see Section 2.5) and adopted the knowledge-based approach considering that it does not require large training dataset or high computing resources to succeed, then we focused on the statistically-based technique to 3D shape description due to its popularity, success, and ease of implementation (see Section 2.6). In particular, we have focused on 3D shape retrieval methods to address the challenges of 3D objects retrieval, and the contribution of this thesis is four-fold as summarised in the next section.

6.2 Review of Thesis Contributions

Storage is a huge concern for big and unstructured data. Majority of 3D models, including the ones evaluated in this thesis are represented with hundreds of thousands of points (other surfaces, upto millions of points) which presents serious challenges for computing resources (memory and time). One of our major thesis contributions is the development of 3D shape retrieval methods (HoGD, APPFD, HAPPS, and APPFD-FK-GMM) that are capable of robustly and accurately describing 3D objects with very minimal points sample (i.e. $3,500$ to $4,500$) representing their surface. Essentially, all our proposed methods reduce the representation of 3D triangular meshes from $200,000+$ vertices, $95,000+$ faces to only about $4,500$ points sample and further provide a compact representation for these surfaces in a way that supports very efficient indexing and matching, for 3D shape retrieval and classification tasks.

We also highlight as major contributions of this thesis, the development and application of above-mentioned four novel and efficient 3D point cloud or triangular mesh surface representations, which are: (i) the *global* HoGD, (ii) the *local* APPFD, (iii) the *hybrid* HAPPS, and (iv) the *global* APPFD-FK-GMM, each of which addresses different aspects of the 3D *Shape search* problem.

6.2.1 Contributions by The Local APPFD Method

An overview of the APPFD computation approach is presented in Figure 4.6. Although the core features extraction technique presented in this implementation is inspired by [252, 136] and [198], our method addresses the limitations with these approaches, thus: (i) The PPF technique by previous work [252, 136, 198] completely depends upon the estimated surface normals, whereby any imprecise normal estimation may produce low-quality descriptors. In order to compensate for this, we propose a novel 2-dimensional local angular feature, ϕ and θ, which is independent of the normal vector, and used to augment the PPF technique to improve its overall robustness and accuracy. (ii) The APPFD descriptor is computed only on selected surface region (i.e. the LSP) around key points of a given 3D surface, rather than the entire surface, and describes local surfaces with very little number of points (as low as 3,500 points), which is an improvement over all other methods. (iii) We adopt the voxel-grid down-sampling technique instead (which is more intuitive and accurate) as a better alternative to other 3D key point detection technique such

as Harris [220, 58] (which, on the other hand, is unreliable for most 3D objects as revealed in Figure 3.17, where wrong, inconsistent, and non-repeatable key points are returned). We show in Figures 3.14 and 3.15 that the down-sampled points are capable of correctly representing the entire geometry of the sampled surface (see Section 3.3.5).

6.2.2 Contribution by The Global HoGD Method

We introduce a new global descriptor, the HoGD described in Section 4.2.2, which is one of the simplest, most intuitive, compact, and computationally efficient description of the global structure of 3D object. It does not depend on surface normals or feature extraction, hence, computes extremely fast and eliminates the extra processing time which is demanded by surface normals estimation. The key advantage of this descriptor is that it can directly be computed from raw mesh vertices or points in point cloud, without initial feature extraction steps (see Figure 4.7). However, although its robustness is directly proportional to the number, N of vertices or points, where higher values of N would improve its capability, it is still able to provide meaningful contribution to the APPFD with as low as $N = 3500, 4500$. Retrieval performances evaluation of the HoGD method are provided in Section 5.6.

Considering that the local APPFD is most suitable for non-rigid objects, the HoGD provides a compact and efficient global shape representation to extend the capabilities of the APPFD in dealing with full rigid 3D objects, especially in heterogeneous dataset which contains both rigid and non-rigid 3D objects, such as the SHREC'14 dataset and datasets with complex surface structure, such as the protein retrieval datasets (see Section 5.2). The combination of these two local and global descriptor yields the HAPPS which is generalisable across different 3D benchmark datasets.

6.2.3 Contributions by The Hybrid HAPPS Method

The HAPPS (see Section 4.2.3) is a hybrid 3D shape descriptor which rely on the descriptive power of the local APPFD and the HoGD to provide reliable signature for both rigid and non-rigid 3D objects, and generalises well across several benchmark datasets. The HAPPS retrieval method provides an improvement over the retrieval abilities of the individual APPFD or HoGD that are combined to form it. We demonstrate this through the experimental evaluations in Section 5.6. Here, the HAPPS-1 method only compare its retrieval performances with those of the HoGD method, using the *"training"* set of the SHREC'21 3D protein benchmark dataset. However, we refer the reader to [188] for a comprehensive retrieval and classification performances of the HAPPS-1 method, where its performances are compared to the performances of several other state-of-the-art methods, but using the *"testing"* set, instead. More importantly, we propose the HAPPS as a framework which allows a combination of the descriptive power of the APPFD with any other global shape descriptor in order to address certain 3D retrieval challenges and improve overall retrieval performance (see Figure 4.8). In addition, Table 6.1 and Table 6.2 provide

SHREC'18	**PROBLEM:**	**TABLE 5.5**	**(Experiment 1)**
State-Of-The-Art (SOTA) Methods [121]	**SOTA Performance Rankings**	**HAPPS-1 Performance Ranking**	**Remark**
· HAPT4 · WKS · 3D FusionNet · GSGW	2^{nd} 3^{rd} 4^{th} 5^{th}	1^{st}	HAPPS-1 method out-performs every other state-of-the-art methods for the SHREC'18 retrieval problem. This ranking is based on the individual performances (considering the evaluation statistics: NN, FT, ST, E, DCG) of each retrieval method.
SHREC'17	**PROBLEM:**	**TABLE 5.9**	**(Experiment 2)**
State-Of-The-Art (SOTA) Methods [142]	**SOTA Performance Rankings**	**HAPPS-1 Performance Ranking**	**Remark**
· BoW-RoPS-DMF-3 · BPHAPT · CDSPF · SnapNet	1^{st} 2^{nd} 4^{th} 5^{th}	2^{nd} and 3^{rd}	HAPPS-1 method ranks 2^{nd} overall position with the NN statistic and 3^{rd} in other statistics, against thirty-five other state-of-the-art methods from eight authors, for the SHREC'17 retrieval problem. This ranking is based on the individual performances (considering the evaluation statistics: NN, FT, ST, E, DCG) of each retrieval method.
SHREC'10	**PROBLEM:**	**TABLE 5.10**	**(Experiment 3)**
State-Of-The-Art (SOTA) Methods [140]	**SOTA Performance Rankings**	**HAPPS-1 Performance Ranking**	**Remark**
· MR-BF-DSIFT-E · DMEVD_Run1 · SQFD(WKS)	1^{st} 2^{nd} 4^{th}	1^{st} and 3^{rd}	HAPPS-1 method ranks 1^{st} overall position with the NN statistic and 3^{rd} place in other statistics, against three other state-of-the-art methods for the SHREC'10 retrieval problem. This ranking is based on the individual performances (considering the evaluation statistics: NN, FT, ST, E, DCG) of each retrieval method.
SHREC'11	**PROBLEM:**	**TABLE 5.15**	**(Experiment 4)**
State-Of-The-Art (SOTA) Methods [139]	**SOTA Performance Rankings**	**HAPPS-1 Performance Ranking**	**Remark**
· SD-GDM-meshSIFT · MDS-CM-BOF_Run1 · OrigM-n12-normA · MLSF · BOGH · FOG+MRR	1^{st} 2^{nd} 3^{rd} 4^{th} 5^{th} 6^{th}	3^{rd} and 7^{th}	HAPPS-1 method ranks 3^{rd} in NN statistic and 7^{th} overall position among nine other state-of-the-art methods for the SHREC'11 retrieval problem. This ranking is based on the individual performances (considering the evaluation statistics: NN, FT, ST, E, DCG) of each retrieval method.
SHREC'19	**PROBLEM:**	**TABLES 5.22 & 5.23**	**(Experiment 5)**
State-Of-The-Art (SOTA) Methods [123]	**SOTA Performance Rankings**	**HAPPS-1 Performance Ranking**	**Remark**
· 3DZD2 · HAPT2 · GASD · ConvLDSNet2	1^{st} 2^{nd} 4^{th} 5^{th}	3^{rd}	HAPPS-1 method ranks third overall against four other state-of-the-art methods for the SHREC'19 protein retrieval problem. This ranking is based on the individual performances (considering the evaluation statistics: NN, FT, ST, E, DCG) of each retrieval method.

Table 6.1: Summaries of the retrieval performance rankings of the HAPPS-1 method compared to other top-most performing state-of-the-art methods on various 3D benchmark datasets (retrieval challenges). These summaries reveal how the performances of our proposed method compares to other state-of-the-art methods for each dataset or retrieval challenge.

SHREC'20	**PROBLEM:**	**TABLE 5.24 & 5.26**	**(Experiment 6)**
State-Of-The-Art (SOTA) Methods [122]	**SOTA Performance Rankings**	**HAPPS Performance Ranking**	**Remark**
· WKS/SGWS · 3DZD/3DZM · HAPT · GraphCNN	1^{st} 2^{nd} 4^{th} 5^{th}	2^{nd} and 3^{rd}	HAPPS ranks 2^{nd} with the NN statistic (Protein level classification) and 3^{rd} overall positions (Protein/Species level classifications) against five other state-of-the-art methods for the SHREC'20 protein retrieval problem. This ranking is based on the individual performances (considering the evaluation statistics: NN, FT, ST, mAP) of each retrieval method.
SHREC'21	**PROBLEM:**	**SOURCE [188]**	
State-Of-The-Art (SOTA) Methods [188]	**SOTA Performance Rankings**	**HAPPS-1 Performance Ranking**	**Remark**
· JHCA_HAPT · 3DZD · GLoFe · MPGCNNs	1^{st} 2^{nd} 4^{th} 5^{th}	2^{nd} and 3^{rd}	HAPPS-1 method competes very closely with the highest-ranking method, ranks 2^{nd} and 3^{rd} overall positions in several other evaluation statistics, against four other state-of-the-art methods, for the SHREC'21 protein retrieval problem. This ranking is based on the individual performances (considering the evaluation statistics: NN, FT, ST, E, DCG, mAP, and many others) of each retrieval method.

Table 6.2: Summaries of the retrieval performance rankings of the HAPPS-1 method compared to other top-most performing state-of-the-art methods on the SHREC'20 and SHREC'21 3D proteins benchmark datasets (retrieval challenges). These summaries reveal how the performances of our proposed method compares to other state-of-the-art methods for each dataset or retrieval challenge.

concise summaries of the overall retrieval performance rankings of our proposed HAPPS method in comparison to several other state-of-the-art (SOTA) methods for each of the dataset (shape retrieval challenge) that these methods have separately been evaluated against. Finally, in Section 6.3.1, we also provide summarised discussions of the retrieval performances of the HAPPS method.

SHREC'12	**PROBLEM:**	**TABLE 5.29**	**(Experiment 7)**
State-Of-The-Art (SOTA) Methods [131]	**SOTA Performance Rankings**	**APPFD Performance Ranking**	**Remark**
· DG1SIFT · DVD+DB+GMR · ZFDR · 3DSP-L2-1000-hik · LSD-sum	1^{st} 2^{nd} 3^{rd} 4^{th} 6^{th}	5^{th}	APPFD ranks 5^{th} overall position with five other state-of-the-art methods for the SHREC'12 3D shape retrieval problem. This ranking is based on the individual performances (considering the evaluation statistics: NN, FT, ST, E, DCG) of each retrieval method.
SHREC'14	**PROBLEM:**	**TABLE 5.34**	**(Experiment 8)**
State-Of-The-Art (SOTA) Methods [132]	**SOTA Performance Rankings**	**APPFD Performance Ranking**	**Remark**
· LCDR-DBSVC · MR-D1SIFT · ZFDR · DBNAA_DERE · KVLAD	1^{st} 2^{nd} 3^{rd} 4^{th} 6^{th}	5^{th}	APPFD ranks 5^{th} overall position again, compared to five other state-of-the-art methods for the SHREC'14 retrieval problem. This ranking is based on the individual performances (considering the evaluation statistics: NN, FT, ST, E, DCG) of each retrieval method.

Table 6.3: Summaries of the retrieval performance rankings of the APPFD method compared to other top-most performing state-of-the-art methods on the SHREC'12 and SHREC'14 3D benchmark datasets (retrieval challenges). These summaries reveal how the performances of our proposed method compares to other state-of-the-art methods for each dataset or retrieval challenge.

6.2.4 Contributions by The Global APPFD-FK Method

It is important to mention again that the APPFD is applicable to both the partial and complete shapes, while the HoGD is applicable to the complete shapes only. Although a combination of these descriptors demonstrated excellent retrieval accuracies through experimental evaluations, there are concerns regarding the extremely high-dimensional final *fv* by the APPFD, where for our 6-dimensional local APPF and the parameter $b_{(APPFD)} = 8$, we get $8^6 = 262,144$-dimensional *fv*. While such *fv* is capable of robustly representing a partial or full 3D object, it is very high-dimensional. The following contributions are made to tackle this problem: (i) We propose the APPFD-FK-GMM framework (see Section 4.2.4 and Figure 4.9) to produce a more compact (i.e. very low-dimensional) global representation of 3D objects, using either the APPFD or HAPPS, and still obtain excellent retrieval accuracies. (ii) This representation utilises the FK [218], which is a robust technique that encodes (agglomerates) independent sets of variable-sized local descriptors into a fixed-sized vector representation for each input 3D object. The outcome, therefore, is a global shape descriptor (more compact and stable), which is a significant

improvement over either of the local APPFD or hybrid HAPPS method. (iii) With linear dimensionality reduction technique (PCA) of derived FV from FK which uses GMM-trained local APPFD, we are able to effectively describe a 3D surface with as low as 162-dimensional *fv* (see Section 4.2.4). Experimental results demonstrate how the retrieval accuracies of this method outperforms and rivals most state-of-the-art methods, including the HAPPS method (see results in Section 5.5.2, for example), (iv) Through experimental evaluations, we demonstrate the suitability of the L_2 norm for matching the APPFD-FK-GMM signature, (v) The final outcome of implementing the APPFD-FK-GMM framework is an improvement to the already robust local APPFD and hybrid HAPPS retrieval methods.

Generally, through experimental evaluations, we independently tested each of our proposed 3D shape retrieval methods in this thesis (APPFD, HAPPS-1&2, and APPFD-FK-GMM) across diverse range of 3D retrieval domains and benchmark datasets, each of which contains considerably very large number of 3D objects and present unique levels of retrieval challenges/difficulties to shape retrieval algorithms. Experimental results demonstrate the robustness and generality of our proposed methods across the different datasets which involve rigid, non-rigid, watertight, and non-watertight 3D meshes and point clouds, where they perform generally well, indicating their tendencies for excellent performances in other 3D computer vision tasks. This is in line with the characteristics of shape descriptors outlined in Section 2.3.1. The APPFD method is the basis for all other retrieval methods we propose in this thesis. This method is characterised by four key parameters, which are: N, r, vs, and $b_{(APPFD)}$ (see Section 5.3.1). We can see from the different experimental evaluations in Chapter 4 (e.g. the experimental evaluations in Section 5.3) that the values of each of these parameter settings are within *close* range for the different datasets or retrieval challenges. Considering that we adopt the knowledge-based approach, which uses hand-crafted feature-extraction techniques, the end results of our methods (see Chapter 4) rivals with most state-of-the-art methods which adopted the data-driven approach, instead.

6.2.5 Summary

Experimental results have revealed that for each of the retrieval methods we propose in this thesis, sampling very little number of points (between 3,500 to 4,500) from the surface of 3D objects is adequate to effectively describe these surfaces to produce robust signatures which can be used to match the objects with impressive retrieval accuracies, for all the benchmark datasets we have evaluated. However, considering the experimental evaluations of the HAPPS method with the SHREC'20 protein dataset (see Section 5.3.5), where only 3,500 points were sampled due to the large size of the dataset. Although the results returned were impressive, other experimental runs/evaluations, such as with the SHREC'18 protein dataset where the points samples are increased to 4,500 points produced even better and higher retrieval accuracies (see Section 5.3.1. Therefore, we expect that a further increase in the number of points samples, N (while adopting the suitable parameter settings) would further improve the overall accuracies of our shape retrieval methods. Note that there would be a trade-off between retrieval accuracies and computational cost. Accordingly, the retrieval accuracies recorded for the SHREC'19 dataset in Section 5.3.5 could fur-

ther be improved by increasing the parameter, N from 3,500 to 4,500 or more.

The proposed 3D shape descriptors in this thesis can easily be integrated into building a robust search engine (3D-CBRS). For an overall 3D shape retrieval pipeline (see Figure 2.3 in Section 2.2.4), it is important to identify and adopt both shape description and matching techniques whose combination produces results that supports efficiency and accuracy at the same time. For example, if the final retrieval accuracies score for an efficient technique combination is slightly lower than that of an inefficient technique, then it might be worth adopting the former technique to mitigate computational costs. Matching our final shape descriptors using different distance metrics (see Section 2.2.3) enable us to determine the best viable alternative matching technique which is suitable for the descriptor. This is important for a 3D shape retrieval system. Through experimental evaluations, we have investigated the suitability of several (dis)similarity metrics and matching methods for our proposed shape retrieval methods.

In our research implementations, all the methods we propose in this thesis rely on point clouds generated from meshes or raw point clouds input. Therefore, if the inputs to our retrieval algorithm are given as triangular meshes, the technique in Section 3.3 are first applied to convert an input 3D mesh to point cloud and its corresponding surface normals. This process is illustrated in Figure 3.8. However, if the input to our algorithm is given as raw point cloud data, as in SHREC'17 PRoNTo dataset [142], we instead adopt the technique described in Section 3.3.1 to estimate their corresponding normal vectors. Since most 3D datasets are presented as a triangular mesh, we first sample N points from the sur(face) of mesh to return the equal number of points for all 3D shapes in the database. The points sampling techniques we use are described in Section 3.2 and 3.3. In conclusion, we adopt point cloud representation of meshes or raw point clouds of shapes for all datasets used in this study.

6.3 Summary of The Retrieval Performances of Our Proposed Methods

In Chapter 5, we presented and discussed the qualitative and quantitative retrieval performances of each of our proposed methods, involving up to ten SHREC benchmark datasets. We summarised the outcome of those experimental evaluations in Section 5.7. The performance evaluation strategy we used involves testing the robustness of each of our proposed methods on at least two datasets, including comparing their overall performance results with several other state-of-the-art methods which are applicable to the same dataset and retrieval challenge. We considered the following standard IR quantitative evaluation metrics: NN, FT, ST, E, DCG, PRC, including the mAP, nDCG, and AUC metrics. This section provides a summary of the retrieval performances (results) obtained in Chapter 4, regarding each of the methods proposed in this thesis.

6.3.1 Summary of The Retrieval Performances of The HAPPS Method

There are 2,267 protein models in the SHREC'18 protein benchmark dataset (see Section 5.2.7). Interestingly, the HAPPS-1 method demonstrates superiority over all the other six state-of-the-art methods that competed for that retrieval challenge which are - 3D-FusionNet, HAPT4, SIWKS, DEM, WKS, and GSGW [121] (see Section 5.3). Our evaluation was done using the *"All"* classification and evaluation category which is the only publicly available GT provided by the track organisers, and we show the quantitative results of this evaluation in Table 5.5. In Section 5.3.2, we tested the HAPPS-1 method on the SHREC'17 PRoNTo dataset, which contains 100 raw point cloud 3D models. We show the quantitative performance results of the HAPPS-1 compared to only the best results of several other state-of-the-art methods in Table 5.9. These results reveal that the HAPPS-1 method ranks 3rd place, out of 9 methods and demonstrates its superiority over six other state-of-the-art methods applicable to this dataset and retrieval challenge, which are - SnapNet, m3DSH-3, MFLO-F-IWKS, SQDF(wks), CDSPF and AlphaVol1 methods in [142]. However, the overall qualitative performance of the HAPPS-1 is not too far from those of the BoW-RoPS-DMF-3 and BPHAPT methods (see Figure 5.16) which performed the highest for this retrieval challenge and dataset.

The overall retrieval performance of the HAPPS method is excellent on all the datasets evaluated against this method. Table 6.1 and Table 6.2 provide overall retrieval performance rankings of the HAPPS method, in comparison to several other state-of-the-art methods for various 3D shape retrieval challenges (datasets). In addition, among all the proposed methods in this thesis, the HAPPS method records the highest overall performances, followed by our latest and improved method, the APPFD-FK-GMM. Essentially, the retrieval performances of the HAPPS method has been tested/evaluated on six different benchmark datasets, which are the SHREC'10, 11, 17, 18, 19 and SHREC'20 protein datasets (see Section 5.2 for more details regarding these datasets). For the SHREC'10 dataset evaluation (see Section 5.3.3), the best experimental run (run-3a) of the HAPPS-1 method rivals with the best overall state-of-the-art method (DMEVD_Run1 in [140]) and outperforms the other two methods (MR-BF-DSIFT-E and SQDF(wks) in [140]) which are applicable to the retrieval challenge/dataset - see Table 5.10. Secondly, regarding the performance evaluation of the HAPPS-1 method on the SHREC'11 retrieval challenge and dataset (see Section 5.3.4), results of the best experimental run of the HAPPS-1 method (run-4a) rivals with the results of three other state-of-the-art methods, which are the FOG+MRR, BOGH, and MLSF methods in [139]. In addition, our results outperforms the T-NoNorm-40Coef, HKS and PatchBOF_150 methods in [139], while the SD-GDM-meshSIFT and OrigM-n12-normA in [139] had the highest overall performance scores above the HAPPS-1, as presented in Table 5.15.

Evaluating the HAPPS-1 method on the SHREC'19 protein retrieval challenge, which involves 5,298 protein models, two classification or GT files were considered, which are the *Proteins* and the *Species* classification levels (see Section 5.3.5). Experimental results demonstrate the superiority of our method with the best experimental run (run-5a) over the results of two other state-of-the-art methods (Con-

vLDSNet2 and GASD in [123]), which also performs very close to the results of two other methods (3DZD2 and HAPT2 in [123]) in terms of quantitative performances as revealed in Table 5.22 and Table 5.23 for the *Proteins* and *Species* level classifications, respectively. Finally, the retrieval performances of the HAPPS-1 and HAPPS-2 methods were examined on the SHREC'20 protein dataset, which contains 588 protein models, using two different GT or classification levels, the *Protein* and *Species* levels (see Section 5.2.9). Experimental results (see Section 5.3.6) demonstrate the superiority of the HAPPS-1 and HAPPS-2 methods over four other state-of-the-art methods, which are: CODSEQ1&CODSEQ2, 3DZD&3DZM, HAPT1-4, and GraphCNN1-4 in [122]. Overall, in terms of qualitative and quantitative performances, our methods ranked very closely to two of the highest performing state-of-the-art methods in both the *Proteins* (see Table 5.24) and *Species* (see Table 5.26) classification levels, which are the WKS/SGWS and 3DZD/3DZM. These results are visualised in PRC plots presented in Figure 5.28.

6.3.2 Summary of The Retrieval Performances of The APPFD Method

The APPFD method was evaluated against two of the most interesting datasets, which are the SHREC'12 generic dataset (see Section 5.2.4) and SHREC'14 large scale comprehensive 3D benchmark dataset (see Section 5.2.5), having 1,200 and 8,987 3D objects, respectively. Each of these datasets consists of rigid and non-rigid 3D models, most of which are non-watertight (i.e. defective). First, on the SHREC'12 dataset and retrieval challenge, experimental results reveals that the APPFD method performed generally low, where it ranked 5th out of 6 state-of-the-art methods used in this dataset/retrieval challenge. Although, the APPFD method outperformed one of the state-of-the-art method (i.e. LSD-sum in [131]), it was outperformed by the DG1SIFT, DVD+DB+GMR, ZFDR and 3DSP-L2-1000-hik methods in [131] (see Table 5.29). The reasons for the low overall performances of the APPFD method were explained in Section 5.4.1, where we observed some of the reasons discussed in Section 3.2.1, regarding lots of defective data in the SHREC'12 dataset (3D triangular meshes), whereas our pre-processing steps (see Section 3.2) was not designed to automatically handle defective data.

We see from experimental evaluations that our proposed method (APPFD) is not very suitable for datasets which are characterised by overly complicated and non-rigid models, etc., because it did not consider how to deal with complicated models with few points (i.e. $3,500$ or $4,,500$ points), articulated models, highly non-rigid models such as humans and snakes in Fig 3.31, etc. It is important to note the applicability of the APPFD method for rigid partial and full shapes, and partial and full shapes subject to affine transformations, since the 6 extracted APPF are invariants only under such transformations.

Regarding the SHREC'14 dataset evaluation with the APPFD method, involving 8,987 rigid and non-rigid triangular meshes, experimental results of three evaluation criteria were examined, which are: (i) comprehensive, (ii) proportionally-weighted and (iii) reciprocally-weighted evaluations (see Section 5.4.2). In all these criteria, the results of the best experimental run (run-8c) for the APPFD method outper-

formed one of five state-of-the-art methods (i.e. KVLAD [132]) for that retrieval challenge/dataset, but got outperformed by the remaining four methods, which are: DBNAA_DERE, MR-D1SIFT, ZFDR, and LCDR-DBSVC methods in [132] (see Table 5.34). However, the performance of the APPFD method is not very far from the best overall performing method - the LCDR-DBSVC, the reasons for its poor performance are clearly obvious from the choice of parameters as explained in Sections 5.4.2 and 5.4.2, including that the proposed method (APPFD) was not designed to deal with overly complicated and non-rigid articulated shapes.

Table 6.3 provides concise summaries of the overall retrieval performance rankings of the APPFD method considering the SHREC'12 and SHREC'14 3D shape retrieval challenges. The performance rankings provided are in comparison to those of several other state-of-the-art methods for those retrieval challenges. It is very obvious from experimental evaluation results (see Section 5.4) that our proposed method (APPFD) and other methods did not perform so well in the SHREC'12 and SHREC'14 retrieval challenge. Similar to the APPFD, the LSD-sum [121] evaluated on the SHREC'12 dataset also depends on local features from the surface of 3D meshes, using up to $3,000$ points samples. Their descriptors were matched locally using the Hungarian algorithm, following k-means clustering. We believe that the low number of points sample (less than 5000 points) is majorly responsible for the poor performance of this method. On the other hand, we observe that methods, such as the DSIFT and DVD [121] performed the highest for this dataset because: (i) the DSIFT primarily uses multi-view (42 views) rendering from which dense local descriptors are captured and further combined using the BoW model, typically known to effectively combine 2D/3D local descriptors/features. (ii) although the DSIFT outperformed the DVD method, this method captures both robust local and global features from the 3D surface using Manifold ranking. In conclusion, we submit that the advantage of the DSIFT method is the dense feature extraction using 42 views all around the surface. Such number of views is enough to capture all the deformations, features, and overall topology/structure of the underlying 3D surface. In addition, the BoW method presents additional advantage to this approach. Alternatively, the additional global features and manifold ranking used by the DVD method adds to its robustness, unlike our proposed APPFD and LSD-sum methods, which only depend on locally extracted features on an incredibly small number of points samples.

Considering the SHREC'14 retrieval challenge, a summary of results from participating methods also support our views regarding the performance of the proposed APPFD method and others. With this dataset, the methods which ranked highest, i.e. the LCDR-DBSVC, followed by the DSIFT [142] used view-based approach of locally extracted features which were combined with advanced feature coding and adaptive ranking. An apparent performance improvement was also recorded by the DBSVC method after the application of Manifold ranking to its local features. Finally, similar to some of our views regarding the SHREC'12 dataset, we can confirm that methods which applied the BoW framework and k-means clustering on local features recorded higher improvements, unlike others and our proposed APPFD method. In Table 6.3, we provide concise summaries of the retrieval performance rankings of the APPFD method on two very interesting 3D benchmark datasets

(SHREC'12 and SHREC'14). These summaries reveal how our method compares to several other state-of-the-art methods for these retrieval challenge/datasets.

6.3.3 Summary of The Retrieval Performances of The APPFD-FK-GMM Method

The APPFD-FK-GMM method has demonstrated impressive retrieval performances across several different benchmark datasets, such as the SHREC'17, SHREC'18, SHREC'19, and the SHREC'20 relief dataset, including SHREC'10 and SHREC'11 datasets, where it has been evaluated upon. However, in this thesis, we only presented and analysed the results using four of the most recent datasets from SHREC 2017 to SHREC 2020. It is important to state that the overall retrieval accuracy of the APPFD-FK-GMM method is remarkably close to that of the HAPPS method in all of the datasets and retrieval tasks these two methods have been applied. However, on the SHREC'20 relief dataset, and for the first time, the APPFD-FK-GMM method performed poorly compared to some state-of-the-art methods for this dataset, mainly because the evaluation criteria for this retrieval task did not rely on the geometry of the 3D surfaces, but strictly on their surface relief patterns. Unfortunately, our methods, including the APPFD-FK-GMM method was not designed for robustness against such surface properties (relief patterns). This also shows the complexity of shape retrieval challenges about how to develop the general-purpose techniques. The prior knowledge is usually required on how to perform 3D-Content-based Shape Retrieval (3D-CBSR) tasks from pre-processing to the shape matching. However, such knowledge is usually not available. A summary of the four different experimental evaluations using the datasets indicated above are presented in the paragraph below.

First, the experimental results using the APPFD-FK-GMM method on the SHREC'17 dataset (see Section 5.5.2) reveals the superiority of the APPFD-FK-GMM method over 13 other state-of-the-art methods, which are SnapNet, PO-HAPT, m3DSH-1 to 6, HAPPS-1, GL-FV-IWKS, GL-SV-IWKS, MFLO-FV-IWKS, MFLO-SV-IWKS, PCDL-FV-IWKS, PCDL-SV-IWKS, all SQDF-based, CDSPF, and AlphaVol1-4 methods in [142], judging from the results in Table 5.7. Note that for this experimental evaluation, the APPFD-FK-GMM method also outperforms the HAPPS-1 method (see Table 5.39).

Secondly, the APPFD-FK-GMM method performs exceptionally well on the SHREC'18 protein dataset, with overall retrieval accuracies being remarkably close to that of the HAPPS-1 method. Similar to the evaluations with the HAPPS-1 method for this dataset (see Sections 5.3.1 and 6.3.1), the APPFD-FK-GMM method also outperforms all other state-of-the-art methods (3D-FusionNet, HAPT1-4, SI-WKS, DEM, WKS, and GSGW) [121], applicable to this retrieval challenge, in the *"All"* classification level (also see the results in Table 5.42).

Thirdly, for both the *Protein* and the *Species* classification levels/GT evaluations in the SHREC'19 protein dataset (see Section 5.5.4), the APPFD-FK-GMM method demonstrates superiority over 2 out of 4 state-of-the-art methods, which are: ConvLDSNet and Ft-PSSC methods, and rivals with the 3DZD and HAPT

SHREC'17	**PROBLEM:**	**TABLE 5.39**	**(Experiment 9)**
State-Of-The-Art (SOTA) Methods [142]	**SOTA Performance Rankings**	**APPFD-FK-GMM Performance Ranking**	**Remark**
· BoW-RoPS-DMF-3 · BPHAPT · HAPPS-1	1^{st} 3^{rd} $3^{rd}/4^{th}$	2^{nd}	APPFD-FK-GMM ranks 2^{nd} overall best, better than the HAPPS-1 method and several other state-of-the-art methods for the SHREC'17 3D retrieval problem. This ranking is based on the individual performances (considering the evaluation statistics: NN, FT, ST, E, DCG) of each retrieval method.
SHREC'18	**PROBLEM:**	**TABLE 5.42**	**(Experiment 10)**
State-Of-The-Art (SOTA) Methods [121]	**SOTA Performance Rankings**	**APPFD-FK-GMM Performance Ranking**	**Remark**
· APPFD-FK-GMM · HAPPS-1 · HAPT4	1^{st} 2^{nd} 3^{rd}	1^{st}	APPFD-FK-GMM ranks 1^{st} overall outperforming every other state-of-the-art methods, including HAPPS-1, for the SHREC'18 protein retrieval problem. This ranking is based on the individual performances (considering the evaluation statistics: NN, FT, ST, E, DCG) of each retrieval method.
SHREC'19	**PROBLEM:**	**TABLE 5.45**	**(Experiment 11)**
State-Of-The-Art (SOTA) Methods [120]	**SOTA Performance Rankings**	**APPFD-FK-GMM Performance Ranking**	**Remark**
· 3DZD2 · HAPPS-1 · APPFD-FK-GMM	1^{st} 2^{nd} 3^{rd}	3^{rd}	APPFD-FK-GMM ranks 3^{rd} overall. Here, we compare this improved method to only the top-two performing methods for the SHREC'19 protein retrieval problem. This ranking is based on the individual performances (considering the evaluation statistics: NN, FT, ST, E, DCG) of each retrieval method.

Table 6.4: Summaries of the retrieval performance rankings of the APPFD-FK-GMM method compared to up to three other top-most performing state-of-the-art methods including HAPPS, on the SHREC'17, SHREC'18 and SHREC'19 3D benchmark retrieval challenges or datasets. These summaries reveal how the performances of our improved method compares to other top-performing methods for each retrieval challenge.

methods in [123], including the HAPPS-1 method. The results in Table 5.44 and Table 5.45 highlight the comparative quantitative analyses of this method with the HAPPS-1 and 3DZD2 methods, respectively. As already mentioned, the lower number of points sample (less that 5000 points) from each of the 3D surface used in this evaluation due to the overall large size of the dataset, and the choice of parameter values, are some of the major reasons why methods such as the 3DZD2 and HAPT methods perform slightly better than our methods in this retrieval challenge, considering that in previous retrieval tasks our methods perform better. Another reason is the inability to completely describe the extremely complicated and non-rigid articulated shapes.The proposed methods can only deal with the full and partial rigid shapes and the full and partial shapes subject to affine transformations, since the 6 extracted features are invariants only under such transformations. However, it is almost impossible to develop a single method applicable to different datasets and applications.

Finally, considering the SHREC'20 relief dataset, this method recorded an overall low performance as earlier mentioned. The following state-of-the-art methods: OH, DFE, DPML, SRNA, meshLBP-so and kd-tree FLANN [231] performed better than the APPFD-FK-GMM method, except the PointNet+SQFD method. However, the inferior performance recorded by the APPFD-FK-GMM method for this retrieval challenge/dataset has already been justified in the first paragraph of this section. In Table 6.4, we provide concise summaries of the retrieval performance rankings for our improved retrieval method (the APPFD-FK-GMM), where we compare its compares to only the top-3 performing methods for the SHREC'17, SHREC'18, and SHREC'19 3D protein retrieval challenges.

6.3.4 Summary

Generally, we have presented concise summaries of the overall retrieval performance rankings of each of our proposed methods, comparing these rankings with those of several other state-of-the-art methods for each dataset (retrieval challenge). However, these tabular summaries are not complete representation of all the research contributions of the proposed methods recorded in this thesis.

Several other criteria were also considered to examine the individual performances of each of these methods, during each experimental evaluation, such as the effects of different parameter settings, distance metrics adopted to match different descriptors, number of points, N sampled from the surface of 3D objects, number of bins, especially, the $b_{(APPFD)}$ used, as well as the number of locally extracted features, etc. For example, we have been able to show in each of the experimental evaluations in Chapter 4 how different parameter settings influence the compactness of the final feature-vector (fv), including the overall robustness and retrieval accuracies of our proposed methods (shape descriptors). Generally, in all the experimental evaluations analysed in this thesis for our proposed methods, the results of experimental runs with the highest overall performances are compared to the best results of several other state-of-the-art methods for that particular retrieval challenge and/or dataset. We refer the reader to Sections 5.3, 5.4, and 5.5 for an in-depth quantitative, as well as qualitative evaluation of our proposed HAPPS-1&2,

APPFD, and improved APPFD-FK-GMM methods, respectively.

6.4 Research Findings

In this research study, we find that the overall performance of a shape retrieval system or method does not only depend on the robustness of the shape descriptor used, but on a number of different factors, such as the choice of distance metric(s) adopted to match the descriptors, since a given distance metric is only suitable for certain types of descriptors, and different metrics perform differently on a given shape descriptor, as demonstrated in our experimental results in Table 5.14, Section 5.3.3, where we first evaluate the overall performances of our descriptors using several different distance metrics to examine and adopt the most suitable metric with best overall results for the descriptor. In addition, the matching approach plays a vital role in the overall performance of a shape retrieval method. For example, the results of adopting local matching (i.e. matching local descriptors) and global matching (i.e. matching global descriptors) for a given shape retrieval method, such as the APPFD would definitely not be the same. We explain this in detain in Section 3.6.1.

Secondly, we find that the descriptive power or robustness of a shape retrieval algorithm or method does not only or strictly depend on the dataset to which the algorithm has been applied, but also on the GT and evaluation criteria. We see this in the experimental results for the SHREC'19 (see Tables 5.44 and 5.45) and SHREC'20 (see Tables 5.24 and 5.26) protein shape retrieval challenge, where two different GT/classification levels (*Proteins* and *Species*) were adopted for performance evaluation in each case, using exactly the same IR evaluation metrics. In the above case, the overall performance results of the *Proteins* classification level are higher than those of the *Species* level. Therefore, if the only GT available was that of the *Species* classification level without the other, those results would be accepted as the final judgement to the retrieval accuracies of the shape retrieval method that produced them, which is not true, considering that a different GT (in this case, the *Proteins* classification level) produces far higher results.

It is interestingly to also note that the size of dataset (i.e. the total number of 3D objects available in a dataset) has some influence on the overall performance rating of a shape retrieval algorithm or method, whereby the larger the size of the dataset, the lower the overall retrieval accuracies, and vice versa. We confirm this, first, by comparing the overall retrieval performances (i.e. experimental results) of the SHREC'10 (see Table 5.12, run-3a) and the SHREC'11 (see Table 5.18, run-4a) retrieval of non-rigid 3D objects, which involves 200 and 600 watertight models, respectively. We can see from these results that the overall performances of the HAPPS-1 method on the SHREC'10 dataset with smaller dataset are higher than those with the SHREC'11 dataset, which is larger, considering exactly the same evaluation metrics. However, the slight differences in the choice of parameter settings for these two datasets (see Tables 5.11 and 5.16) evaluation could also account for the overall results obtained.

Currently, as already stated in Section 1.1, to the best of our knowledge, there is hardly a knowledge-based 3D shape descriptor that has been tested across a wide va-

riety of 3D benchmark datasets (each of which presents a unique retrieval challenge to the shape descriptor or retrieval algorithm) while also recording excellent retrieval performances across board, except for the 3D shape descriptors we have proposed in this thesis. We demonstrate and confirm how generic our retrieval methods are, through numerous experimental evaluations and analyses, where the retrieval performances of each of our proposed retrieval methods (see Section 4.2) have been validated on at least two different benchmark datasets as in Chapter 4. Basically, the APPFD is the main or baseline algorithm for both the HAPPS and APPFD-FK-GMM methods, and considering that these algorithms have been evaluated on a minimum of ten different SHREC benchmark datasets (Table 5.1), where for each retrieval challenge/dataset, most of the other state-of-the-art methods whose retrieval performances were evaluated are data-driven, and no other knowledge-based approach has been evaluated with all the datasets presented in Table 5.1, except our APPFD-based methods. Experimental results (see Chapter 4) validates that our knowledge-based 3D shape descriptors or retrieval methods performs so well, on average across all the ten different 3D benchmark datasets.

Recently, 3D shape retrieval methods known for high performance rankings are the ones which adopts the data-driven approach, which deals with large training and testing datasets and some complicated feature extraction and shape description techniques (see Section 2.5.1). Alternatively, all our proposed methods adopt a rather simpler knowledge-based approach (see Section 2.5.2) to produce retrieval and classification results that outperforms the the data-driven methods in most retrieval challenges (datasets) and competes very closely with some data-driven methods in other cases, in terms of retrieval performances (these are seen in the experimental evaluation results in Chapter 5.

In this thesis, we present and evaluate a set of 3D shape descriptors (retrieval methods) that has been tested against all the issues summarised in these research findings, and they have demonstrated exceptional performances across different benchmark datasets, including outperforming most state-of-the-art methods applicable to those datasets and retrieval challenges.

6.5 Future Research Direction

In this section, we explain what should be done following the research work presented in this thesis and how other researchers who follow this research, its methodology, or concepts are expected to get involved in order to contribute and/or improve on the current work. Several research techniques have been adopted and discuss in different sections of this thesis. In Chapter 5, retrieval results of several experimental evaluations have also been reported for all the 3D shape retrieval methods we propose in this thesis. In these experiments, we have considered several different evaluation approaches, evaluation metrics, retrieval challenges (datasets), as well as data representations or format, etc. Many factors could be responsible for the overall retrieval performances of a 3D shape retrieval method, such as choice of matching method, number of points sample (i.e. LoD), side of database, data source, pre-processing, and feature-extraction technique applied.

Experimental results in Chapter 5 and those clearly summarising the overall retrieval performances ranking of our proposed methods compared to other state-of-the-art methods (Table 6.1 through to Table 6.4) reveals a lot of interesting issues, few of which are:

- Some retrieval methods worked better on certain datasets, opposed to others. What could be learned from this, for future research direction (as stated earlier in the thesis) is that different datasets presents completely unique retrieval challenge to retrieval algorithms (method) due to some variations (i.e. rigid, non-rigid, water-tight, non-water-tight, and/or defective surface) characterised by these different datasets. Hence, in developing a new method or improving on existing one, it is important to consider these data variations.

- Our proposed retrieval methods, such as the HAPPS and APPFD-FK-GMM methods performs exceptionally well in terms of rankings on several different datasets (retrieval challenges). However, some experimental results with the HAPPS method reveals its staggering performances (i.e. 2^{nd} or 3^{rd} ranking) closely below the best overall ranking, even with different parameter settings. Since this method relies on the baseline APPFD method, the very high final feature-vector dimension of this descriptor could influence these results. Future research direction into this aspect would be to further investigate the effects of the very high-dimension of the APPFD method and possibly implement with lower feature-vector dimension and improved performances.

- The very high length of the APPFD is characterised with very many zero columns (i.e. bins). This is practically irrelevant to the (dis)similarity when any two of these descriptors are being matched and impacts the overall performances of the APPFD. Empirically, shortening the final feature-vector of the APPFD would greatly improve its performances. Therefore, we expect future research work on this to devise a technique that effectively remove all zero bins/columns from the APPFD method.

- For such minimal number of surface points sample ($3,500$ to $4,500$) used by our proposed methods to describe a given 3D model, we would highly appraise their retrieval performances. Considering this, we can say that our proposed methods have met the characteristics of a good 3D shape descriptor (explained in Section 2.3.1). Future research direction may be interested to investigate the possibilities of further performance improvements by increasing the 3D surface's LoD (i.e. the value of N) with appropriate parameter settings. However, doing so would present a trade-off between performance and computing resources.

Experimental results show how parameter settings of our proposed methods affects the overall retrieval performances. This could also account for the reason some other methods do/not perform well on certain datasets. Basically, several other factors such as incorrectly estimated surface normal (which other features depend upon) can affect performance of some methods work well on certain datasets, for methods relying on surface normals. Other future research directions are explained below.

First, our proposed APPFD method in this thesis (see Section 4.2.1) involves a binning approach where all the locally extracted APPF are stacked together before multi-dimensional binning technique is applied to all 6-feature dimension - see Section 4.2.1. The final *fv* derived becomes the normalised, flattened multi-dimensional histogram. *This is because, we adopted the global matching approach for all our descriptors, instead of local matching* (see Section 2.2.6). Sadly, a lot of shape information or details would have been lost during the process of stacking together the locally-extracted APPF. Although this is the case for our most-recent improvement (i.e. the APPFD-FK-GMM method, as described in Section 4.2.4), in future research, we hope to be able to compute the APPFD separately for each LSP and subsequently adopt the local matching approach, as opposed to global matching of full shapes/surfaces. We hope that this concept would further improve the overall performances (accuracy and robustness) of, strictly, the APPFD method for 3D shape retrieval and/or classification tasks.

Secondly, the overall performance of our proposed APPFD method relies on its parameter settings and the features extracted: five *"angular"* features and one *"length"* feature. Finding appropriate parameter combination (settings) that would yield consistent optimum performance (i.e. retrieval results) remains a big challenge for the features extraction and shape descriptor approach we have adopted. Future work might need to consider ways to improve upon this.

Thirdly, regarding the HoGD method, we would love to independently evaluate this method on different datasets and investigate the effects of different number of points sample or vertices, including the $b_{(HoGD)}$ parameter value, on its retrieval performances. This is essential in order to further assess the overall performances our the HAPPS method, with the contribution of the HoGD.

Fourthly, the primary goal of combining two or more shape descriptors (i.e. *local+local*, *local+global*, or *global+global*) is to improve the overall retrieval performance of the resultant hybrid approach, which is expected to be better than the individual descriptors themselves. Unfortunately, this is not always the case, as we have witnessed cases, where combining different descriptors result to poor overall performance, instead. An example is the PointNet+SQFD(run3) descriptor in Table 5.46. Several other similar cases also exist. Essentially, the combination of local+local, local+global, and global+global descriptors are applicable to different categories of shapes: local+local are applicable to partial and full shapes, while the latter two are applicable to the full shapes only. However, we believe that the best approach to combining different shape descriptors is yet to be directly determined, including that further investigation is needed to adopt the best possible combination of the APPFD and HoGD or any other descriptor to produce a more robust HAPPS method. This is subject to future work. In Section 4.2.3, we mentioned the need to further investigate the impact of combining two or more descriptors with variable lengths (i.e. $117, 649$ and 196), using different similarity metrics and selected datasets.

Fifthly, although we have investigated the overall performances of our methods on datasets containing a combination of rigid and non-rigid 3D objects, we

have mainly evaluated the performance of our retrieval methods on datasets that exclusively contain non-rigid 3D objects, and demonstrated the ability of our proposed methods to generalise across a diverse domain of 3D shape retrieval challenge. Our inability to test our methods on dataset(s) that exclusively contain rigid CAD objects is due to unavailability of such benchmark datasets in the public domain. However, in future work, we are interested in creating a rigid benchmark dataset with GT, and evaluating our retrieval methods on dataset(s) which exclusively contains rigid 3D CAD models, in order to further investigate the generality of our methods and possibly improve on them if necessary.

Another important issue to stress is regarding the accuracy of estimated surface normal, considering that our proposed methods somehow rely on this. We mentioned in Section 3.3.2 how *"inaccurately estimated normals would adversely affect our final shape descriptor"*, including subsequent feature extraction steps. In Section 3.3.4, we empirically show how the normals simultaneously computed during the BI phase produce more accurate outward-pointing normals, unlike with the PCA-based technique that applies after pre-sampling of 3D surface points - see Figures 3.12 and 3.13. However, in future work, it may be useful to do an ablation study about which method is better for shape retrieval between using the BI based approach or the one estimated from the sampled points vis PCA approach, rather than just criticizing the latter.

Finally, we have considered possibilities to build/develop a 3D-CBRS desktop or web-based user-interface application which would provide non-technical users (students or new researchers) the ability to apply our proposed methods to solve real-world problem, by being able to perform actual 3D objects retrieval, test out the different methods and parameter setting, apply different methods with different parameters with ease, etc. However, the goal of such application is for academic and research purpose, and to support teaching and learning of the subject (especially to undergraduate students and early career researchers in 3D computer vision and pattern recognition), in addition to being further developed into a functional product or brand for personal or commercial use.

Appendix A

Thesis-related Publications

In the course of this research, the following list of publications have resulted from this thesis, which forms part of our research contributions. These submissions have been peer-reviewed and published as conference and journal articles.

A.1 Nonrigid 3D Shape Retrieval with HAPPS: A Novel Hybrid Augmented Point Pair Signature [175]

Author: Ekpo Otu, Reyer Zwiggelaar, David Hunter, Yonguai Liu, **Article Type:** Conference Paper, **Journal:** 2019 International Conference on Computational Science and Computational Intelligence (CSCI), **Publisher:** IEEEXplore, **Pages:** 662 - 668, **Year:** 2019, **DOI:** https://doi.org/10.1109/CSCI49370.2019.00124.

Abstract: "A robust, yet computationally efficient signature for describing 3D shape remains a challenge for 3D computer vision and related applications. Having a signature that is generalizable across a wider range of datasets becomes another important research issue. This paper proposes a novel Hybrid signature, the Augmented Point Pair Signature (HAPPS), that is robust, highly discriminating, efficient, and capable of effectively representing 3D point cloud and polygon mesh surfaces. We tested the overall performances of HAPPS on three standardized benchmark datasets for 3D shape retrieval: The Shape Retrieval Contest 2018 (SHREC'18) protein shapes benchmark, with 2,267 protein conformers, SHREC'17 Point cloud Retrieval of Nonrigid Toys (PRoNTo), with 100 3D point clouds, and SHREC'10 Nonrigid shape retrieval having 200 triangular meshes. Using 6 standard retrieval performance metrics to evaluate our results, we demonstrated the superiority of our HAPPS retrieval method over several other state-of-the-art methods for the SHREC'18 protein dataset, while also competing side-by-side with the best 2 performing methods for the other benchmark datasets."

A.2 SHREC 2020: Multi-domain protein shape retrieval challenge [122]

Author: Florent Langenfeld, Matthieu Montes, Ekpo Otu, Reyer Zwiggelaar, David Hunter, Yonguai Liu, Yu-Kun Lai, Paul L. Rosin, Tunde Aderinwale, Genki Terashi, Charles Christoffer, Daisuke Kihara, Halim Benhabiles, Karim Hammoudi, Adnane Cabani, Feryal Windal, Mahmoud Melkemi, Andrea Giachetti, Stelios Mylonas, Apostolos Axenopoulos, Petros Daras, Yuxu Peng, **Article Type:** Journal Paper, **Journal:** Computers & Graphics, **Publisher:** Elsevier, **Volume:** 91, **Pages:** 189–198, **Year:** 2020, **DOI:** https://doi.org/10.1016/j.cag.2020.07.013.

Abstract: "Proteins are natural modular objects usually composed of several domains, each domain bearing a specific function that is mediated through its surface, which is accessible to vicinal molecules. This draws attention to an understudied characteristic of protein structures: surface, that is mostly unexploited by protein structure comparison methods. In the present work, we evaluated the performance of six shape comparison methods, among which three are based on machine learning, to distinguish between 588 multi-domain proteins and to recreate the evolutionary relationships at the protein and species levels of the SCOPe database.

The six groups that participated in the challenge submitted a total of 15 sets of results. We observed that the performance of all the methods significantly decreases at the species level, suggesting that shape-only protein comparison is challenging for closely related proteins. Even if the dataset is limited in size (only 588 proteins are considered whereas more than 160,000 protein structures are experimentally solved), we think that this work provides useful insights into the current shape comparison methods performance and highlights possible limitations to large-scale applications due to the computational cost."

A.3 SHREC 2020: Retrieval of digital surfaces with similar geometric reliefs [163]

Author: Elia Moscoso Thompson, Silvia Biasotti, Ekpo Otu, Reyer Zwiggelaar, David Hunter, Yonghuai Liu, Andrea Giachetti, Claudio Tortorici, Naoufel Werghi, Ahmad Shaker Obeid, Stefano Berretti, Hoang-Phuc Nguyen-Dinh, Minh-Quan Le, Hai-Dang Nguyen, Minh-Triet Tran, Leonardo Gigli, Santiago Velasco-Forero, Beatriz Marcotegui, Ivan Sipiran, Benjamin Bustos, Ioannis Romanelis, Vlassis Fotis, Gerasimos Arvanitis, Konstantinos Moustakas, Yoko Arteaga, and Ramamoorthy Luxman, **DOI:** https://doi.org/10.1016/j.cag.2020.07.011, **Article Type:** Journal Paper, **Journal:** Computers & Graphics, **Publisher:** Elsevier, **Volume:** 91, **Pages:** 199-218, **Year:** 2020.

Abstract: "This paper presents the methods that have participated in the SHREC 2020 contest on retrieval of surface patches with similar geometric reliefs and the analysis of their performance over the benchmark created for this challenge. The goal of the context is to verify the possibility of retrieving 3D models only based on the reliefs that are present on their surface and to compare methods that are

suitable for this task. This problem is related to many real world applications, such as the classification of cultural heritage goods or the analysis of different materials. To address this challenge, it is necessary to characterize the local 'geometric pattern' information, possibly forgetting model size and bending. Seven groups participated in this contest and twenty runs were submitted for evaluation. The performances of the methods reveal that good results are achieved with a number of techniques that use different approaches."

A.4 SHREC 2021: Retrieval and classification of protein surfaces equipped with physical & chemical properties (Recent submission, March 2021)

Author: Andrea Raffo, Ulderico Fugacci, Ekpo Otu, Reyer Zwiggelaar, David Hunter, Yonghuai Liu, and others, **Article Type:** Journal Paper, **Journal:** Computers & Graphics, **Publisher:** Elsevier, **Year:** 2021.

Challenge: "The aim of this SHREC'21 track is to evaluate the performance of retrieval and classification algorithms for protein surfaces characterized by physicochemical properties. Starting from a set of protein structures in different conformational states observed via NMR experiments and deposited in the PDB repository, we build their Solvent Excluded Surface (SES) by the freely available software NanoShaper. The track is jointly organized by IMATI-CNR and the CONCEPT lab at IIT. The final report of this SHREC'21 track will be submitted as a joint contribution to the international journal Computers & Graphics and will follow a two-stage review process. The paper will be authored by the track coordinators and all participants who submitted their results."

Comment Five different groups from five different countries (including ours in the UK) participated in this (i.e. the SHREC 2021 protein retrieval and classification challenge). We adopted two different methods based on the variations in the dataset. The first method is our HAPPS method presented in this thesis and the second method is a novel exploratory data analysis based method called HP4-EDA. These implementations can be found ***on this link***. The outcome of performance evaluation for all submitted methods from all participating groups reveal that both our methods ranked second place, overall. This work has recently been submitted to Computers & Graphics journal for publication.

A.5 SHREC 2021: Surface-based protein domains retrieval (Recent submission, March 2021)

Author: Matthieu Montes, Florent Langenfeld, Ekpo Otu, Reyer Zwiggelaar, David Hunter, Yonghuai Liu, and others, **Article Type:** Journal Paper, **Journal:** Computers & Graphics, **Publisher:** Elsevier, **Year:** 2021.

Challenge: "The aim of this track is to assess the performance of shape retrieval methods on a dataset of related multi-domain protein surfaces. Domains are structural as well as functional sub-units of proteins, that can exist independently of the rest of the protein. Usually, proteins are made of two or more of such domains, and are the level at which protein interactions and functions are studied. To compare proteins at the domain level for similarities is a common task in structural biology, biochemistry or drug discovery. Proteins can be described as non-rigid surfaces representing their solvent-excluded surface (SES) as defined by Connoly (Connoly et al., J Appl Cryst. 1983). Additional, biologically-relevant information can be provided, such as electrostatics, to further describe these molecular shapes. This track proposes a set of representing the conformational space of 10 query domains extracted from the PFAM database (El-Geabli et al., NAR, 2019) as well as 554 surfaces of multi-domain proteins. Compared to the previous Protein Shape Retrieval contests, we focus on the evaluation of the performance to retrieve 10 individual domains among a set of 554 multi-domains protein surfaces."

Comment Five different groups from five different countries (including ours in the UK) participated in this (i.e. the SHREC 2021 Surface-based protein domains retrieval challenge). We applied our latest 3D retrieval and classification method, the APPFD-FK-GMM to this problem which presented two varieties of datasets, each posing a unique difficulty to retrieval algorithms. Experimental evaluation results for all submitted methods from all participating groups reveal that our method ranked also ranked second-overall in performance for both datasets. This work has recently been submitted to Computers & Graphics journal for publication.

Appendix B

Code and Data For Experimental Evaluations

We decide to publish the codes and datasets reported in this thesis in order for the reader or interested person to be able to reproduce exactly the results we have obtained. We expect that making the codes and functions used to produce the results reported in this thesis widely available (online), would have the following benefits:

- Allow potentially interested researchers to further improve on our work or create their new work based on our code.
- We understand that there may be salient aspects in the code that is bughouse to explain in writing, and that the code would contain everything we have tried to explain in the thesis.
- We also think that making our implementation details widely (publicly) available would provide higher possibilities of finding and fixing potential issues relating to our implementation, which leads to more potential improvements.

B.1 Code To Allow Reproducibility of Our Experimental Results

The respective code to reproduce the experimental results we have presented in this thesis could be found on the following URLs:

B.1.1 Code for the APPFD Algorithm

Please visit ***this link*** for access to the code implementation of the APPFD method (see Section 4.2.1), which can be used to reproduce all the resuts presented in Section 5.4. Also included are clear instructions on how to use these codes.

B.1.2 Code for the HoGD Algorithm

Please visit ***this link*** for access to the code implementation of the HoGD method (see Section 4.2.2), which we combined with the APPFD to produce all the HAPPS-

1 method, described in Section 4.2.3. Also included are clear instructions on how to use these codes.

B.1.3 Code for the HAPPS Algorithm

Please visit ***this link*** for access to the code implementation of the HAPPS method (see Section 4.2.3), which can be used to reproduce all the resuts presented in Section 5.3. Also included are clear instructions on how to use these codes.

B.1.4 Code for the APPFD-FK-GMM Algorithm

Please visit ***this link*** and ***this link***, for access to the code implementation of the APPFD-FK-GMM method (see Section 4.2.4), which can be used to reproduce all the resuts presented in Section 5.5. Also included are clear instructions on how to use these codes.

B.1.5 Other Related Utility Code and Software

By *"Utility Codes and Software"*, we refer to some of the most-relevant functions, modules, code snipets, and software that have been helpful for the entire 3D shape retrieval research. All of the software used for this research are Open Source. For example, see ***MeshLab*** visualisation and processing software for 3D mesh and point cloud data. Others are: ***Matplotlib***, for plotting and visualisation of data/results; ***SciPy***, for scientific computation functions; ***NumPy***, for numerical computing and effective handling of arrays and unstructured data, etc. The PSB also provides series of utility codes and sample datasets for 3D shape retrieval, which can be obtained by ***visiting this link***, for the codes and ***visiting this link***, to obtain the datasets. We refer the reader to also ***visit this link***, for any other functions or code implementations we developed for this research work.

B.1.6 Performance Evaluation Codes

Besides the utility codes and software presented in this section, we specifically provide all the performance evaluation codes used to produce all the qualitative and quantitative results presented in Chapter 4. We therefore refer the reader to ***this link***, for access to these codes and GT or CI files.

B.2 Data To Allow Reproducibility of Our Experimental Results

The respective dataset and Ground Truth file(s), including Classification Index file(s) to reproduce the expeprimental resuls we have presented in this thesis could be found on the following URLs:

B.2.1 Datasets, Ground Truths and Evaluation Code

We refer the reader to Section 5.2, for a detailed description of all the datasets evaluated in this thesis. Considering that it would be relatively expensive to upload all

of these datasets onto cloud storage for sharing purposes, we instead refer the reader to the respective SHREC track where these datasets can be directly downloaded. In Table 5.1, a summarised list of all the datasets are presented, including references to their respective sources. The reader is encouraged to download any required dataset by following the appropriate reference. However, in the event that any of the URLs fails to load, the reader should please email to ***Ekpo Otu***.

Bibliography

[1] Princeton Shape Benchmark (PSB). *Classification File Format*. https://shape.cs.princeton.edu/benchmark/documentation/classification_format.html. Accessed: 2020-09-01.

[2] Alexander Agathos et al. "Retrieval of 3D Articulated Objects Using a Graph-based Representation." In: *3DOR* 2009 (2009), pp. 29–36.

[3] Ceyhun Burak Akgül et al. "Density-based 3D shape descriptors". In: *EURASIP Journal on Advances in Signal Processing* 2007.1 (2006), p. 032503.

[4] Khaled Alhamzi, Mohammed Mahfouz Elmogy, and Sherif Barakat. "3D Object Recognition Based on Local and Global Features Using Point Cloud Library". In: 2015.

[5] Civilian American and European Surface Anthropometry Resource Project—CAESAR. *The most comprehensive source for body measurement data*. http://store.sae.org/caesar/. Accessed: 2019-08-02.

[6] Tarik Filali Ansary, Mohamed Daoudi, and Jean-Philippe Vandeborre. "A bayesian 3-d search engine using adaptive views clustering". In: *IEEE Transactions on Multimedia* 9.1 (2006), pp. 78–88.

[7] I Atmosukarto and P Naval. *A survey of 3D model retrieval systems*. 2003.

[8] Mathieu Aubry, Ulrich Schlickewei, and Daniel Cremers. "The wave kernel signature: A quantum mechanical approach to shape analysis". In: *2011 IEEE international conference on computer vision workshops (ICCV workshops)*. IEEE. 2011, pp. 1626–1633.

[9] North American Aviation. *Smithsonian National Air and Space Museum*. https://airandspace.si.edu/. Accessed: 2020-08-01.

[10] G. Barequet and S. Kumar. "Repairing CAD models". In: *Proceedings. Visualization '97 (Cat. No. 97CB36155)*. Oct. 1997, pp. 363–370. DOI: 10.1109/VISUAL.1997.663904.

[11] Vincent Barra and Silvia Biasotti. "Learning Kernels on Extended Reeb Graphs for 3D Shape Classification and Retrieval". In: *Eurographics Workshop on 3D Object Retrieval, Girona, Spain, 2013. Proceedings*. 2013, pp. 25–32. DOI: 10.2312/3DOR/3DOR13/025-032. URL: https://doi.org/10.2312/3DOR/3DOR13/025-032.

[12] Bruce G Baumgart. "A polyhedron representation for computer vision". In: *Proceedings of the May 19-22, 1975, national computer conference and exposition*. ACM. 1975, pp. 589–596.

[13] Neslihan Bayramoglu and A. Aydin Alatan. "Comparison of 3D Local and Global Descriptors for Similarity Retrieval of Range Data". In: *Neurocomput.* 184.C (Apr. 2016), pp. 13–27. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2015.08.105. URL: https://doi.org/10.1016/j.neucom.2015.08.105.

[14] Matthew Berger et al. "State of the Art in Surface Reconstruction from Point Clouds". In: *Eurographics 2014 - State of the Art Reports*. Vol. 1. EUROGRAPHICS star report 1. Strasbourg, France, Apr. 2014, pp. 161–185. DOI: 10.2312/egst.20141040. URL: https://hal.inria.fr/hal-01017700.

[15] Helen M. Berman et al. "The Protein Data Bank". In: *Nucleic Acids Research* 28.1 (Jan. 2000), pp. 235–242. ISSN: 0305-1048. DOI: 10.1093/nar/28.1.235. eprint: https://academic.oup.com/nar/article-pdf/28/1/235/9895144/280235.pdf. URL: https://doi.org/10.1093/nar/28.1.235.

[16] Paul J Besl and Neil D McKay. "Method for registration of 3-D shapes". In: *Sensor fusion IV: control paradigms and data structures*. Vol. 1611. International Society for Optics and Photonics. 1992, pp. 586–606.

[17] Gary Bishop and David M Weimer. "Fast phong shading". In: *ACM SIGGRAPH Computer Graphics* 20.4 (1986), pp. 103–106.

[18] Federica Bogo et al. "FAUST: Dataset and evaluation for 3D mesh registration". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 3794–3801.

[19] E. R. Boroson and N. Ayanian. "3D Keypoint Repeatability for Heterogeneous Multi-Robot SLAM". In: *2019 International Conference on Robotics and Automation (ICRA)*. May 2019, pp. 6337–6343. DOI: 10.1109/ICRA.2019.8793609.

[20] Davide Boscaini et al. "Learning class-specific descriptors for deformable shapes using localized spectral convolutional networks". In: *Computer Graphics Forum*. Vol. 34. 5. Wiley Online Library. 2015, pp. 13–23.

[21] O Bottema. "On the area of a triangle in barycentric coordinates". In: *Crux Mathematicorum* 8.8 (1982), pp. 228–231.

[22] Lev M Bregman. "The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming". In: *USSR computational mathematics and mathematical physics* 7.3 (1967), pp. 200–217.

[23] Alexander M Bronstein, Michael M Bronstein, and Ron Kimmel. *Numerical geometry of non-rigid shapes*. Springer Science & Business Media, 2008.

[24] Alexander M Bronstein et al. "Shape google: Geometric words and expressions for invariant shape retrieval". In: *ACM Transactions on Graphics (TOG)* 30.1 (2011), pp. 1–20.

[25] Michael M Bronstein and Iasonas Kokkinos. "Scale-invariant heat kernel signatures for non-rigid shape recognition". In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE. 2010, pp. 1704–1711.

[26] M Brusco et al. "3D registration by textured spin-images". In: *Fifth International Conference on 3-D Digital Imaging and Modeling (3DIM'05)*. IEEE. 2005, pp. 262–269.

[27] Anders Glent Buch et al. "Local Point Pair Feature Histogram for Accurate 3D Matching." In: *BMVC*. 2018, p. 143.

[28] Benjamin Bustos and Tobias Schreck. *Feature-Based 3D Object Retrieval.* 2009.

[29] Benjamin Bustos et al. "Feature-based similarity search in 3D object databases". In: *ACM Computing Surveys (CSUR)* 37.4 (2005), pp. 345–387.

[30] Thorsten M Buzug. "Computed tomography". In: *Springer Handbook of Medical Technology*. Springer, 2011, pp. 311–342.

[31] Owen Carmichael, Daniel Huber, and Martial Hebert. "Large data sets and confusing scenes in 3-d surface matching and recognition". In: *Second International Conference on 3-D Digital Imaging and Modeling (Cat. No. PR00062)*. IEEE. 1999, pp. 358–367.

[32] John-Marc Chandonia, Naomi K Fox, and Steven E Brenner. "SCOPe: classification of large macromolecular structures in the structural classification of proteins—extended database". In: *Nucleic acids research* 47.D1 (2019), pp. D475–D481.

[33] John-Marc Chandonia, Naomi K Fox, and Steven E Brenner. "SCOPe: manual curation and artifact removal in the structural classification of proteins–extended database". In: *Journal of molecular biology* 429.3 (2017), pp. 348–355.

[34] Angel X. Chang et al. *ShapeNet: An Information-Rich 3D Model Repository*. Tech. rep. arXiv:1512.03012 [cs.GR]. Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[35] Ding-Yun Chen. "Three-dimensional model shape description and retrieval based on lightfield descriptors". PhD thesis. Ph. d., NTU CSIE, 2003.

[36] Ding-Yun Chen et al. "On visual similarity based 3D model retrieval". In: *Computer graphics forum*. Vol. 22. 3. Wiley Online Library. 2003, pp. 223–232.

[37] Qiang Chen et al. "3D CAD model retrieval based on the combination of features". In: *Multimedia Tools and Applications* 74 (Jan. 2014). DOI: 10.1007/s11042-013-1850-9.

[38] Duke University Chris Tralie. *ShapeGoogle*. https://github.com/sandmman/ShapeGoogle/tree/master/models_off/. Accessed: 2020-09-08.

[39] Peter Claes et al. "Automatic, robust and accurate 3D modelling based on variational implicit surface". In: *KUL/ESAT/PSI/0405* (2004).

[40] IMATI CNR. *Instituite of Applied Mathematics and Information Technology (IMATI)*. http://www.ge.imati.cnr.it/index.php/. Accessed: 2020-09-08.

[41] Michael L Connolly. "Analytical molecular surface calculation". In: *Journal of applied crystallography* 16.5 (1983), pp. 548–558.

[42] Jonathan Corney et al. "Coarse filters for shape matching". In: *IEEE Computer Graphics and Applications* 22.3 (2002), pp. 65–74.

[43] CVonline. *The Earth Mover's Distance*. http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/RUBNER/emd.htm. Accessed: 2020-10-02.

[44] Christopher M Cyr and Benjamin B Kimia. "3D object recognition using shape similiarity-based aspect graph". In: *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*. Vol. 1. IEEE. 2001, pp. 254–261.

[45] James Davis et al. "Filling Holes In Complex Surfaces Using Volumetric Diffusion". In: Feb. 2002, pp. 428–441. ISBN: 0-7695-1521-4. DOI: 10.1109/TDPVT.2002.1024098.

[46] Jesse Davis and Mark Goadrich. "The relationship between Precision-Recall and ROC curves". In: *Proceedings of the 23rd international conference on Machine learning*. 2006, pp. 233–240.

[47] Dawson-Haggerty et al. *Trimesh*. Version 3.2.0. Dec. 8, 2019. URL: https://trimsh.org/.

[48] Alberto Del Bimbo and Pietro Pala. "Content Based Retrieval of 3D Data". In: *Progress in Pattern Recognition, Image Analysis and Applications*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 13–24. ISBN: 978-3-540-30463-0.

[49] Martin Dillon. *Introduction to modern information retrieval: G. Salton and M. McGill. McGraw-Hill, New York (1983)*. Pergamon, 1983.

[50] H Quynh Dinh and Steven Kropac. "Multi-resolution spin-images". In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*. Vol. 1. IEEE. 2006, pp. 863–870.

[51] B. Drost et al. "Model globally, match locally: Efficient and robust 3D object recognition". In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. June 2010, pp. 998–1005. DOI: 10.1109/CVPR.2010.5540108.

[52] Shaoyi Du et al. "Affine iterative closest point algorithm for point set registration". In: *Pattern Recognition Letters* 31.9 (2010), pp. 791–799.

[53] Richard O Duda, Peter E Hart, and David G Stork. "Pattern classification second edition john wiley & sons". In: *New York* 58 (2001), p. 16.

[54] Jean-Luc Dugelay, Atilla Baskurt, and Mohamed Daoudi. *3D object processing: compression, indexing and watermarking*. John Wiley & Sons, 2008.

[55] Eurographics. *3D Object Retrieval 2020 (3DOR'20)*. https://www.eg.org/wp/event/3d-object-retrieval-2020-3dor20/. Accessed: 2020-09-09.

[56] Eurographics. *Eurographics Workshop on 3D Object Retrieval 2019*. https://diglib.eg.org/page/BIBTEX_3e462073-be9e-4315-a951-82be24e864ff/. Accessed: 2019-08-02.

[57] Rui Fang et al. "A New Shape Benchmark for 3D Object Retrieval". In: *Advances in Visual Computing*. Ed. by George Bebis et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 381–392. ISBN: 978-3-540-89639-5.

[58] S. Filipe and L. A. Alexandre. "A comparative evaluation of 3D keypoint detectors in a RGB-D Object Dataset". In: *2014 International Conference on Computer Vision Theory and Applications (VISAPP)*. Vol. 1. Jan. 2014, pp. 476–483.

[59] Alex Flint, Anthony Dick, and Anton Van Den Hengel. "Thrift: Local 3d structure recognition". In: *9th Biennial Conference of the Australian Pattern Recognition Society on Digital Image Computing Techniques and Applications (DICTA 2007)*. IEEE. 2007, pp. 182–188.

[60] Matthieu Montes Florent Langenfeld. *SHREC 2018 Protein Shape Retrieval Dataset- Conservatoire National des Arts-et-Metiers*. `http://shrec2018.drugdesign.fr/`. Accessed: 2020-09-08.

[61] Matthieu Montes Florent Langenfeld. *SHREC 2019 Protein Shape Retrieval Dataset- Conservatoire National des Arts-et-Metiers*. `http://shrec2019.drugdesign.fr/`. Accessed: 2020-09-08.

[62] Naomi K Fox, Steven E Brenner, and John-Marc Chandonia. "SCOPe: Structural Classification of Proteins—extended, integrating SCOP and ASTRAL data and classification of new structures". In: *Nucleic acids research* 42.D1 (2014), pp. D304–D309.

[63] Bent Fuglede and Flemming Topsoe. "Jensen-Shannon divergence and Hilbert space embedding". In: *International Symposium onInformation Theory, 2004. ISIT 2004. Proceedings.* IEEE. 2004, p. 31.

[64] Thomas Funkhouser et al. "A search engine for 3D models". In: *ACM Transactions on Graphics (TOG)* 22.1 (2003), pp. 83–105.

[65] Pablo Gainza et al. "Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning". In: *Nature Methods* 17.2 (2020), pp. 184–192.

[66] Ran Gal, Ariel Shamir, and Daniel Cohen-Or. "Pose-oblivious shape signature". In: *IEEE transactions on visualization and computer graphics* 13.2 (2007), pp. 261–271.

[67] Yue Gao and Qionghai Dai. "View-based 3d object retrieval: Challenges and approaches". In: *IEEE multimedia* 21.3 (2014), pp. 52–57.

[68] A Giachetti et al. "Shrec'16 Track: Retrieval of Human Subjects from Depth Sensor Data". In: *Eurographics Workshop on 3D Object Retrieval*. Eurographics. Lisbon, Portugal, May 2016, pp. 1–6. URL: `https://hal.archives-ouvertes.fr/hal-01314067`.

[69] Ian Gibson et al. *Additive manufacturing technologies*. Vol. 17. Springer, 2014.

[70] Daniela Giorgi, Silvia Biasotti, and Laura Paraboschi. "Shape retrieval contest 2007: Watertight models track". In: *SHREC competition* 8.7 (2007).

[71] Afzal Godil et al. "Benchmarks, performance evaluation and contests for 3D shape retrieval". In: *Proceedings of the 10th Performance Metrics for Intelligent Systems Workshop*. 2010, pp. 42–47.

[72] Afzal Godil et al. "SHREC'09 Track: Generic shape retrieval." In: *3DOR*. 2009, pp. 61–68.

[73] Adrien Gressin et al. "Towards 3D lidar point cloud registration improvement using optimal neighborhood knowledge". In: *ISPRS journal of photogrammetry and remote sensing* 79 (2013), pp. 240–251.

[74] A. Gueziec et al. "Converting sets of polygons to manifold surfaces by cutting and stitching". In: *Proceedings Visualization '98 (Cat. No.98CB36276)*. Oct. 1998, pp. 383–390. DOI: 10.1109/VISUAL.1998.745327.

[75] Kan Guo, Dongqing Zou, and Xiaowu Chen. "3d mesh labeling via deep convolutional neural networks". In: *ACM Transactions on Graphics (TOG)* 35.1 (2015), pp. 1–12.

[76] Yulan Guo et al. "A novel local surface feature for 3D object recognition under clutter and occlusion". In: *Information Sciences* 293 (2015), pp. 196–213.

[77] Yulan Guo et al. "Rotational projection statistics for 3D local surface description and object recognition". In: *International journal of computer vision* 105.1 (2013), pp. 63–86.

[78] Xian-Feng Hana et al. *A comprehensive review of 3D point cloud descriptors*. 2018. arXiv: 1802.02297 [cs.CV].

[79] Jonathon S Hare et al. "Mind the gap: another look at the problem of the semantic gap in image retrieval". In: *Multimedia Content Analysis, Management, and Retrieval 2006*. Vol. 6073. International Society for Optics and Photonics. 2006, p. 607309.

[80] Li He, Xiaolong Wang, and Hong Zhang. "M2DP: A novel 3D point cloud descriptor and its application in loop closure detection". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pp. 231–237.

[81] Tommy Hinks et al. "Point Cloud Data Conversion into Solid Models via Point-Based Voxelization". In: *Journal of Surveying Engineering* 139 (Jan. 2012), pp. 72–83. DOI: 10.1061/(ASCE)SU.1943-5428.0000097.

[82] Stefan Hinterstoisser et al. "Going further with point pair features". In: *European conference on computer vision*. Springer. 2016, pp. 834–848.

[83] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. "A fast learning algorithm for deep belief nets". In: *Neural computation* 18.7 (2006), pp. 1527–1554.

[84] Geoffrey E Hinton and Ruslan R Salakhutdinov. "Reducing the dimensionality of data with neural networks". In: *science* 313.5786 (2006), pp. 504–507.

[85] Richard Hoffman and Anil K Jain. "Segmentation and classification of range images". In: *IEEE transactions on pattern analysis and machine intelligence* 5 (1987), pp. 608–620.

[86] Christoph M Hofmann. *Geometric and solid modeling: an introduction*. Morgan Kaufmann, 1989.

[87] Stefan Holzer et al. "Adaptive neighborhood selection for real-time surface normal estimation from organized point cloud data using integral images". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2012, pp. 2684–2689.

[88] Berthold Klaus Paul Horn. "Extended gaussian images". In: *Proceedings of the IEEE* 72.12 (1984), pp. 1671–1686.

[89] S-M Hu et al. "Modifying the shape of NURBS surfaces with geometric constraints". In: *Computer-Aided Design* 33.12 (2001), pp. 903–912.

[90] George Ioannakis et al. "RETRIEVAL—An Online Performance Evaluation Tool for Information Retrieval Methods". In: *IEEE Transactions on Multimedia* 20.1 (2017), pp. 119–127.

[91] Natraj Iyer et al. "Three-dimensional shape searching: state-of-the-art review and future trends". In: *Computer-Aided Design* 37.5 (2005), pp. 509–530.

[92] Subramaniam Jayanti et al. "Developing an engineering shape benchmark for CAD models". In: *Computer-Aided Design* 38.9 (2006), pp. 939–953.

[93] Hervé Jégou et al. "Aggregating local image descriptors into compact codes". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2011). URL: `http://hal.inria.fr/inria-00633013`.

[94] Shuangshuang Jin, Robert R Lewis, and David West. "A comparison of algorithms for vertex normal computation". In: *The visual computer* 21.1-2 (2005), pp. 71–82.

[95] Andrew E Johnson. "Spin-images: a representation for 3-D surface matching". In: (1997).

[96] Andrew E. Johnson and Martial Hebert. "Using spin images for efficient object recognition in cluttered 3D scenes". In: *IEEE Transactions on pattern analysis and machine intelligence* 21.5 (1999), pp. 433–449.

[97] Andrew Edie Johnson and Martial Hebert. "Efficient multiple model recognition in cluttered 3-D scenes". In: *Proceedings. 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No. 98CB36231)*. IEEE. 1998, pp. 671–677.

[98] K Sparck Jones and Cornelis Joost Van Rijsbergen. "Information retrieval test collections". In: *Journal of documentation* (1976).

[99] Karen Sparck Jones and Peter Willett. *Readings in information retrieval*. Morgan Kaufmann, 1997.

[100] Sing Bing Kang and Katsushi Ikeuchi. "The complex EGI: A new representation for 3-D pose determination". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 15.7 (1993), pp. 707–721.

[101] Michael Kazhdan et al. "A reflective symmetry descriptor for 3D models". In: *Algorithmica* 38.1 (2004), pp. 201–225.

[102] Michael M Kazhdan. "Shape representations and algorithms for 3D model retrieval". PhD thesis. Princeton University Princeton, 2004.

[103] Ismail Khalid Kazmi, Lihua You, and Jian Jun Zhang. "A survey of 2D and 3D shape descriptors". In: *2013 10th International Conference Computer Graphics, Imaging and Visualization*. IEEE. 2013, pp. 1–10.

[104] David G Kendall. "The diffusion of shape". In: *Advances in applied probability* 9.3 (1977), pp. 428–430.

[105] Mohamed A Khamsi and William A Kirk. *An introduction to metric spaces and fixed point theory*. Vol. 53. John Wiley & Sons, 2011.

[106] Lilita Kiforenko et al. "A performance evaluation of point pair features". In: *Computer Vision and Image Understanding* 166 (2018), pp. 66–80. ISSN: 1077-3142. DOI: https://doi.org/10.1016/j.cviu.2017.09.004. URL: http://www.sciencedirect.com/science/article/pii/S1077314217301601.

[107] Duck Hoon Kim et al. "A new mpeg-7 standard: Perceptual 3-d shape descriptor". In: *Pacific-Rim Conference on Multimedia*. Springer. 2004, pp. 238–245.

[108] Kitware. *VTK*. https://vtk.org/Wiki/VTK/. Accessed: 2020-08-28.

[109] K. Klasing et al. "Comparison of surface normal estimation methods for range sensing applications". In: *2009 IEEE International Conference on Robotics and Automation*. May 2009, pp. 3206–3211. DOI: 10.1109/ROBOT.2009.5152493.

[110] Jan J Koenderink and Andrea J Van Doorn. "Surface shape and curvature scales". In: *Image and vision computing* 10.8 (1992), pp. 557–564.

[111] Josip Krapac, Jakob Verbeek, and Frédéric Jurie. "Modeling spatial layout with fisher vectors for image categorization". In: *2011 International Conference on Computer Vision*. IEEE. 2011, pp. 1487–1494.

[112] Elena Kudryavtseva. "Topology of the spaces of Morse functions on surfaces". In: *arXiv preprint arXiv:1104.4792* (2011).

[113] S. Kullback and R. A. Leibler. "On Information and Sufficiency". In: *Ann. Math. Statist.* 22.1 (Mar. 1951), pp. 79–86. DOI: 10.1214/aoms/1177729694. URL: https://doi.org/10.1214/aoms/1177729694.

[114] Solomon Kullback. *Information theory and statistics*. Courier Corporation, 1997.

[115] Chen-Tsung Kuo and Shyi-Chyi Cheng. "3D model retrieval using principal plane analysis and dynamic programming". In: *Pattern Recognition* 40.2 (2007), pp. 742–755.

[116] Camille Kurtz et al. "On combining image-based and ontological semantic dissimilarities for medical image retrieval applications". In: *Medical image analysis* 18.7 (2014), pp. 1082–1100.

[117] H Laga et al. "Bag of words and local spectral descriptor for 3D partial shape retrieval". In: *Proceedings of the Eurographics Workshop on 3D Object Retrieval (3DOR'11)*. Citeseer. 2011, pp. 41–48.

[118] Florent Langenfeld et al. "Protein Shape Retrieval Contest". In: *Eurographics Workshop on 3D Object Retrieval*. Ed. by Silvia Biasotti, Guillaume Lavoué, and Remco Veltkamp. The Eurographics Association, 2019, pp. 25–31. ISBN: 978-3-03868-077-2. DOI: 10.2312/3dor.20191058.

[119] Florent Langenfeld et al. "Protein Shape Retrieval Contest". In: *Eurographics Workshop on 3D Object Retrieval*. Ed. by Silvia Biasotti, Guillaume Lavoué, and Remco Veltkamp. The Eurographics Association, 2019, pp. 25–31. ISBN: 978-3-03868-077-2. DOI: 10.2312/3dor.20191058.

[120] Florent Langenfeld et al. "Protein Shape Retrieval Contest". In: (2019).

[121] Florent Langenfeld et al. "SHREC 2018–Protein Shape Retrieval". In: *Eurographics Workshop on 3D Object Retrieval*. 2018, pp. 53–61.

[122] Florent Langenfeld et al. "SHREC 2020: Multi-domain protein shape retrieval challenge". In: *Computers & Graphics* 91 (2020), pp. 189–198. ISSN: 0097-8493. DOI: https://doi.org/10.1016/j.cag.2020.07.013. URL: https://www.sciencedirect.com/science/article/pii/S0097849320301151.

[123] Florent Langenfeld et al. "SHREC19 Protein Shape Retrieval Contest". In: (2019).

[124] Afshan Latif et al. "Content-based image retrieval and feature extraction: a comprehensive review". In: *Mathematical Problems in Engineering* 2019 (2019).

[125] Scikit Learn. *Nearest Neighbors Classification Algorithms*. https://tinyurl.com/scilitlearnNN. Accessed: 2021-08-25.

[126] Scikit Learn. *Precision-Recall*. https://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html. Accessed: 2020-09-01.

[127] Soochahn Lee et al. "A new 3-D model retrieval system based on aspect-transition descriptor". In: *European Conference on Computer Vision*. Springer. 2006, pp. 543–554.

[128] Thibault Lescoat et al. "A survey on data-driven dictionary-based methods for 3d modeling". In: *Computer Graphics Forum*. Vol. 37. 2. Wiley Online Library. 2018, pp. 577–601.

[129] Clement Leung and Horace Ip. "Benchmarking for Content-Based Visual Information Search." In: Jan. 2000, pp. 442–456.

[130] B. Li et al. "Large Scale Comprehensive 3D Shape Retrieval". In: *Proceedings of the 7th Eurographics Workshop on 3D Object Retrieval*. 3DOR '14. Strasbourg, France: Eurographics Association, 2014, pp. 131–140. ISBN: 978-3-905674-58-3. DOI: 10.2312/3dor.20141059. URL: https://doi.org/10.2312/3dor.20141059.

[131] B. Li et al. "SHREC'12 Track: Generic 3D Shape Retrieval". In: *Proceedings of the 5th Eurographics Conference on 3D Object Retrieval*. EG 3DOR'12. Cagliari, Italy: Eurographics Association, 2012, pp. 119–126. ISBN: 978-3-905674-36-1. DOI: 10.2312/3DOR/3DOR12/119-126. URL: http://dx.doi.org/10.2312/3DOR/3DOR12/119-126.

[132] B. Li et al. "SHREC'14 Track: Large Scale Comprehensive 3D Shape Retrieval". English. In: *Eurographics Workshop on 3D Object Retrieval*. ., 2014.

[133] Bao Li et al. "Robust normal estimation for point clouds with sharp features". In: *Computers & Graphics* 34.2 (2010), pp. 94–106.

[134] Bo Li, Yijuan Lu, and Henry Johan. "Sketch-based 3D model retrieval by viewpoint entropy-based adaptive view clustering". In: *Proceedings of the Sixth Eurographics Workshop on 3D Object Retrieval*. 2013, pp. 49–56.

[135] Chunyuan Li and A Ben Hamza. "A multiresolution descriptor for deformable 3D shape retrieval". In: *The Visual Computer* 29.6-8 (2013), pp. 513–524.

[136] Xizhi Li et al. "Invariant local shape descriptors: classification of large-scale shapes with local dissimilarities". In: *Proceedings of the Computer Graphics International Conference*. ACM. 2017, p. 9.

[137] Yang Li and Edwin R Hancock. "Face recognition using shading-based curvature attributes". In: *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.* Vol. 3. IEEE. 2004, pp. 538–541.

[138] Z. Lian et al. "Non-rigid 3D Shape Retrieval". In: *Proceedings of the 2015 Eurographics Workshop on 3D Object Retrieval*. 3DOR '15. Zurich, Switzerland: Eurographics Association, 2015, pp. 107–120. ISBN: 978-3-905674-78-1. DOI: 10.2312/3dor.20151064. URL: https://doi.org/10.2312/3dor.20151064.

[139] Z. Lian et al. "SHREC'11 Track: Shape Retrieval on Non-rigid 3D Watertight Meshes". In: *Eurographics Workshop on 3D Object Retrieval*. Ed. by H. Laga et al. The Eurographics Association, 2011. ISBN: 978-3-905674-31-6. DOI: 10.2312/3DOR/3DOR11/079-088.

[140] Zhouhui Lian et al. "SHREC'10 track: Non-rigid 3D shape retrieval". In: Jan. 2010, pp. 101–108. DOI: 10.2312/3DOR/3DOR10/101-108.

[141] Point Cloud Library. *The PCD (Point Cloud Data) file format*. http://pointclouds.org/documentation/tutorials/pcd_file_format.php. Accessed: 2019-07-30.

[142] F. A. Limberger et al. "Point-cloud Shape Retrieval of Non-rigid Toys: SHREC'17 Track". In: *Proceedings of the Workshop on 3D Object Retrieval*. 3Dor '17. Lyon, France: Eurographics Association, 2017, pp. 75–84. DOI: 10.2312/3dor.20171056. URL: https://doi.org/10.2312/3dor.20171056.

[143] Wilfried Linder. *Digital photogrammetry*. Vol. 1. Springer, 2009.

[144] Yi Liu, Hongbin Zha, and Hong Qin. "Shape topics: A compact representation and new algorithms for 3d partial shape retrieval". In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*. Vol. 2. IEEE. 2006, pp. 2025–2032.

[145] Zhenbao Liu et al. "High-level semantic feature for 3D shape based on deep belief networks". In: *2014 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE. 2014, pp. 1–6.

[146] Jobst Loffler. "Content-based retrieval of 3D models in distributed web databases by visual shape information". In: *2000 IEEE Conference on Information Visualization. An International Conference on Computer Visualization and Graphics*. IEEE. 2000, pp. 82–87.

[147] K Berker Logoglu, Sinan Kalkan, and Alptekin Temizel. "Cospair: colored histograms of spatial concentric surflet-pairs for 3D object recognition". In: *Robotics and Autonomous Systems* 75 (2016), pp. 558–570.

[148] Oscar Lopes et al. "Spherical blurred shape model for 3-D object and pose recognition: Quantitative analysis and HCI applications in smart environments". In: *IEEE transactions on cybernetics* 44.12 (2014), pp. 2379–2390.

[149] Graciela Lara López et al. "Comparative analysis of shape descriptors for 3D objects". In: *Multimedia Tools and Applications* 76.5 (2017), pp. 6993–7040.

[150] Rongrong Lu et al. "Three-dimensional object recognition using an extensible local surface descriptor". In: *Optical Engineering* 56.12 (2017), pp. 1–13. DOI: 10.1117/1.OE.56.12.123109. URL: https://doi.org/10.1117/1.OE.56.12.123109.

[151] Athanasios Mademlis et al. "Combining topological and geometrical features for global and partial 3-D shape retrieval". In: *IEEE Transactions on Multimedia* 10.5 (2008), pp. 819–831.

[152] Said Mahmoudi and Mohamed Daoudi. "3D models retrieval by using characteristic views". In: *Object recognition supported by user interaction for service robots*. Vol. 2. IEEE. 2002, pp. 457–460.

[153] A. P. Majtey, P. W. Lamberti, and D. P. Prato. "Jensen-Shannon divergence as a measure of distinguishability between mixed quantum states". In: *Phys. Rev. A* 72 (5 Nov. 2005), p. 052310. DOI: 10.1103/PhysRevA.72.052310. URL: https://link.aps.org/doi/10.1103/PhysRevA.72.052310.

[154] Christopher D Manning, Hinrich Schütze, and Prabhakar Raghavan. *An Introduction To Information Retrieval*. Cambridge university press, 2008.

[155] Jonathan Masci et al. "Geodesic convolutional neural networks on riemannian manifolds". In: *Proceedings of the IEEE international conference on computer vision workshops*. 2015, pp. 37–45.

[156] Wolfram Mathworld. *Dot Product*. http://mathworld.wolfram.com/DotProduct.html. Accessed: 2019-10-15.

[157] Florent Langenfeld Matthieu Montes. *SHREC 2020: Track on Multi-domain Protein Shape Retrieval, Dataset- Conservatoire National des Arts-et-Metiers.* http://shrec2020.drugdesign.fr/. Accessed: 2020-08-01.

[158] Boris Mederos, Luiz Velho, and Luiz Henrique de Figueiredo. "Robust smoothing of noisy point clouds". In: *Proc. SIAM Conference on Geometric Design and Computing*. Vol. 2004. 1. 2003, p. 2.

[159] Niloy J Mitra and An Nguyen. "Estimating surface normals in noisy point cloud data". In: *Proceedings of the nineteenth annual symposium on Computational geometry*. ACM. 2003, pp. 322–328.

[160] Kaichun Mo et al. "PartNet: A Large-Scale Benchmark for Fine-Grained and Hierarchical Part-Level 3D Object Understanding". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019.

[161] Todd K Moon. "The expectation-maximization algorithm". In: *IEEE Signal processing magazine* 13.6 (1996), pp. 47–60.

[162] E Moscoso Thompson et al. "Retrieval of Gray Patterns Depicted on 3D Models". In: (2018).

[163] Elia Moscoso Thompson et al. "SHREC 2020: Retrieval of digital surfaces with similar geometric reliefs". In: *Computers & Graphics* 91 (2020), pp. 199–218. ISSN: 0097-8493. DOI: https://doi.org/10.1016/j.cag.2020.07.011. URL: https://www.sciencedirect.com/science/article/pii/S0097849320301138.

[164] Marieke Mur et al. "Human object-similarity judgments reflect and transcend the primate-IT object representation". In: *Frontiers in psychology* 4 (2013), p. 128.

[165] Isabella Stewart Gardner Museum. *Thirteen Works: Explore the Gardner's Stolen Art*. https://www.gardnermuseum.org/. Accessed: 2020-08-01.

[166] Ramanathan Muthuganapathy and Karthik Ramani. "SHape REtrieval contest 2008: CAD models." In: *Shape Modeling International*. 2008, pp. 221–222.

[167] myminifactory. *3D Model Design and Sharing Community Platform: MyMiniFactory*. https://www.myminifactory.com/. Accessed: 2020-08-01.

[168] Marcin Novotni and Reinhard Klein. "3D Zernike descriptors for content based shape retrieval". In: *Proceedings of the eighth ACM symposium on Solid modeling and applications*. 2003, pp. 216–225.

[169] Ryutarou Ohbuchi, Masatoshi Nakazawa, and Tsuyoshi Takei. "Retrieving 3D shapes based on their appearance". In: *Proceedings of the 5th ACM SIGMM international workshop on Multimedia information retrieval*. 2003, pp. 39–45.

[170] Ryutarou Ohbuchi et al. "Salient local visual features for shape-based 3D model retrieval". In: *2008 IEEE International Conference on Shape Modeling and Applications*. IEEE. 2008, pp. 93–102.

[171] Ryutarou Ohbuchi et al. "Shape-similarity search of three-dimensional models using parameterized statistics". In: *10th Pacific Conference on Computer Graphics and Applications, 2002. Proceedings.* IEEE. 2002, pp. 265–274.

[172] Mats HM Olsson et al. "PROPKA3: consistent treatment of internal and surface residues in empirical p K a predictions". In: *Journal of chemical theory and computation* 7.2 (2011), pp. 525–537.

[173] Robert Osada et al. "Matching 3D models with shape distributions". In: *Proceedings International Conference on Shape Modeling and Applications*. IEEE. 2001, pp. 154–166.

[174] Robert Osada et al. "Shape distributions". In: *ACM Transactions on Graphics (TOG)* 21.4 (2002), pp. 807–832.

[175] E. Otu et al. "Nonrigid 3D Shape Retrieval with HAPPS: A Novel Hybrid Augmented Point Pair Signature". In: *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*. 2019, pp. 662–668. DOI: 10.1109/CSCI49370.2019.00124.

[176] Daoshan OuYang and Hsi-Yung Feng. "On the normal vector estimation for point cloud data from smooth surfaces". In: *Computer-Aided Design* 37.10 (2005), pp. 1071–1079.

[177] Pete Pachal. *Windows 3D: Inside Microsoft's ambitious plan to bring 3D to everyone*. https://mashable.com/2016/10/26/microsoft-3d-platform/?europe=true. Accessed: 2019-04-08.

[178] Panagiotis Papadakis et al. "3D Object Retrieval using an Efficient and Compact Hybrid Shape Descriptor". In: 2008.

[179] Panagiotis Papadakis et al. "Panorama: A 3d shape descriptor based on panoramic views for unsupervised 3d object retrieval". In: *International Journal of Computer Vision* 89.2-3 (2010), pp. 177–192.

[180] Eric Paquet et al. "Description of shape information for 2-D and 3-D objects". In: *Signal processing: Image communication* 16.1-2 (2000), pp. 103–122.

[181] Geogios Passalis, Theoharis Theoharis, and Ioannis A Kakadiaris. "Ptk: A novel depth buffer-based shape descriptor for three-dimensional object retrieval". In: *The Visual Computer* 23.1 (2007), pp. 5–14.

[182] RCSB PDB. *Protein Data Bank*. https://www.rcsb.org/. Accessed: 2019-06-02.

[183] Nick Pears, Yonghuai Liu, and Peter Bunting. *3D imaging, analysis and applications*. Vol. 3. Springer, 2012.

[184] Fernando Pérez-Cruz. "Kullback-Leibler divergence estimation of continuous distributions". In: *2008 IEEE international symposium on information theory*. IEEE. 2008, pp. 1666–1670.

[185] princeton. *Object File Format (.off)*. http://segeval.cs.princeton.edu/public/off_format.html/. Accessed: 2019-09-15.

[186] Scientific Python. *cKDTree*. https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.cKDTree.html. Accessed: 2018-09-06.

[187] Andrea Raffo. *SHREC'21 track: Retrieval and classification of protein surfaces equipped with physical & chemical properties*. http://shrec.ge.imati.cnr.it/shrec21_protein/. Accessed: 2021-09-08.

[188] Andrea Raffo et al. "SHREC 2021: Retrieval and classification of protein surfaces equipped with physical and chemical properties". In: *Computers & Graphics* 99 (2021), pp. 1–21. ISSN: 0097-8493. DOI: https://doi.org/10.1016/j.cag.2021.06.010. URL: https://www.sciencedirect.com/science/article/pii/S0097849321001254.

[189] Gholamreza Rafiee, Satnam Singh Dlay, and Wai Lok Woo. "A review of content-based image retrieval". In: *2010 7th International Symposium on Communication Systems, Networks & Digital Signal Processing (CSNDSP 2010)*. IEEE. 2010, pp. 775–779.

[190] Alexander Reshetov, Alexei Soupikov, and William R. Mark. "Consistent normal interpolation". In: *ACM Trans. Graph.* 29 (2010), p. 142.

[191] Princeton Shape Retrieval and Analysis Group. *Utility Code*. http://shape.cs.princeton.edu/benchmark/. Accessed: 2019-08-28.

[192] Maximilian Riesenhuber and Tomaso Poggio. "Models of object recognition". In: *Nature neuroscience* 3.11 (2000), pp. 1199–1204.

[193] Emanuele Rodolà et al. "A Scale Independent Selection Process for 3D Object Recognition in Cluttered Scenes". In: *International Journal of Computer Vision* 102.1-3 (2013), pp. 129–145. DOI: 10.1007/s11263-012-0568-x. URL: https://app.dimensions.ai/details/publication/pub.1016939603%20and%20https://iris.unive.it/bitstream/10278/35245/1/10.1007_s11263-012-0568-x.pdf.

[194] David F Rogers. *An introduction to NURBS: with historical perspective*. Elsevier, 2000.

[195] Reihaneh Rostami et al. "A Survey on Data-Driven 3D Shape Descriptors". In: *Computer Graphics Forum*. Vol. 38. 1. Wiley Online Library. 2019, pp. 356–393.

[196] Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. "A metric for distributions with applications to image databases". In: *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*. IEEE. 1998, pp. 59–66.

[197] Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. "The earth mover's distance as a metric for image retrieval". In: *International journal of computer vision* 40.2 (2000), pp. 99–121.

[198] R. B. Rusu, N. Blodow, and M. Beetz. "Fast Point Feature Histograms (FPFH) for 3D registration". In: *2009 IEEE International Conference on Robotics and Automation*. May 2009, pp. 3212–3217. DOI: 10.1109/ROBOT.2009.5152473.

[199] R. B. Rusu and S. Cousins. "3D is here: Point Cloud Library (PCL)". In: *2011 IEEE International Conference on Robotics and Automation*. May 2011, pp. 1–4. DOI: 10.1109/ICRA.2011.5980567.

[200] R. B. Rusu et al. "Aligning point cloud views using persistent feature histograms". In: *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Sept. 2008, pp. 3384–3391. DOI: 10.1109/IROS.2008.4650967.

[201] R. B. Rusu et al. "Fast 3D recognition and pose using the Viewpoint Feature Histogram". In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Oct. 2010, pp. 2155–2162. DOI: 10.1109/IROS.2010.5651280.

[202] R. B. Rusu et al. "Learning informative point classes for the acquisition of object model maps". In: *2008 10th International Conference on Control, Automation, Robotics and Vision*. Dec. 2008, pp. 643–650. DOI: 10.1109/ICARCV.2008.4795593.

[203] Radu Bogdan Rusu et al. "Persistent Point Feature Histograms for 3D Point Clouds". In: 2008.

[204] Samuele Salti, Federico Tombari, and Luigi Di Stefano. "SHOT: Unique signatures of histograms for surface and texture description". In: *Computer Vision and Image Understanding* 125 (2014), pp. 251–264. ISSN: 1077-3142. DOI: https://doi.org/10.1016/j.cviu.2014.04.011. URL: http://www.sciencedirect.com/science/article/pii/S1077314214000988.

[205] Julia Sanchez et al. "Robust normal vector estimation in 3D point clouds through iterative principal component analysis". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 163 (2020), pp. 18–35.

[206] Tefko Saracevic. "Evaluation of evaluation in information retrieval". In: *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*. 1995, pp. 138–146.

[207] N Schlömer et al. *Meshio*. 2019.

[208] Johannes Lutz Schönberger and Jan-Michael Frahm. "Structure-from-Motion Revisited". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[209] Johannes Lutz Schönberger et al. "Pixelwise View Selection for Unstructured Multi-View Stereo". In: *European Conference on Computer Vision (ECCV)*. 2016.

[210] Scratchapixel. *Rasterization: a Practical Implementation*. `https://www.scratchapixel.com/lessons/3d-basic-rendering/rasterization-practical-implementation/rasterization-stage`. Accessed: 2020-09-08.

[211] Scratchapixel. *Ray Tracing: Rendering a Triangle*. `https://www.scratchapixel.com/lessons/3d-basic-rendering/ray-tracing-rendering-a-triangle/barycentric-coordinates`. Accessed: 2020-09-08.

[212] J. Serafin, E. Olson, and G. Grisetti. "Fast and robust 3D feature extraction from sparse point clouds". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Oct. 2016, pp. 4105–4112. DOI: 10.1109/IROS.2016.7759604.

[213] Ying Shan et al. "Shapeme histogram projection and matching for partial object recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.4 (2006), pp. 568–577.

[214] Yu-Te Shen et al. "3D model search engine based on lightfield descriptors". In: *Proc. eurographics*. 2003.

[215] Philip Shilane et al. "The Princeton Shape Benchmark". In: *In Shape Modeling International*. 2004, pp. 167–178.

[216] Philip Shilane et al. "The princeton shape benchmark". In: *Proceedings Shape Modeling Applications, 2004.* IEEE. 2004, pp. 167–178.

[217] Kaleem Siddiqi et al. "Retrieving articulated 3-D models using medial surfaces". In: *Machine vision and applications* 19.4 (2008), pp. 261–275.

[218] Karen Simonyan et al. "Fisher vector faces in the wild." In: *BMVC*. Vol. 2. 3. 2013, p. 4.

[219] Ayan Sinha, Jing Bai, and Karthik Ramani. "Deep learning 3D shape surfaces using geometry images". In: *European Conference on Computer Vision*. Springer. 2016, pp. 223–240.

[220] Ivan Sipiran and Benjamin Bustos. "A Robust 3D Interest Points Detector Based on Harris Operator". In: *Proceedings of the 3rd Eurographics Conference on 3D Object Retrieval*. 3DOR '10. Norrköping, Sweden: Eurographics Association, 2010, pp. 7–14. ISBN: 978-3-905674-22-4. DOI: 10.2312/3DOR/3DOR10/007-014. URL: http://dx.doi.org/10.2312/3DOR/3DOR10/007-014.

[221] Ivan Sipiran and Benjamin Bustos. "Harris 3D: a robust extension of the Harris operator for interest point detection on 3D meshes". In: *The Visual Computer* 27.11 (2011), p. 963.

[222] Colin Smith. *On vertex-vertex systems and their use in geometric and biological modelling*. University of Calgary, 2006.

[223] Sima Sobhiyeh et al. "Hole Filling in 3D Scans for Digital Anthropometric Applications". In: *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE. 2019, pp. 2752–2757.

[224] Chresten R Søndergaard et al. "Improved treatment of ligands and coupling effects in empirical calculation and rationalization of p K a values". In: *Journal of chemical theory and computation* 7.7 (2011), pp. 2284–2295.

[225] Bastian Steder et al. "Point feature extraction on 3D range scans taking into account object boundaries". In: *2011 IEEE International Conference on Robotics and Automation*. IEEE. 2011, pp. 2601–2608.

[226] Jian Sun, Maks Ovsjanikov, and Leonidas Guibas. "A concise and provably informative multi-scale signature based on heat diffusion". In: *Computer graphics forum*. Vol. 28. 5. Wiley Online Library. 2009, pp. 1383–1392.

[227] Gary KL Tam et al. "Registration of 3D point clouds and meshes: a survey from rigid to nonrigid". In: *IEEE transactions on visualization and computer graphics* 19.7 (2012), pp. 1199–1217.

[228] Johan WH Tangelder and Remco C Veltkamp. "A survey of content based 3D shape retrieval methods". In: *Proceedings Shape Modeling Applications, 2004.* IEEE. 2004, pp. 145–156.

[229] Johan WH Tangelder and Remco C Veltkamp. "A survey of content based 3D shape retrieval methods". In: *Multimedia tools and applications* 39.3 (2008), pp. 441–471.

[230] Michael Taylor. "Using multiple 2D projections to characterize 3D irregular particles". In: *Proceedings of the 12th Annual Symposium of the International Center for Aggregates Research*. 2004.

[231] Elia Moscoso Thompson. *SHREC'20 track: Retrieval and classification of surface patches with similar geometric relief*. http://shrec.ge.imati.cnr.it/shrec20_pattern_retr/index.html/. Accessed: 2020-09-08.

[232] Grit Thürrner and Charles A. Wüthrich. "Computing Vertex Normals from Polygonal Facets". In: *Journal of Graphics Tools* 3.1 (1998), pp. 43–46. DOI: 10.1080/10867651.1998.10487487. eprint: https://doi.org/10.1080/10867651.1998.10487487. URL: https://doi.org/10.1080/10867651.1998.10487487.

[233] Robert F Tobler and Stefan Maierhofer. "A mesh data structure for rendering and subdivision". In: (2006).

[234] Federico Tombari, Samuele Salti, and Luigi Di Stefano. "Performance evaluation of 3D keypoint detectors". In: *International Journal of Computer Vision* 102.1-3 (2013), pp. 198–220.

[235] Federico Tombari, Samuele Salti, and Luigi Di Stefano. "Unique signatures of histograms for local surface description". In: *European conference on computer vision*. Springer. 2010, pp. 356–369.

[236] Rob Tuytel. *Texture Haven*. `https://texturehaven.com/`. Accessed: 2020-04-23.

[237] McGill University. *McGill 3D Shape Benchmark*. `http://www.cim.mcgill.ca/~shape/benchMark`. Accessed: 2019-04-08.

[238] Oregon State University. *Design Repository*. `https://design.engr.oregonstate.edu/repo/`,`http://ftest.mime.oregonstate.edu/repo/browse/`. Accessed: 2019-08-02.

[239] Princeton University. *3D Model Search Engine*. `https://gist.github.com/danoneata/9927923/`. Accessed: 2016-05-06.

[240] Princeton University. *3D ShapeNets: A Deep Representation for Volumetric Shapes*. `http://3dshapenets.cs.princeton.edu/`. Accessed: 2019-08-02.

[241] Oliver Van Kaick et al. "A survey on shape correspondence". In: *Computer Graphics Forum*. Vol. 30. 6. Wiley Online Library. 2011, pp. 1681–1707.

[242] J-P Vandeborre, Vincent Couillet, and Mohamed Daoudi. "A practical approach for 3D model indexing by combining local and global invariants". In: *Proceedings. First International Symposium on 3D Data Processing Visualization and Transmission*. IEEE. 2002, pp. 644–647.

[243] Remco Veltkamp. *Imaging and Multimedia*. `http://www.cs.uu.nl/centers/give/imaging/3Drecog/3Dmatching.html`. Accessed: 2020-11-20.

[244] Remco Veltkamp et al. "SHREC2006: 3D Shape Retrieval Contest". In: (Jan. 2006).

[245] Remco C Veltkamp et al. *SHREC2006: 3d shape retrieval contest*. 2006.

[246] Colin C Venters and Matthew Cooper. "A review of content-based image retrieval systems". In: *JISC Technology Applications Programme, http://www.jtap. ac. uk/reports/htm/jtap-054. html* (2000).

[247] Visionair. *The Shape Repository*. `http://visionair.ge.imati.cnr.it/ontologies/shapes/`,`http://visionair.ge.imati.cnr.it/ontologies/shapes/viewmodels.jsp`. Accessed: 2019-05-24.

[248] Marinus T Vlaardingerbroek and Jacques A Boer. *Magnetic resonance imaging: theory and practice*. Springer Science & Business Media, 2013.

[249] Dejan Vranic and Dietmar Saupe. "3D shape descriptor based on 3D Fourier transform". In: *EURASIP*. 2001, pp. 271–274.

[250] Dejan V Vranic and Dietmar Saupe. "3D model retrieval." PhD thesis. Citeseer, 2004.

[251] Dejan V Vranic, Dietmar Saupe, and Jörg Richter. "Tools for 3D-object retrieval: Karhunen-Loeve transform and spherical harmonics". In: *2001 IEEE Fourth Workshop on Multimedia Signal Processing (Cat. No. 01TH8564)*. IEEE. 2001, pp. 293–298.

[252] Eric Wahl, Ulrich Hillenbrand, and Gerd Hirzinger. "Surflet-pair-relation histograms: a statistical 3D-shape representation for rapid classification". In: *Fourth International Conference on 3-D Digital Imaging and Modeling, 2003. 3DIM 2003. Proceedings.* IEEE. 2003, pp. 474–481.

[253] Raoul Wessel, Ina Blümel, and Reinhard Klein. "A 3D shape benchmark for retrieval and automatic classification of architectural data". In: *Proceedings of the 2nd Eurographics conference on 3D Object Retrieval*. Eurographics Association. 2009, pp. 53–56.

[254] Wikipedia. *Dot Product*. `https://en.wikipedia.org/wiki/Dot_product`. Accessed: 2019-10-15.

[255] Wikipedia. *Euclidean distance*. `https://en.wikipedia.org/wiki/Euclidean_distance`. Accessed: 2020-10-02.

[256] Wikipedia. *Kullback–Leibler divergence*. `https://en.wikipedia.org/wiki/Kullback-Leibler_divergence`. Accessed: 2020-10-02.

[257] Wikipedia. *Lidar*. `https://en.wikipedia.org/wiki/Lidar`. Accessed: 2020-09-23.

[258] Wikipedia. *Microsoft Kinect*. `https://en.wikipedia.org/wiki/Kinect`. Accessed: 2020-09-23.

[259] Wikipedia. *Polygon mesh*. `https://en.wikipedia.org/wiki/Polygon_mesh/`. Accessed: 2019-08-01.

[260] Wikipedia. *Voxel*. `https://en.wikipedia.org/wiki/Voxel`. Accessed: 2020-09-23.

[261] Wikipedia. *Wasserstein metric*. `https://en.wikipedia.org/wiki/Wasserstein_metric`. Accessed: 2020-10-02.

[262] Guy Windreich, Nahum Kiryati, and Gabriele Lohmann. "Voxel-based surface area estimation: from theory to practice". In: *Pattern Recognition* 36.11 (2003), pp. 2531–2541.

[263] W. Wohlkinger and M. Vincze. "Ensemble of shape functions for 3D object classification". In: *2011 IEEE International Conference on Robotics and Biomimetics*. Dec. 2011, pp. 2987–2992. DOI: 10.1109/ROBIO.2011.6181760.

[264] Kaufui V Wong and Aldo Hernandez. "A review of additive manufacturing". In: *International scholarly research notices* 2012 (2012).

[265] Jin Xie, Meng Wang, and Yi Fang. "Learned binary spectral shape descriptor for 3d shape correspondence". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 3309–3317.

[266] Dong Xu and Yang Zhang. "Generating triangulated macromolecular surfaces by Euclidean distance transform". In: *PloS one* 4.12 (2009), e8140.

[267] Jiaqi Yang et al. "TOLDI: An effective and robust approach for 3D local shape description". In: *Pattern Recognition* 65 (2017), pp. 175–187.

[268] Yubin Yang, Hui Lin, and Yao Zhang. "Content-based 3-D model retrieval: A survey". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 37.6 (2007), pp. 1081–1098.

[269] University of Youk. *SHREC 2017 Point Cloud Retrieval of Non-rigid Toys (PRoNTo) Dataset*. `https://www.cs.york.ac.uk/cvpr/pronto/dataset.html/`. Accessed: 2020-09-08.

[270] Meng Yu et al. "3D model retrieval with morphing-based geometric and topological feature maps". In: *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.* Vol. 2. IEEE. 2003, pp. II–656.

[271] Juefei Yuan et al. "SHREC'18 track: 2D scene sketch-based 3D scene retrieval". In: *11th Eurographics Workshop on 3D Object Retrieval, 3DOR 2018*. Eurographics Association. 2018, pp. 29–36.

[272] Cha Zhang and Tsuhan Chen. "Efficient feature extraction for 2D/3D objects in mesh representation". In: *Proceedings 2001 International Conference on Image Processing (Cat. No. 01CH37205)*. Vol. 3. IEEE. 2001, pp. 935–938.

[273] Dengsheng Zhang and Guojun Lu. "Shape-based image retrieval using generic Fourier descriptor". In: *Signal Processing: Image Communication* 17.10 (2002), pp. 825–848.

[274] Juan Zhang et al. "Retrieving articulated 3-d models using medial surfaces and their graph spectra". In: *International workshop on energy minimization methods in computer vision and pattern recognition*. Springer. 2005, pp. 285–300.

[275] Lisha Zhang et al. "Survey on 3D shape descriptors". In: *FundaÃgao para a Cincia ea Tecnologia, Lisboa, Portugal, Tech. Rep. Technical Report, DecorAR (FCT POSC/EIA/59938/2004)* 3 (2007).

[276] Sicheng Zhao et al. "View-based 3D object retrieval via multi-modal graph learning". In: *Signal Processing* 112 (2015), pp. 110–118.

[277] Y. Zhong. "Intrinsic shape signatures: A shape descriptor for 3D object recognition". In: *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*. Sept. 2009, pp. 689–696. DOI: `10.1109/ICCVW.2009.5457637`.

[278] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. "Open3D: A Modern Library for 3D Data Processing". In: (Jan. 2018).

[279] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. "Open3D: A modern library for 3D data processing". In: *arXiv preprint arXiv:1801.09847* (2018).

[280] Tinghui Zhou et al. "Learning dense correspondence via 3d-guided cycle consistency". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 117–126.

[281] Emanoil Zuckerberger, Ayellet Tal, and Shymon Shlafman. "Polyhedral surface decomposition with applications". In: *Computers & Graphics* 26.5 (2002), pp. 733–743.