Spatial Relationship Based Scene Analysis and Synthesis

XI ZHAO

Doctor of Philosophy

Institute of Perception, Action and Behaviour

School of Informatics

University of Edinburgh

2014

Abstract

In this thesis, we propose a new representation, which we name Interaction Bisector Surface (IBS), that can describe the general nature of spatial relationship. We show that the IBS can be applied in 3D scene analysis, retrieval and synthesis.

Despite the fact that the spatial relationship between different objects plays a significant role in describing the context, few works have focused on elaborating a representation that can describe arbitrary interactions between different objects. Previous methods simply concatenate the individual state vectors to produce a joint space, or only use simple representations such as relative vectors or contacts to describe the context. Such representations do not contain detailed information of spatial relationships. They cannot describe complex interactions such as hooking and enclosure.

The IBS is a data structure with rich information about the interaction. It provides the topological, geometric and correspondence features that can be used to classify and recognize interactions. The topological features are at the most abstract level and it can be used to recognize spatial relationships such as enclosure, hooking and surrounding. The geometric features encode the fine details of interactions. The correspondence feature describes which parts of the scene elements contribute to the interaction and is especially useful for recognizing character-object interactions. We show examples of successful classification and retrieval of different types of data including indoor static scenes and dynamic scenes which contain character-object interactions. We also conduct an exhaustive comparison which shows that our method outperforms existing approaches.

We also propose a novel approach to automatically synthesizing new interactions from example scenes and new objects. Given an example scene composed of two objects, the open space between the objects is abstracted by the IBS. Then, an translation, rotation and scale equivariant feature called shape coverage feature, which encodes how the point in the open space is surrounded by the environment, is computed near the IBS and around the open space of the new objects. Finally, a novel scene is synthesized by conducting a partial matching of the open space around the new objects with the IBS. Using our approach, new scenes can be automatically synthesized from example scenes and new objects without relying on label information, which is especially useful when the data of scenes and objects come from multiple sources.

Lay Summary

3D virtual scenes, which are composed of virtual objects and characters, are important for applications such as virtual environments, 3D computer games and computer animation. A huge amount of 3D scene data is produced by amateur and professional artists on a daily basis and uploaded to the internet for public usage. In this thesis, two main problems relating to the management of such data are explored: how to let the computer understand the context of the 3D scenes and how to let the computer automatically produce realistic 3D scenes. The most important finding is that by using information on how the objects in the 3D scenes are located with respect to each other, the scene data can be better analysed and that scenes with more a complex context can be automatically produced.

Acknowledgements

I would like to express my very great appreciation to my supervisor Taku Komura for his patient guidance that enable me to accomplish this thesis. I would also like to thank my second superior Bob Fisher for his support and encouragement over these years. Advice given by Prof. Chris Williams and Prof. Alexander Belyaev has been a great help in the thesis revision.

I appreciate He Wang, Myung Geol Choi and Hubert P.H.Shum for their help. My research benefited a lot from many inspiring discussions with them. I thank my colleagues from office and CGVU Group. Especially I want to thank Adam Barnett, Joseph Henry and Peter Sandilands for their help with my English and research. I would miss the lunch time spent with them.

I would like to offer my special thanks to Xing Chen. She dragged me out of the office to watch birds and search for wild flowers. She also helped me with the proofreading a lot.

I acknowledge the funding of this study provided by the China Scholarship Council, Edinburgh University and TOMSY project.

Last, but most importantly, I would like to thank my family for all their support during my PhD study.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*XI ZHAO*)

Xi Zhao
16/9/2014

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

There is a growing demand for 3D virtual scenes, which are composed of objects, or objects and characters, for applications such as virtual environments, 3D computer games and computer animation. A huge amount of data is produced by amateur and professional artists on a daily basis and uploaded to the internet for public usage. Two main topics of research are emerging due to the increasing interests in such data: efficient data management based on the scene context, and automatic synthesis of novel scenes from example scenes.

Data Management To manage a large amount of scene data, a technique to automatically extract the key features of the scene is needed. As the data are usually not tagged, or tagged in an inconsistent manner, the contents and context of the scenes are not clear until the data is loaded into the software and visualized. As this process is very inefficient, a feature that describes the context of the scene is needed for indexing the data. Such a feature would also be useful for content-based retrieval, i.e., finding scenes similar to a given example from the database. In such cases, the system needs to rank the scenes in the database based on similarity of content and any other conditions imposed by the user.

The scene may also include dynamic components such as characters. This type of scene must be indexed based on not only the static arrangement of the objects but also the way in which the characters interact with the objects. For example, the user may wish to search for data where a character plays on a slide, or carries an object from one place to another in a room. In such cases, a feature that describes the time-varying relationship of the characters and objects in the scene is needed.

Scene Synthesis Artists or designers may wish to synthesize novel scenes based on the examples in the database. There are two topics of research in this area: automatic production of new 3D objects from the examples, and automatic arrangement of the objects in the 3D space. The former is an exciting topic that takes into account design, function and composition simultaneously. Many researchers in computer graphics are working on this topic and we will review previous works in Chapter 2. The latter can be considered as a problem of learning the "spatial relationships", which is a term we use in this thesis to refer to how objects are located with respect to one another in the scene.

In terms of arrangement of multiple objects in the scene, there are often rules that we need to follow. For example, a chair is usually located in front of a desk. If the spatial relationships between the objects can be extracted and encoded, this information can be useful for producing new scenes.

1.2 Problem Statement

As scenes are composed of multiple objects with different geometries, the spatial relationship between them becomes a key feature for analysis and synthesis of novel scenes. The objects are often located so that a group of objects function well as a unit, especially for spaces where humans conduct activities. In the previous chair-and-desk example, to add a character motion of "sitting down" to the scene, there must be enough space between the chair and the desk to let the body fit in well. The desk should also be within reaching distance. An ideal representation of this would be descriptive enough so that the system can evaluate all these aspects.

Despite the importance of the spatial relationships, existing methods [Fisher et al., 2011, 2012] describe the spatial relationships using rather simple representations such as relative vectors, relative orientation, contacts and Euclidean distance, which do not provide detailed information about the relationships. As a result, complex relationships such as "a hat hanging on a hook" or "a bird inside a cage" cannot be well described; the relationship "hang" cannot be well defined by contacts, distance, or orientation, especially when considering the wide range of geometry of both the hat and the hook. Issues can also occur even for comparatively simple relationships such as "a chair tucked under the desk", for example, high chairs only fit well with high tables, while small scale chairs for children do not fit well with office desks.

For dynamic scenes that include characters as well as objects, the representation must again be expressive enough to distinguish complex, time-varying relationships between the human body and objects, and robust against the wide variation of characters (morphology and geometry), objects (geometry, size and topology) and interactions (kinematics, contacts and timing). For example, characters can grasp a cup and drink the water in it in different ways: the hand grasps the cup according to the geometry and topology of the cup. If there is a handle, the character is likely to pass its fingers through the handle to hold it. But it will simply conduct a power grasp if there is no handle. The character might just slightly sip it if the water is hot, but may tilt the cup, even upside-down, if they are very thirsty or when there is not much water left in it. Although the overall context of these scenes are consistent, there is a chance that they could be recognized as different if existing representations about character movements based on joint angles or joint positions are used.

Due to the low expressiveness and robustness of existing representations of spatial relationships, synthesizing novel scenes from examples is not an easy task. In fact, existing methods rely heavily on manually tagged labels (such as “chair”, “bookshelf” etc) and pairwise relations (such as “a sofa is likely to be in front of a TV”) [Fisher et al., 2011, 2012; Yu et al., 2011]. Such labels are not always available and manually adding them is time consuming. Most importantly, even if such labels are available, their methods are inadequate to deal with complex interactions due to the lack of detailed spatial relationship information.

1.3 Goal of the Thesis

The focus of this research is to alleviate the problems of existing methods by developing a new type of representation of spatial relationships. We propose a representation called Interaction Bisector Surface (IBS). The IBS is a collection of points that are equidistant from at least two objects in the scene. It is a subset of the Voronoi diagram and describes the geometric and topological nature of the space between different objects in the scene. By computing a topological feature set called the Betti numbers, we can detect relationships such as encirclements, interlocks and enclosures, which characterize scenes such as a house surrounded by fences, a lady with a hand bag hanging on her arm, or an object contained in a box. The geometric nature of the relationships can be analysed using the shape of the IBS, the direction of its normal vectors, and the distance between the objects and the IBS. The computation of the IBS makes minimal assumptions about the forms of data input, which can be polygon meshes, skeletons, or point-clouds, making it applicable to a wide range of existing data.

Although Voronoi diagrams and medial axis type structures have been widely applied in shape matching and shape reconstruction, to the best of our knowledge, few applies them in scenes composed of multiple objects and characters. We observe that this rich feature can also be computed in the open space between objects and can describe the spatial relationship effectively.

The goal of this project is to validate the following claim:

The IBS is an effective representation of spatial relationship, which offers the following advantages:

- **Richness and expressiveness**: The IBS can encode and index complex spatial relationships.
- **Robustness**: It is insensitive to slight modifications in the geometry of the objects unless the objects are in close proximity.
- **Continuity**: Its shape gradually changes while the characters move in the scene.

We examine our claim by providing extensive experimental results on different types of scene data.

1.4 Thesis Overview

In this thesis, we propose the Interaction Bisector Surface (IBS), which is a subset of the Voronoi diagram, as a representation for the spatial context of 3D scenes.

The definition of the IBS and its computation, properties, and features that we use for scene matching and synthesis are described in Chapter 3.

We then show that the geometric and topological features of the IBS can be applied for scene classification and content-based scene retrieval. We show that the IBS-based features outperform simple relationship features used in previous works for classification of scenes composed of two objects. We also show that the IBS can be applied to the composing of a hierarchical structure that describes scenes composed of multiple objects, which is useful as a data structure for content-based scene retrieval. Based on the scene structure and the features of IBS, our system can find similar functional objects in the database, purely based on the relationship information. The results of these experiments are published in [Zhao et al., 2014]. More details can be found in Chapter 4.

We then show that the IBS can also be applied to the indexing of dynamic scenes where human characters interact with objects. We propose a multiresolution distance function based on the time-varying spatial relationships between a character and objects in a scene, which can effectively describe the context of the animation sequence. We present results of applying it to content-based retrieval of dynamic scenes and show that it outperforms traditional approaches. More details can be found in Chapter 5.

Finally, we propose a method to synthesize novel scenes composed of a pair of objects from an example scene. Given an example scene composed of two objects, the geometry of the open space is extracted and encoded by the IBS structure. Then a novel feature that we call shape coverage feature, which can effectively describe how a point in the space is surrounded by the environment, is computed. Finally, new scenes are synthesized by conducting a partial matching of the IBS and the space around the new objects. Our work on scene synthesis is in its early stage, but our preliminary experiments present some exciting and promising results. This work is the first of its kind, which synthesizes scenes with complex interactions. More details can be found in Chapter 6.

1.5 Contribution

The contribution of the thesis is that it suggests:

- A rich representation of the spatial relationship between different objects in a scene, which can encode not only its geometric nature but also its topological nature,
- an automated mechanism to build a hierarchical structure from a scene composed of multiple objects based on their spatial relationships,
- an effective context-based model retrieval approach that is purely based on the spatial relationships between the objects,
- a multiresolution feature that can elaborately encode dynamic character-object interactions,
- a method to synthesize novel scenes where multiple objects closely interact with one another,

- a scheme to index the open space around objects using a novel feature that we call the shape coverage feature,
- and a partial matching scheme for the open space around objects based on geometric hashing.

Chapter 2

Related Work

Both the concepts and mathematical models of spatial relationships have wide applications in different fields. The concepts stem from psychology and linguistics, while building mathematical models representing spatial relationships is a topic in computer graphics and computer vision. In this chapter, we first introduce the background of "spatial relationship representation", which is the focus of our research. We then review the research done on shape analysis, scene analysis and scene synthesis with respect to spatial relationship representations. Lastly, we also review the work related to the medial axis and bisector surface, as our main contribution is to propose a new spatial relationship representation based on the medial axis.

2.1 Background

Spatial relationship indicates the location of one object with respect to the location of a second object.

[Carlson-Radvansky et al., 1999]

Spatial relationship emphasizes how objects are positioned with respect to one another.

[Haining, 2003]

For each picture(example pictures shown in Figure 2.1), elicit one or more descriptions that represent what consultants feel is a natural response, in everyday speech, to the question Where is the [object pointed at by arrow]? (The task designed to elicit expression of spatial relationships.)

[Bowerman and Pederson, 1992]

This section introduces the background of spatial relationships, and gives readers

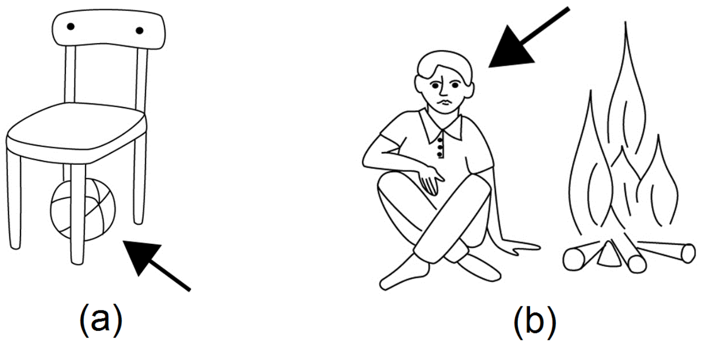


Figure 2.1: (a) An example of the relationship “under”, (b) An example of the relationship “beside”. Figures from [Bowerman and Pederson, 1992]

a clear idea that how our research fits into the big picture. We list research fields which care about the spatial relationship, and briefly introduce how they explain and use spatial relationships.

How people understand and express spatial relationships is widely studied by psychologists and linguists. Their research provides guidelines for designing and evaluating mathematical models for simulating human perception. Here we list some examples to show that our work actually benefits from these research. Feist [2000] mentioned that geometric descriptions “are prominent in both lexicographers’ definitions and linguists’ treatment of spatial terms”, although they “do not cover the range of use”. An example from psychology research is by Goldstein [2009]. They listed and analysed six laws which explain how perceptual groupings happen. Perceptual grouping means that human tend to recognize a scene at the group level when observing it from a global perspective, by grouping elements together based on proximity, continuation, uniformity, etc. These laws help us to explore ways of representing spatial relationships in large scenes. We apply the proximity law in our scene structure construction and also did a user study to analyse this law (Chapter 4). Xu and Kemp [2010] explored how spatial concepts are constructed from more basic components, and whether they can be constructed by a universal set of spatial primitives. These primitives, for example “above”, “beside”, “in”, “on”, etc. gave us an idea of the most common types of spatial relationship perceived by people, and help us to build ground truth for evaluating our representation (Section 4.1).

Qualitative spatial representation is an abstraction of the space where objects are located [Cohn and Hazarika, 2001] and traditionally it is pursued by researchers from many disciplines including geography, robotics, computer vision, and AI. It is a topological representation and can be extracted from attributes such as orientation, distance, size, shape, etc. Qualitative spatial reasoning, which is based on qualitative spatial representation, has wide applications in Geographic Information System(GIS), robotic navigation, high level vision, etc. An example which uses this type of representation for spatial scene retrieval is by [Bruns and Egenhofer, 1996]. To find the most similar configurations of a target scene, topological relations and qualitative relations based on distance and orientation were used for assessing the similarity between scenes.

Spatial relationships also attract the attention of biologists, as the interaction between molecules, proteins or other biological elements play a central role in numerous biological processes. The explanation, recognition and prediction of this kind of interaction can help for molecular recognition, protein docking, protein folding, etc. The docking problem, for example, with the geometric representation, can be boiled down to two main problems to solve: matching and scoring functions selection [Halperin et al., 2002]. One type of solution is “interface”-based algorithms. For example, the Voronoi diagram can be used to represent the spatial relation between two parts [Ray et al., 2005].

Spatial relationships are widely used in some branches of computer vision and computer graphics, including 2D image analysis, and 3D data analysis, 3D scene synthesis. Here the input data contains the spatial information of points (location of 2D or 3D vertices) and corresponding features. This information is combined and abstracted in different ways to compute features for further retrieval, classification and synthesis. As the topics of 3D shape analysis, 3D scene analysis and 3D scene synthesis are the most closely related work to this thesis, we give more detailed review for them in Section 2.2, Section 2.3 and Section 2.4 respectively.

Spatial relationships have been used in many different fields (summarized in Figure 2.2). However, most researchers work on how to use and analyse a given spatial relationship representation, rather than explore new representations of the spatial relationships. The traditional representations such as distance and orientation features have been proven useful and effective in many cases. But in biology and computer science, the data might contain complex spatial relationships which simple representations cannot encode well. For example, the shapes of the molecules and the interactions between them can be very complex, and the 3D scenes may also contain tangles. Currently not much work has been done to represent such complex relationships. Our work aims to fix this gap in the field, and propose a new spatial relationship representation. We explore the applications of our method in the field of computer graphics. Our method is also potentially useful for applications in other related fields.

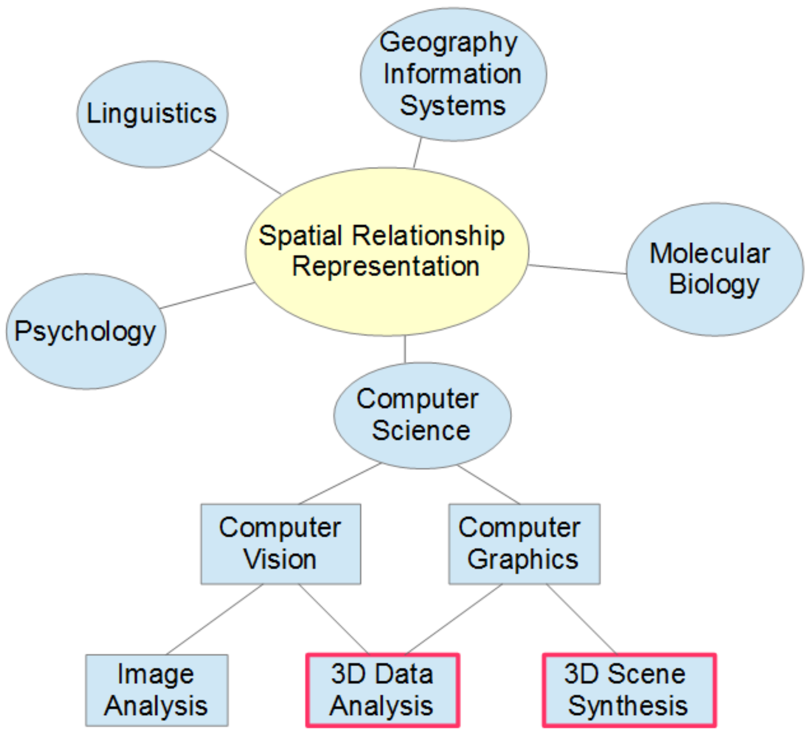


Figure 2.2: Summary of applications of spatial relationships. The topics of this thesis is highlighted with red borders.

2.2 3D Shape Analysis

Various methods have been proposed for feature extraction of both 2D and 3D shapes. We restrict our discussion mainly to 3D shapes. However, for completeness, we also include the algorithms which were first proposed for 2D, but have been extended for 3D shapes. We categorize the features used for shapes into three classes, based on the underlying level of richness of spatial relationship information: statistics-based features Section 2.2.1, spatial-map-based features Section 2.2.2 and skeleton-based features Section 2.2.3. In Section 2.2.4, we also introduce a newly emerging topic called shape structure analysis. This topic has attracted much attention recently.

2.2.1 Statistics-Based Features

Statistics-based shape feature extraction usually consists of two steps: first, compute local features for each vertex, second use the local features individually or compute a histogram over the whole shape. Next, we list some methods which use statistics based features, and explain how they the encode spatial relationships.

Zaharia and Preteux [2001] proposed using a feature called the *shape index* which is based on local curvature information. They build a distribution based on this feature which they called the *shape spectrum*. As the shape index is only computed from the current vertex and its immediate neighbours, this method does not consider the spatial relation between vertices which are far away from each other. Although similar shapes lead to a similar shape spectrum, there is a chance that different shapes also have similar shape spectrum. Another well known method is called the shape distribution [Osada et al., 2002]. Here they defined a range of shape functions which measure the spatial relationship between a certain number of random points sampled on the shape. Shape context [Belongie et al., 2000] at a vertex captures the distribution of the remaining points relative to it. This method was used for shape matching and recognition in both 2D [Belongie et al., 2002] and 3D [Körtgen et al., 2003]. Another similar local descriptor is the *spin image* [Johnson and Hebert, 1999], which is computed by projecting the shape to a local coordinate system generated from an oriented point. The resulting "relative position" is a representation of the spatial relationship between a local point and the rest of the shape.

As a histogram only provides statistical results based on the feature values, whether it can represent spatial relationships effectively depends on how much spatial relationships information is encoded in the local features.

2.2.2 Spatial-Map-Based Features

Spatial-map-based methods use or compute local features for each vertex, and then put the local feature into a "spatial structure" in which each element corresponds to a physical location or section of the data.

One common type of spatial structure is a spatial histogram. To avoid the loss of spatial information when putting local features into a histogram, the shape can be segmented spatially first, and then a histogram is computed for each segment and the histograms are all used. The *SIFT*(Scale-invariant feature transform) [Lowe, 1999] [Scovanner et al., 2007] used a spatial histogram of the image gradients in characterizing the appearance of a key point. Other examples which use spatial histograms include [Lazebnik et al., 2006] and [Ankerst et al., 1999]. The *bag of words* method was first applied to object recognition in [Csurka et al., 2004]. It represents an image as a histogram of "words" which are different local features, and does not consider any topological structure information between the words. To avoid losing spatial information, Lazebnik et al. [2006] augment the bag of words method with a *spatial pyramid*, which encodes the spatial information in a multiresolution way. Their method repeatedly subdividing the image to regular grids and computing histograms of local features at increasingly fine resolution. In [Ankerst et al., 1999], instead of segmenting the shape into regular grids, they used a *shape histogram*, which is the histogram based on a different type of partitioning of the space, for example the “shell” model, “sector” model or “spider web” model.

Spatial map methods in general are not rotation invariant unless the shape is normalized first and partitioned in a very careful way. Spherical harmonics were used to overcome this problem. Decomposing the shape function into its harmonics, Kazhdan et al. [2003] build a histogram by summing the harmonics coefficients within each frequency, and use this as the features of the shape. For volume data (the shape represented by voxel grid), they intersect the data with concentric spheres, and build a 2D histogram of decomposition parameters indexed by both the frequency of harmonic function and radius of spheres. But as the data in each concentric sphere regions are processed separately, the spatial relation along the radial direction is lost. Based on spherical harmonics, 3D Zernike descriptors [Novotni and Klein, 2003] encode shape coherence in the radial direction as well as the direction along a sphere.

The spatial-map-based method encodes the spatial relationship at a higher level compared with the global distribution method. However as it is either partially based on histograms (spatial histogram), or based on basis function representations (spherical harmonics transformation), it provides little direct detail for finding actual transformations for shape matching. This property limits this type of method mainly to be used for shape comparison.

2.2.3 Skeleton-Based Features

Extracting the skeleton/graph structure for a 3D shape is a well studied area. It is about building a topological structure of the data in a very abstract way. This type of abstraction can be useful for partial matching, shape correspondence, shape abstraction, shape deformation, and of course measuring the similarity between shapes.

The process of this type of methods usually includes shape segmentation and building relationships between the segments. The main point is that instead of considering spatial relationships between pairs of points, we can compute the relationships between larger segments of the shape. There are various methods proposed. They can be classified into two types according to the representation of the abstraction model: the curve skeleton and the medial axis. The curve skeleton aims to construct an extremely thin skeleton or a curve-like representation, while the medial axis in 3D is usually a non-manifold mesh.

Curve skeleton A curve skeleton can be built in two ways:

The first way is to segment the boundary mesh, and use the adjacency between segments to build the skeleton. The *Reeb graph* [Hilaga et al., 2001], a skeletal structure defined based on a height function over the shape manifold, belongs to this type. It can be constructed in a multiresolution way to capture different levels of details. In a follow-up work, Tung and Schmitt [2004] augment the Reeb graph with geometric features for each node which provide a richer description of shape compared with the methods which only use the topology structure. Another example is [Katz and Tal, 2003], which extracts a skeleton structure from a minimum-cut-based mesh decomposition. A similar solution was used in computer vision for image retrieval based on the bag of words method, which segments an image to "words" and builds topological structure between the words in an implicit way. Previously we mentioned that a spatial histogram can be used for the bag of words method to avoid losing spatial information. Another way is to build vocabulary for spatial relationships. Bronstein and Bronstein [2010] and Ovsjanikov et al. [2009] considered the adjacency between pairs of words and represent the spatial relationship as the frequency of appearance of nearby words.

The second way is to build a curve skeleton without segmenting the mesh. Sundar et al. [2003] used the thinning method proposed by Gagvani and Silver [1999] for skeleton computation. This method is semi-automatic: two end points of the skeleton are defined by the user, and then valid inside voxels are added in between iteratively. Another way is to use surface contraction methods [Au et al., 2008; Jiang et al., 2013]. This type of method can build a multiresolution structure by stopping the progressive process with different thresholds. Recently, researchers are also interested in building graph structures from incomplete and noisy point cloud data [Huang et al., 2013]. We refer the reader to [Cornea et al., 2007] for a more detailed review.

Medial axis The medial axis, which was first proposed by Blum [1967], is a standard way for extracting the skeleton of a 2D shape. But the medial axis of a 3D shape is normally a non-manifold surface, and it is well known that it is sensitive to small perturbations on the boundary. So the unsimplified medial axis is not very convenient to use directly for shape processing. But as medial axis points are naturally located in the centre of the shape, they can be further pruned to be a more simple curve skeleton.

For example, in [Chang and Kimia, 2009], they build a graph, the nodes of which are the *shock points*: the points have more than two closest points on the mesh. As the core algorithm of this thesis is closely related to the medial axis, we review the medial axis related methods in more detail later (Section 2.5).

Both the curve skeleton and medial axis approaches have their advantages and disadvantages. The curve skeleton is a simple structure which is easy to save and use, however the relationship between skeleton and mesh might be overly compressed and lost. The advantage of the medial axis is that each medial vertex encodes very rich spatial information which is quite useful for applications such as reconstruction and manipulation, while the computation of the medial axis is more costly and requires more post processing. A notable recent work by Thiery et al. [2013] tried to compromise between these two types of representations. They develop a structure called a *spherical skeleton*, which is based on the contraction of the original mesh, and also keeps the radius information at medial axis points.

2.2.4 Shape Structure Analysis

The features introduced in the previous three sections belong to traditional shape analysis and naturally lead on to a higher level technique: discovering shape structures. Shape structure analysis is about analysis of relations between shape parts. Although parts decomposition of shapes and building relationships between parts are a very old topics in computer vision, e.g. [Dickinson et al., 1992; Fischler and Elschlager, 1973], in graphics, the structure analysis of 3D models, though attracting a lot of attention, is only an emerging area of studies. The underlying reason for the popularity of this research topic is that people are not satisfied with shape comparison/retrieval and low level editing like detail-preserving shape deformation. To reduce the burden on designers and modellers, methods are needed for higher level deformation, editing and synthesis of 3D shapes with large variances in geometry and topology. For this purpose, structure analysis is needed.

Shape structure is about the arrangement and relations between shape parts [Mitra et al., 2013]. To build a shape structure, we need to first find the parts, and a suitable representation for them. A part can be a segment of the original mesh or an abstraction. It can either be specified by the user or learned from a set of shapes. Then the relations between the parts should capture how they are correlated. The relations play the most important rule for the shape structure.

Several methods of analysing the structure of man-made objects have been proposed during the last ten years. These methods are focused on building structures for single objects. The spatial relationship representations they use are mainly based on contact or adjacency information between shape parts within the object. Here we introduce the most influential work in this topic. Readers are referred to [Mitra et al., 2013] for a more comprehensive review.

Symmetry is quite an important feature for man-made objects. Wang et al. [2011] compute hierarchical pyramids of single objects based on symmetry and contact information. To find the nodes of the hierarchical graph, they use normalized cuts guided by shape concavity [Golovinskiy and Funkhouser, 2008], and then use a symmetry-driven enhancement strategy to improve the symmetry of the segmentation. Symmetry was also used in [Jain et al., 2012]. Here the shape is pre-segmented and the symmetry and proximity relations between parts are pre-analysed.

Using co-analysis-based method to build a general structure for a set of shapes is also a popular topic. Given the prior that a set of shapes are related, co-analysis was believed to be able to extract richer and more reliable information for each single shape. Sidi et al. [2011], Golovinskiy and Funkhouser [2009] and Huang et al. [2011] used co-analysis to find stable segmentation and correspondence between parts. Kalogerakis et al. [2012] produced a probabilistic model of the shape structure for various types of models. Given segmented models, the geometric features and adjacency information of the shape components are learned and their relations are represented by probability. This shape representation is convenient for producing plausible combinations of components from existing shapes. van Kaick et al. [2013] used a co-hierarchical analysis to learn the structure of a set of shapes. They proposed an unsupervised cluster-and-select scheme which learns both the structural similarity and variability in the set.

Zheng et al. [2013] also built a graph structure from single objects based on the spatial relationship of the components. The difference here is that they further extract sub-structures from the shape graph. This is useful as the sub-structures keep the mutual relationship between parts. Similar sub-structures usually imply similar functionality.

2.2.5 Summary and Discussion

Figure 2.3 shows a summary of the shape feature extraction algorithms from the point of view of how the spatial relationship is encoded. From left to right, more spatial relationship information is encoded.

What type of algorithm should be used depends on the application: global distribution methods and spatial map methods usually give low dimensional features, which are fast to compute and convenient to use for shape comparison and retrieval; while the skeleton based methods are ideal for animation purposes, shape correspondences and reconstruction.

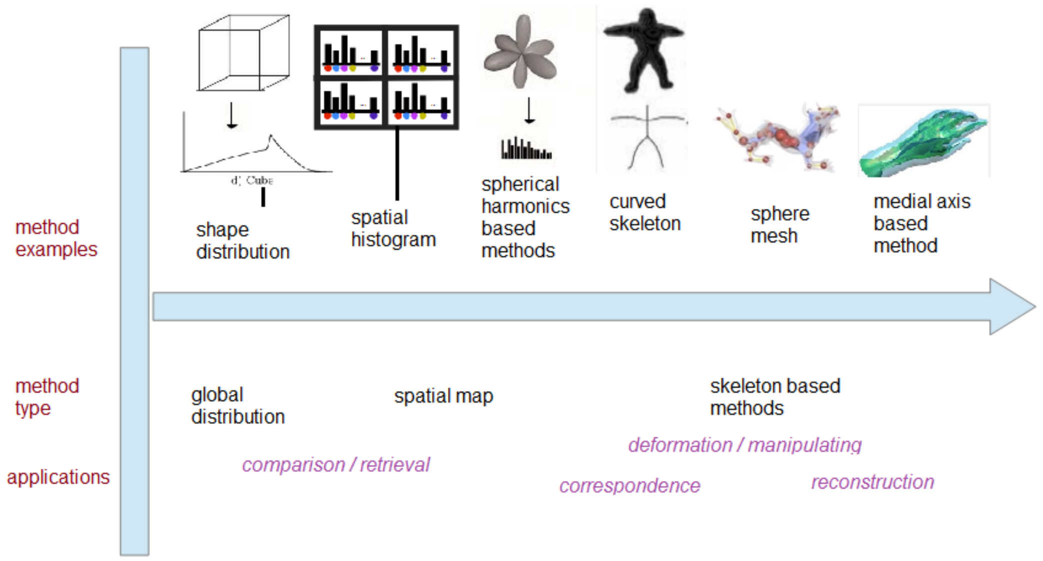


Figure 2.3: An overview of spatial relationship representation methods. The amount of spatial relationship informations encoded increases from the left to the right.

A shape is essentially a collection of spatial relationships. It is about how the shape elements are laid out in the 3D space. More specifically, it is about how the shape elements are located with respect to other elements. We find that the most well known features and representations of 3D shape are actually about representing and organizing spatial relationships for the shape elements. For example the shape distribution feature uses the distance between each pair of points in the shape. Each distance is a simple representation of the spatial relationship between two points. The resulting histogram is an accumulation of the pairwise distances. The only difference between our research and single shape analysis is that, the data we consider are scenes with separate objects while a single shape is normally a single connected unit. If we consider a shape as a set of separate elements, then there is no fundamental difference between a single shape and a scene with multiple objects. As the medial axis is a richer representation for a shape, the medial axis can also be used for building a richer representation of spatial relationships. The new representation we propose in this thesis is based on this observation.

2.3 Multi-Part Scene Analysis

A scene is a 2D or 3D space containing at least one object or character. Single objects can be considered to be a specific type of scene. In this section, we restrict our topic to scenes which contain more than one object. More specifically, we first introduce the ways to represent static scenes with multiple objects, then we talk about scenes which contain character-environment interaction and finally we look at character-character interaction.

2.3.1 Analysis of Static Multi-Object Scenes

Detailed and natural looking 3D scenes are useful for games, animations, virtual worlds, indoor design, etc. Manually making such scenes can be tedious and time consuming. Recently, online 3D model repositories have attracted more and more attention, for example the Google warehouse [Navigation Ltd, 2006] and Stanford scene database [Fisher et al., 2012]. These types of databases contain not only single model data but also multi-part scenes. To make use of this data, automatic scene processing algorithms are needed. The key for scene processing is the representation of scene structure. This research topic is just emerging and there have been few works in the area. Next we list the most important progress recently.

Fisher et al. [2011] proposed to construct scene graphs based on contextual groups and contact information between objects. The relationships they used include enclosure, horizontal support, vertical contact, oblique contact. These relationships are detected by distance or angle between surfaces of different objects. Paraboschi et al. [2007] use the distance from the barycentre, height distance and geodesic distance as the metric and compose a graph Laplacian to encode the relationship of adjacent objects. The spatial relationship are highly abstracted by simple binary information of contacts. Yu et al. [2011] encode the relationships between furniture using metrics such as distance, orientation and ergonomic measures. The complex interactions are manually labelled by the users in these studies due to the difficulty of automatically learning them. Fisher et al. [2012] learn contexts from example scenes by using Bayesian networks and mixture models. The relationship between adjacent objects are represented by relative vectors, and are compared by bipartite matching.

Similar to the single model processing, the processing of multi-object scenes is also about scene retrieval, finding correspondences, editing, adaptation and synthesis. Although processing a single object and a scene are conceptually similar, in most cases we cannot directly apply the methods for single objects to scenes. The difference between a single object and a scene is that firstly, the "parts" are different. For a single object, each part is usually a shape segment with simple geometry, while the part in a scene is more likely to be a single object or a group of objects. Secondly, the relationships between parts are different. Parts of a single shape are usually not isolated, they would always connect to at least one other part of the shape. But the parts in a scene are more likely to be separated while keeping certain spatial relationships between them. If we want to segment a scene, a shape segmentation algorithm may not be applied directly. If we want to extract scene structure, the simple relation representation used for single objects may not be enough. Also, for further processing the scene data for adaptation or synthesis, a new optimization strategy would be needed.

2.3.2 Analysis of Character-Object Interaction

Human motion analysis (both in 2D and 3D) is a highly active research area due to its application for surveillance, medical studies, video indexing and animations. By processing image sequences or 3D animations, which include the movements of people, it aims to recognize what people are doing and how the action takes place. Although human-action recognition has received considerable attention, only a few researchers consider the interaction between characters or between a character and the environment. There are two related topics which consider character-object interaction: one aims to recognize human motion, the other is for object recognition.

Environment is an important clue for understanding human motion. There are several studies that use "textual description" for representing human actions in an environment. Bobick and Pinhanez [1997] describe different conditions of scenes by applicability rules (for example "in-contact") and identify which applicability rules have been satisfied. Thus, they use the context as well as visual information to acquire a high-level understanding of human-object interaction. In the work by Ayers and Shah [1998, 2001], they developed a system analysing the actions of people in a room. The system reduces a video sequence into a smaller series of key frames and the layout of the scene in key frames is detected by using low-level computer vision techniques. With prior knowledge and the detected changes in the scene, their system is able to recognize the actions, such as entering the room, opening a cabinet, etc. However, as the recognition conditions for each action must be defined specifically by using thresholds, recognizable actions are limited.

Another topic of research is in using the “affordance” information to recognize objects in a scene. Affordance is defined as: “*A property of an object or an aspect of the environment, esp. relating to its potential utility, which can be inferred from visual or other perceptual signals; (more generally) a quality or utility which is readily apparent or available*” [Simpson and Weiner, 1989]. It is closely related to the functional property of an objects and is essential for learning object categories [Grabner et al., 2011].

Many methods were proposed for extracting affordance related features. They consider the interaction between the character and objects and used for a description of the context of the scene. Peursum et al. [2004, 2005] proposed a feature called the *interaction signature*. Their system extracts a stick-figure skeleton from a character’s silhouette. The interaction signature encodes the relative position of regions in the scene with respect to the skeleton. In [Stark et al., 2008], given a guidance video which contains the human-object interaction, they first segment the object out and then detect the parts of the object where the interaction happens. They use the *k-Adjacent Segments* [Ferrari et al., 2008] feature to represent the shape of the interaction part. This representation, which they called the *affordance cue* is further used for functional object classification. In [Kjellström et al., 2011], their system learns the functional property of objects based on both object feature and posture feature. The interaction is represented by correlation information extracted from the object and posture features of the training data. Their method is applied to guide a robot hand to perform an action with the same effect.

The affordance principle can also be applied for multiple part scenes or even 3D scanned scene data. Aksoy et al. [2010] proposed to model the frames in a video as a graph, whose nodes are the segmented regions and whose edges are the relationship between them. The topological changes in the graph can be used to detect the structural changes in the scene, which characterise the interaction. In recent work by Grabner et al. [2011], they extend the scope to 3D data. To classify the 3D objects by their functional property, they put an actor into the scene who performs an action. They detect the polygons from the character model which are closest to the objects, and build a Gaussian model based on the distance and the intersection between the polygon pairs. The detection of the functionality features for a new scene are formulated as a probability estimation of all different transformations which locate the character model into the scene.

Most of the previous research related to character-object interaction was in trying to analyse the interactions within video data. The representations they use are mostly the contact-, adjacency-, or distance-based measures between the object and the character body parts. These methods normally heavily depend on the recognition of body parts and the objects. As more 3D interaction data is available and 3D cameras like the Kinect are becoming more popular, analysis of 3D data will attract more attention. The 3D interaction motion data is different from video data. With device like the Kinect, it is quite possible to get the labelled character data, which means the body parts can be recognized more easily. To analyse this type of data, new methods are needed for measuring the approximation/adjacency and distance between body parts and objects, because simple distance measures or contact/non-contact types of representation are not enough to represent complex interactions. The new representation of spatial relationship we proposed can be applied for this problem. In Chapter 5, we explain how we analyse the character-object motion based on the new representation.

2.3.3 Analysis of Character-Character Interactions

Most of the previous research was aimed at understanding interactions in remote scenes at a semantic level: for example, the interaction between different characters could be "approach" "depart"[Park and Aggarwal, 2004] and "walk together" [Oliver et al., 2000]. What is more, the majority of the existing techniques either represent the character as a point and compute features based on its trajectory [Oliver et al., 2000] or as a simple moving box and achieve recognition by interpreting the interaction types between the bounding boxes [Park and Aggarwal, 2004, 2000]. A typical example in crowd animation field is [Kwon et al., 2008]. They handled the spatial relationships between characters in group motions by encoding the neighbourhood formations and individual trajectories as Laplacian coordinates [Sorkine et al., 2004]. When editing the trajectories, the relative spatial arrangements of characters are preserved by applying Laplacian mesh editing techniques.

Description and understanding of person-to-person interaction at a detailed level with information about individual body parts has rarely been addressed, although there have been some recent works which take into account spatial relationships in more detailed ways. Park and Aggarwal [2000] represent human interaction as the change of the relative configuration between two characters in space and time. The K-nearest neighbour classifier is applied to the parametric human-interaction models for recognition. They later published another paper [Park and Aggarwal, 2004] providing a natural-language description of several human interaction motions. The interaction is defined based on the distance and orientations of body parts. Ho et al. [2010] used a structure called an *interaction mesh* to represent the spatial relationship between nearby body parts. The interaction mesh is a volumetric mesh defined by the joints of the characters and the vertices of the objects/environment. They proposed an automatic method that used the interaction mesh for editing and retargeting motions of close interaction with/without tangles. Tang et al. [2012] encoded the interaction of multiple characters in a similar fashion. They applied Delaunay tetrahedralization to the joints composing the character skeletons and define a metric based on Hamming distance for comparison.

These methods are potentially useful for producing character animation and controlling characters/robots to accomplish tasks. Their representation of spatial relationships is also a useful exploration for the problem of representing complex relationships, which is the main focus of our research.

2.3.4 Summary and Discussion

The previous research we introduced in this section emphasized and led to one conclusion: spatial relationships are important for multi-part scene analysis. These methods provide us with ideas of the most important information which should be encoded by the spatial relationship representation. Analysis of 3D scenes is still a young research field. The work of Grabner et al. [2011]; Ho et al. [2010]; Peursum et al. [2004, 2005] are the first few steps in the topic of complex relationship representation. In this thesis, we follow in this direction. We propose a more general representation which can be applied for both static scenes and character-object interaction scenes.

2.4 Scene Synthesis

In this section, we review data-driven methods of scene synthesis. To produce new scenes from existing ones is only one type of scene synthesis method, but it is an important step towards the long-standing goal of semantic scene/shape processing.

2.4.1 Single Object Synthesis

Single object synthesis is used to produce new objects based on input examples. For synthesizing a new object, the input can be a single example or a group of examples. Here we mainly focus on the methods based on a group of input examples, and then briefly introduce methods which synthesize new scenes from a single example.

Single object synthesis methods normally start from structural analysis. In some cases the structure information can also be defined by the user [Funkhouser et al., 2004; Sharf et al., 2006]. The result of this step is the shape part segments and the relationships between them, which provide the object structure information. The structure can then be used as a guideline to produce new models, by deforming and shuffling model parts. The "shuffling" step includes two main steps: first is finding a qualified part to shuffle, which requires the sub-part comparison and selection; second is gluing parts together, which mainly requires morphing parts to make sure the result looks natural when they are glued together. Next we list some recent work with more details.

Modelling based on examples was pioneered by Funkhouser et al. [2004]. Given a user defined replacement segment, their system applies partial matching between the segment and models in the database. Their matching is based on the earthmover's distance which considers the minimum cost to transform points from one shape to the other. To attach two parts, they first find correspondence between the boundary contour of the two parts, then add edges to connect the corresponding vertices and then smooth the region around the seams. A similar algorithm was proposed in [Sharf et al., 2006]. They combine the processes of building correspondence and smoothing by using a soft-ICP algorithm.

Jain et al. [2012] proposed a method to interpolate complex shapes. They first build hierarchical structures of the input shapes based on symmetry, and then find the correspondence between parts based on these structures. To produce new shapes, they blend corresponding shape parts of the gives shapes by using linear interpolation to produce new parts. To assemble new parts, they use a mass-spring system to enforce the contact constraints between parts. Blending two shapes can be interesting but also restrictive. Their system may produce an improper shape and cannot guarantee the plausibility of the result.

Another type of method is to shuffle the similar parts of existing objects. There are several strategies proposed to select the parts for a new shape. Instead of simply producing new shapes from one group of shapes, Xu et al. [2012] used a generative scheme which can produce a new generation from a previous generation, while considering the user's preference. They used many-to-many part exchange and also proposed a part mutation algorithm to enhance the variety. Kalogerakis et al. [2012] trained a probabilistic model on a set of segmented shapes. These model represents probabilistic relationships between the features of shape parts. To synthesize new models, they build tree structures in the following manner: start from an empty root node, the tree is then continuously expanded by choosing nodes for next level based on a random variable. To snap parts together, as in Xu et al. [2012] they also use contact as constraints. They find the initial placements by optimizing the matching between the contact points. They further locally smooth the shape to glue the two parts. Instead of shuffling single parts of a shape and selecting plausible results from a set of combinations, Zheng et al. [2013] exchange stable sub-structures which keep the functionality of the shape. To glue components, they also apply similar optimization as in [Kalogerakis et al., 2012].

The synthesis of models can be based on only a single object. Inverse procedural modelling was used in [Bokeloh et al., 2010] to abstract the structure present in an example into a set of rules. They then synthesize novel models based on these rules. For 3D model retargeting, which aims to generate various models while keeping the local structural style consistent, the input model is difficult to parsed by grammar-based methods. In this case, the structure extraction of the input model can usually be done in a semi-automatic way with users helping to define: the hierarchy [Lin et al., 2011] or constraints for variation [Bao et al., 2013]. Single input and rule-based synthesis methods are mainly applied to building or urban design problems, as the variance/duplication of parts make more sense for this type of data. While for man-made models like chairs or planes, which usually have fixed structure, using data-driven methods is more promising for plausible results.

2.4.2 Multi-Object Scene Synthesis

There are two main types of methods for synthesizing scenes with multiple objects. Rule-based methods and data-driven methods. We first review the rule-based methods, and then introduce the data-driven methods.

There are many approaches to generate furniture arrangements by following specific rules. This type of method consists mainly of two steps: first, provide the object with descriptions of their spatial relationship with the rest of environment; second, add objects to the scene while keeping the spatial relationship constraints.

The spatial relationship constraints of an object usually include geometric features of the shape [Akazawa et al., 2005; Germer and Schwarz, 2009], for example the size, bounding box and occupancy space; its semantic relation to other objects [Akazawa et al., 2005; Germer and Schwarz, 2009], for example "having parent-child type of relation with another object" or "being supporter of another object"; its qualitative relation [Akazawa et al., 2005; Larive et al., 2004], for example "in front of" and "contact with"; and also the general physical constraints [Germer and Schwarz, 2009; Larive et al., 2004], for example "not overlapping with others" and "keeping horizontal direction". Anthropometric constraints can also be used as guidelines [Merrell et al., 2011], for example, a scene's ability to support circulation, conversation, visual balance, etc.

To produce new scenes, Akazawa et al. [2005] added objects into the scene one by one randomly while keeping the constraints. Germer and Schwarz [2009] also used an incremental strategy: they consider each object as an agent, each time an agent seeking to attach itself to a parent object. Larive et al. [2004] used a stochastic method to place objects in a more realistic way. When users want to insert an object into a room, they first use the stochastic method to place the object in a possible location and then try to find a better location by trying several small displacements. Merrell et al. [2011] interactively produce suggestions when users pick the set of elements. They represent the furniture layout guidelines as terms in a density function and treat manual placement of pieces as subspace constraints. As many acceptable configurations are created, they used stochastic sampling to suggest layouts.

While most of the previous work uses incremental methods or optimization over a predefined set of objects, Yeh et al. [2012] formulate the space of layouts as a probability distribution and use a sampling algorithm to synthesize layouts. Their system can produce scenes without specifying the number of objects. As the probabilistic model is manually designed, their system can only generate examples that were expressed by the model.

If good quality examples are available, it is possible to learn from the input data and do a data-driven synthesis. Yu et al. [2011] proposed a method to learn the spatial relationships between different pieces of furniture based on ergonomic factors. The spatial relationships they use include the distance and orientation to the nearest wall, support information (if supported by floor or other objects) and pairwise relation to other objects (selected by user). They then defined a cost function based on the spatial information which considers accessibility, visibility, pathway between doors, similarity to the training data and pairwise relation between objects. The global optimization (minimization of the cost function) was computed by simulated annealing. Fisher et al. [2012] proposed a system which learned all relationships automatically from data. They used Bayesian networks and Gaussian mixtures models to represent position and orientation relationship between all pairs of object categories. To create new scenes, they first sampled from the Bayesian network to find a set of objects occurring in the scene and assign the object a valid location, and then they do global optimization by using simulated annealing to refine the locations iteratively.

Synthesizing scenes with multiple objects is more difficult than synthesizing single objects. The reasons are as follows: firstly, it is difficult to classify scenes, as they have no fixed structure. It is very difficult to build correspondence between scene elements, so the previous "shuffle parts" method cannot be used. Secondly, it is difficult to build a structure for a scene. For a single model, we can consider segments as nodes, and their connection/contact as edges to build the graph structure. But the objects in a scene are not necessarily in contact with one another, so building a graph structure is less straightforward. Thirdly, as it is possible that there is no contact information which can be used, how to "glue" two objects to make new scene is not as easy as for single objects.

2.4.3 Summary and Discussion

Scene synthesis is quite useful as it can reduce the labour of modellers, and enable people without a modelling background to produce their own scenes easily. The synthesis of single models and synthesis of big scenes make use of different techniques. However, they share the similar basic idea: to find the spatial relationships between parts and reassemble the parts together to make new scenes. As spatial relationship representation is so important for scene synthesis, the plausibility of the synthesis results is actually an evaluation of the representation quality. We also apply our new representation to scene synthesis, and mainly focus on the pairwise interactions.

2.5 Medial Axis and Bisector Surface

The medial axis, which was first proposed by Blum [1967], has attracted a huge amount of attention as it has wide applications. Here we use the definitions given in [Lieutier, 2004]. Given a bounded open set O, the medial axis is the set of points which have at least two closest points on the boundary ∂O. Examples of medial axis for 2D and 3D shapes are shown in Figure 2.4.

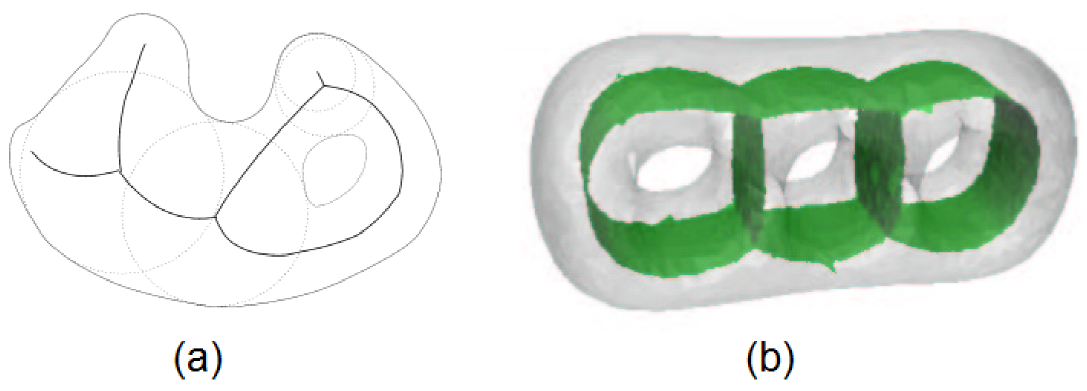


Figure 2.4: Medial axis examples: (a) the bold line is the medial axis for the 2D shape with the thin line boundary [Lieutier, 2004], (b) the green mesh is the medial axis for the 3D shape "3 hole torus" [Dey and Zhao, 2004].

In this section, we focus exclusively on the medial axis, because the new spatial relationship representation we propose is computed based on the medial axis. This new representation is actually a subset of the medial axis for open space.

2.5.1 Computation

There has been considerable work done on medial axis computation. Regarding their methods for finding the medial axis elements, they can be classified into two main categories: the accurate method and the approximation method.

Next we introduce the core idea of each type of method with some typical examples. Please note that for the purpose of brevity we list only a few of papers in this area. The readers are referred to the papers [Attali et al., 2009; Okabe et al., 2009] for a more comprehensive review of medial axis computation.

2.5.1.1 Accurate methods

The exact computation of the bisector surface can be used to compute an accurate medial axis, for example in [Culver et al., 1999; Hanniel et al., 2007]. The bisector of two geometrical elements (points, curves, surfaces, etc.) is the locus described by a variable point that moves to remain equidistant with respect to these elements [Farouki and Johnstone, 1994]. In principle, an analytical solution of the bisector surface of such a set (boundary elements) can be computed by using algebra. Although there are works which tackle the case of algebraic curve segments whose bisectors have rational parameterizations [Elber and Kim, 1997; Farouki and Johnstone, 1994], the algebraic difficulties in computing the bisector of boundary elements are significant, and satisfactory implementations are rare [Attali et al., 2009].

Many practical methods for computing the medial axis of 3D shapes are based on a tracing approach. This set of methods are usually based on the observation that medial axes consist of different types of elements: sheets, seams and junctions, and they connect to each other by following some certain rules. A sheet, which is a trimmed quadric surface, is produced by two boundary elements. A seam, which is an algebraic curve with rational coefficients, is produced by three or more boundary elements. A junction is a point which is produced by four or more boundary elements. Based on the classification of medial axis elements, Culver et al. [1999] and Sherbrooke et al. [1995] build tracing methods to produce an accurate medial axis in the following manner: Starting from a junction point, they trace out all adjacent seams, and check the feasibility of them. They start tracing from every junction point, and get all the elements of the medial axis. This result needs to be trimmed, as the medial axis between adjacent boundary elements need to be deleted. Finally, they can get the continuous representation of the medial axis of a 3D polyhedra. The difference between these two methods is that Sherbrooke et al. [1995] uses "offset sweep", a hypersurface one dimension higher, for all boundary elements, and then the intersection between sheets determines the medial axis. Culver et al. [1999] compute the accurate bisector surface between boundary elements based on exact arithmetic. Milenkovic [1993] and Reddy and Turkiyyah [1995] also use tracing algorithm to follow the edge of the medial axis. Instead of considering boundary elements such as vertices, line segments and faces, they subdivide the boundary to tetrahedrons, and calculate the medial axis vertices by finding the centroids of the tetrahedrons one at a time. Yang et al. [2004] use a sampling maximum ball and the separation angle between samples to find medial axis points. They also use a tracing algorithm to build the medial axis.

2.5.1.2 Voronoi diagram based approximation method

The second set of methods are discrete method which approximates the medial axis from a discrete representation of the shape. These types of methods normally start from sampling points on the shape boundary, and then use the Voronoi diagram or Delaunay triangulation to approximate the medial axis.

Denote the Euclidean distance between two points p and q by $dist(p,q)$. Let $P := \{p_1, p_2, ..., p_n\}$ be a set of n distinct points in the plane; these points are sites. Voronoi diagram of a set of points P is defined as the subdivision of the plane into n cells, with the property that a point q lies in the cell corresponding to a site p_i if and only if $dist(q, p_i) < dist(q, p_j)$ for each $p_j \in P$ with $j \neq i$ [De Berg et al., 2000]. Computing the Voronoi diagram is an important topic in computational geometry. There are a large body of methods which have been proposed, roughly they are: incremental algorithm [Imai, 1996; Milenkovic, 1993; Reddy and Turkiyyah, 1995], the sweep line algorithm [Fortune, 1986], the wavefront based algorithm [Held, 1994], the divide and conquer algorithm [Lee, 1982], and the qhull algorithm [Barber et al., 1996].

One type of method approximates the medial axis of an object by sampling points on the surface of the object, and computing the Voronoi diagram of these points. The theory behind this type of method is that a subset of Voronoi vertices of the shape converges to the medial axis as the sampling rate increases. Amenta et al. [2001] select the subset of Voronoi vertices, called *poles* to approximate the medial axis. The poles of a site are the furthest vertices of its Voronoi cell in the interior and exterior of the shape. They connect poles based on their power diagram cells to form the medial axis. Dey and Zhao [2004] approximates the medial axis from the Voronoi diagram in a scale and density independent manner. They used the *angle condition* and *ratio condition* to select the Voronoi facet to form the medial axis. They proved that the medial axis they produced converges to the exact medial axis when the sampling density on the shape approaches infinity.

Delaunay triangulation which is the dual of the Voronoi diagram is also commonly used for computing the medial axis. Yu et al. [1991] proposed an algorithm based on the Delaunay triangulation of the points on the boundaries. The Delaunay triangulation result consists of a set of tetrahedrons, the circumscribing spheres of which are called Delaunay spheres. They use the centre of the Delaunay spheres to approximate the medial axis. Sheehy et al. [1995] also use Delaunay triangulation to find the medial axis points. Here instead of only computing a discrete set of points, they build edges between medial vertices using a tracing algorithm.

To summarize, the advantage of this type of method is that it can work on any type of boundary representation, and does not require the volume grid structure of the 3D shape. The disadvantage is that deriving any error bounds on the output is difficult and overall complexity is not well understood [Hoff et al., 1999].

2.5.2 Simplification and Stability

The instability of the medial axis with respect to standard measures such as the *Hausdorff distance* is well known. Small modifications in an input shape can induce large modification of its medial axis. More specifically, small modifications in the shape boundary results in fluctuating branches, while the rest of the medial axis is unchanged. This observation is described as the *semicontinuous* nature of the medial axis [Attali et al., 2009].

The simplification of the medial axis is usually a step to delete the sensitive part of the medial axis. This process can be used to increase the stability of the medial axis, and sometimes is referred to as *pruning*. Following the pipeline in [Attali et al., 2009], to approximate the medial axis of shape X, we first find Y which is an approximation of the shape of X. Second step is to construct $M(Y)$, which is the medial axis or the Voroni diagram for Y. Third, pruning is done to find a subset of $M(Y)$ which is the approximate medial axis of X. We show the whole process in Figure 2.5, the most challenging step of which is the pruning process.

$$\begin{tikzcd}
X \arrow[r, "\textsc{Approximate}"] \arrow[d, dotted, "\textsc{Approximate Medial Axis}"'] & Y \arrow[d, "\textsc{Construct Medial Axis}"] \\
\mathcal{P}[\mathcal{M}[Y]] & \mathcal{M}[Y] \arrow[l, "\textsc{Prune}"']
\end{tikzcd}$$

Figure 2.5: An approximation $P[M[Y]]$ of the medial axis of a shape X can be found by the medial axis of a shape Y approximating X. Figure from [Attali et al., 2009].

The 2D medial axis can be approximated by simply selecting the Voronoi vertices which are located inside the shape [Brandt, 1994]. The pruning process is more complex for 3D, as in 3D not all the Voronoi vertices are located close to the medial axis. In [Amenta et al., 2001], they defined a special type of Voronoi vertex, the *poles*, which are the farthest vertex of Voronoi cell in the interior and exterior of the shape. They further find the connectivity between poles by using the power diagram. Poles are proven to approximate the medial axis effectively, and are efficient for shape reconstruction. The disadvantage of this method is that some Voronoi vertices close to the medial axis may be discarded when selecting the poles. The labelling of the poles also requires a robust way to differentiate between the inner and outer area of the shape, which may

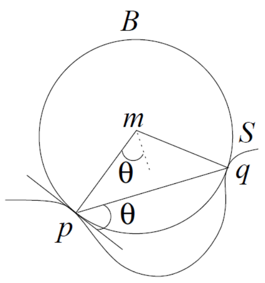


Figure 2.6: A medial axis point m, its separation angle θ, and the corresponding medial ball. Figure from [Dey and Zhao, 2004].

not be easy for some type of data. Another type of well known pruning criteria is based on *separation angle* [Dey and Zhao, 2004; Foskey et al., 2003; Michikawa and Suzuki, 2010; Yang et al., 2004]. As shown in Figure 2.6, this method considers the Voronoi vertex or medial axis point m of a shape S and its medial ball B. The points p and q are the closest sites on the shape which are the contact points between B and S. The separation angle usually defined as the angle between mp and mq, or half of this angle. This angle criterion is scaling independent and commonly used as a parameter to control the level of detail of the medial axis. The radius of the medial ball is also used as a pruning criterion. Chazal and Lieutier [2004] proposed M_λ the λ-medial axis, which is the set of points with medial balls with radius at least λ. They also discussed its stability under the Hausdorff distance. Choi and Lee [2000] analyse both the angle and radius criteria and prove that the medial axis under these two criteria is stable and within the prescribed error bound of the Hausdorff distance.

2.5.3 Applications

The medial axis has been widely applied for problems such as shape analysis, object recognition, reconstruction, path planning. Here we give some examples of applications in the field of computer graphics.

The medial axis can be used for shape recognition. The medial axis points can be classified to different types. The connection between the singular points can then form an condensed structure which can be represented by a graph. Then the shape recognition problem is boiled down to a skeleton matching problem. This method has been applied to both 2D [Shokoufandeh et al., 1999] and 3D [Chang and Kimia, 2009] shape recognition.

The medial axis has also been used for 3D mesh reconstruction. Amenta et al. [2001] compute the poles (the furthest medial axis points) and the power diagram for the poles. They then approximate the shape by the power diagram faces which are the boundary between the power diagram cells belonging to outer poles and the power diagram cells belonging to inner poles.

The process of computing the medial axis naturally provides the corresponding information between the medial axis elements and the shape elements. This correspondence can pass information from medial axis to the shape, for example, the geometry and the topology of medial axis. Hisada et al. [2002] detect salient features of a shape by finding edges on the skeleton. They also find salient shape regions that are extracted via segmenting the skeleton into patches.

The medial axis is also useful for mesh deformation. Yoshizawa et al. [2003] produce a skeleton from the Voronoi vertices within the input mesh. Instead of using poles, they find corresponding inner Voronoi vertices for each mesh vertex, and then compute the arithmetic mean of these Voronoi vertices and connect them according to the mesh edges. This means the connectivity of the skeleton is equivalent to the mesh. Their skeleton is not the exact medial axis, but it is more stable than using poles directly. When the skeleton mesh is deformed, they reconstruct the shape based on the correspondence deformation on the skeleton.

2.5.4 Summary and Discussion

Using the medial axis is a well developed field. It is viewed as a universal model for shapes, although its computational complexity and unstable nature is well known. The new spatial relationship we propose shares the following advantages of the medial axis:

- It is a compact representation, which is usually a skeleton-like mesh and can be further reduced to a graph structure. By using the medial axis, the shape comparison problem can be converted to a graph matching problem, for which many more classic methods can be applied.
- It is an abstraction of the space within the shape, and encodes the global spatial relation between shape vertices.
- Each medial axis element has corresponding sample pairs on the original shape. This correspondence can be quite useful for feature extraction or animation.

- It is straightforward to build a multiresolution medial axis. Many applications can benefit from this hierarchical nature of the medial axis.
- It is complete, which means the shape can be reconstructed from the medial axis if it is not pruned.

2.6 Summary

In the previous work, the processing of scene data only makes use of simple spatial relationship representations, which limit the previous methods to simple interactions. On the other hand, the medial axis, which is a well known tool for shape abstraction, has proven a rich descriptor to 3D shape.

By studying these topics we introduces in this chapter, we are inspired by the work which using medial axis to abstract the 3D shape, and propose to apply the medial axis to the spatial relationship representation.

In the following chapter, we give more details of how we use the medial-axis-based method to compute our new representation: the interaction bisector surface.

Chapter 3

Interaction Bisector Surface

In this chapter, we propose the Interaction Bisector Surface (IBS) as a new representation of spatial relationships.

We first introduce the definition (see Section 3.1) and computation (see Section 3.2) of IBS. The IBS is computed in a discrete way. It is an approximation of the exact generalized Voronoi diagram. We proposed the use of this method because it can by easily applied to any type of input 3D data. Next, we discuss the properties of IBS with respect to the following properties of 3D mesh: manifold/non-manifold, orientation, open/close, local connectivity and topology (see Section 3.3). We also introduce a method of sampling on the IBS in Section 3.4.2. By defining a weight for each IBS element, samples can be extracted by using a weighted sampling scheme. The resulting samples represent the most important area of IBS. Lastly, Section 3.4 explains methods for extracting features from IBS. The Betti numbers are used as topology feature, while the statistical-histograms-based features are used for describing the geometry of IBS.

3.1 Definition

Given N point sets $S_1, S_2, ...S_N$ in the 3D space where $S_i = \{p^i_1, p^i_2, ..., p^i_{n_i}\}$, an Interaction Bisector Surface (IBS) divides the space into N regions with following properties:

- Points from the same point set lie in exactly one region.
- If a point $q \notin \{S_1 \cup S_2 \cup ...S_N\}$ lies in the same region as S_i, then the Hausdorff distance(in the Euclidean space) between set $\{q\}$ and S_i will be shorter than the Hausdorff distance between set $\{q\}$ and S_j, where S_j is any other point set.

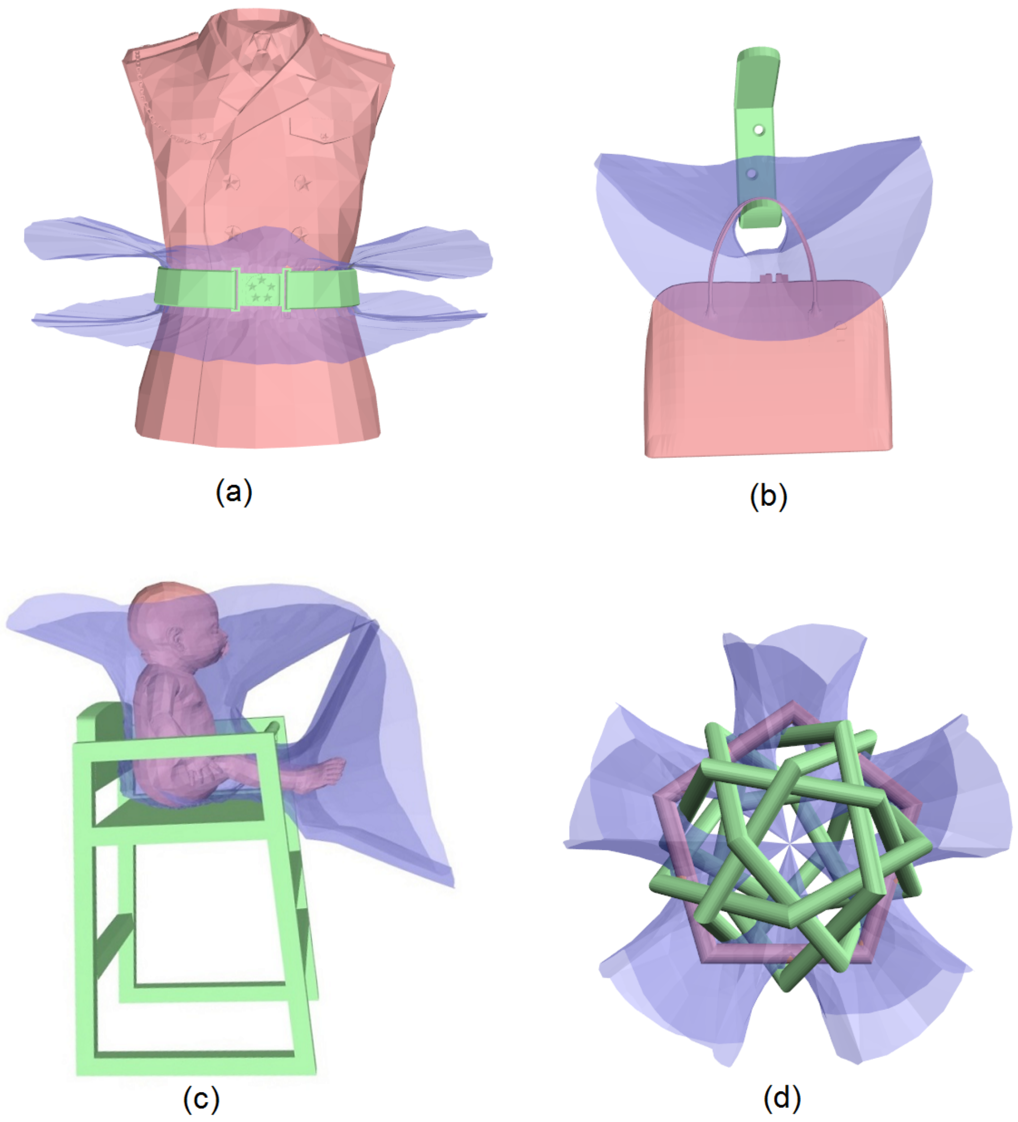


Figure 3.1: Examples of the Interaction Bisector Surface (the blue surface) for two parts of the 3D scene (shown as red and green). (a) belt on uniform, (b) bag on hook, (c) baby on chair, (d) a pentagon tangled with five other pentagons.

That is, the IBS is the set of points equidistant from two sets of points sampled on different objects.

Examples of the IBS for different scenes are shown as blue surfaces in Figure 3.1. Although the IBS can reach infinity (the same as the classical Voronoi diagram), we truncate it with a bounding sphere (details are given in Section 3.2). Despite the possibility that the IBS can produce a complicated polyhedral complex, it tends to form smooth shapes with stable topology when computed from objects in daily life such as those presented in the thesis.

3.2 Computation

Here we give details about how we compute the IBS for a given scene. We start by sampling points on the surfaces of the scene models uniformly, and then compute the Voronoi diagram for all these samples. The Quickhull algorithm [Barber et al., 1996] was used in this process. The result of the Quickhull algorithm is a simplicial complex consisting of polygons called *ridges*. Every ridge is equidistant to the two sample points which produce it. Hence there is a correspondence between ridges and the sample points. Assuming that the scene data is pre-segmented into objects, which is usually the case in scene data, we only select ridges that correspond to sample points from two different objects for computing the IBS. These steps are shown in Figure 3.2.

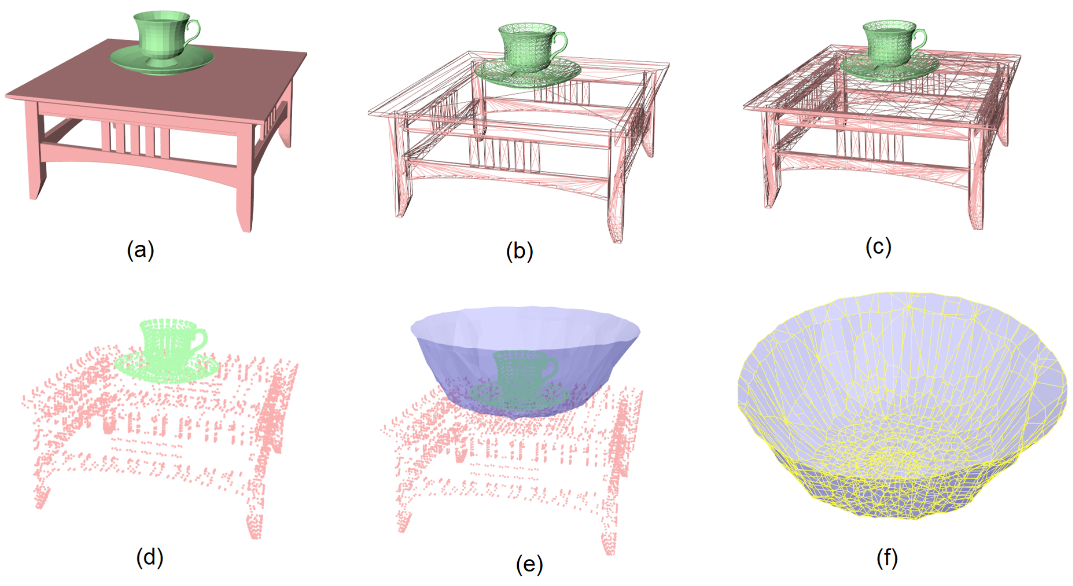


Figure 3.2: Steps for computing the IBS. Given a segmented scene (a) (in this example the scene has objects: a cup and a table), which is composed of polygon meshes (b), we first subdivide the mesh to triangles of similar size (c) and then take the centre points of each triangle (d). The IBS is computed (e). It is a polygon mesh (f).

As the IBS by definition could reach infinity, we trim it by adding a bounding sphere to the scene data before computing the Voronoi diagram. The bounding sphere is found in the following way. We first find the minimum axis aligned bounding box of the scene, and use the centre of bounding box as the centre of the bounding sphere. The diameter of the sphere is set to 1.5 times the diagonal of the bounding box.

Special attention is needed if two objects are very close to each other, as there is a chance that the IBS will penetrate the objects due to the inadequate sampling density.

In this case, we iteratively refine the IBS by the following process: if penetrations are found between the IBS and any of the objects, we sample more on the parts of the object where the penetrations happen, and recompute the whole IBS. We repeat this process until no penetration happens or the repetition time is larger than a threshold. Figure 3.3 shows the IBS between a table and a coffee cup. In this example, penetrations are eliminated after four iterations.

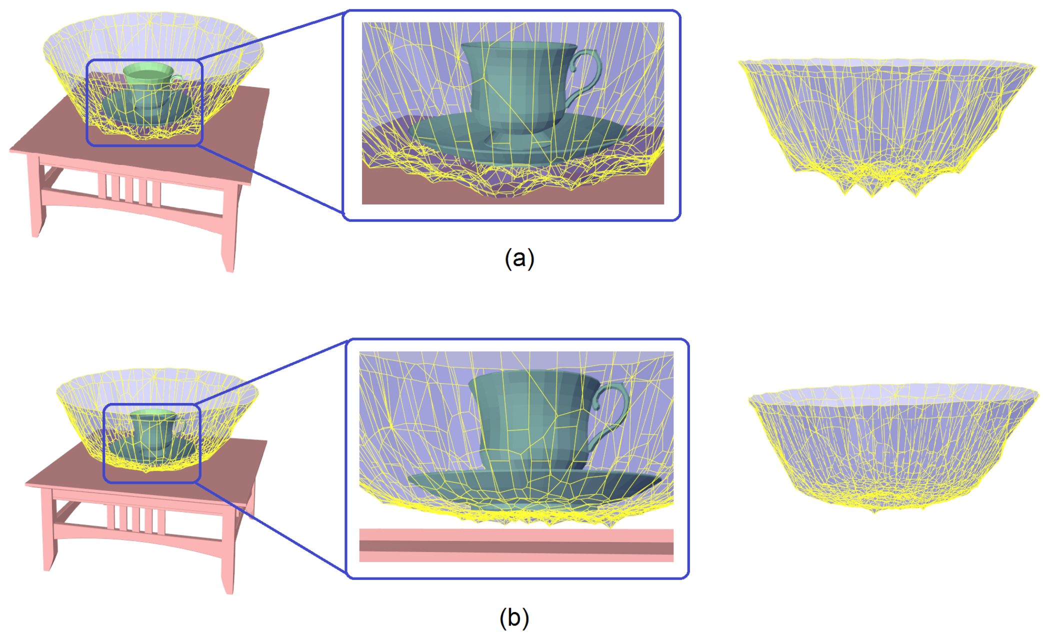


Figure 3.3: (a) Penetrations between the models and the IBS, which are caused by inadequate sampling on objects. (b) After 4 iterations of refining processes, there is no penetration, and the shape of the IBS becomes smoother. (The big gap between the cup and the table is for visualization.)

Although the topological structure of the medial axis can be sensitive to subtle geometric changes of the relevant surfaces, the IBS is rather robust against such changes as it is computed between two objects. The instability of the medial axis is due to the "fluctuating spikes" [Attali et al., 2009], which are produced by concave dips on the surfaces (the grey branches in Figure 3.4). As the IBS is only produced between separate objects, such spikes are not included in its structure and it is therefore less likely to be affected by subtle geometric changes of the object. More examples of the IBS of of 3D object pairs are shown in Figure 3.1 and Figure 3.10.

Given a scene, we only need to compute the IBS for the whole scene once, and it will contain the spatial relationship of every pair of adjacent objects. We denote the

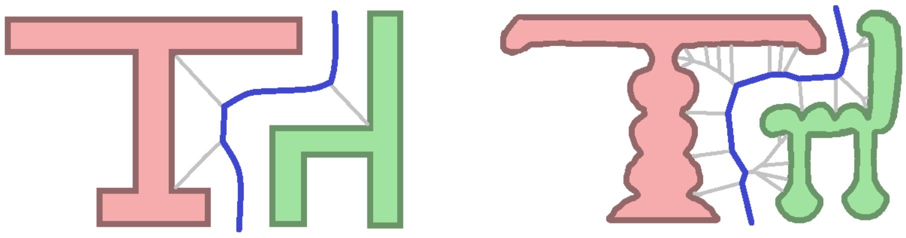


Figure 3.4: The IBS (the blue line) is the stable part of the medial axis (the blue line and the grey lines). It does not fluctuate significantly under subtle geometric changes.

subset of the IBS between object i and object j as $IBS(i, j)$. Furthermore, a subset of the IBS between two groups of objects, g_x and g_y, can be represented by $IBS(g_x, g_y) = \bigcup IBS(i, j)$ where $i \in g_x$ and $j \in g_y$.

3.3 Properties

In this section, we discuss the IBS from a different point of view: the properties of the IBS as a 3D mesh. From the definition of the IBS, we can see that the IBS is a division of the 3D space with certain properties. All the points on the IBS are in equal distance with at least two elements of the scene. As the IBS is a generalized Voronoi diagram, it also shares the properties of the classic Voronoi diagram. However in addition to the relationship between the object points and the IBS, the IBS itself, as a 3D mesh, has its own particular properties.

The IBS is a 3D mesh composed of polygons. From the examples in Figure 3.5 we can see that the polygons are not necessarily triangles and the size of the polygons has a large variance. The mesh density, which is the number of polygons in a unit area, depends on the following three factors: the density of samples on the objects, the shape of the objects and the distance between the sample pairs which correspond to the IBS. The vertex of one polygon will always be the vertex of other polygons if it lies on the edge of different polygons. In other words, the situation in Figure 3.6(a) will never happen. Additionally, the IBS polygons only share edges, which means the polygons will never intersect with each other as shown in Figure 3.6(b).

The IBS can be a single component or multiple components. The single component is common (most examples we show in this thesis are single component IBS). Multiple component IBS can happen for two possible reasons: first, if the input has more than

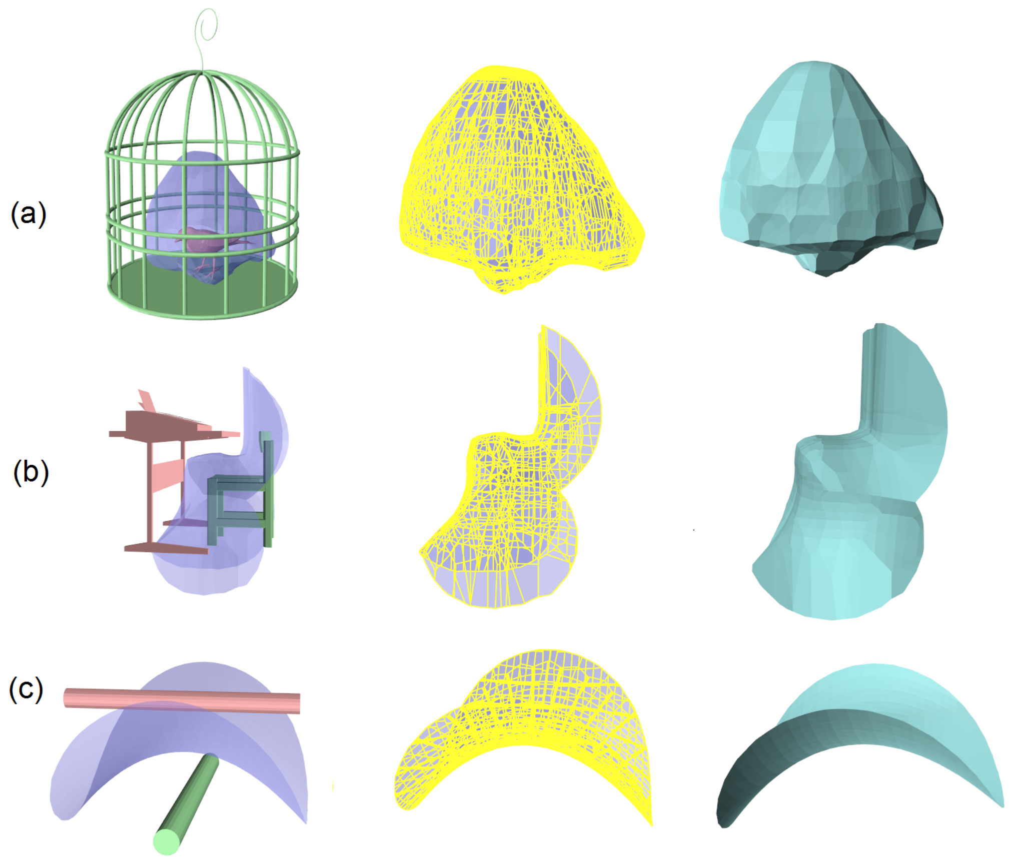


Figure 3.5: The mesh structure of IBS examples.

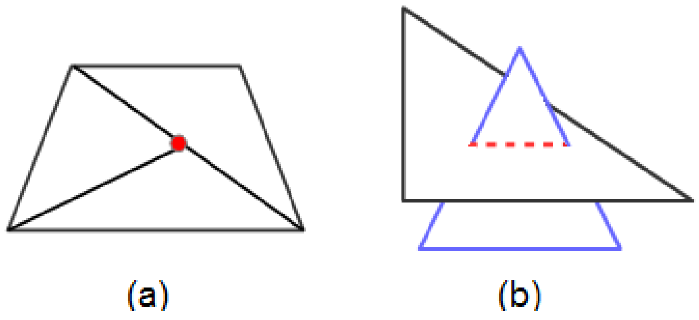


Figure 3.6: Two cases of non-manifold mesh elements.

two objects strictly located on a line, the IBS will contain two or more parallel surfaces which will never intersect (see Figure 3.7(a)); second, the trimming process may cause separations. Figure 3.7 shows an example whose IBS components connect at a location which is far away from the scene centre (Figure 3.7(b)), but they may be separated with a different trimming parameter (Figure 3.7(c)).

The IBS can be a manifold or non-manifold mesh. A manifold IBS is normally

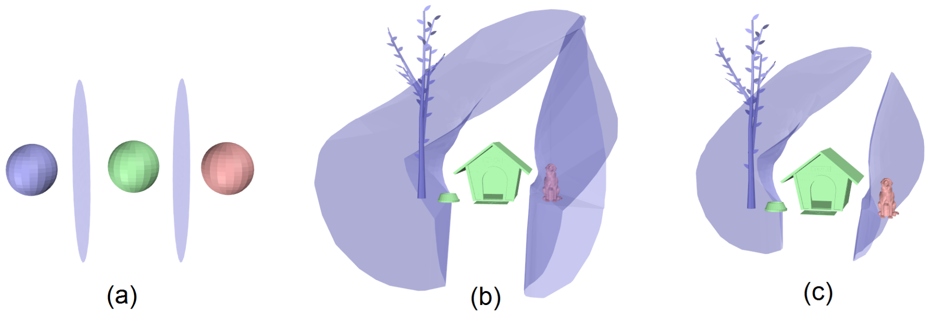


Figure 3.7: IBS examples which contain multiple components.

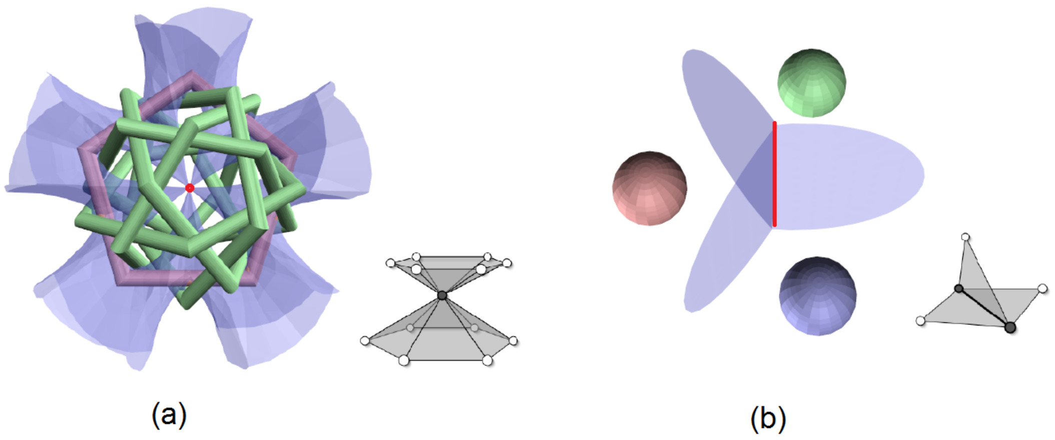


Figure 3.8: Two examples of non-manifold IBS. The red point in (a) is a non-manifold vertex, and the red edge in (b) a non-manifold edge.

produced by objects with spatial relationships, such as “beside” or “enclose” shown in Figure 3.5. Complex relationships can produce non-manifold meshes. For example, in Figure 3.8(a), the different faces of IBS are incident to a vertex in the middle, and Figure 3.8(b) shows how a set of IBS vertices which are shared by three surfaces form a non-manifold edge.

The boundary of a 3D mesh is a set of edges only incident to one face. The untrimmed IBS will never have such type of edge, so it has no boundary. However, we always trim the IBS, so the IBS we use in this thesis, can be meshes either with (Figure 3.5(b, c)) or without (Figure 3.5(a)) a boundary.

The IBS is always orientable. The Mobius strip or Klein bottle type of shape never occurs with the IBS. Because if the IBS were not orientable, an ant walking on the IBS mesh could start from one side and end with the other side. This conflicts with the definition of IBS, which has the restriction that the IBS must always separate the space and each object must have its own space. So the IBS will never have “holes” in the mesh which connect two different object regions together.

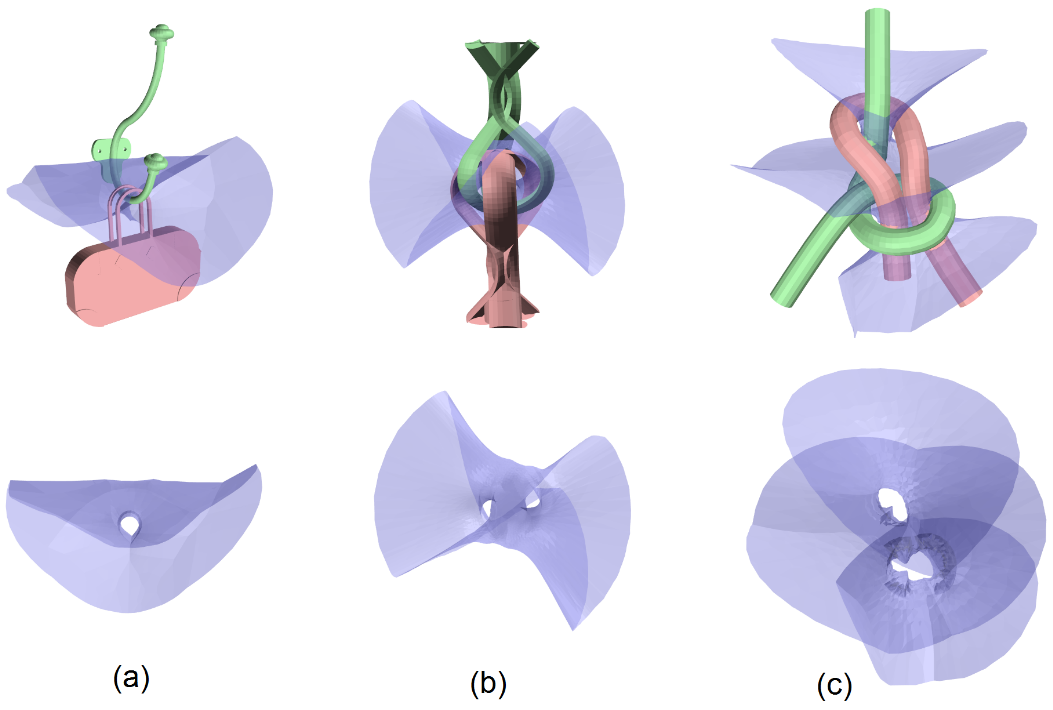


Figure 3.9: Examples of the IBS which contain handles and tunnels.

Although possessing no boundaries (except boundaries produced by trimming) or holes, IBS meshes may still contain interesting global topological features such as handles or tunnel (see examples in Figure 3.9). Handles or tunnels normally happen when two objects are tangled with each other, or when one part is surrounded by another part locally. If handles and tunnel in the IBS can be detected and located automatically, it will be quite helpful for analysing complex interactions. Currently, we find features which can compute the number of the handles and tunnels, but locating the handles and tunnels on 3D mesh is still a open problem. Dey et al. [2013] detect the locations of handles and tunnels on a 3D mesh by using a Reeb graph. But as their method is limited to meshes without boundaries, it cannot be directly applied to the IBS.

In summary, the IBS is a orientable 3D polygon mesh. It can be a single component mesh or a mesh with multiple components. It can be either a manifold or non-manifold mesh. The untrimmed IBS has no boundary, but if the IBS is originally a open mesh, it will have a boundary after the trimming process. In Section 3.4, we will explain how we use Betti numbers to capture the topological properties of the IBS including the number of components, whether it is an open or closed surface, and the number of handles and tunnels.

3.4 Features

In this section, we give details about how to compute the topological and geometric features of the IBS.

3.4.1 Topological Features of the IBS

Topological descriptions of relationships are succinct and robust against small geometric variations. Consider a ball in a box. The description “in” here is unaffected by the ball position or orientation as long as it is inside the box. Thus, capturing the topological nature of the interaction between two objects is crucial in understanding relationships. A good indicator of the topological nature is the Betti numbers of the IBS. We will first briefly give the definition of Betti numbers and then demonstrate how it can be applied as a feature to classify complex interactions.

The Betti number is a concept in algebraic topology. Formally, the k-th Betti number refers to the number of independent k-dimensional surfaces [Carlsson, 2009]. We make use of the second (denoted as b_1) and third (denoted as b_2) Betti numbers in this research. They represent the number of two-dimensional or “circular” holes (b_1), and the number of three-dimensional closed regions (b_2). Intuitively speaking, b_1 represents the number of “cuts” needed to transform a shape into a flat sheet. For example, objects that are laterally surrounded by others, such as a house surrounded by fences (see Figure 3.10 (c)), forms an IBS of a cylindrical shape, resulting in $b_1 = 1$. For objects tangled with other objects, such as toilet roll on a holder (see Figure 3.10 (d)), a partial torus is generated, resulting in $b_1 = 2$. b_1 can be even larger under complex interactions whose IBS involves a lot of loops (see Figure 3.10 (e)). b_2 represents the number of closed surfaces. In our scenario, it counts how many objects are wrapped by other objects (see Figure 3.10 (b)). We compute Betti numbers from the mesh data by the incremental algorithm [Delfinado and Edelsbrunner, 1995].

3.4.2 Sampling the IBS

To capture the most descriptive area of the IBS, we propose a sampling method. By using this method, we can sample points on the most important area of the IBS, and further use these points for computing features for the IBS.

Figure 3.11 shows that the areas which are closer to the models and the centre of the scene (highlighted by red circles) are more descriptive than the areas which are a

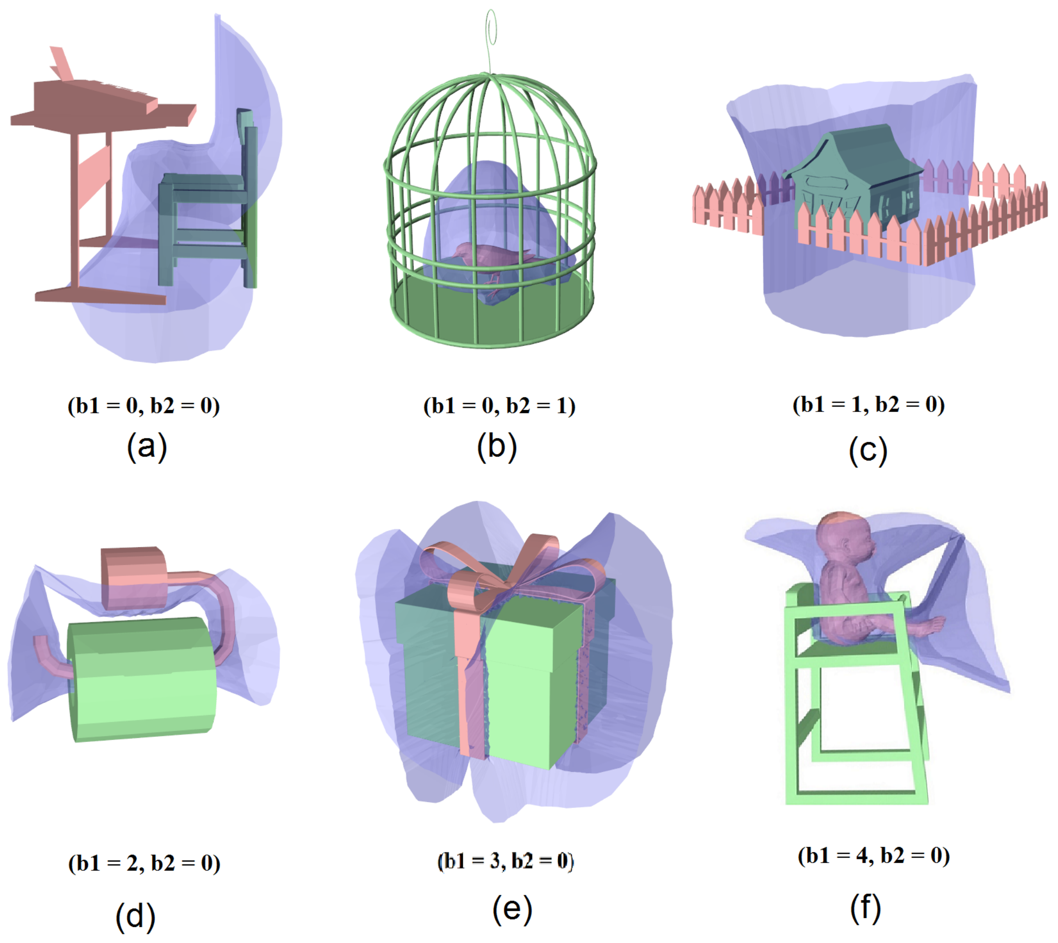


Figure 3.10: The IBS (in blue) of two object scenes (a) a table and a chair, (b) a bird in a cage, (c) a house surrounded by fences, (d) toilet roll on a holder, (e) a gift box and a ribbon and their Betti numbers.

in greater distance from the models (highlighted by blue circles) with respect to the spatial relationships. The relationships between parts in both scene(a) and scene(b) are “one part hooking onto the other”. But if we use the global shape of the IBS’ to compare these two scenes, they might be recognized as very different, because the area in blue circles of the two IBS’ are quite different. The sampling results are shown in the right column of Figure 3.11. We can see that by using this sampling method, we can find the areas within the red circles, which enable use to capture the similarity between these two interactions.

Here we describe how we sample points on the IBS. This is done by calculating the weights of the triangles composing the IBS. First, we triangulate the IBS and define a direction angle α for each triangle. A triangle T on the IBS is equidistant to sample

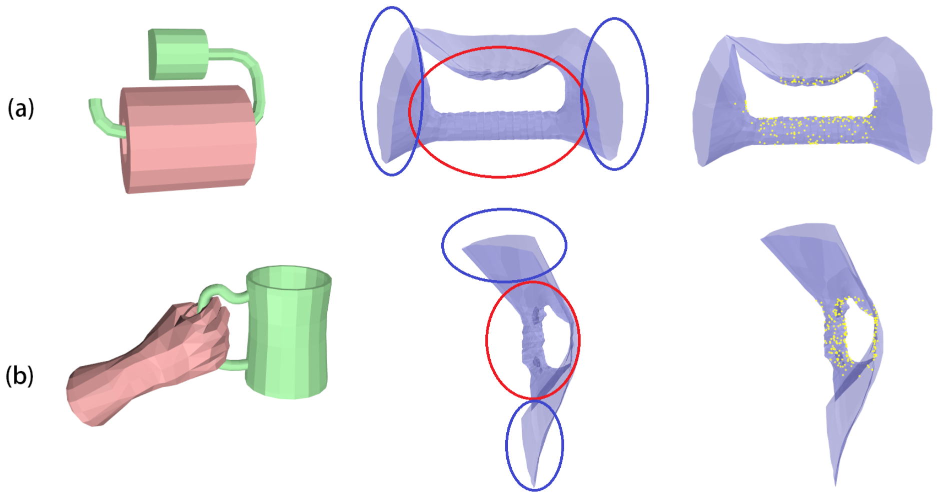


Figure 3.11: Left: The examples (a) and (b) are similar interactions, which contain one object hook onto another. Middle: Although the global shape of their IBS is different, they share very similar central area which is highlighted by the red circles. Right: The samples we computed are located in the circled area of the IBS and capture the similarity of these two interactions.

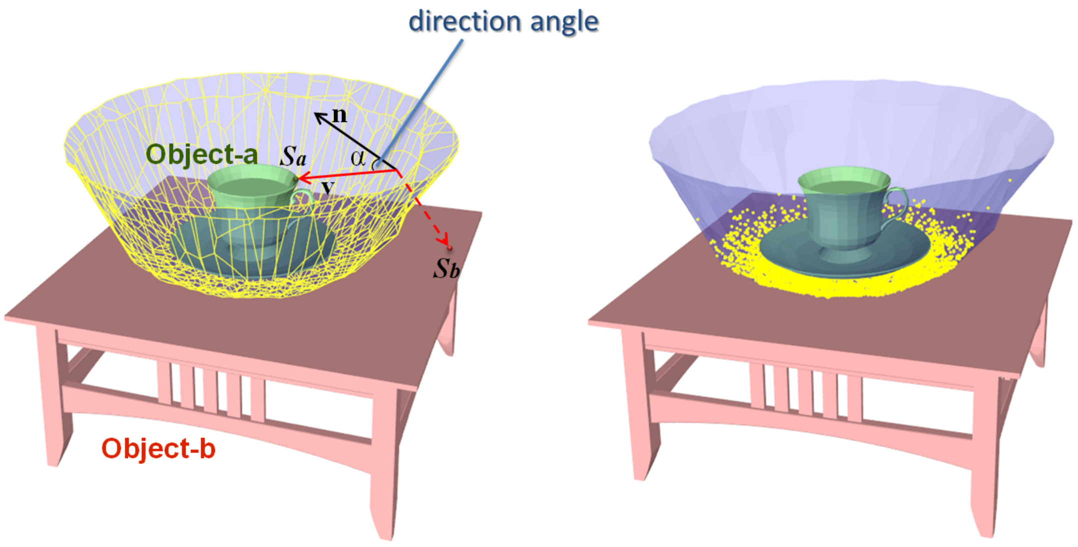


Figure 3.12: (a) The direction angle of a polygon on the IBS, (b) The sampling result.

points, s_a and s_b on object-a and object-b respectively. Let us define a vector **v**, which from the centre of T to s_a, and the normal vector **n** of T pointing towards the side of object-a. α is the angle between **v** and **n** (see Figure 3.12(a)) . Note that this angle is the same if we compute it between T and object-b because the normal is flipped in that case. The larger the angle is, the higher the chance that the sample point is far away from the objects defining it and less informative about the interaction. We compute a weight $W(T)$:

$$W(T) = W_{\textbf{area}}(T) * W_{\textbf{scene-distance}}(T) * W_{\textbf{angle}}(T) \tag{3.1}$$

where $W_{\textbf{area}}(T)$ is the area of triangle T and $W_{\textbf{angle}}$ is computed as:

$$W_{\textbf{angle}} = \begin{cases} 1 - \frac{\alpha}{45^\circ} & \text{if } \alpha < 45^\circ \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

$W_{\textbf{scene-distance}}$ is computed by:

$$W_{\textbf{scene-distance}} = (1 - \frac{d}{D})^n \tag{3.3}$$

d is the distance between the center of T and s_a (or s_b). $D = d_{\textbf{diag}}/2$ where $d_{\textbf{diag}}$ is the length of the diagonal of the bounding box of the whole scene. We empirically set n equal to 20.

We normalized W(T) for all the triangles before final sampling. Given the target number of samples N, the number of samples on each triangle computed by $n_i = N \times W(T_i)$. Finally, we randomly sample n_i points on triangle i (for all triangles). Figure 3.12(b) shows the result of the weighted sampling.

3.4.3 Geometric Features of the IBS

Although the Betti numbers can distinguish the qualitative difference of interactions, they cannot distinguish subtle differences. For example, the IBS of two boxes laterally adjacent to each other has exactly the same Betti numbers as that of an apple in a bowl. To address this problem, we evaluate the following geometric attributes of the IBS:

1. geometric shape,
2. distribution of the direction vectors
3. distribution of distance between the IBS and the objects.

These features are computed at points sampled on the IBS. As different parts of the IBS are not equally descriptive of the relationship, we use the importance-based sampling scheme that is described in Section 3.4.2. In brief, more points are sampled where the IBS is in close proximity with the objects defining it.

Geometric Shape of the IBS The geometric shape of the IBS is useful for comparing the nature of the interactions. For example, when one object is simply parallel to another, the IBS will become planar, but it will form a bowl shape when one object is surrounded by another object.

Various shape descriptors can be considered for the IBS. One possibility is to use the curvature profile; however, the curvature data can be unstable as the IBS may include ridges with sharp turns. This occurs when the mapping of the closest point between the IBS and the object becomes discontinuous due to the concavity of the object. Also, the IBS may be either an open or closed surface.

Taking into account these characteristics, we use the Point Feature Histogram (PFH) descriptor [Rusu et al., 2008b]; The PFH is a histogram of the relative rotation between each pair of normals in the whole point cloud. It describes the local geometrical properties by generalizing the mean curvature at every point. It provides an overall pose and density invariant feature which is robust to noise. The PFH was applied for 3D point cloud classification [Rusu et al., 2008b] and registration [Rusu et al., 2008a].

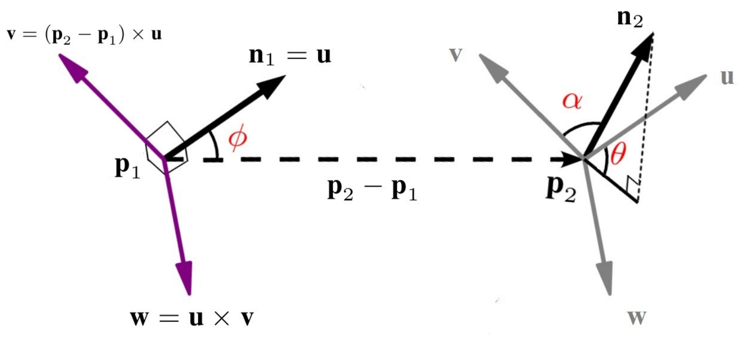

$\mathbf{v} = (\mathbf{p}_2 - \mathbf{p}_1) \times \mathbf{u}$
$\mathbf{n}_1 = \mathbf{u}$
ϕ
$\mathbf{p}_1$
$\mathbf{p}_2 - \mathbf{p}_1$
$\mathbf{w} = \mathbf{u} \times \mathbf{v}$
$\mathbf{n}_2$
v
u
α
θ
$\mathbf{p}_2$
w

Figure 3.13: PFH angles.

The PFH is a feature for encoding the geometry of a point cloud. Given two points (Figure 3.13), p_1 and p_2, with normals n_1 and n_2, three unit vectors (u, v and w) are built by the following procedure: 1) u is the normal vector of p_1, 2) $v = u \times \frac{p_2 - p_1}{d}$, 3) $w = u \times v$. $d = \|p_2 - p_1\|_2$. Then the difference between n_1 and n_2 are represented by three angles (α, θ, ϕ) which are computed as: $\alpha = v \cdot n_2$, $\phi = u \cdot \frac{p_2 - p_1}{d}$, $\theta = arctan(w \cdot n_2, u \cdot n_2)$. The triplet $< \alpha, \phi, \theta >$ is computed for each pair of points in the k-neighbourhood, and are binned into a histogram. Usually each angle is divided into b equal parts, and the triplet can form a b^3 size histogram in which each bin represents a unit combination of the value ranges for each value.

In our case, we compute the triplet for each pair of points in the point cloud which is a set of samples computed by using the method described previously. We set $b = 5$.

So the PFH feature we use is a 125-length vector. For each sample point on a given IBS, we compute the 125 dimension histogram of the relative rotation angles between the normal vector at the sample point and those of the other sample points. So we produce a set of histograms for the whole IBS. Then we follow a method proposed in [Alexandre, 2012]. We compute the centroid and the standard deviation for each dimension of the histogram set, and use the resulting 250 dimension vector as the final feature of the IBS.

Direction The normal vectors of sample points on an IBS contain the direction information about the spatial relationship. For example, if all the normal vectors of the IBS are facing upwards, one of the objects forming the IBS is above the other.

The direction of the IBS normal is defined so that it points toward the reference object. In our definition, spatial relations are unidirectional. The relationship of A with respect to B is different from B with respect to A. We first need to specify the reference object. As the IBS normal can be pointing either side of the IBS surface, we use the one that is defined on the side the reference object exists.

The direction feature of the IBS is computed as follows. As the relationship such as "above" and "below" should be defined with respect to the reference object, we assume that the $+z$ axis of the scene is the vertical direction, and use the angle between the normal vector and $+z$ direction (denoted here by θ) to compute the direction feature. We compute θ for each sample on an IBS, and produce a histogram with 10 bins in the range of 0 to π. The number of samples which fall into each bin is counted, and normalized against the total number of samples.

Distance Between the Object Surface and IBS The distribution of the distance between the IBS and the object surface is descriptive about the relations of the two objects. The larger the distance is, the less likely that the two objects are closely related. We produce a histogram with 10 bins whose range is between 0 to $0.5 \times d$, where d is the diagonal distance of the bounding box of the two objects. We compute the distance for each sample on the IBS, and accumulate the number of sample points that fall into each bin. The histogram is normalized by the total number of samples and is used as another geometric feature.

3.5 Summary and Discussion

In this chapter, we have proposed a new descriptor called the Interaction Bisector Surface to capture spatial relationships between 3D objects. The IBS is the cornerstone of our research and it provides a new perspective to model spatial relationships.

For the computation of the IBS, we use a sampling-based approach in which points are sampled on the object surfaces and then the Quickhull algorithm is used. This is a heuristic approach that does not guarantee the exact topology and geometry of the resulting medial axis. This could be an issue if we need to match the homotopy of the IBS. In order to avoid such confusion, we use abstract topology features (the second and third Betti numbers) whose values are less influenced by the accuracy of the IBS. Although the Betti numbers can be affected by topological noise such as holes, these are unlikely to appear in the bisector surfaces as they are defined between distinct separate objects. Also, the geometric features of the IBS are statistical values that are less influenced by the accuracy of the IBS. In addition, exact methods to compute the medial axis [Culver et al., 1999] and bisector surface [Elber and Kim, 1997] cannot be practically applied to a set of high resolution meshes. In summary, our method makes use of features that are less computationally costly and less affected by parameter values.

The topological and geometrical features we proposed in this chapter will be used for scene analysis (see Chapter 4). More specifically, we use these features to compare objects in the scenes based on the spatial relationship between the object and its environment.

The IBS also has an important property that it is located in the *open space* with certain conditions: it is the bone of the open space with radius information and it corresponds to the shape elements in the scene. We will later describe a new feature for the interaction between the character and the object based on the correspondence between the IBS to the scene elements (see Chapter 5). And we will also explore the features on the IBS from open space point of view. These features are essential for the synthesis of new scenes (see Chapter 6).

Chapter 4

Relationship-Based Static Scene Analysis

In this chapter, we present three experiments. The first experiment is the supervised classification of pairwise interactions (see Section 4.1). We collected and labelled a database of scenes which contains two objects, and make use of the topological and geometric features proposed in Section 3.4 for the classification. The second experiment is about building structures for scenes with multiple objects (see Section 4.2). The core of the method is defining a closeness measure between objects by using the correspondence between IBS and scene elements, and the weight of IBS defined in Section 3.4.2. In the third experiment, we propose a method for retrieval of scene objects based on the spatial relationships (see Section 4.3).

4.1 Relationship-Based Classification

In this section, we show how geometrical and topological features in different combinations help in classifying pair-object relationships.

We first explain the method by which a pair-object database was built, then the set-up of the supervised classification, and finally we present the experiment result and evaluation.

4.1.1 Pair-Object Database

The data set we use contains 1381 items, each of which is an object pair. We ask the user to label the pairs based on their spatial relations. The database consists of

Examples	Description	Examples	Description
1, 2	Enclose	3, 4	Encircle
5	Interlocked	6	Side by side, similar sizes
7, 8	Tucked in	9, 10	Side by side, one considerably higher
11, 12	Loosely above	13, 14	On top of
15, 16	Partially inside, with open areas		

Table 4.1: Different Types of Interactions

16 classes. We show one example in each class in Figure 4.1. Descriptions for these classes are summarized in Table 4.1. Note that there are some scenes from different classes with identical geometry but different object order, as the spatial relation between two objects are not symmetric. For example, there are two types of relation “enclose”, one object is enclosed by another and one object encloses the other. this is the same with other relation types except 5 and 6 in Figure 4.1.

In order to facilitate the description, we refer to the interactions with numbers $b_1 = 0, b_2 = 0$ as *simple relations*, and all others as *complex relations*. The first part of our database contains 1289 items that are extracted from the Stanford Scene Database used in [Fisher et al., 2012]. Since this mainly consists of simple relations, we denote them as S. We manually labelled the data into meaningful classes, which turn out to be 11 classes (class 6 to class 16 in Figure 4.1). The second part of the database contains 92 examples of complex relations labelled into 5 classes (class 1 to 5 in Figure 4.1) by the user. As all data in this part represents complex relations, we refer to it as C. More examples from these 16 classes are shown in Appendix A.

4.1.2 Classification Experiment

We do the experiments first on S and C individually and then on the whole database $S+C$. In each experiment, the data is split into a training set and a testing set in the ratio of 7:3. We performed classification on different combinations of features to investigate how individual features and combinations of them influence the classification. Specifically, we test PFH (P), PFH + Direction (PDI), PFH + Direction+Distance (PDD) and PFH + Direction + Distance + Betti number (PDDB). The feature vector in each experiment is a concatenation of the involved individual features. Each type of features is first normalized into the range of [0,1]. In different experiments, we use different com-

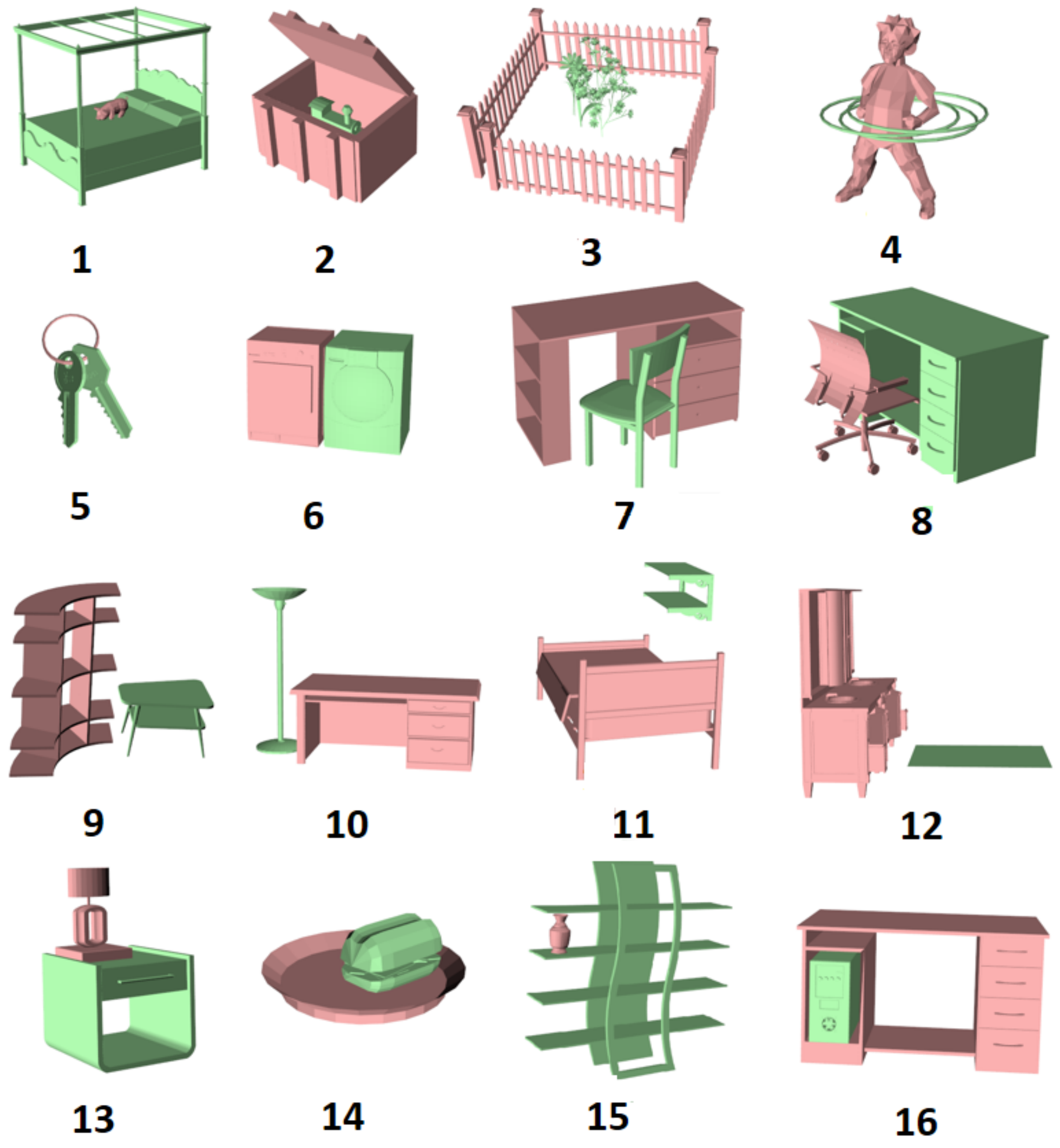


Figure 4.1: Examples from 16 classes in our database. One example for each class. The 3D models shown above are from the Princeton Shape Benchmark [Shilane et al., 2004] (1-5) and the Stanford Scene Database [Fisher and Hanrahan, 2010] (6-16).

binations of the normalized features. In other words, if we use a fixed-length feature vector to represent each feature with non-zero values on its corresponding dimensions and all other value zeroed, then the concatenation can also be seen as linearly summing up several features. We tried different weights for this linear combination to achieve good results. Empirically an equal weighting scheme is used for all the classification experiments. For comparison, we also tested two features: absolute height displacement and absolute radial separation used in [Fisher and Hanrahan, 2010] for the whole data set, denoted by DIS.

We chose Support Vector Machines (SVMs) [Boser et al., 1992; Cortes and Vapnik, 1995] for the classification task because of their simplicity. Specifically, we use the soft margin method [Cortes and Vapnik, 1995]. For our multi-class problem, a *one-v.s.-one* scheme is used. For the kernel, we use a Radial Basis Function (RBF), $K(x,y) = e^{-\theta\|x-y\|^2}$ where x and y are the concatenated feature vectors. To find the best parameter values, we do 5-fold cross-validation and hierarchical grid search. We start with coarser grids, then subdivide the best grid for another iteration of search until the improvement of the accuracy falls below 0.001 or we reach a maximum iteration. Finally, we train the model with the whole training set again using the optimal values and then test it. For implementation, we use libSVM [Chang and Lin, 2011].

4.1.3 Evaluation and Comparison

The prediction accuracy is shown in Table 4.2. The first column consists of the data set. The first row lists features. The cells are filled with prediction accuracies of each experiment. They are calculated by feeding the testing data set into the trained SVM classifier and the accuracy is the percentage of correctly classified data.

Overall, IBS features are more discriminative for relationship classification than DIS used in [Fisher and Hanrahan, 2010]. Note that in the paper [Fisher and Hanrahan, 2010] , they achieve good retrieval results by using other information such as labelling, but we aim to avoid using this as such data is not always available.

For further comparison of features within PDDB, one can see that the PFH descriptor of the IBS already gives good results for this 16-class classification problem. On top of PFH, the direction improves the result. On the complex data set, the distance is a bit detrimental to the result. This is because distance and direction can contradict one another in this data set. However, we find that the prediction accuracy of the complex data set is also 100% when just using PFH and Betti numbers. It reflects the fact that for complex relationships, direction and distances are not discriminative enough. It also shows that Betti numbers provide vital information, especially in classifying complex interactions.

One noteworthy point is that there is a slight decrement of accuracy from PDD to PDDB on the S cases. One cause is the discrete nature of Betti numbers. Relations with similar geometric features may have different Betti numbers. Also, penetrations between objects can cause the Betti numbers to be calculated incorrectly. Although most scenes in the Stanford Database do not have penetrations, some still exist for

	P	PDI	PDD	PDDB	DIS
S	78.33%	84.07%	85.90%	82.77%	41.25%
C	80.00%	96.00%	92.00%	100.00%	44.00%
S+C	76.47%	81.37%	83.33%	84.31%	38.73%

Table 4.2: Prediction Accuracy.

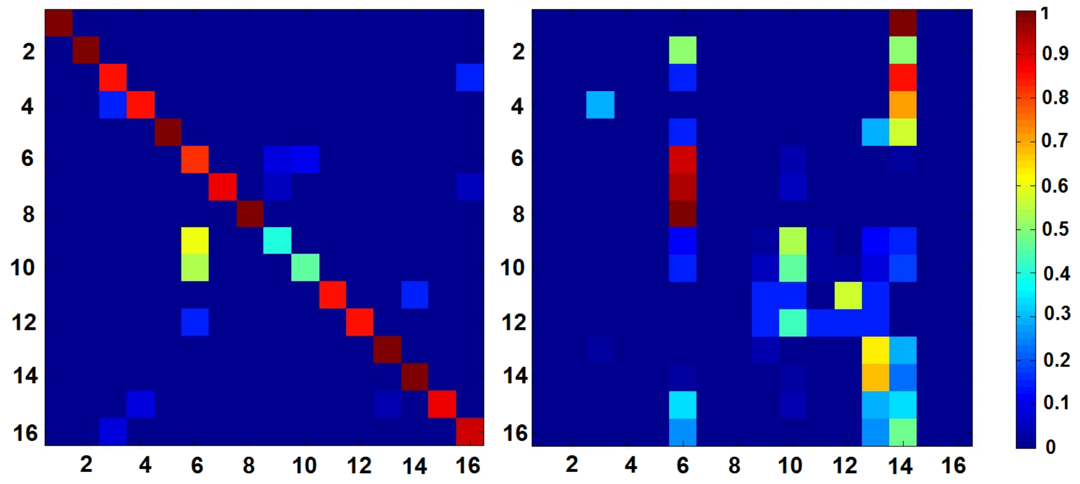


Figure 4.2: Confusion matrices of PDDB (Left) and DIS (Right) on the whole data set (16 classes). Results are normalized within each column.

geometrically adjacent objects. A preprocessing stage to exclude such penetrations would improve the results.

In Figure 4.2 we plotted the confusion matrix. The values are normalized for each column. The classes that have the lowest prediction accuracies are class 9 and 10 (Figure 4.2 left). Most of their misclassified instances are in class 6. Shown in Figure 4.1 and Table 4.1, classes 6, 9 and 10 all have a "side-by-side" relationship. But classes 9 and 10 have one object higher than the other. The height difference in some scenes is not big enough to be classified correctly. This is the main source of the prediction error.

Performance Table 4.3 shows the timing and accuracy information of cross-validation and training on the whole data set ($S+C$). The first row contains the best average accuracy of the 5-fold cross validation during the hierarchical grid search. The second row contains the time consumption for the grid search and the last row contains the training time after we find the optimal values of the parameters.

The configuration of the computer where these numbers are calculated is: Intel i7-

	PDDB	DIS
Cross-validation accuracy	83.56%	39.36%
Time for cross-validation (secs)	580.56	106.07
Time for training (secs)	0.22	0.06

Table 4.3: Performance

2760QM CPU, 8GB memory, Windows 7 Professional 64 bit and Matlab R2012a (64 bit).

4.2 Structure Analysis of 3D Scenes

In this section, we propose a method to automatically build a hierarchy out of a scene by making use of the IBS data. The method is an adapted version of the Hierarchical Agglomerative Clustering (HAC) algorithm, see e.g. [Hastie et al., 2009]. The resulting scene structure is used later for content-based relationship retrieval.

We first give the motivation, then functions to compare inter-object and inter-group relations and an algorithm for constructing a hierarchy based on spatial relations.

4.2.1 Motivation

The idea of representing scenes through graph structures has been applied in content-based scene retrieval [Fisher and Hanrahan, 2010; Fisher et al., 2011] and synthesis [Fisher et al., 2012; Yu et al., 2011]. In previous works, the relationships between objects in a 3D scene are either embedded at the design stage or computed based on contact [Fisher and Hanrahan, 2010; Fisher et al., 2011; Yu et al., 2011]. Examples of such scene graphs are shown in Figure 4.3 (b)(d).

The major difference between our method and previous works is that we adopt a multiresolution structure that encodes not only the spatial relationships between individual objects but also those between the object groups, these are more descriptive of the scene, especially when the number of scene components is large.

Let us describe the advantage of considering the inter-group relationship with an example. For the sake of simplicity, we shall call an object group a *community*. A community only containing an object and its immediately surrounding objects is called the *local community* of the object. A larger community containing other objects further

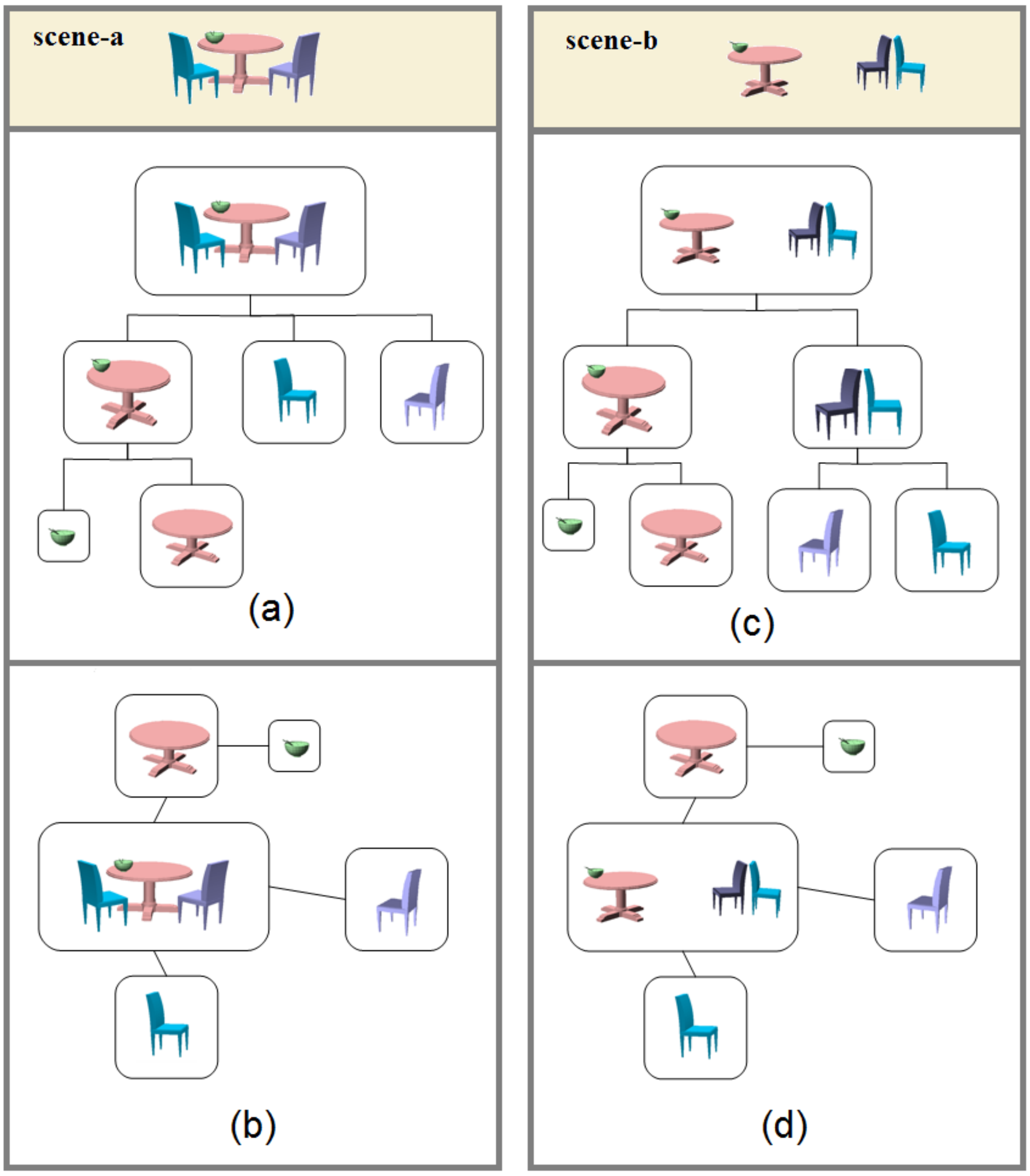


Figure 4.3: Scene structures of two example scenes: scene-a and scene-b. (a) and (c) show the hierarchical structures produced by our method; (b) and (d) show the scene graph used in [Fisher et al., 2011].

away in the scene is called the *extended community* of the object. In scene-b of Figure 4.3, the status of the bowl can be described through its local community (the table and the bowl) first, and then further described by the relation between the the bowl's local community and other communities (the two chairs) in the room. This description is far easier to recognize than using the raw, low level relationships of all the individual objects, such as is done by Fisher and Hanrahan [2010]; Fisher et al. [2011] as shown in scene-d in Figure 4.3. The reason behind this is that humans tend to recognize a scene at the group level when observing it from a global perspective [Goldstein, 2009], by aggregating objects based on proximity, continuation, uniformity, etc. Our multiresolution representation is also more descriptive than the raw graph description by Fisher et al. [2011]. This can be seen through scene-a and scene-b in Figure 4.3; the two are the same under the raw graph representation (Figure 4.3 (b) and (d)) while the objects are grouped based on the spatial relationships and distinguished in our multiresolution representation (Figure 4.3 (a) and (c)).

The terms "local" and "extended" community are only used for description purpose, and we do not arbitrarily classify neighbours into such categories. The inter-community relationships are produced by first grouping individual objects into communities of closer objects and then recursively grouping them into larger communities. The details of this procedure are described in the following subsection. This structure naturally forms different abstraction levels of the scene. Given a reference object, the inter-community relationships on each level reflect the relationships between the reference object and the scene at different abstraction levels.

4.2.2 Closeness Measure and Hierarchy Construction

To formally define the hierarchy and the relationships between one object and its environment, we define a measure called *closeness* between communities that can contain only an object or a set of objects. Given a scene $\mathcal{S}$ with n communities, $G = \{g_0, g_1, ..., g_{n-1}\}$, the closeness measure between any two communities, g_x and g_y, is defined as below:

$$R_c(g_x, g_y) = R_{\textbf{ratio}}(g_x, g_y) + R_{\textbf{ratio}}(g_y, g_x) \tag{4.1}$$

$$\begin{aligned} R_{\textbf{ratio}}(g_x, g_y) &= \frac{W\big(IBS(g_x, g_y)\big)}{W\big(IBS(g_x, G \setminus g_x)\big)} \\ IBS(g_x, g_y) &= \bigcup_{i \in g_x, j \in g_y} IBS(i, j) \end{aligned} \tag{4.2}$$

where $IBS(i, j)$ represents the IBS subset shared by object i and j. The function W computes the weighting of the IBS region. Note that simply computing the area of $IBS(i, j)$ does not give a good measure of the importance as mentioned in Section 3.4.2. In practice, we use $W\big(IBS(i, j)\big) = n$ where n is the number of sample points (that is described in Section 3.4.3) on the IBS shared between object i and j instead of computing its actual area. This is to give more weight where the two communities are closely interacting with each other.

$R_{\textbf{ratio}}(g_x, g_y)$ is the *commitment* of g_x towards g_y; $R_{\textbf{ratio}}(g_x, g_y)$ is larger if g_x shares a larger amount of the IBS with g_y than with other communities. It also means g_x commits more to g_y than to any other communities. Note that $R_{\textbf{ratio}}(g_x, g_y)$ is not necessarily symmetric. Essentially, R_c measures the relationship between two communities under the context of the whole scene.

With R_c as a distance function, we present an adopted HAC algorithm to build a hierarchical structure of a scene. This hierarchy is built iteratively in a bottom-up fashion. Starting from individual objects (leaf nodes of the tree), we measure the R_c between nodes and group them into nodes that represent bigger communities. A merge can combine more than two nodes. This process is repeated until the whole scene is merged into one big single node. The details of the approach can be found in Algorithm 1. Figure 4.3 (a) and (c) show simple examples.

4.2.3 Experiment and Evaluation

We built hierarchical structures for 130 scenes in the Stanford Database by using Algorithm 1. One example of the results is shown in Figure 4.4. More results are shown in the Appendix B.

The parameter τ controls the speed of merging when building the hierarchy. A lower τ will merge more nodes together in each round, which means that the number of levels will be smaller compared to the structure built with a higher τ. The choice of τ should depend on the nature of the data as well as the higher-level application. For the Stanford Scene Database, we found $\tau = 0.32$ gives visually reasonable structures for most of the scenes. The average number of levels of the hierarchical structure under this τ setting is 2.89.

Evaluation and Analysis As the scene structure will be used as the input for content-based retrieval, the stability of the hierarchical structure with respect to the parameter setting is important. We evaluate the stability of our HAC algorithm in this experiment, and its benefits for retrieval will be evaluated together with the retrieval results in the next section.

Following the scheme by Goodman and Kruskal [1954], which has been employed to compute the stability of hierarchical algorithms for image segmentation [MacDonald et al., 2006] and speech classification [Smith and Dubes, 1980], we assess the stability

```
Data: A scene $S = \{\text{object}_1, \text{object}_2, ..., \text{object}_m\}$,
      grouping threshold $\tau$, $0 \leq \tau \leq 1$
Result: A hierarchy $H$

Initialization :
The first level of grouping $G^0 = \{g_1, g_2, ...g_m\}, g_i = \text{object}_i (1 \leq i \leq m)$ ;
The current level $G = G_0$ ;
$H = \emptyset$ ;

Compute and sample the IBS ;
while $size(G) > 1$ do
    Define next level $G' = \emptyset$;
    $H \leftarrow H \cup G$;
    n = size($G$);
    compute matrix $\mathbf{M1}_{n*n}$: $\mathbf{M1}_{i,j} \leftarrow R_c(i,j)$ (Equation 4.1) ;
    compute $\mathbf{M2}_{n*n}$: $\mathbf{M2}_{i,j} \leftarrow \begin{cases} 1 & \text{if } g_i \text{ and } g_j \text{ have contact(s)} \\ 0 & \text{otherwise} \end{cases}$
    for $0 \leq i,j \leq n$ do
        if $\mathbf{M2} \neq \mathbf{0}$ then
            $\mathbf{M3}_{i,j} \leftarrow \mathbf{M1}_{i,j} * \mathbf{M2}_{i,j}$;
        else
            $\mathbf{M3}_{i,j} \leftarrow \mathbf{M1}_{i,j}$;
        end
        if $\mathbf{M3}_{i,j} > \tau$ then
            if $\exists g', g' \subset G', g_i \subset g'$ or $g_j \subset g'$ then
                if $g_j \not\subset g'$, $g\prime \leftarrow g\prime \cup g_j$, else $g\prime \leftarrow g\prime \cup g_i$;
            else
                build $g' \leftarrow g_i \cup g_j$ and $G' \leftarrow G' \cup g'$ ;
            end
        end
    end
    $G \leftarrow G'$;
end
$H \leftarrow H \cup G$
```

Algorithm 1: Automatic Hierarchy Construction

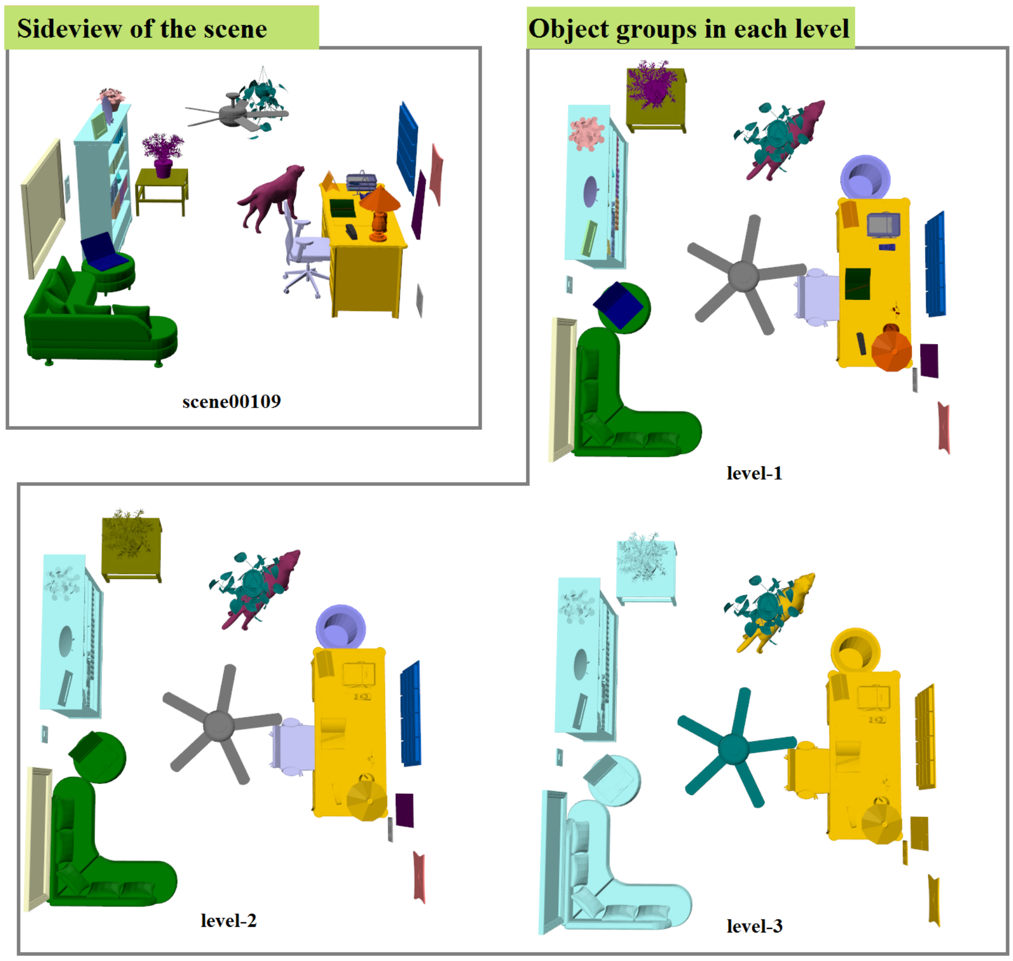


Figure 4.4: An example of the hierarchical structure. Objects in the same group are shown in the same colour.

of the generation of the hierarchical structure using a consistency measure (denoted here by γ) of how the merges happen under different parameter settings. Briefly speaking, γ $(-1 \leq \gamma \leq 1)$ is the difference between the probability that the “correct” and the “wrong” order occurs. See Appendix C for the details of computing γ. In our algorithm, the grouping threshold τ is the main parameter. We check the stability of the structure under different values of τ. For each scene, we compute five hierarchies by varying τ from 0.1 to 0.5 with an interval of 0.1. Then, we compute the γ for every pair within the five hierarchies. Finally, we use the mean of the γ, denoted here by $\hat{\gamma}$, as the indicator of the stability of our method. We compute $\hat{\gamma}$ for 130 scenes, and the average $\hat{\gamma}$ is 0.989, with the lowest stability of 0.779 (for scene00042).

4.3 Relationship-Based Retrieval

In this section, we describe how we use the features of the IBS and the scene structure computed in Section 4.2 to compute the similarity of interactions. We propose a method to represent the spatial relationship for objects in big 3D scenes in a hierarchical way based on the scene structure and define distance functions for comparing relationships with regard to different levels of detail.

4.3.1 Similarity Measure for Relationships Between Two Objects

Given an IBS, we can compute its features $\boldsymbol{f} = \{f^b, f^{\mathrm{PFH}}, f^{\mathrm{dir}}, f^{\mathrm{dis}}\}$. The four items are Betti numbers, PFH, direction, and distance respectively as explained in Section 3.4. To compare two IBS features $\boldsymbol{f}_1$ and $\boldsymbol{f}_2$, we first use a simple Kronecker delta kernel as a measure for the topological features:

$$\delta(f_1^{\mathrm{b}}, f_2^{\mathrm{b}}) = \begin{cases} 1 & \text{if } f_1^{\mathrm{b}} = f_2^{\mathrm{b}} \\ 0 & \text{otherwise} \end{cases} \tag{4.3}$$

Next we define a measure for the geometric features of the IBS that uses the normalized L_1 distance [1] of the PFH, direction and distance features:

$$\begin{aligned} d_{\mathbf{geo}}(\boldsymbol{f}_1, \boldsymbol{f}_2) = a \cdot L_1(f_1^{\mathrm{PFH}}, f_2^{\mathrm{PFH}}) + b \cdot L_1(f_1^{\mathrm{dir}}, f_2^{\mathrm{dir}}) \\ + c \cdot L_1(f_1^{\mathrm{dis}}, f_2^{\mathrm{dis}}) \end{aligned} \tag{4.4}$$

where $a+b+c = 1$ $(0 \leq a,b,c \leq 1)$. As the three features are in different ranges, we apply the *Inverse Variance Weighting scheme* [Hartung et al., 2011] to the L_1 distance of three features. We set $a = 0.1$, $b = 0.4$ and $c = 0.5$ in our experiments.

We combine the topology and geometry measures of the IBS, and compute the final similarity between two IBS by:

$$s_{\mathbf{sr}}(\boldsymbol{f}_1, \boldsymbol{f}_2) = \delta(f_1^{\mathrm{b}}, f_2^{\mathrm{b}})^w \cdot (1 - d_{\mathbf{geo}}(\boldsymbol{f}_1, \boldsymbol{f}_2)) \tag{4.5}$$

where w is a "switch" for using topological features. From the experiment in Section 4.1 we can see that the Betti number is quite useful for complex interactions like tangles or enclosures, while it can contradict geometric features for data that contains penetrations. The function for measuring the similarity between two sets of IBS features should be defined based on the nature of the dataset and the purpose of retrieval.

[1]The L1 distances between the PFH, direction and distance features are range from 0 to 2. We normalized them into the range [0, 1].

If the data mainly contains complex relations, w should be 1 so that the Betti number is used as a filter for different interaction types with respect to its topology; if the data mainly contains simple relations, which have Betti numbers $b_1 = 0, b_2 = 0$, w should be set to 0 to speed up the computation and avoid the influence of possible penetrations.

4.3.2 Similarity Measure for Local Communities

We now describe how we can compare two objects with respect to their local communities. We define a profile of object o_i in a local community $g = \{o_1, o_2, ...o_m\}$ by:

$$\boldsymbol{f}_{\mathbf{local}_i} = \bigcup_{1 \le j \le m, j \ne i} \boldsymbol{f}_{i,j} \tag{4.6}$$

where $\boldsymbol{f}_{i,j}$ is the feature computed from the IBS between o_i and o_j. Therefore, $\boldsymbol{f}_{\mathbf{local}_i}$ is the set of IBS features between o_i and all the other objects in g. We call $\boldsymbol{f}_{\mathbf{local}_i}$ the *local profile* of o_i. Given two objects o_i, o_i' from different communities g and g', their local profiles $\boldsymbol{f}_{\mathbf{local}_i}$ and $\boldsymbol{f}_{\mathbf{local}_{i'}}$ are first computed. Then we can compute the similarity between o_i and o_i' under the contexts of their local communities. We define a similarity measure $s_{\mathbf{local}}$, normalized in a way similar to the graph kernel normalization [Fisher et al., 2011]:

$$s_{\mathbf{local}}(i,i') = \frac{K(i,i')}{max(K(i,i),K(i',i'))} \tag{4.7}$$

$$K(i,i') = \sum_{\boldsymbol{f}_1 \in \boldsymbol{f}_{\mathbf{local}_i}} \sum_{\boldsymbol{f}_2 \in \boldsymbol{f}_{\mathbf{local}_{i'}}} s_{\mathbf{sr}}(\boldsymbol{f}_1, \boldsymbol{f}_2) \tag{4.8}$$

where $s_{\mathbf{sr}}$ is defined in Equation 4.5.

4.3.3 Similarity Measure for Extended Neighbourhood

After defining $s_{\mathbf{local}}$, we are ready to combine it with the hierarchical structure and define a profile for an object o_i at every level of the hierarchy. For a scene S and its hierarchy, we assume that the leaf nodes are at level 1. Let l_d denote the nodes at dth level so that l_d is a set of communities $\{g_1^d, g_2^d, ..., g_m^d\}$. Assume an object $o_i \in g_x^d, 1 \le x \le m$, a profile of o_i at the dth level is defined as:

$$\boldsymbol{f}_{\mathbf{ext}_i}^d = \bigcup_{1 \le y \le m, y \ne x} \boldsymbol{f}_{g_x^d, g_y^d} \tag{4.9}$$

where $\boldsymbol{f}_{g_x^d, g_y^d}$ is the IBS feature set computed from $IBS(g_x^d, g_y^d)$, which is the IBS subset shared by community g_x^d and g_y^d. Given two objects o_i and o_i', we can compute their profiles $\boldsymbol{f}_{\mathbf{ext}_i}^d$ and $\boldsymbol{f}_{\mathbf{ext}_{i'}}^d$. Then the similarity between o_i and o_i' at level d can be computed by:

$$s_{\mathbf{ext}_d}(i,i') = \frac{K_e(i,i')}{max(K_e(i,i),K_e(i',i'))} \tag{4.10}$$

$$K_e(i,i') = \sum_{\boldsymbol{f}_1 \in \boldsymbol{f}_{\mathbf{ext}_i}^d} \sum_{\boldsymbol{f}_2 \in \boldsymbol{f}_{\mathbf{ext}_{i'}}^d} s_{\mathbf{sr}}(\boldsymbol{f}_1, \boldsymbol{f}_2) \tag{4.11}$$

Finally, given a search depth parameter $d_{\mathbf{depth}}$, we can find the similarity between object o_i and o_i' by accumulating their similarities from level 1 to $d_{\mathbf{depth}}$:

$$s_{\mathbf{all}}(i,i') = \sum_{d=1}^{d_{\mathbf{depth}}} \gamma^{d-1} s_{\mathbf{ext}_d}(i,i') \tag{4.12}$$

where γ is set to 0.5 in our experiments. This is the contextual similarity for two objects in the database up to a given level.

A detailed example can be found in Figure 4.5. Assume that we want to compare side table t_1 and side table t_2 in two scenes. If $d_{\mathbf{depth}} = 1$, then only $s_{\mathbf{local}}(t_1,t_2)$ is calculated. The only objects involved are the objects on top of t_1 and t_2. If $d_{\mathbf{depth}} = 2$, then $s_{\mathbf{ext}_1} = s_{\mathbf{local}}(t_1,t_2)$ and $s_{\mathbf{ext}_2}(t_1,t_2)$ is calculated based on the IBS subset between the red areas and other areas within level 2. Finally, $s_{\mathbf{all}}(t_1,t_2) = s_{\mathbf{ext}_1} + 0.5 \times s_{\mathbf{ext}_2}(t_1,t_2)$.

Our idea to take account of the extended communities when comparing the status of an object in a scene resembles the part-in-whole queries in [Shapira et al., 2010]. In this paper, a hierarchical structure of objects is constructed, and when one part of an object is compared with other parts of another object, how the part is located with respect to the other parts in the hierarchy is taken into account and the similarity is computed based on the maximum flow in a bipartite graph. While their method focuses on the geometrical similarity of the parts in the hierarchy, our method computes similarities purely based on the relationship similarities. Also, we change the weights according to the distance such that the extended neighbourhood is less influential to the results; this is due to the nature of the data we handle.

4.3.4 Experiment and Evaluation

Experiment We tested the capability of our algorithm for content-based scene retrieval using a data set that consists of 130 scenes, which come from the Stanford Scene

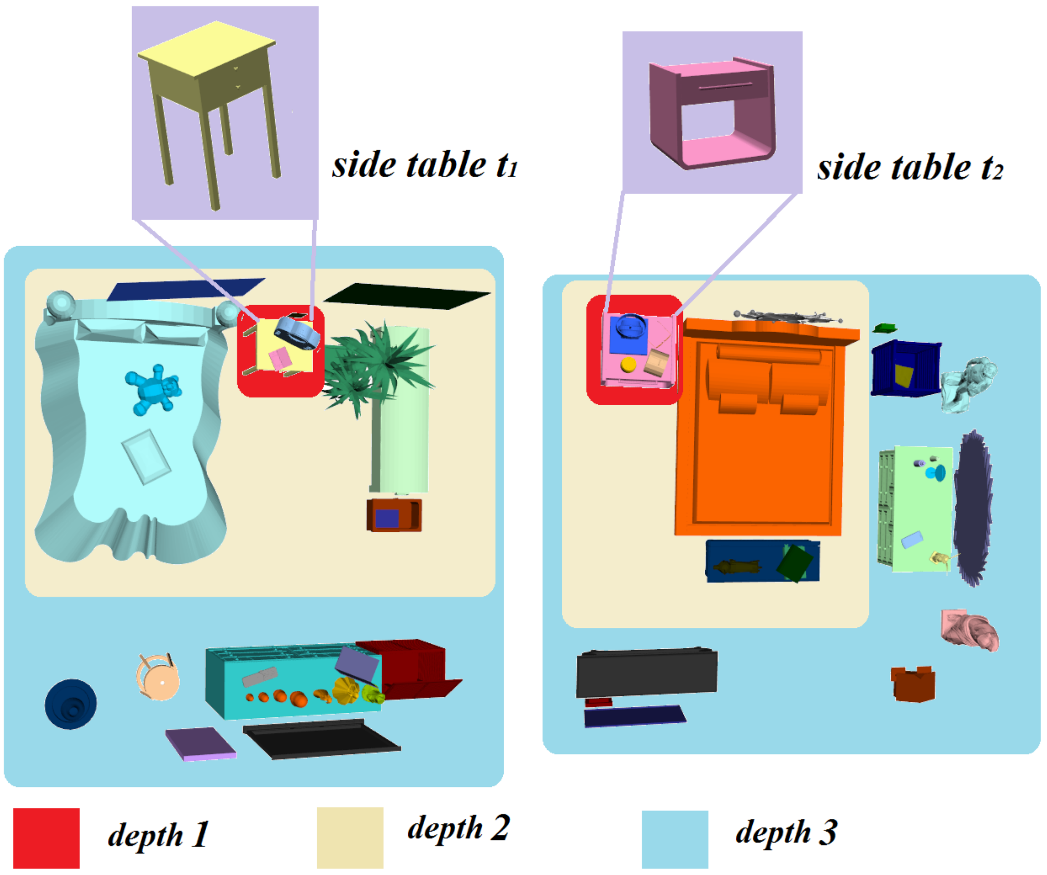


Figure 4.5: An example of hierarchical comparison. The two side tables are the "centre" objects we want to compare. The red, yellow and light blue regions contain the side table's neighbours on level 1(bottom level), level 2, and level 3 of the scene structures respectively. The 3D models are from the Stanford Scene Database [Fisher and Hanrahan, 2010].

Database. We first calculated the hierarchical structures using the techniques explained in Section 4.2. The user then selects any single object from a scene as a query. Then the system returns objects in any scene which have similar spatial relationships to the query.

Evaluation and Comparison

The retrieval results of an example scene with different depth $d_{\textbf{depth}}$ are shown in Figure 4.9. Compared with the results in Figure 4.10 that use the displacement features (DIS) [Fisher and Hanrahan, 2010], it can be observed that our system returns much more contextually similar results. We also show retrieval results for different queries in Figure 4.11. More retrieval results are presented in the Appendix E and F.

In order to evaluate our system, we prepared manually labelled data, which is pro-

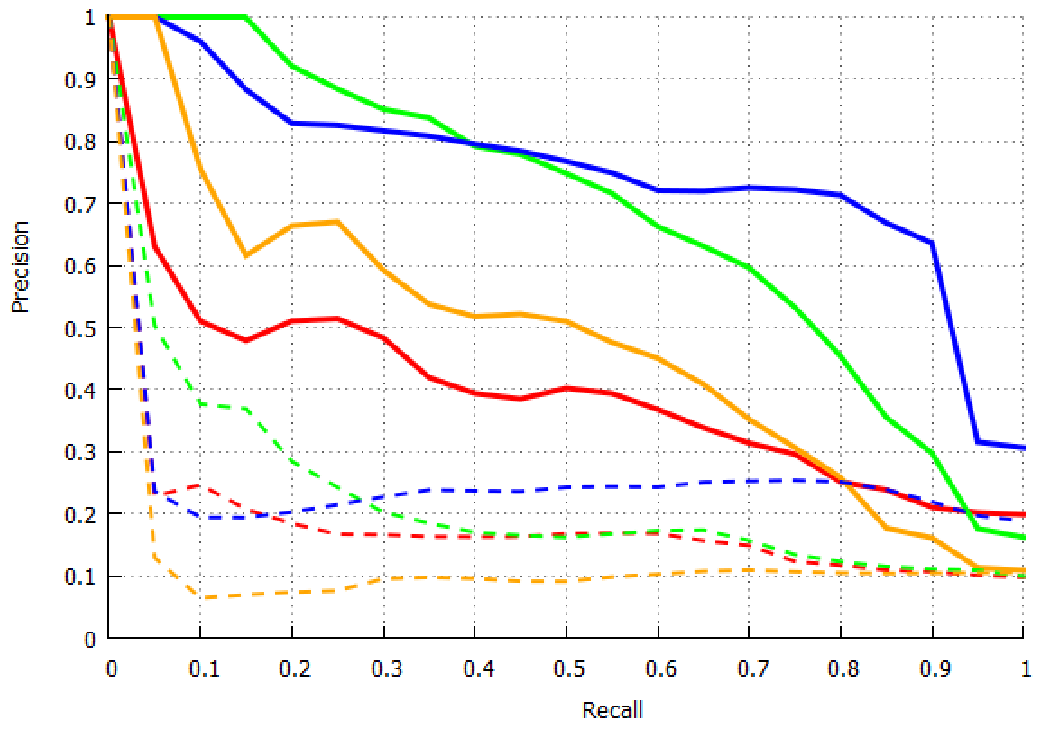


Figure 4.6: The solid lines show the precision-recall curve for our algorithm on four test scenes. The dashed lines show the precision-recall curve when using the DIS feature. The Ids of the four queries in the Stanford Scene Database are: scene00050-object8(red), scene00118-object14(green), scene00109-object14(blue), scene00087-object37(yellow).

duced as follows. Four query objects were selected from the Stanford Scene Database, and then for each query object an additional 500 objects are randomly selected from the database. The set of 500 objects were shown to the user with the scene in random order, and the user was asked if the object has similar spatial relations with the surrounding objects as compared to the corresponding query object. We label the spatial relations as similar if more then half of the users think they are similar. Ten users including students and staffs from different schools in the university took part in the user study. For more details about the user study, please refer to the Appendix D.

For quantitative evaluation of the results, the precision and recall curves (pr-curve) are drawn based on the search results and the ground-truth (manually labelled) data. In Figure 4.6, we present the curves based on our features (solid lines) and DIS (dashed lines). It can be observed that all of the solid lines show significantly higher precision than the dashed lines of the same colour, which indicates that our features out-perform the DIS feature. Our algorithm returns 50% of similar results with a precision of at least 40%, meaning that at least 1 in every 2.5 resulting scenes is desirable.

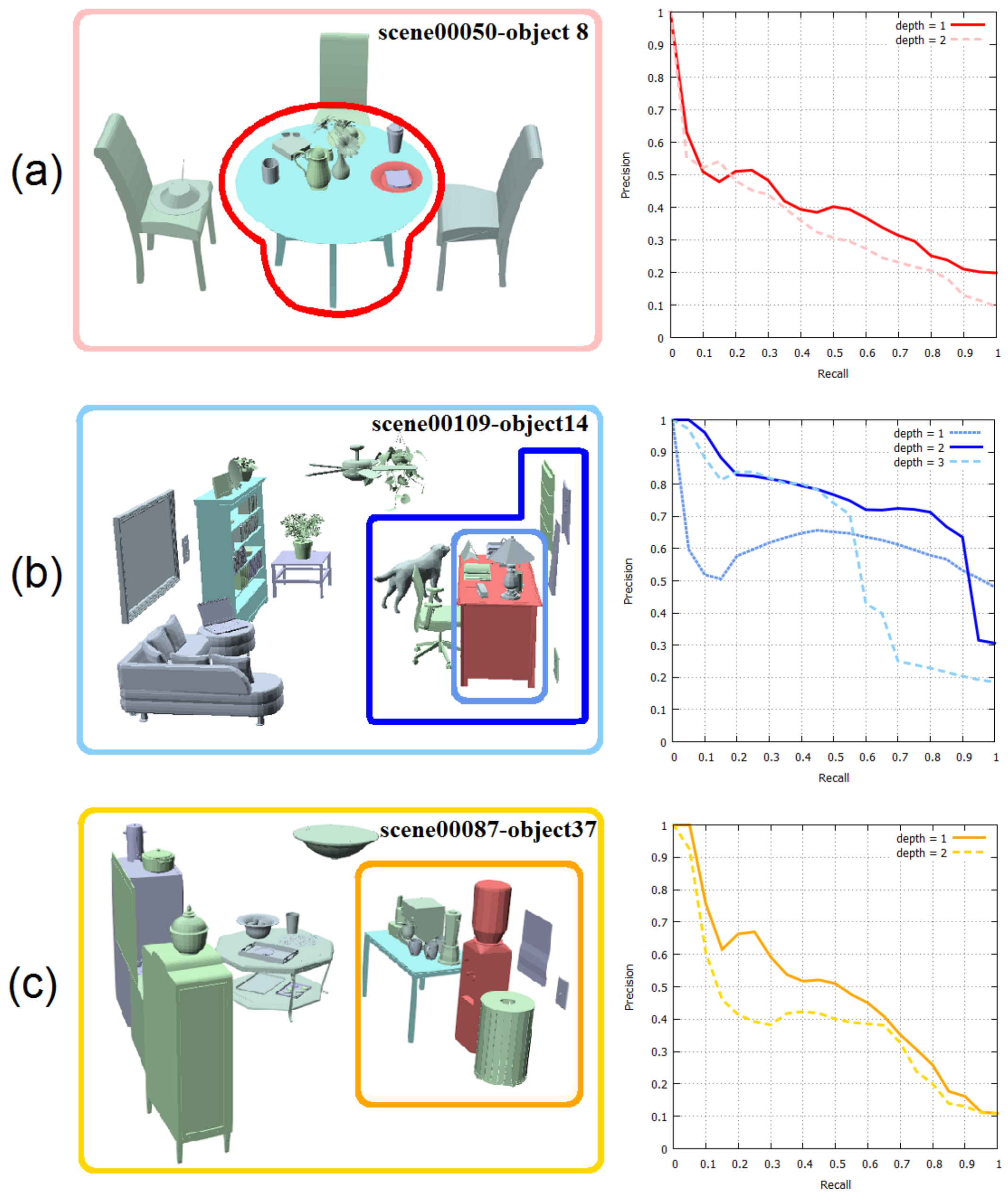


Figure 4.7: Scene regions and precision-recall curves for different d_{depths}. Each row corresponds to one query used for the user study. The red object in the scene is the query object. The boundary lines in the left column show the regions correspondent to the p-r curves of the same colour.

We now show the results of analysing the extent of the community ($d_{\mathbf{depth}}$) the users take into account when comparing the similarity of relationships. Figure 4.7 shows the pr-curves for three query scenes with different $d_{\mathbf{depth}}$ values. In Figure 4.7 (a), the pr-curve for $d_{\mathbf{depth}} = 1$ gives slightly higher precision than $d_{\mathbf{depth}} = 2$, which means that the users tend to pay more attention to the immediate neighbours of the "plate" (which are the toast, the table, and the other objects on the table) than those in the extended community. The results in Figure 4.7 (c) are similar to (a), but there is a larger difference in precision between the two depths. In Figure 4.7 (b), $d_{\mathbf{depth}} = 2$ gives the highest precision, which means the users also consider the objects in the extended communities when evaluating the similarity of the spatial relations. We can assume that factors such as the scale of the scene and the density of objects effect the perception of the neighbourhood. The scene in Figure 4.7(b) has more objects densely located around the desk compared with the desk in Figure 4.7(c) and the scale is larger than the scene shown in Figure 4.7(a).

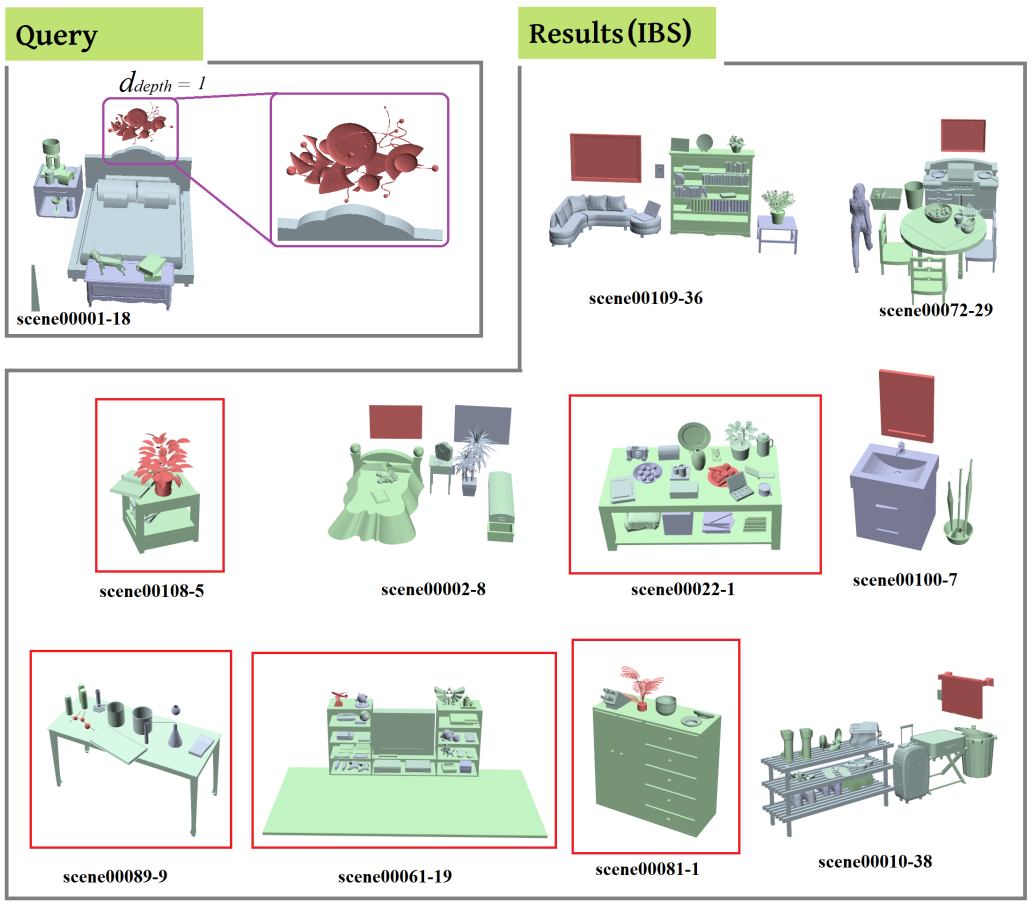


Figure 4.8: Failure case. The results with red bounding box are not satisfactory, as they are "object on table" whereas the query is "object hanging above another object".

Figure 4.8 shows a failure case for our method on the Stanford Scene Database, with half of the top ten results (object on table) not matching the query object (decoration hang above the bed). As the bottom of the decoration is almost in contact with the bed head, it is similar to the "one object on another" examples in terms of spatial relationship. This can be a typical failure case of retrieval results not being consistent with the user's intuition due to the fact that we do not take into account geometry information of the individual objects nor their semantic labels. The failure cases are mostly removed from the retrieval results when the decoration is lifted a little bit so that the relationship between the decoration and the bed head are not misunderstood as contacts.

4.4 Summary

In this chapter, we first showed how the features of the IBS can be applied to 3D scene classification. The rich information among numerous types of object relations is well contained in the IBS. Its capacity for describing these relationships lies in its topological and geometric features. Betti numbers enable us to recognize very sophisticated relations while the distance, the direction and the shape of the IBS further indicate nuances at a finer level.

In addition, we have proposed an automated mechanism to understand the structure of big scenes consisting of a large number of objects. Knowing the hierarchy of big scenes is crucial for applications such as content-based relationship retrieval. Because of the nature of the IBS, the calculation naturally rules out relationships between objects that are too far from each other or have too many objects between them. The structure of the IBS segments the scene into groups at every level. Therefore, we are able to automatically build up a hierarchical structure that contain meaningful groups. We also propose similarity metrics based on the IBS that effectively distinguish different types of spatial relationships between objects or object groups. These metrics are equipped with the scene hierarchy so that comparison can be made between objects based on their contexts.

The discriminative power of the IBS deteriorates when the distance between objects increases. When there are just two objects in the scene and they are far apart from each other, we suspect that IBS features can be replaced by simpler features used in [Fisher and Hanrahan, 2010] such as the height displacement and radial separation. As we focus on understanding relationships in this research, individual geometry plays a less important role. This is different from previous works. Hence, for applications such as retrieval, it might cause confusion when the user tries to retrieve scenes not only with similar relationships but also with similar geometries.

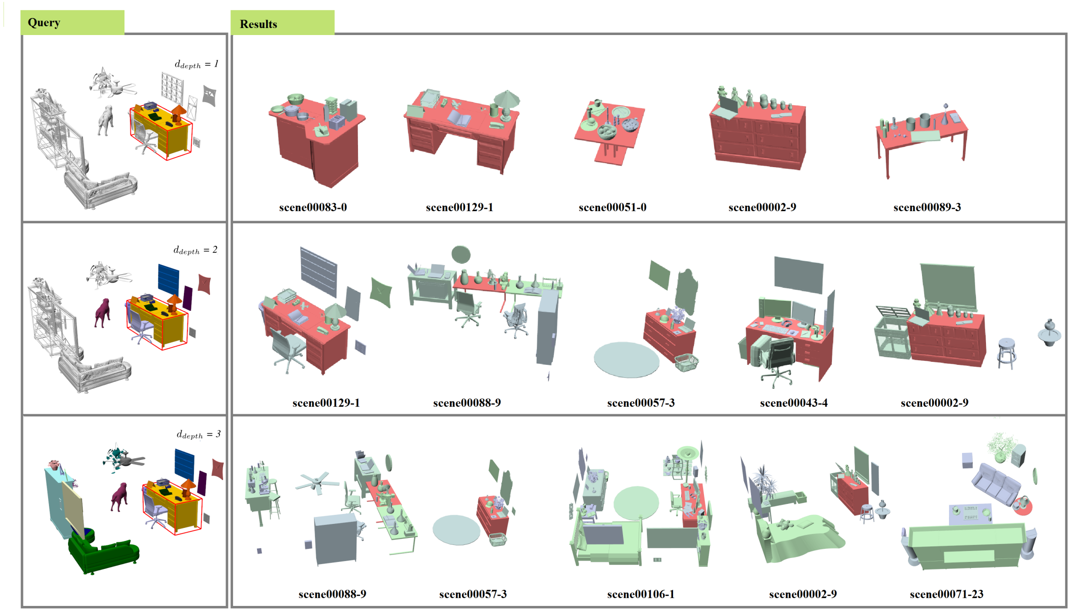


Figure 4.9: Retrieval results by using IBS features. In the query scene (left), the object with a bounding box (the desk) is the query object. We show the other objects within the search depth in colour while leaving the rest of the scene in grey. The resulting scenes from left to right are shown in order of similarity (right). The red object is the retrieved object with a similar context to the desk. We also only render the objects within the search depth for the resulting scenes.

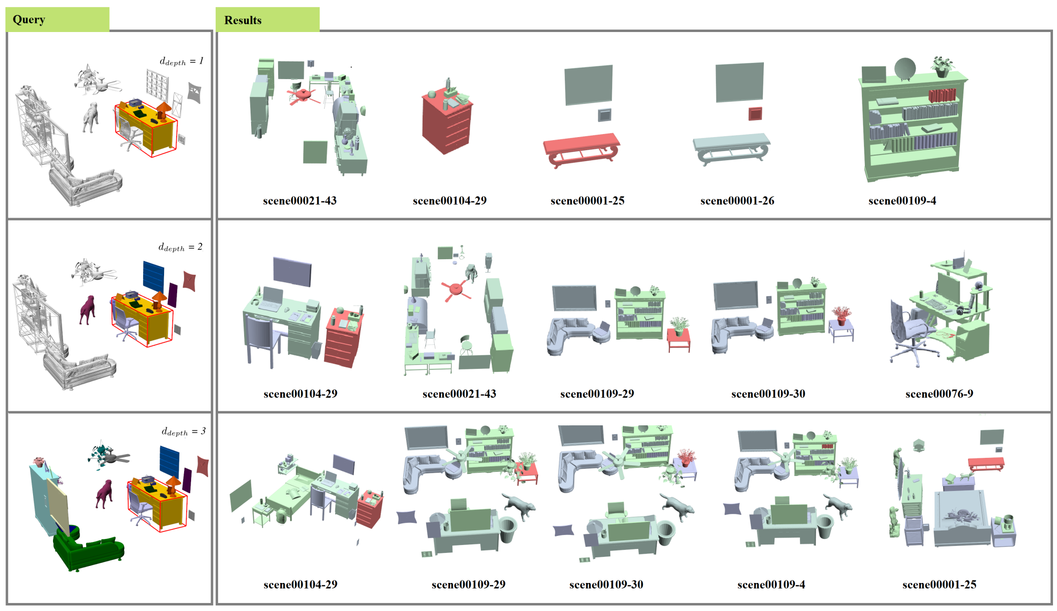


Figure 4.10: Retrieval results found by using the DIS feature.

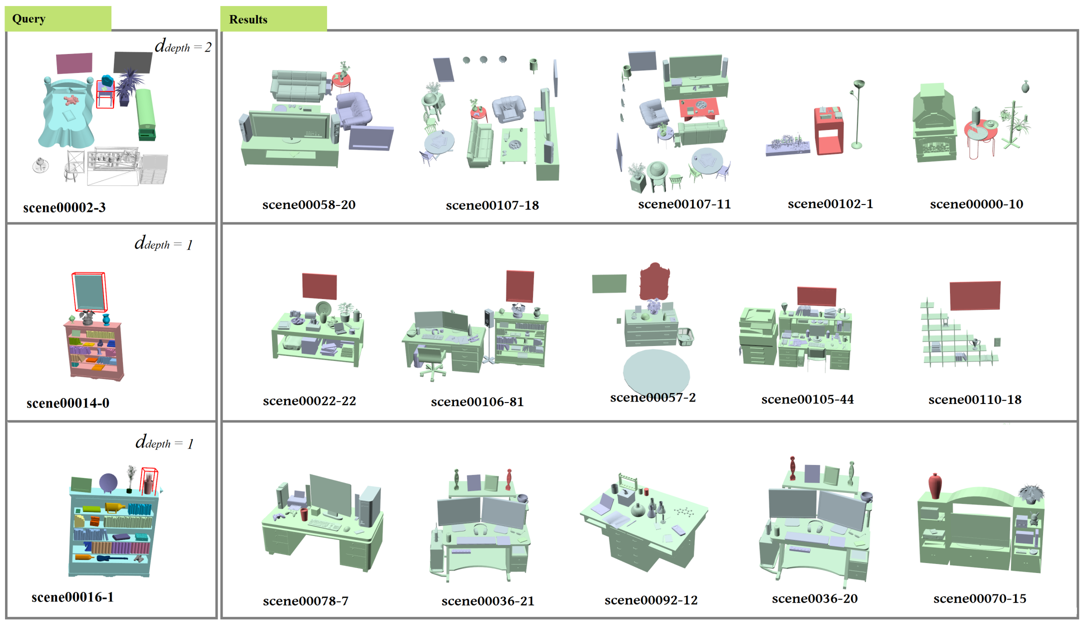


Figure 4.11: More retrieval results found by using IBS features. The 3D scenes are from the Stanford Scene Database [Fisher and Hanrahan, 2010].

We believe that the potential for using the IBS for spatial relation representation has not been fully explored. Potentially, any feature which can be extracted from 3D mesh can be used on the IBS. We hope to inspire new ideas for using other features on the IBS in different applications.

Chapter 5

Relationship-Based Character-Object Interaction Analysis

In this chapter, we describe a novel multiresolution approach to the classification and retrieval of interactions between human characters and objects by using the IBS structure described in Chapter 3. Based on a multiresolution representation of the body structure and the IBS between the character and the object, we compute a correspondence matrix hierarchy that describes which parts of the character's skeleton compose the IBS and how much they contribute to the interaction. Different scenes are then compared by considering both the correspondence matrix and the PFH feature of the IBS. Through the experiment results, we show that our IBS-based method outperforms existing techniques in motion classification and retrieval, which implies that contextual information plays a significant role for scene and interaction description.

5.1 Motivation

The number of scene databases that include both human activities and the environment is increasing due to ready-to-use 3D scanning devices such as the Microsoft Kinect. Such data includes both the 3D movements of the human, in the format of joint angles or joint positions, and the object or environment, in a point cloud or 3D mesh format.

Despite the growth in the size of such data, little work has been done for the indexing and retrieval of such data. Most existing work focuses either on human motion data, or object and static scene data. A naive solution for analysing the interaction data is to compute the features of the individual scene elements and couple them into an extended feature vector. Such a "bag-of-words" type of concept is not ideal for indexing and retrieval that require high precision. Because the precision of this type of approach is not very high due to the lack of contextual information that describes how the human is interacting with the objects.

Although the IBS representation proposed in Chapter 3 is well suited for extracting the geometric and topological nature of close interactions, contextual information about which part of the body is interacting with the object must be further provided when describing character-object interactions. An example is shown in Figure 5.1. Here, the geometry and topology of the IBS are very similar though humans will not recognize the two actions as similar, due to the large difference in the body parts involved in the interaction.

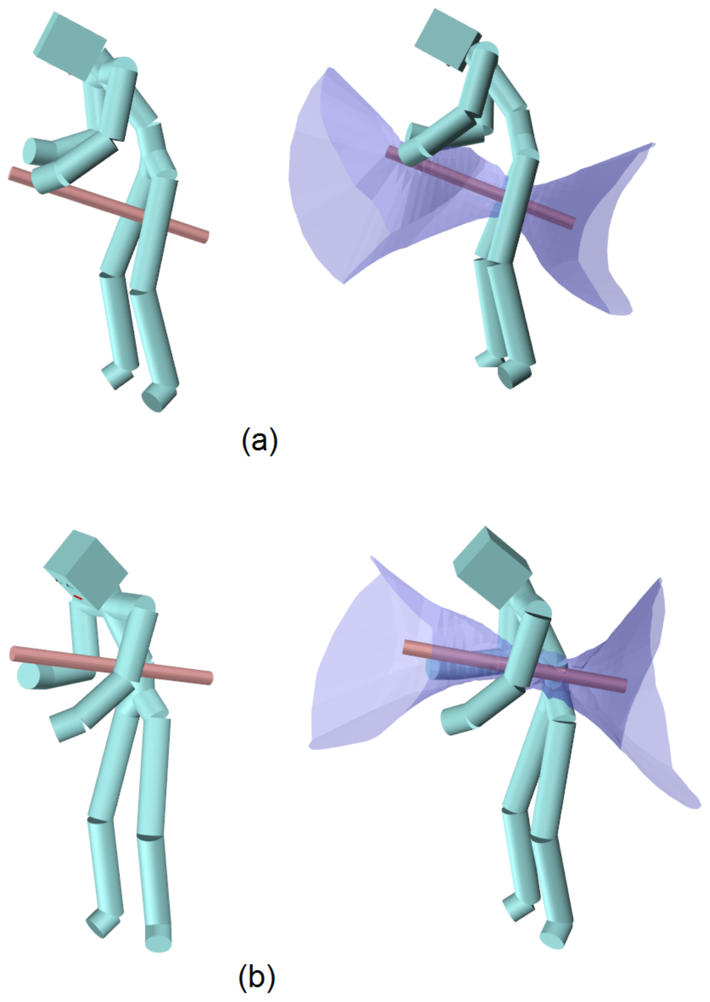


Figure 5.1: Different interactions which produce IBS with similar shapes: (a) a stick between legs, (b) a stick under the arm.

5.2 Objective

We propose a new approach to indexing and retrieving animation scenes where the human body closely interacts with objects using the IBS. The contributions of this chapter are as follows:

- A multiresolution representation that can describe the fine details of the interactions between a human and an object or the environment.
- A distance function that can compute the similarities of the interaction motions based on the context of the scene.
- An extensive evaluation of the proposed representation and distance function by comparing it with existing metrics.

5.3 Representation and Distance Function

In this section we propose a multiresolution feature of IBS to describe the interaction between a character and an object. After building a multiresolution structure of the character body (see Section 5.3.1), a multiresolution feature is computed based on the correspondence between IBS and the character body (see Section 5.3.2), and then we define a function to compare different interactions based on the multiresolution feature (see Section 5.3.3).

5.3.1 Body Hierarchy

Here we present a multiresolution structure of the body model, which is used for adding contextual information to the IBS and computing the similarity of two scenes.

We build a structure of the body as shown in Figure 5.3 based on the Biovision hierarchical skeleton (BVH) structure shown in Figure 5.2. The body is divided into smaller regions as we go down the hierarchy. At the first level, the body is divided into the upper and the lower region, and in the second level they are further divided into the limbs and the torso areas. Once we reach the individual rigid bones (i.e., the upper arm, head, thigh), we further segment the body into smaller regions. Here we represent each body part by cylinders. They are segmented along the central axis and the circumference. However, this can be generalized to arbitrary 3D polygon meshes using subdivision schemes such as [Peyre and Cohen, 2004].

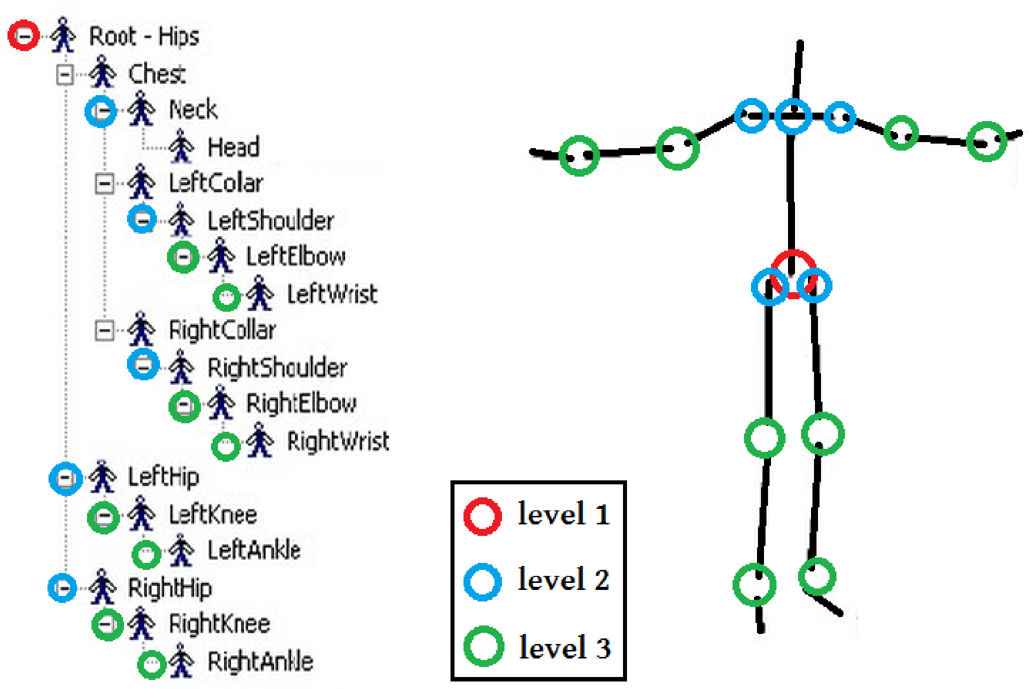


Figure 5.2: BVH skeleton structure. The circles indicate the segmentation locations of the first three levels.

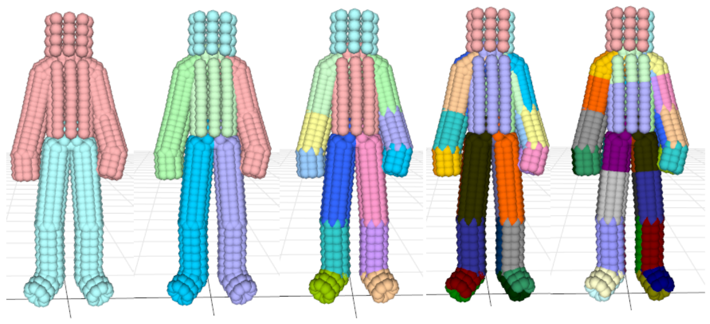


Figure 5.3: The character body is segmented into 2, 6, 14, 28 and 38 regions. Each colour corresponds to a unique label.

5.3.2 Representation for Interaction

In order to represent the context of the 3D scene, we make use of the IBS and further enhance it with a new feature to handle interactions between human characters and 3D scenes. Next, we explain how to compute this new feature.

We evaluate the details of the relationship between the body and the object by examining how much each part of the body contributes to the IBS. This is done by projecting the multiresolution structure of the body defined in the previous section onto the IBS according to the correspondence information between the IBS and the body-part of the character, which can be obtained when computing the IBS. For each ridge on the IBS r_i, there is a pair of corresponding samples (s_i^c, s_i^o). s_i^c belongs to the character body, and s_i^o to the object. We then give the s_i^c's label to r_i. We show in Figure 5.4 the labelled IBS which correspond to different levels of the character body segmentation.

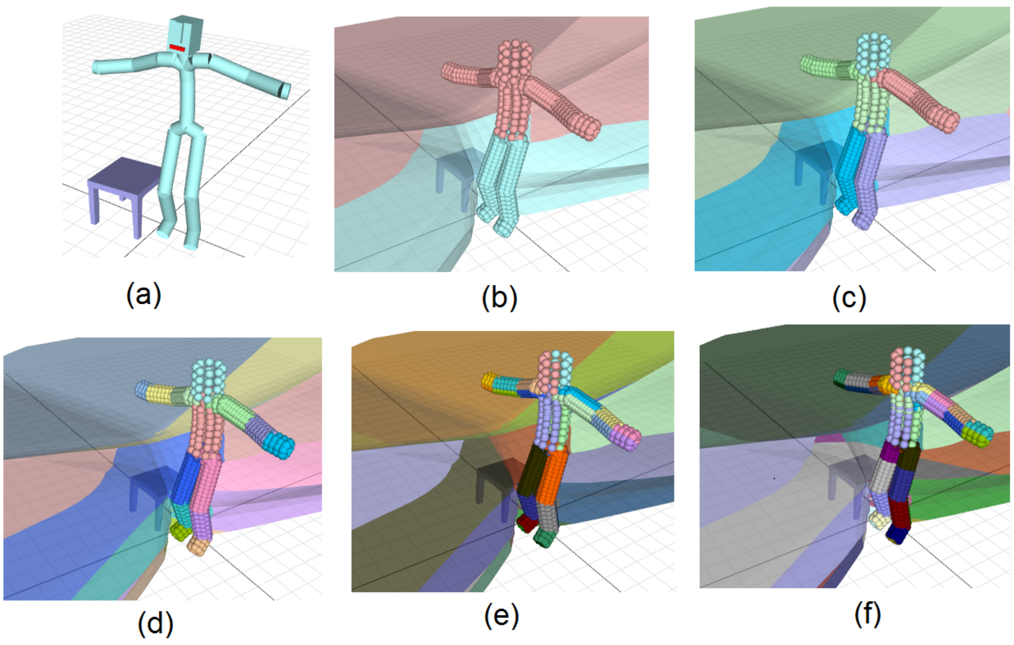
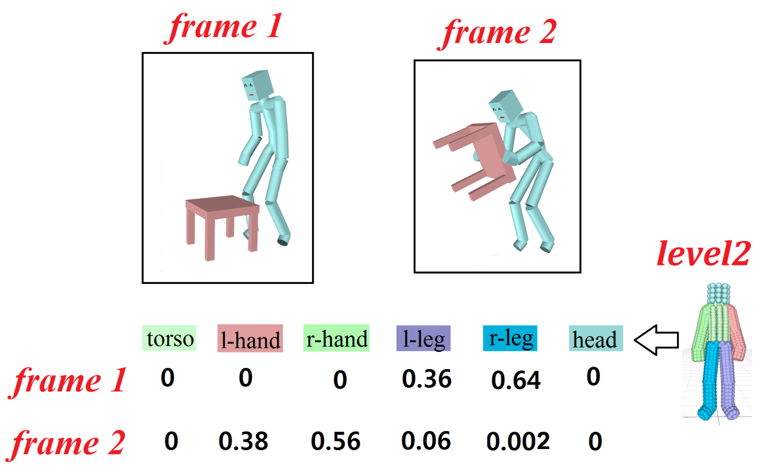


Figure 5.4: Labelling the IBS according to the character segmentation: (a) the original scene: the character stands by the desk; (b) the character is segmented into two parts: the upper body (red) and the lower body (green), and the label information is passed to the corresponding ridges on the IBS. The red area on the IBS corresponds to the upper body and the green area the lower body; (c)(d)(e)(f) projecting other levels of body segmentation to the IBS.

We can then define a feature vector set $\mathbf{V} = \{\mathbf{V}^1, \mathbf{V}^2, ..., \mathbf{V}^N\}$ based on the labelled IBS, where N is the number of resolutions. Let us assume that we are evaluating the segmentation of IBS at resolution r. In the current resolution r, the character body is segmented into n_r regions, each of which is defined as b_i^r $(0 \leq i < n_r)$. Let us then assume that there is a corresponding region on the IBS, which is defined as IBS_i^r (Note that not all the b_i^r has corresponding IBS_i^r). The relationship feature of the current level $\mathbf{V}^r$ is a n_r dimensional vector, whose items correspond to the body regions and indicate how much the body region contributes to the interaction. The element i of the correspondence vector $\mathbf{V}^r$ is defined as:

$$\mathbf{V}^r(i) = \begin{cases} w_i & \text{If } IBS_i^r \text{ exists} \\ 0 & \text{otherwise.} \end{cases} \tag{5.1}$$

in which the weight w_i is computed by summing up the weight of each ridge w_j^{ridge} in IBS_i^r. w_j^{ridge} is computed by Equation 3.1. The zero value in $\mathbf{V}^r$ means that the corresponding body part does not produce any IBS region, and thus does not contribute to the interaction. Figure 5.5 shows the correspondence vectors at level 2 of two different frames in a "pick up the table" motion.



	torso	l-hand	r-hand	l-leg	r-leg	head
frame 1	0	0	0	0.36	0.64	0
frame 2	0	0.38	0.56	0.06	0.002	0

Figure 5.5: Example correspondence vectors for two frames.

Theoretically, we can compute the multiresolution structure for both parts of the scene and project the structure on to the IBS. In this case, each region of the IBS corresponds to a pair of segments of both interaction parts; and $\mathbf{V}$ for each resolution can be represented by a 2D matrix. The multiresolution segmentation can be done on any type of data (mesh, point clouds, skeleton, etc). As long as we can find the correspondence between the scene objects, we can compare the similarity between IBS projected regions. However, we only segment the character skeleton here, not the object. This is because the geometry of the objects with which the character interacts can be inconsistent. We have found it more practical to compute a vector based only on the body labels.

5.3.3 Distance Function

We now describe how we compute the distance between two configurations represented by the multiresolution vectors that we defined in the previous section. Ideally, the distance function must quantify how much movement is needed for the body to move from one state to the other while not colliding with the object. Due to the large degree of freedom of the human body, such a motion planning problem is extremely difficult to solve and requires a huge amount of computation. Here we provide an approximation by assuming that the object shifts around the body and that the body has enough flexibility to avoid collisions when moving from one configuration to another.

Simply comparing two feature vectors at one resolution, say the highest resolution $\mathbf{V}^N$, does not result in a value that describes how much movement the object needs to shift around the body. This is not ideal for indexing and retrieval, where the scenes must be ordered according to the similarity.

Our solution is a multiresolution distance function, which is defined as follows:

$$d_{multi}(\mathbf{V},\mathbf{V}') = \sum_{r=1}^{N} w_r \cdot \frac{1}{n_r} \|\mathbf{V}^r - \mathbf{V}'^r\| \tag{5.2}$$

where w_r is weight for level r and n_r the length of the vector in level r. Figure 5.6 shows an example of computing the distance between two frames. In practice, we set w_r as the r-th power of 0.5. Even though the dissimilarity of the configurations cannot be fully described at a single level, it can be computed as we go up the hierarchy and consider more levels of the detail. To show the change of $\mathbf{V}$ during the motion, we computed the distance between frames of a “pick up the bin” motion by using Equation 5.2 and visualized the distance matrix in Figure 5.7.

To compare two interactions, we also consider the difference between the PFH features that we defined in Section 3.4. This is necessary as the multiresolution feature only describes the local proximity, so that the overall difference of postures cannot be distinguished. For example, configurations shown in Figure 5.8 (a) and (b) will result in similar correspondence features. So both the local and global features of the IBS are taken into account for the final distance:

$$d_{relation} = w_1 \times d_{multi} + w_2 \times d_{PFH}. \tag{5.3}$$

In the next section, we will explain how we use this distance measure for comparing two interaction motions.

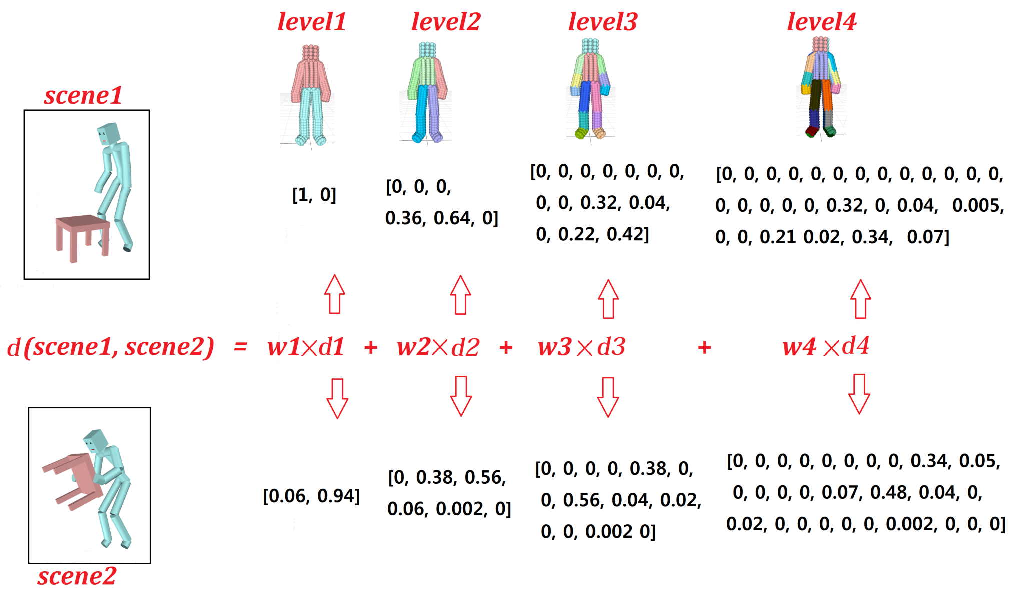


Figure 5.6: The correspondence vectors and the computation of the multiresolution distance.

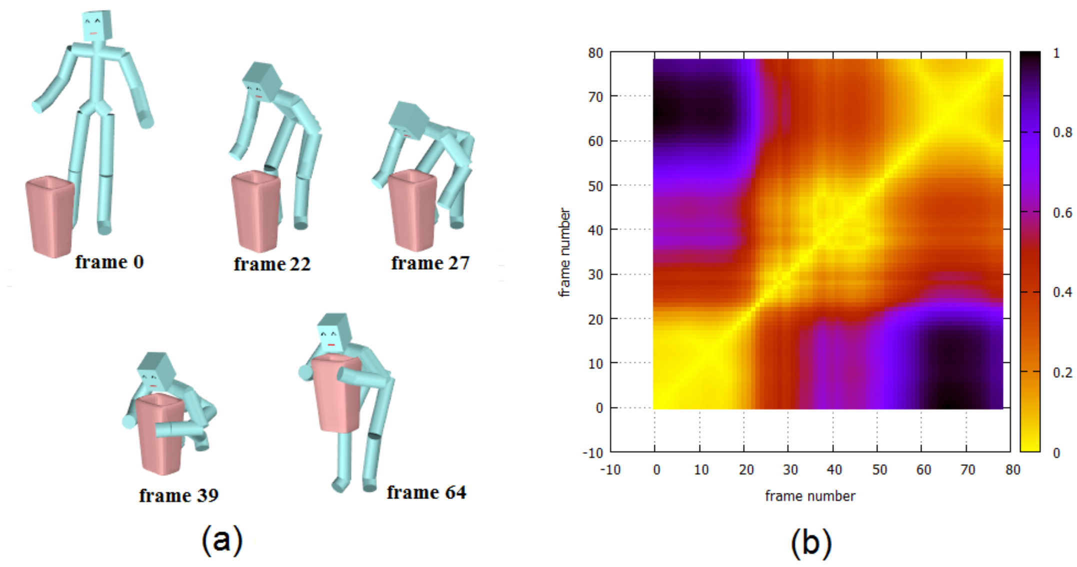


Figure 5.7: (a) the "pick up the bin" motion, (b) the distance matrix of this motion. Item $(i, j)$of the matrix is d_{multi} between frame i and frame j. The brighter colour indicates lower distance.

5.4 Motion Comparison

In this section, we first extract keyframes, which are the representative frames of the entire motion (see Section 5.4.1). Then we compute the distance between different motions by Dynamic Time Warping (DTW) in Section 5.4.2.

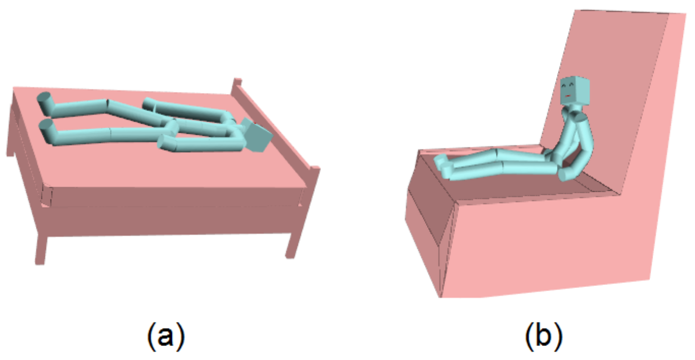


Figure 5.8: The global shape of the IBS is also important: (a) and (b) are example scenes which produce similar correspondence vectors, while their IBS have different global shapes.

5.4.1 Key-frame Extraction

The keyframe extraction is done to abstract the motion so that the memory usage and the computational cost for motion indexing and retrieval can be reduced.

We compute keyframes based on the context of the scene. Previous methods, which are only based on the posture of the characters, is not suitable when the scene involves close interactions of multiple characters and objects. Here we use the distance function defined in the previous section to extract the frames that are different with respect to the spatial relationships.

We first build a distance table by computing the distance between each pair of frames in one motion by using Equation 5.2. The first two keyframes are the two frames with the maximum distance. To add another keyframe, we use the "farthest point strategy", which is finding the frame farthest (in other words, which has maximum distance) from existing keyframes. If this distance is greater than a threshold, it is chosen as a keyframe. This process comes to an end when no more frame can be added to the keyframe list. We set the threshold to 0.15 in our experiments. Figure 5.9 shows two series of keyframes computed by this method.

The keyframes are used as the representative frames of each interaction. When computing the distance between different interactions, we apply DTW, which will be explained in the next section, only to the keyframes rather than the entire motion, thus reducing the online costs and memory.

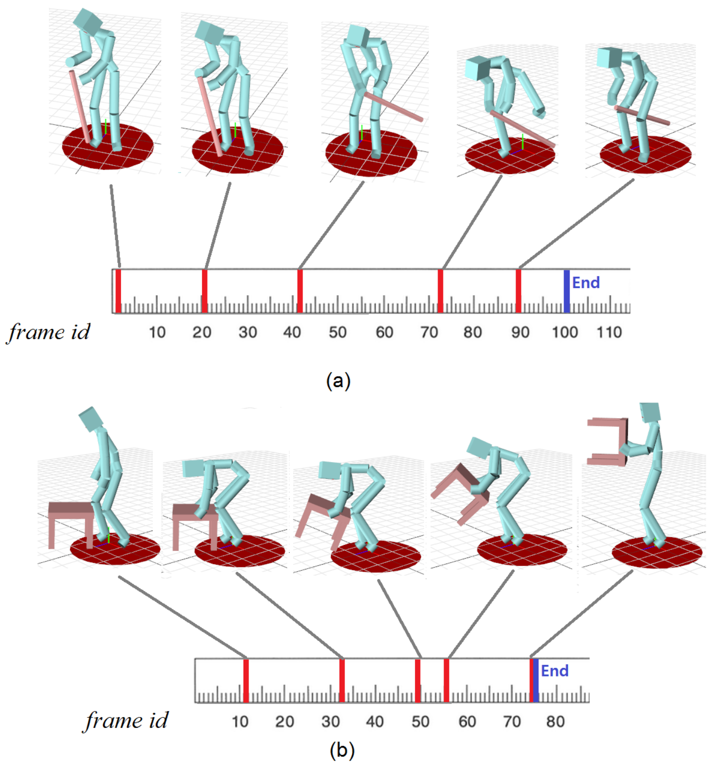


Figure 5.9: Examples of the key frames for motion (a) "ride on the stick" and (b) "pick up the table".

5.4.2 Dynamic Time Warping for Motion Comparison

Dynamic Time Warping (DTW) is a technique to align different time-series data, which is widely used for the alignment of speech, motion and video data. The key idea of DTW is to compute a cost matrix between every pair of input frames, and find a minimum cost path, which is typically located near the main diagonal of the matrix. The readers are referred to [Müller, 2007] for a more comprehensive explanation of DTW and its application.

We can get a similarity measure between two motions by applying DTW to the series of keyframes extracted using the method described in the previous section. Selecting the appropriate features and distance measure of the frames is the key of using DTW. In our case, the distance between frames is computed by Equation 5.3. In the next section, we will show motion retrieval results using this method.

Object	Number of motions	Object	Number of motions
1. Large bin	19	8. Hula hoop	30
2. Small bin	22	9. Pistol	3
3. Box	25	10. Rifle	5
4. Chair	12	11. Ball	7
5. Table	25	12. Book	2
6. Broom	17	13. Cup	3
7. Tube	14	14. Hat	1

Table 5.1: Interaction Motion List

5.5 Experiments and Results

In this section, we first describe the interaction dataset in our experiments (Section 5.5.1), then the results of a single frame retrieval (Section 5.5.2), and lastly the retrieval of the motions (Section 5.5.3).

5.5.1 Dataset

The interaction data used in our experiments includes both the motion of the character and the object. We follow the approach in [Sandilands et al., 2012a], where a magnetic motion capture system is used to track both the movements of the body parts and the object. The main advantage of using this magnetic motion capture system is that it does not suffer from occlusion problems. The system records both the translation and orientation of the each sensor, so the configuration of each rigid bone/object can be recovered accurately by a single sensor.

For the experiments in this section, we captured 164 short interaction motion clips between human and different objects. We used 14 sensors on the actor and 2 sensors on the object. The sensor locations on the actor are shown in Figure 5.10. We also used 21 motion clips captured by Sandilands et al. [2012a]. In total, there are interactions with 13 types of objects. The description of the objects and the motions are listed in table 5.1. All the motion data used in our experiments have been published in the Edinburgh Interaction Database [Sandilands et al., 2012b].

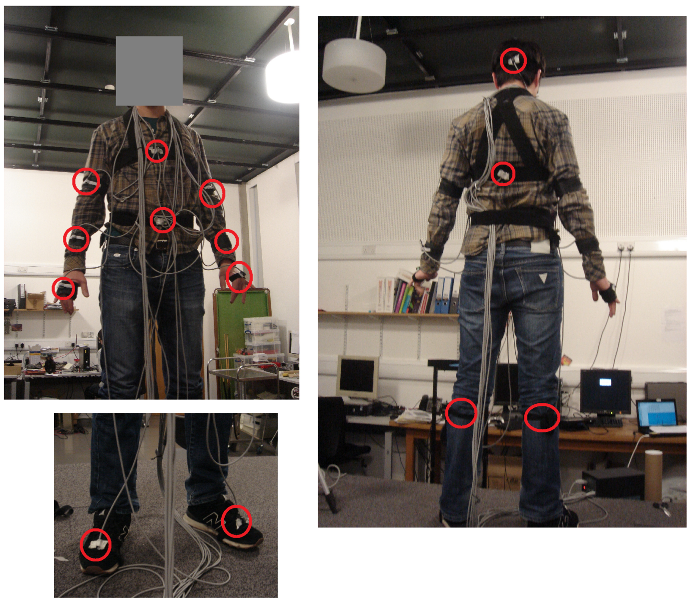


Figure 5.10: Locations of magnetic sensors (highlighted by red circles) on the body when capturing the motions.

5.5.2 Retrieval by Single Frame

To examine the multiresolution feature and the distance function we proposed in Section 5.3, we conducted a frame-based retrieval. Given a query frame, our system compares it to all the frames in the database and ranks the results according to the similarity. The query can be a frame from an existing interaction in the database or a scene designed by the user. In Figure 5.11 we show the retrieval results for four example queries using Equation 5.3 (considering both multiresolution feature and the PFH feature).

We also compare the retrieval results when using different features including PFH feature (PFH), multiresolution feature (MULTI), as described in Section 5.3.2 and Section 5.3.3, two features together (PFH+MULTI), and a traditional position-based feature denoted as POS+OBJ. POS+OBJ feature is a vector whose items include the position of the body joints (POS) and the centre position of the object (OBJ). The object position is defined in the character's root coordinate system. The difference between POS+OBJ features is measured by $L1$ distance. Figure 5.12 shows retrieval results of using these features. We compare the proposed feature (PFH+MULTI) with the others and discuss about the advantage of the proposed feature below:

MULTI: As the query is a frame where the character is sitting on a chair, the system returns interactions where the object is in proximity to the legs when the MULTI feature is used. However, as the MULTI feature does not consider the overall shape of the

IBS, it cannot distinguish the “ride on” posture from other types of interactions where the legs are in proximity to the objects.

PFH: The PFH feature extracts the characteristics of the body which is half wrapped around the object, and therefore such postures are returned as high-rank frames. But as the PFH feature does not contain any information on which body part produces the interaction, sometimes the system returns arm-related interactions rather than leg-related ones. Compared with MULTI and PFH, the MULTI+PFH returns more relevant results, which are the scenes with an object between the legs.

POS+OBJ: When POS+OBJ feature is used, some unrelated interactions come to the top (such as holding a rifle). The context of the scene is not explicitly described in this feature and some interactions where the character takes similar postures under different contexts can be ranked high in this case.

Evaluation of Multiresolution Distance

To justify the usage of the multiresolution distance, we show the results of using different sets of resolutions for frame-based retrieval in Figure 5.13. As in previous experiments, four levels of resolutions are prepared for the character body. From the top to bottom, we show the results of computing the multiresolution distance when gradually increasing the resolution. Considering the spatial relationship between the character and the object in the query, the most relevant results should be the character carrying an object on the right shoulder. From the results we can see that the most similar scene is moving to the top rank when we add in more levels of details.

5.5.3 Retrieval by Motion

In this section, we do the retrieval of interaction motion by using three types of features: MULTI+PFH (Section 5.3.3), POS+OBJ (Section 5.5.2) and PFH (Section 3.4). The retrieval results based on these features are compared and evaluated.

We manually classify 185 motion clips into the following four categories and use this classification as the ground truth:

- Pick up an object (39 clips)
- Put down an object on the floor (33 clips)
- Hold and manipulate an object (34 clips)

- Stepping over an object (27 clips)

Other 52 motions are labelled as "other" which are dissimilar from each other.

To show the resulting similarities of all motions in the database, we visualized the similarity matrix in Figure 5.14. The final motion similarity values are between 0 (low similarity) and 1 (high similarity). From Figure 5.14(a) we can see that by using IBS-based measure, the intra-class distance is generally smaller than the inter-class distance (except in the "other" class), while in Figure 5.14(b) the location-based distance does not show clear patterns of classes.

To quantitatively compare the retrieval performances, we plotted the precision-recall curve of the four classes of motions when using the MULTI+PFH (solid red), PFH (dashed orange) and POS+OBJ (dotted blue) distance in Figure 5.15.

MULTI+PFH preserves more information about the spatial relationship between the body and the object and therefore performs much better than the POS+OBJ. One way to improve the performance of the POS+OBJ feature is to increase the numbers of sample points on the object. However, there is a difficulty in deciding how to sample such points, as the geometry can be significantly different among objects and finding correspondence between objects is not easy.

We can also see that the MULTI+PFH performs much better than PFH. Because PFH does not consider which body parts are contributing to the interaction, as has been discussed in Section 5.1. The experiment results support our reasoning of PFH's limitation and show that the multiresolution feature is an important supplement for describing character-object interactions.

Two examples of the retrieval results when using the proposed MULTI+PFH distance are shown in Figure 5.16 and Figure 5.17.

5.6 Conclusion

In this chapter, we explored the spatial relationship representation for 3D character-object interactions. We argue that using a representation that explicitly describes the spatial relationship between the body and objects can greatly improve the performance of indexing dynamic scenes. We developed a new multiresolution representation that describes the spatial relationship between a character and an object, and use it for keyframe extraction and motion retrieval. Through the experiments we showed our representation performs much better than existing ones.

Although motion analysis is a well developed area, very few work copes with the problem of character-object interactions. Also, previous methods of retrieval mainly focus on individual items, such as the kinematics of characters, object geometry, manually provided labels or simple contextual information based on relative vectors and contacts. There has been little attempt to define a representation that describes the rich nature of context.

One advantage of indexing scenes based on the spatial relationship is that we can be relieved from analysing the fine details of the object geometry, which is far more difficult than analysing the IBS. Objects can have arbitrary topology, making the computing of the correspondence of objects a very difficult problem. In contrast, it is easier to find the correspondence between character bodies. Comparing the context by our method becomes a much simpler problem than comparing the geometry of all the objects in the scene.

We did not use the topological information of the IBS such as the Betti numbers, because the body can easily penetrate the objects in dynamic scenes due to the low accuracy of the motion capture data. We can only attach one sensor per rigid body part and we have only 20 sensors for the entire body, which is not enough for capturing the fine details of the body movements.

In the future, we would like to try our method on different types of interaction data. For example, the interaction data captured by the Kinect in [Peursum et al., 2004, 2005]. One big challenge is that the objects captured by camera are not complete models. The incompleteness of the object geometry does influence the accuracy of IBS. Finding a way to overcome this problem is one direction of our future research.

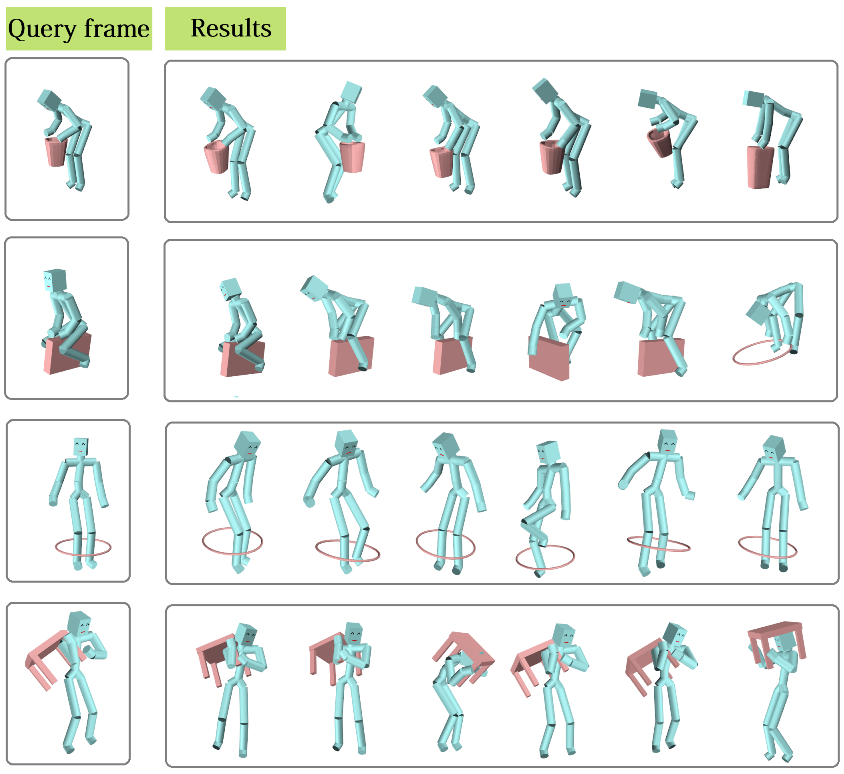


Figure 5.11: Queries and retrieval results by using Equation 5.3 (MULTI+PFH).

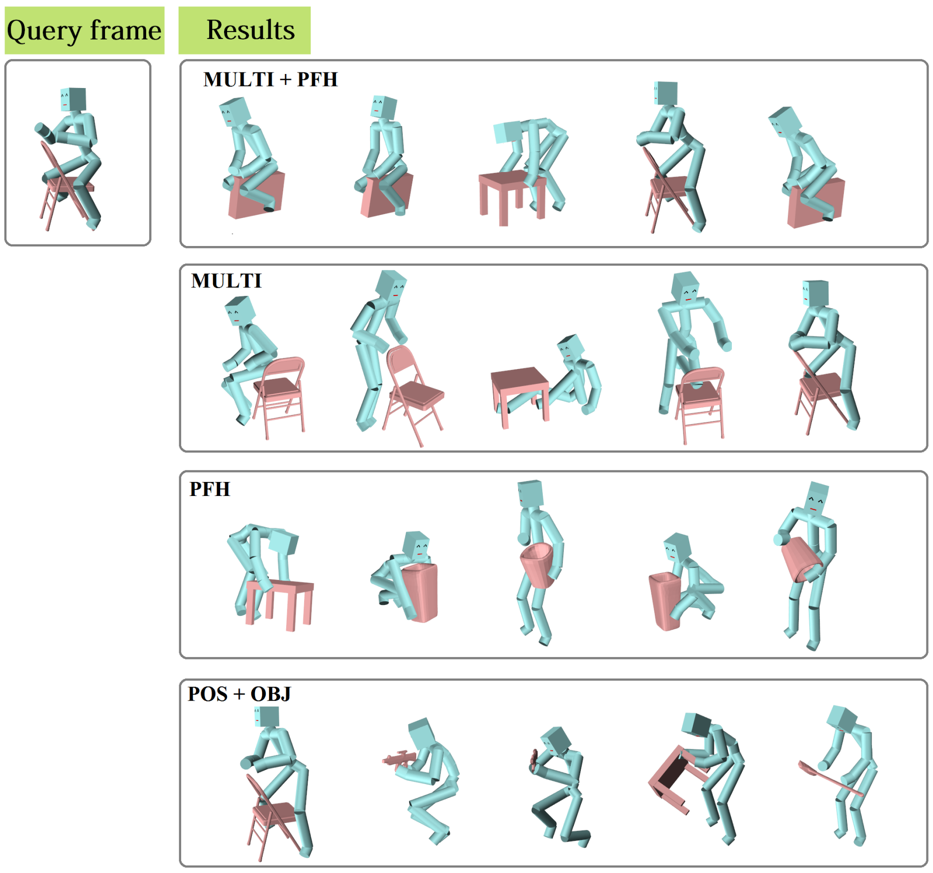


Figure 5.12: Comparison of the retrieval results by using different features.

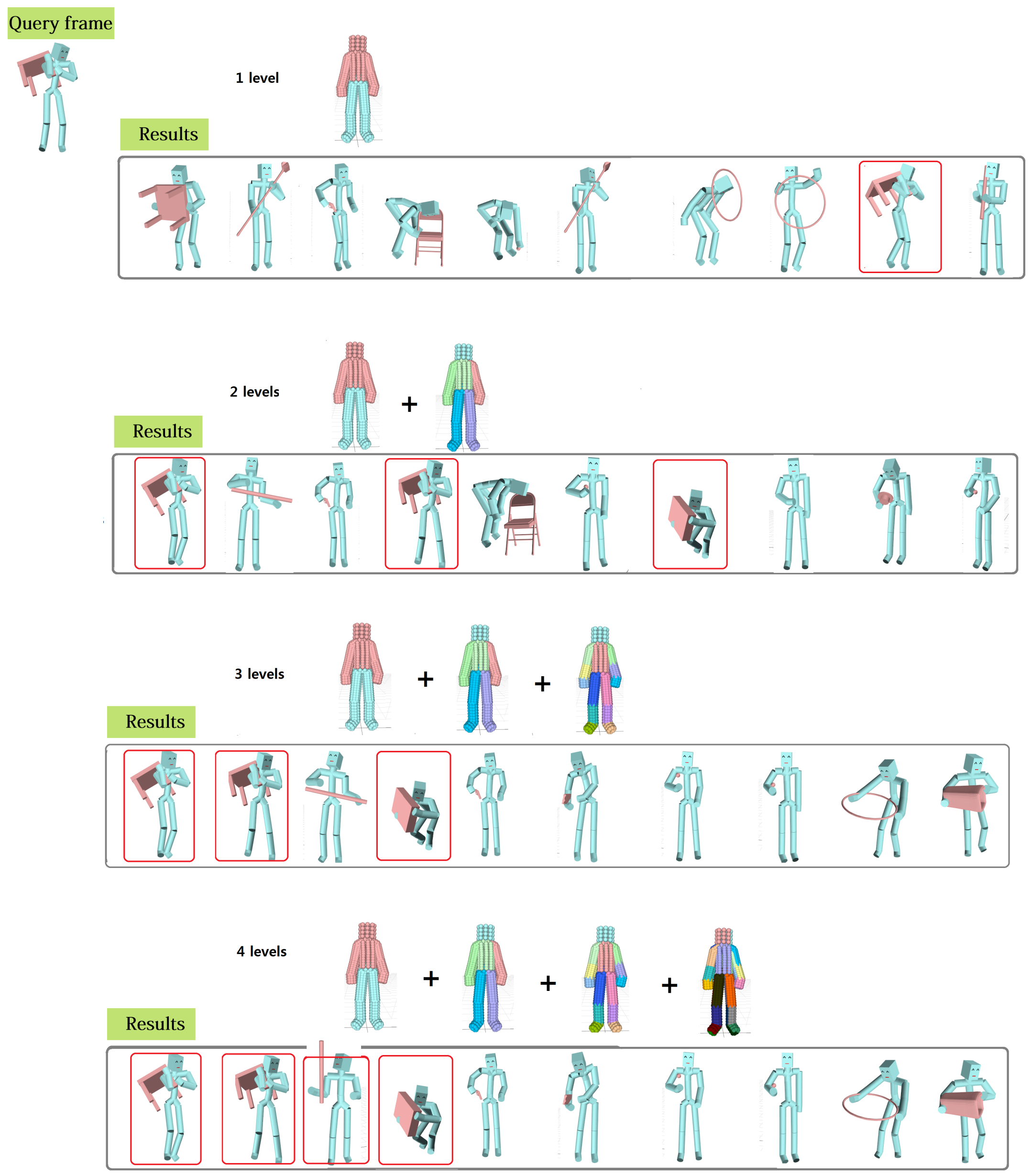


Figure 5.13: The most similar frames for different level settings. The results are the most similar frames in the order of similarity. The good results are highlighted with red borders. This figure aims to show how the retrieval results can be improved when higher levels of detail are added.

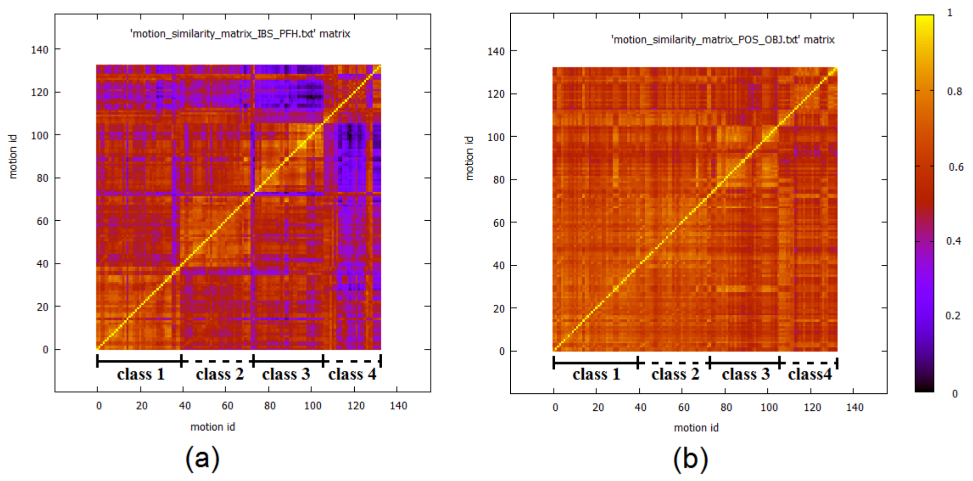


Figure 5.14: Similarity matrix of our database which shows the similarity between motions based on features (a) MULTI+PFH and (b) POS+OBJ.

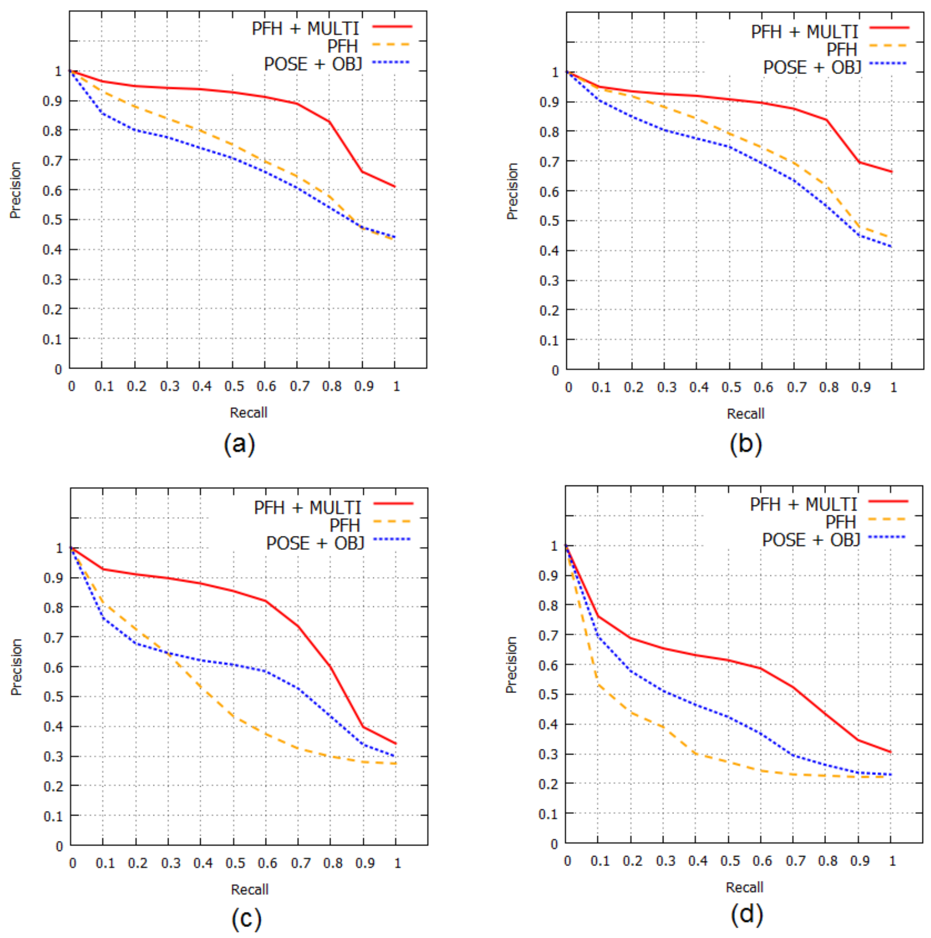


Figure 5.15: Precision-Recall curves for (a) class 1: "pick up" motion, (b) class 2: "put down" motion, (c) class 3: "carry and move" motion and (d) class 4: "step over" motion.

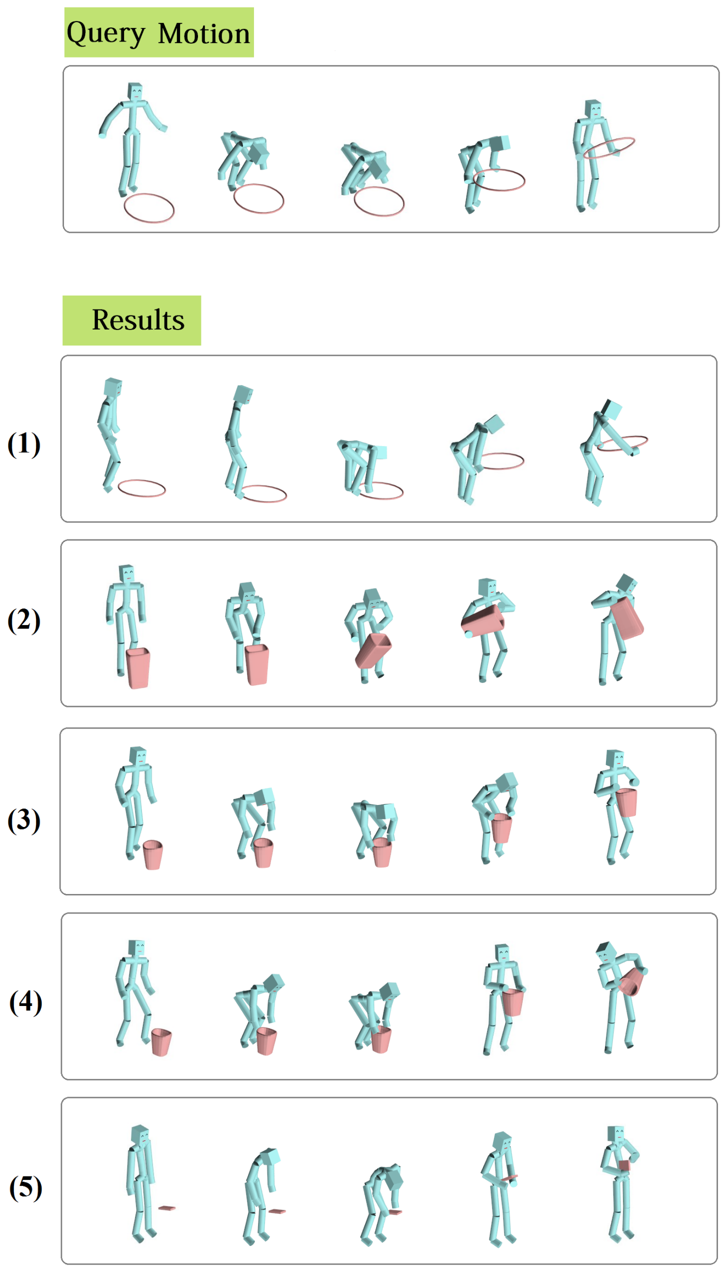


Figure 5.16: Retrieval results for the motion “pick up the hoop”. The results are ordered by similarity.

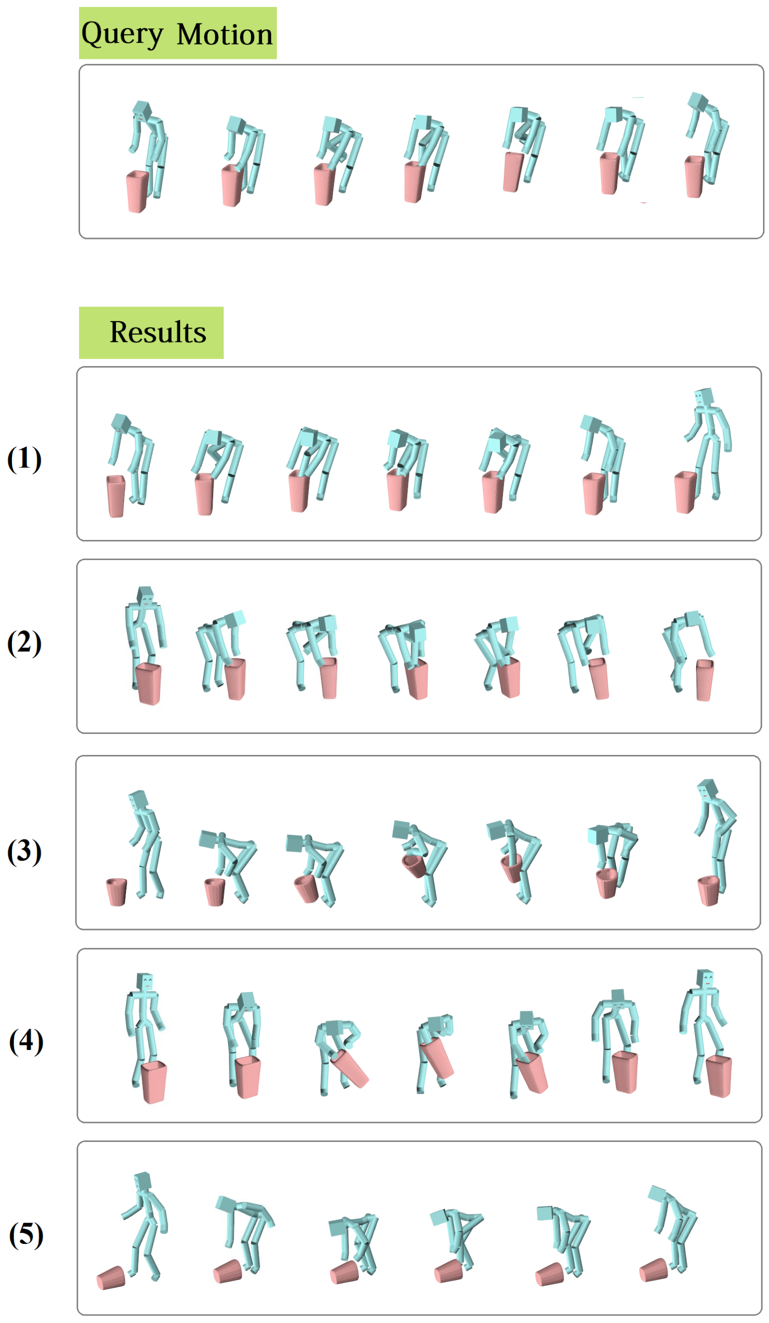


Figure 5.17: Retrieval results for the motion "step into the bin". The results are ordered by similarity.

Chapter 6

Scene Synthesis

In this chapter, we propose a novel approach to synthesize new scenes from example scenes by replacing objects with new ones. The novelty of our approach is that we encode the spatial relationship between the objects using the IBS, and by conducting a partial matching of the open space around the new objects with that of the original scene, examine if new objects can replace existing ones while preserving the original spatial relationships. In order to do this, we compute a translation, rotation and scale equivariant [1] feature which we call the "shape coverage feature (SCF)" (see Section 6.3 for details). This feature can effectively encode the relative location of a point in the open space with respect to objects around it. Using our approach, new scenes can be automatically synthesized from existing scenes and new objects. Our system is flexible in terms of input data as the object data can be in an arbitrary format, for example a point cloud, triangle mesh or polygon soup, and does not require any label information. As a result, the users can make use of arbitrary data from different sources for synthesizing new scenes.

6.1 Motivation

There is an increase in the amount of research which analyses the structure of man-made objects or composition of scenes composed of multiple objects, and which makes use of the knowledge learnt to automatically synthesize new contents. The objective of such work is to improve the efficiency of the design process.

Existing methods of automatic scene synthesis only use simple features such as

[1] If the input is transformed...then the inferences made by the system should co-transform in a stable and predictable way; this is termed *equivariance*. [Kivinen and Williams, 2011]

contacts or relative vectors to describe the spatial relationship between objects. Complex relationships, including hooking, enclosing, surrounding, and tucked in, cannot be described by such simple descriptors. So existing methods make many assumptions about the objects in the database, for example that they are all labelled and in the same scale. As a result, it is not possible to use arbitrary object data, such as those that can be downloaded from the web, or those which might be captured using the Kinect.

Here we use an example in Figure 6.1 to illustrate the synthesis problem. The input scene (c) is one which contains a chair tucked under a table, and a new table (d) and a new chair (e). The goal is to synthesize a new scene with the new chair tucked under the new table (as shown in (f)).

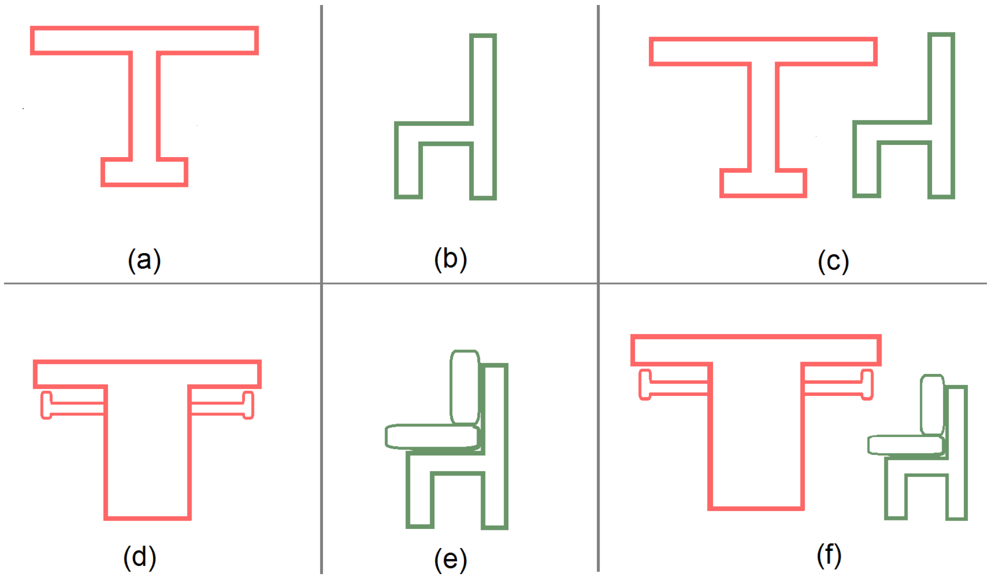


Figure 6.1: Synthesis example: (a) Object "table" in the input scene (the old table), (b) object "chair" in the input scene (the old chair), (c) input scene, (d) the novel table, (e) the novel chair, (f) the goal of the synthesis.

A naive solution for this problem would be to conduct a matching of the geometry between the single objects, and then align the objects based on the matching result. In the case of the example of the chair tucked under the desk, the new table and chair can be respectively aligned to the table and chair in the original scene. However, although this method seems straightforward, it may fail because of following reasons.

Firstly, existing shape matching methods normally match objects based on their geometry rather than function. As a result, they will not work well when the geometry of the novel objects is very complex, such as those shown in Figure 6.2. The novel table in Figure 6.2 (a) is only half the size of the one in the given scene, and has some objects on it. The novel chair in Figure 6.2 (b) is actually a stool next to a shelf, but this group of furniture can be used as a chair. In such cases, the novel and the old objects have very different geometry, which existing shape matching methods cannot deal with, even though the novel objects can produce a scene with a similar function.

Secondly, even if the shape of the novel objects is simple and thus alignment easier, the synthesis result may still be invalid. For example, by directly matching the novel table and chair in Figure 6.1 (d)(e) to the table and chair in Figure 6.1 (c), we can get a result as shown in Figure 6.3 (b). Note that the space between the novel table and chair is much narrower and cannot keep the same function as in the original scene. For instance, while a virtual character can sit between the table and the chair in Figure 6.3(c), he may not be able to do so in the Figure 6.3 (b) because of the lack of space. A desired solution is shown in Figure 6.1 (f), in which both of the objects are scaled and the spatial relationship between them well-kept. For scenes in which multiple objects are in close proximity, the open space between the objects becomes an important feature. In the case of the example of the chair tucked under the table, the orange parts shown in Figure 6.4, which are composed of surfaces from different objects, mainly define the spatial relationship of "tucked under". Simply matching the geometry of single objects does not work well in such cases.

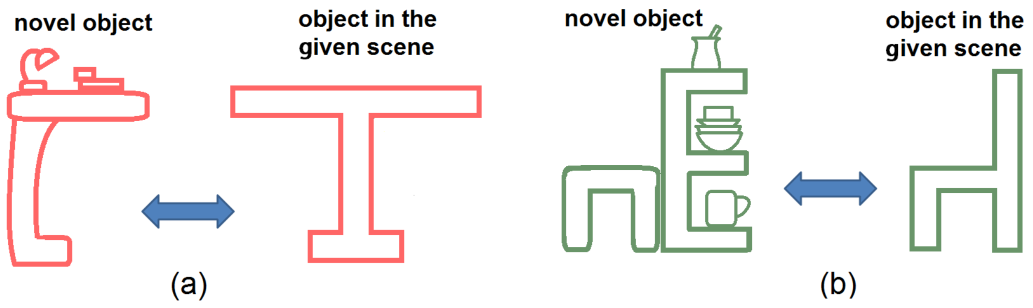


Figure 6.2: When the novel object is more complex, it is not clear how the matching between the novel and the old objects should be conducted.

In this research, we use the open space between the objects as a feature for examination if the novel objects can produce a scene with the desired relationship. More specifically, we use the IBS in the original scene as a soft constraint for synthesizing new scenes composed of novel objects. In order to do this, we first compute the IBS and the features of the open space around it in the original scene (Figure 6.5 step(1)) and then match them with the open space around the novel objects ((Figure 6.5 step(2)), which results in a novel scene with the same context ((Figure 6.5 step(3))).

Using our method, new scenes can be automatically synthesized from existing

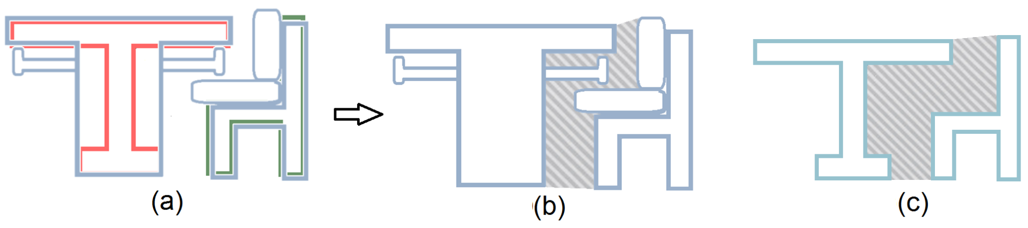


Figure 6.3: If the novel and the old objects are matched as in (a), the synthesis result (b), compared with the given scene (c), has very narrow space between the two objects.

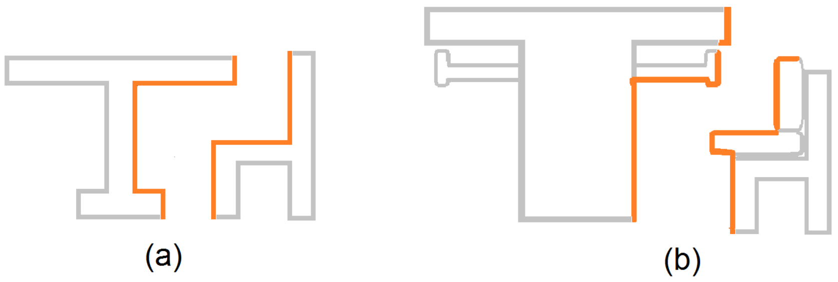


Figure 6.4: The orange lines show the most important part of object for defining the spatial relationship “tuck uncer”.

scene and novel objects. This is useful for applications including computer animation, computer games and virtual environments. As our approach is purely geometry based, it does not make many assumptions about the data. Therefore, data which are designed by the user using 3D modelling software, captured by scanning devices or simply downloaded from the internet can all be easily applied.

6.2 System Pipeline

We show the pipeline of our system in Figure 6.6, which is composed of two main stages. In the feature extraction stage, the open space is analysed, and the IBS and the shape coverage feature (SCF) are computed. In the matching stage, the open space around the IBS and the new objects are matched using geometric hashing.

In the feature extraction stage, we first compute the IBS of the example scene (Figure 6.6 step (1)). We then sample points on the IBS and compute the SCF and distance feature at these sample points (Figure 6.6 step (3)). The SCF is computed with respect to each object in the scene. For the given new objects, we compute the distance field and SCF in the open space around the objects (Figure 6.6 step (2)) by

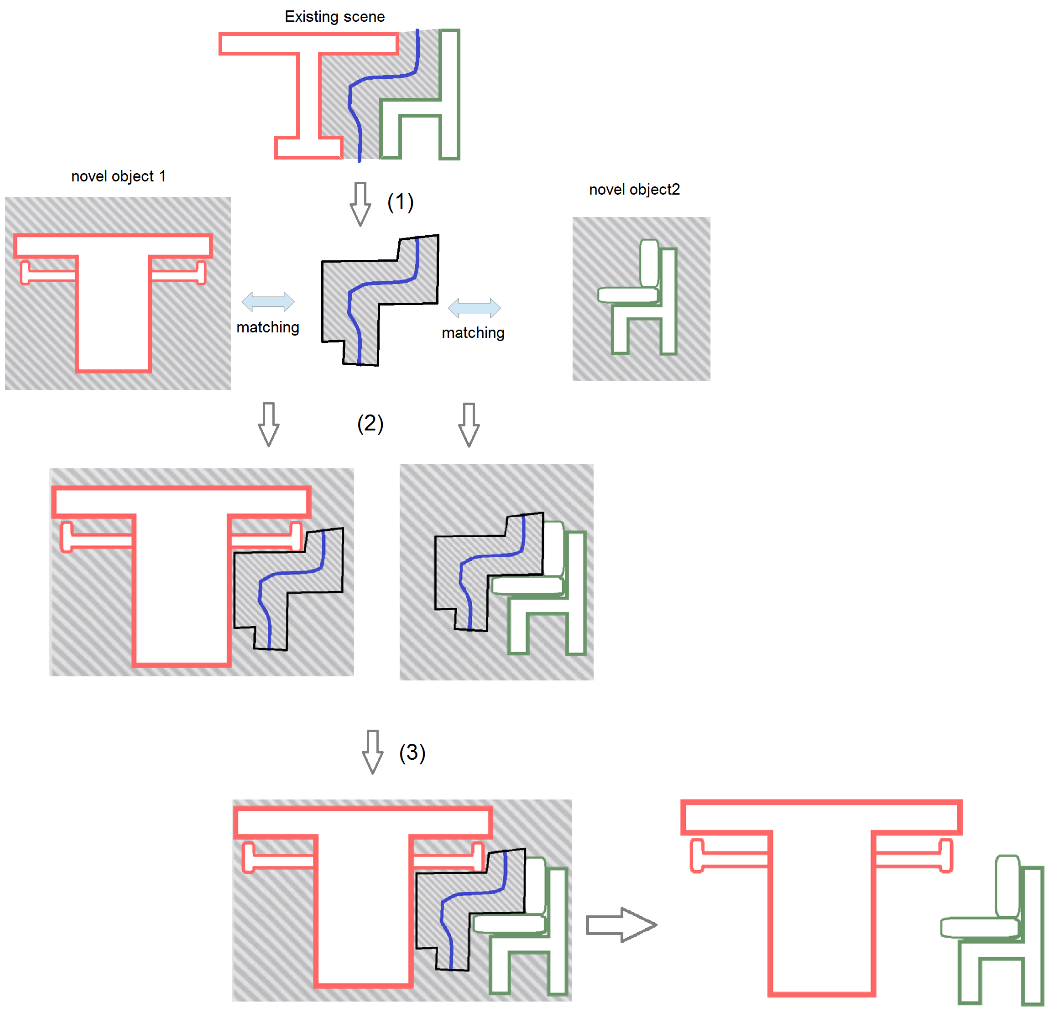


Figure 6.5: Overall idea of our solution. (1) Compute the IBS and features of the open space around it in the original scene. (2) Match the IBS' open space with the open space around the novel objects. (3) The resulting scene is produced by combining the matching results of the two novel objects.

uniform sampling.

In the matching stage, the samples that share similar SCF with those sampled on the IBS in the example scene are selected (step (4)). These selected samples are the candidate points for matching. Next, a partial matching is conducted between the IBS points and the candidate points (Figure 6.6 step (5)). This is followed by a refinement of the matching result by a local optimization (Figure 6.6 step (6)). Finally, the two objects are combined and a new scene is produced (Figure 6.6 step (7)).

In the rest of this chapter, each step (except step (1) which has been introduced in Section 3.2) will be described in detail, step (2) in Section 6.3, steps (3, 4, 5) in

Section 6.4.1, step (6) in Section 6.4.2 and step (7) in Section 6.4.3.

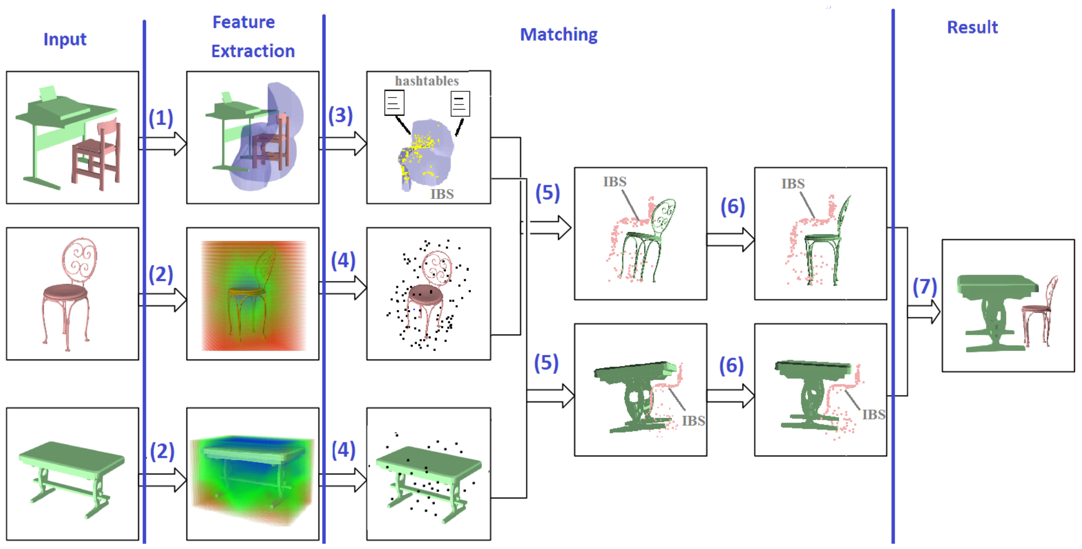


Figure 6.6: The outline of our method: (1) Computing the IBS and the features for the open space around the IBS, (2) computing the features for the open space points of the new objects, (3) sampling points on the IBS and computing two sets of features for these points, building two hash tables based on each set of features, (4) finding candidate points, (5) partially matching the open space points and the IBS points by using geometric hashing, (6) local refinement, and (7) the final combination of the two objects.

This chapter contributes:

- A method to synthesize novel scenes where multiple objects interact closely with one another,
- A scheme to index the open space around objects using a novel translation, rotation and scale equivariant feature which we name shape coverage feature (SCF), and
- A partial matching scheme for the open space around objects based on geometric hashing.

6.3 Feature of Open Space: Shape Coverage Feature

In this section, we propose a new feature that describes the characteristics of the open space with respect to the surrounding objects. This is used for conducting a partial matching of the open space in the example scene and that around the new objects.

Previous features that describe the relationship of points in the open space with respect to objects are not descriptive enough to be used for our purpose. For example, features such as the distance to the closest object or the locations where those values have discontinuity in the gradient space (medial axis vertices) are often used as a feature in the space. However, such features only provide the distance and direction information about the closest points (or set of points), but not any on the remaining geometries. Similarly, another feature often used in graphics is called the generalized winding numbers [Jacobson et al., 2013; Wang, 2012], which quantifies how much a point in open space is surrounded by surfaces. Again, the feature is significantly abstracted to a single value and is not a strong feature to evaluate the relation of a point with the surrounding objects.

In order to cope with these issues, we propose a new feature: the shape coverage feature (SCF), which quantifies the distance as well as the coverage information in all directions. It provides much richer information about the relation of a point and the surrounding objects. In order to reduce the memory usage per point, we define the feature in the frequency space using spherical harmonics. The details of the computation are as follows.

Given a point P in the open space, we cast rays in all directions (see Figure 6.9 (a)). More specifically, we uniformly sample $2B \times 2B$ points on a unit sphere whose centre is at P, and cast rays along the direction from the centre to each sample (see Figure 6.7 (a)). Each ray can be indexed by (i, j), where i is the index along the longitude and j the index along the latitude. We then check whether the ray collides with any object (Figure 6.9 (b)). If it does, then we record the distance d from P to the intersection point. d is set to infinity when the ray does not collide with any object.

We then define a function called the *volume diameter function* (VDF) as:

$$f_{VD}(i, j) = \{\frac{d^{min}}{d(i, j)} | 0 \leq i, j \leq 2B\} \tag{6.1}$$

where i, j are the ray indices and d^{min} is the minimum distance among all rays. Note that the range of the elements of f_{VD} is between 0 and 1. The infinite ray results in a value of 0, while the ray which has minimum distance results in a value of 1.

Finally, we compute the harmonics of this function (Figure 6.9 (d)) and use the absolute value of the harmonic coefficients to build the SCF vector. An example vector of the spherical harmonics (SH) coefficients is shown in Figure 6.8. Because of the symmetry in the rows of the values, we only use one side plus the middle column of the values (as highlighted in Figure 6.8). Therefore, the SCF is a vector that

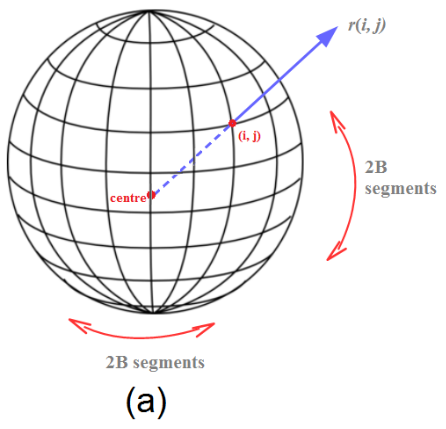
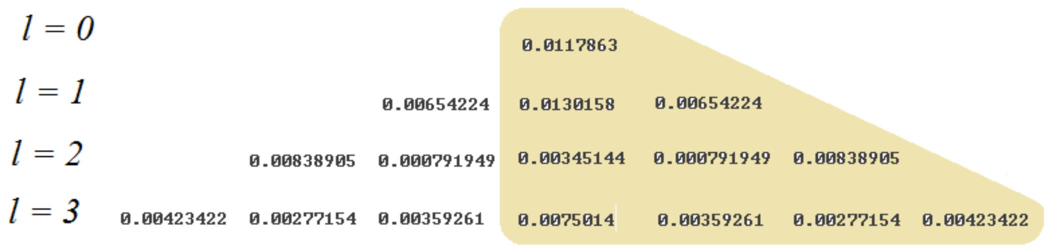


Figure 6.7: (a) To find the ray directions, we sample the unit sphere based on equally spaced angles. (b) In the SDF method [Shapira et al., 2008], the rays are cast within a cone.



$l = 0$				0.0117863			
$l = 1$			0.00654224	0.0130158	0.00654224		
$l = 2$		0.00838905	0.000791949	0.00345144	0.000791949	0.00838905	
$l = 3$	0.00423422	0.00277154	0.00359261	0.0075014	0.00359261	0.00277154	0.00423422

Figure 6.8: An example output of the absolute values of the spherical coefficients. The highlighted parts are the values we use to build the SCF vector.

preserves the coefficients in each frequency. The total dimension of this vector is $\sum_{l=0}^{L-1} n_l = (L+1)L/2$, where L is the maximum resolution of the SH considered. In practice, we set $L = 4$ (l is $0, 1, 2, 3$), resulting in a vector whose size is 10. We use SpharmonicKit [Kostelec, 2008] for computing the spherical harmonics coefficients in our experiments.

The ray-cast method was inspired by [Shapira et al., 2008], which measures the diameter of the object's volume by casting rays from each point on the mesh (Figure 6.7(b)). There are two main differences between our ray-casting strategy and their method. First, we cast the rays from open space points not from the points on the object mesh. Second, we cast rays to all directions instead of only within a cone.

The SCF is rotation and scale equivariant. It is scale equivariant because the absolute length of the ray is normalized by the minimum length of all rays. We do this because the information we are most interested in is the distribution of the rays and the

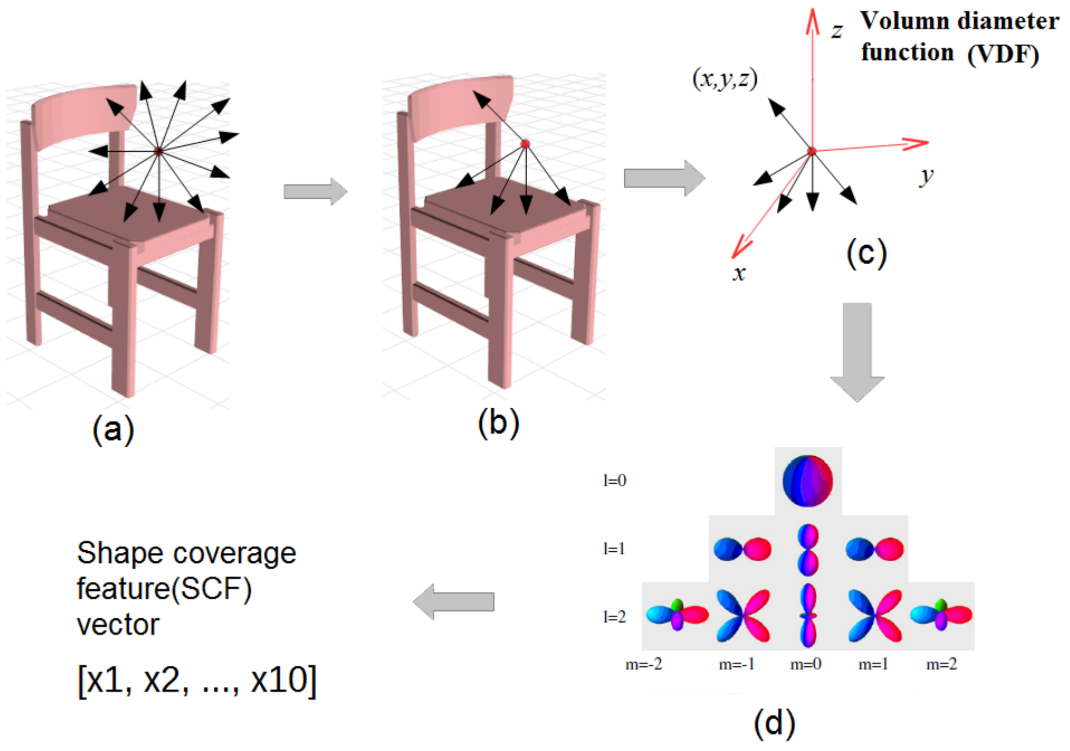


Figure 6.9: Computation of SCF for a point: (a) cast rays from this point; (b) find the rays intersect with the object; (c) compute the VDF; (d) compute the spherical harmonics coefficients for the VDF [Papadakis et al., 2007]; (e) build the SCF vector from the coefficients.

relative length of rays rather than their absolute length. By further applying the spherical harmonics transformation, the VDF is transformed to a rotation equivariant feature. In Figure 6.10, we visualize the first three dimensions of SCF vector of the open space around a chair model under different scale and orientation. It can be observed that the distribution of the values is equivariant under such transformations. This feature is also robust with respect to the local geometric details of the model. In Figure 6.11 we visualize the SCF values for two hooks. Although they have very different local shapes, the overall distribution of the SCF value is similar.

Using a larger bandwidth B to compute spherical harmonics will result in values with higher precision. In [Papadakis et al., 2007], they set B to 40 for computing 3D shape features. As we need to compute the SCF for a large number of points around the shape rather than only compute once per shape as in [Papadakis et al., 2007], we use a smaller B to speed up the process. To qualify the precision of this setting, we use the SCF computed by $B = 40$ as a benchmark feature and evaluate the difference between the benchmark feature and the feature computed by smaller

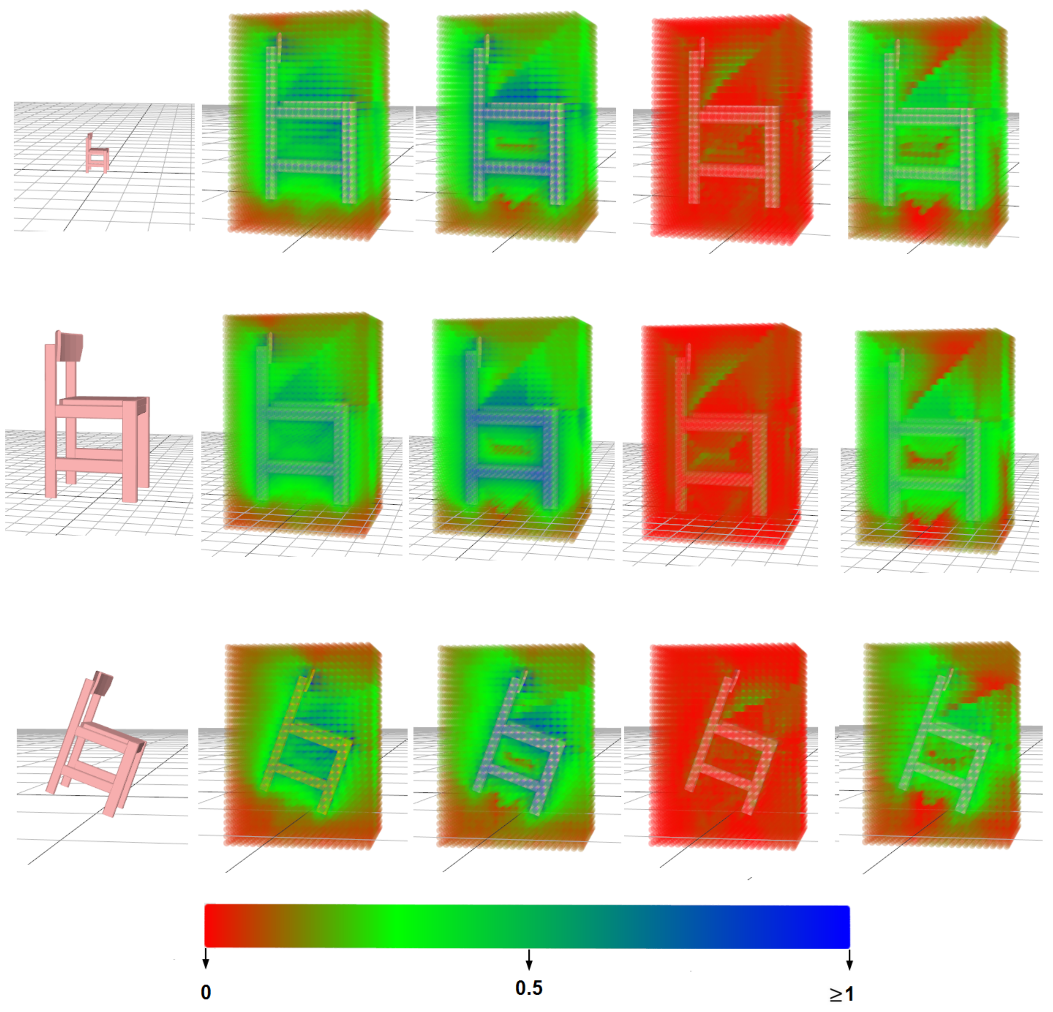


Figure 6.10: The SCF is scale and rotation equivariant. We show the SCF vector values of the points in a volume surrounding the chair model. From the left to the right, the first four items of the SCF vector are visualized. Blue indicates higher value, red lower.

B. For each dimension of the feature for the current B, we measure the average of the difference to the benchmark feature value over all open space points. Figure 6.12 shows a plot of the curves which reveals how the difference changes as B increases. The feature value range is around $[0, 1.6]$ for the chair model and $[0, 2.5]$ for the flower model. The curves show that the difference is lower than 0.03 (approximately 2%) for the chair model and 0.015(approximately 0.6%) for the flower model when $B = 10$.

The element operation for computing the SCF is ray-mesh intersection detection. We use an octree spatial index structure for the mesh model to accelerate this operation. In practice, we apply MiniLight library [Ainsworth, 2013] for this purpose. The computation time is shown in table 6.1. The total time depends on both the complexity

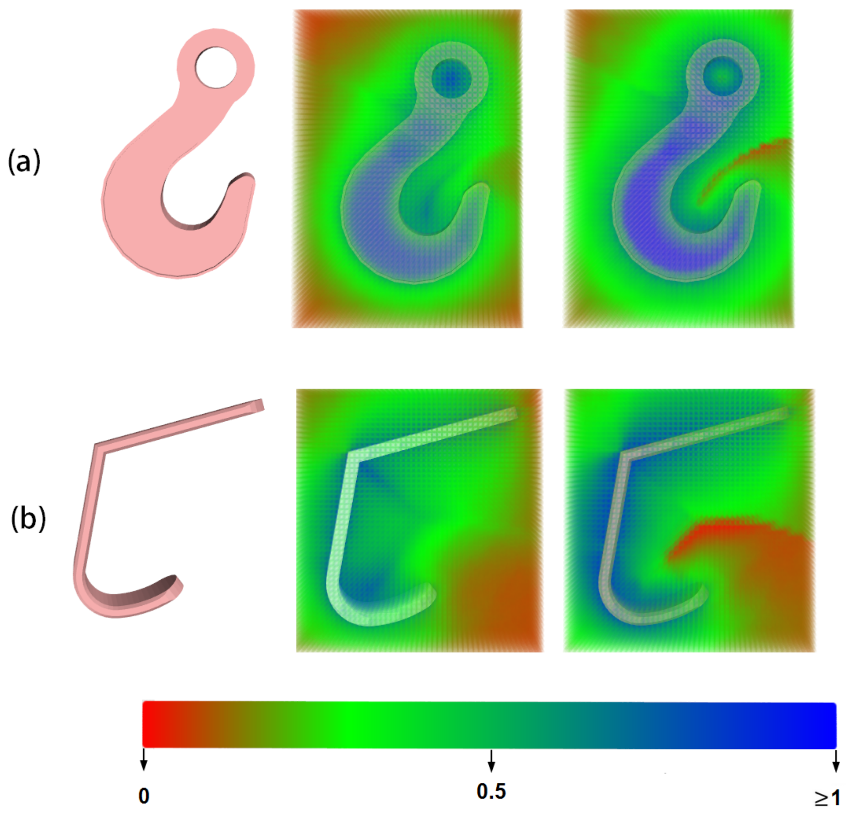


Figure 6.11: The distribution of the SCF around hooks of different geometry. The middle and right columns are the visualization of the first and second item of the SCF vector respectively. Blue indicates higher value, red lower.

of the object mesh and the number of open space points considered. It takes around 1ms for each SCF computation.

	No. Face	No. Open Space Points	Total Time (sec.)	Time per Point (sec.)
chair	688	4485	2.708	0.000603
table	136	21142	10.74	0.000508
flower	8452	9180	8.72	0.000950
vase	2592	14400	9.772	0.000679
hook	800	4200	4.485	0.001068

Table 6.1: Timing results for computing the SCF using $B = 10$ ($2B \times 2B = 400$ rays) for each open space point. Testing done on i7-2600 CPU, 3.4GHz with 16G RAM.

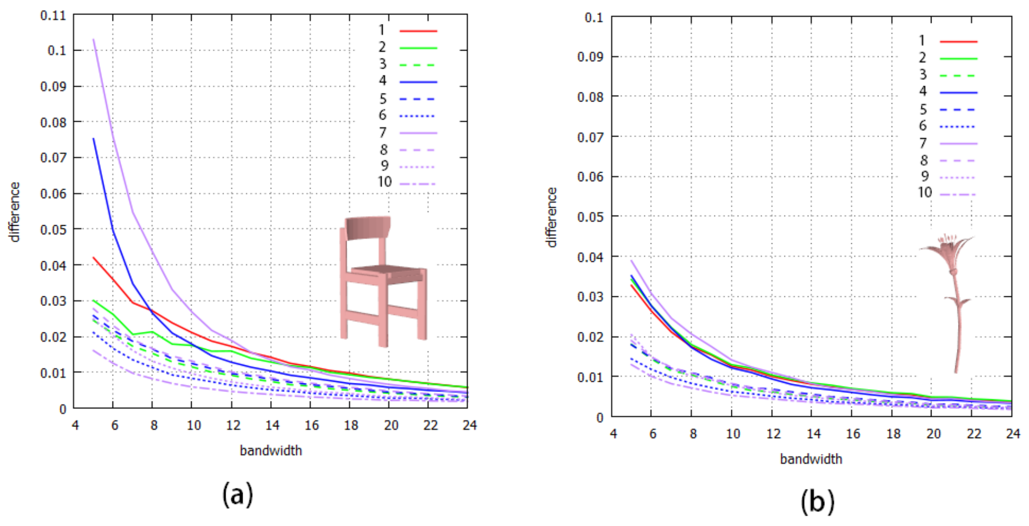


Figure 6.12: The difference to the benchmark feature is reduced when the bandwidth B is increased for (a) the chair model and (b) the flower model. The lines 1 to 10 represent the dimension 1 to 10 of the SCF vector.

6.4 Synthesis

Using the SCF computed for both the original scene and the novel objects, we can synthesize novel scenes by replacing the old objects with new objects using feature matching. We first find the coarse match using geometric hashing (see Section 6.4.1), and then conduct a finer registration using local optimization (see Section 6.4.2).

6.4.1 Indexing the Open Space by Geometric Hashing

In this section we conduct a partial matching between the open space around the IBS, where the interaction between the objects takes place, and the open space around the novel objects. A good match of the open space can be produced even when the fine geometry of the objects is different; and we can easily reject examples where there are obstacles that prevent us from producing similar scenes.

Although various methods can be considered for such kinds of volumetric partial matching, we use geometric hashing [Wolfson and Rigoutsos, 1997]. The advantage of geometric hashing is that it is an affine-invariant approach, which allows objects with different scales and orientation to be used as the input data. Also, we can use it in combination with the SCF to quickly reject objects that cannot produce a scene similar to the example.

In order to apply the geometric hashing method, we need to extract point clouds from the input data (the open space around IBS and the open space around the new objects) and prepare the features of these points. We sample points on the IBS and use these points as the input of geometric hashing. More specifically, we conduct a weighted sampling on the IBS by the method introduced in Section 3.4.2. Two hundred points (“IBS feature points”) are sampled for each IBS in practice. The SCF are computed with respect to each object in the example scene at the IBS feature points. Therefore, the IBS has two sets of SCF, one set per side (as shown in Figure 6.13 (step 1)). We build two hash tables for the two sets of features (see Figure 6.13 (step 2)). Next, for each new object, we sample points uniformly in its open space and compute the SCF for each sample point. The above computation can be done in advance during the feature extraction stage. For each new object, among the points uniformly sampled in the open space, we select n points for each IBS sample point that have the most similar SCF (we set $n = 10$ in our experiments). These points (“candidate points”) are combined and used as the feature points of the open space of the new object. If the similarity between the SCF of the candidate points and the IBS feature points are too small, it is likely that this object cannot produce a similar scene and is therefore is rejected at this stage.

Now the IBS feature points and the candidate points are matched to fit the new object into the scene using geometric hashing. More specifically, we first find a list of bases for the candidate points, then transform these points according to each basis. Then the transformed candidate points are mapped to the IBS hash table and our system returns the item from the table with the highest matching score. The transformation corresponding to this best match is computed and the object can be transformed into the coordinate system defined by the IBS. This process is done for all new objects (see Figure 6.6 (5)). This approach is similar to the method in [Gal and Cohen-Or, 2006], where geometric hashing is applied for partial matching 3D surface data.

We use Algorithm 2 to summarize the content of this section. The parts (a), (b) and (c) correspond to the step (3), (4) and (5) in Figure 6.6. The readers are referred to Appendix G for details of geometric hashing algorithm.

The fitting results by this approach are not perfectly accurate as we conduct sampling to match the IBS and the open space around the new objects. This issue is addressed by local optimization which is described in the next subsection.

```
Data: Existing scene, novel objects
Result: A set of transformation matrix for matching IBS with each novel
        object's open space

(a): Prepare IBS
Compute IBS for the existing scene
Sample the IBS to get a point set $S_{IBS}$
Compute SCF feature for each point in $S_{IBS}$
for each novel object $o_i$ do
    (b): Find candidate point set
    Sample points in $o_i$'s open sapece, get a point set $S_i$
    Compute SCF feature for each point in $S_i$
    Compute Candidate point set $S_i^{ca}$ which is a subset of $S_i$:
    for Each point $p_j$ in $S_{IBS}$ do
        find $S_{current}^{ca}$ which contains most similar points from $S_i$ with respect to
        $p_j$'s SCF feature
        $S_i^{ca} \leftarrow S_i^{ca} \cup S_{current}^{ca}$
    end
    (c): Find transform $T_i$
    Existing model $\leftarrow S_{IBS}$
    Input model $\leftarrow S_i^{ca}$
    $T_i$ = GeometricHashing(Existing model, Input model)
end
```

Algorithm 2: Indexing the open space by geometric hashing

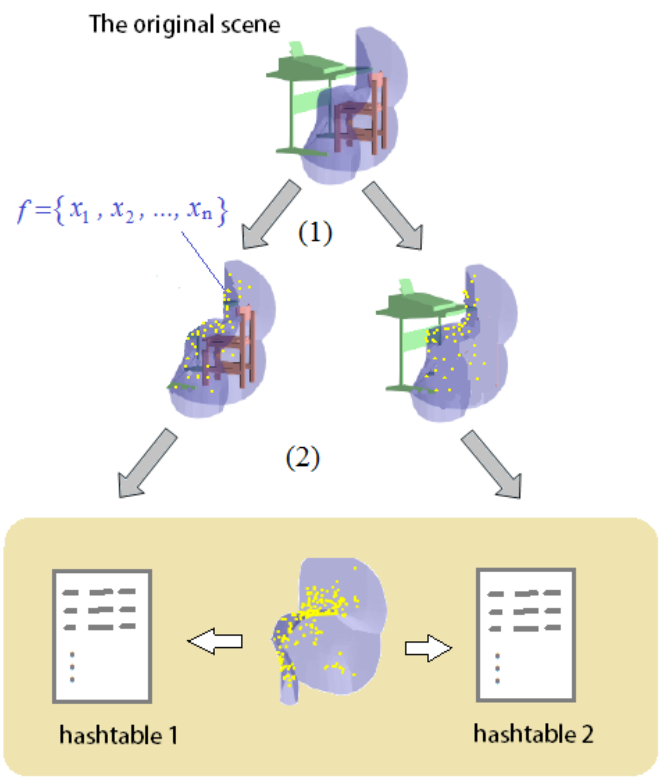


Figure 6.13: (1) Computing SCF for the IBS with respect to each object; (2) Computing hash table for each set of features.

6.4.2 Refinement by Local Optimization

Using the coarse matching results computed in Section 6.4.1 as an initial state, we conduct a local optimization to improve the matching quality. This is done by iteratively adjusting the location of the IBS with respect to the new object.

The matching quality is evaluated by how much the distance feature of IBS in the original scene is retained in the new scene. The distance feature, defined as $d^{original}$, is the distance between the IBS feature point and the closest point on the object in the original scene. For the new scene, a point closest to each IBS feature point is found on the new object. The closest distance is denoted here by $d^{current}$. To quickly find $d^{current}$, we pre-compute the distance field of the new object. Given the current location of a IBS feature point in the new object's coordinate system, the closest distance can be easily looked up in the distance field. Then, how much the distance feature is retained in the current configuration can be evaluated by the following function:

$$\mathbf{diff}(d^{current}, d^{original}) = \sum_{i=1}^{N} |d_i^{current} - d_i^{original}| \tag{6.2}$$

where N is the number of samples on IBS.

We now describe how we iteratively update the relative configuration of the IBS with respect to the new object by minimizing the difference of the distance feature. To iteratively minimize the $\mathbf{diff}(d^{current}, d^{original})$, we use a similar strategy as in [Zheng et al., 2009]. At every iteration, we find an "ideal" location of each IBS feature point by the following method. From the current location, we decide to move either in the gradient or negative gradient direction by comparing $d_i^{current}$ and $d_j^{original}$: if $d_i^{current} > d_j^{original}$, we move towards the object, otherwise we move away from it. This decision is made individually for all the IBS feature points. In the k_{th} step, the next goal of the IBS feature points can be computed as:

$$V^{k+1} = V^k + M \tag{6.3}$$

where V^k is a $3 \times N$ matrix that includes the location of all the IBS feature points in the k-th step, V^{k+1} a matrix of the goal positions in the next iteration and M a $3 \times N$ matrix that includes the direction vectors at each point. The columns of M are computed by $\mathbf{m}_i = a_i\, \mathbf{g}(v_i^k)$, where $\mathbf{g}(v)$ is the gradient vector of the distance field at the point v, and a_i is called a move parameter, whose sign indicates whether the move is in the gradient direction or in the negative gradient direction. The absolute value of a_i controls the step size.

The optimal rigid transformation to register V^k to V^{k+1} is computed by first computing a covariance matrix A of the two matrices:

$$A = (V^k - \bar{V}^k)^T (V^{k+1} - \bar{V}^{k+1}) \tag{6.4}$$

where $\bar{V}$ is a matrix whose rows consist of the mean values of V. Then we apply the SVD decomposition to A as $A = USV^T$. The rotation and translation matrix of the rigid transformation is obtained by

$$R = UV^T, \quad T = \bar{V}^k - \bar{V}^{k+1} R^T. \tag{6.5}$$

The rigid transformations are applied to V^k to get the updated location of the IBS feature points, which are fed into V^k in 6.3 for the next iteration. After every iteration, we compute $\mathbf{diff}(d^{current}, d^{original})$ and check if it is small enough. We stop the iteration when $\mathbf{diff}(d^{current}, d^{original})$ or the resulting update are smaller than thresholds. $\mathbf{diff}(d^{current}, d^{original})$ of the last iteration is used as a score of the resulting match.

The local optimization step is shown in Figure 6.6 (6). To show the effect more clearly, we give another example in Figure 6.14. In this example, the table is the new object. The chair is from the original scene and keeps its relative location with respect to the IBS. The location of IBS is updated iteratively with the chair moving towards the location that produces the same spatial relationship as in the original scene. The example shows that although the initial location of the chair is only roughly in front of the table, it fits better after a few iterations.

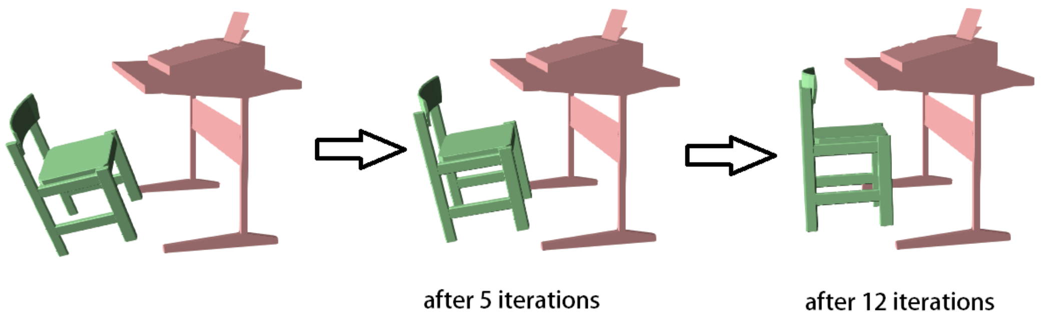


Figure 6.14: The effect of the refinement process.

6.4.3 Final Combination

New scenes are produced by fitting new objects to both sides of the IBS using the geometric hashing and local optimization described above. It is possible that one object is matched to the IBS in different ways. But, for simplicity, we currently only keep the best match per object. Given a set of new objects and an IBS extracted from an example scene, we examine all the combinations to produce new scenes. The last step is to check if the new scene contains any penetration between the two objects. If collision happens, the combination is discarded, otherwise we accept the result as a qualified new scene. In Figure 6.15 we show the combination process on a table-chair example.

6.5 Results and Discussion

In this section, we show examples where our method is applied to create new scenes.

Using three example scenes where chairs are tucked under desks and a set of chairs and desks/tables, scenes are automatically produced (see Figure 6.16). Another example is one object hung on another as shown in Figure 6.17. Using three example scenes where a bag is hung on a hook and a series of new bag/baskets and hook models, new scenes are automatically synthesized. These results are sorted in the order of the fitting

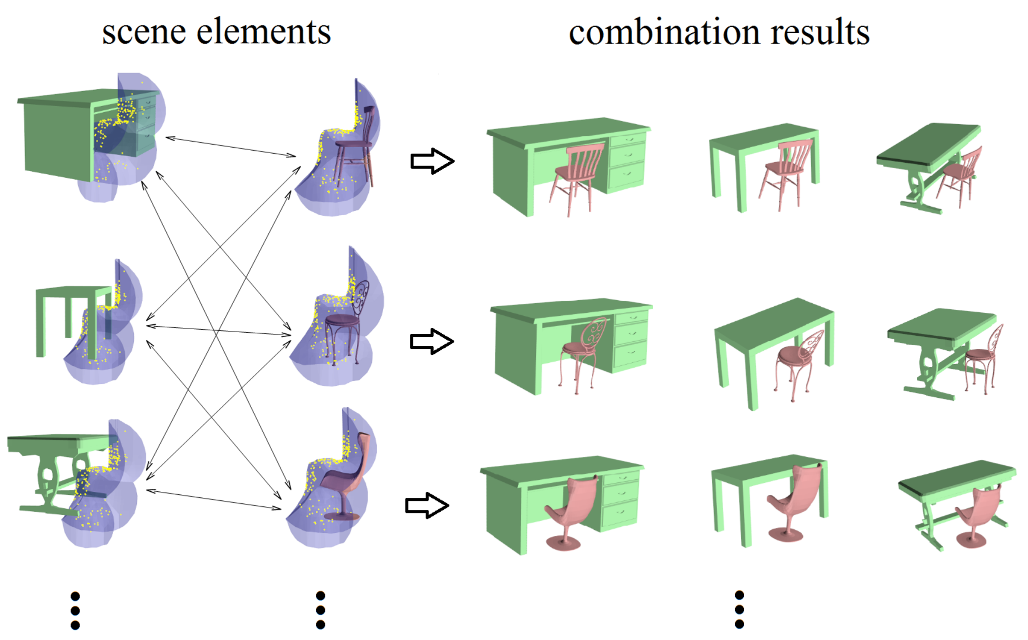


Figure 6.15: The scene elements and how they are combined to create new scenes.

score. The main cost of our method is in geometric hashing. With the table-chair example, the matching of each object to the IBS takes 661 seconds on average and with the hook-bag example it takes 226 seconds. We used computer with i7-2600 CPU, 3.4GHz and 16G RAM for this experiment.

Some failure cases are shown in Figure 6.18. Although the spatial relationship between the objects is similar to those shown in the example scene, the scenes look unnatural as the furniture/objects are tilted in a way that would not occur in real life. Normally people do not position a desk as in Figure 6.18 (a) or hook two bags together as in Figure 6.18 (c).

To cope with this problem, we add in directional constraints for the input objects by manually specifying their upright positions. When we match the new objects to the IBS, we measure how much the resulting configuration deviates from the upright position. If the angle is larger than a threshold, we discard the result. In this experiment, we set the threshold to 20 degrees. By using directional constraints, our system automatically discards all the unnatural results similar to those in Figure 6.18. With the hook-bag example, a large number of results are discarded due to their not fulfilling the direction constraints. This is because the bag-and-hook type of models have radially symmetric geometry along the handle. For example, the handle of a bag is usually a half circle, the different parts of which can all possibly match the IBS. Many hook-bag results are also discarded due to penetrations happening during the combination. This is because the IBS is very close to the objects in the given example. As a result, a penetration can easily happen, when, for example, a very narrow bag stripe is hung on a very wide hook (see Figure 6.18 (d)).

An alternative approach to synthesizing new scenes from example scenes is by applying 3D shape matching to the individual objects and replacing the existing objects in the scene with the new ones according to the match. This approach may work well when there is enough space between the objects or when the relationship between the objects is comparatively simple. However, this approach does not work well for complex spatial relationships, In Figure 6.19 we show that if we use a shape matching method to synthesize the hook-bag scene, we end up with results like (b), instead of (c).

In contrast, our method focuses mainly on spatial relationships. Our representation, the IBS, is adaptive to different types of interactions. It takes into account the fine details of the object geometry when the interaction involves very close contacts and only the abstract geometric shape when the objects are farther away from each other.

6.6 Summary

We have presented an algorithm for synthesizing new scenes composed of two objects from existing scene data and novel objects. Our method is designed to handle complex spatial relationship and is robust to models with very different geometry. The key advantage of our method is that it matches the IBS with the open space around the new objects, which relieves us from directly matching the object geometry.

This research is still in its early stage. In the future, we plan to test our algorithm on a larger dataset and develop methods to use it for multiple-object scene synthesis. We are also interested in extending our approach for dynamic scenes where the interacting parts change their relationship with one another over time, such as those in which animated characters are involved.

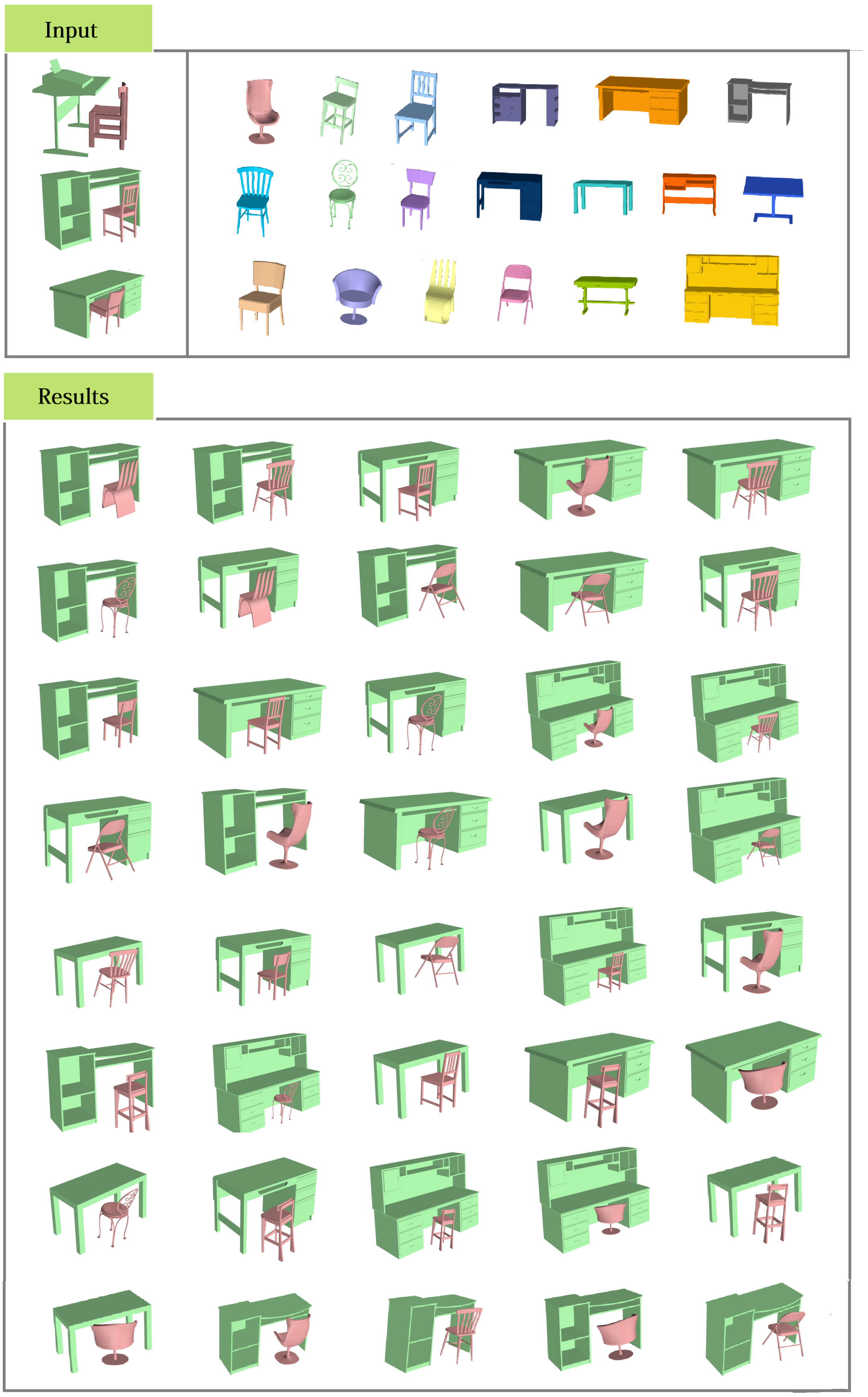


Figure 6.16: Using three example scenes where a chair is tucked under a desk (top, left) and a set of chairs and desks/tables (top, right), scenes are automatically synthesized (bottom).

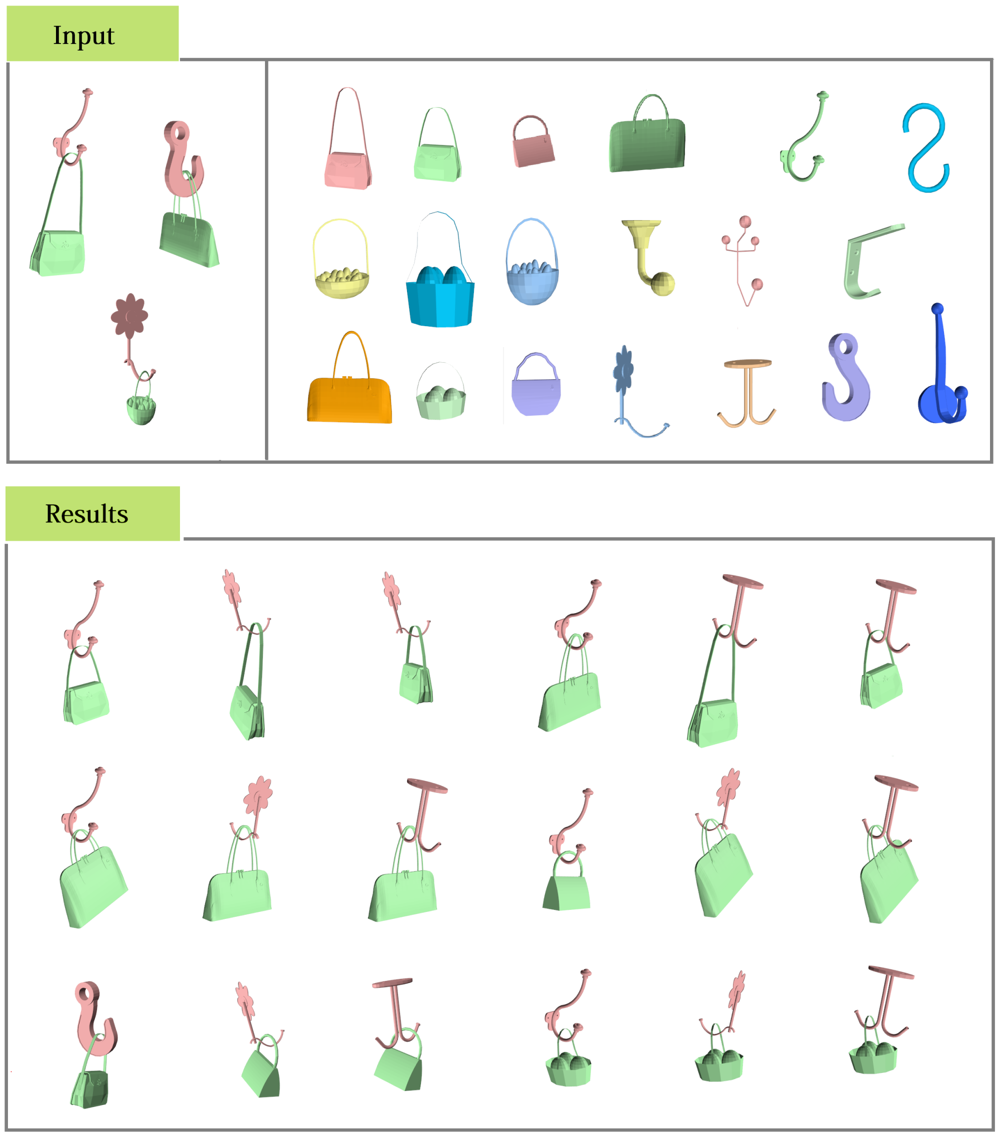


Figure 6.17: Another example of scene synthesis: using three example scenes of a bag hung on a hook (top, left) and a series of hooks and bags (top, right), scenes are automatically synthesized (bottom).

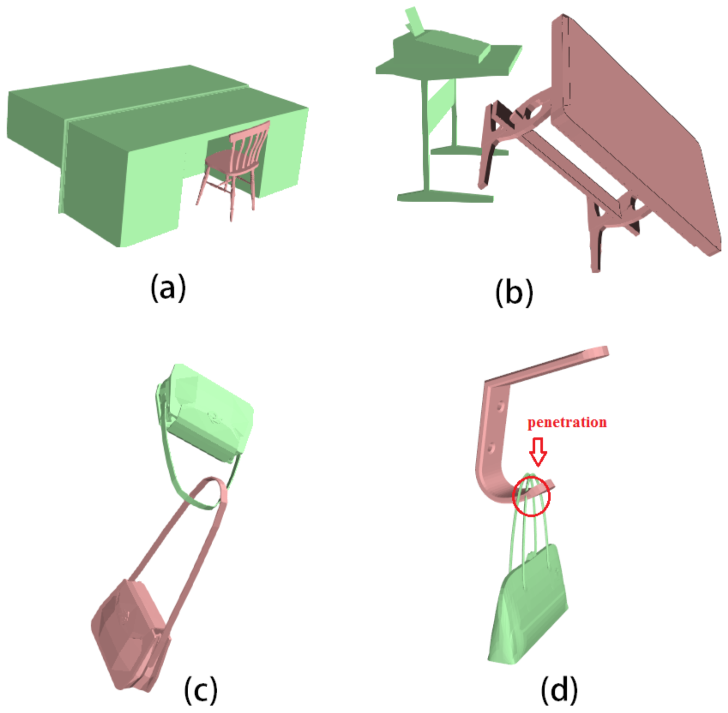


Figure 6.18: Some invalid results. (a) (b) (c) types of results are eliminated by the direction constraint. Results like (d) are eliminated by collision detection.

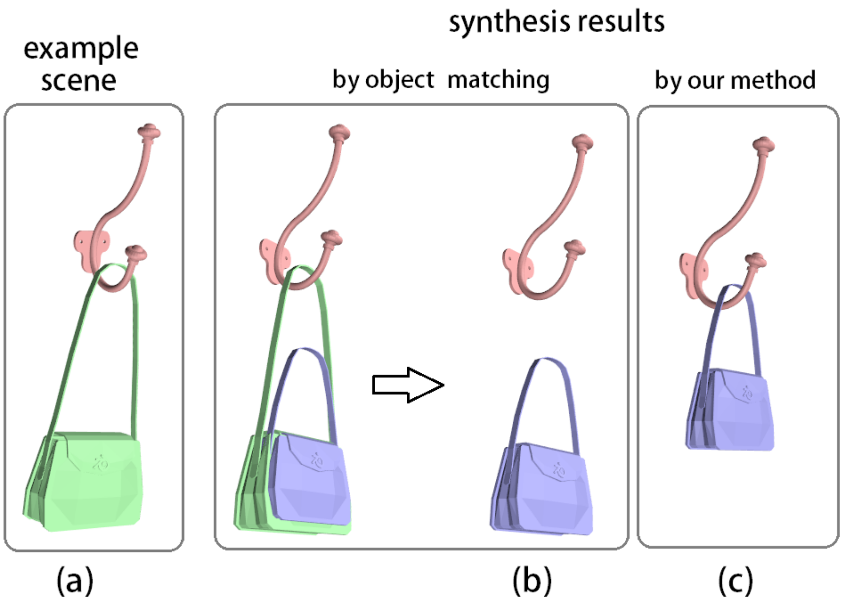


Figure 6.19: Comparing our method with a object-matching-based method.

Chapter 7

Conclusions

The study has set out to explore the representation of spatial relationship, which is very important for 3D scene analysis and synthesis. Most previous works use very simple representations about spatial relationship, which cannot describe or process complex interactions such as “hook on” or “encircle”. We propose a new representation called Interaction Bisector Surface(IBS) to solve this problem. We develop algorithms to use this representation in different applications and conduct extensive experiments to evaluate its effectiveness.

7.1 Findings and Contributions

The most important contribution of this thesis is that we proposed to represent the open space between objects or between objects and characters using the IBS and to extract and apply its features for scene analysis, indexing, classification, retrieval and synthesis. The IBS is expressive and robust and encodes continuous changes of interactions in motions. We apply this representation for solving the three following problems and provide a comprehensive analysis and evaluation of this new representation.

Firstly, we applied the IBS to static scene analysis. The classification experiment of pairwise interactions shows that the features of IBS can recognize both simple and complex spatial relationships and that they greatly outperform previous methods. This result proves that compared with previous features, the IBS contains much richer information about spatial relationships. We also propose a method to build a hierarchical structure for scenes by using the adjacency information defined by the IBS. Based on the scene structure, we developed a content-based scene retrieval system.

Secondly, we useed the IBS for the analysis of character-object interaction. Information on which part of the body interacts with the object is important for describing the context of such interaction. Therefore, using only the topological and geometric features of the IBS is not enough for the analysis of such interactions. In order to cope with this problem, we developed a multiresolution feature based on the correspondence between the IBS and the interaction parts. The experiment shows that this new feature can describe the gradual change of the spatial relationship during the interaction and that it outperforms traditional techniques of human motion recognition. This experiment shows that the IBS can be easily applied even to articulated objects such as a character body and that it is an effective tool for analyzing dynamic scenes.

Lastly, we applied the IBS to an even more challenging problem, which is how to automatically synthesize scene data. We create new scenes by matching the IBS computed from existing scenes and the open space around new objects. The experiments show that our algorithm can create plausible results purely from the geometry data without using any manually tagged labels. Another important contribution of this research is that we provided a new perspective of 3D data analysis: indexing interactions using the open space. We believe that besides the shape coverage feature and the partial matching algorithms we proposed in this thesis, more research can be done on the usage of open space.

From these three applications, one can easily see that the IBS is a useful representation of spatial relationships. It contains rich information, is robust to different types of data, and includes the key features for effective classification and scene synthesis. It can also effectively segment movements along the timeline based on the change of the spatial relationships with the objects in the scene.

Through our work, one can learn the way IBS describes the spatial relationship, the procedures of using the basic features of IBS, and the basic techniques of applying the IBS for scene processing problems. More importantly, the IBS can be a very basic tool for any 3D data analysis where multiple objects interact with one another.

7.2 Limitations and Future Plan

In the future, we plan to look further into applying the IBS for scene analysis and synthesis. We currently conduct scene analysis and synthesis purely based on the geometric data. An issue to resolve will be how to integrate such an approach with existing methods that make use of labels and human knowledge about object allocation. We believe incorporating our spatial-relationship-based features would benefit the existing scene retrieval and synthesis systems.

Another direction of our future work is to analyze the interaction motion captured by 3D scanners. Although RGBD devices such as Kinect are getting cheaper and their precision is improving, making it easier to obtain data that could be used as the input for our system, such data could be incomplete due to occlusions. Currently our system assumes complete data where the interaction parts in the scene are well segmented and does not seriously penetrate with each other. How to make our system more robust against the incomplete data or data with penetrations is a problem that needs to be solved. One possible solution is to reconstruct the objects from the scanned point cloud and remove the artifacts from the motion data by making use of existing scene data. We can match the point cloud with the objects in the database based not only on the geometry of the point cloud but also the IBS. This can extend to the human motion as well. The incomplete human movements captured by the RGBD camera may be reconstructed by referring to the complete motion data in the database using the IBS.

Lastly, we also plan to apply our synthesis algorithm to the creation of scenes with multiple objects. Currently we can only produce scenes composed of two objects. To generalize our algorithm beyond pairwise interactions, one possibility is to reduce the large scene to a meaningful subset of pairwise relationships and to use the pairwise objects to assemble large scenes. Another possible solution is to match multiple objects altogether simultaneously to the IBS.

7.3 Summary

In summary, this thesis raises a question of whether there is an effective representation about the spatial relationship between scene elements which interact with one another. The extensive experiments have given an positive answer. We have shown that our new representation can be used for static scene analysis, character-object interaction analysis, and scene synthesis. We believe that our techniques can greatly augment existing 3D scene analysis and synthesis systems. However, it must be noted that, at the current stage, the incompleteness of data could influence the precision of the IBS, and that the partial matching process of the synthesis is expensive, especially for complex 3D models. However, with the fast development of motion capture devices and high speed computing, there is a hope that the use of the IBS for large-scale 3D data may be further deepened and expanded.

Appendix A

Object-Pair Database

Here we show more examples in the object-pair database which are used for the classification experiment in Section 4.1.

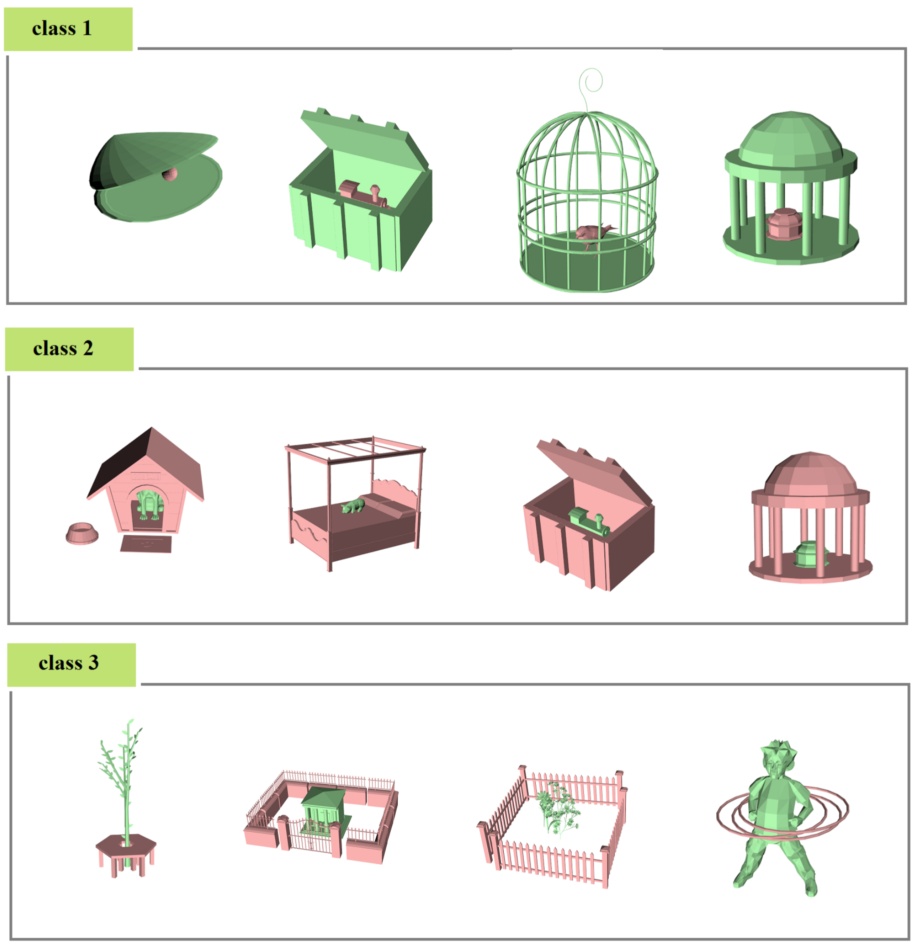


Figure A.1: More examples from the 16-classes database.

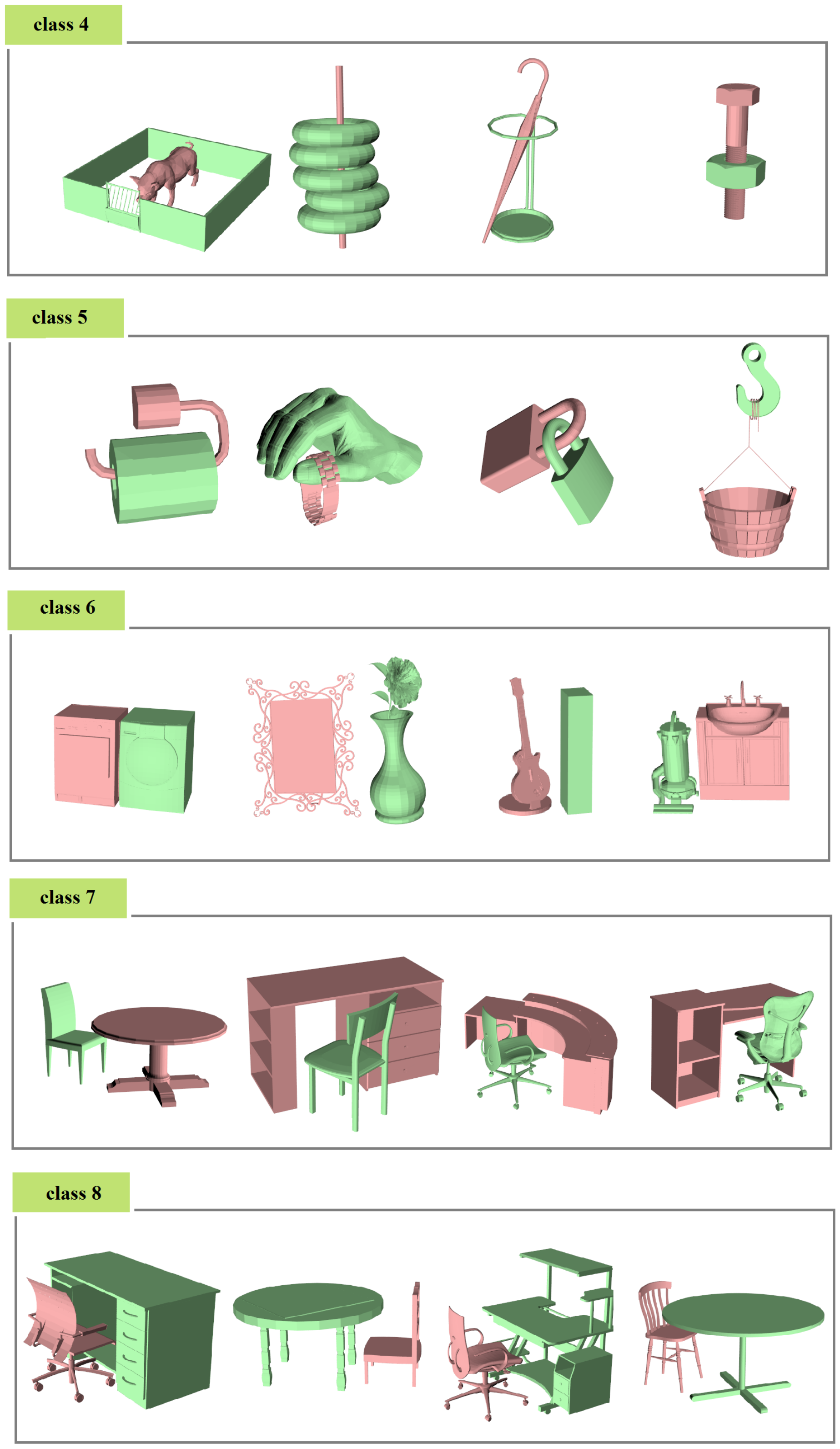


Figure A.2: More examples from the 16-classes database.

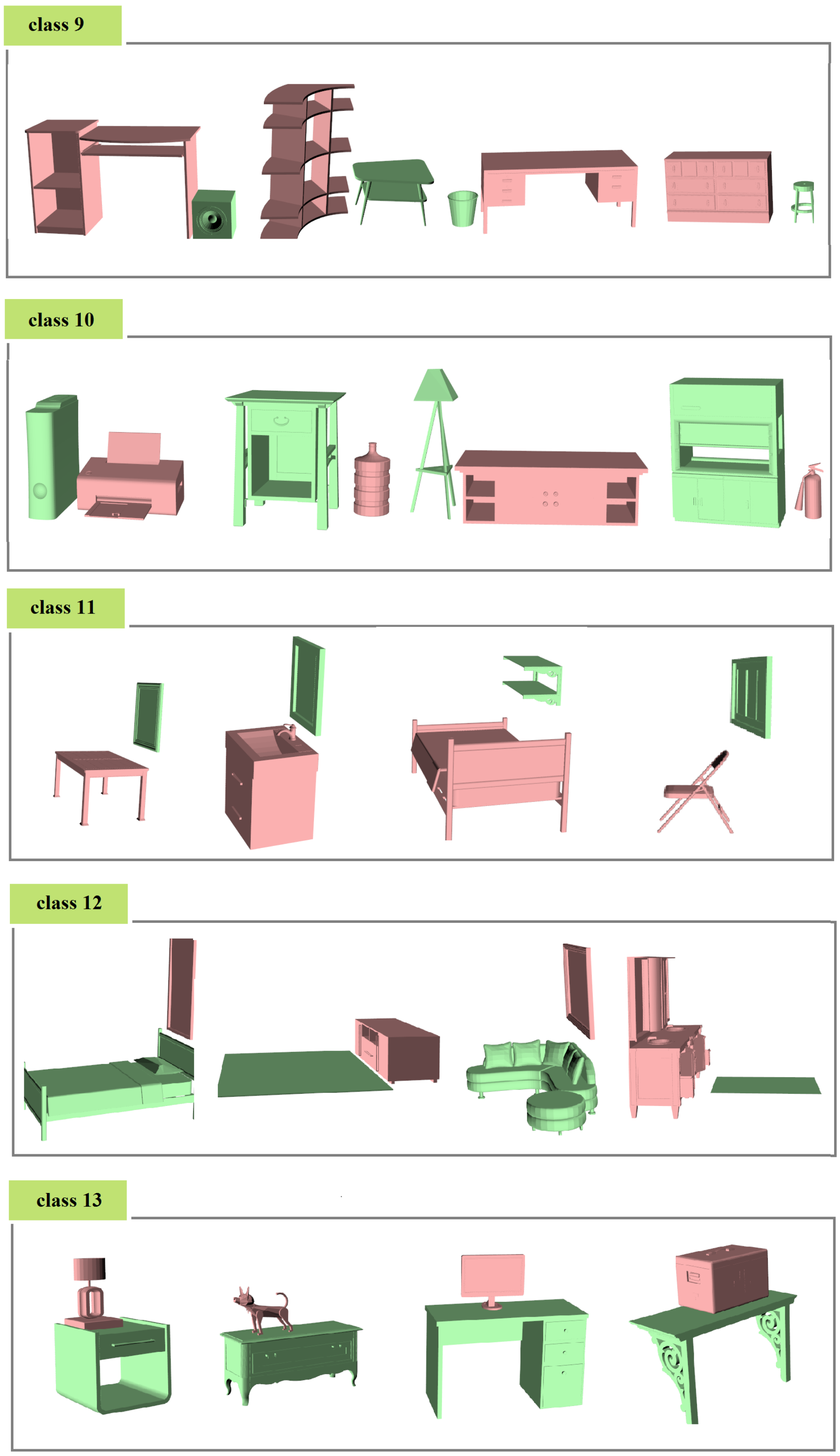


Figure A.3: More examples from the 16-classes database.

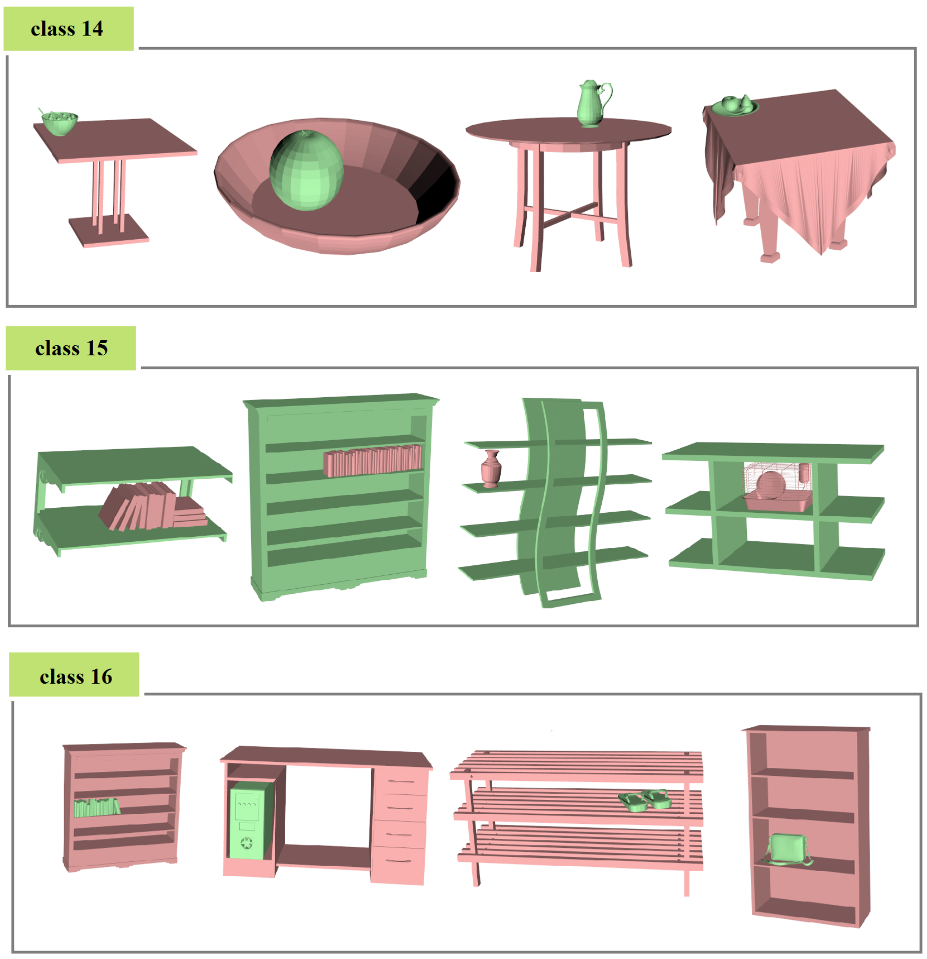


Figure A.4: More examples from the 16-classes database.

Appendix B

Scene Structure Results

Here we show more examples of scene structures produced by our method presented in Section 4.2. This section is a supplementary to Section 4.2.3.

As the number of objects in some scenes is quite large, it is difficult to visualize the structure by a graph. Here we indicate the object groups in each level of the scene structure using different colours.

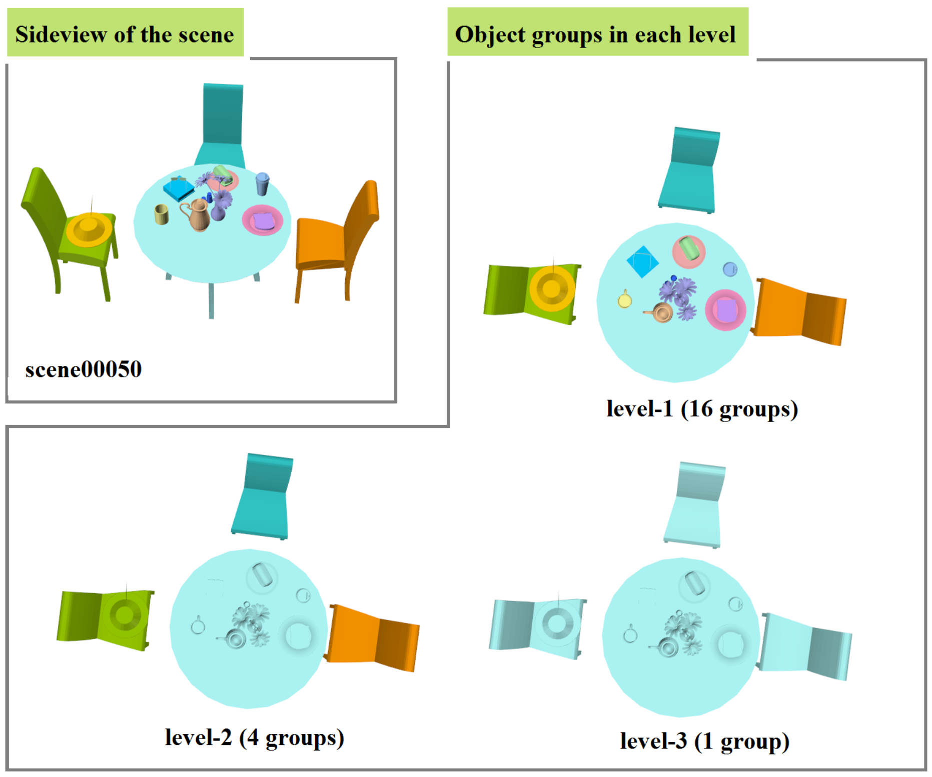


Figure B.1: Example 1 of 9.

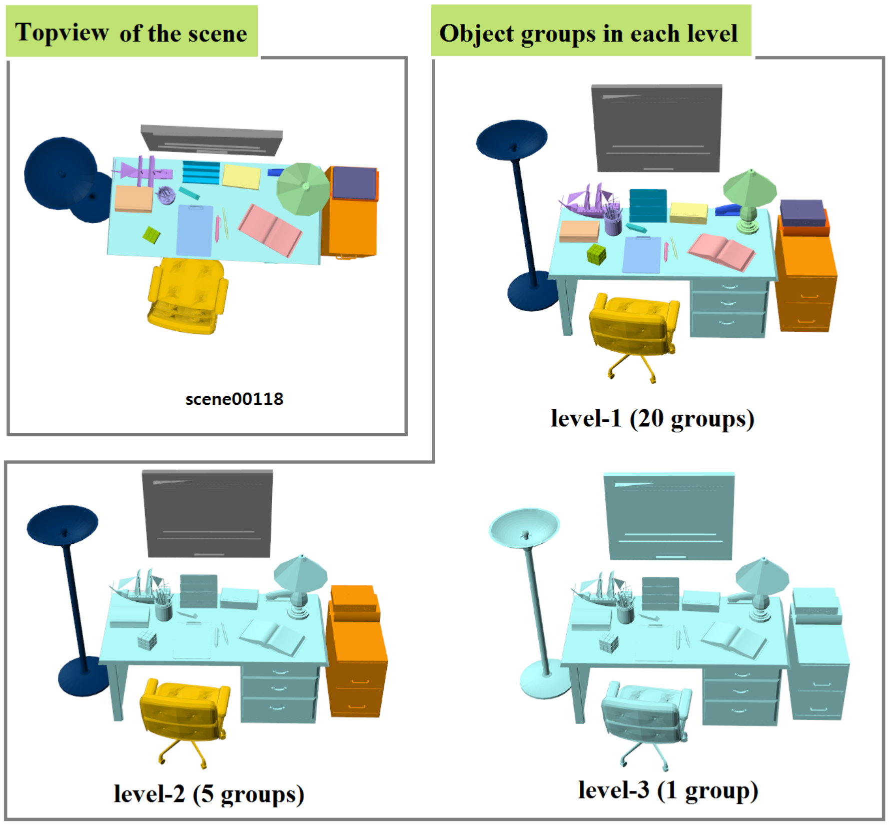


Figure B.2: Example 2 of 9.

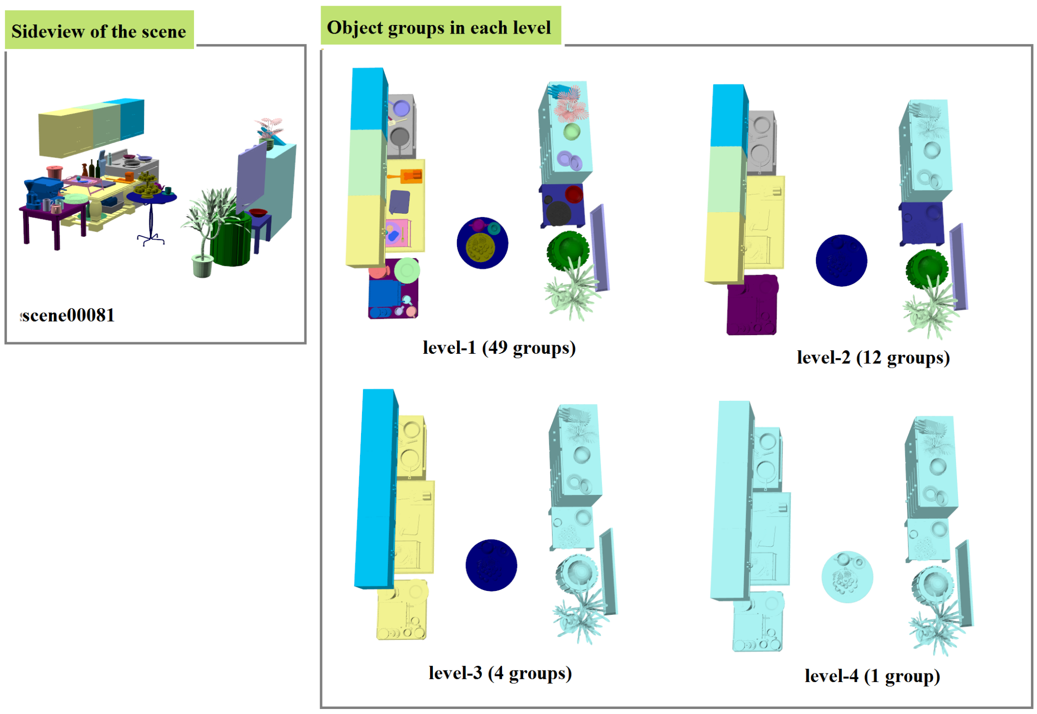


Figure B.3: Example 3 of 9.

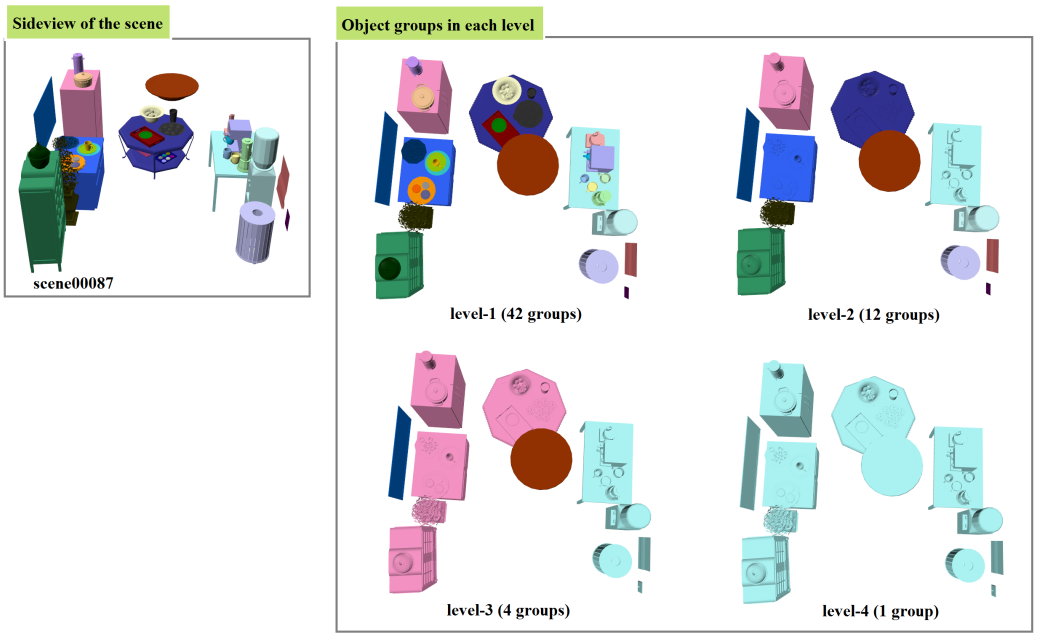


Figure B.4: Example 4 of 9.

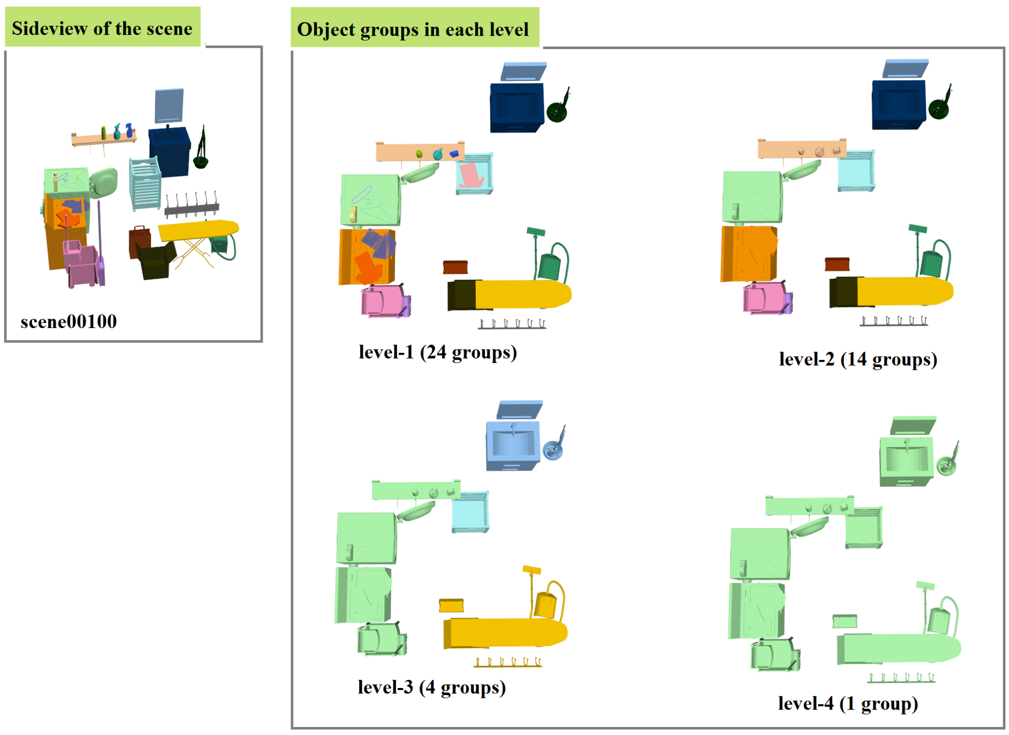


Figure B.5: Example 5 of 9.

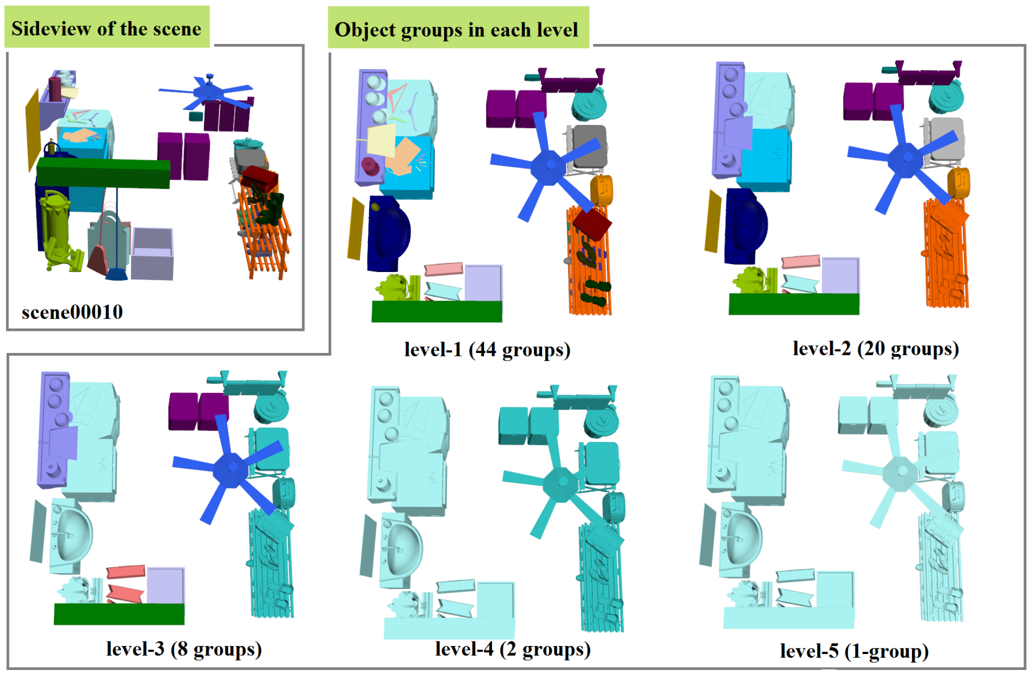


Figure B.6: Example 6 of 9.

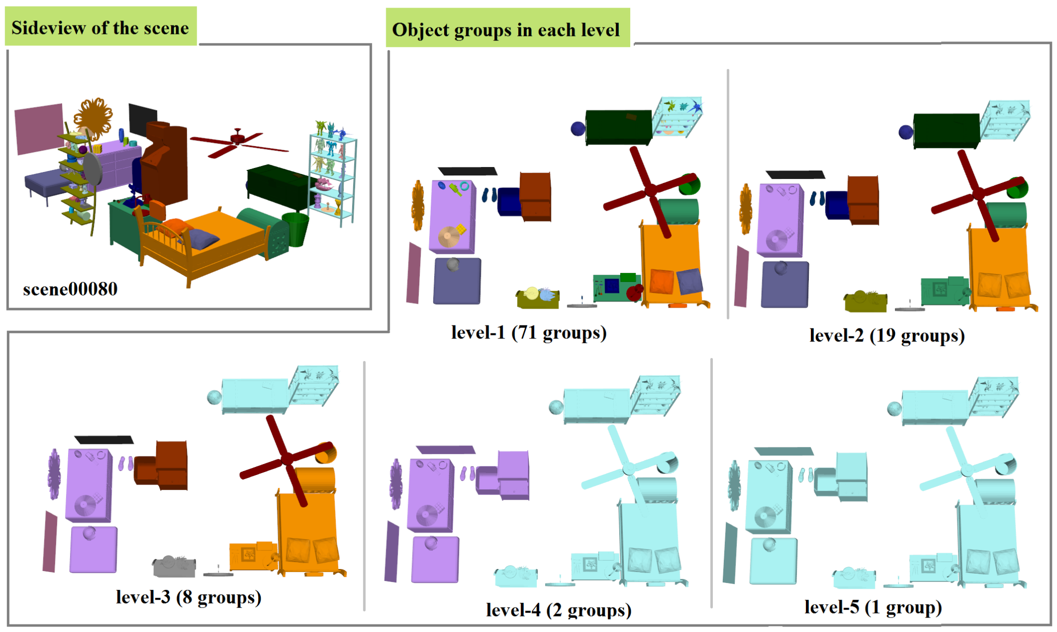


Figure B.7: Example 7 of 9.

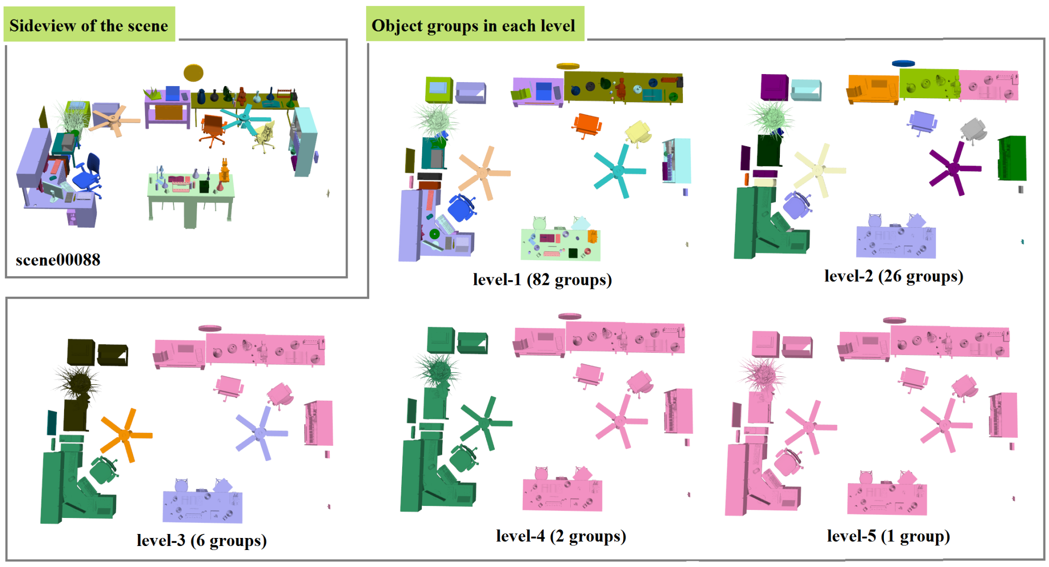


Figure B.8: Example 8 of 9.

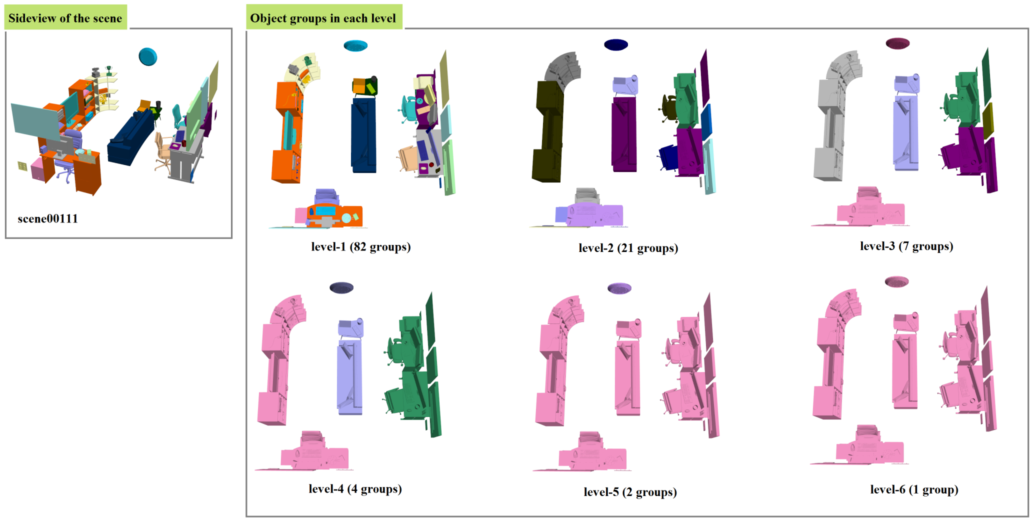


Figure B.9: Example 9 of 9.

Appendix C

Stability

Here we present the definition of γ introduced by Goodman and Kruskal [1954]. This value was computed for evaluating the scene structure stability in Section 4.2.3.

Consider two hierarchical structures for the same data, h_1 and h_2, and two pairs of elements of this data $p_i = (x_{i1}, x_{i2})$ and $p_j = (x_{j1}, x_{j2})$. The rank $r(h, p)$ is defined as the level at which the two elements of pair p first appear in the same cluster in hierarchy h. Over all the pairs of elements in the data, set

$$\begin{aligned}
\Pi_s =& P_r\{(r(h_1,p_i) < r(h_1,p_j) \wedge r(h_2,p_i) < r(h_2,p_j)) \vee \\
& (r(h_1,p_i) > r(h_1,p_j) \wedge r(h_2,p_i) > r(h_2,p_j))\} \\
\Pi_d =& P_r\{(r(h_1,p_i) < r(h_1,p_j) \wedge r(h_2,p_i) > r(h_2,p_j)) \vee \\
& (r(h_1,p_i) > r(h_1,p_j) \wedge r(h_2,p_i) < r(h_2,p_j))\} \\
\Pi_t =& P_r\{(r(h_1,p_i) = r(h_1,p_j)) \vee (r(h_1,p_i) = r(h_1,p_j))\}
\end{aligned} \tag{C.1}$$

Then

$$\gamma = \frac{\Pi_s - \Pi_d}{1 - \Pi_t} \tag{C.2}$$

γ measures the difference between the probabilities of "right order" and "wrong order". In other words γ shows how much more probable it is to get the same rather than different orders in two hierarchies. It ranges from -1 for inconsistency to 1 for consistency.

Appendix D

About The User Study

Here we provide more details of the user study in Section 4.3.4 which was used for evaluating the results of the relationship-based retrieval experiment (Section 4.3).

What are the four queries used in the user study?

The four queries are the following objects from the Stanford Scene database:

- Object 8 of scene00050 (the plate)
- Object 37 of scene00087 (the water dispenser)
- Object 14 of scene00118 (the chair)
- Object 14 of scene00109 (the desk)

The queries and scenes are shown in Figure D.1. The query objects are highlighted with red bounding boxes and arrows.

How were these queries selected?

As we can only ask users to label a limited number of retrieval results, we need to find some representative queries that fulfil the three following criteria:

- The query object's relationship with its environment should be common in the database. We need to make sure that the number of relative results is not too small. Otherwise the evaluation results may be biased.
- The four queries should be as different as possible to cover a wide range of spatial relationships in the scene data.
- As we also want to evaluate the retrieval results for different levels of neighbourhood, the query object should have more than one layer of neighbours.

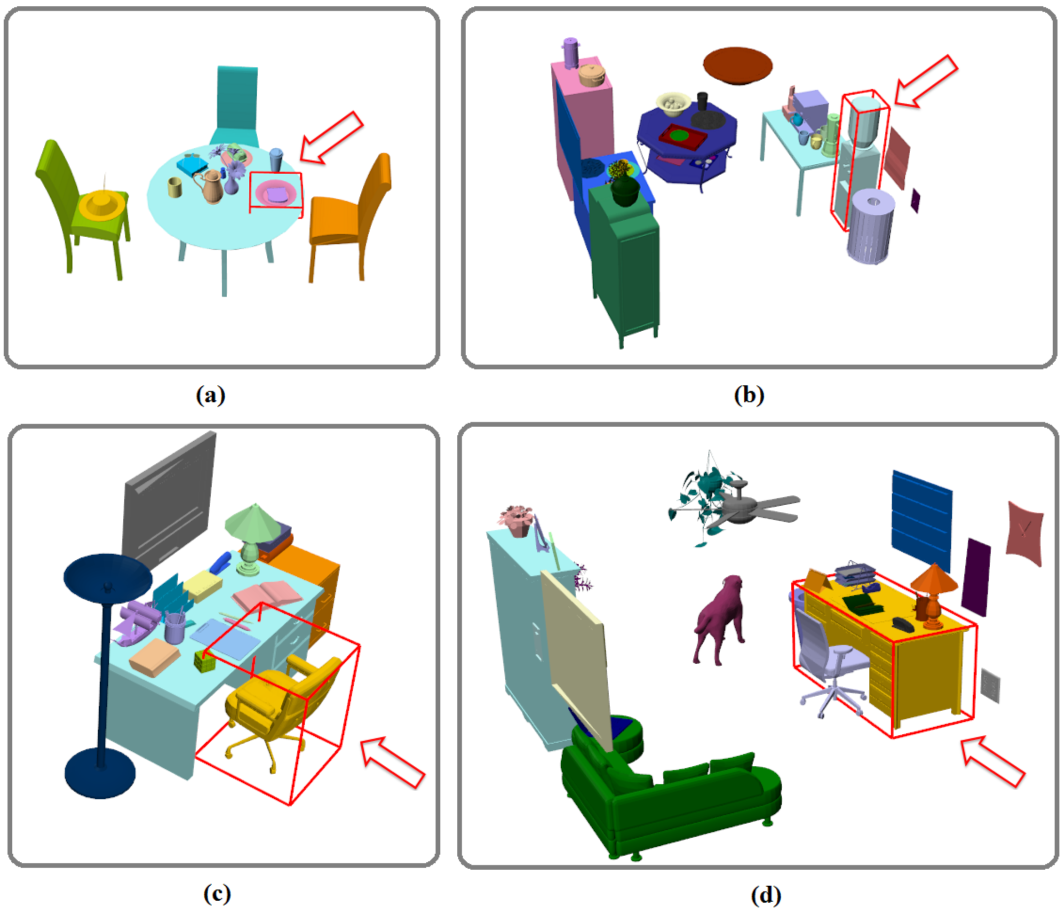


Figure D.1: Four queries used in the user study: (a) Object 8 of scene00050 (the plate). (b) Object 37 of scene00087 (the water dispenser). (c) Object 14 of scene00118 (the chair). (d) Object 14 of scene00109 (the desk)

The number of query set which fulfils these criteria is still large. We then randomly selected a set of query which turned out to be the four objects we use here. These four objects are in different size, have different relationship with its neighbours, different functions, and, according to the tree structure of the scene, different number of levels of neighbourhood.

The Instruction for the Users

The interface for users to label the retrieval results is shown in Figure D.2.

We gave the users the following instructions:

The user needs to label 500 scenes for each query. It takes about 15 minutes to finish one query.

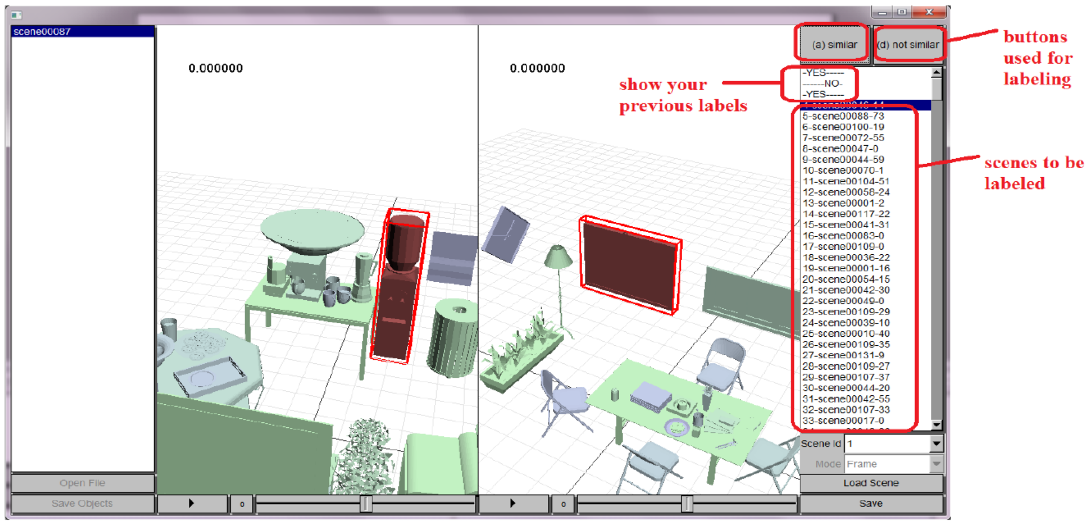


Figure D.2: The interface for the user study.

In Figure D.2, both the window on the left and the right have a red object with a bounding box. What you need to decide is if the two red objects here are similar (according to spatial relationship between themselves and their environment). If you think their spatial relationships with the environment are similar, click the "similar" button; if not, click the "not similar" button.

Please note that, you should NOT make your decisions based on color, shape, or size.

The Labelling Result

Each user labelled 500 results for each query. We consider a scene as related to the query if more than half of the users label it as related.

According to the labelling result, there are **46** related results for the plate (object 8 of scene00050), **50** for the water dispenser (object 37 of scene00087), **49** for the chair (object 14 of scene00118) and **85** for the table (object 14 of scene00109) among the 500 results.

Appendix E

Retrieval Results 1

Here we show the retrieval results for the four queries which are evaluated by user's labels in Section 4.3.4. Note that here the retrieval was done on the whole database.

We show retrieval results for all depths (from 1 to the maximal depth) for each query. We also show both the results of our method(IBS) and the displacement feature(DIS) for comparison.

The query object is always shown in a red bounding box in the query scene. The result objects are shown in red, and the result scenes are ordered by their similarity(left-to-right, top-to-bottom). The note below each result with the format "XXXX-YY" means the current result is the YY-th object of the scene named XXXX. The object ID and the scene names are from the Stanford Scene Database.

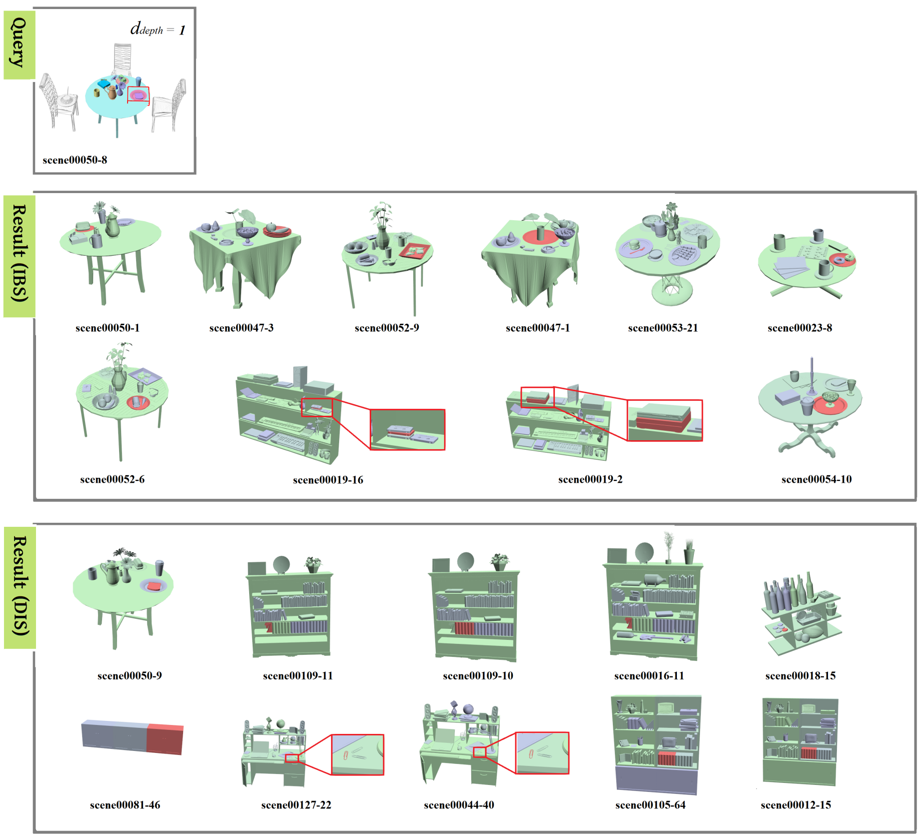


Figure E.1: Comparison (query 1 of 4).

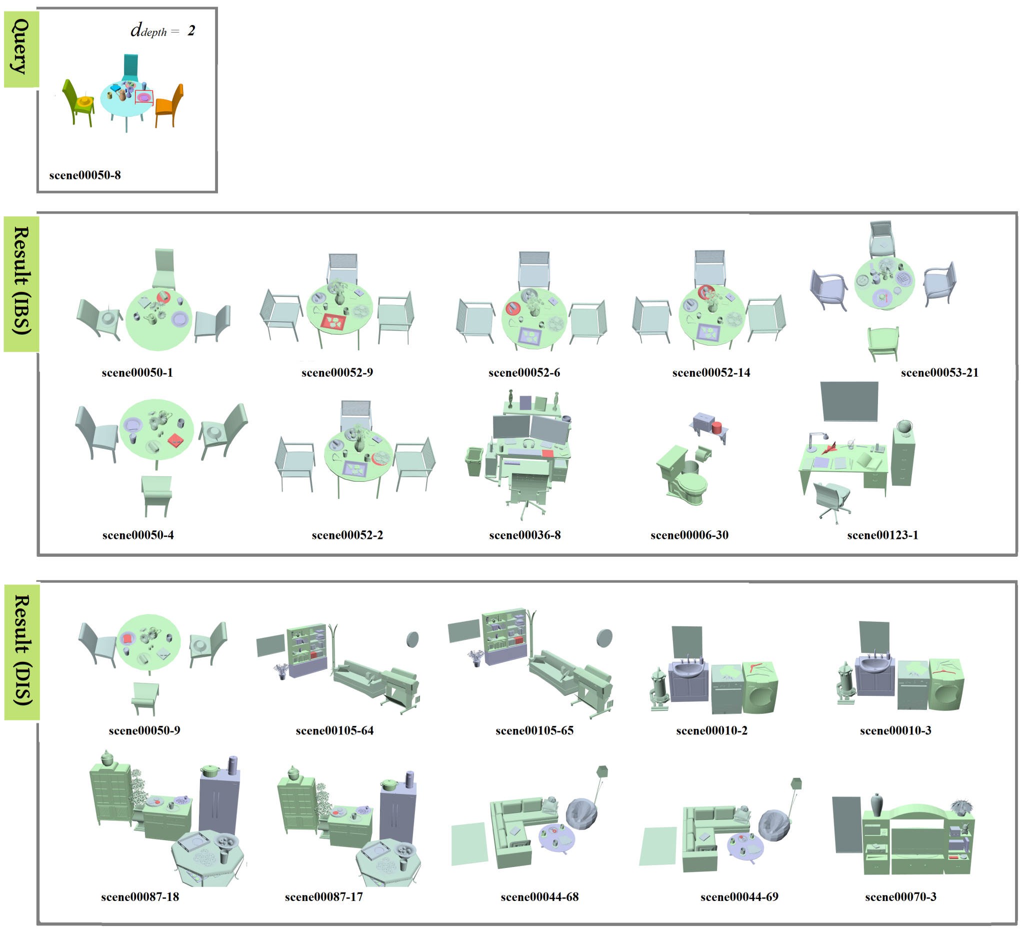


Figure E.2: Comparison (query 1 of 4).

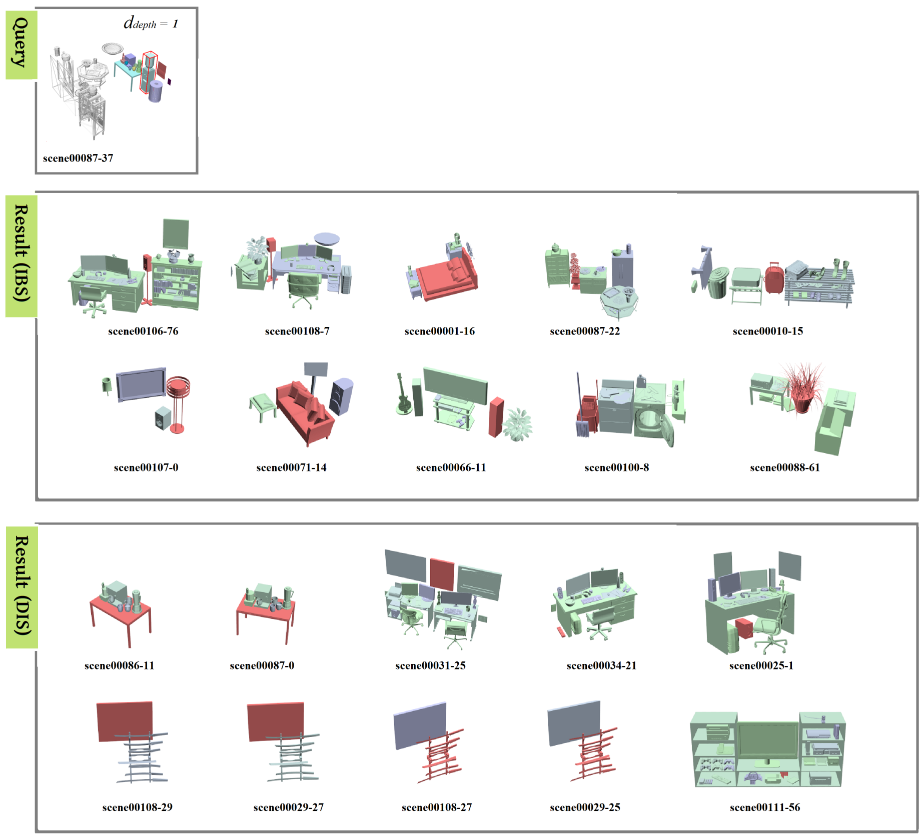


Figure E.3: Comparison (query 2 of 4).

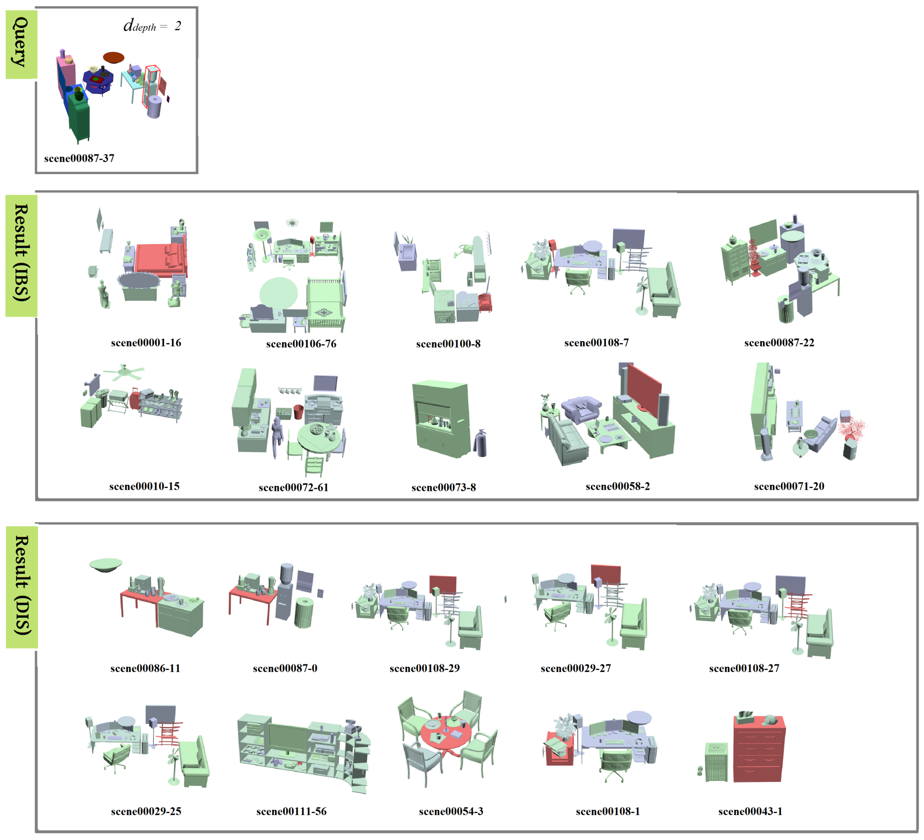


Figure E.4: Comparison (query 2 of 4).

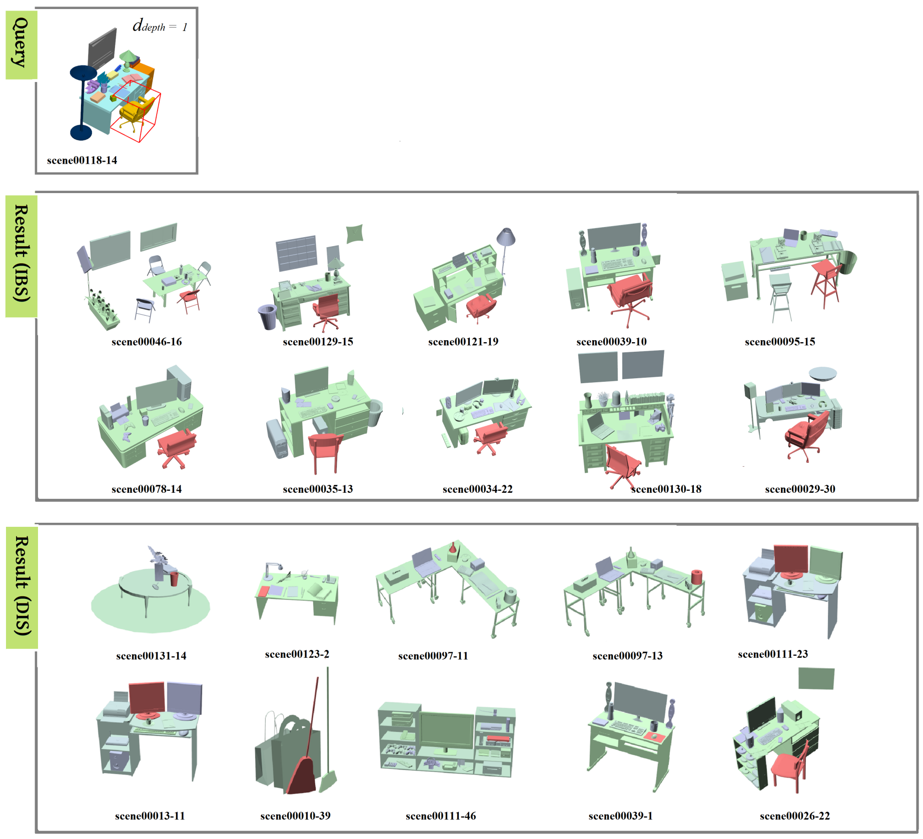


Figure E.5: Comparison (query 3 of 4).

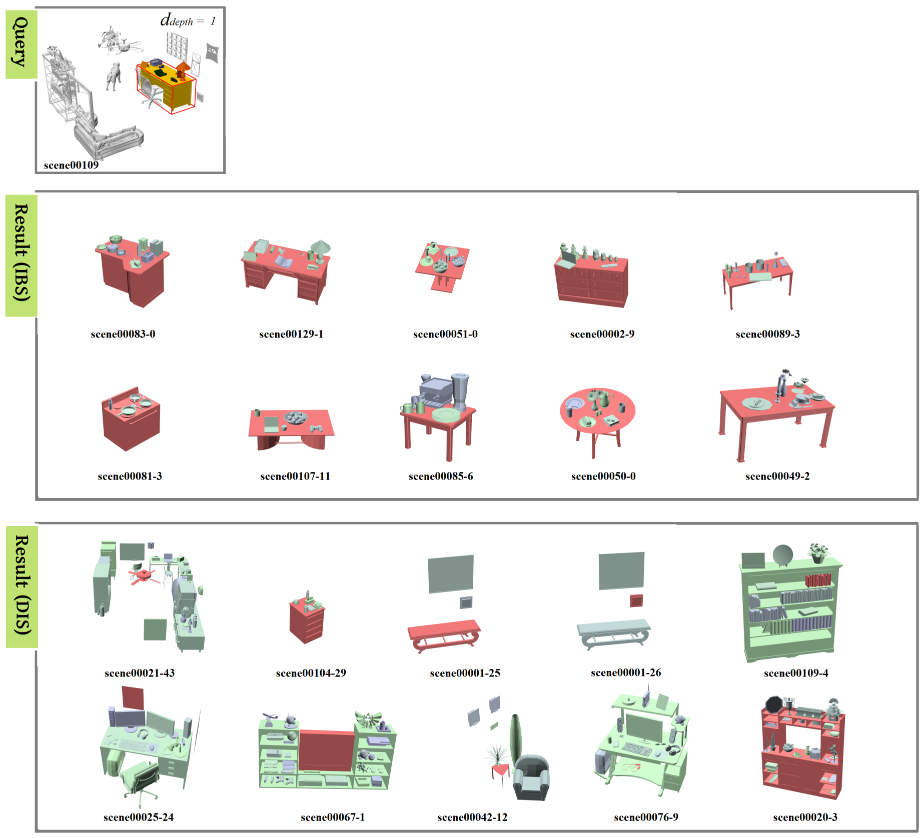


Figure E.6: Comparison (query 4 of 4).

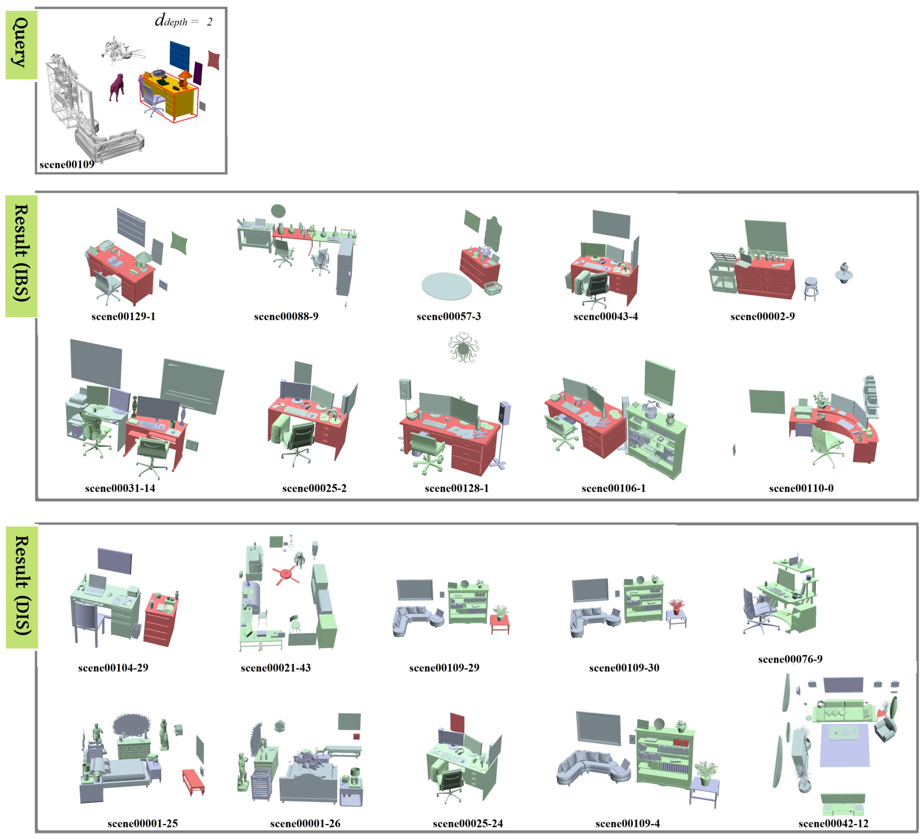


Figure E.7: Comparison (query 4 of 4).

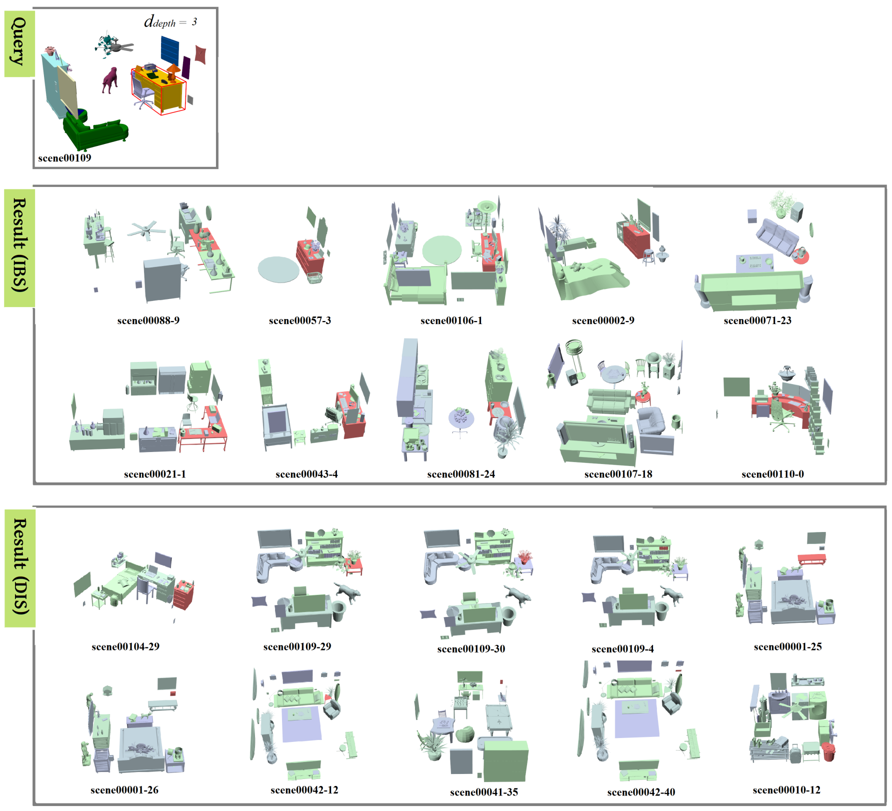


Figure E.8: Comparison (query 4 of 4).

Appendix F

Retrieval Results 2

Here we show more retrieval results produced by our method presented in Section 4.3. This section is a supplementary to the Section 4.3.4.

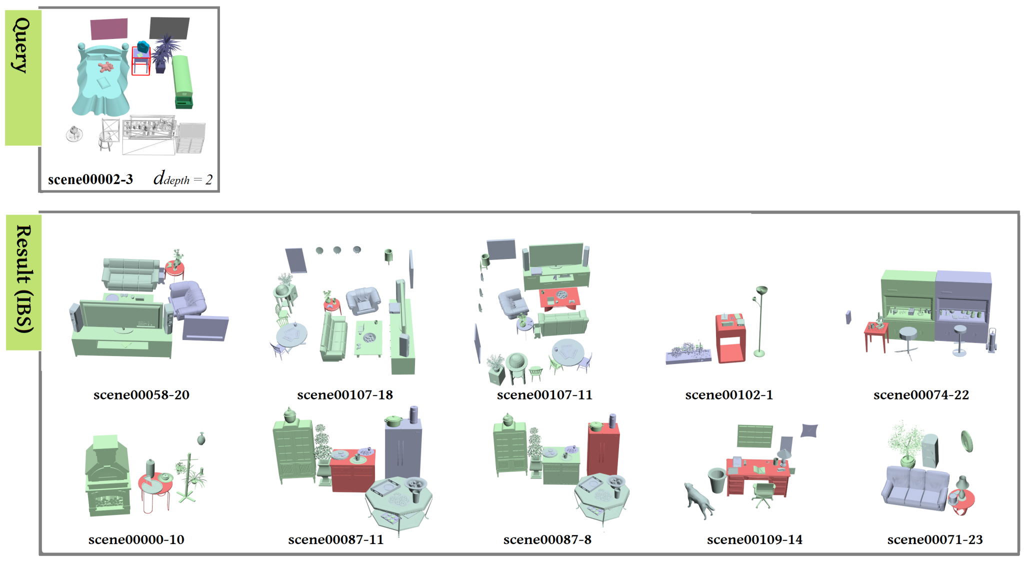


Figure F.1: Retrieval result (example 1 of 6).

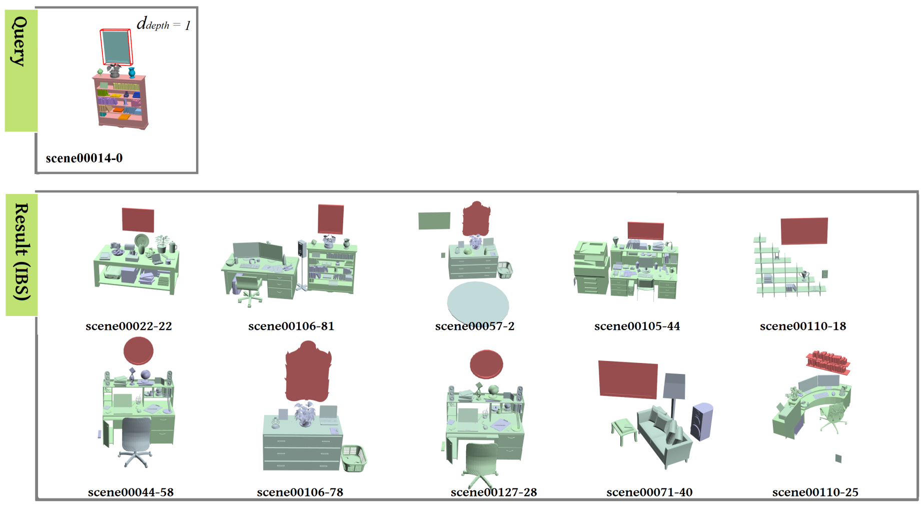


Figure F.2: Retrieval result (example 2 of 6).

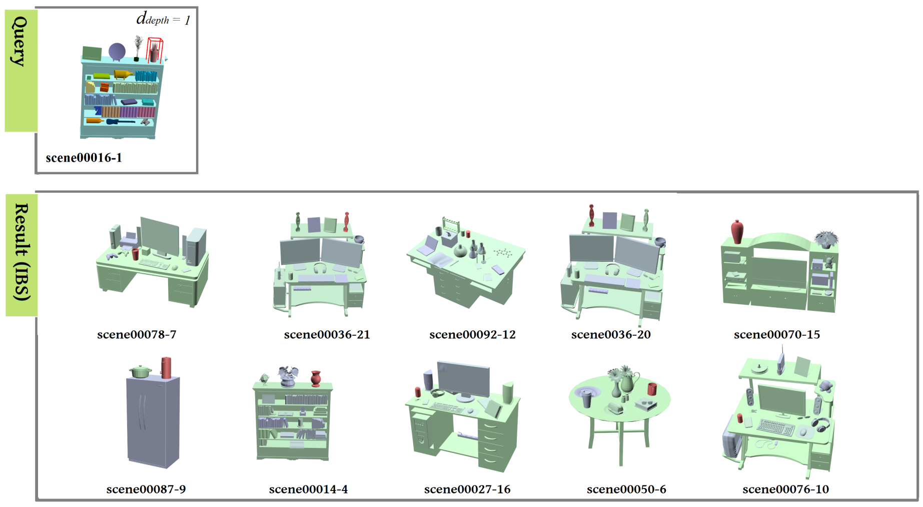


Figure F.3: Retrieval result (example 3 of 6).

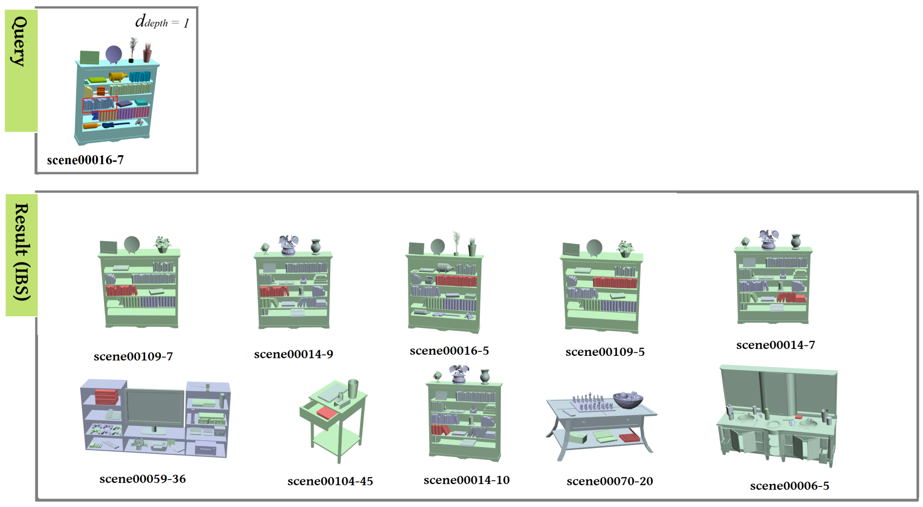


Figure F.4: Retrieval result (example 4 of 6).

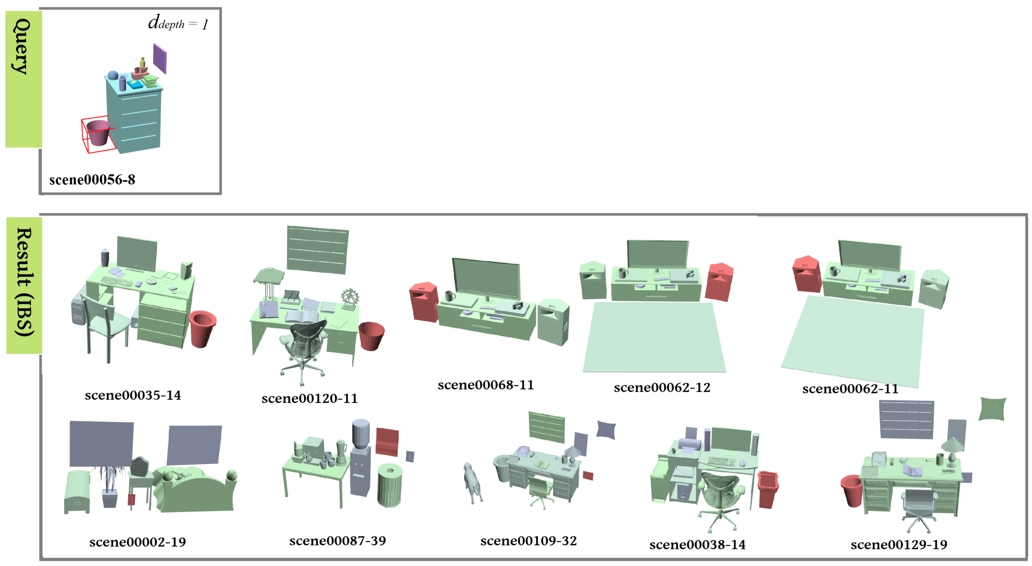


Figure F.5: Retrieval result (example 5 of 6).

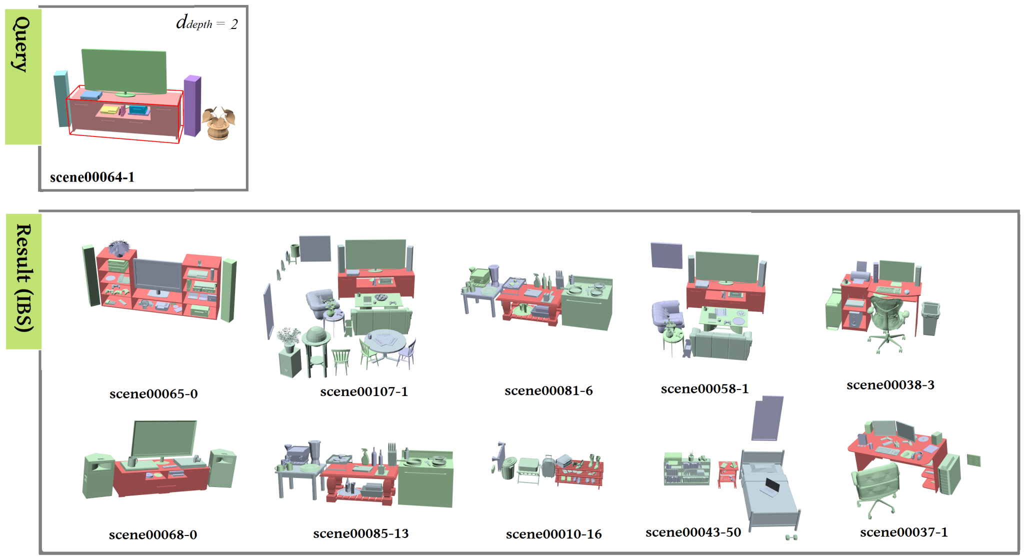


Figure F.6: Retrieval result (example 6 of 6).

Appendix G

Geometric Hashing

Geometric hashing, a technique originally developed in computer vision for matching geometric features against a database of such features, find use in a number of other areas. Matching is possible even when the recognizable database objects have undergone transformations or when only partial information is present. The technique is highly efficient and of low polynomial complexity.

[Wolfson and Rigoutsos, 1997]

The underlying ideas of geometric hashing are to save all the geometric configurations of a set of points (the matching target) into a hash table, and use this hash table to quickly detect if a new set of points are present in the target. We show the pipeline of the geometric hashing method in Figure G.1.

The first stage of the method, the preprocessing, is to build the hash table. The concept "basis" is a pair of points (in 2D) and a triplet of points (in 3D) from the point set . Here we use 2D point set as example. Let's take the pair of points p_i, p_j as a basis of the targeting point set. We transform the whole set so that the magnitude of vector $\mathbf{p_i p_j}$ equals 1 and is in the direction of the positive x axis. The remaining points are also transformed with the same transformation. The transformed point set is used as one item in the hash table with the entry (i, j). The contents of the hash table can be computed offline.

In the recognition phase, each pair of points from the given point set are chosen as a candidate basis, and for each basis the point set is transformed so that the magnitude of the basis vector equals 1 and is in the direction of the positive x axis. Then each of the transformed points in the set is mapped to the hash table, and a voting schema is used to measure how good the mapped points match to the hash table items.

A more detailed description of the method is shown in Figure G.2.

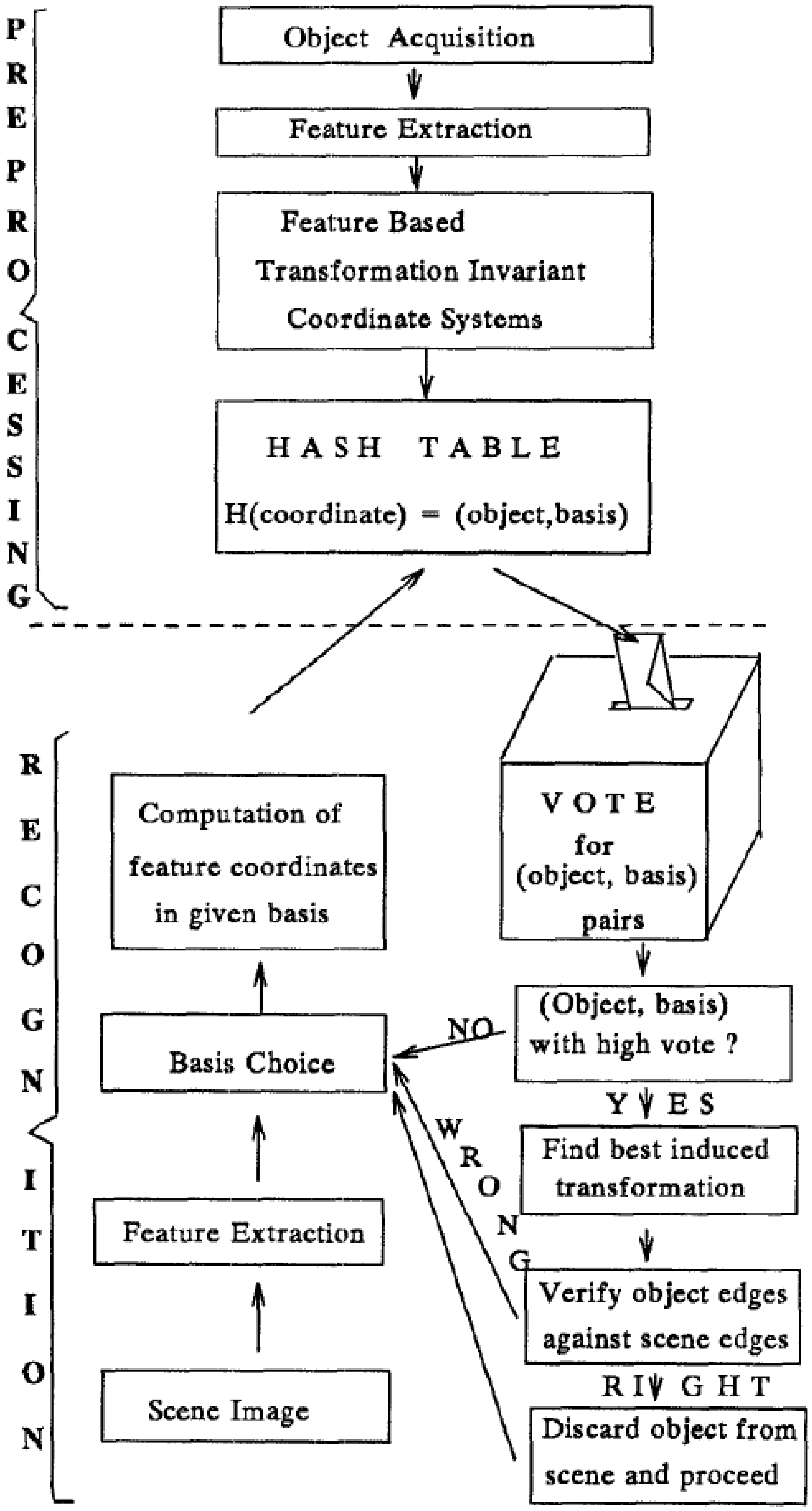
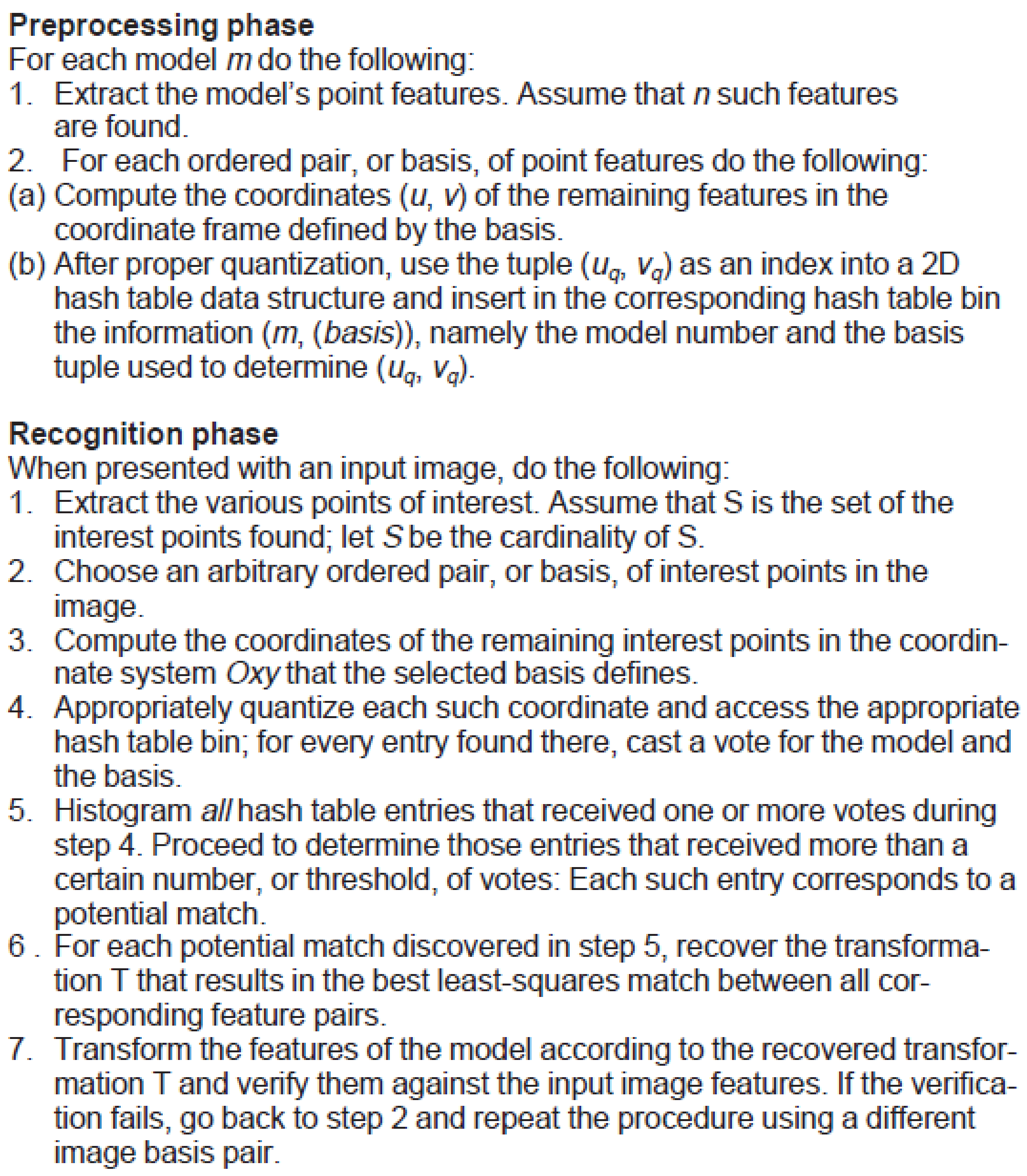

PREPROCESSING
Object Acquisition
Feature Extraction
Feature Based Transformation Invariant Coordinate Systems
HASH TABLE
H(coordinate) = (object,basis)
RECOGNITION
Computation of feature coordinates in given basis
VOTE for (object, basis) pairs
(Object, basis) with high vote ?
NO
Basis Choice
YES
Find best induced transformation
WRONG
Feature Extraction
Verify object edges against scene edges
RIGHT
Scene Image
Discard object from scene and proceed

Figure G.1: The pipeline of the geometric hashing method. Cited from [Lamdan and Wolfson, 1988]



Preprocessing phase

For each model *m* do the following:

1. Extract the model's point features. Assume that *n* such features are found.
2. For each ordered pair, or basis, of point features do the following:

(a) Compute the coordinates (*u*, *v*) of the remaining features in the coordinate frame defined by the basis.

(b) After proper quantization, use the tuple (u_q, v_q) as an index into a 2D hash table data structure and insert in the corresponding hash table bin the information (*m*, (*basis*)), namely the model number and the basis tuple used to determine (u_q, v_q).

Recognition phase

When presented with an input image, do the following:

1. Extract the various points of interest. Assume that S is the set of the interest points found; let *S* be the cardinality of S.
2. Choose an arbitrary ordered pair, or basis, of interest points in the image.
3. Compute the coordinates of the remaining interest points in the coordinate system *Oxy* that the selected basis defines.
4. Appropriately quantize each such coordinate and access the appropriate hash table bin; for every entry found there, cast a vote for the model and the basis.
5. Histogram *all* hash table entries that received one or more votes during step 4. Proceed to determine those entries that received more than a certain number, or threshold, of votes: Each such entry corresponds to a potential match.
6. For each potential match discovered in step 5, recover the transformation T that results in the best least-squares match between all corresponding feature pairs.
7. Transform the features of the model according to the recovered transformation T and verify them against the input image features. If the verification fails, go back to step 2 and repeat the procedure using a different image basis pair.

Figure G.2: The two stages of the geometric hashing system. Cited from [Wolfson and Rigoutsos, 1997]

Appendix H

Publication and Acknowledgements

The content of Chapter 3 and Chapter 4 in this thesis has been published in the following paper:

Xi Zhao, He Wang, and Taku Komura. 2014. "Indexing 3D Scenes Using the Interaction Bisector Surface." ACM Transactions on Graphics (TOG) 33 (3): 22.

The experiments in Section 4.1 was done with collaboration of He Wang.

The interface of the retrieval system used in Section 4.3 and Section 5.5 was made with the help from Myung Geol Choi.

Bibliography

Harrison Ainsworth. Minilight: a minimal global illumination renderer, 2013. URL http://www.hxa.name/minilight/. 98

Yoshiaki Akazawa, Yoshihiro Okada, and Koichi Niijima. Automatic 3d scene generation based on contact constraints. In *Proc. Conf. on Computer Graphics and Artificial Intelligence*, page 593–598, 2005. URL http://3ia2008.teiath.gr/3ia_previous_conferences_cds/2005/Papers/Papers/Paper02.pdf. 24

E.E. Aksoy, A. Abramov, F. Worgotter, and B. Dellen. Categorizing object-action relations from semantic scene graphs. In *2010 IEEE International Conference on Robotics and Automation (ICRA)*, pages 398–405, May 2010. doi: 10.1109/ROBOT.2010.5509319. 19

Luıs A. Alexandre. 3d descriptors for object and category recognition: a comparative evaluation. In *Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vilamoura, Portugal*. Citeseer, 2012. URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.309.6312&rep=rep1&type=pdf. 46

Nina Amenta, Sunghee Choi, and Ravi Krishna Kolluri. The power crust. In *Proceedings of the Sixth ACM Symposium on Solid Modeling and Applications*, SMA '01, page 249–266, New York, NY, USA, 2001. ACM. ISBN 1-58113-366-9. doi: 10.1145/376957.376986. URL http://doi.acm.org/10.1145/376957.376986. 28, 29, 31

Mihael Ankerst, Gabi Kastenmüller, Hans-Peter Kriegel, and Thomas Seidl. 3d shape histograms for similarity search and classification in spatial databases. In *Advances in Spatial Databases*, page 207–226. Springer, 1999. URL http://link.springer.com/chapter/10.1007/3-540-48482-5_14. 11, 12

Dominique Attali, Jean-Daniel Boissonnat, and Herbert Edelsbrunner. Stability and computation of medial axes-a state-of-the-art report. In *Mathematical foundations of scientific visualization, computer graphics, and massive data exploration*, page 109–125. Springer, 2009. URL http://link.springer.com/chapter/10.1007/b106657_6. ix, 26, 27, 29, 36

Oscar Kin-Chung Au, Chiew-Lan Tai, Hung-Kuo Chu, Daniel Cohen-Or, and Tong-Yee Lee. Skeleton extraction by mesh contraction. In *ACM SIGGRAPH 2008 Papers*, SIGGRAPH '08, page 44:1–44:10, New York, NY, USA, 2008. ACM. ISBN

978-1-4503-0112-1. doi: 10.1145/1399504.1360643. URL http://doi.acm.org/10.1145/1399504.1360643. 13

Douglas Ayers and Mubarak Shah. Recognizing human actions in a static room. In *Applications of Computer Vision, 1998. WACV'98. Proceedings., Fourth IEEE Workshop on*, page 42–47. IEEE, 1998. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=732856. 18

Douglas Ayers and Mubarak Shah. Monitoring human behavior from video taken in an office environment. *Image and Vision Computing*, 19(12): 833–846, 2001. URL http://www.sciencedirect.com/science/article/pii/S0262885601000476. 18

Fan Bao, Michael Schwarz, and Peter Wonka. Procedural facade variations from a single layout. *ACM Trans. Graph.*, 32(1):8:1–8:13, February 2013. ISSN 0730-0301. doi: 10.1145/2421636.2421644. URL http://doi.acm.org/10.1145/2421636.2421644. 23

C. Bradford Barber, David P. Dobkin, and Hannu Huhdanpaa. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS)*, 22(4): 469–483, 1996. URL http://dl.acm.org/citation.cfm?id=235821. 28, 35

S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):509–522, April 2002. ISSN 0162-8828. doi: 10.1109/34.993558. 11

Serge Belongie, Jitendra Malik, and Jan Puzicha. Shape context: A new descriptor for shape matching and object recognition. In *NIPS*, volume 2, page 3, 2000. URL http://digital.cs.usu.edu/~xqi/Teaching/CS5650F03/FinalProject/ShapeDescriptor.Belongie.00.pdf. 11

Harry Blum. A transformation for extracting new descriptors of shape. *Models for the perception of speech and visual form*, 19(5):362–380, 1967. URL http://pageperso.lif.univ-mrs.fr/~edouard.thiel/rech/1967-blum.pdf. 13, 25

Aaron F. Bobick and Claudio S. Pinhanez. Controlling view-based algorithms using approximate world models and action information. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, page 955–961. IEEE, 1997. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=609443. 18

Martin Bokeloh, Michael Wand, and Hans-Peter Seidel. A connection between partial symmetry and inverse procedural modeling. *ACM Transactions on Graphics (TOG)*, 29(4):104, 2010. URL http://dl.acm.org/citation.cfm?id=1778841. 23

Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, page 144–152. ACM, 1992. URL http://dl.acm.org/citation.cfm?id=130401. 51

Melissa Bowerman and Eric Pederson. *Topological relations picture series*. Stephen C. Levinson (ed.), Space stimuli kit 1.2, Nijmegen: Max Planck Institute for Psycholinguistics., November 1992. URL http://fieldmanuals.mpi.nl/volumes/1992/bowped/. ix, 7, 8

J. W. Brandt. Convergence and continuity criteria for discrete approximations of the continuous planar skeleton. *CVGIP: Image Understanding*, 59(1):116–124, January 1994. ISSN 1049-9660. doi: 10.1006/ciun.1994.1007. URL http://www.sciencedirect.com/science/article/pii/S1049966084710072. 29

Alexander M. Bronstein and Michael M. Bronstein. Spatially-sensitive affine-invariant image descriptors. In *Computer Vision–ECCV 2010*, page 197–208. Springer, 2010. URL http://link.springer.com/chapter/10.1007/978-3-642-15552-9_15. 13

Tom Bruns and Max Egenhofer. Similarity of spatial scenes. In *Seventh international symposium on spatial data handling*, page 31–42. Delft, The Netherlands, 1996. URL http://www.spatial.maine.edu/~max/sdh-similarity.pdf. 9

L. A. Carlson-Radvansky, E. S. Covey, and K. M. Lattanzi. ”what” effects on ”where”: Functional influences on spatial relations. *Psychological Science*, 10(6):516–521, November 1999. ISSN 0956-7976, 1467-9280. doi: 10.1111/1467-9280.00198. URL http://pss.sagepub.com/lookup/doi/10.1111/1467-9280.00198. 7

Gunnar Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46(2):255–308, 2009. URL http://www.ams.org/journals/bull/2009-46-02/S0273-0979-09-01249-X/. 41

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011. URL http://dl.acm.org/citation.cfm?id=1961199. 51

Ming-Ching Chang and Benjamin B. Kimia. Measuring 3d shape similarity by matching the medial scaffolds. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, page 1473–1480. IEEE, 2009. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5457437. 14, 30

F. Chazal and A. Lieutier. Stability and homotopy of a subset of the medial axis. In *Proceedings of the Ninth ACM Symposium on Solid Modeling and Applications*, SM ’04, page 243–248, Aire-la-Ville, Switzerland, Switzerland, 2004. Eurographics Association. ISBN 3-905673-55-X. URL http://dl.acm.org/citation.cfm?id=1217875.1217913. 30

Sung Woo Choi and Seong-whan Lee. Stability analysis of medial axis transform under relative hausdorff distance. In *Proc. 15th ICPR*, page 139–142, 2000. 30

A.G. Cohn and S.M. Hazarika. Qualitative spatial representation and reasoning: An overview. *Fundamenta Informaticae*, 46(1):1–29, January 2001. URL http://iospress.metapress.com/content/91BFAQ1JUUJ50Y9K. 8

N.D. Cornea, D. Silver, and P. Min. Curve-skeleton properties, applications, and algorithms. *IEEE Transactions on Visualization and Computer Graphics*, 13(3):530–548, May 2007. ISSN 1077-2626. doi: 10.1109/TVCG.2007.1002. 13

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995. URL http://link.springer.com/article/10.1007/BF00994018. 51

Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, volume 1, page 1–2, 2004. URL http://217.109.185.161/layout/set/print/content/download/20785/148346/file/2004_010.pdf. 11

Tim Culver, John Keyser, and Dinesh Manocha. Accurate computation of the medial axis of a polyhedron. In *Proceedings of the fifth ACM symposium on Solid modeling and applications*, page 179–190. ACM, 1999. URL http://dl.acm.org/citation.cfm?id=304030. 26, 27, 47

Mark De Berg, Marc Van Kreveld, Mark Overmars, and Otfried Cheong Schwarzkopf. *Computational geometry*. Springer, 2000. URL http://link.springer.com/chapter/10.1007/978-3-662-04245-8_1. 28

Cecil Jose A. Delfinado and Herbert Edelsbrunner. An incremental algorithm for betti numbers of simplicial complexes on the 3-sphere. *Computer Aided Geometric Design*, 12(7):771–784, 1995. URL http://www.sciencedirect.com/science/article/pii/016783969500016Y. 41

Tamal K. Dey and Wulue Zhao. Approximating the medial axis from the voronoi diagram with a convergence guarantee. *Algorithmica*, 38(1):179–200, 2004. URL http://link.springer.com/article/10.1007/s00453-003-1049-y. ix, 26, 28, 30

Tamal K. Dey, Fengtao Fan, and Yusu Wang. An efficient computation of handle and tunnel loops via reeb graphs. *ACM Trans. Graph.*, 32(4):32:1–32:10, July 2013. ISSN 0730-0301. doi: 10.1145/2461912.2462017. URL http://doi.acm.org/10.1145/2461912.2462017. 40

S.J. Dickinson, AP. Pentland, and Azriel Rosenfeld. 3-d shape recovery using distributed aspect matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):174–198, February 1992. ISSN 0162-8828. doi: 10.1109/34.121788. 14

Gershon Elber and Myung-Soo Kim. The bisector surface of freeform rational space curves. In *Proceedings of the Thirteenth Annual Symposium on Computational Geometry*, SCG '97, page 473–474, New York, NY, USA, 1997. ACM. ISBN 0-89791-878-9. doi: 10.1145/262839.263091. URL http://doi.acm.org/10.1145/262839.263091. 27, 47

Rida T. Farouki and John K. Johnstone. Computing point/curve and curve/curve bisectors. In *Proceedings of the 5th IMA Conference on the Mathematics of Surfaces*, page 327–354, New York, NY, USA, 1994. Clarendon Press. ISBN 0-19-853483-3. URL http://dl.acm.org/citation.cfm?id=646871.709541. 26, 27

Michele I. Feist. *On In and On: An investigation into the linguistic encoding of spatial scenes*. PhD thesis, NORTHWESTERN UNIVERSITY, 2000. URL http://www.academia.edu/download/31047808/thesis.pdf. 8

V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid. Groups of adjacent contour segments for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(1):36–51, January 2008. ISSN 0162-8828. doi: 10.1109/TPAMI.2007.1144. 19

Martin A Fischler and R.A Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on Computers*, C-22(1):67–92, January 1973. ISSN 0018-9340. doi: 10.1109/T-C.1973.223602. 14

Matthew Fisher and Pat Hanrahan. Context-based search for 3d models. In *ACM SIGGRAPH Asia 2010 Papers*, SIGGRAPH ASIA '10, page 182:1–182:10, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0439-9. doi: 10.1145/1866158.1866204. URL http://doi.acm.org/10.1145/1866158.1866204. x, xi, 50, 51, 53, 54, 62, 66, 68

Matthew Fisher, Manolis Savva, and Pat Hanrahan. Characterizing structural relationships in scenes using graph kernels. In *ACM Transactions on Graphics (TOG)*, volume 30, page 34. ACM, 2011. URL http://dl.acm.org/citation.cfm?id=1964929. xi, 2, 3, 17, 53, 54, 55, 60

Matthew Fisher, Daniel Ritchie, Manolis Savva, Thomas Funkhouser, and Pat Hanrahan. Example-based synthesis of 3d object arrangements. *ACM Transactions on Graphics (TOG)*, 31(6):135, 2012. URL http://dl.acm.org/citation.cfm?id=2366154. 2, 3, 17, 25, 49, 53

S Fortune. A sweepline algorithm for voronoi diagrams. In *Proceedings of the Second Annual Symposium on Computational Geometry*, SCG '86, page 313–322, New York, NY, USA, 1986. ACM. ISBN 0-89791-194-6. doi: 10.1145/10515.10549. URL http://doi.acm.org/10.1145/10515.10549. 28

Mark Foskey, Ming C. Lin, and Dinesh Manocha. Efficient computation of a simplified medial axis. In *Proceedings of the Eighth ACM Symposium on Solid Modeling and Applications*, SM '03, page 96–107, New York, NY, USA, 2003. ACM. ISBN 1-58113-706-0. doi: 10.1145/781606.781623. URL http://doi.acm.org/10.1145/781606.781623. 30

Thomas Funkhouser, Michael Kazhdan, Philip Shilane, Patrick Min, William Kiefer, Ayellet Tal, Szymon Rusinkiewicz, and David Dobkin. Modeling by example. In *ACM Transactions on Graphics (TOG)*, volume 23, page 652–663. ACM, 2004. URL http://dl.acm.org/citation.cfm?id=1015775. 22

Nikhil Gagvani and Deborah Silver. Parameter-controlled volume thinning. *Graphical Models and Image Processing*, 61(3):149–164, May 1999. ISSN 1077-3169. doi: 10.1006/gmip.1999.0495. URL http://www.sciencedirect.com/science/article/pii/S1077316999904951. 13

Ran Gal and Daniel Cohen-Or. Salient geometric features for partial shape matching and similarity. *ACM Transactions on Graphics (TOG)*, 25(1):130–150, 2006. URL http://dl.acm.org/citation.cfm?id=1122507. 101

T. Germer and M. Schwarz. Procedural arrangement of furniture for real-time walkthroughs. *Computer Graphics Forum*, 28(8):2068–2078, 2009. ISSN 1467-8659. doi: 10.1111/j.1467-8659.2009.01351.x. URL http://onlinelibrary.wiley.com/doi/10.1111/j.1467-8659.2009.01351.x/abstract. 24

E. Goldstein. *Sensation and Perception*. Cengage Learning, February 2009. ISBN 0495601497. 8, 55

Aleksey Golovinskiy and Thomas Funkhouser. Randomized cuts for 3d mesh analysis. In *ACM Transactions on Graphics (TOG)*, volume 27, page 145. ACM, 2008. URL http://dl.acm.org/citation.cfm?id=1409098. 15

Aleksey Golovinskiy and Thomas Funkhouser. Consistent segmentation of 3d models. *Computers & Graphics*, 33(3):262–269, June 2009. ISSN 00978493. doi: 10.1016/j.cag.2009.03.010. URL http://linkinghub.elsevier.com/retrieve/pii/S0097849309000454. 15

Leo A. Goodman and William H. Kruskal. Measures of association for cross classifications*. *Journal of the American Statistical Association*, 49(268):732–764, 1954. URL http://www.tandfonline.com/doi/abs/10.1080/01621459.1954.10501231. 56, 123

H. Grabner, J. Gall, and L. Van Gool. What makes a chair a chair? In *2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1529–1536, June 2011. doi: 10.1109/CVPR.2011.5995327. 19, 21

Robert P. Haining. *Spatial Data Analysis: Theory and Practice*. Cambridge University Press, April 2003. ISBN 9780521774376. 7

Inbal Halperin, Buyong Ma, Haim Wolfson, and Ruth Nussinov. Principles of docking: An overview of search algorithms and a guide to scoring functions. *Proteins: Structure, Function, and Genetics*, 47(4):409–443, June 2002. ISSN 0887-3585, 1097-0134. doi: 10.1002/prot.10115. URL http://doi.wiley.com/10.1002/prot.10115. 9

I. Hanniel, M. Ramanathan, G. Elber, and M. S. Kim. Precise voronoi cell extraction of free-form planar piecewise c1-continuous closed rational curves. *International Journal of Computational Geometry & Applications*, 17(05):453–486, October 2007. ISSN 0218-1959, 1793-6357. doi: 10.1142/S0218195907002446. URL http://www.worldscientific.com/doi/abs/10.1142/S0218195907002446. 26

Joachim Hartung, Guido Knapp, and Bimal K. Sinha. *Statistical meta-analysis with applications*, volume 738. John Wiley & Sons, 2011. URL http://books.google.co.uk/books?hl=zh-CN&lr=&id=JEoNB_2NONQC&oi=fnd&pg=PT9&dq=statistical+meta+analysis+with+applications&ots=eHfLYkIOxF&sig=cO_UjKly1yVJIFW6AYkvR-MOaOQ. 59

Trevor Hastie, Robert Tibshirani, Jerome Friedman, T. Hastie, J. Friedman, and R. Tibshirani. *The elements of statistical learning*, volume 2. Springer, 2009. URL http://link.springer.com/content/pdf/10.1007/978-0-387-84858-7.pdf. 53

Martin Held. On computing voronoi diagrams of convex polyhedra by means of wavefront propagation. In *CCCG*, page 128–133, 1994. URL ftp://129.49.108.37/geometry/cccg94.ps.gz. 28

Masaki Hilaga, Yoshihisa Shinagawa, Taku Kohmura, and Tosiyasu L. Kunii. Topology matching for fully automatic similarity estimation of 3d shapes. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, page 203–212. ACM, 2001. URL http://dl.acm.org/citation.cfm?id=383282. 13

Masayuki Hisada, Alexander G. Belyaev, and Tosiyasu L. Kunii. A skeleton-based approach for detection of perceptually salient features on polygonal surfaces. *Computer Graphics Forum*, 21(4):689–700, 2002. ISSN 1467-8659. doi: 10.1111/1467-8659.00627. URL http://onlinelibrary.wiley.com/doi/10.1111/1467-8659.00627/abstract. 31

Edmond SL Ho, Taku Komura, and Chiew-Lan Tai. Spatial relationship preserving character motion adaptation. In *ACM Transactions on Graphics (TOG)*, volume 29, page 33. ACM, 2010. URL http://dl.acm.org/citation.cfm?id=1778770. 21

Kenneth E. Hoff, III, John Keyser, Ming Lin, Dinesh Manocha, and Tim Culver. Fast computation of generalized voronoi diagrams using graphics hardware. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, page 277–286, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co. ISBN 0-201-48560-5. doi: 10.1145/311535.311567. URL http://dx.doi.org/10.1145/311535.311567. 28

Hui Huang, Shihao Wu, Daniel Cohen-Or, Minglun Gong, Hao Zhang, Guiqing Li, and Baoquan Chen. L1-medial skeleton of point cloud. *ACM TOG*, 32, 2013. URL http://web.siat.ac.cn/~huihuang/Skeleton/skeleton_small.pdf. 13

Qixing Huang, Vladlen Koltun, and Leonidas Guibas. Joint shape segmentation with linear programming. In *ACM Transactions on Graphics (TOG)*, volume 30, page 125. ACM, 2011. URL http://dl.acm.org/citation.cfm?id=2024159. 15

Toshiyuki Imai. A topology oriented algorithm for the voronoi diagram of polygons. pages 107–112. Carleton University Press, 1996. URL http://www.bibsonomy.org/bibtex/231415e005ec2a2597bd48fb9febec46c/dblp. 28

Alec Jacobson, Ladislav Kavan, and Olga Sorkine-Hornung. Robust inside-outside segmentation using generalized winding numbers. *ACM Trans. Graph*, 32:4, 2013. URL http://igl.ethz.ch/projects/winding-number/robust-inside-outside-segmentation-using-generalized-winding-numbers-siggraph pdf. 95

Arjun Jain, Thorsten Thormählen, Tobias Ritschel, and Hans-Peter Seidel. Exploring shape variations by 3d-model decomposition and part-based recombination. *Computer Graphics Forum*, 31(2pt3):631–640, 2012. ISSN 1467-8659. doi: 10.1111/j.1467-8659.2012.03042.x. URL http://onlinelibrary.wiley.com/doi/10.1111/j.1467-8659.2012.03042.x/abstract. 15, 22

Wei Jiang, Kai Xu, Zhi-Quan Cheng, Ralph R. Martin, and Gang Dang. Curve skeleton extraction by coupled graph contraction and surface clustering. *Graphical Models*, 75(3):137–148, May 2013. ISSN 1524-0703. doi: 10.1016/j.gmod.2012.10.005. URL http://www.sciencedirect.com/science/article/pii/S1524070312000732. 13

AE. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):433–449, May 1999. ISSN 0162-8828. doi: 10.1109/34.765655. 11

Evangelos Kalogerakis, Siddhartha Chaudhuri, Daphne Koller, and Vladlen Koltun. A probabilistic model for component-based shape synthesis. *ACM Transactions on Graphics (TOG)*, 31(4):55, 2012. URL http://dl.acm.org/citation.cfm?id=2185551. 15, 23

Sagi Katz and Ayellet Tal. Hierarchical mesh decomposition using fuzzy clustering and cuts. In *ACM SIGGRAPH 2003 Papers*, SIGGRAPH '03, page 954–961, New York, NY, USA, 2003. ACM. ISBN 1-58113-709-5. doi: 10.1145/1201775.882369. URL http://doi.acm.org/10.1145/1201775.882369. 13

Michael Kazhdan, Thomas Funkhouser, and Szymon Rusinkiewicz. Rotation invariant spherical harmonic representation of 3d shape descriptors. In *Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, page 156–164. Eurographics Association, 2003. URL http://dl.acm.org/citation.cfm?id=882392. 12

Jyri J. Kivinen and Christopher K. I. Williams. Transformation equivariant boltzmann machines. In *Proceedings of the 21th International Conference on Artificial Neural Networks - Volume Part I*, ICANN'11, page 1–9, Berlin, Heidelberg, 2011. Springer-Verlag. ISBN 978-3-642-21734-0. URL http://dl.acm.org/citation.cfm?id=2029556.2029557. 89

Hedvig Kjellström, Javier Romero, and Danica Kragić. Visual object-action recognition: Inferring object affordances from human demonstration. *Computer Vision and Image Understanding*, 115(1):81–90, 2011. URL http://www.sciencedirect.com/science/article/pii/S107731421000175X. 19

Peter J. Kostelec. SpharmonicKit, 2008. URL http://www.cs.dartmouth.edu/~geelong/sphere/. 96

Marcel Körtgen, Gil-Joo Park, Marcin Novotni, and Reinhard Klein. 3d shape matching with 3d shape contexts. In *The 7th central European seminar on computer graphics*, volume 3, page 5–17, 2003. URL http://cg.tuwien.ac.at/hostings/cescg/CESCG-2003/MKoertgen/paper.pdf. 11

Taesoo Kwon, Kang Hoon Lee, Jehee Lee, and Shigeo Takahashi. Group motion editing. *ACM Transactions on Graphics*, 27(3):1, August 2008. ISSN 07300301. doi: 10.1145/1360612.1360679. URL http://portal.acm.org/citation.cfm?doid=1360612.1360679. 20

Yehezkel Lamdan and Haim J. Wolfson. Geometric hashing: A general and efficient model-based recognition scheme. In *ICCV*, volume 88, page 238–249, 1988. URL http://www.cs.utexas.edu/~grauman/courses/spring2007/395T/395T/papers/Lamdan88.pdf. xv, 136

Mathieu Larive, Olivier Le Roux, and Véronique Gaildrat. Using meta-heuristics for constraint-based 3d objects layout. *3IA'04*, 2004. URL http://francophone.teiath.gr/3ia_previous_conferences_cds/2004/Papers/02.pdf. 24

Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, page 2169–2178. IEEE, 2006. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1641019. 11, 12

D. T. Lee. Medial axis transformation of a planar shape. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-4(4):363–369, July 1982. ISSN 0162-8828. doi: 10.1109/TPAMI.1982.4767267. 28

André Lieutier. Any open bounded subset of rn has the same homotopy type as its medial axis. *Computer-Aided Design*, 36(11):1029–1046, September 2004. ISSN 0010-4485. doi: 10.1016/j.cad.2004.01.011. URL http://www.sciencedirect.com/science/article/pii/S0010448504000065. ix, 25, 26

Jinjie Lin, Daniel Cohen-Or, Hao Zhang, Cheng Liang, Andrei Sharf, Oliver Deussen, and Baoquan Chen. Structure-preserving retargeting of irregular 3d architecture. In *ACM Transactions on Graphics (TOG)*, volume 30, page 183. ACM, 2011. URL http://dl.acm.org/citation.cfm?id=2024217. 23

David G. Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, page 1150–1157. Ieee, 1999. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=790410. 11

D. MacDonald, J. Lang, and M. McAllister. Evaluation of colour image segmentation hierarchies. In *The 3rd Canadian Conference on Computer and Robot Vision, 2006*, pages 27–27, June 2006. doi: 10.1109/CRV.2006.31. 56

Paul Merrell, Eric Schkufza, Zeyang Li, Maneesh Agrawala, and Vladlen Koltun. Interactive furniture layout using interior design guidelines. In *ACM SIGGRAPH 2011 Papers*, SIGGRAPH '11, page 87:1–87:10, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0943-1. doi: 10.1145/1964921.1964982. URL http://doi.acm.org/10.1145/1964921.1964982. 24

Takashi Michikawa and Hiromasa Suzuki. Non-manifold medial surface reconstruction from volumetric data. In *Proceedings of the 6th International Conference on Advances in Geometric Modeling and Processing*, GMP'10, page 124–136, Berlin, Heidelberg, 2010. Springer-Verlag. ISBN 3-642-13410-6, 978-3-642-13410-4. doi: 10.1007/978-3-642-13411-1_9. URL http://dx.doi.org/10.1007/978-3-642-13411-1_9. 30

Victor Milenkovic. Robust construction of the voronoi diagram of a polyhedron. In *In Proc. 5th Canad. Conf. Comput. Geom*, page 473–478, 1993. 27, 28

Niloy Mitra, Michael Wand, Hao (Richard) Zhang, Daniel Cohen-Or, Vladimir Kim, and Qi-Xing Huang. Structure-aware shape processing. In *SIGGRAPH Asia 2013 Courses*, SA '13, page 1:1–1:20, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2631-5. doi: 10.1145/2542266.2542267. URL http://doi.acm.org/10.1145/2542266.2542267. 14, 15

Meinard Müller. DTW-based motion comparison and retrieval. In *Information Retrieval for Music and Motion*, pages 211–226. Springer Berlin Heidelberg, January 2007. ISBN 978-3-540-74047-6, 978-3-540-74048-3. URL http://link.springer.com/chapter/10.1007/978-3-540-74048-3_10. 78

Trimble Navigation Ltd. 3d warehouse, 2006. URL https://3dwarehouse.sketchup.com/index.html. 17

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*, page 216–225. ACM, 2003. URL http://dl.acm.org/citation.cfm?id=781639. 12

Atsuyuki Okabe, Barry Boots, Kokichi Sugihara, and Sung Nok Chiu. *Spatial tessellations: concepts and applications of Voronoi diagrams*, volume 501. John Wiley & Sons, 2009. URL http://books.google.co.uk/books?hl=zh-CN&lr=&id=dT7YH3mjeeIC&oi=fnd&pg=PP2&dq=concepts+and+applications+of+voronoi+diagrams&ots=82dx3ur7kk&sig=NnH2EhT1KNwTI8ylXTPUdlOevlE. 26

Nuria M. Oliver, Barbara Rosario, and Alex P. Pentland. A bayesian computer vision system for modeling human interactions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):831–843, 2000. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=868684. 20

Robert Osada, Thomas Funkhouser, Bernard Chazelle, and David Dobkin. Shape distributions. *ACM Transactions on Graphics (TOG)*, 21(4):807–832, 2002. URL http://dl.acm.org/citation.cfm?id=571648. 11

Maks Ovsjanikov, Alexander M. Bronstein, Michael M. Bronstein, and Leonidas J. Guibas. Shape google: a computer vision approach to isometry invariant shape retrieval. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, page 320–327. IEEE, 2009. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5457682. 13

Panagiotis Papadakis, Ioannis Pratikakis, Stavros Perantonis, and Theoharis Theoharis. Efficient 3d shape matching and retrieval using a concrete radialized spherical projection representation. *Pattern Recognition*, 40(9):2437–2452, 2007. URL http://www.sciencedirect.com/science/article/pii/S0031320307000106. xiv, 97

Laura Paraboschi, Silvia Biasotti, and Bianca Falcidieno. Comparing sets of 3d digital shapes through topological structures. In *Graph-Based Representations in Pattern Recognition*, page 114–125. Springer, 2007. URL http://link.springer.com/chapter/10.1007/978-3-540-72903-7_11. 17

Sangho Park and J. K. Aggarwal. Semantic-level understanding of human actions and interactions using event hierarchy. In *Computer Vision and Pattern Recognition Workshop, 2004. CVPRW'04. Conference on*, page 12–12. IEEE, 2004. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1384801. 20, 21

Sangho Park and Jake K. Aggarwal. Recognition of human interaction using multiple features in gray scale images. In *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, volume 1, page 51–54. IEEE, 2000. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=905274. 20

Patrick Peursum, Svetha Venkatesh, Geoff AW West, and Hung Hai Bui. Using interaction signatures to find and label chairs and floors. *Pervasive Computing, IEEE*, 3(4):58–65, 2004. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1369162. 19, 21, 83

Patrick Peursum, Geoff West, and Svetha Venkatesh. Combining image regions and human activity for indirect object recognition in indoor wide-angle views. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, page 82–89. IEEE, 2005. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1541242. 19, 21, 83

Gabriel Peyre and Laurent Cohen. Surface segmentation using geodesic centroidal tesselation. In *3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on*, page 995–1002. IEEE, 2004. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1335424. 71

Nicolas Ray, Xavier Cavin, Jean-Claude Paul, and Bernard Maigret. Intersurf: dynamic interface between proteins. *Journal of molecular graphics and modelling*, 23 (4):347–354, 2005. URL http://www.sciencedirect.com/science/article/pii/S109332630400107X. 9

Jayachandra M Reddy and George M Turkiyyah. Computation of 3d skeletons using a generalized delaunay triangulation technique. *Computer-Aided Design*, 27(9):677–694, September 1995. ISSN 0010-4485. doi: 10.1016/0010-4485(94)00025-9. URL http://www.sciencedirect.com/science/article/pii/0010448594000259. 27, 28

Radu Bogdan Rusu, Zoltan Csaba Marton, Nico Blodow, and Michael Beetz. Persistent point feature histograms for 3d point clouds. In *Proc 10th Int Conf Intel Autonomous Syst (IAS-10), Baden-Baden, Germany*, page 119–128, 2008a. URL http://books.google.co.uk/books?hl=zh-CN&lr=&id=0CgdTrB1AEIC&oi=fnd&pg=PA119&dq=persistent+point+feature+histograms+for+3d+point+clouds&ots=jmo4HpjHmj&sig=txqar9YEjZRlGdIftRMu7Na788U. 45

R.B. Rusu, Z.C. Marton, N. Blodow, and M. Beetz. Learning informative point classes for the acquisition of object model maps. In *10th International Conference on Control, Automation, Robotics and Vision, 2008. ICARCV 2008*, pages 643–650, December 2008b. doi: 10.1109/ICARCV.2008.4795593. 45

Peter Sandilands, Myung Geol Choi, and Taku Komura. Capturing close interactions with objects using a magnetic motion capture system and a RGBD sensor. In Marcelo Kallmann and Kostas Bekris, editors, *Motion in Games*, number 7660 in Lecture Notes in Computer Science, pages 220–231. Springer Berlin Heidelberg, January 2012a. ISBN 978-3-642-34709-2, 978-3-642-34710-8. URL http://link.springer.com/chapter/10.1007/978-3-642-34710-8_21. 79

Peter Sandilands, Myung Geol Choi, and Taku Komura. Edinburgh interaction database, 2012b. URL http://www.ipab.inf.ed.ac.uk/cgvu/InteractionDatabase/interactiondb.html. 79

Paul Scovanner, Saad Ali, and Mubarak Shah. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th international conference on Multimedia*, page 357–360. ACM, 2007. URL http://dl.acm.org/citation.cfm?id=1291311. 11

L. Shapira, S. Shalom, A. Shamir, D. Cohen-Or, and H. Zhang. Contextual part analogies in 3d objects. *International Journal of Computer Vision*, 89(2-3):309–326, September 2010. ISSN 0920-5691, 1573-1405. doi: 10.1007/s11263-009-0279-0. URL http://link.springer.com/10.1007/s11263-009-0279-0. 61

Lior Shapira, Ariel Shamir, and Daniel Cohen-Or. Consistent mesh partitioning and skeletonisation using the shape diameter function. *The Visual Computer*, 24(4):249–259, April 2008. ISSN 0178-2789, 1432-2315. doi: 10.1007/s00371-007-0197-5. URL http://link.springer.com/10.1007/s00371-007-0197-5. xiii, 96

Andrei Sharf, Marina Blumenkrants, Ariel Shamir, and Daniel Cohen-Or. SnapPaste: an interactive technique for easy mesh composition. *The Visual Computer*, 22(9-11):835–844, 2006. URL http://link.springer.com/article/10.1007/s00371-006-0068-5. 22

Damian J. Sheehy, Cecil G. Armstrong, and Desmond J. Robinson. Computing the medial surface of a solid from a domain delaunay triangulation. In *Proceedings of the third ACM symposium on Solid modeling and applications*, page 201–212. ACM, 1995. URL http://dl.acm.org/citation.cfm?id=218062. 28

Evan C. Sherbrooke, Nicholas M. Patrikalakis, and Erik Brisson. Computation of the medial axis transform of 3-d polyhedra. In *Proceedings of the third ACM symposium on Solid modeling and applications*, page 187–200. ACM, 1995. URL http://dl.acm.org/citation.cfm?id=218059. 27

Philip Shilane, Patrick Min, Michael Kazhdan, and Thomas Funkhouser. The princeton shape benchmark. In *Shape modeling applications, 2004. Proceedings*, page 167–178. IEEE, 2004. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1314504. x, 50

Ali Shokoufandeh, Sven J. Dickinson, and Steven W. Zucker. Shock graphs and shape matching. *International Journal of Computer Vision*, 35(1):13–32, 1999. URL http://link.springer.com/article/10.1023/A:1008102926703. 30

Oana Sidi, Oliver van Kaick, Yanir Kleiman, Hao Zhang, and Daniel Cohen-Or. Unsupervised co-segmentation of a set of shapes via descriptor-space spectral clustering. In *ACM Transactions on Graphics (TOG)*, volume 30, page 126. ACM, 2011. URL http://dl.acm.org/citation.cfm?id=2024160. 15

John Andrew Simpson and Edmund SC Weiner. *The Oxford english dictionary*, volume 2. Clarendon Press Oxford, 1989. 19

Stephen P. Smith and Richard Dubes. Stability of a hierarchical clustering. *Pattern Recognition*, 12(3):177–187, 1980. ISSN 0031-3203. doi: 10.1016/0031-3203(80)90042-4. URL http://www.sciencedirect.com/science/article/pii/0031320380900424. 56

O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel. Laplacian surface editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, SGP '04, page 175–184, New York, NY, USA, 2004. ACM. ISBN 3-905673-13-4. doi: 10.1145/1057432.1057456. URL http://doi.acm.org/10.1145/1057432.1057456. 20

Michael Stark, Philipp Lies, Michael Zillich, Jeremy Wyatt, and Bernt Schiele. Functional object class detection based on learned affordance cues. In *Computer Vision Systems*, page 435–444. Springer, 2008. URL http://link.springer.com/chapter/10.1007/978-3-540-79547-6_42. 19

Hari Sundar, Deborah Silver, Nikhil Gagvani, and S. Dickinson. Skeleton based shape matching and retrieval. In *Shape Modeling International, 2003*, page 130–139. IEEE, 2003. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1199609. 13

Jeff K.T. Tang, Jacky C.P. Chan, Howard Leung, and Taku Komura. Interaction retrieval by spacetime proximity graphs. *Comp. Graph. Forum*, 31(2pt2):745–754, May 2012. ISSN 0167-7055. doi: 10.1111/j.1467-8659.2012.03033.x. URL `http://dx.doi.org/10.1111/j.1467-8659.2012.03033.x`. 21

Jean-Marc Thiery, Émilie Guy, and Tamy Boubekeur. Sphere-meshes: shape approximation using spherical quadric error metrics. *ACM Transactions on Graphics (TOG)*, 32(6):178, 2013. URL `http://dl.acm.org/citation.cfm?id=2508363.2508384`. 14

Tony Tung and Francis Schmitt. Augmented reeb graphs for content-based retrieval of 3d mesh models. In *Shape Modeling Applications, 2004. Proceedings*, page 157–166. IEEE, 2004. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1314503`. 13

Oliver van Kaick, Kai Xu, Hao Zhang, Yanzhen Wang, Shuyang Sun, Ariel Shamir, and Daniel Cohen-Or. Co-hierarchical analysis of shape structures. *ACM Trans. Graph.*, 32(4):69:1–69:10, July 2013. ISSN 0730-0301. doi: 10.1145/2461912.2461924. URL `http://doi.acm.org/10.1145/2461912.2461924`. 15

He Wang. Control of objects with a high degree of freedom. 2012. URL `http://www.era.lib.ed.ac.uk/handle/1842/7950`. 95

Yanzhen Wang, Kai Xu, Jun Li, Hao Zhang, Ariel Shamir, Ligang Liu, Zhiquan Cheng, and Yueshan Xiong. Symmetry hierarchy of man-made objects. In *Computer graphics forum*, volume 30, page 287–296. Wiley Online Library, 2011. URL `http://onlinelibrary.wiley.com/doi/10.1111/j.1467-8659.2011.01885.x/full`. 15

Haim J. Wolfson and Isidore Rigoutsos. Geometric hashing: An overview. *Computing in Science and Engineering*, 4(4):10–21, 1997. URL `http://www.computer.org/csdl/mags/cs/1997/04/c4010.pdf`. xv, 100, 135, 137

Kai Xu, Hao Zhang, Daniel Cohen-Or, and Baoquan Chen. Fit and diverse: set evolution for inspiring 3d shape galleries. *ACM Transactions on Graphics (TOG)*, 31(4):57, 2012. URL `http://dl.acm.org/citation.cfm?id=2185553`. 22, 23

Yang Xu and Charles Kemp. Constructing spatial concepts from universal primitives. In *32nd Annual Conference of the Cognitive Science Society. Prize for Computational Modeling in Language*, 2010. URL `http://palm.mindmodeling.org/cogsci2010/papers/0058/paper0058.pdf`. 8

Yuandong Yang, Oliver Brock, and Robert N. Moll. Efficient and robust computation of an approximated medial axis. In *Proceedings of the Ninth ACM Symposium on Solid Modeling and Applications*, SM '04, page 15–24, Aire-la-Ville, Switzerland, Switzerland, 2004. Eurographics Association. ISBN 3-905673-55-X. URL `http://dl.acm.org/citation.cfm?id=1217875.1217879`. 27, 30

Yi-Ting Yeh, Lingfeng Yang, Matthew Watson, Noah D. Goodman, and Pat Hanrahan. Synthesizing open worlds with constraints using locally annealed reversible jump MCMC. *ACM Transactions on Graphics (TOG)*, 31(4):56, 2012. URL `http://dl.acm.org/citation.cfm?id=2185552`. 24

Shin Yoshizawa, Alexander G. Belyaev, and Hans-Peter Seidel. Free-form skeleton-driven mesh deformations. In *Proceedings of the eighth ACM symposium on Solid modeling and applications*, page 247–253. ACM, 2003. URL `http://dl.acm.org/citation.cfm?id=781643`. 31

Lap-Fai Yu, Sai-Kit Yeung, Chi-Keung Tang, Demetri Terzopoulos, Tony F. Chan, and Stanley J. Osher. Make it home: automatic optimization of furniture arrangement. page 1. ACM Press, 2011. ISBN 9781450309431. doi: 10.1145/1964921.1964981. URL `http://portal.acm.org/citation.cfm?doid=1964921.1964981`. 3, 17, 24, 53

Xinhua Yu, John A. Goldak, and Lingxian Dong. Constructing 3-d discrete medial axis. In *Proceedings of the First ACM Symposium on Solid Modeling Foundations and CAD/CAM Applications*, SMA '91, page 481–489, New York, NY, USA, 1991. ACM. ISBN 0-89791-427-9. doi: 10.1145/112515.112582. URL `http://doi.acm.org/10.1145/112515.112582`. 28

T Zaharia and F Preteux. Three-dimensional shape-based retrieval within the MPEG-7 framework. In *Proceedings SPIE Conference on Nonlinear Image Processing and Pattern Analysis XII*, volume 4304, pages 133–145, January 2001. 10

Xi Zhao, He Wang, and Taku Komura. Indexing 3d scenes using the interaction bisector surface. *ACM Trans. Graph.*, 33(3):22:1–22:14, June 2014. ISSN 0730-0301. doi: 10.1145/2574860. URL `http://doi.acm.org/10.1145/2574860`. 5

Bo Zheng, Ryo Ishikawa, Takeshi Oishi, Jun Takamatsu, and Katsushi Ikeuchi. A fast registration method using IP and its application to ultrasound image registration. *IPSJ Transactions on Computer Vision and Applications*, 1:209–219, 2009. URL `http://ci.nii.ac.jp/naid/130000123069/`. 104

Youyi Zheng, Daniel Cohen-Or, and Niloy J. Mitra. Smart variations: Functional substructures for part compatibility. In *Computer Graphics Forum*, volume 32, page 195–204. Wiley Online Library, 2013. URL `http://onlinelibrary.wiley.com/doi/10.1111/cgf.12039/full`. 15, 23