THE UNIVERSITY OF QUEENSLAND

AUSTRALIA

Fast and Accurate Image and Video Analysis on Riemannian Manifolds

Kun Zhao

B.SoftEng., M.Eng.

A thesis submitted for the degree of Doctor of Philosophy at
The University of Queensland in 2016

School of Information Technology and Electrical Engineering

Abstract

Recent times have seen increasing attention on representing images and videos on Riemannian manifolds. Such representations offer new means of preserving intrinsic data structures, which Euclidean-based representations often fail to capture. Preserving the intrinsic structure can bring superior benefits in terms of richer representations and robustness to variations. However, to intrinsically operate on the manifold incurs expensive computation complexity because of using non-linear operators. Furthermore, it is difficult to extend the existing computer vision techniques which were originally developed in Euclidean space into the manifold. To address these issues, recent research often uses extrinsic approaches such as tangent space projection and kernelised approaches. Unfortunately, these methods are not free of drawbacks in terms of low accuracy and high computational load. To that end, this research studies approaches for analysing manifold features which achieve superior performance from the manifold structures whilst computational complexity can be massively reduced. This thesis illustrates the general steps of manifold approaches for image and video analysis, here called manifold scheme, followed by discussions on the possible ways to reduce the computational complexities. Guided by the manifold scheme, this research further proposes two frameworks to analyse manifold features: random projection on Riemannian manifolds and convex hull on Symmetric Positive Definite manifolds. To solve the problems posed by each framework, we present three solutions. Through experiments on several computer vision tasks, we verified that our proposed methods can massively reduce the computational load, while still achieving competitive performance. In addition, our manifold scheme shows another possible way to reduce the computational load of manifold approaches through the generating manifold model using effective lower-level features with low dimensionality. Thus, this thesis proposes a landmark manifold model generated by effective landmark points for facial emotion recognition, which shows competitive performance with much lower computational complexity compared to state-of-the-art manifold methods.

Declaration by author

This thesis is composed of my original work, and contains no material previously published or written by another person except where due reference has been made in the text. I have clearly stated the contribution by others to jointly-authored works that I have included in my thesis.

I have clearly stated the contribution of others to my thesis as a whole, including statistical assistance, survey design, data analysis, significant technical procedures, professional editorial advice, and any other original research work used or reported in my thesis. The content of my thesis is the result of work I have carried out since the commencement of my research higher degree candidature and does not include a substantial part of work that has been submitted to qualify for the award of any other degree or diploma in any university or other tertiary institution. I have clearly stated which parts of my thesis, if any, have been submitted to qualify for another award.

I acknowledge that an electronic copy of my thesis must be lodged with the University Library and, subject to the policy and procedures of The University of Queensland, the thesis be made available for research and study in accordance with the Copyright Act 1968 unless a period of embargo has been approved by the Dean of the Graduate School.

I acknowledge that copyright of all material contained in my thesis resides with the copyright holder(s) of that material. Where appropriate I have obtained copyright permission from the copyright holder to reproduce material in this thesis.

Publications during candidature

Journal Papers:

1. Kun Zhao, Azadeh Alavi, Arnold Wiliem and Brian C. Lovell: **Efficient Clustering on Riemannian Manifolds: A Kernelised Random Projection Approach**: In *Pattern Recognition, Vol. 51, pp. 333–345*, 2016.

Conference Papers:

1. Kun Zhao, Siqi Yang, Arnold Wiliem and Brian C. Lovell: **Landmark Manifold: Revisiting the Riemannian Manifold Approach for Facial Emotion Recognition:** In *IEEE International Conference on Pattern Recognition*, 2016, Accepted.

2. Kun Zhao, Arnold Wiliem, Shaokang Chen and Brian C. Lovell: **Manifold Convex Hull (MACH): Satisfying a Need for SPD**: In *IEEE International Conference on Image Processing*, 2016, Accepted.

3. Liangchen Liu, Arnold Wiliem, Shaokang Chen, Kun Zhao and Brian C. Lovell: **Determining the best attributes for surveillance video keywords generation**: In *IEEE International Conference on Identity, Security and Behaviour Analysis*, 2016.

4. Kun Zhao, Arnold Wiliem and Brian C. Lovell: **Kernelised Orthogonal Random Projection on Grassmann Manifolds with Applications to Gait and Action Recognition**: In *IEEE International Conference on Identity, Security and Behaviour Analysis*, 2015.

5. Azadeh Alavi, Arnold Wiliem, Kun Zhao, Brian C. Lovell and Conrad Sanderson: **Random projections on manifolds of symmetric positive definite matrices for image classification**: In *IEEE Winter Conference on Applications of Computer Vision*, 2014.

Publications included in this thesis

1. Kun Zhao, Arnold Wiliem, Brian. C. Lovell: **Kernelised Orthogonal Random Projection on Grassmann Manifolds with Applications to Gait and Action Recognition**: In *IEEE International Conference on Identity, Security and Behaviour Analysis*, 2015. Incorporated within Chapter 4.

Contributor	Statement of contribution
Kun Zhao (Candidate)	Conception and design of algorithm (70%) Design of experiments (80%) Dataset creation (90%) Paper writing (85%)
Arnold Wiliem	Conception and design of algorithm (30%) Design of experiments (20%) Dataset creation (10%) Paper writing and editing (10%)
Brian C. Lovell	Paper editing and review (5%)

2. Kun Zhao, Azadeh Alavi, Arnold Wiliem, Brian C. Lovell: **Efficient Clustering on Riemannian Manifolds: A Kernelised Random Projection Approach**: In *Pattern Recognition*, Vol. 51, pp. 333–345, 2016. Incorporated within Chapter 4.

Contributor	Statement of contribution
Kun Zhao (Candidate)	Conception and design of algorithm (75%) Design of experiments (80%) Dataset creation (95%) Paper writing (70%)
Azadeh Alavi	Conception and design of algorithm (5%) Paper writing and editing (5%)
Arnold Wiliem	Conception and design of algorithm (20%) Design of experiments (20%) Dataset creation (5%) Paper writing and editing (20%)
Brian C. Lovell	Paper editing and review (5%)

3. Kun Zhao, Arnold Wiliem, Shaokang Chen, Brian C. Lovell: **Manifold Convex Hull (MACH): Satisfying a Need for SPD**: In *IEEE International Conference on Image Processing*, 2016. Accepted. Incorporated within Chapter 5.

Contributor	Statement of contribution
Kun Zhao (Candidate)	Conception and design of algorithm (75%) Design of experiments (90%) Dataset creation (90%) Paper writing (80%)
Arnold Wiliem	Conception and design of algorithm (10%) Design of experiments (10%) Dataset creation (5%) Paper writing and editing (10%)
Shaokang Chen	Conception and design of algorithm (15%) Dataset creation (5%) Paper writing and editing (5%)
Brian C. Lovell	Paper editing and review (5%)

4. Kun Zhao, Siqi Yang, Arnold Wiliem and Brian C. Lovell: **Landmark Manifold: Revisiting the Riemannian Manifold Approach for Facial Emotion Recognition:** In *IEEE International Conference on Pattern Recognition*. Accepted. Incorporated within Chapter 6.

Contributor	Statement of contribution
Kun Zhao (Candidate)	Conception and design of algorithm (70%) Design of experiments (80%) Paper writing (80%)
Siqi Yang	Conception and design of algorithm (5%) Design of experiments (10%) Paper writing and editing (10%)
Arnold Wiliem	Conception and design of algorithm (25%) Design of experiments (10%) Paper writing and editing (5%)
Brian C. Lovell	Paper editing and review (5%)

Contributions by others to the thesis

The work contained in this thesis was carried out by the author under the guidance and supervision of her advisors, Professor Brian C. Lovell and Dr Arnold Wiliem. Part of the work contained in this thesis was carried out by the author in collaboration and discussion with Dr Shaokang Chen.

Statement of parts of the thesis submitted to qualify for the award of another degree

None.

Acknowledgements

It has have been fortunate to work as a PhD student in Security and Surveillance Research Group at The University of Queensland. The decent scene of team spirit, the effective work discipline and environment made my PhD journey much more enjoyable.

I would like to first express the eternal gratitude to my primary advisor, Professor Brian C. Lovell for all continual support, help, patience and encouragement throughout my PhD journey.

Many thanks to my associate advisor, Dr Arnold Wiliem. He has been a great mentor. His brilliant ideas, amazing passion and constant support made my research path much lighter.

I would like to thank Dr Shaokang Chen for the useful discussions and help during my PhD candidature.

A special acknowledgement goes to Dr Daniel F. Smith. I am extremely grateful for his time and effort on the review of the draft of my papers and thesis. His invaluable suggestions indeed made my work much better.

I am also grateful to my current team members, Liangchen Liu, Teng Zhang, Johanna Carvajal Gonzalez, Siqi Yang, Binbin Di and Wai Yan Kyaw San, for sharing the laughs, as well as the frustrations. I wish all the continuous success in their PhD study.

Finally and most importantly, I would like to thank my dear family. I would not have achieved the success without their constant love, support, encouragement and confidence in me. I am always indebted to them.

Keywords

Riemannian manifold, random projection, convex hull, emotion recognition

Australian and New Zealand Standard Research Classifications (ANZSRC)

ANZSRC code: 080104, Computer Vision, 50%
ANZSRC code: 010401, Applied Statistics, 20%
ANZSRC code: 170203, Knowledge Representation and Machine Learning 30%

Fields of Research (FoR) Classification

FoR code: 0801, Artificial Intelligence and Image Processing, 60%
FoR code: 0104, Statistics, 40%

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms and Abbreviations

AAM	Active Appearance Model
AIRM	Affine Invariant Riemannian Metric
CFSS	Coarse-to-Fine Shape Searching
CP	Cluster Purity
CSLDS	Compressive Sensing LDS
CLM	Constrained Local Model
DCT	Discrete Cosine Transform
DNLSSA	Non-Linear Stationary Subspace Analysis
CHISD	Convex Hull based Image Set Distance
Geo-NN	Geodesic Nearest Neighbour
HOG	Histogram of Oriented Gradients
HPE	Histogram Plus Epitome
KGRP	Kernelised Gaussian Random Projection
KORP	Kernelised Orthonormal Random Projection
KPCA-RP	Kernelised Principal Component Analysis Random Projection
KLSH	Kernelised Locality-Sensitive Hashing
KSVM	Kernel Support Vector Machine
LBP	Local Binary Patterns
LDA	Linear Discriminant Analysis
LDS	Linear Dynamical Systems model
LE	Log-Euclidean
LogE-SR	Log-Euclidean Sparse Representation
LSH	Locality Sensitive Hashing
MACH	Manifold Convex Hull

MACH	Manifold Convex Hull
NMI	Normalized Mutual Information
PCA	Principal Component Analysis
PLS	Partial Least Squares
RKHS	Reproducing Kernel Hilbert Space
RLPP	Riemannian Locality Preserving Projection
RI	Rand Index
ROSE	Random projection On SPD manifold for Image Classification
SANP	Sparse Approximated Nearest Points
SD	Stein Divergence
SDALF	Symmetry-Driven Accumulation of Local Features
SDM	Supervised Descent Method
SIFT	Scale-Invariant Feature Transform
SOA	Spatio-temporal Orientation Analysis
SPD	Symmetric Positive Definite
SPD-Sc	Sparse coding for Symmetric Positive Definite points
SRC	Sparse Representation Classifier
SVD	Singular Value Decomposition
SVM	Support Vector Machine
TSC	Tensor Sparse Coding

Mathematical Notation

$\boldsymbol{x}$	a vector (lower-case)
$\boldsymbol{A}$	a matrix (upper-case)
$\mathcal{M}$	a manifold (upper-case, calligraphy typeface)
Sym_d^+	space of $d \times d$ dimensional symmetric positive definite matrices
$\mathcal{G}_{d,p}$	a $d \times p$ dimensional Grassmann manifold
ϕ	homeomorphism
d_g	geodesic distance
d_{LE}	Log-Euclidean distance
$\mathcal{N}(\mu, \boldsymbol{\Sigma})$	a gaussian distribution with μ as the mean and $\boldsymbol{\Sigma}$ as the covariance matrix
$\boldsymbol{A}^T$	transpose of matrix $\boldsymbol{A}$
$\boldsymbol{A}^{-1}$	inverse of matrix $\boldsymbol{A}$
$\lvert\boldsymbol{A}\rvert$	determinant of matrix $\boldsymbol{A}$
$\{\boldsymbol{X}_i\}_{i=1}^n$	set of n elements: $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n\}$

Chapter 1

Introduction

> Learn from yesterday, live for today, hope for tomorrow. The important thing is not to stop questioning.
>
> Albert Einstein

Chapter Summary: *There has been an increasing interest in representing images and videos with the consideration of their critical natural structures that usually lie on the manifold. Unfortunately, current manifold approaches in computer vision suffer from some drawbacks in terms of accuracy loss and computational burden. Exploiting fast and accurate approaches to process the data lying in the manifold, will benefit various computer vision applications.*

1.1 Overview

Discriminative representations that capture the critical structure of data form the basis of many algorithms in computer vision. In recent years, reformulating vision data (images/videos) over Riemannian manifolds has been growing in importance due to its effectiveness in various computer vision applications. In this thesis, we look into efficient manifold approaches for image and video analysis. This chapter first gives a brief introduction to the manifold. Then we introduce our research problem and its significance. Furthermore, the research questions and objectives are illustrated in this chapter. Finally, the contributions and thesis outline are summarised.

1.2 Introduction to Manifolds

The computer vision community has witnessed increasing interest in embedding the manifold geometry in the development of vision frameworks. This is mainly due to the fact that despite the large size of vision data (images and videos), the critical structure of the data actually lies on much lower-

dimensional manifolds [124]. Finding the low-dimensional structure can massively simplify these high-dimensional vision data and avoid the curse of dimensionality [41].

The manifold is a low-dimensional space embedded in a higher Euclidean space [138]. For example, given a set of face images of one person taken with a moving camera under the same conditions, one could represent each face image using $n \times m$ pixel values, resulting in a data point in $\mathbb{R}^{nm}$. However, the intrinsic dimensionality of this image set space is simply the number of different orientations of the camera. Thus, the intrinsic structure of this image set actually is a curve embedded in $\mathbb{R}^{nm}$. The curve is an example of a one-dimensional manifold. If other conditions such as lighting and scaling, are also changing, the intrinsic dimensionality of the set of images would increase. This creates a d-dimensional manifold structure embedded in $\mathbb{R}^{nm}$, where $d \ll nm$. [124].

The manifold structure has been successfully utilised in the manifold learning area, which focuses on non-linear dimensionality reduction techniques that seek a low dimensional representation of a set of high-dimensional points [45, 93]. They assume that the structure of the underlying manifold is unknown. Contrary to this, recently, manifold approaches that assume that the underlying geometry is known have attracted increasing interests in the computer vision community [27, 109, 114, 144]. This thesis focuses on this latter research that is based on the known manifold geometry.

In the context of manifold with known geometry, the vision data is represented by matrices, which are usually informative, robust to affine transformations and illumination changes [114, 144]. Therefore, a variety of computer vision applications have enjoyed tremendous success by the use of manifold features (*e.g.,* dynamic textures [27], human action recognition [56], face recognition [109], visual tracking [114] and pedestrian detection [144]). Furthermore, the application of manifold features is not limited to computer vision. For instance, other areas such as speech recognition [164] and Magnetic Resonance Imaging (MRI) [88] are also found to benefit from manifold features.

1.3 Research Problem and Significance

One of the key properties of the manifold is its non-linearity. The distances between data points are not straight lines, but a curved geodesic. Thus, even the simplest Euclidean operations such as addition and subtraction are may not be valid in the manifold. To develop the statistical tools, the manifold is usually endowed with a Riemannian metric in its tangent space, which leads to the concept of Riemannian manifolds [111]. The Riemannian metric enables one to define the geometric concepts such as distances and angles. Riemannian manifold approaches have been proven to have superior accuracy in computer vision [109, 114, 144]. When it comes to the practical applications, where huge amounts of visual data such as videos and images are uploaded every second [21], it is a necessity to make the manifold approaches possess low computational complexities.

Unfortunately, the computations of the Riemannian metric tend to be extremely high, especially when the manifold dimensionality is large. Also, since the Riemannian metric only operates in a tangent space, often methods employing the metric require multiple iterations to compute their solutions. One such example is a method to compute mean of a distribution of points. Obviously, this

further increases computational load. Furthermore, since the Riemannian manifolds is not a Euclidean space in general, many popular Euclidean-based learning algorithms including Support Vector Machines (SVM), Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) cannot be applied directly on the Riemannian manifolds.

Some existing proposals address these issues either by flattening the manifold structure by projecting the manifold points onto a tangent space [166], or by embedding the manifold points onto the Reproducing Kernel Hilbert Space (RKHS) and then using the kernelised learning algorithms [76]. Here, we call this group of methods as extrinsic methods.

However, manifold topological structure will be distorted when projecting manifold points to a tangent space, as only the distances between the origin of the tangent space and other points are well preserved. Alternatively, learning algorithms in RKHS usually can achieve superior performance, however, they are generally computationally demanding especially when the cardinality of the data is extremely large. Therefore, when developing the mapping function for extrinsic methods, it is critical to find a projection space that can preserve manifold structure, but avoid the use of kernelised learning algorithms. This provides a direction to develop more efficient manifold approaches.

One popular machine learning approach to process manifold data is the Geodesic Nearest Neighbour due to its simplicity. In principal, the Geodesic Nearest Neighbour is fairly similar to the Nearest Neighbour method in Euclidean space. Despite its simplicity and efficacy, the Geodesic Nearest Neighbour is suffered from issues faced by the Euclidean Nearest Neighbour. In general, Nearest Neighbour approaches are sensitive to noise and more inferior to the other more complex approaches such as the Support Vector Machine and nearest convex-hull approach [115]. Thus, if one could generalise the advanced Euclidean-based classifiers to Riemannian manifolds, the performance of the Geodesic Nearest Neighbour would be improved. However, it is vital to devise efficient solutions to the manifold classifiers, as high computational complexities will make the classifiers impractical to use.

Additionally, to represent the vision data as a point in a manifold, one usually computes the covariance matrix or the singular value decomposition of a group of lower-level features. We note that, to obtain superior performance, previous work tends to use high-dimensional lower-level features to generate the manifold points. For example, computing the covariance matrices of the Histogram of Oriented Gradients (HOG) features of video frames [97] results in high-dimensional manifold representations. However, the computational complexity of the manifold approaches is highly related to the dimensionality of the manifold. Currently, it is still challenging to find more effective lower-level features with low dimensionality to generate the manifold points.

Without suitable solutions, the computational issues will hinder the applications of Riemannian manifold approaches for computer vision applications which demand real-time performance. Consequently, this thesis focuses on the research questions listed in the following section. We believe that solving these research questions will fuel the application of manifold approaches in computer vision.

1.3.1 Research Questions

In the light of previous discussions, we list the research questions as follows:

Q1: Is it possible to devise an extrinsic mapping function that projects manifold data points into a Euclidean space wherein computational complexity is considerably reduced? If so, can the manifold structure be well-preserved in the newly created space produced by such a mapping function?

Q2: Is it possible to improve the performance and computational time of the Geodesic Nearest Neighbour by utilising intrinsic classifiers on manifolds?

Q3: Can we reduce the dimensionality of the manifold points by finding discriminative lower-level features with low dimensionality to generate the manifold model?

1.3.2 Aim and Objectives

The main aim of this thesis is to develop fast and accurate Riemannian manifold approaches for image and video analysis. The specific objectives of this thesis are listed as follows:

- Formulate a mapping function for the manifold valued data: this mapping function should project manifold points onto a Euclidean space wherein the computational complexity is considerably reduced and similar performance to the state-of-the-art kernelised learning algorithms can be achieved.
- Formulate classifiers on manifolds to improve the performance of the Geodesic Nearest Neighbour by use of the intrinsic manifold structures, and providing solutions to reduce the computational complexities of this formulation.
- Find effective lower-level features with low dimensionality to generate the manifold representations for the specific computer vision application. This will produce a low-dimensional manifold and in turn lead to low computational load.

1.4 Contributions

This thesis provides a number of significant contributions. We present the general steps of manifold approaches for image and video analysis, here called manifold scheme. Based on this scheme, we discuss several possible ways to reduce the computational cost without obvious accuracy loss when analysing the manifold features. There are three main components contributing to this discussion:

- Component-1: Developing geometric-aware mapping functions for extrinsic methods;
- Component-2: Exploring effective intrinsic methods on manifolds with fast approximated solutions;
- Component-3: Selecting discriminative lower-level features with low dimensionality to produce the more effective-yet-efficient manifold representations.

Following the above three components that are in the consideration of reducing the computational cost of manifold approaches, we furthermore propose two novel frameworks in Chapter 4 and Chapter 5 relating to Component-1 and Component-2. Additionally, in Chapter 6, we propose an efficient model, called the landmark manifold model, for facial emotion recognition. This work uses the simple low-dimensional facial landmark locations to generate the manifold representations. This is an application-specific study for Component-3.

The first framework, called the random projection framework, for clustering purpose, significantly reduces the computational complexity, but still maintains excellent performance.

The contributions with regards to the random project framework are listed as follows:

- Proposing a kernelised random projection framework, which can map manifold points onto a Euclidean space. In general, the term projection is not well defined in Riemannian manifolds. Therefore, we address this via the Reproducing Kernel Hilbert Space (RKHS) constructed from a small subset of data;

- From our framework, it becomes clear that random hyperplane generation is essential. We first adapt the random projection method proposed in our previous work [5] to this framework and rename it as Kernelised Gaussian Random Projection (KGRP). Also, we further develop two projection hyperplane generation algorithms which also follow in our framework: Kernelised Orthonormal Random Projection (KORP) and Kernel Principal Component Analysis Random Projection (KPCA-RP);

- Evaluating our random projection framework on several clustering tasks in computer vision. It is shown that the random projection framework maintains the excellent performance whilst massively reducing computational complexity by over two orders of magnitude in some cases.

- As the random projection framework are not restricted to clustering tasks, we additionally evaluate the performance on two classification tasks: Action and Gait-based Gender Recognition. Comparisons with kernelised classifiers show that the proposed random projection framework achieve nearly 3-fold speed up on average whilst maintaining the accuracy.

The second framework, Manifold Convex Hull (MACH), is a generalisation of the Euclidean nearest convex hull classifier to the manifold. Specifically, the contributions of this framework are listed as follows:

- Proposing a novel mathematical framework, here called MACH, to solve classification tasks on manifolds. Especially, we define the convex combinations and the nearest convex hull distance over the manifold;

- Proposing a solution to the optimisation problem of the framework, MACH-1, which preserves the intrinsic manifold structure by the use of Affine Invariant Riemannian Metric (AIRM). To reduce the computational complexity, two approximated solutions are provided: MACH-2 and MACH-3;

- Evaluating the proposed MACH on several computer vision applications, which shows that MACH significantly outperforms the other intrinsic classifiers and has competitive performance to the-state-of-the-art methods.

Additionally, we revisit the manifold approaches for emotion recognition. We found that using the simple landmark locations as features can establish an efficient manifold model for emotion recognition. The contributions regarding the landmark manifold include:

- Proposing landmark manifold that models the extracted facial landmark points on one single Riemannian manifold. The landmark manifold model is computationally efficient, while achieving competitive performance with the state-of-art manifold methods for facial emotion recognition;
- Investigating the effects of two popular facial landmark tracking methods on the proposed landmark manifold;
- Proposing a fusion method to improve the performance of manifold model based on the estimated landmarks.

1.5 Outline

The reminder of this thesis is as follows. Chapter 2 provides the overview of the background theory and related works. Chapter 3 illustrates the general steps of Riemannian manifold approaches for image and video analysis. Also, we discuss some possible ways to reducing the computational load for Riemannian manifold approaches. Chapter 4 presents the random projection framework followed by the three hyperplane generation methods. Chapter 5 elaborates the convex hull framework and provides three optimisation methods to solve the nearest convex hull classification problem. Chapter 6 presents the landmark manifold approach for facial emotion recognition. The three chapters 4, 5 and 6 represent the main work of this thesis and they include the relevant research problem, proposed approaches, relevant experimental results and discussions. The conclusion and possible future directions are presented in Chapter 7.

Figure 1.1 shows the flow chart that illustrated the connection between the chapters in this thesis.

- **Chapter 2: Background Theory and Literature Review.** This chapter first provides a brief introduction of the relevant theory used in this thesis. Specially, it includes the definitions and explanations of manifold geometry. Also it elaborates the geodesic distances and kernel functions for two popular types of manifolds- Symmetric Definite Positive manifold and Grassmann manifolds. Then, this chapter reviews the current progress of Riemannian manifolds in computer vision.
- **Chapter 3: Manifold Scheme for Image and Video Analysis**. This chapter illustrate the general steps of manifold approaches for image and video analysis, here called manifold scheme.

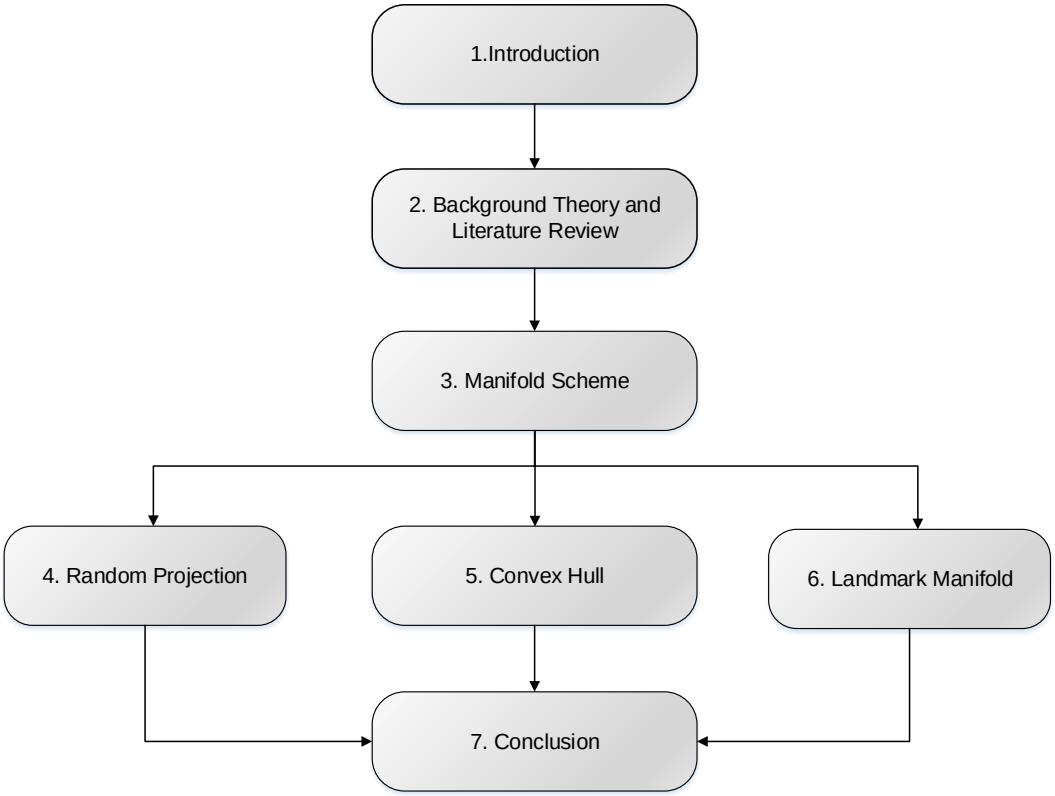


Figure 1.1: Flow chart illustrating the connection between the chapters in this thesis.

Based on this scheme, we discuss some possible ways to reduce the computational complexity when developing the manifold approaches.

- **Chapter 4: Random Projection on Riemannian Manifolds.** This chapter proposes the random projection framework to tackle the high computation complexity problem when performing image or video analysis on Riemannian manifolds. Specifically, we propose three different projection generation methods for our framework. Through the evaluation on large data sets, we demonstrate that our proposed framework massively reduces computational complexity by over two orders of magnitude in some cases while still maintaining the competitive accuracy.

- **Chapter 5: Convex Hull on Symmetric Positive Definite (SPD) Manifolds.** In this chapter, we study the convex hull analysis on SPD manifolds. We formalise a framework for the nearest convex hull classifier on SPD manifolds. To solve the optimisation problem in our proposed framework, three different solutions, MACH-1, MACH-2 and MACH-3, are presented. The latter two solutions are approximated methods that are used to speed up the optimisation process. The efficiency of our framework is verified by several computer vision applications.

- **Chapter 6: Landmark Manifold to Recognize Facial Emotions.** In this chapter, we study the Riemannian manifold approaches for facial emotion recognition. We propose to model the landmark locations on one single Riemannian manifold, which shows excellent performance

and low computational complexity compared to the state-of-the-art Riemannian manifold approaches that use multiple kernel functions and manifold models for facial emotion recognition. Specifically, we study the landmark manifold in terms of SPD and Grassmann manifold. Also, we conduct experiments by using two popular landmark estimation methods to evaluate the effects on the proposed landmark manifold. Finally, a fusion method is proposed to further improve the performance of the landmark manifold constructed by the estimated landmark locations.

- **Chapter 7: Conclusions and Further Work.** This chapter concludes the thesis and also provides the possible directions for our future work.

Chapter 2

Background Theory and Literature Review

> Look deep into nature, and then
> you will understand everything better.
>
> Albert Einstein

Chapter Summary: *Understanding the mathematical theory of the manifolds, as well as the current progress of the manifold approaches in computer vision, allows a more detailed observation of the components that contribute to the proposed framework.*

2.1 Overview

This chapter introduces the background theory and related works that provide the foundation for the rest of the thesis. First, the concepts of manifolds are presented, followed by the explanation of the tangent space and Riemannian metrics. Then we introduce two popular types of Riemannian manifolds in computer vision community: Symmetric Positive Definite manifolds and Grassmann manifolds. The recent progress in use of Riemannian manifold geometry in computer vision is also summarised in this chapter.

2.2 The Geometry of Riemannian Manifolds

Intuitively, a manifold, $\mathcal{M}$, is a generalisation of curves and surfaces to higher dimensions [138]. It is locally Euclidean, where every point has a neighbourhood that is homeomorphic to an open subset of $\mathbb{R}^n$. For example, the earth can be regarded as a manifold where each location has a flat map (refer to Figure 2.1). More formally, we can define a topological manifold [138] as follows:

Definition 2.2.1 *A topological manifold of dimension n is a second countable Hausdorff space $\mathcal{M}$ for which each point has a neighbourhood homeomorphic to an open set in $\mathbb{R}^n$.*

Remarks. The space $\mathcal{M}$ is a hausdorff space, if for every $\boldsymbol{X}, \boldsymbol{Y} \in \mathcal{M}$, there are two open subsets $\mathcal{U}_i$, $\mathcal{U}_j \subseteq \mathcal{M}$ with $\mathcal{U}_i \cap \mathcal{U}_j = \emptyset$ where $\boldsymbol{X} \in \mathcal{U}_i$ and $\boldsymbol{Y} \in \mathcal{U}_j$ or there is one open subset $\mathcal{U}_i \subseteq \mathcal{M}$ where

Figure 2.1: One example of manifold.

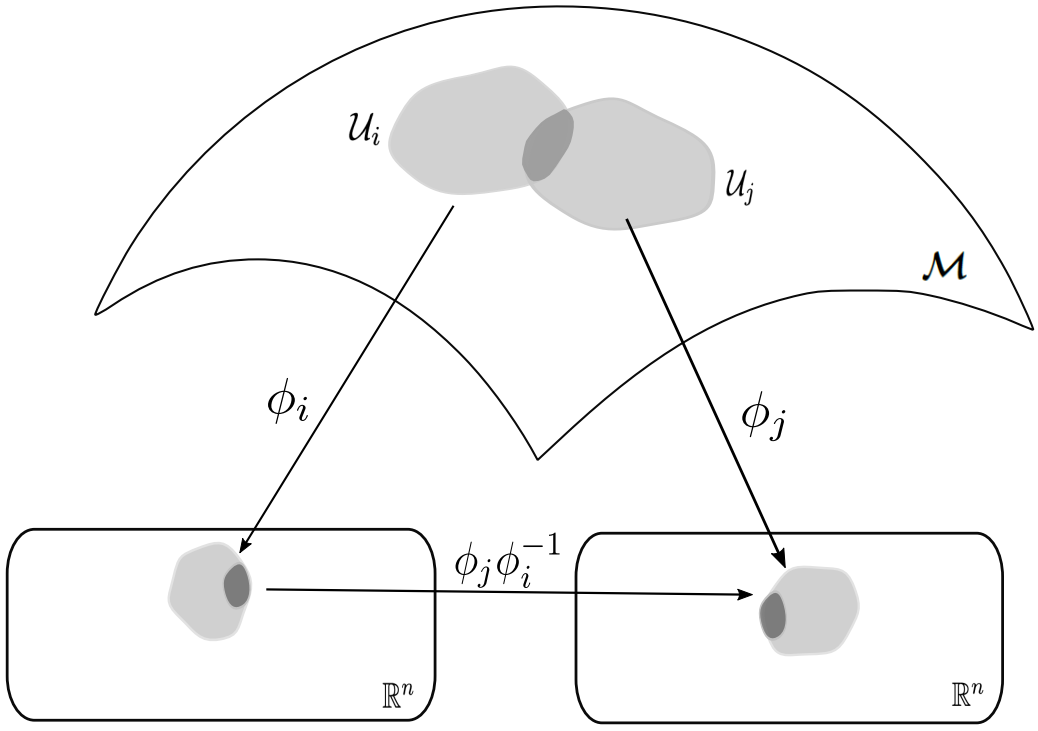


Figure 2.2: Transition maps

$\boldsymbol{X}, \boldsymbol{Y} \in \mathcal{U}_i$. If the open subset $\mathcal{U}_i$ has a countable subset, then $\mathcal{M}$ is second countable. To further explain the concepts of manifolds, we need to review the following related definitions.

Definition 2.2.2 ***Chart, Parameterisation.*** *Given a manifold $\mathcal{M}$ in $\mathbb{R}^n$, for two open subsets $\mathcal{U} \subseteq \mathcal{M}$ and $\mathcal{V} \subseteq \mathbb{R}^n$, there is a homeomorphism $\phi : \mathcal{U} \mapsto \mathcal{V}$ such that $\phi(\mathcal{U}) = \mathcal{V}$. The pair $(\mathcal{U}, \phi)$ is called a chart on $\mathcal{M}$. The inverse map ϕ^{-1} is a parameterisation of $\mathcal{U}$ [138].*

Remarks. Recall that for every point $\boldsymbol{P} \in \mathcal{M}$ there is a neighbourhood $\mathcal{U} \subseteq \mathcal{M}$ and $\phi(\boldsymbol{P}) = (x_1(\boldsymbol{P}), \ldots, x_n(\boldsymbol{P}))$, $x_i(\boldsymbol{P}) \in \mathcal{V}$. We call $x_1(\boldsymbol{P}), \ldots, x_n(\boldsymbol{P})$ the local coordinates of $\boldsymbol{P}$. The coordinates on a chart enable one to apply the Euclidean computations. Intuitively, the chart provides a flattened local map at one point of the manifold.

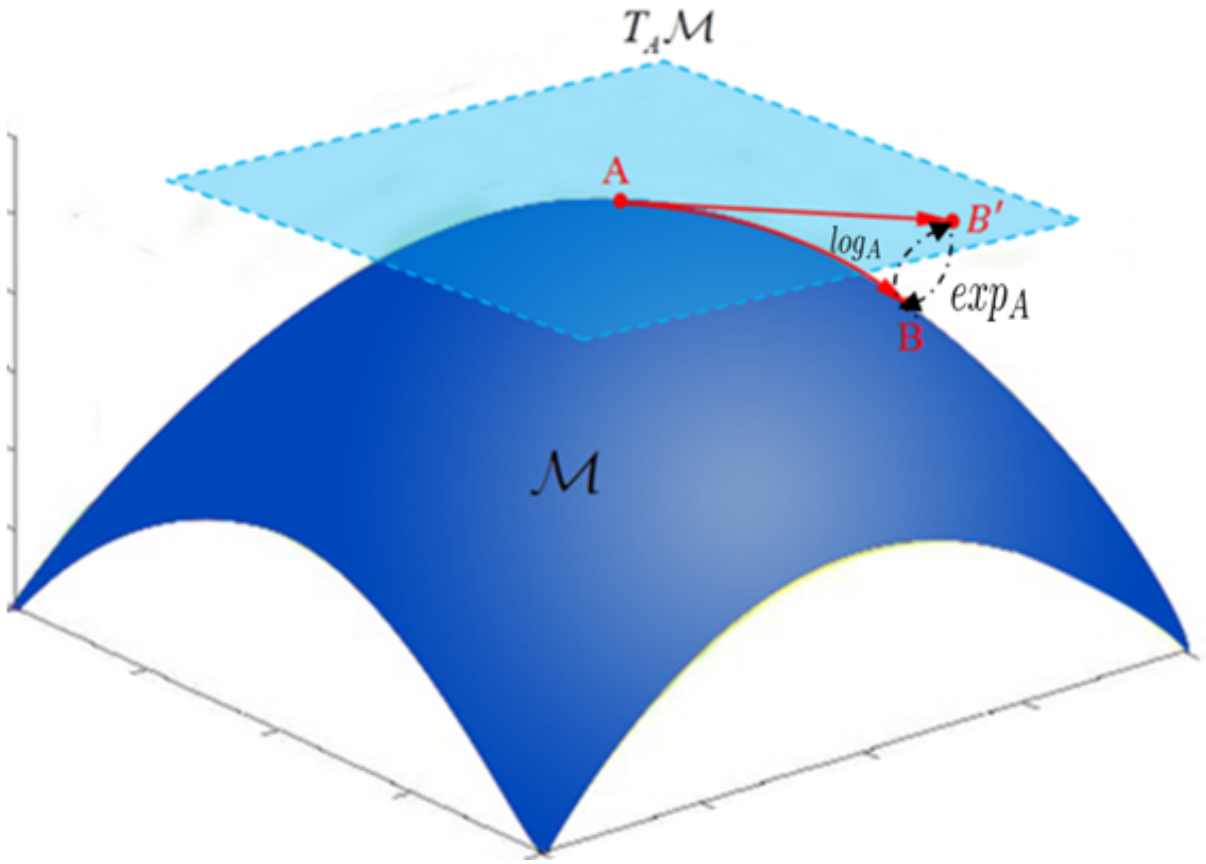


Figure 2.3: Logarithm and exponential mapping

Definition 2.2.3 ***Atlas, Transition Maps.*** *An atlas on $\mathcal{M}$ is a collection of charts $\mho = \{(\mathcal{U}_i, \phi_i)\}$ such that $\mathcal{M} = \bigcup_i \mathcal{U}_i$. The homeomorphisms $\phi_j \phi_i^{-1} : \phi_i(\mathcal{U}_i \cap \mathcal{U}_j) \mapsto \phi_j(\mathcal{U}_i \cap \mathcal{U}_j)$ are the transition maps or coordinate transformations between two charts [138] (Refer to Figure 2.2).*

Remarks. If all transition maps on the manifold are C^∞ differentiable, which means that the infinite continuous partial derivatives for all these maps exist, we term it as a smooth manifold. Equipped with a Riemannian metric endowed on the tangent space, the smooth manifold then becomes a Riemannian manifold [138].

Definition 2.2.4 ***Riemannian metric.*** *A Riemannian metric g on a smooth manifold is an inner product $g_A : T_{\boldsymbol{A}}(\mathcal{M}) \times T_{\boldsymbol{A}}(\mathcal{M}) \mapsto \mathbb{R}$ on each of the tangent spaces $T_{\boldsymbol{A}}(\mathcal{M})$ of $\mathcal{M}$ [138].*

Definition 2.2.5 ***Tangent Space.*** *The tangent space $T_{\boldsymbol{A}}(\mathcal{M})$ is the set of all tangent vectors at the point $\boldsymbol{A}$ constrained to move on the manifold $\mathcal{M}$, where a tangent vector is the derivative of a differentiable curve passing through the point $\boldsymbol{A}$ [138].*

Remarks. Illustrated in Figure 2.3, a tangent space $T_{\boldsymbol{A}}(\mathcal{M})$, can be thought as a plane tangent to the surface of the manifold $\mathcal{M}$ at the point $\boldsymbol{A}$. Note that the tangent space is a vector space that allows all of the Euclidean statistical tools. Therefore, through mapping back and forth between $\mathcal{M}$ and $T_{\boldsymbol{A}}(\mathcal{M})$, one can compute the statistics for any arbitrary point $\boldsymbol{A} \in \mathcal{M}$. Give a curve $\gamma(t)$ on the

manifold, one can compute the instantaneous speed vector $\dot{\gamma}(t)$ and proceed by integrating this value along the curve, then the length of the curve can be computed by [110]:

$$\mathcal{L}_a^b(\gamma) = \int_a^b ||\dot{\gamma}(t)||_{\gamma(t)} dt \ . \tag{2.1}$$

Definition 2.2.6 ***Geodesic Distance.*** The geodesic distance from point $\boldsymbol{A}$ to $\boldsymbol{B}$ on the manifold is the length of the shortest curve between the two points, which can be formulated by the following equation [110]:

$$\text{dist}_{\mathcal{M}}(\boldsymbol{A}, \boldsymbol{B}) = \min_{\gamma} \mathcal{L}(\gamma), \gamma(0) = \boldsymbol{A}, \gamma(1) = \boldsymbol{B} \ . \tag{2.2}$$

However, the geodesic distance derived from Riemannian metric are not always available in closed form, so one can also adopt other distance functions which are available in closed form to compute the distances. We will show some examples of the distance functions in Section 2.2.1 and Section 2.2.2. As shown in Figure 2.3, the mapping from $T_{\boldsymbol{A}}(\mathcal{M})$ to $\mathcal{M}$ is called exponential map. It maps each vector a to the point of the manifold reached in a unit time [110]:

$$\exp_{\boldsymbol{A}} : T_{\boldsymbol{A}}(\mathcal{M}) \mapsto \mathcal{M}, b \mapsto \exp_{\boldsymbol{A}}(b) = \gamma_{(\boldsymbol{A},b)}(1) \ . \tag{2.3}$$

This function defines a local diffeomorphism from a sufficiently small neighbourhood of 0 in the tangent space $T_{\boldsymbol{A}}(\mathcal{M})$ into a neighbourhood of the point $\boldsymbol{A} \in \mathcal{M}$. The inverse map of $\exp_{\boldsymbol{A}}$ is called logarithm map:

$$\log_{\boldsymbol{A}} = \exp_{\boldsymbol{A}}^{(-1)} : \mathcal{M} \mapsto T_{\boldsymbol{A}}(\mathcal{M}), \boldsymbol{B} \mapsto \log_{\boldsymbol{A}}(\boldsymbol{B}) \ , \tag{2.4}$$

which projects the points from the $T_{\boldsymbol{A}}(\mathcal{M})$ to the manifold $\mathcal{M}$.

From now on, a manifold in this thesis will refer to a Riemannian manifold. Recently, two manifolds have been used in the computer vision applications. The Symmetric Positive Definite (SPD) matrices endowed with a Riemannian metric form a SPD manifold (*e.g.,* covariance region descriptors and diffusion tensors). Linear subspaces of a Euclidean space, which form another type of Riemannian manifold named the Grassmann manifold, are commonly used to model image sets and video frames.

The statistical analysis of manifold-valued data has been explored by Pennec *et al.* [10, 110, 111], Absil *et al.* [3] and Turaga *et al.* [139]. In the following sections, we give a brief introduction of some statistical analysis on SPD and Grassmann manifolds, respectively.

2.2.1 Symmetric Positive Definite Manifolds

The Symmetric Positive Definite (SPD) manifold is one of the most popular Riemannian manifolds in computer vision. One method to model images/videos onto the SPD manifold is to compute the covariance matrix of a group of descriptors extracted from the video/image. The covariance matrix fuses multiple features into one single compact representation that captures the second order statistics

independent of the feature dimensions, and elements represent the correlations between the features. Also, as the computations of covariance involve subtracting the feature mean, the noise can be filtered out. When these descriptors are independent, with an appropriate Riemannian metric, the covariance matrices will induce the SPD manifold structure. In other words, these matrices can be considered as points on the SPD manifolds.

The geodesic distance between two SPD points is defined as the length of the shortest curve on the manifold. A widely used distance function is Affine Invariant Riemannian Metric (AIRM) [111]:

$$\mathrm{d}_g(\boldsymbol{X}, \boldsymbol{Y}) = ||\log(\boldsymbol{X}^{-\frac{1}{2}}\boldsymbol{Y}\boldsymbol{X}^{-\frac{1}{2}})||_F \ , \boldsymbol{X}, \boldsymbol{Y} \in \boldsymbol{\mathcal{M}} \ . \tag{2.5}$$

We can also use the recently introduced Log-Euclidean (LE) metric [10] and Stein Divergence (SD) [133] to compute the distance between two points on SPD manifolds:

$$\mathrm{d}_{LE}(\boldsymbol{X}, \boldsymbol{Y}) = ||\log \boldsymbol{X} - \log \boldsymbol{Y}||_F \ , \boldsymbol{X}, \boldsymbol{Y} \in \boldsymbol{\mathcal{M}} \ . \tag{2.6}$$

$$\mathrm{d}_{SD}(\boldsymbol{X}, \boldsymbol{Y}) = \log\left(\det\left(\frac{\boldsymbol{X}+\boldsymbol{Y}}{2}\right)\right) - \frac{1}{2}\log\left(\det\left(\boldsymbol{X}\boldsymbol{Y}\right)\right) \ . \tag{2.7}$$

The Gaussian kernel with d_{LE} and then can be respectively formulated by:

$$\mathrm{K}_{LED}(\boldsymbol{X}, \boldsymbol{Y}) = \exp(-\beta \cdot ||\log(\boldsymbol{X}) - \log(\boldsymbol{Y})||_F^2) \tag{2.8}$$

and

$$\mathrm{K}_{SD}(\boldsymbol{X}, \boldsymbol{Y}) = \exp(-\beta \cdot \log\left(\det\left(\frac{\boldsymbol{X}+\boldsymbol{Y}}{2}\right)\right) - \frac{1}{2}\log\left(\det\left(\boldsymbol{X}\boldsymbol{Y}\right)\right)) \ . \tag{2.9}$$

Note that, in order to become a Mercer kernel, the Gaussian kernel with SD requires β to be of the form: $\beta \in \left\{\frac{1}{2}, \frac{2}{2}, ..., \frac{d-1}{2}\right\}$. Compared to the distance d_g defined in Eqn. (2.5), the computational cost of d_{LE} defined in Eqn. (2.6) and d_{SD} defined in Eqn. (2.7) is much less. Nevertheless, d_g is more accurate. For further discussions on SPD manifolds, the readers are referred to [10, 110, 111].

2.2.2 Grassmann Manifolds

The Grassmann manifold $\mathcal{G}_{n,d}$ is the set of all the d-dimensional subspaces of the vector space $\mathbb{R}^n$[3]. Each subspace $\mathcal{S} \in \mathcal{G}_{n,d}$ can be denoted by a matrix in $\mathbb{R}^{n\times d}$ whose columns are the orthonormal subspace basis. The geodesic distance between two Grassmann points can be computed using their basis principal angles. More specifically, let $\boldsymbol{X}$, $\boldsymbol{Y}$ be two orthonormal matrices of size $n \times d$. The principal angles, $0 \leq \theta_1 \leq ... \leq \theta_m \leq \pi/2$, between two subspaces $\mathrm{span}(\boldsymbol{X})$ and $\mathrm{span}(\boldsymbol{Y})$ are defined recursively in [3] by:

$$\cos\theta_k = \max_{\boldsymbol{u_k}\in span(X)} \max_{\boldsymbol{v_k}\in span(Y)} \boldsymbol{u_k}^\top \boldsymbol{v_k} \tag{2.10}$$

subject to

$$\begin{aligned}&\boldsymbol{u_k}^\top \boldsymbol{u_k} = 1, \boldsymbol{v_k}^\top \boldsymbol{v_k} = 1,\\ &\boldsymbol{u_k}^\top \boldsymbol{u_i} = 0, \boldsymbol{v_k}^\top \boldsymbol{v_i} = 0, (i = 1, ..., k-1)\ .\end{aligned}$$

The principal angles can be computed from the Single Value Decomposition (SVD) of $\boldsymbol{X}^\top \boldsymbol{Y}$ [3]:

$$\boldsymbol{X}^\top \boldsymbol{Y} = \boldsymbol{U}(\cos\theta)\boldsymbol{V}^\top\ , \tag{2.11}$$

where $\cos\theta = \mathrm{diag}\,(\cos\theta_1, ..., \cos\theta_m)$. The principal angles are related to the geodesic distance by:

$$d_G^2(\boldsymbol{X}, \boldsymbol{Y}) = \sum_i \theta_i^2\ . \tag{2.12}$$

Projection kernel is one of the most popular kernels used over Grassmann manifolds and has been shown to have superior performance with low computation complexity [59, 148]. It can be formulated as:

$$\mathrm{K}(\boldsymbol{X}, \boldsymbol{Y}) = \beta \cdot ||\boldsymbol{X}^\top \boldsymbol{Y}||_F^2\ . \tag{2.13}$$

We refer readers to [3, 139] for further treatment on Grassmann manifolds.

2.3 Literature Review

The Riemannian manifold geometry has been widely used in various areas of science and engineering. For example, in 1910s, Albert Einstein developed the general theory of relativity using the Riemannian manifold theory. In the branch of control system design, many theoretic solutions were rooted in manifold geometry [43, 87, 150]. The employment of Riemannian manifold theory can be also found in other areas such as speech recognition [164] and Magnetic Resonance Imaging [88].

In computer vision, the Riemannian manifold geometry has been used as an important tool to explore the solutions in various computer vision applications such as object tracking, face recognition and action recognition.

One of the most successful applications using SPD manifolds is the region covariance matrices based object tracking [90, 91, 114, 142, 160]. The covariance matrices are first introduced as the object descriptors by Tuzel *et al.* [142]. It has been shown that covariance matrices derived from appearance silhouettes are robust to affine transformations, illumination changes, and camera parameters variations [114, 142]. These enable SPD manifolds to perform well in surveillance applications that include multi-cameras [143].

In the face recognition application, the use of SPD manifolds was explored in several publications. Pang *et al.* [109] represented a face as a set of Gabor features whose covariance matrix was modelled on SPD manifolds. Wang *et al.* proposed Covariance Discriminative Learning that projected SPD points onto the kernel space and then applied Linear Discriminant Analysis (LDA) / Partial Least Squares (PLS) for the face recognition task [151]. Sirvalingam *et al.* [130], Harandi *et al.* [66] and

Cherian *et al.* [31] applied sparse decomposition to reformulate a SPD point as a linear combination of dictionary atoms. There are also some applications such as human cell classification [48], texture classification [31, 32, 66, 72, 76, 129, 142, 161] and action recognition [2, 32, 56, 57, 64, 148, 167] that found improved performance by using of SPD manifolds.

Another school of thought is to model videos or image sets as linear subspaces, which lie on another type of Riemannian manifold, the Grassmann manifold. For example, in the face recognition application, a set of face images of the same person is represented as linear subspaces [15, 60, 58, 68, 63, 74, 77, 92, 98, 102, 139, 152, 153, 163], which can be regarded as a point on the Grassmann manifold. Liu *et al.* proposed a stochastic gradient technique to solve an optimization problem on Grassmann manifolds for face recognition [98]. Lin *et al.* derived a maximum effective information criterion for optimising the projections on Grassmann manifolds for face recognition [92]. Hamm *et al.* described kernelised Linear Discriminant Analysis (LDA) with projection and the Binet–Cauchy metric to perform the face recognition tasks [60, 58]. Wang *et al.* explored kernel functions with the geodesic distance on Grassmann manifolds [152]. Harandi *et al.* utilised graph embedding on Grassmann manifolds for face recognition from videos [68]. Jayasumana *et al.* [77] presented that the kernelised SVM with projection Gaussian kernel on Grassmann manifolds achieved superior performance than other Grassmann manifold methods for face recognition such as Grassmann Discriminant Analysis [60] and Graph-embedding Grassmann Discriminant Analysis [68].

Another popular application that utilized the Grassmann manifold is the action recognition [7, 69, 100, 103, 127, 139, 140, 141, 146]. Turaga *et al.* modelled human activities by Time-Varying Linear Dynamic Systems (TV-LDS) [139]. Then each action is represented by a trajectory over the Grassmann manifold. Harandi *et al.* [69] proposed to represent the spatio-temporal aspect of the action by subspaces of a Grassmann manifold. Then, they embedded this manifold into kernel space to tackle the problem of action classification. More recently, Slama *et al.* represented the 3D skeletal sequence as point on the Grassmann manifold and utilised tangent spaces to address the action recognition task [100]. Similarly, the geometry of Grassmann manifold also were explored in other computer vision applications such as objection recognition [60, 139, 158], gender classification [61, 65], Hand Gesture Recognition [61, 62].

The above computer vision problems usually fall into two main categories: clustering and classification. In the following sections, we review the recent manifold methods to tackle the classification and clustering on manifolds, respectively.

2.3.1 Clustering on Riemannian Manifolds

Clustering analysis is a critical tool for understanding visual data. It is an automated process that groups unlabelled data into subsets (here called clusters) that may express the underlying structure of the data [40, 75]. For instance, significant amounts of visual data such as videos and pictures are uploaded every second [21]. Indeed, this is the case for YouTube where 100 hours of video are uploaded every minute [1]. Although these videos have titles and some additional meta-information,

it is often desirable to automatically group the videos in terms of specific criteria such as visual similarity or detected objects.

Despite the fact that clustering methods have been studied since the $1950s$ [50, 75], applying such methods directly on data represented on Riemannian manifolds is not trivial, as Riemannian manifolds generally do not conform to Euclidean space [111, 139]. To address this, one could use manifold tangent spaces which are locally homeomorphic to Euclidean space [111]. However, this brings another challenge to applying existing clustering algorithms as some general algebraic operations are not well defined [110]. For instance, K-means requires the computation of the mean within a cluster which cannot be computed directly. To this end, Pennec *et al.* [110] reformulated the computation of the mean as a solution to an optimisation problem. Using this formulation, the mean point is considered as the point over the manifold minimising the geodesic distance (*i.e.,* the true distance on the manifold between two points) from the mean point to all other points. The algorithm to solve this problem is called Riemannian centre of mass [80, 110]. Thanks to the Riemannian centre of mass, Turaga *et al.* [139] extended the K-means algorithm into the Riemannian manifold, which is regarded as intrinsic K-means and has been applied to activity-based video clustering. Intrinsic K-means has further demonstrated better performance than Euclidean-based methods (for example, Protein Clustering [134]).

We shall categorise these methods as intrinsic methods as discussed in the above paragraph. Generally, intrinsic methods that completely honour the manifold topology lead to higher accuracy. Unfortunately, the computational cost of intrinsic methods is extremely high since these need to map all of the data to tangent spaces repeatedly.

Extrinsic methods, on the other hand, seek solutions that may not completely consider the manifold topology [48, 49, 65, 76, 78, 166]. The most simplistic way, here called Log Euclidean methods, is to embed all of the points into a designated tangent space at a particular point [9]. Log Euclidean methods can be considered as flattening the manifold. This has been used in various computer vision applications, such as human action recognition [48] and cell classification [166]. This addresses the computational cost issues suffered by the intrinsic methods, as the tangent space is homeomorphic to the Euclidean space and well-known Euclidean clustering approaches such as K-means can be directly applied. Unfortunately, as the flattening step distorts the pair-wise distances in regions far from the origin of the tangent space, accuracy is severely compromised. Much of the value of the manifold approach is lost.

Other approaches that fall in the extrinsic method category are kernel-based approaches [65, 76, 78], such as kernel K-means. In essence, the data in manifold are first embedded into the Reproducing Kernel Hilbert Space (RKHS) [125]. As the embedding function is defined implicitly, generally kernel-based approaches make use of the inner products in the RKHS in their formulation. These inner products are then arranged in a Gram matrix. It is often observed that the right choice of kernel could significantly improve the performance [76]. Furthermore, in general, kernel inner products with specified metrics have much less computational complexity than geodesic distances [5, 127]. With these properties, kernel-based approaches could be suitable to address issues suffered in both the in-

trinsic approach and the Log Euclidean approach. Unfortunately, the kernel-based approaches cannot scale easily, as the Gram matrix computation is $O(n^2)$ where n is the number of data points. Also, it is often quite challenging to kernelise the existing algorithms that do not have known kernelised versions [24]. Furthermore, Nikhil *et al.* demonstrated that clustering data in the RKHS may lead to unexpected results since the clusters obtained in the RKHS may not exhibit the structure of the original data [108].

2.3.2 Classification on Riemannian Manifolds

To solve the classification problem, one needs to employ appropriate classifiers for manifold. As the geometry of Riemannian manifold is non-Euclidean, Euclidean-based classifiers such as Support Vector Machine (SVM) cannot be directly applied.

One line of research is to apply intrinsic methods which directly operate on the manifold. In this scenario, the true manifold geometry will be fully considered and this usually yields superior and robust classification performance. In its simplest form, one could use the nearest neighbour classifier utilising the true geodesic distance [111]. It has been shown that this approach can achieve good accuracy; however, it is generally computationally expensive especially if the size of dataset is huge or the manifold-valued data possesses large dimensions [72]. Additionally, the nearest neighbour classifier is susceptible to outliers in the data [113] and often inferior to other classifiers such as SVM, nearest convex hull and sparse representation classifier.

Recently, some researchers pursued to generalise Euclidean-based classifiers to manifolds. For example, Sivalingam *et al.* proposed tensor sparse coding for Symmetric Positive Definite (SPD) manifolds [129]. Since their method used log-determinant divergence to model the loss function, it suffered from high computational complexity. Also, the manifold structure is not well-preserved by the log-determinant divergence, thus the accuracy is not excellent. Cerian *et al.* proposed Riemannian Sparse Coding for SPD manifolds [31], which used the geodesic distance to preserve the manifold structure. However, the use of the matrix addition forces the summation of the weighted SPD points out of the manifold in some cases.

The computational complexity issue in the intrinsic methods could be addressed by mapping all the data points onto a tangent space at a designated location [48, 61, 136, 144, 166]. Unfortunately, as it has been discussed multiple times, this mapping may adversely affect the performance since it will significantly distort the manifold structure in regions far from the tangent space location.

Another popular school of thought is to map manifold points onto a Reproducing Kernel Hilbert Space (RKHS) and apply kernel-based classifiers [61, 70, 76, 148]. As Euclidean geometry applies in the RKHS, this can be thought of a decoupling between machine learning and data representation [81]. Nevertheless, to choose an appropriate kernel could be challenging. In fact, several works have been devoted to address this [76, 148]. When the kernel is poorly chosen it leads to inferior performance [148]. In addition, kernel methods are generally computationally demanding especially when training data is extremely large [155].

Table 2.1: Summary of the popular methods compared to our proposal.

Approach	Exploits Manifold Structure	Accuracy	Computational Complexity
Intrinsic Methods [31, 110, 139, 134, 161]	Yes	High	High
Log-Euclidean Methods [48, 49, 136, 166]	Minimal	Low	Low
Kernel Methods [65, 70, 76, 78, 148]	Approximately	High	Moderate
The aim of This Thesis	Approximately	High	Low

2.4 Conclusions of Literature Review

From the discussions in Section 2.3, we found that the manifold approaches for the clustering and classification problems can be broadly categorised into two groups: intrinsic and extrinsic methods (*e.g.,* Log-Euclidean methods and Kernel methods). We summarise the advantages and shortcomings of these manifold approaches in Table 2.1.

Although the existing methods give advancement in computer vision, additional work is still needed when dealing with computational complexity issues and the trade-off between approximation and accuracy. For example, intrinsic methods that precisely exploit the manifold structure by using the Riemannian metric usually possess high accuracy. However, the computational complexity is high. Log-Euclidean methods produce a first-order approximation of the manifold structure by the tangent space at a particular point. These methods massively reduce the computational load of the original manifold. However, the accuracy is usually sacrificed. Kernel methods utilise kernel functions that embed the Riemannian manifold points into RKHS. Generally, one can achieve high accuracy. However, the computational burden of kernel calculations and kernel-based learning algorithms is high for large datasets. Our aim in this thesis is to develop fast and accurate manifold approach, which preserves the critical manifold structure, as well as is computationally efficient.

2.5 Summary

In this chapter, we presented the overview of Riemannian geometry. Two types of Riemannian manifolds, SPD manifold and Grassmann manifold were introduced with the definition of distance and kernel functions. In addition, the current progress regarding Riemannian manifold approaches in computer vision was discussed. In the next chapter, the manifold approach scheme for image and video analysis are presented to provide clues to improve the current manifold approaches for image and video analysis.

Chapter 3

Manifold Scheme for Image and Video Analysis

> Everything should be made as simple as possible, but not simpler.
>
> Albert Einstein

Chapter Summary: *The general steps of manifold approaches for image and video analysis are presented, which give clues to the possible ways to reduce the computational complexities of current manifold approaches.*

3.1 Overview

In this chapter, we present the general steps of manifold approaches for image and video analysis, here called the manifold scheme. Based on the components of the scheme, we discuss some possible ways to reduce the computational load for Riemannian manifold approaches, whilst the performance is free from obvious loss.

3.2 Manifold Scheme for Image and Video Analysis

There are some general steps to analyse images and videos using manifold geometry. For the sake of clarity, we call these general steps as the manifold scheme, which is illustrated in Figure 3.1. The first step is to obtain the lower-level features and then model these features into manifolds.

To model vision data to the SPD manifold, the most popular method is to use region covariance [142] (refer to Figure 3.2). Specifically, given an image I, the lower-level feature vectors $[\boldsymbol{z}_1, \boldsymbol{z}_2, ..., \boldsymbol{z}_S]$ from the region of interest of this image are first extracted. For example, the fea-

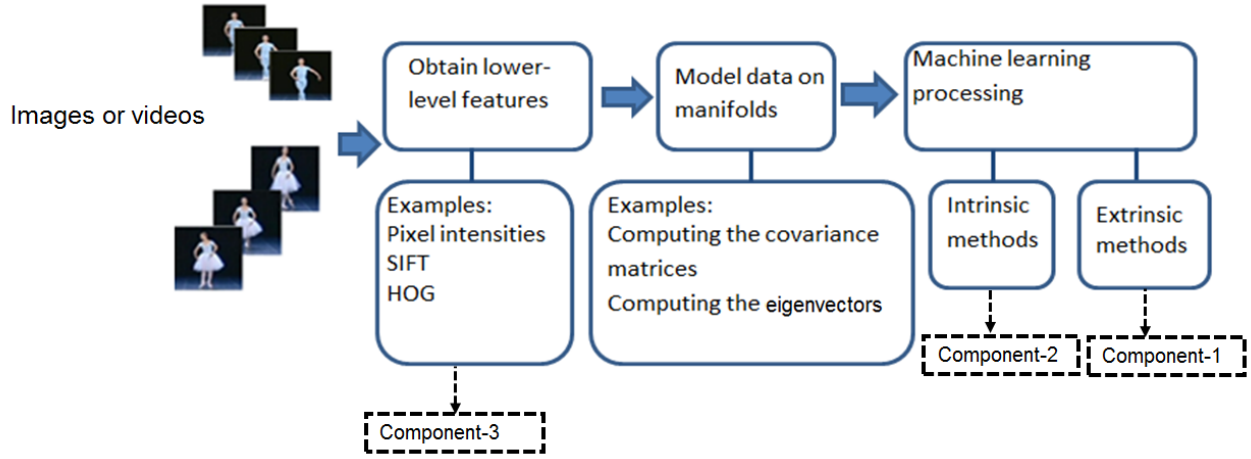


Figure 3.1: Manifold scheme for image and video analysis

ture vector z_i can be associated to image intensities and derivatives inside the region R: $\boldsymbol{z}_i = [I(x,y), |I_x|, |I_y|, |I_{xx}|, |I_{yy}|]$. Then the region covariance C_R can be computed by:

$$\boldsymbol{C}_R = \frac{1}{S-1}\sum_{i=1}^{S}(\boldsymbol{z}_i - \mu)(\boldsymbol{z}_i - \mu)^\top \ , \tag{3.1}$$

where μ is the vector of the means of the corresponding features within the region R. For the image set problem, where each exemplar is represented by a set of images or a single video, the modelling process is illustrated in Figure 3.3. the lower-level feature vector such as pixel intensities, SIFT and HOG for each image or video frame is first extracted. Then the entire image set can be represented as: $F = [\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_n]$, where $f_i \in \mathbb{R}^d$ is the i-th image or video frame with d-dimensional feature description. Then one can model the image set on the SPD manifold by computing the covariance matrix of these lower-level features as follows:

$$\begin{aligned} \boldsymbol{C} &= \frac{1}{n-1}\sum_{i=1}^{n}(\boldsymbol{f}_i - \bar{\boldsymbol{f}})(\boldsymbol{f}_i - \bar{\boldsymbol{f}})^\top \ , \\ \bar{\boldsymbol{f}} &= \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{f}_i \ . \end{aligned} \tag{3.2}$$

Alternatively, one can represent the image set by a linear subspace P via the Singular Value Decomposition method on these lower-level features (refer to Eqn.(3.3)), which is a data point on the Grassmann manifold.

$$\sum_{1}^{n}\boldsymbol{f}_i\boldsymbol{f}_i^\top = \boldsymbol{P}\Lambda\boldsymbol{P}^\top \ . \tag{3.3}$$

After modelling the images or videos on the manifold, one needs to apply some learning algorithms to tackle the computer vision tasks. As discussed in Section 2.3, these algorithms can be broadly categorized by intrinsic methods and extrinsic methods. Recall that the main goal of this thesis is to develop fast and accurate manifold-based approaches for image and video analysis. In the next section, we discuss some possible directions that we can consider to reach our goal.

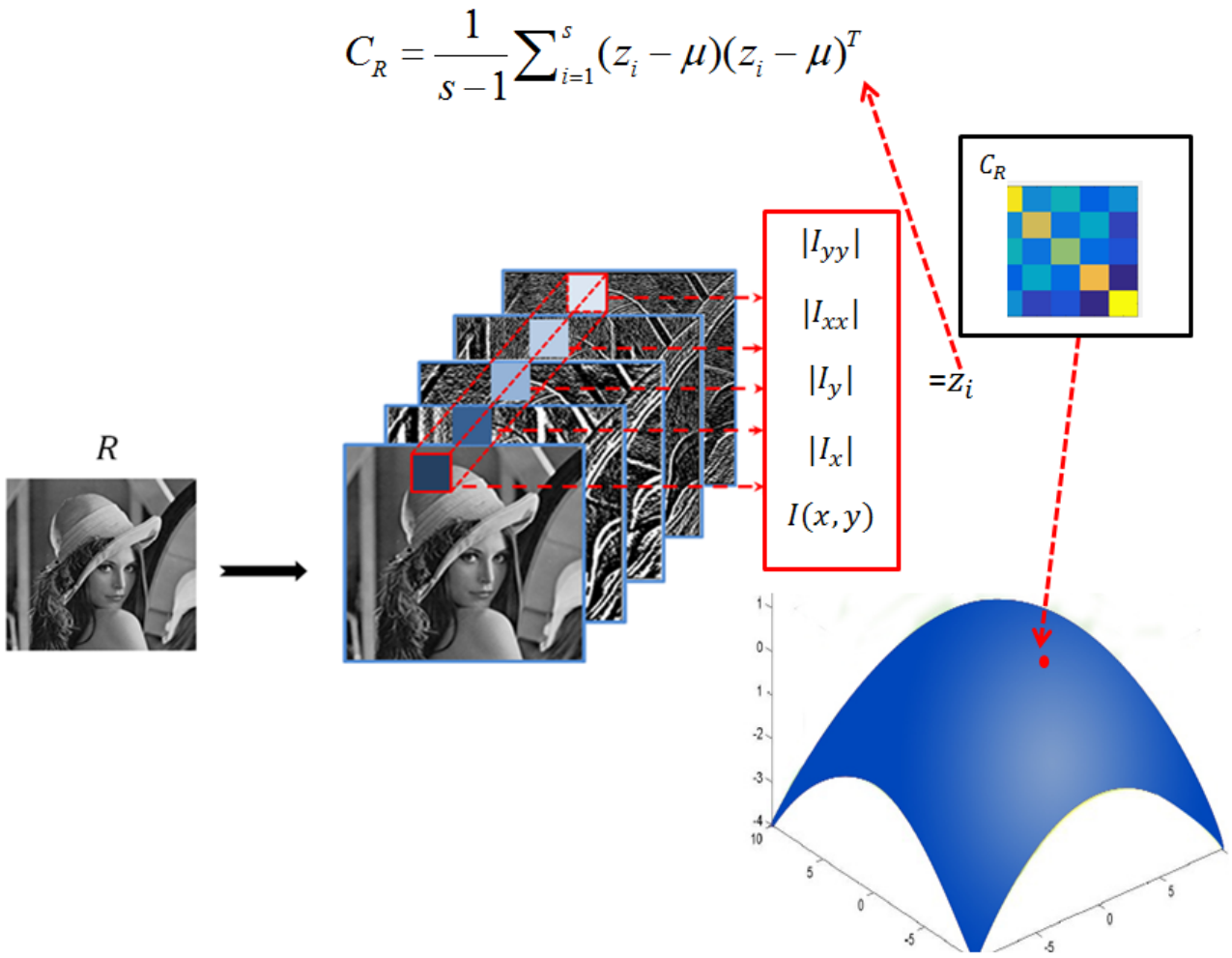


Figure 3.2: An illustration of how to model vision data on SPD manifold by computing the region covariance

3.3 Discussions

Given the scheme illustrated in Figure 3.1, we found that there are many directions for making progress in developing fast and accurate manifold approaches. We give some brief discussions on a few of those directions. There are three main components contributing to this discussion: Component-1:Developing geometric-aware mapping functions for extrinsic methods; Component-2: Exploring effective intrinsic methods on manifolds with fast approximated solutions and Component-3: Selecting discriminative lower-level features with low dimensionality to produce the more effective-yet-efficient manifold representations.

Component-1: Developing geometric-aware mapping functions for extrinsic methods. The easiest way that one can speed up the manifold approaches is to develop a mapping function which can project manifold points onto a Euclidean space whilst preserving the critical manifold structure. Then all the Euclidean based-methods could be used. This will avoid the manifold operators, resulting in much less computational cost. To ensure the excellent performance, the mapping function should be in a form that minimises the loss from the manifold to the Euclidean space. In Chapter 4, we

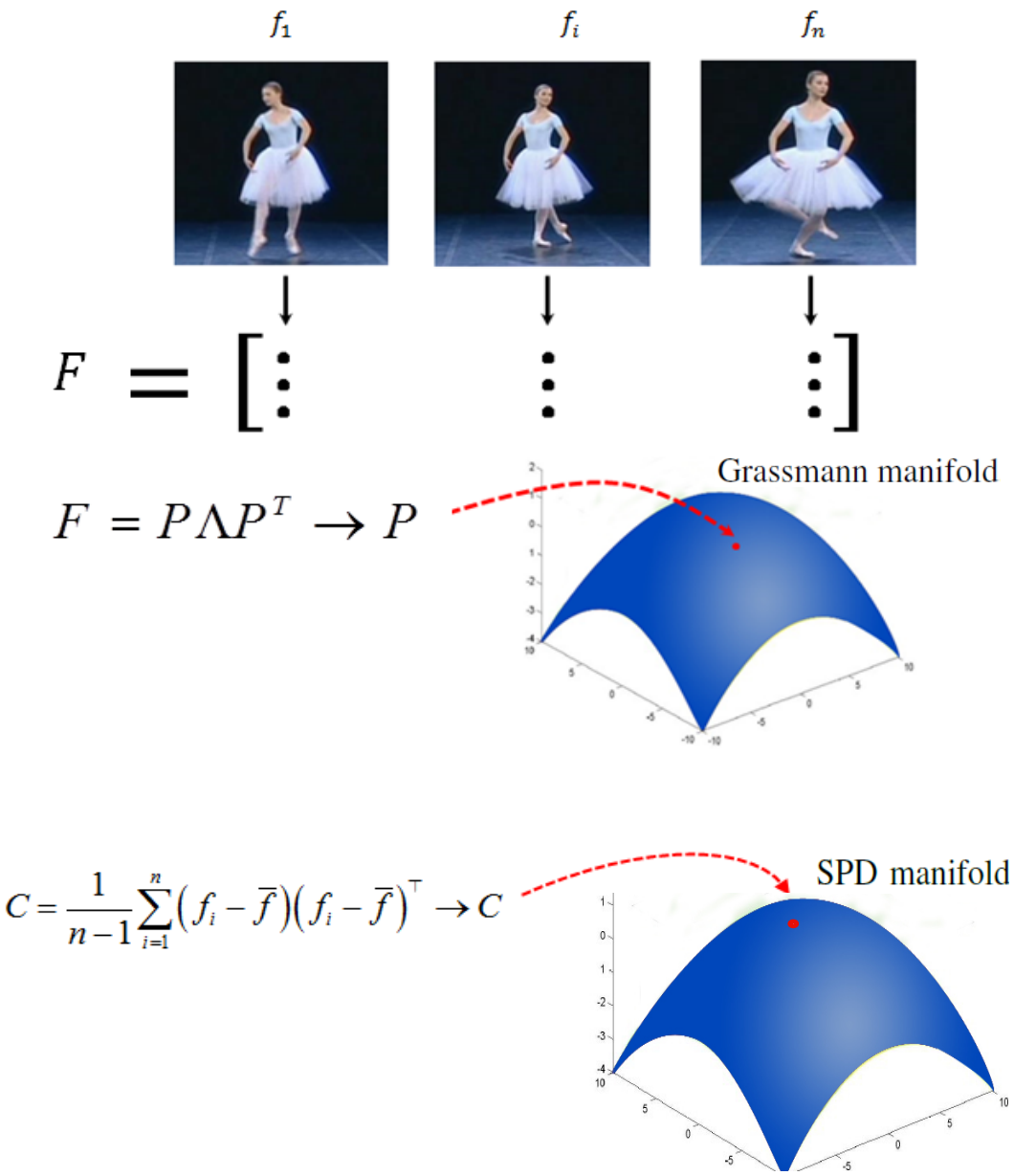


Figure 3.3: An illustration on how to represent a video or an image set using SPD and Grassmann manifolds.

propose a random projection framework with three extrinsic mapping functions that show excellent performance with fast speed in several computer vision applications.

Component-2: Exploring effective intrinsic methods on manifolds with fast approximated solutions. Unlike Component-1, Component-2 focuses on improving the accuracy and computational complexity of specific intrinsic methods, which involve the use of Riemannian metric. As discussed in Section 2.3.2, intrinsic methods tend to have high accuracy as they are using the Riemannian metric. However, due to the nonlinearity of manifold, it is usually non-trivial to find the global solutions for the optimisation problems. Thus, there are limited numbers of intrinsic methods have been developed, especially for classification problems. The goal of Component-2 is to explore more effective intrinsic methods to improve existing ones, in terms of accuracy and computational time. Furthermore,

Table 3.1: Theoretical complexity analysis for computations on manifolds.

Computations	Complexity	
	Sym_d^+	$\mathcal{G}_{d,q}$
Computing the covariance matrix	$O(d^2)$	n/a
Computing the SVD	n/a	$O(d^3)$
Exponential map	$O(d^3)$	$O(q^3)$
Logarithm map	$O(d^3)$	$O(q^3)$
Geodesic distance	$O(d^3)$	$O(q^3)$

in this component, we are interested whether it is possible to further improve the intrinsic method computational complexity. We note that the intrinsic methods usually require multiple projections between the manifold and the tangent spaces, which makes them suffer from the highly demanding computational load. One way to alleviate this limitation is to find an approximated solution to the optimisation problem defined in the intrinsic method. The optimality of the approximation may not be guaranteed or perfect in some cases. However, it may be possible to find an approximation that does not significantly sacrifice the method accuracy whilst computationally more efficient.

In this thesis, we focus on intrinsic classifiers on manifolds. One of most widely used intrinsic classifier for manifold points is the Geodesic Nearest Neighbour. More specifically, In Chapter 5, the thesis will study on how to improve the Geodesic Nearest Neighbour classifier by developing a novel intrinsic method. The computational time of the novel intrinsic method will then be sped up by proposing various effective approximation methods.

Component-3: Selecting discriminative lower-level features with low dimensionality to produce the more effective-yet-efficient manifold representations. The first step illustrated in Figure 3.1 will output the d-dimensional feature vectors, which mainly determine the dimensionality of the manifold points. The dimensionality of the manifold points highly impacts the computational cost of the following steps. For example, computing the covariance matrix requires $O(d^2)$ and computing the SVD requires $O(d^3)$ as listed in Table 3.1. Also, the logarithm and exponential mapping often used in the learning process, requires $O(d^3)$ for SPD manifolds. Thus, finding the effective lower-level feature vectors with low dimensionality can obviously reduce the computational load of the following analysis process. In Chapter 6, the thesis discusses an application-based study to show it is possible that the lower-level features benefit the computational complexity of the manifold approach.

3.4 Summary

To explain the process of manifold approaches, we presented a manifold scheme for image and video analysis in this chapter. With the goal of developing fast and accurate manifold approaches for image and video analysis, we further discussed some possible directions to reduce the computational cost while maintaining the excellent accuracy for computer vision applications. In next chapter, we propose the random projection framework following the Component-1 in this scheme.

Chapter 4

Random Projection on Riemannian Manifolds

There are two ways to live:
you can live as if nothing is a miracle;
you can live as if everything is a miracle.

Albert Einstein

Chapter Summary: *Based on the Johnson-Lindenstrauss lemma, we propose the random projection framework, which projects the manifold points onto a Euclidean space by sufficient random projections, wherein the pairwise distances of the manifold points are still well-preserved, but the complexity is significantly reduced.*

4.1 Overview

In this chapter, we first review the definition and generation method of random projection in Euclidean space. We then extend these notions into the Riemannian manifold and propose the random projection framework for manifold points. This framework implements random projections for manifold points via kernel space, which can preserve the geometric structure of the original space, but is computationally efficient. Here, we introduce three methods that follow our framework. We then validate our framework on several clustering and classification tasks problems by comparing against several recent state-of-the-art methods on Riemannian manifolds. Experimental results demonstrate that our framework maintains the performance whilst massively reducing computational complexity by over two orders of magnitude in some cases.

4.2 Introduction

In this chapter, we propose a random projection framework for Riemannian manifolds, which significantly reduces the computational complexity, but still maintains acceptable performance. The

inspirations are drawn from the random projection for Euclidean spaces which has enjoyed success in various domains [16, 55, 84] due to its simplicity and theoretical guarantees [4]. The contributions of this chapter are listed as follows:

1. We propose a random projection framework for manifold features. In general, the term projection is not well defined in Riemannian manifolds. Therefore, we address this via the RKHS constructed from a small subset of data.

2. From our framework, it becomes clear that random hyperplane generation is essential. Thus, we describe three generation algorithms which are followed in our framework: (1) Kernelised Gaussian Random Projection (KGRP); (2) Kernelised Orthonormal Random Projection (KORP) and (3) Kernel Principal Component Analysis Random Projection (KPCA-RP).

4.3 Random Projection in Euclidean Space

In Euclidean space, the random projection embeds original data into a much lower dimensional space whilst preserving the geometric structure [147]. This can significantly reduce the computational complexity of learning algorithms, such as classification or clustering. For instance, as a result, this is used to achieve real time performance in object tracking [119].

A point $\boldsymbol{x} \in \mathbb{R}^d$ in Euclidean space can be projected into a random k-dimensional subspace ($k \ll d$) via a set of randomly generated hyperplanes $\{\boldsymbol{r}_1\}_{i=1}^{k}$ where $\boldsymbol{r}_i \in \mathbb{R}^d$. This can be formulated as:

$$f(x) = \boldsymbol{x}^\top \boldsymbol{R}\,, \tag{4.1}$$

where $\boldsymbol{R}$ is the random matrix that arranges the random hyperplanes as column vectors. Note that in order to minimise distortions produced by the projection, the matrix $\boldsymbol{R}$ should possess a particular property. We introduce this property in Definition 1. When the random projection matrix $\boldsymbol{R}$ possesses such a property, then the Johnson-Lindenstrauss Lemma (JL-Lemma) [79] applies.

Lemma 4.3.1 *[Johnson-Lindenstrauss Lemma [79]] For any ϵ such that $\epsilon > 0$, and any set of points $\mathcal{X}$ with $|\mathcal{X}| = n$ upon projection to a uniform random k-dimension subspace where $k \geq \mathrm{O}(\epsilon^{-2} \log\, n)$, the following property holds for every pair $\boldsymbol{u}, \boldsymbol{v} \in \mathcal{X}$, $(1-\epsilon)||\boldsymbol{u}-\boldsymbol{v}||^2 \leq ||\,\mathrm{f}(\boldsymbol{u}) - \mathrm{f}(\boldsymbol{v})||^2 \leq (1+\epsilon)||\boldsymbol{u}-\boldsymbol{v}||^2$, where $\mathrm{f}(\boldsymbol{u}), \mathrm{f}(\boldsymbol{v})$ are the projections of $\boldsymbol{u}, \boldsymbol{v}$.*

Remarks The JL-Lemma principally states that a set of high dimensional points can be embedded using a set of uniform random hyperplanes into lower dimensional space wherein the pairwise distance between two points is well preserved (with high probability). The original proof of JL-Lemma uses quite challenging geometric approximation machinery [79]. Frankl and Meahara [52] simplified that proof by considering a projection into k random orthonormal vectors. Recently there have been several properties of the random matrix where JL-Lemma still applies. We shall call the type of projection wherein the random matrix has properties that allow the JL-Lemma to be applied as a JL-Type projection.

Definition 1 (JL-Type projection) *Let $\boldsymbol{R} = [\boldsymbol{r}_1 \cdots \boldsymbol{r}_k], \boldsymbol{r}_i \in \mathbb{R}^d$ be a random matrix whose columns are the random hyperplanes. The projection $\mathrm{f}(\boldsymbol{u}) = \boldsymbol{R}^\top \boldsymbol{u}, \boldsymbol{u} \in \mathbb{R}^d, \mathrm{f}(\boldsymbol{u}) \in \mathbb{R}^k$ is called JL-Type projection when the matrix $\boldsymbol{R}$ possesses at least one of the following properties:*

1. *The columns of $\boldsymbol{R}$ are orthogonal unit-length vectors [52];*
2. *Each element in $\boldsymbol{R}$ is selected independently from a standard Gaussian distribution $N(0, 1)$ or uniform distribution $U(-1, 1)$ [8];*
3. *$\boldsymbol{R}$ is a sparse matrix, whose elements belong to $\{-1, 0, +1\}$ with the probability $\{1/6, 2/3, 1/6\}$ [89].*

We note that Property 1 in Definition 1 considers columns of the random matrix $\boldsymbol{R}$ as the basis of a random space, thus they are required to be pairwise orthogonal [52]. To this end, one needs to apply an orthogonalisation technique such as the Gram-Schmidt method [157] on $\boldsymbol{R}$. Arriaga *et al.* [8] proved that it suffices to use random non-orthonormal matrices with independent elements chosen from some distributions which are listed in Property 2 of Definition 1. Recently, Li *et al.* [89] proposed a sparse random projection matrix presented in Property 3 of Definition 1. The sparse random projection achieves a further threefold speed-up as only $1/3$ of the matrix have non-zero elements.

We note that the random projection is not data driven. It means that it does not need a set of labelled training data, making it suitable for unsupervised learning scenarios such as clustering [20, 118].

4.4 Random Projection in Riemannian Manifolds

To apply the random projection on points residing in the Riemannian manifold is not trivial, due to the notion of projection itself being generally not well defined. We approach this problem by reformulating the problem in the RKHS. Recall that, the random matrix containing column vector of hyperplanes $\boldsymbol{r}_i$ should be generated from a particular process. Thus, the projection of each individual dimension into the projected space is carried out as follows:

$$f_i(\boldsymbol{x}) = \boldsymbol{x}^\top \boldsymbol{r}_i \,, \tag{4.2}$$

where $f_i(\cdot)$ is the i-th dimension of the projected vector $\boldsymbol{x}$.

In the RKHS, the above formulation can be rewritten as:

$$f_i(\boldsymbol{x}) = \phi(\boldsymbol{x})^\top \boldsymbol{r}_i \,, \tag{4.3}$$

where $\phi(\cdot)$ is the function that embeds the input space into the RKHS. Note that, in this case, the hyperplane $\boldsymbol{r}_i$ is now defined in the RKHS, $\boldsymbol{r}_i \in \mathcal{H}$. The projection in the RKHS can be considered as the inner product which is defined as the kernel similarity function.

Eqn. (4.3) provides insight that the JL-Type projection could be achieved as long as one could generate the hyperplanes that follow one of the above properties in Definition 1 in the RKHS. In

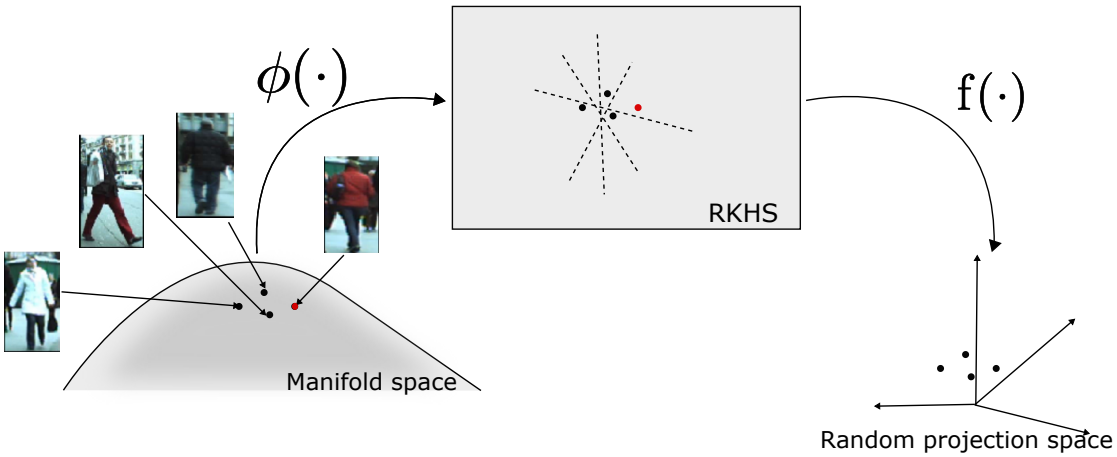


Figure 4.1: The illustration of our proposed framework. We first generate the hyperplanes in RKHS. Each point in the manifold is then mapped into the projected space via the kernel inner product.

similar fashion, when the data point $\boldsymbol{x}$ is replaced by a point $\boldsymbol{X}$ in manifold $\boldsymbol{X} \in \boldsymbol{\mathcal{M}}$, then one could use Eqn. (4.3) as the framework to achieve JL-Type projection in the manifold. As such, we propose a random projection framework for manifold points, which is briefly illustrated in Figure 4.1. This hyperplane generation is the central idea in this chapter. First, we generate the hyperplanes over the RKHS. The points over the manifold are then projected into the projected space by using the specified kernel similarity function, such as the Gaussian kernel or projection kernel. Once the manifold points have been embedded into the projected space, we apply the Euclidean-based learning methods (*e.g.,* using K-means algorithm to perform clustering).

In this chapter, we explore three hyperplane generation methods for manifold points: (1) KGRP; (2) KORP and (3) KPCA-RP. The diagram of our proposed generation methods is illustrated in Figure 4.2. Briefly speaking, the hyperplanes are generated using a randomly selected subset from the dataset. The projection made by the hyperplanes will follow one of the properties in Definition 1. We will elaborate on the generation process and theoretical analysis in the following section. Note that the descriptions of generation process of our methods are based on the clustering scenarios and can be easily adapted to classification scenarios (refer to Section 4.6 for details).

4.4.1 Kernelised Gaussian Random Projection (KGRP)

In the KGRP method, the hyperplanes are generated from the standard Gaussian distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Each hyperplane $\boldsymbol{r}_i \in \mathcal{H}$ is assumed to be spanned by a group of data points randomly selected. To this end, first a subset $\mathcal{S}$ containing p points $\{\phi(\boldsymbol{X}_1), \ldots, \phi(\boldsymbol{X}_p)\}$ is randomly chosen from the dataset, $\phi(\boldsymbol{X}_i)$ is the representation of manifold points $\boldsymbol{X}_i$ in the RKHS. Each data point $\phi(\boldsymbol{X}_i)$ from the subset is considered as a vector generated from a particular distribution D with unknown mean $\boldsymbol{\mu}$ and unknown covariance $\boldsymbol{\Sigma}$. Thanks to the Central Limit Theorem (CLT) [117], one can still produce standard Gaussian distribution data points from these data. More precisely, the CLT states that when the number of data points grows larger, the difference between the population mean and the sample mean approximates the normal distribution $\mathcal{N}(\mathbf{0},\boldsymbol{\Sigma})$. As such, we first randomly select t, $t < p$, data

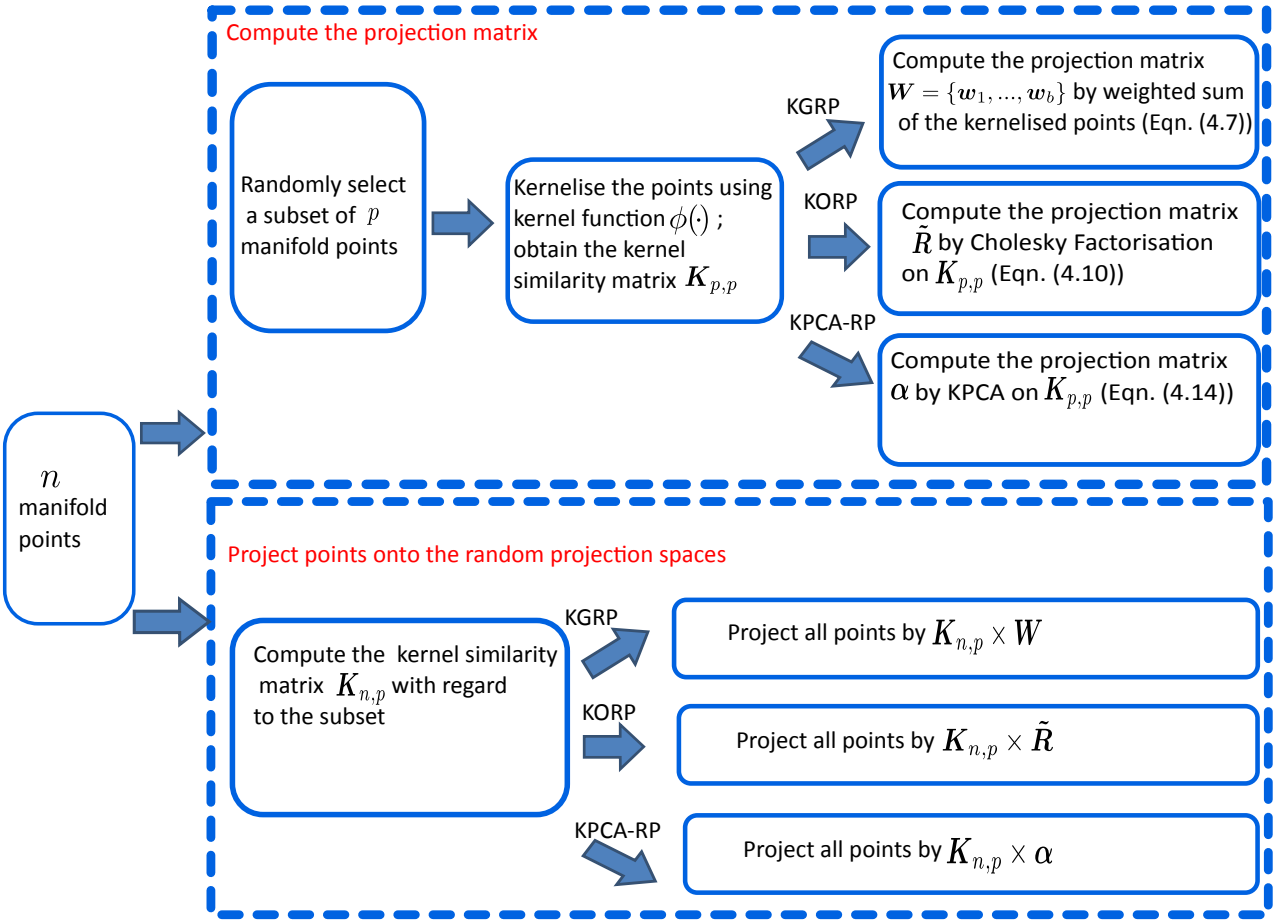


Figure 4.2: The diagram of our proposed generation methods: KORP, KGRP and KPCA-RP.

points from $\mathcal{S}$ and let these points be the set $\mathcal{S}_1 \subset \mathcal{S}$. Let $\boldsymbol{z}_t = \frac{1}{t}\sum_{i \in \mathcal{S}_1} \phi(\boldsymbol{X}_i)$ be the sample mean over $\mathcal{S}_1$. By applying the CLT and the Whitening transform [42], the vector $\boldsymbol{r}_i = \boldsymbol{\Sigma}^{-\frac{1}{2}}\sqrt{t}(\boldsymbol{z}_t - \boldsymbol{\mu})$ can be considered as the point generated from a standard Gaussian distribution; thus $\boldsymbol{r}_i$ could be used as a random projection hyperplane. Therefore, we denote our embedding function that projects data points in the RKHS to the random projection space by:

$$f(\phi(\boldsymbol{X}_i)) = \phi(\boldsymbol{X}_i)^T \boldsymbol{\Sigma}^{-\frac{1}{2}}\sqrt{t}(\boldsymbol{z}_t - \boldsymbol{\mu}) \; . \tag{4.4}$$

The mean is implicitly estimated as $\boldsymbol{\mu} = \frac{1}{p}\sum_{i=1}^{p} \phi(\boldsymbol{X}_i)$, and the covariance matrix $\boldsymbol{\Sigma}$ is also formed over the p data points. In order to compute Eqn. (4.4), one could use a similar approach to that of Kernel Principal Component Analysis (KPCA) [122]. Specifically, let the Eigen-decomposition of the covariance matrix $\boldsymbol{\Sigma}$ and the kernel matrix over p data points $\boldsymbol{K}_{\mathcal{S}}$, be $\boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^\top$ and $\boldsymbol{U}\boldsymbol{\Theta}\boldsymbol{U}^\top$ respectively. Based on the fact that the non-zero eigenvalues of $\boldsymbol{V}$ are equal to the non-zero eigenvalues of $\boldsymbol{\Theta}$, Kulis-Grauman [83] proved that Eqn. (4.4) is the same as:

$$\sum_{i=1}^{p} \boldsymbol{w}(i)(\phi(\boldsymbol{X}_i)^T \phi(\boldsymbol{X})) \; , \tag{4.5}$$

Algorithm 1: Kernelized Gaussian Random Projection (KGRP)

Input: the entire dataset: a set of manifold-valued data points $\{\boldsymbol{X}_i\}_{i=1}^n$, $\boldsymbol{X}_i \in \boldsymbol{\mathcal{M}}$; the size of $\mathcal{S}$: p; the desired projected space dimensionality : b

Output: $\{\boldsymbol{x}_i\}_{i=1}^n$, $\boldsymbol{x}_i \in \mathbb{R}^b$ the data points in the projected space

1: Randomly select p points $\{\boldsymbol{X}_i\}_{i=1}^p$ from the entire dataset
2: Compute the Kernel Gram matrix $\boldsymbol{K}_{\mathcal{S}}$ over points $\{\boldsymbol{X}_i\}_{i=1}^p$, $\boldsymbol{K}_{\mathcal{S}} = \phi(\boldsymbol{X}_i)^\top \phi(\boldsymbol{X}_j), \forall \boldsymbol{X}_i, \forall \boldsymbol{X}_j \in \{\boldsymbol{X}_i\}_{i=1}^p$, let $\mathcal{S} = \{\phi(\boldsymbol{X}_i)\}_{i=1}^p$ denote the representations for these p points in the RKHS
3: Compute the projection matrix $\boldsymbol{W} = \{\boldsymbol{w}_1, ..., \boldsymbol{w}_b\}$, $\forall \boldsymbol{w}_i \in \mathbb{R}^p$
4: **for** $i = 1 \rightarrow b$ **do**
5: $\quad \mathcal{S}_1 \leftarrow$ Randomly select t data points from $\mathcal{S}$
6: $\quad \boldsymbol{e}_{\boldsymbol{\mathcal{S}}} = [\Delta_1, ..., \Delta_p]$ if $\phi(\boldsymbol{X}_i) \in \mathcal{S}_1$, $\Delta_i = 1$; otherwise $\Delta_i = 0$
7: $\quad \boldsymbol{w}_i = \sqrt{\frac{p-1}{t}} \boldsymbol{K}_{\mathcal{S}}^{-\frac{1}{2}} \boldsymbol{e}_{\mathcal{S}}$
8: **end for**
9: Project each point $\boldsymbol{X}_i$ into the random projection space: $\boldsymbol{x}_i = \tilde{\boldsymbol{K}} \boldsymbol{W}$, where $\tilde{\boldsymbol{K}}$ is the Gram matrix between $\boldsymbol{X}_i$ and the points $\{\boldsymbol{X}_i\}_{i=1}^p$

where

$$\boldsymbol{w}(i) = \frac{1}{t} \sum_{j=1}^{p} \sum_{l \in \mathcal{S}_1} \boldsymbol{K}_{ij}^{-\frac{3}{2}} \boldsymbol{K}_{jl} \,. \tag{4.6}$$

Note that $\mathcal{S}_1$ is the set of t points which are randomly selected from $\mathcal{S}$. Further, defining $\boldsymbol{e}$ as a vector of all ones, and $\boldsymbol{e}_{\mathcal{S}_1}$ as a zero vector with ones in the entries corresponding to the indices of $\mathcal{S}_1$, the expression in Eqn. (4.6) can be further simplified to:

$$\boldsymbol{w} = \sqrt{\frac{p-1}{t}} \boldsymbol{K}_{\mathcal{S}}^{-\frac{1}{2}} \boldsymbol{e}_{\mathcal{S}_1} \,. \tag{4.7}$$

The pseudo code for KGRP is summarised in Algorithm 1. We note that the above formulation was first described for developing the kernelise locality sensitive hashing method in Euclidean scenarios [83]. We then adapted the method in our previous work [5] to perform random projection on SPD manifolds for classification purposes. In this chapter, we further apply the method for clustering on Riemannian manifold problems.

We note that the total computational complexity of the KGRP algorithm is $O(np + p^3 + np^2)$. Specifically, there are three factors contributing to the computational complexity:

1. Computing the kernel Gram matrix $\boldsymbol{K}_{n,p}$ between n points and p selected points which requires $O(np)$ operations ($p \ll n$);

2. Generating the random hyperplanes, necessitates calculation of the kernel matrix $\boldsymbol{K}_{\mathcal{S}}^{-1/2}$ for the p points in $\mathcal{S}$ which requires $O(p^3)$ operations;

3. Projecting all of the data points into the random projection space which requires $O(np^2)$ operations;

4.4.2 Kernelised Orthonormal Random Projection (KORP)

In the second method, we generate orthonormal random hyperplanes (*i.e.,* the first property). We first present the following Lemma that relates the JL-Lemma to the margin of the linear hyperplane in supervised learning settings [18].

Lemma 4.4.1 *Consider any distribution over labelled examples in Euclidean space such that there exists a linear separator* $\boldsymbol{w}^\top \cdot \boldsymbol{x} = 0$ *with margin* λ*. If we draw* $d \geq \frac{8}{\varepsilon}\left[\frac{1}{\lambda^2}\ln\frac{1}{\delta}\right]$ *examples* $\boldsymbol{z}_1, \cdots, \boldsymbol{z}_d$ *iid from this distribution, with probability* $\geq 1-\delta$*, there exists a vector* $\boldsymbol{w}'$ *in span* $(\boldsymbol{z}_1, \cdots, \boldsymbol{z}_d)$ *that has error at most* ε *at margin* $\frac{\lambda}{2}$ *[18].*

Proof. We refer the readers to [18] for the proof of this Lemma.
Remarks. Lemma 4.4.1 essentially states that, with a high probability, the margin is still well preserved (with error at most ε) when the hyperplane $\boldsymbol{w}'$ is selected from the space spanned by a subset of the data points. Note that, as suggested in [126], when the margin is well preserved, then the angle and distance between points are also well preserved.

This Lemma can also be applied for cases where the data points are in the RKHS. This is because the RKHS is essentially an infinite-dimensional Euclidean space [18]. Given a set of points which are linearly separable with margin λ under a particular kernel function, we draw d random examples $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_d$ from the same distribution. Then, according to Lemma 4.4.1, with probability $\geq 1-\delta$, there exists a separator in RKHS $\boldsymbol{w}' \in \mathcal{H}$ and $\boldsymbol{w}' = \alpha_1\phi(\boldsymbol{x}_1) + \cdots + \alpha_d\phi(\boldsymbol{x}_d)$ with error rate at most ε. Note that as $\boldsymbol{w}'^\top \cdot \phi(\boldsymbol{x}) = \alpha_1 \mathrm{K}(\boldsymbol{x}, \boldsymbol{x}_1) + ... + \alpha_d \mathrm{K}(\boldsymbol{x}, \boldsymbol{x}_d)$, we then can simply consider the vector of $[\mathrm{K}(\boldsymbol{x}, \boldsymbol{x}_1) \cdots \mathrm{K}(\boldsymbol{x}, \boldsymbol{x}_d)]$ as the feature representation of $\boldsymbol{x}$ in the space spanned by $\{\phi(\boldsymbol{x}_i)\}_{i=1}^d$. In other words, the $\mathrm{K}(\boldsymbol{x}, \boldsymbol{x}_i)$ is considered as the i-th feature of $\boldsymbol{x}$. We can further formalise this observation with the following Corollary [18].

Corollary 4.4.2 *If distribution* P *has margin* λ *in the RKHS, then with the probability* $\geq 1-\delta$*,* $d = \frac{8}{\varepsilon}\left[\frac{1}{\lambda^2}\ln\frac{1}{\delta}\right]$*, if* $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_d$*, are drawn from the same distribution, the mapping:* $\mathrm{F}_1(\boldsymbol{x}) = [\mathrm{K}(\boldsymbol{x}, \boldsymbol{x}_1), \cdots, \mathrm{K}(\boldsymbol{x}, \boldsymbol{x}_d)]$ *produces a distribution* $\mathrm{F}_1(P)$ *on labelled examples in* $\mathbb{R}^d$ *that is linearly separable with error at most* ε *[18].*

Remarks. The above Corollary suggests the following points:

- One could generate random projection hyperplanes by randomly selecting a subset of data points in RKHS and then projecting a point into this space by using $\mathrm{F}_1(\boldsymbol{x})$;
- This projection is a JL-Type projection.

In light of these facts, for our case, we randomly select p points, denote $\mathcal{S} = \{\phi(\boldsymbol{X}_1), \cdots, \phi(\boldsymbol{X}_p)\}$ as the implicit representations of the p points in RKHS. However, as it is possible that some hyperplanes are not linearly independent, then the hyperplanes could be highly correlated. To that end, one needs to orthogonalise the hyperplane set $\mathcal{S}$ [18]. In this work, we apply QR decomposition [157] to construct a set of orthonormal basis from the original basis spanning the same subspace. Let us

arrange the original basis $\{\phi(\boldsymbol{X}_i)\}_{i=1}^{p}$ into a matrix $\boldsymbol{A}$. Then the matrix $\boldsymbol{A}$ can be decomposed into $\boldsymbol{Q}$ and $\tilde{\boldsymbol{R}}$ as follows:

$$\boldsymbol{A} = [\phi(\boldsymbol{X}_1), \cdots, \phi(\boldsymbol{X}_p)] = \boldsymbol{Q}\tilde{\boldsymbol{R}}\,, \tag{4.8}$$

where $\boldsymbol{Q}$ is the orthonormal basis and $\tilde{\boldsymbol{R}}$ is the upper triangular matrix. Assuming that we have the orthonormal basis $\boldsymbol{Q}$, then we can observe the following when a data point $\phi(\boldsymbol{X})$ is projected into the orthonormal basis $\boldsymbol{Q}$:

$$\begin{aligned}
\phi(\boldsymbol{X})^\top \boldsymbol{Q} &= \phi(\boldsymbol{X})^\top \boldsymbol{Q}\tilde{\boldsymbol{R}}\tilde{\boldsymbol{R}}^{-1} \\
&= \phi(\boldsymbol{X})^\top [\phi(\boldsymbol{X}_1), ..., \phi(\boldsymbol{X}_p)]\tilde{\boldsymbol{R}}^{-1} \\
&= [\phi(\boldsymbol{X})^\top \phi(\boldsymbol{X}_1), ..., \phi(\boldsymbol{X})^\top \phi(\boldsymbol{X}_p)]\tilde{\boldsymbol{R}}^{-1} \\
&= [\mathrm{K}(\boldsymbol{X}, \boldsymbol{X}_1), ..., \mathrm{K}(\boldsymbol{X}, \boldsymbol{X}_p)]\tilde{\boldsymbol{R}}^{-1}\,.
\end{aligned} \tag{4.9}$$

In other words, one only needs to determine the upper triangular $\tilde{\boldsymbol{R}}$ in order to do the projection. We note that as the original basis $\{\phi(\boldsymbol{X}_i)\}_{i=1}^{p}$ are in the RKHS then it is not trivial to apply the QR decomposition to matrix $\boldsymbol{A}$. To that end, we first multiply the matrix $\boldsymbol{A}$ by its transpose. By doing this, we will get the kernel matrix $\boldsymbol{K}_{\mathcal{S}}$, where $\boldsymbol{K}_{\mathcal{S}}(i,j) = \phi(\boldsymbol{X}_i)^\top \phi(\boldsymbol{X}_j), \forall\, \phi(\boldsymbol{X}_i)$ and $\forall\, \phi(\boldsymbol{X}_j) \in \mathcal{S}$. Thus:

$$\begin{aligned}
\boldsymbol{K}_{\mathcal{S}} &= \boldsymbol{A}^\top \boldsymbol{A} \\
&= (\boldsymbol{Q}\tilde{\boldsymbol{R}})^\top \boldsymbol{Q}\tilde{\boldsymbol{R}} \\
&= \tilde{\boldsymbol{R}}^\top \boldsymbol{Q}^\top \boldsymbol{Q}\tilde{\boldsymbol{R}} \\
&= \tilde{\boldsymbol{R}}^\top \tilde{\boldsymbol{R}}\,.
\end{aligned} \tag{4.10}$$

We can employ the Cholesky Factorisation [157] on the kernel matrix $\boldsymbol{K}_{\mathcal{S}}$, in order to compute the upper triangular $\tilde{\boldsymbol{R}}$. Algorithm 2 outlines the algorithm for the proposed Kernelised Orthonormal Random Projection (KORP).

The computational complexity of KORP depends on the following steps:

1. Computing the kernel Gram matrix between the entire dataset and the subset $\mathcal{S}$ which requires $O(np)$ operations;
2. Applying Cholesky Factorisation on the kernel Gram matrix of the p points in $\mathcal{S}$ which requires $O(p^3)$ operations;
3. Applying the matrix inverse of the right triangular matrix $\tilde{\boldsymbol{R}}$ which demands $O(p^3)$ operations;
4. Projecting all of the data points into the orthonormal space with $O(np^2)$ operations;

Hence, the total computational complexity is $O(np + p^3 + np^2)$.

Algorithm 2: Kernelised Orthonormal Random Projection (KORP)

Input: the entire dataset: a set of manifold-valued data points $\{\boldsymbol{X}_i\}_{i=1}^n$, $\boldsymbol{X}_i \in \boldsymbol{\mathcal{M}}$; the desired projected space dimensionality : p

Output: $\{\boldsymbol{x}_i\}_{i=1}^n$, $\boldsymbol{x}_i \in \mathbb{R}^p$ the data points in the projected space

1: Randomly select p points $\{\boldsymbol{X}_i\}_{i=1}^p$ from the entire dataset
2: Compute the kernel Gram matrix $\boldsymbol{K}_{\mathcal{S}}$ over points $\{\boldsymbol{X}_i\}_{i=1}^p$ $\boldsymbol{K}_{\mathcal{S}} = \phi(\boldsymbol{X}_i)^\top \phi(\boldsymbol{X}_j), \forall \boldsymbol{X}_i, \forall \boldsymbol{X}_j \in \{\boldsymbol{X}_i\}_{i=1}^p$
3: Apply Cholesky Factorisation to the kernel matrix $\boldsymbol{K}_{\mathcal{S}} = \tilde{\boldsymbol{R}}\tilde{\boldsymbol{R}}^\top$
4: Project each point $\boldsymbol{X}_i$ into the random projection space: $\boldsymbol{x}_i = \tilde{\boldsymbol{K}}\tilde{\boldsymbol{R}}^{-1}$, where $\tilde{\boldsymbol{K}}$ is the Gram matrix between $\boldsymbol{X}_i$ and the points $\{\boldsymbol{X}_i\}_{i=1}^p$,

4.4.3 KPCA-based Random Projection (KPCA-RP)

Inspired by the previous method, one can derive orthonormal projections using the Kernel PCA (KPCA). More precisely, after generating random projection hyperplanes by randomly selecting the subset $\mathcal{S}$, one can obtain the principal components of the data points in $\mathcal{S}$ by applying the KPCA. The principal components of $\mathcal{S}$ are then considered as the set of orthogonal random projection hyperplanes. Finally, following Eqn. (4.3), the entire data points can be projected into the random projection space using the hyperplanes.

Let us suppose $\boldsymbol{C}$ is the covariance matrix of the points in $\mathcal{S}$ which have been centred:

$$\boldsymbol{C} = \frac{1}{p}\sum_{i=1}^{p} \phi(\boldsymbol{X}_i)\phi(\boldsymbol{X}_i)^\top. \tag{4.11}$$

To apply KPCA, one needs to solve the generalised eigen-decomposition problem:

$$\tau \boldsymbol{V} = \boldsymbol{C}\boldsymbol{V}\ . \tag{4.12}$$

Following the same argument as KPCA [122], the eigenvectors of the covariance matrix C lie in the span of $\phi(\boldsymbol{X}_1), \phi(\boldsymbol{X}_2), .., \phi(\boldsymbol{X}_p)$:

$$\boldsymbol{V}_k = \sum_{i=1}^{p} \alpha_i^k \phi(\boldsymbol{X}_i)\ , \tag{4.13}$$

where the set $\{\alpha_i^k\}_{i=1}^p$ can be determined by solving the following equation:

$$p\tau\boldsymbol{\alpha} = \boldsymbol{K}_{\mathcal{S}}\boldsymbol{\alpha}\ , \tag{4.14}$$

where $\boldsymbol{\alpha} = [\boldsymbol{\alpha}^1 \cdots \boldsymbol{\alpha}^k]$ is a matrix wherein each column represents the vector $\boldsymbol{\alpha}^k = [\alpha_1^k \cdots \alpha_p^k]^\top$ whose elements are the linear combination coefficients presented in Eqn. (4.13) and $\boldsymbol{K}_{\mathcal{S}}$ is the kernel matrix of the set $\mathcal{S}$. Note that the above equation suggests that the vector $\boldsymbol{\alpha}^k$ is one of the eigenvectors of $\boldsymbol{K}_{\mathcal{S}}$.

Let $\{\boldsymbol{V}_k\}_{k=1}^{p}$ be the set of principal components extracted from Eqn. (4.12). To project a point into the principal component $\boldsymbol{V}_k$, we perform:

$$\phi(\boldsymbol{X})^\top \cdot \boldsymbol{V}_k = \sum_{i=1}^{p} \alpha_i^k \phi(\boldsymbol{X})^\top \phi(\boldsymbol{X}_i) \; . \tag{4.15}$$

In the following, we present a theorem that guarantees that projections into the principal components of the subset $\mathcal{S}$ achieves JL-Type projection.

Theorem 4.4.3 *If a set of points can be separated by a margin λ in the RKHS, then with probability $\geq 1-\delta$, if $\mathcal{S} = \{\phi(\boldsymbol{X}_1), ..., \phi(\boldsymbol{X}_p)\}$, $\boldsymbol{X}_i \in \boldsymbol{\mathcal{M}}$, $\phi(\boldsymbol{X}_i) \in \mathcal{H}$ are drawn from the same distribution for $p = \frac{8}{\varepsilon}\left[\frac{1}{\lambda^2}\ln\frac{1}{\delta}\right]$, the mapping $\mathrm{F}_2(x) = \mathrm{F}_1(x)[\boldsymbol{\alpha}^1 \cdots \boldsymbol{\alpha}^p]$, where $\boldsymbol{\alpha}^k$ is the k-th eigenvector of $\boldsymbol{K}_{\mathcal{S}}$, achieves JL-Type projection with error at most ε.*

Proof. As presented in Corollary 4.4.2, $\mathrm{F}_1(x)$ is the function that maps a point into a random projection space wherein the set of hyperplanes $\mathcal{S}$ is randomly selected from a set of given points. It is known that principal components of $\mathcal{S}$ represent the orthonormal bases spanning the subspace spanned by $\mathcal{S}$. Henceforth, computing the principal components of $\mathcal{S}$ can be considered as orthogonalisation of the hyperplanes.

Remarks. The above theorem states that applying KPCA on $\mathcal{S}$ means orthogonalising the hyperplanes in $\mathcal{S}$. Therefore, the difference between KPCA-RP and KORP is related to how the hyperplanes are orthogonalised. We present the KPCA-RP pseudo code in Algorithm 3.

Algorithm 3: KPCA-based Random Projection (KPCA-RP)

Input: the entire dataset: a set of manifold-valued data points $\{\boldsymbol{X}_i\}_{i=1}^{n}$, $\boldsymbol{X}_i \in \boldsymbol{\mathcal{M}}$; the desired projected space dimensionality : p

Output: $\{\boldsymbol{x}_i\}_{i=1}^{n}$ the data points in the projected space

1: Randomly select p points $\{\boldsymbol{X}_i\}_{i=1}^{p}$ from the entire dataset
2: Compute the kernel Gram matrix $\boldsymbol{K}_{\mathcal{S}}$ over points $\{\boldsymbol{X}_i\}_{i=1}^{p}$ $\boldsymbol{K}_{\mathcal{S}} = \phi(\boldsymbol{X}_i)^\top \phi(\boldsymbol{X}_j), \forall \boldsymbol{X}_i, \forall \boldsymbol{X}_j \in \{\boldsymbol{X}_i\}_{i=1}^{p}$
3: Apply KPCA to kernel matrix $\boldsymbol{K}_{\mathcal{S}}$ to obtain the eigenvectors α.
4: Project each point $\boldsymbol{X}_i$ into the random projection space: $\boldsymbol{x}_i = \tilde{\boldsymbol{K}}\alpha$, where $\tilde{\boldsymbol{K}}$ is the Gram matrix between $\boldsymbol{X}_i$ and the $\{\boldsymbol{X}_i\}_{i=1}^{p}$

In terms of calculating the computational complexity of the KPCA-RP algorithm, one needs to consider the following factors:

1. Computing the kernel Gram matrix between the entire dataset and the subset $\mathcal{S}$, which requires $O(np)$ operations;
2. Applying KPCA on the kernel Gram matrix of subset $\mathcal{S}$, which requires $O(p^3)$ operations;
3. Projecting all of the data points into the orthonormal space, which requires $O(np^2)$ operations;

Hence, the total computational complexity is $O(np + p^3 + np^2)$.

4.5 Experimental Results on Clustering Tasks

For the clustering tasks, we evaluate our proposal using six benchmark datasets:

- Ballet dataset [154];
- UCSD traffic dataset [26];
- UCF101 Human actions dataset [131];
- Brodatz texture dataset [116];
- KTH-TIPS2b material dataset [23];
- HEp-2 Cell ICIP2013 dataset [71].

In our evaluation, we consider each video of the first three datasets (*i.e.,* Ballet, UCSD and UCF101) as an image set which can be effectively modelled as a point on Grassmann manifolds. In addition, we use SPD manifold to model images of the latter three datasets (*i.e.,* Brodatz, KTH-TIPS2b and HEp-2 Cell ICIP2013). To demonstrate the efficacy of our framework, we report the clustering performance and the run time.

4.5.1 Datasets and Feature Extraction

Ballet action dataset (Ballet) [154] - The Ballet dataset presents sequences of videos of ballet actions. More precisely, it comprises 44 sequences with 8 different actions: R-L presenting, L-R presenting, Presenting, Jump & swing, Jump, Turn, Step, and Stand still (see Figure 4.3a for examples). These ballet actions were performed by two men and one woman, resulting in significant intra-class variations such as speed, clothing and movements. In this evaluation, each video is considered as an image set. We then represent each image set as a point in the Grassmann manifold. To that end, all the videos are down sampled to 16×16 pixels. A Grassmann point is extracted for every 6 consecutive frames. Technically, we first vectorise each frame into a column vector and arrange them into a 256×6 tall matrix (*i.e.,* $256 = 16 \times 16$). The matrix can be considered as a subspace and the orthonormal bases spanning the subspace can be determined by applying the Singular Value Decomposition (SVD). The set of orthonormal bases is considered as a Grassmann point [127]. We use the projection kernel (see Eqn. (2.13)) in this evaluation.

UCSD traffic dataset (UCSD) [26] - The UCSD traffic dataset consists of 254 video sequences collected from the highway traffic over two days in Seattle (see Figure 4.3b for examples). It contains a variety of traffic patterns and weather conditions (*i.e.,* overcast, raining, sunny). In total, there are 44 sequences of heavy traffic (slow, stop and go speeds), 45 sequences of medium traffic (reduced speed), and 165 sequences of light traffic (normal speed). To extract a Grassmann point, we first randomly select half the number of frames from each video. Each frame in each sequence is downsized to 140×161 pixels and further normalised by subtracting the mean frame and dividing the variance.

Then, we apply the two dimensional Discrete Cosine Transform (DCT) on the frame and use the DCT coefficients as the feature vector for each frame. SVD is applied on the feature vectors of the frames to obtain the set of orthonormal bases. We also choose the projection kernel (see Eqn. (2.13)) for this dataset.

UCF101 Human Actions dataset (UCF101) [131] - This dataset consists of $13,320$ videos that belong to 101 categories. For example, Applying Eye Makeup, Blow Dry Hair and Mixing Batter (refer to Figure 4.3c). For each video, we first extract the normalised pixel intensities as features for all the frames. Then the SVD is applied on these features of each video to obtain the Grassmann manifold point. Thus, in this dataset, there are $13,320$ manifold points in total. Projection kernel (see Eqn. (2.13)) is used.

(a)

(b)

(c)

Figure 4.3: Examples from (a) Ballet action dataset [154] (b) UCSD traffic dataset [26] and (c) UCF101 dataset [131]

Brodatz texture dataset (Brodatz) [116] - For the Brodatz dataset (refer to Figure 4.4a for examples) we follow the protocol presented in [129]. The protocol includes 3 subsets with different numbers of classes: 5-class-texture (5c, 5m, 5v, 5v2, 5v3), 10-class-texture (10, 10v) and 16-class-texture (16c, 16v). Each image is down-sampled to 256×256 pixels and divided into 64 32×32 pixel size regions. A feature vector $f(x, y)$ for each pixel is calculated using the grayscale intensities and absolute values of the first- and second-order derivatives of spatial feature vectors. It can be illustrated as:

$$f(x,y) = \left[I\left(x,y\right), \left|\frac{\partial I}{\partial x}\right|, \left|\frac{\partial I}{\partial y}\right|, \left|\frac{\partial^2 I}{\partial x^2}\right|, \left|\frac{\partial^2 I}{\partial y^2}\right| \right] . \tag{4.16}$$

Each region is represented by a covariance matrix (SPD matrix) formed from these feature vectors. The Gaussian Kernel with Log-Euclidean distance (see Eqn. (2.8)) is used for this dataset.

KTH-TIPS2b material dataset (KTH-TIPS2b) [23] - This dataset contains 11 material categories captured under 4 different illuminations, in 3 poses and at 9 scales (refer to Figure 4.4b). Thus, there are $3 \times 4 \times 9 = 108$ images for each sample in one category, with 4 samples per material. We extract a 20-dimensional feature vector for each pixel in the image:

$$f(x,y) = [\boldsymbol{I}(x,y), \boldsymbol{Y}(x,y), \boldsymbol{C}_b(x,y), \boldsymbol{C}_r(x,y), F^1_{(x,y)}(\boldsymbol{Y}) \cdots F^{16}_{(x,y)}(\boldsymbol{Y})] , \tag{4.17}$$

where $\boldsymbol{I}(x, y)$ is the image grey level value at location (x, y); $\boldsymbol{Y}$, $\boldsymbol{C}_b$ and $\boldsymbol{C}_r$ are the perceptually uniform CIELab colour space; The filter banks F^i consist of different offset Gaussians applied on the luminance channel $\boldsymbol{Y}$ [137]. The covariance matrix is computed once the feature vectors are extracted from every pixel location. This becomes the image representation over a SPD manifold. For the manifold kernel in this dataset, we use Gaussian kernel with the Stein Divergence (see Eqn. (2.9)) as this has been shown to be effective in various classification problem domains [5, 6].

HEp-2 Cell ICIP2013 dataset [71] - This dataset contains $13,596$ cell images that include six cell patterns namely Centromere, Golgi, Homogeneous, Nucleolar, Nuclear Membrane, and Speckled (refer to Figure 4.4c). The cell boundary of every cell image is described by a mask image of the same size. For each cell image, we first extract the following feature vector of each pixel that belongs to the cell content:

$$f(x,y) = \left[\left|\frac{\partial I}{\partial x}\right|, \left|\frac{\partial I}{\partial y}\right|, I\left(x,y\right), \left|\frac{\partial^2 I}{\partial x^2}\right|, \left|\frac{\partial^2 I}{\partial y^2}\right|, \arctan\left(\left|\frac{\partial I}{\partial x}\right| / \left|\frac{\partial I}{\partial y}\right|\right) \right] . \tag{4.18}$$

Then, the covariance matrix (SPD matrix) is formed from these feature vectors extracted from each image. We also use Gaussian kernel with the Stein Divergence (see Eqn. (2.9)) for the evaluation on this dataset.

4.5.2 Experimental Settings

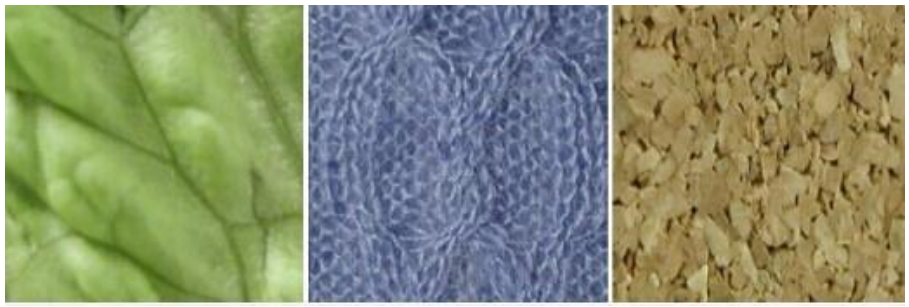
In the clustering tasks, we first randomly project the points and then apply K-means. As such, for each dataset, we first run each proposed projection method 10 times to generate 10 different random

projection representations. Then, for each representation, we run the K-means algorithm 10 times, resulting in each method being repeated 100 times for each evaluation. The average of clustering performance and run time were reported. As the source of variation for the other approaches is predominantly on the initial cluster seeds of K-means, we only repeat the experiment 10 times to obtain the average clustering performance and run time.

All of the approaches are tuned to give the best performance. We find the optimum size of set $\mathcal{S}$ as follows:

- Ballet: 100;
- UCSD: 90;
- UCF101: 101;
- Brodatz: 100;
- KTH-TIPS2b: 48;

(a)



(b)

(c)

Figure 4.4: Examples from (a) BRODATZ texture dataset [116], (b) KTH-TIPS2b material dataset [23] and (c) HEp-2 Cell ICIP2013 dataset [71].

- HEp-2 Cell ICIP2013: 60.

In addition, for KGRP, we set the number of dimensionality, b, to 300.

To measure the clustering quality, there are two main types of metrics: internal metrics based on the distances between data points in the space, and external metrics based on the labels of the data [104]. The clustering task in our proposed framework is performed in a transformed space which may have different scale to other spaces used by comparable methods such as LogE (see below for further discussion on LogE). This may make the internal metrics such as Dunn Index [44] unsuitable in our case. Thus, we choose four external metrics to measure the clustering quality: Rand Index (RI), Cluster Purity (CP), F-Measure and Normalized Mutual Information (NMI). More specifically, RI measures the percentage of decisions that are correct and defined as:

$$RI = \frac{TP + TN}{TP + FP + FN + TN}\,, \tag{4.19}$$

where a True Positive (TP) decision assigns two similar points to the same cluster, a True Negative (TN) decision assigns two dissimilar points to different clusters, a False Positive (FP) decision assigns two dissimilar points to the same cluster and a False Negative (FN) decision assigns two similar points to different clusters. CP measures the cluster quality by counting the number of correctly assigned points defined as:

$$\mathrm{CP}\left(\Omega, \mathcal{C}\right) = \frac{1}{N}\sum_{k} \left|\omega_k \cap c_j\right|\,, \tag{4.20}$$

where $\Omega = \{\omega_1, \omega_2, ..., \omega_K\}$ is the set of clusters and $\mathcal{C} = \{c_1, c_2, ..., c_J\}$is the set of classes. F-Measure penalises the FN more strongly by defined as follows:

$$\text{F-Measure} = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}\,, \tag{4.21}$$

where $P = \frac{TP}{TP+FP}$, $R = \frac{TP}{TP+FN}$ and $\beta > 1$. In this chapter, we set $\beta = 5$, which is one of the most commonly used value for computing the F-Measure. NMI measures the mutual dependence between the clusterω_k and class c_j. It can be formulated as:

$$\begin{aligned}
&\mathrm{NMI}\left(\Omega, \mathcal{C}\right) = \frac{I(\Omega, \mathcal{C})}{[H(\Omega) + H(\mathcal{C})]/2}\,,\\
&I\left(\Omega, \mathcal{C}\right) = P\left(\omega_k \cap c_j\right) \log \frac{P\omega_k \cap c_j}{P(\omega)_k p(c_j)}\,,\\
&H\left(\Omega\right) = -\sum_{k} P(\omega_k) \log P(\omega_k)\,,\\
&H\left(\mathcal{C}\right) = -\sum_{k} P(c_k) \log P(c_k)\,,
\end{aligned} \tag{4.22}$$

where $P(\omega_k)$, $P(c_j)$, and $P(\omega_k \cap c_j)$ represent the probabilities of a point being cluster ω_k, class c_j, and in the intersection of ω_k and c_j, respectively. Interested readers are referred to [104] for further explanation of each metric. we also measure the run time (in seconds) of each approach on every

evaluation. The run time is measured from the kernel matrix computation until the completion of the clustering process. Finally, we report the average run time of the approaches.

Our proposal is contrasted to six approaches: (1) Intrinsic K-means [139]; (2) Log-Euclidean K-means [48]; (3) Kernel K-means [40, 76]; (4) KPCA K-means [76, 122]; (5) Sigma set K-means [72] and (6) Grassmann clustering [127]. The following is the brief description of each approach.

Intrinsic K-means (Intrinsic): To cluster a set of manifold points, Intrinsic K-means works directly on the manifold using the appropriate geodesic distance [139]. We note that as the intrinsic approach is generally very slow, we stop the Intrinsic K-means after 100 iterations.

Log-Euclidean K-means (LogE): We first project all of the manifold points into the tangent space at the identity [9]. Once projected, each point will be vectorised into a column vector. As for SPD manifolds, we follow the work in [111] that uses only the upper triangular elements. This trick will reduce the final representation dimensionality, markedly reducing the run time on the subsequent process. Unfortunately the trick cannot be used on Grassmann manifolds since the representation for a point on the Grassmann manifold is not a symmetric matrix. In this case, all the elements are used in the final representation. This could adversely affect the overall run time when the manifold dimensionality is high. In the final step, K-means algorithm is applied. Log-Euclidean k-means has been used for clustering large amount of manifold data [48].

Kernel K-means: This approach embeds manifold points into RKHS. Then Kernel K-means is applied to perform clustering [40, 76].

KPCA K-means (KPCA): All manifold points are first embedded into RKHS. Then, KPCA is used for projecting the points in RKHS into the space spanned by the principal components [76, 122]. Finally, the K-means is applied.

Sigma set K-means (SIS): Hong *et al.* [72] proposed a novel descriptor for SPD manifolds which simplifies the computations of distance and mean. Using their proposed descriptors, we apply K-means with novel efficient computations of mean and distance.

Grassmann clustering (G-clustering) Shirazi *et al.* [127] proposed a clustering method for Grassmann manifolds which use the eigenvectors of the normalised projection kernel matrix as the new features of Grassmann points.

4.5.3 Comparative Analysis on Clustering Quality

Tables 4.1, 4.2, 4.3, 4.4, 4.5 and 4.6 report the average clustering quality of each individual approach applied on each dataset. In general, our proposed methods perform well and show a close match to KPCA K-means and Kernel K-means. Also, the performance of the proposed approaches is similar to each other. These factors suggest that the proposed projection approaches possess the JL-Type projection properties. Furthermore, we find that the proposed approaches in some cases have markedly better performance than the Kernel K-means. One of the possible reasons could be that the random projection reduces the eccentricity of original Gaussian-distributed clusters and make clusters in projected spaces more spherical [36].

Table 4.1: The clustering quality with variance (in %) measured by Rand Index (RI), Cluster Purity (CP), F-Measure and Normalized Mutual Information (NMI) on Ballet dataset. The best performance is in bold. We refer to Section 4.5.2 for further explanation of each approach.

Methods/Measurements	RI	CP	F-Measure	NMI
Intrinsic [139]	73.68±0.00	34.92±0.00	33.81 ± 0.00	21.73 ± 0.00
LogE [48]	78.23±0.15	20.85±2.66	14.81 ± 0.37	3.91 ± 0.81
G-clustering [127]	76.41±0.07	18.63±0.58	16.39 ± 0.25	3.51 ± 0.47
Kernel K-means [40, 76]	**79.89±0.80**	40.86±3.06	32.92 ± 3.21	32.00 ± 2.73
KPCA [76, 122]	78.62±2.14	42.30±3.33	36.27 ± 2.68	34.80 ± 3.48
KGRP	78.02±1.79	41.89±2.43	37.98 ± 2.79	34.05 ± 2.41
KORP	78.28±1.68	**42.54±2.37**	**38.68 ± 2.81**	**35.30 ± 2.80**
KPCA-RP	77.81±1.94	41.90±2.31	38.23 ± 3.11	34.64 ± 2.75

Table 4.2: The clustering quality with variance (in %) measured by Rand Index (RI), Cluster Purity (CP), F-Measure and Normalized Mutual Information (NMI) on UCSD dataset. The best performance is in bold. We refer to Section 4.5.2 for further explanation of each approach.

Methods/Measurements	RI	CP	F-Measure	NMI
Intrinsic [139]	73.26±0.00	74.70±0.00	**75.15 ± 0.00**	36.18 ± 0.00
LogE [48]	55.24±3.25	67.23±2.66	40.39 ± 2.81	19.11 ± 3.59
G-clustering [127]	50.68±0.11	64.82±0.00	34.31 ± 0.12	0.92 ± 0.29
Kernel K-means [40, 76]	69.98±7.06	77.96±4.77	57.34 ± 10.22	45.50 ± 9.71
KPCA [76, 122]	**77.90±5.97**	80.08±2.96	**69.29 ± 7.56**	**51.31 ± 6.09**
KGRP	75.61±3.48	79.64±2.07	66.97 ± 5.17	48.29 ± 3.80
KORP	77.25±1.25	**80.18±0.74**	68.99 ± 1.62	50.58 ± 1.67
KPCA-RP	76.46±2.79	79.64±1.68	68.60 ± 3.50	49.74 ± 3.02

Table 4.3: The clustering quality with variance (in %) measured by Rand Index (RI), Cluster Purity (CP), F-Measure and Normalized Mutual Information (NMI) on UCF101 dataset. The best performance is in bold. We refer to Section 4.5.2 for further explanation of each approach.

Methods/Measurements	RI	CP	F-Measure	NMI
Intrinsic [139]	97.53 ± 0.00	12.94 ± 0.00	7.43 ± 0.00	27.65 ± 0.00
LogE [48]	97.89 ± 0.02	8.21 ± 0.15	3.62 ± 0.06	18.68 ± 0.07
Kernel K-means [40, 76]	97.71 ± 0.06	15.97 ± 0.48	8.80 ± 0.35	32.35 ± 0.31
KPCA [76, 122]	97.69 ± 0.02	**17.66 ± 0.33**	**9.47 ± 0.19**	**33.66 ± 0.18**
KGRP	**97.90 ± 0.03**	15.38 ± 0.28	7.40 ± 0.15	30.96 ± 0.21
KORP	**97.90 ± 0.02**	15.69 ± 0.28	7.62 ± 0.17	31.47 ± 0.17
KPCA-RP	97.89 ± 0.03	15.66 ± 0.33	7.59 ± 0.17	31.38 ± 0.23

Table 4.4: The clustering quality with variance (in %) measured by Rand Index (RI), Cluster Purity (CP), F-Measure and Normalized Mutual Information (NMI) on BRODATZ dataset. The best performance is in bold. We refer to Section 4.5.2 for further explanation of each approach.

Methods/Measurements	RI	CP	F-Measure	NMI
Intrinsic [139]	92.29±0.00	79.05±0.00	74.20 ± 0.00	75.94 ± 0.00
SIS [72]	91.42±0.00	76.99±0.00	69.68 ± 0.00	72.84 ± 0.00
LogE [48]	92.04±0.78	78.34±2.34	74.10 ± 2.10	76.13 ± 1.45
Kernel K-means [40, 76]	93.15±0.95	81.40±2.75	75.62 ± 2.13	78.19 ± 1.83
KPCA [76, 122]	**93.89±0.22**	82.60±1.14	**76.64 ± 0.66**	**79.44 ± 0.57**
KGRP	93.47±0.78	82.22±2.34	75.84 ± 1.82	78.49 ± 1.49
KORP	93.66±0.77	82.58±2.32	76.30 ± 1.81	79.11 ± 1.50
KPCA-RP	93.77±0.84	**82.81±2.49**	76.39 ± 1.93	79.16 ± 1.56

Table 4.5: The clustering quality with variance (in %) measured by Rand Index (RI), Cluster Purity (CP), F-Measure and Normalized Mutual Information (NMI) on KTH-TIPS2b dataset. The best performance is in bold. We refer to Section 4.5.2 for further explanation of each approach.

Methods/Measurements	RI	CP	F-Measure	NMI
Intrinsic [139]	86.99±0.00	49.45±0.00	36.19 ± 0.00	44.20 ± 0.00
SIS [72]	80.81±0.00	41.62±0.00	**44.45 ± 0.00**	40.47 ± 0.00
LogE [48]	85.94±0.60	45.19±1.32	33.48 ± 1.01	40.69 ± 0.82
Kernel K-means [40, 76]	88.35±0.35	52.59±1.37	41.11 ± 1.01	**51.08 ± 0.82**
KPCA [76, 122]	**88.48±0.40**	**53.38±1.53**	41.22 ± 1.35	50.97 ± 0.90
KGRP	88.41±0.42	53.15±1.34	40.48 ± 0.99	49.87 ± 1.01
KORP	88.36±0.39	53.04±1.10	40.61 ± 0.93	50.06 ± 0.92
KPCA-RP	88.35±0.44	53.45±1.35	40.21 ± 0.91	49.97 ± 1.09

Table 4.6: The clustering quality with variance (in %) measured by Rand Index (RI), Cluster Purity (CP), F-Measure and Normalized Mutual Information (NMI) on HEp-2 Cell ICIP2013 dataset. The best performance is in bold. We refer to Section 4.5.2 for further explanation of each approach.

Methods/Measurements	RI	CP	F-Measure	NMI
Intrinsic [139]	73.96 ± 0.00	44.02 ± 0.00	35.69 ± 0.00	22.65 ± 0.00
SIS [72]	74.50 ± 0.00	39.50 ± 0.00	27.32 ± 0.00	18.01 ± 0.00
LogE [48]	74.80 ± 0.95	46.00 ± 2.37	34.75 ± 0.86	23.64 ± 1.29
Kernel K-means [40, 76]	73.96 ± 2.14	46.45 ± 3.30	$\mathbf{37.29 \pm 3.29}$	24.29 ± 1.95
KPCA [76, 122]	$\mathbf{75.74 \pm 2.87}$	48.48 ± 1.94	34.23 ± 2.20	25.29 ± 0.00
KGRP	75.72 ± 0.31	$\mathbf{49.05 \pm 1.10}$	34.83 ± 0.84	$\mathbf{25.74 \pm 0.82}$
KORP	75.63 ± 0.62	48.70 ± 2.34	34.73 ± 1.67	25.49 ± 1.72
KPCA-RP	75.72 ± 0.41	48.70 ± 2.56	34.48 ± 1.74	25.46 ± 1.93

Intrinsic K-means gives us reasonable results as it directly works on manifold. Compared to the intrinsic approach, LogE has a worse performance in most of datasets. An exception is on the Ballet dataset where the intrinsic approach has a worse Rand Index than the LogE. We conjecture that this is caused by the failure of the intrinsic algorithm to converge in 100 iterations. Nevertheless, the other performance metrics such as CP, F-Measure and NMI for the intrinsic approach in this dataset still show reasonable performance. The worse performance for LogE is due to significant distortion of the pairwise distance produced when the points are projected into a tangent space. The G-clustering has a better Rand Index than the intrinsic approach in the Ballet dataset, which is a similar conclusion drawn in the original work proposing the approach [127]. Note that the measurements for clustering performance are different from that in [127]. In most cases, the G-clustering is not robust as the performance of G-clustering measured by CP, F-measure and NMI is usually low. In addition, we do not report the G-clustering results for the UCF101 dataset, as the K-means does not converge within a specified amount of time.

We found the performance of our proposed methods does not change significantly, when the parameters are varied. Figures 4.5 and 4.6 show two examples of the clustering results of KPCA-RP and KORP with different parameters on the Ballet and HEp-2 Cell ICIP2013 dataset, respectively. We note that the results on the other datasets also exhibit similar trends. This suggests that the issue raised in [108], where different parameters may adversely alter the kernel space, may not have significant effect on our work. We conjecture that this might be due to the selected manifold kernels crafted to capture the manifold intrinsic structure. However, if in the case where the parameter choice of the manifold kernel significantly contributes to the clustering results, one could use a randomly selected small subset of data to perform the parameter search.

The evaluation has clearly shown that our proposal has similar performance to the kernel methods such as KPCA K-means and Kernel K-means. Indeed, these results alone do not give us much advantage over the other methods. However, we now present the main advantage of our proposal which is a direct consequence of applying random projection.

4.5.4 Run Time Comparative Analysis

Table 4.7 presents the average run time of the individual approach on each dataset. One of the striking observations from this table is that our proposed approaches have very fast run times. In some cases (*i.e.,* Ballet, UCSD and UCF101 datasets) they outperform the LogE which is expected to be the fastest method. The bottleneck suffered by LogE in these datasets is from the high dimensionality of the feature vectors significantly slowing the K-means algorithm. Note that, although the run time of LogE on Brodatz, KTH-TIPS2b and HEp-2 Cell ICIP2013 dataset is quicker than our proposed methods, the clustering quality shown in Tables 4.4, 4.5 and 4.6 is much worse than that of ours.

The proposed approaches are considerably faster than the kernel approaches such as KPCA K-means and Kernel K-means. This is because the proposed approaches only compute the kernel matrix on a small subset of data points. The benefit will become more pronounced for large datasets such as KTH-TIPS2b, UCF101 and HEp-2 Cell ICIP2013 datasets where our proposed approach achieves 57.5 (*i.e.,* $\frac{675.75}{11.75} \approx 57.5$), 19.8 (*i.e.,* $\frac{2019.55}{101.87} \approx 19.8$) and 122.5 times (*i.e.,* $\frac{2172.87}{17.73} \approx 112.5$) speed up, respectively. Thus, the proposed approaches will contribute significantly to the clustering of large amount of images or video data for practical applications.

The speed up gained by the proposed approaches is attributed to the effect of applying random projection into a reduced projection space. The proposed approaches also have additional advantages over the kernel approaches as they do not need to compute the kernel matrix on the entire dataset.

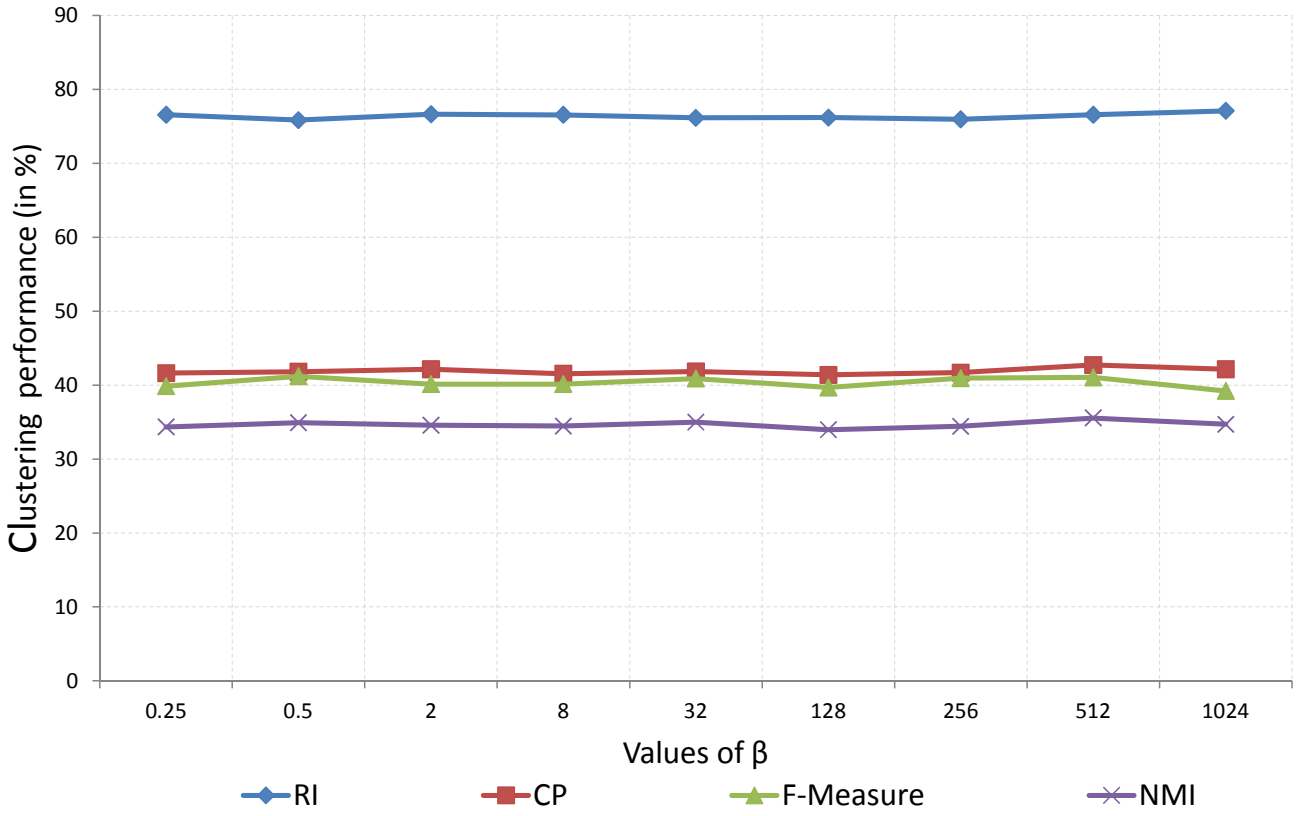


Figure 4.5: Clustering quality of the proposed KPCA-RP when the kernel parameter β was varied on the Ballet dataset. The clustering quality is measured by: Rand Index (RI), Cluster Purity (CP), F-Measure and Normalized Mutual Information (NMI).

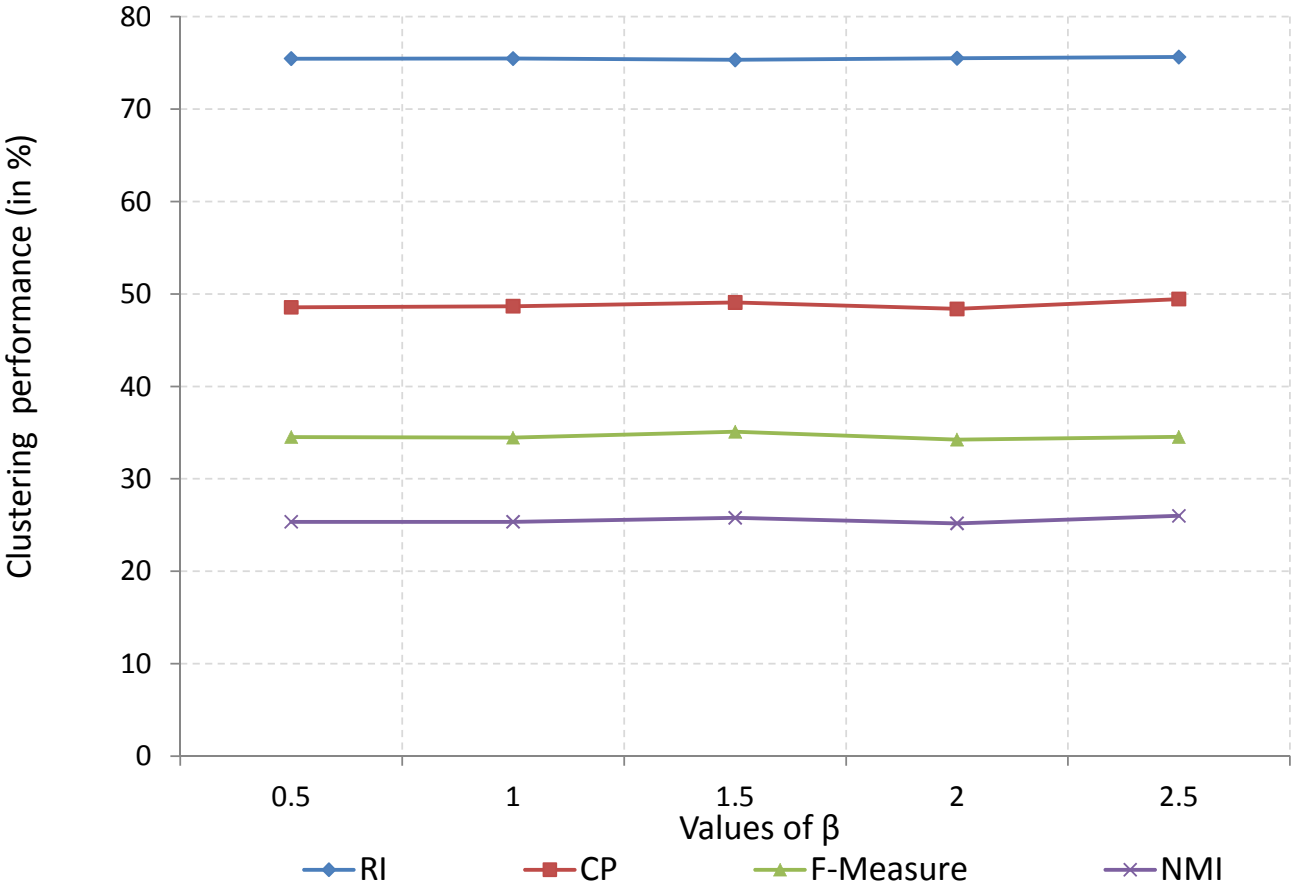


Figure 4.6: Clustering quality of the proposed KORP when the parameter β is varied on the HEp-2 Cell ICIP2013 dataset. The clustering quality is measured by: Rand Index (RI), Cluster Purity (CP), F-Measure and Normalized Mutual Information (NMI).

In addition, we analyse the computational complexity of each method in Table 4.8. In general, each method has two main steps: (1) Data pre-processing and (2) K-means steps. Data pre-processing may include kernel computation and/or projection. Whilst, K-means step comprises cluster membership and cluster mean computations. In Intrinsic K-means, the pre-processing step is not required. To calculate mean of each cluster, one need to use the intrinsic mean, denoted Riemannian centre of mass [80, 110] that requires multiple iterations to converge. The intrinsic distance is also used for membership computation. For LogE, each manifold point needs to be projected onto the Log-Euclidean space. This projection is done once. Then, K-means is applied in the Log-Euclidean space. The computational complexity of KPCA and Kernel K-means follows quadratic and cubic growth, respectively. However, our proposed methods have linear growth, as the number of data points, n, is much bigger than the size of subset, p. This further corroborates the results presented in Table 4.7.

4.5.5 Further Analysis

In this section, we analyse the parameters contributing to the performance and run time of the proposed methods. Due to space limitations, we only show the performance measured by RI and CP. Note that the performance measured by F-Measure and NMI also follows the same trends. An obvious pa-

Table 4.7: The run time (in seconds) of the approaches on each dataset. Lower run time is better. As in each iteration of K-means, the run time is extremely similar, we report the average run time of each approach without variance. The datasets presented in the first three columns (*i.e.,* Ballet, UCSD and UCF101) are modelled in Grassmann manifolds, whilst the other three (*i.e.,* Brodatz, KTH-TIPS2b and HEp-2 Cell ICIP2013 (shorten as Cell)) are modelled in SPD manifolds. The last three rows are the proposed approaches. SIS and G-clustering are only applicable for SPD manifolds and Grassmann manifolds, respectively. We refer to Section 4.5.2 for further explanation of each approach.

Methods/Dataset	Ballet	UCSD	UCF101	Brodatz	KTH-TIPS2b	Cell
Intrinsic [139]	3966.49	1990.02	1.64×10^5	24.63	938.95	564.49
SIS [72]	N/A	N/A	N/A	4.77	60.43	185.81
LogE [48]	3.35	1.55	9088.11	**0.15**	**4.85**	**2.32**
G-clustering [127]	2.81	0.74	N/A	N/A	N/A	N/A
Kernel K-means [40, 76]	1.06	0.70	2019.55	22.57	675.75	2172.87
KPCA [122, 76]	1.47	0.73	6.11×10^4	22.42	699.34	2881.10
KGRP	**0.51**	0.53	238.64	7.08	14.61	21.95
KORP	0.58	**0.49**	**101.87**	7.03	11.75	17.73
KPCA-RP	0.60	**0.49**	102.79	7.75	12.28	17.73

Table 4.8: Computational Complexity of the approaches on each dataset. The dimensionality of SPD and Grassmann points is $d \times d$ and $q \times d$, respectively. For convenience, $\mathcal{G}$ is used to represent Grassmann manifold in this table. Note that: n is the number of points; m is the number of clusters; ℓ is the number of iterations of K-means; ℓ_{kar} is the number of iterations of the Riemannian centre of mass; b is the dimensionality of the random projection space generated by KGRP and p is the dimensionality of the random projection space generated by KORP and KPCA-RP ($p = |\mathcal{S}|$).

	Compute Kernel	Compute Projection	Compute Mean	Compute Membership	Overall Complexity
Intrinsic(SPD) [139]	N/A	N/A	$O(\ell\ell_{kar}nd^3)$	$O(\ell nmd^3)$	$O(\ell\ell_{kar}nd^3 + \ell nmd^3)$
Intrinsic($\mathcal{G}$) [139]	N/A	N/A	$O(\ell\ell_{kar}n(qd^2 + d^3))$	$O(\ell nm(qd^2 + d^3))$	$O((\ell\ell_{kar}n + \ell nm)(qd^2 + d^3))$
SIS [72]	N/A	$O(nd^3)$	$O(\ell nd^2)$	$O(\ell nmd^3)$	$O(\ell nmd^3)$
LogE(SPD) [48]	N/A	$O(nd^3)$	$O(\ell nd^2)$	$O(\ell nmd^2)$	$O(nd^3 + \ell nmd^2)$
LogE($\mathcal{G}$) [48]	N/A	$O(nqd^2)$	$O(\ell nqd)$	$O(\ell nmqd)$	$O(nqd^2 + \ell nmqd)$
G-clustering [127]	$O(n^2)$	$O(n^3)$	$O(\ell n^2)$	$O(\ell n^2 m)$	$O(n^3 + \ell n^2 m)$
Kernel K-means [40, 76]	$O(n^2)$	N/A	N/A	$O(\ell n^2 m)$	$O(\ell n^2 m)$
KPCA [122, 76]	$O(n^2)$	$O(n^3)$	$O(\ell n^2)$	$O(\ell n^2 m)$	$O(n^3 + \ell n^2 m)$
KGRP	$O(np)$	$O(p^3 + np^2)$	$O(\ell npb)$	$O(\ell nmb)$	$O(np + p^3 + np^2 + \ell nmb)$
KORP	$O(np)$	$O(p^3 + np^2)$	$O(\ell np)$	$O(\ell nmp)$	$O(np + p^3 + np^2 + \ell nmp)$
KPCA-RP	$O(np)$	$O(p^3 + np^2)$	$O(\ell np)$	$O(\ell nmp)$	$O(np + p^3 + np^2 + \ell nmp)$

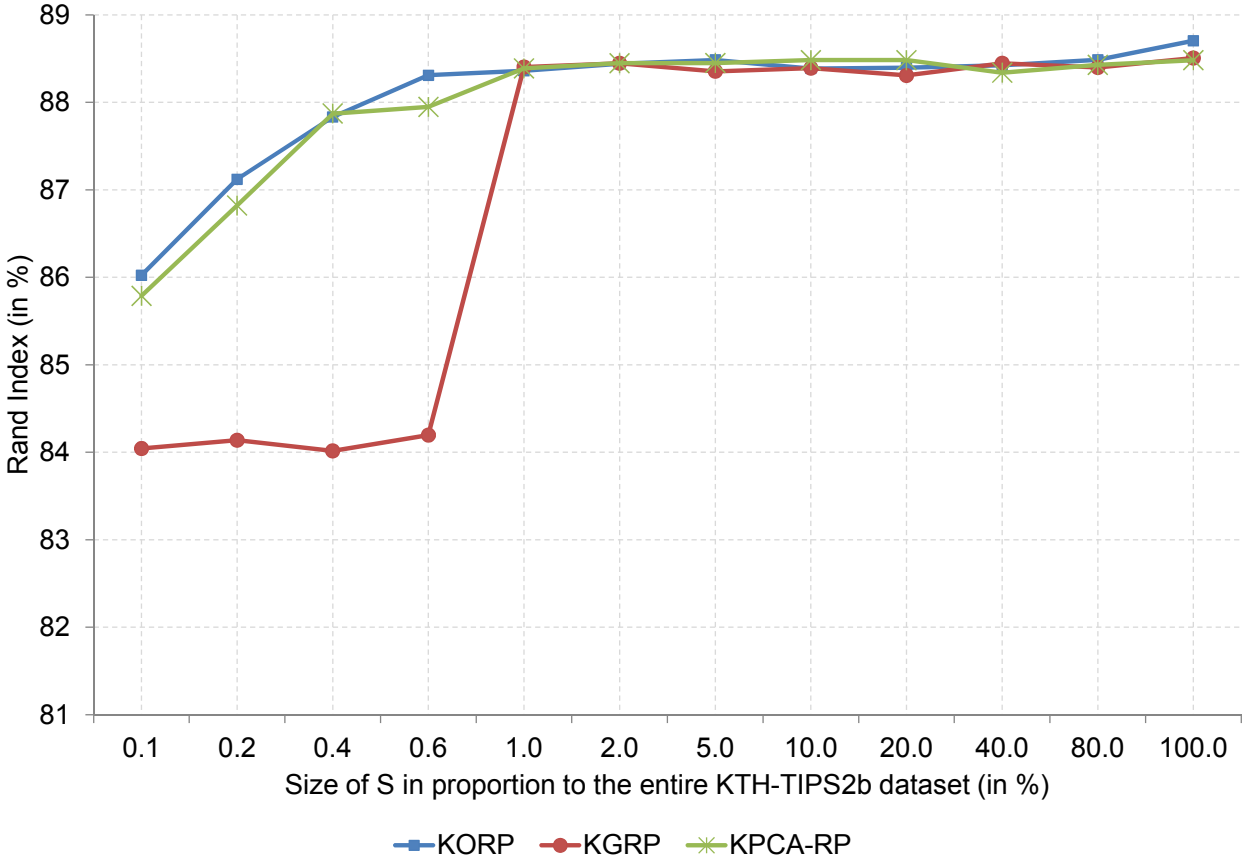


Figure 4.7: The Rand Index (in %) of the proposed approaches when the size of set $\mathcal{S}$ is progressively increased on the KTH-TIPS2b dataset. KGRP: Kernelised Gaussian Random Projection; KORP: Kernelised Orthonormal Random Projection; KPCA-RP: Kernel PCA based Random Projection.

rameter is the projected space dimensionality, k. When k is small, each data point will be represented in a much smaller feature vector, resulting in faster K-means clustering processes. Another parameter is $|\mathcal{S}|$, the size of set $\mathcal{S}$ which determines the run time of the kernel matrix computation. As $|\mathcal{S}|$ gets larger, it takes longer to compute the kernel matrix. Smaller $|\mathcal{S}|$ gives more advantage to the proposed methods over the kernel approaches such as Kernel K-means and KPCA that require kernel computation on the entire data points. We note that k and $|\mathcal{S}|$ have an interesting relationship. More precisely, for KORP and KPCA-RP, $|\mathcal{S}|$ determines the projected space dimensionality, k. Therefore, it is desirable to make $|\mathcal{S}|$ as small as possible whilst still preserving as much of the pairwise distance.

In contrast to KORP and KPCA-RP, KGRP separates the projected space dimensionality to $|\mathcal{S}|$. Nevertheless, we found that $|\mathcal{S}|$ still plays an important role in the overall system performance. To verify this, we vary $|\mathcal{S}|$ on the KTH-TIPS2b. As we can see from Figures 4.7 and 4.8, the performance of the proposed approaches increases as $|\mathcal{S}|$ is progressively increased. The performance increase stops when $|\mathcal{S}|$ reaches a particular value. In this analysis we also found that the performance of KORP and KPCA-RP is markedly better than KGRP when $|\mathcal{S}|$ is considerably small. A possible reason is that the CLT requires the set $\mathcal{S}$ to have a minimum number of elements (normally 30) in order to make the theorem applicable.

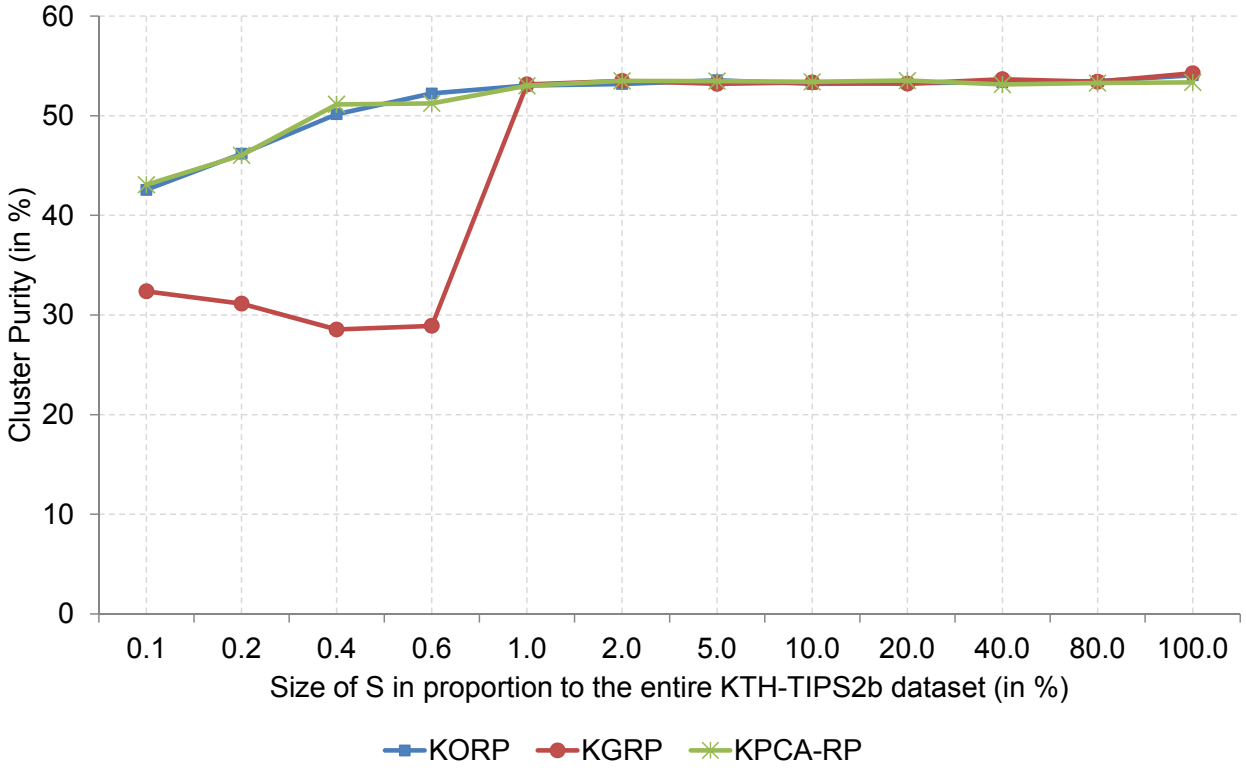


Figure 4.8: The Cluster Purity (in %) of the proposed approaches when the size of set $\mathcal{S}$ is progressively increased on the KTH-TIPS2b dataset. KGRP: Kernelised Gaussian Random Projection; KORP: Kernelised Orthonormal Random Projection; KPCA-RP: Kernel PCA based Random Projection.

The above observation suggests the following facts about $|\mathcal{S}|$:

- $|\mathcal{S}|$ determines the run time for all the proposed approaches (*i.e.,* on the kernel computation);
- $|\mathcal{S}|$ also contributes to the K-means run time for KORP and KPCA-RP;
- The lower bound of $|\mathcal{S}|$ in the KGRP is related to the lower bound of the CLT and (4) the lower bound of $|\mathcal{S}|$ for KORP and KPCA-RP is related to the lower bound of k.

The JL-Lemma relates k to the total number of data points, n (refer to Lemma 4.3.1). This relationship seems unfavourable for KORP and KPCA-RP as this would mean $|\mathcal{S}|$ increases as n increases. Fortunately, Lemma 4.4.1 and Theorem 4.4.3 suggest that k is related to the margin between classes. This means that we now need only consider the separating margin to select $|\mathcal{S}|$. To further corroborate this empirically, we apply the proposed approaches by varying the dataset size of the KTH-TIPS2b. We assume that the margin is relatively unchanged though the dataset size is varied. More precisely, we first fix $|\mathcal{S}|$ for each proposed approach. Then we randomly select the data points from the KTH-TIPS2b to create a smaller version of the dataset. The proposed approaches are applied on these smaller subsets of the dataset. Note that although $|\mathcal{S}|$ is fixed, we still select the elements of $\mathcal{S}$ from the given subset. The results shown in Figures 4.9 and 4.10 suggest that the proposed

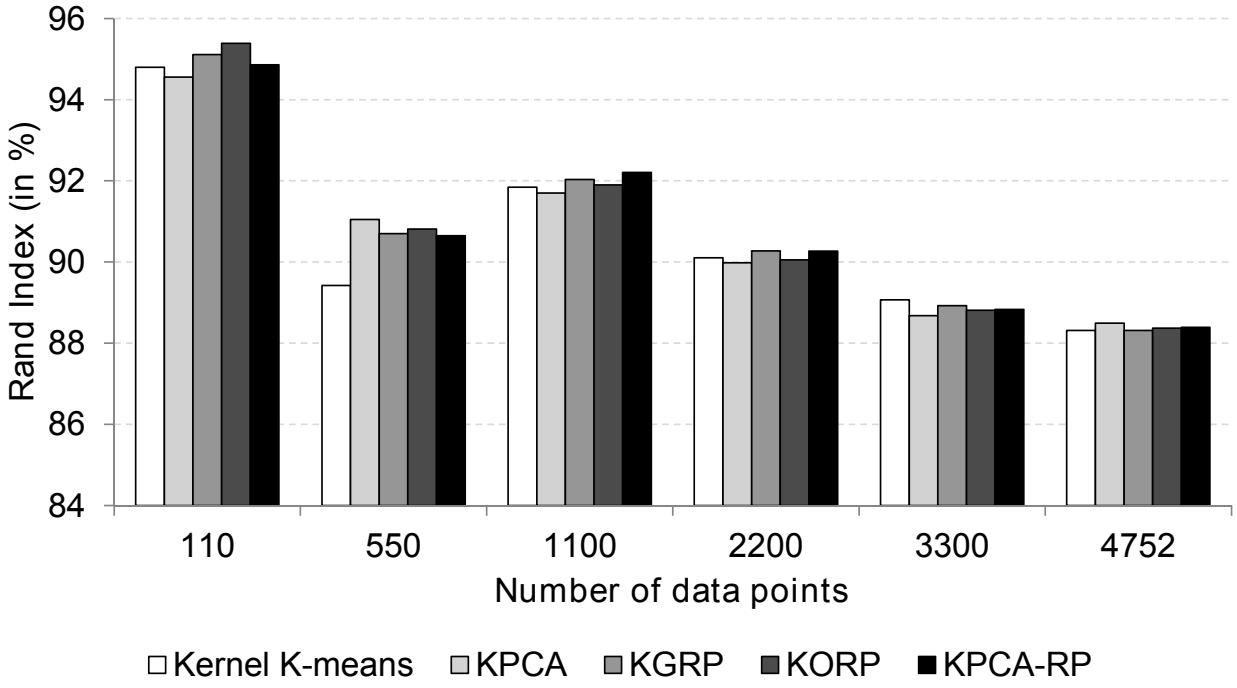


Figure 4.9: The Rand Index (in %) of the proposed approaches, Kernel K-Means and KPCA applied on subsets of KTH-TIPS2b with various sizes. We fix $|\mathcal{S}|$ for all subsets.

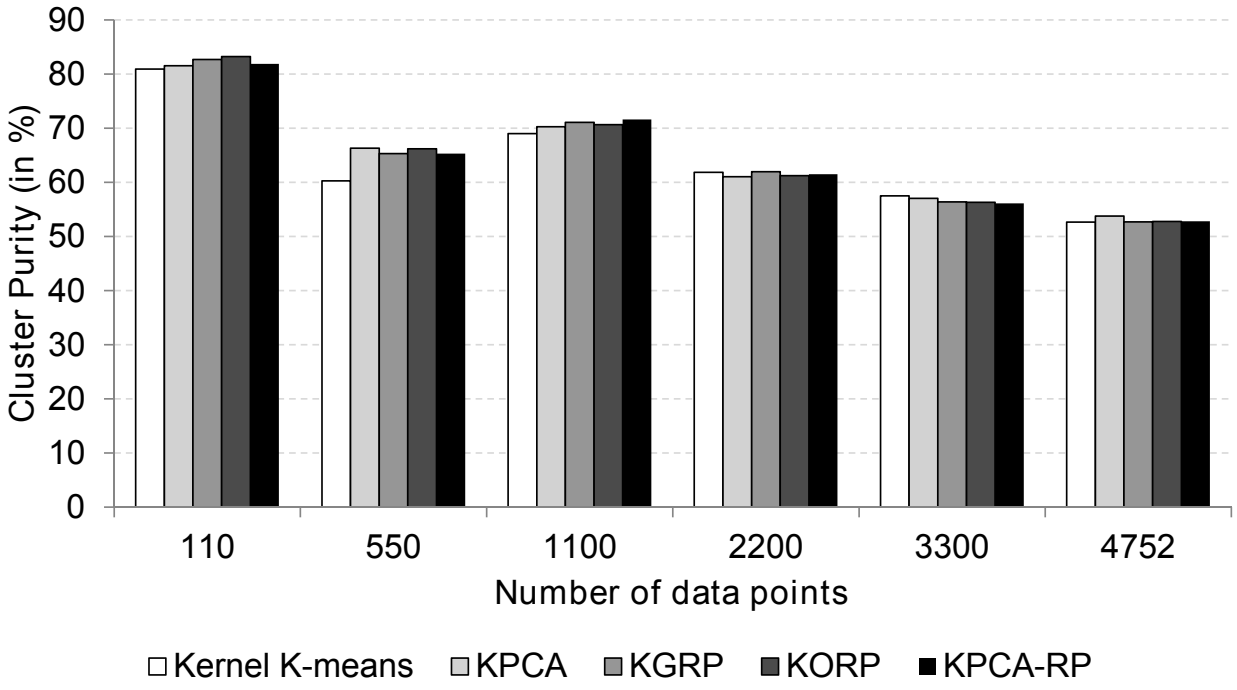


Figure 4.10: The Cluster Purity (in %) of the proposed approaches, Kernel K-Means and KPCA applied on subsets of KTH-TIPS2b with various sizes. We fix $|\mathcal{S}|$ for all subsets.

approaches still have on par performance with both Kernel K-means and KPCA K-means, suggesting that $|\mathcal{S}|$ relates to the margin separation between classes.

4.6 Experimental Results on Classification Tasks

We note that our random projection framework also can be used for classification tasks, where one can randomly select the subset $\mathcal{S}$ from the training data to generate the random hyperplanes. Then one can project all the training and test data onto the random projection space spanned by $\mathcal{S}$. In the random projection framework, the random hyperplanes can be generated by standard Gaussian distribution (KGRP) and orthonormal projections (KORP and KPCA-RP). From the experimental results in Section 4.5, KGRP is inferior to KORP and KPCA-RP when $|\mathcal{S}|$ is extremely small. KORP and KPCA-RP always performed similar. we simply choose KORP to test the orthonormal hyperplanes on the classification tasks. To verify the efficiency of the random projection framework on classification, we validate our proposed method on two applications: gait-based gender recognition using CASIAB dataset [165] and action recognition using UT-Tower dataset [29].

In the experiments, each video of the datasets is considered as an image set which can be effectively modelled as one point on the Grassmann manifold. We introduce the datasets and experimental settings as the following.

4.6.1 Datasets and Feature Extraction

CASIAB gait dataset (CASIAB) — This dataset contains 124 individuals (93 males and 31 females). The gait of each subject was captured from 11 viewpoints (refer to Figure 4.11). There are several video sequences from each subject under one of the 11 viewing angles. Every video is represented by a Gait Energy Image (GEI) [23] of size 32×32. The subspaces of order 3 in each Grassmann manifold point are generated from the Singular Value Decomposition (SVD) of 11 GEIs with different viewpoints. In total, we generate 731 points on $\mathcal{G}_{\mathbf{1024,3}}$. Note that the aim of our experiment is different to the previous work in [94, 168]. In this chapter, the classification task for this dataset is a two-class problem, gait-based gender recognition. We randomly select 10 male and 10 female as training data. The remaining individuals are used for testing. The experiment is repeated 10 times and the average accuracy and run time are reported.

UT-Tower action dataset (UT-Tower) — This dataset consists of 108 video sequences with a frame resolution of 360×240 and a frame rate of 10fps (refer to Figure 4.11). Each of the 9 actions is performed 2 times by 6 individuals, so there are 12 video sequences per action category. For each sequence, we extract 2 Grassmann points. To that end, each sequence is divided into two groups of frames: the first-half and second-half groups. For each group, we downsize the frames to 16×16 and then generate one Grassmann point on $\mathcal{G}_{\mathbf{128,8}}$ by performing the SVD on the normalised pixel intensities of 8 video frames. Thus, there are 216 Grassmann point in total. For the evaluation, we use the standard protocol of this dataset: leave-one-out cross validation where the points from one randomly selected individual are used for testing and the remaining points from the other five individuals are used for training. We randomly select the elements of $\mathcal{S}$ used for generating hyperplanes from the training data. In addition, we ensure that the set $\mathcal{S}$ has elements belong to the first-half and second-half groups. The average accuracy and run time are reported.

4.6.2 Experimental Settings

To evaluate the discriminative power of the orthonormal space generated by the proposed method, we use three popular classifiers: Nearest Neighbour (NN), Support Vector Machine (SVM) and Sparse Representation-based Classifier (SRC) [159]. The proposed orthonormal space is compared with the four types of space:

- Intrinsic space: the intrinsic manifold using the geodesic distance [3];
- Log-Euclidean space: the tangent space at the identity point [139];
- RKHS: the Reproducing Kernel Hilbert Space generated by manifold-based Kernel defined in Eqn. (2.13) [65];
- ROSE: a random projection space generated by standard Gaussian hyperplanes [5].

For KORP and ROSE, we repeat the experiment 10 times, and average accuracy and run time are reported. Each time, we randomly select 80 points for CASIAB dataset and 90 points for UT-Tower dataset to generate the hyperplanes. Thus, the dimensionality of the projected space is 80 and 90 respectively. In the RKHS, we used the kernelised classifiers: Kernel SVM [19, 76] and Kernelised SRC [53]. We use the Kernel presented function in Eqn. (2.13) for both proposed and kernelised methods.

4.6.3 Comparative Analysis

Table 4.9 reports the classification accuracy and the run time of the proposed and several recent methods. The NN in the intrinsic space which is directly working in the manifold, achieves reasonable accuracy. However, the accuracy of NN are generally much worse than the other classification machineries such as SVM and SRC. The methods that in the Log-Euclidean space which is generated by

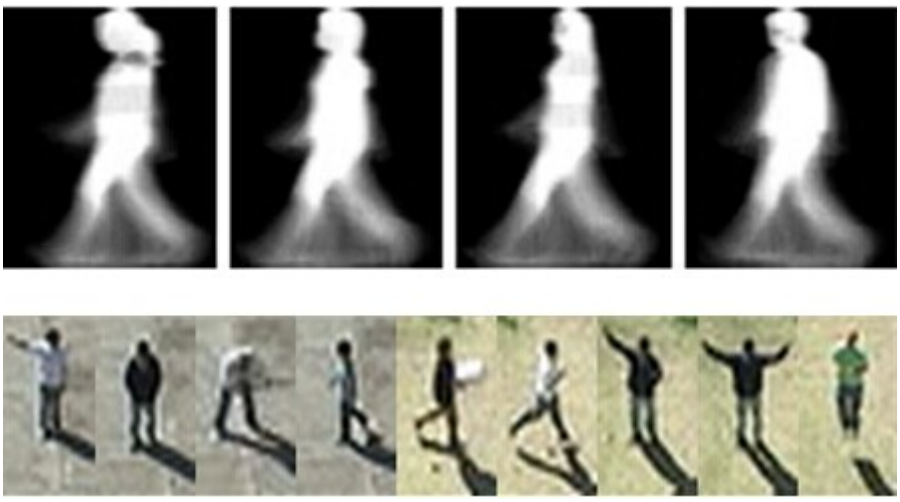


Figure 4.11: Top row: Examples from CASIAB gait dataset [165]. Bottom row: Examples from UT-Tower action dataset [29].

Table 4.9: The average classification accuracy (in %) and run time of the approaches on each dataset. Intrinsic: Intrinsic manifold using Geodesic distances [3]; Log-Euclidean: Log-Euclidean space [139]; ROSE: random projection space with standard Gaussian hyperplanes [5]; RKHS: Reproducing Kernel Hilbert Space generated by manifold-based Kernel defined in Eqn. (2.13) [65]; KORP: our proposed method.

Methods	CAISAB		UT-Tower	
	Accuracy	Run time	Accuracy	Run time
Nearest Neighbour				
Intrinsic	85.70%	4.32	85.65%	0.33
Log-Euclidean	75.81%	14.73	29.17%	**0.03**
ROSE	84.76%	**1.76**	81.72%	0.06
KORP without orthogonality	85.18%	1.89	85.83%	0.06
KORP(proposed)	**86.37%**	2.11	**86.86%**	0.06
Support Vector Machine (SVM)				
Log-Euclidean	87.25%	13.58	47.22%	**0.03**
RKHS	**93.97%**	3.60	90.28%	0.11
ROSE	92.40%	1.89	89.62%	0.06
KORP without orthogonality	89.16%	1.88	88.94%	0.06
KORP(proposed)	93.85%	**1.86**	**91.99%**	0.06
Sparse Representation Classification (SRC) [159]				
Log-Euclidean	85.41%	104.94	43.52%	24.89
RKHS	**94.90%**	385.30	90.74%	28.37
ROSE	92.94%	104.62	90.80%	14.85
KORP without orthogonality	90.75%	**33.97**	92.36%	**11.24**
KORP(proposed)	92.97%	67.85	**92.45%**	14.45

flattening the manifold at the identity point perform poorly due to the distortion of the global manifold structure. Compared to the random projection space generated by ROSE [5], the proposed orthonormal space offers more discriminatory power especially on UT-tower dataset. We conjecture that orthonormal hyperplanes in KORP are more discriminative than the Gaussian hyperplanes generated by ROSE. To evaluate the contribution of the orthonormalisation step, we also report the performance of KORP without that step. Results show that the orthonormalisation process improves the performance distinctly in most cases. The kernelised classifiers in the RKHS obtain appealing accuracy since RKHS usually offers much more discriminative power with infinite features. Likewise, using linear classifiers on the random orthonormal space, KORP achieves a close match of accuracy to kernelised classifiers in the RKHS. This suggests that one can replace the complex kernelised classifiers with much simpler linear ones via the proposed KORP. Additionally, we find that the performance of KORP is not susceptible to different selections of elements in $\mathcal{S}$. The standard deviations of KORP

accuracy (over the 10 repetitions) on CAISAB for NN, SVM and SRC respectively: are 1.97%, 0.72% and 0.39%; UT-Tower: 1.94%, 1.49% and 1.57%.

To further verify the advantage of our proposed method, we also report the run time of each approach in Table 4.9. The run time in the Log-Euclidean space highly depends on the dimensionality of the Grassmann points. Note that the manifold dimensionality of CASIAB and UT-Tower dataset is $\mathcal{G}_{1024,3}$, $\mathcal{G}_{128,8}$. Thus, algorithms in the Log-Euclidean space for CASIAB dataset are much slower than UT-Tower dataset. Compared to the Log-Euclidean and ROSE, our proposed method KORP requires similar or slightly longer run time in some cases, however, it generally achieves much better accuracy. In contrast to the kernelised classifiers in the RKHS whose accuracy are comparable with ours, we find that our proposed method achieves significant speed up, which is nearly 3-fold on average. Thus, our proposed method offers an excellent trade-off between accuracy and run time; important factors for video surveillance applications.

4.7 Summary

In this chapter, we proposed a novel framework with random projection to tackle the learning problems over Riemannian manifolds. Based on the framework, we presented three random projection methods for manifold points: KGRP, KORP and KPCA-RP. Through experiments on several computer vision applications, we demonstrated that our proposed framework achieved significant speed increases while maintaining clustering performance in comparison to the other conventional methods such Kernel K-means. Furthermore, we analysed the parameters that impact the performance and run time of our proposed methods. As the random projection framework can also be used for the classification tasks, we additionally tested our proposed methods on several classification applications, which also exhibited significant speed up without obvious accuracy loss comparing to the kernelised classifiers. Chapter 5 continues by proposing the convex hull framework that following the Component-2 in the manifold scheme (refer to Figure 3.1).

Chapter 5

Convex Hull on Symmetric Positive Definite Manifolds

> You can never solve a problem on the level on which it was created.
>
> Albert Einstein

Chapter Summary: *To improve the performance and computational time of geodesic nearest neighbour, we extend the Euclidean convex hull model to the manifold. Also, fast approximated solutions to the nearest convex hull problem in manifold are proposed.*

5.1 Overview

In this chapter, we extend the nearest convex hull classification method to Symmetric Positive Definite (SPD) manifolds. Specifically, we first review the nearest convex hull classifier in Euclidean space. Then, we present the proposed nearest convex hull classifier over SPD, which utilises a novel mathematical framework, named Manifold Convex Hull (MACH). To solve the optimisation problem posed by the MACH, we propose three solutions. The intrinsic solution MACH-1 achieves superior performance than several recent methods. To further solve the computational issue of MACH-1, we propose approximated solutions MACH-2 and MACH-3, which possess lower computational load without significant accuracy loss.

5.2 Introduction

SPD manifold features have been shown to have excellent performance in a number of image/video classification tasks. Unfortunately, SPD manifolds naturally possess non-Euclidean geometry, so existing Euclidean machineries cannot be used directly. The simplest yet most popular classifier in SPD manifolds is the Geodesic Nearest Neighbour. This chapter studies possible ways to improve

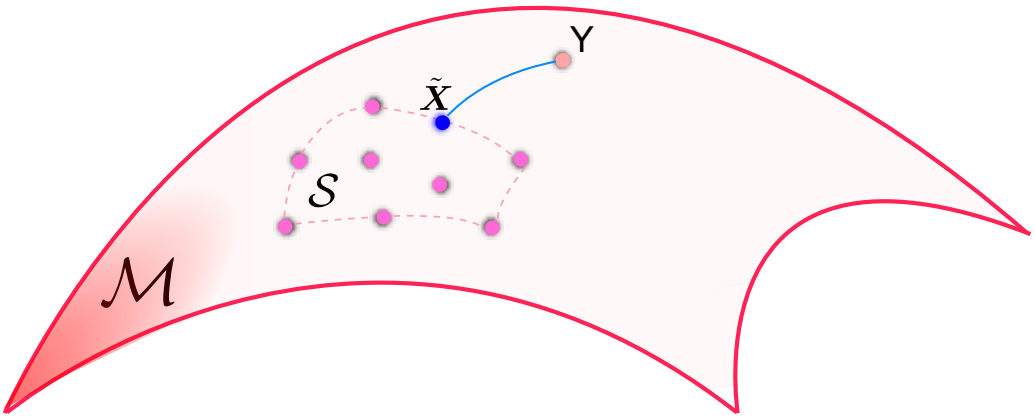


Figure 5.1: Illustration of the proposed Manifold Convex Hull (MACH) approach. Given a set of SPD points $\mathcal{S}$, we construct a convex hull by the convex combinations of these points under certain conditions. To classify a SPD point $\boldsymbol{Y}$, we compute the smallest distance between the convex hull and the point $\boldsymbol{Y}$. This distance can be viewed as the distance between the point $\boldsymbol{Y}$ and $\tilde{\boldsymbol{X}}$ which is generated from a convex combination of points in $\mathcal{S}$.

the accuracy and computational complexity of the Geodesic Nearest Neighbour. As discussed in Section 5.3.2, the Geodesic Nearest Neighbour suffers from various issues.

One possible option to address the classification issues is by modelling a set of points over the manifold as a convex hull, which has been successfully applied in the Euclidean space [25, 30, 73, 106, 115, 169]. A convex hull models all possible convex linear combinations of the samples in a hull, which is shown to be less sensitive to noise [25]. However, due to the non-linear topological structure, it is not clear how to perform convex hull analysis over the manifolds.

In this chapter, we propose to perform convex hull analysis on the SPD manifolds. To our knowledge this is the first time convex hull analysis is studied in the context of SPD manifolds.

We propose convex hull analysis on the SPD manifold by the following:

- We present a novel mathematical framework, here called Manifold Convex Hull (MACH) to solve classification tasks on manifolds; especially, we define the convex combinations and the nearest convex hull distance over the SPD manifold;
- We propose three optimisation methods MACH-1, MACH-2 and MACH-3 for the proposed framework.
- In the experiments, we show that the proposed nearest convex hull classifiers significantly outperform the other intrinsic classifiers and have competitive performance to the state-of-the-art methods.

5.3 Nearest Convex Hull Classification in Euclidean Space

In this section, we will first provide a brief introduction of convex hull model in the Euclidean space. Then the conventional formulation for nearest convex hull classification will be presented.

5.3.1 Convex Hull Model

Given a set of points $\mathcal{S} = \{\boldsymbol{x}_i\}_{i=1}^N$, $\boldsymbol{x}_i \in \mathbb{R}^d$, the convex hull model of $\mathcal{S}$, is the intersection of all half-spaces which contain $\mathcal{S}$. Alternatively speaking, the convex hull of $\mathcal{S}$ includes all convex combinations of S. One can construct a convex hull $\mathcal{C}$ by:

$$\mathcal{C} = \left\{ \sum_{i=1}^{N} w_i \boldsymbol{x}_i \right\}, \sum_{i=1}^{N} w_i = 1, w_i \geq 0. \tag{5.1}$$

The convex hull distance of a point $\boldsymbol{y}$ to the set $\mathcal{S}$ is defined as the minimal distance of $\boldsymbol{y}$ to any point in the convex hull $\mathcal{C}$ constructed from $\mathcal{S}$ by the following:

$$\begin{aligned} \mathrm{d}^2_{cvx}(\boldsymbol{y}, \mathcal{S}) &= \min_{\tilde{\boldsymbol{x}} \in \mathcal{C}} ||\boldsymbol{y} - \tilde{\boldsymbol{x}}||_2^2 \\ &= \min_{w_i} ||\boldsymbol{y} - \sum_{i=1}^{N} w_i \boldsymbol{x}_i||_2^2, \\ &\quad s.t. \sum_{i=1}^{n} w_i = 1 \ , w_i \geq 0, i = 1, 2, ..., n \ . \end{aligned} \tag{5.2}$$

The solution to the above optimisation problem is the closest distance between $\tilde{\boldsymbol{x}} = \sum_{i=1}^{N} w_i \boldsymbol{x}_i$, that belongs to the convex set generated from $\mathcal{S}$, and the query point $\boldsymbol{y}$.

The convex hull model has been widely used in numerous real world applications, for instance, geographical information systems, robot navigation and micromagnetic modelling [12]. In the computer vision community, convex hull model can also be used to address classification problems [25, 30, 73, 115, 169].

5.3.2 Nearest Convex Hull Classification

To tackle the classification tasks using convex hull model, each class is modelled as the smallest convex set representing the distribution of the training data [106]. Jiang *et al.* [115] and Zhou *et al.* [169] used the convex hull of all points in each class to represent the class. On the other hand, Vincent *et al.* [149] proposed to use k-local convex hulls of each class. Recently, the convex hull model also shows superior performance for image set classification tasks [25, 30, 73], in which one can use geometric distances to compare the sets. Some properties of the nearest convex hull classifiers are attractive [25, 115]:

- Similar to the nearest neighbour classifier, the nearest convex hull classifier assigns the label of the test point to a given class without any information from other classes;
- It is robust to the noise and less sensitive to small number of training samples;
- Convex hulls are affine invariant— this means the models do not change when affine transformation is applied on their samples;

- Compared to the nearest neighbour classifier, the convex hull classifier provides infinite samples that constructed by linear combinations of training samples, whereas nearest neighbour only uses a limited training set;
- Compared to SVM, the convex hull classifier can be directly used for multi-class problems — SVM usually needs to decompose multi-class problems into many two-class problems.

For a m-class classification problem, let $\boldsymbol{y}$ be a query and $\mathcal{S}_i$ be the set of samples from i-th class, the classifier finds the nearest convex hull as follows:

$$\min_i \mathrm{d}^2_{cvx}(\boldsymbol{y}, \mathcal{S}_i) \ , i \in \{1, ..., m\} \ . \tag{5.3}$$

The nearest convex hull classifier is similar to the nearest neighbour classifier, although there is a difference in how the training samples are represented. In the nearest neighbour classifier, the training samples are represented explicitly, whereas in the nearest convex hull classifier, most of training samples are implicitly represented by convex combinations of the set of training samples.

5.4 Manifold Convex Hull (MACH) for SPD Manifold

In this section, we propose the **MA**nifold **C**onvex **H**ull for SPD manifold (MACH). First, the formulations related with manifold convex hull are introduced. Then we further provide three practical solutions to use convex hull for classification tasks on SPD manifolds.

5.4.1 Formulations for Manifold Convex Hull

As mentioned in Section 5.3.1, the convex hull of a set of points $\mathcal{S}$ in Euclidean space can be constructed by the intersection of all half-spaces that contain $\mathcal{S}$. However, It is not trivial to generalise this convex hull construction process to the SPD manifold due to the space curvature. SPD manifolds generally do not admit half-spaces.

Fortunately, one can use the horoball as the replacement of half-space on the SPD manifolds [51]. The horoball is a ball fixed at a point, whose radius is allowed to grow to infinity. In Euclidean space, the horoball actually produces a half-space passing through the fixed point. In SPD manifold, the horoball is not flat due to the curvature of this space. However, it is guaranteed to be convex and closed, acting as an effective proxy for the half-space to define the construction of the convex hull in SPD manifolds. To that end, according to Fletcher's work [51], we can use the following theorem to present the existence of convex hull in the SPD manifold.

Theorem 5.4.1 *Given a group of points $\mathcal{S}$ on the SPD manifold, $\mathcal{S} = \{\boldsymbol{X}_i\}_{i=1}^N$, $\boldsymbol{X}_i \in Sym_d^+$, the intersection of all horoballs containing $\mathcal{S}$ would yield a ball hull. The ball hull can be regarded as a generalisation of Euclidean convex hull to SPD manifolds.*

Proof. We refer the readers to [51] for the proof of this theorem.

Remarks. Figure 5.2 illustrates how the horoballs can be used to construct a ball hall. The above theorem shows that this concept can be generalised into the SPD manifold.

To use the convex hull as a geometric tool to perform the classification tasks, there are two issues to be addressed: (1) how to the mathematical definition of the convex hull model on SPD manifolds, and (2) for a query point $\boldsymbol{Y}$, how to compute the distance between $\boldsymbol{Y}$ and the convex hull model.

Following the convex hull conception in Euclidean space, we first give the general mathematical definition of convex hull on SPD manifolds as follows:

Definition 5.4.2 *The convex hull model of a group of points $\mathcal{S} = \{\boldsymbol{X}_i\}_{i=1}^N$, $\boldsymbol{X}_i \in Sym_d^+$, is all the convex combinations of these points and formulated as:*

$$\mathcal{C}_{SPD} = \bigoplus_{i=1}^{N} w_i \boldsymbol{X}_i \,, \tag{5.4}$$

where $w_i \in \mathbb{R}$ and $\bigoplus$ is the convex combination operator over SPD manifolds.

The definitions of convex combination operator in a metric space can be found in Terán's work [135]. Specially, for SPD manifolds, we can represent a convex combination of the given points by the results minimising a weighted sum of squared distances [54, 135]. Thus, the convex hull model then can be illustrated using the following theorem:

Theorem 5.4.3 *The weighted sum of squared distances allows the construction of the convex hull $\mathcal{C}_{SPD}$ for the group of points $\mathcal{S} = \{\boldsymbol{X}_i\}_{i=1}^N$, $\boldsymbol{X}_i \in Sym_d^+$:*

$$\mathcal{C}_{SPD} = \left\{ \tilde{\boldsymbol{X}} | \forall \tilde{\boldsymbol{X}} \in \arg\min_{\tilde{\boldsymbol{X}}} \sum_{i=1}^{N} w_i d_g^2(\boldsymbol{X}_i, \tilde{\boldsymbol{X}}),\ s.t. \sum_{i=1}^{N} w_i = 1, w_i \geq 0 \right\} , \tag{5.5}$$

where d_g is the geodesic distance and $w_i \in \mathbb{R}$ is the weight of the i-th point $\boldsymbol{X}_i$.

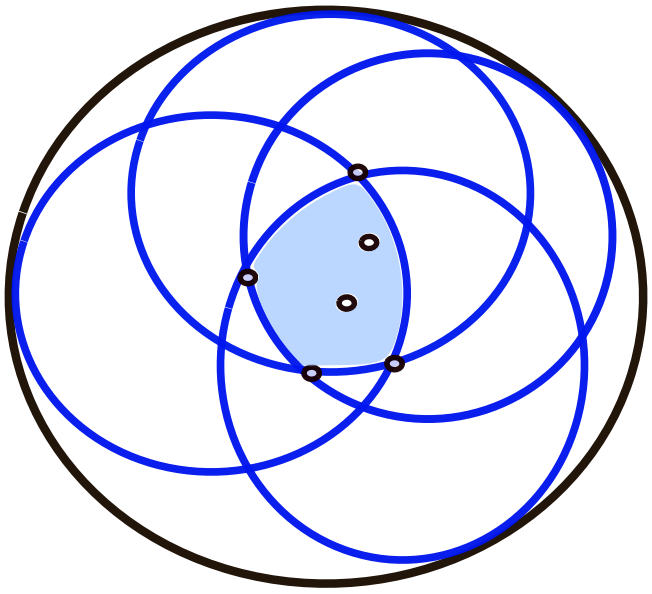


Figure 5.2: An illustration of a ball hull constructed by the intersection of horoballs [51].

Proof. We refer the readers to [54] for the proof of this theorem.

This theorem shows that the convex hull of $\mathcal{S}$ is the set of all $\tilde{\boldsymbol{X}}$ that minimised the weighted sum of squared distances to each point $\boldsymbol{X}_i$ within $\mathcal{S}$. After we present the mathematical definition of convex hull in SPD manifold, we then need to define how to compute the distance between a query point $\boldsymbol{Y}$ and a convex hull model $\mathcal{C}_{SPD}$.

Note that the convex hull is the smallest convex set including the given set of points. The distance between a query point $\boldsymbol{Y}$ and a convex hull model $\mathcal{C}_{SPD}$ can be defined as the geodesic distance from a point to a convex set in the manifold [145]:

$$\mathrm{d_{cvx}}^2(\boldsymbol{Y}, \mathcal{C}_{SPD}) \quad = \quad \min \mathrm{d_g}^2(\boldsymbol{Y}, \tilde{\boldsymbol{X}}), \forall \tilde{\boldsymbol{X}} \in \mathcal{C}_{SPD} \; . \tag{5.6}$$

Eqn. (5.6) states that the distance between a query point $\boldsymbol{Y}$ and a convex hull model $\mathcal{C}_{SPD}$ is the smallest geodesic distance between $\boldsymbol{Y}$ and points in $\mathcal{C}_{SPD}$.

Now, we are ready to define the nearest convex hull classifier on SPD manifolds. Similar to Euclidean space, given a m-class classification problem, the nearest convex hull classifier on SPD manifolds can be formulated as:

$$F(i) = \min_i \mathrm{d}^2_{cvx}(\boldsymbol{Y}, \mathcal{C}^i_{SPD}) \; , \tag{5.7}$$

where $\mathcal{C}^i_{SPD}$ defined in Eqn. (5.5), is the convex hull model for training class i ($i = 1, ..., m$,, where m is the number of classes). To classify a query point $\boldsymbol{Y}$, $\boldsymbol{Y} \in Sym^+_d$, one need to compute the distance between $\boldsymbol{Y}$ and each convex hull $\mathcal{C}^i_{SPD}$. Finally, the query point $\boldsymbol{Y}$ would be assigned the label of the training class wherein the distance between the convex hull $\mathcal{C}^i_{SPD}$ and $\boldsymbol{Y}$ is the nearest.

In the following sections, we present three solutions to address the above optimisation problems. Specifically, we first present a solution that intrinsically solved the convex hull classification problem on SPD manifolds. Then, for the consideration of efficiency, we propose two approximate solutions based on different relax conditions.

5.4.2 Convex Hull Distance Based on Riemannian Centre of Mass — MACH-1

To solve the optimisation problem in Eqn. (5.7), one need to compute the distances between the query point $\boldsymbol{Y}$ and all the convex hull models for the dataset. To that end, as defined in Eqn. (5.6), one need to find the point $\tilde{\boldsymbol{X}}$ that from the convex hull model, which has the minimum distance with the query point $\boldsymbol{Y}$. where $\tilde{\boldsymbol{X}}$ is defined in Eqn. (5.5). If the weights $\{w_i\}_{i=1}^N$ is fixed to some certain values, $\tilde{\boldsymbol{X}}$ will become a weighted Riemannian centre of mass of the given group of points [105]. Thus, convex hull model defined in Eqn. (5.5) contains all the possible weighted Riemannian centre of mass of the group of points, which is generated by varying the weights $\{w_i\}_{i=1}^N$.

According to the solution of weighted Riemannian centre of mass that adapted the Gauss-Newton scheme [111], to obtain the nearest point $\tilde{\boldsymbol{X}}$, the following equation can be achieved in the t-th iteration of the optimisation process:

$$\tilde{\boldsymbol{X}}_{t+1} = \exp_{\tilde{\boldsymbol{X}}_t}(\sum_{i=1}^{N} w_i \log_{\tilde{\boldsymbol{X}}_t}(\boldsymbol{X}_i)) \tag{5.8}$$

When the convergence reaches, it implies $\tilde{\boldsymbol{X}}_{t+1} \approx \tilde{\boldsymbol{X}}_t$. Thus,

$$\sum_{i=1}^{N} w_i \log_{\tilde{\boldsymbol{X}}_t}(\boldsymbol{X}_i) \approx 0 \tag{5.9}$$

Since $\tilde{\boldsymbol{X}}_{\boldsymbol{t}}$ is the nearest point to the query point $\boldsymbol{Y}$, one can approximately replace $\tilde{\boldsymbol{X}}_{\boldsymbol{t}}$ with $\boldsymbol{Y}$ in Eqn. (5.9). This implies that the sum of the tangent vectors $\sum_{i=1}^{N} w_i \log_{\boldsymbol{Y}}(\boldsymbol{X}_i)$ should be closed to zero. For the sake of notational simplicity, we use $\boldsymbol{L}_i = \log_{\boldsymbol{Y}}(\boldsymbol{X}_i)$, which is the Riemannian logarithmic map $\log_{\boldsymbol{Y}}(\boldsymbol{X}_i) : \mathcal{M} \mapsto T_{\boldsymbol{Y}}(\mathcal{M})$ given by:

$$\log_{\boldsymbol{Y}}(\boldsymbol{X}_i) = \boldsymbol{Y}^{\frac{1}{2}} \log(\boldsymbol{Y}^{-\frac{1}{2}} \boldsymbol{X}_i \boldsymbol{Y}^{-\frac{1}{2}}) \boldsymbol{Y}^{\frac{1}{2}} \tag{5.10}$$

where $\log$ denotes the matrix logarithm. Therefore, we actually can use the following objective function to compute the distance between $\boldsymbol{Y}$ and the convex hull model $\mathcal{C}_{SPD}$:

$$\begin{aligned} d^2_{cvx}(\boldsymbol{Y}, \mathcal{C}_{SPD}) &= \mathrm{d}^2_g(Y, \tilde{\boldsymbol{X}}) \\ &\approx \min_{\{w_i\}_{i=1}^N} \sum_{i=1}^{N} w_i^2 ||\boldsymbol{L}_i||^2_{\boldsymbol{Y}} \\ &= \min_{\{w_i\}_{i=1}^N} \sum_{i=1}^{N} w_i^2 \operatorname{Tr}(\boldsymbol{Y}^{-1} \boldsymbol{L}_i \boldsymbol{Y}^{-1} \boldsymbol{L}_i) \\ &= \min_{\{w_i\}_{i=1}^N} \sum_{i=1}^{N} \operatorname{Tr}(\boldsymbol{Y}^{-1} w_i \boldsymbol{L}_i \boldsymbol{Y}^{-1} w_i \boldsymbol{L}_i)) \,, \\ & s.t. \sum_{i=1}^{N} w_i = 1 \,, w_i \geq 0, i = 1, 2, ..., n \,. \end{aligned} \tag{5.11}$$

where $||\cdot||_{\boldsymbol{Y}}$ denotes the norm induced by the inner product on $T_{\boldsymbol{Y}}(\mathcal{M})$ and $\boldsymbol{L}_i = \log_{\boldsymbol{Y}}(\boldsymbol{X}_i)$ that can be computed by Eqn. 5.10. The optimisation can be addressed by the nonlinear programming solver presented in [22].

5.4.3 Convex Hull Distance Based on a Confined Set — MACH-2

Solving the optimisation in Eqn. (5.11) can be computational expensive. To speed up the optimisation process, we propose an approximate solution by relaxing the construction conditions of the

convex hull model in Eqn. (5.5). Our approximation is based on the following propriety of weighted Riemannian centre of mass (refer to [85] for proof):

$$\tilde{\boldsymbol{X}} = \arg\min_{\tilde{\boldsymbol{X}}} \sum_{i=1}^{N} w_i d_g^2(\boldsymbol{X}_i, \tilde{\boldsymbol{X}}) \preceq \sum_{i=1}^{N} w_i \boldsymbol{X}_i \tag{5.12}$$

Thus, one can utilise $\sum_{i=1}^{N} w_i \boldsymbol{X}_i$ as a relaxed construction of the convex hull model $\mathcal{C}_{SPD}$. However, to ensure the distance function:

$$d_{\text{cvx}}^2(\boldsymbol{Y}, \mathcal{C}_{SPD}) = \min_{\{w_i\}_{i=1}^N} d_g^2(\boldsymbol{Y}, \sum_{i=1}^{N} w_i \boldsymbol{X}_i) \,, \tag{5.13}$$

to be convex, one need further narrow down the set representing the convex hull model to the following set:

$$\mathcal{A} := \left\{ w | \sum_{i=1}^{N} w_i \boldsymbol{X}_i \preceq \boldsymbol{Y}, \sum_{i=1}^{N} w_i = 1 \,, w_i \geq 0 \right\} \,. \tag{5.14}$$

This means that the approximated convex hull model can be constructed by only using the weighted linear combinations of samples which under-determine the query point $\boldsymbol{Y}$.

For the convexity proof of the function in Eqn.(5.13), we refer readers to [31]. Note that, the difference between our work and the work in [31] is that we restrict the sum of w_i equals one and the optimization is based on each class. These two points are the essential requirements of convex hull that gives us much better performance comparing with [31].

Following [31], to solve the above optimization problem on this restricted set, we first rewrite the objective function in Eqn. (5.13) by setting $\boldsymbol{M} = \sum_{i=1}^{N} w_i \boldsymbol{X}_i$:

$$\begin{aligned} \mathrm{f}(\boldsymbol{w}) &= d_g^2(\boldsymbol{Y}, \sum_{i=1}^{N} w_i \boldsymbol{X}_i) \\ &= || \log(\boldsymbol{Y}^{-\frac{1}{2}} \boldsymbol{M} \boldsymbol{Y}^{-\frac{1}{2}}) ||_F^2 \\ &= \mathrm{tr} \left\{ \log(\boldsymbol{Y}^{-\frac{1}{2}} \boldsymbol{M} \boldsymbol{Y}^{-\frac{1}{2}})^\top \log(\boldsymbol{Y}^{-\frac{1}{2}} \boldsymbol{M} \boldsymbol{Y}^{-\frac{1}{2}}) \right\} \,, \end{aligned} \tag{5.15}$$

where $\boldsymbol{w}$ is an N-dimensional vector whose elements are the set of convex linear combination weights $\{w_i\}_{i=1}^N$ and tr is the matrix trace. Then the partial derivative of the above function can be defined as follows:

$$\begin{aligned} \frac{\partial \mathrm{f}}{\partial w_i} =& 2 \,\mathrm{tr} \left\{ \log(\boldsymbol{Y}^{-\frac{1}{2}} \boldsymbol{M} \boldsymbol{Y}^{-\frac{1}{2}}) (\boldsymbol{Y}^{-\frac{1}{2}} \boldsymbol{M} \boldsymbol{Y}^{-\frac{1}{2}})^{-1} \right. \\ & \left. \times \frac{\partial (\boldsymbol{Y}^{-\frac{1}{2}} \boldsymbol{M} \boldsymbol{Y}^{-\frac{1}{2}})}{\partial w_i} \right\} \,, \end{aligned}$$

where

$$\begin{aligned}\frac{\partial(\boldsymbol{Y}^{-\frac{1}{2}}\boldsymbol{M}\boldsymbol{Y}^{-\frac{1}{2}})}{\partial w_i} &= \boldsymbol{Y}^{-\frac{1}{2}}\frac{\partial \boldsymbol{M}}{\partial w_i}\boldsymbol{Y}^{-\frac{1}{2}} \\ &= \boldsymbol{Y}^{-\frac{1}{2}}\boldsymbol{X_i}\boldsymbol{Y}^{-\frac{1}{2}} \ .\end{aligned} \tag{5.16}$$

Substituting Eqn. (5.16) to Eqn. (5.16), we get the following expression:

$$\frac{\partial \mathrm{f}}{\partial w_i} = 2\,\mathrm{tr}\left\{\log(\boldsymbol{Y}^{-\frac{1}{2}}\boldsymbol{M}\boldsymbol{Y}^{-\frac{1}{2}})\boldsymbol{Y}^{\frac{1}{2}}\boldsymbol{M}^{-1}\boldsymbol{X}_i\boldsymbol{Y}^{-\frac{1}{2}}\right\} \ . \tag{5.17}$$

We use a gradient-projection scheme that essentially performs the iteration:

$$\begin{aligned} w_i^{k+1} &= \mathcal{P}\left[w_i^k - \lambda_k \frac{\partial \mathrm{f}}{w_i^k}\right] \ , \\ s.t. \sum_{i=1}^{N} w_i = 1 \ , w_i \geq 0, i = 1, 2, ..., n, w_i \in \mathcal{A} \ , \end{aligned} \tag{5.18}$$

where $\mathcal{P}$ is the projection function operator defined via:

$$\mathcal{P}(w) \equiv w \mapsto \arg\min_{w'} \frac{1}{2}\|w' - w\|_2^2, w' \in \mathcal{A} \ . \tag{5.19}$$

In the implementation, we use the Spectral Projected Gradient method (SPG) [17] implemented by Schmidt *et al.* in [121]. We name this proposed solution as MACH-2.

5.4.4 Convex Hull Distance Based on LE Metric— MACH-3

The geodesic distance defined in Eqn. (2.5) can be computational expensive. Thus, for the third solution, we consider to use approximate metric to compute the distances for Eqn. (5.6) and Eqn. (5.5). Specifically, we use the Log-Euclidean (LE) metric defined in Eqn. (2.6). The advantage of using LE metric is that the calculation could be hundreds of times faster. Also, the weighted mean computed by LE metric have been proven to be similar or even equal to the Riemannian centre of mass $\tilde{\boldsymbol{X}}$(defined in Eqn. (5.5)) in some cases [10]. This fact guarantees that our approximation does not severely affect

the accuracy of the convex hull classifier. We named this solution as MACH-3, where the optimisation problem in Eqn. (5.6) can be rewritten as:

$$\begin{aligned} d^2_{\text{cvx}}(\boldsymbol{Y}, \mathcal{C}_{SPD}) &= \min_{\{w_i\}_{i=1}^N} \left\| \log(\boldsymbol{Y}) - \sum_{i=1}^{N} w_i \log(\boldsymbol{X_i})) \right\|_F^2 \\ &= \min_{\{w_i\}_{i=1}^N} [(\log \boldsymbol{Y} \cdot \log \boldsymbol{Y}) \\ &- 2\sum_{i=1}^{N} w_i (\log \boldsymbol{Y} \cdot \log \boldsymbol{X}_i) \\ &+ \sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j (\log \boldsymbol{X}_i \cdot \log \boldsymbol{X}_j)] \,, \end{aligned} \tag{5.20}$$

where $\log$ is the matrix logarithm, $\log(\boldsymbol{X})$ can be thought as embedding the manifold points onto a tangent space which has Euclidean geometry. We also apply the convexity constraints $\sum_{i=1}^{N} w_i = 1$ and $w_i \geq 0$ to the above optimisation problem. Since Euclidean geometry applies in the space generated by the LE metric, we can simply vectorise the point $\boldsymbol{A} = \log(\boldsymbol{X})$ [111] as:

$$\begin{aligned} \text{Vec}(\boldsymbol{A}) = [a_{1,1}, \sqrt{2}a_{1,2}, a_{2,2}, \sqrt{2}a_{1,3}, \sqrt{2}a_{2,3}, a_{3,3}, \\ ..., \sqrt{2}a_{1,d}, ..., \sqrt{2}a_{d-1,d}, a_{d,d}]^\top \,. \end{aligned} \tag{5.21}$$

Also, the term $\log \boldsymbol{Y} \cdot \log \boldsymbol{Y}$ in Eqn. (5.20) is constant and can be excluded from Eqn. (5.20). Finally, the optimisation problem is rewritten as:

$$\begin{aligned} \min_{\boldsymbol{w}} \boldsymbol{w}^\top \boldsymbol{D} w - 2\,\text{Vec}(\log \boldsymbol{Y})\boldsymbol{D} \\ s.t.\ \boldsymbol{e}^\top \boldsymbol{w} = 1, \boldsymbol{w} \geq \boldsymbol{0} \,, \end{aligned} \tag{5.22}$$

where $\boldsymbol{D} = [\text{Vec}(\log \boldsymbol{X}_1) \cdots \text{Vec}(\log \boldsymbol{X}_n)]$, $\boldsymbol{e} = [1 \cdots 1]^\top$. The above problem is a quadratic optimisation problem which can be solved using a standard quadratic programming solver.

5.5 Experimental Results

We evaluated the proposed manifold convex hull on four computer vision applications:

- Traffic scene classification using UCSD traffic dataset [26];
- Object recognition using ETH80 object dataset [86];
- Texture classification using Brodatz dataset [116];
- Person re-identification using ETHZ dataset [46].

For our comparisons, we considered three baseline classifiers on SPD manifolds: (1) Geodesic Nearest Neighbour (Geo-NN); (2) Kernel SVM (KSVM); and (3) Sparse coding (SPD-Sc).

Geo-NN — We used the intrinsic geodesic distance as presented in Eqn. (2.5).
KSVM— An effective implementation of KSVM, LibSVM [28], was used in conjunction with the Stein divergence kernel [132] which has excellent performance for SPD manifolds [67].
SPD-Sc — Sparse coding with SPD points. Here, the state of the art solution of SPD sparse coding proposed in [31] was used. Once the sparse coefficients were determined, the sparse coding classifier proposed in [159] was used as the classifier.

To further demonstrate the effectiveness of our framework, we also present comparisons with a number of recent methods on each dataset.

5.5.1 Traffic Scene Classification Using UCSD Dataset

The UCSD traffic dataset [26] comprises 254 video sequences collected from the highway traffic in Seattle over two days (see Figure 5.3 for examples). It contains a variety of traffic patterns and weather conditions (*i.e.,*, overcast, raining, sunny). There are three different classes:

- Heavy traffic (44 sequences);
- Medium traffic (45 sequences);
- Light traffic (165 sequences).

Each frame in each sequence was downsized to 140×161 pixels and further normalized by subtracting the mean frame and dividing the variance. Then, we applied the two dimensional Discrete Cosine Transform (DCT) on the frame and used the DCT coefficients as the feature vector for each frame. In the consideration of successive frame variation of each sequence, we generated the SPD manifold features by computing the covariance matrix of 15 successive frames in each video sequence. In particular, the selection was random and repeated 10 times.

We compare the performance of our convex hull methods, MACH-1, MACH-2 and MACH-3 with the baselines, Geo-NN, KSVM and SPD-Sc. Table 5.1 shows our methods outperform all of these baselines. In addition, Table 5.1 also presents the performance of the recent methods such as Linear Dynamical Systems model (LDS) [120], Compressive Sensing LDS (CSLDS) [120], Spatio-temporal Orientation Analysis (SOA) [39] and Non-Linear Stationary Subspace Analysis (DNLSSA) [11]. The accuracy of our methods is considerably better than both LDS and CSLDS. We achieve competitive performance to the state-of-the-art results of DNLSSA and SOA. It is noteworthy to mention that DNLSSA and SOA are considerably more complex. As SOA requires matching distributions of space-time orientation structure and DNLSSA solves an optimisation problem in the kernel space with manifold regularization. Furthermore, our methods do not require any parameter tuning.

5.5.2 Object Recognition Using ETH80 Dataset

The ETH80 dataset [86] contains eight object categories. In each category, there are 10 different object instances. For each object instance, ETH80 provides 41 images of different views as one image set. The typical variations are shown in Figure 5.4.

Figure 5.3: Example frames from the UCSD traffic video dataset [26]. The frames depict various traffic congestion conditions and can be categorized as heavy traffic (bottom row), medium (middle row) and light (top row).

Method	Accuracy
LDS [120]	87.5
CS-LDS [120]	89.1
SOA [39]	**95.2**
DNLSSA - RBF Kernel [11]	94.5
SPD-Sc [31]	90.9
KSVM	93.7
Geo-NN	91.3
MACH-1 (proposed)	94.07
MACH-2 (proposed)	94.5
MACH-3 (proposed)	94.1

Table 5.1: Traffic Scene classification task: average accuracy (in %) on the UCSD traffic video dataset [26].

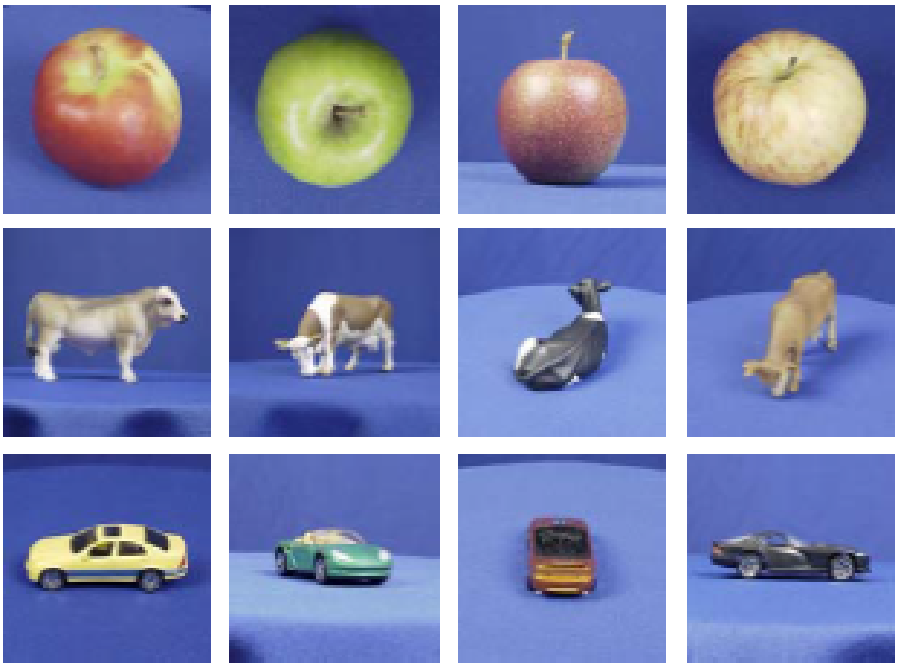


Figure 5.4: Sample images from ETH-80 object dataset [86].

Method	Accuracy
CHISD [25]	77.3
SANP [73]	75.5
SPD-Sc [31]	88.5
KSVM	89.5
Geo-NN	87.3
MACH-1 (proposed)	92.25
MACH-2 (proposed)	90.5
MACH-3 (proposed)	**93.3**

Table 5.2: Object recognition task: average accuracy (in %) on the ETH80 dataset [86].

We applied two dimensional DCT on the images and used the 80-dimensional DCT coefficients as the feature vector for each image. To obtain the SPD features, we then computed the 80×80 covariance matrix of each image set. For each category, we used five randomly chosen instances for training and five for testing. The random selection of train and test was repeated 10 times and the average performance is reported here.

The experimental results for this dataset are summarized in Table 5.2. The maximum accuracy of 93.3% is achieved by MACH-3, which is nearly six percentage points better than Geo-NN. We also compare our proposed methods with Convex Hull based Image Set Distance (CHISD) [25] and Sparse Approximated Nearest Points (SANP) [73]. Again, the proposed methods outperform these methods. Note that CHISD and SANP are Euclidean-based convex hull methods. The performance improvement from CHISD and SANP is significant. This could be attributed to using manifold features and the nearest convex hull classifier to do this classification task.

Method	Accuracy
LE-SR [57]	66.3
TSC [128]	79.8
RLPP [70]	86.1
SPD-Sc [31]	77.6
KSVM	82.6
Geo-NN	84.9
MACH-1 (proposed)	87.98
MACH-2 (proposed)	**88.0**
MACH-3 (proposed)	87.8

Table 5.3: Texture recognition task: average accuracy on the Brodatz texture dataset [116].

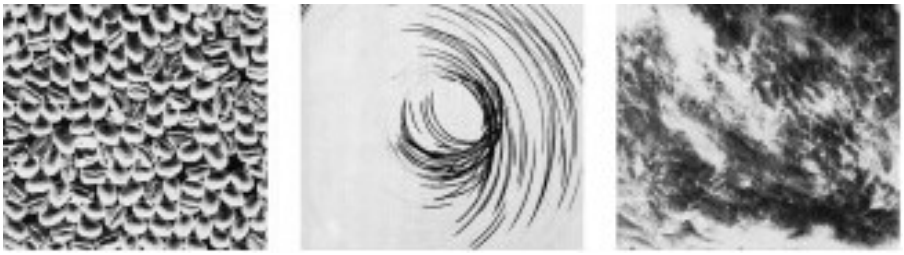


Figure 5.5: Example images from the BRODATZ texture dataset [116].

5.5.3 Texture Classification Using Brodatz Dataset

We followed the protocol presented in [129]. This protocol includes three subsets with different numbers of classes: 5-class-texture (5c, 5m, 5v, 5v2, 5v3); 10-class-texture (10, 10v) and 16-class-texture (16c, 16v).

Each image was resized to 256×256 pixels and divided into 64 regions with size 32×32. A feature vector $F(x, y)$ for each pixel was calculated using the grayscale intensity and absolute values of the first- and second-order derivatives of spatial features:

$$F(x,y) = \left[I(x,y), \left|\frac{\partial I}{\partial x}\right|, \left|\frac{\partial I}{\partial y}\right|, \left|\frac{\partial^2 I}{\partial x^2}\right|, \left|\frac{\partial^2 I}{\partial y^2}\right| \right] . \tag{5.23}$$

Then, each region was represented by a covariance matrix (SPD matrix) formed from these feature vectors. For each scenario, only five SPD points per class were randomly selected as training, the rest were used for test. The random selection of training/testing data was repeated 10 times.

Table 5.3 compares the proposed MACH-1, MACH-2 and MACH-3 to various methods. Note that the number of training data per class is only five. The proposed methods significantly outperform all the baselines. This corroborates the previous findings in the Euclidean space [169], that suggest the nearest convex hull classifier is generally more robust to small number of training samples. Compared to the recent methods such as LE-SR [57], TSC [128] and RLPP [70], the proposed methods perform much better.

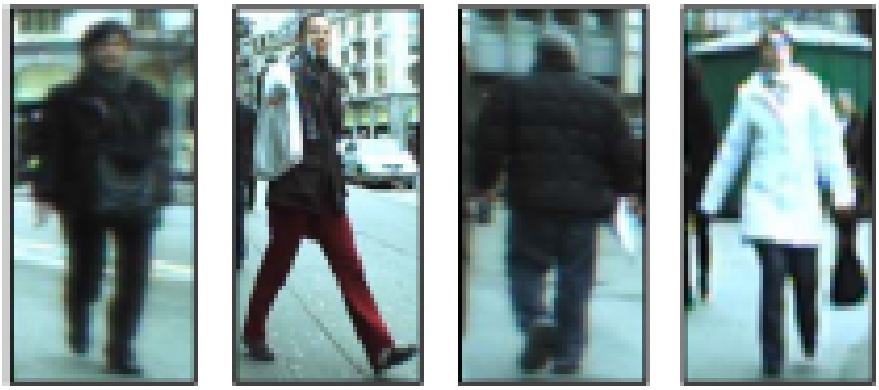


Figure 5.6: Example images from the ETHZ data set [46].

5.5.4 Person Re-identification Using ETHZ Dataset

For the final task we considered the person re-identification task using the modified version [123] of the ETHZ dataset [46]. The dataset was captured from a moving camera, containing wide variations in the appearance of people.

The dataset is divided into three sequences. Sequence 1 contains 83 pedestrians captured in 4,857 images, sequence 2 contains 35 pedestrians captured in 1,936 images, and sequence 3 contains 28 pedestrians captured in 1,762 images. Examples of images are shown in Figure 5.6.

We first resized all the images into 64×32 pixels. Then the SPD features were generated by computing covariance matrix of the pixels feature vectors defined as:

$$F_{x,y}= \left[x, y, R_{x,y}, G_{x,y}, B_{x,y}, R'_{x,y}, G'_{x,y}, B'_{x,y}, R''_{x,y}, G''_{x,y}, B''_{x,y}\right] \ , \tag{5.24}$$

where x and y represent the position of a pixel, while $R_{x,y}$, $G_{x,y}$ and $B_{x,y}$ represent the corresponding colour information, respectively.

In this experiment, 10 randomly selected images per class were used as training set, while the rest were formed as testing. To obtain a reliable average performance, the random selection of training and testing data was repeated 10 times.

The recognition accuracy of different methods is reported in Table 5.4. Our proposed methods are the best two among all methods and achieve considerably better results than the three baselines: Geo-NN, KSVM and SPD-Sc. Furthermore, the proposed methods significantly outperform PLS [46] and HPE [13]. This is could be attributed to the efficacy of our proposed classifier with SPD manifold features.

Comparisons between MACH-1, MACH-2 and MACH-3: As shown in our experiments, MACH-1, MACH-2 and MACH-3 have on par performance in most cases. Table 5.5 shows the running time for the three methods. MACH-1 that requires projection onto the tangent space at each query point is the slowest method. However, the approximation methods, MACH-2 and MACH-3, are much more efficient.

Method	Accuracy
PLS [46]	77.3
HPE [13]	84.2
SPD-Sc [31]	90.1
KSVM	89.5
Geo-NN	87.1
MACH-1 (proposed)	**93.35**
MACH-2 (proposed)	92.0
MACH-3 (proposed)	90.6

Table 5.4: Person re-identification task: average accuracy (in %) on the ETHZ dataset [46].

Method	UCSD	ETH80	BRODATZ	ETHZ
MACH-1	110.03	46.04	279.62	28486.08
MACH-2	19.83	46.15	160.74	2338.49
MACH-3	2.64	3.74	33.09	1511.71

Table 5.5: Time comparison: The run time (in seconds) of MACH-1, MACH-2 and MACH-3 on each dataset.

Comparisons between MACH and Geo-NN. As mentioned in the section 5.3, Geo-NN is sensitive to the small number of training data. To verify this, we performed an empirical experiment on ETH80. In particular, we increased the training data for Geo-NN using artificial points randomly generated by using the weighted Riemannian centre of mass of the training points [105]. We found that 160 artificial training points is required for Geo-NN to achieve the comparable accuracy with MACH. Unfortunately, this process is extremely expensive, as the generation of these artificial points which used weighted Riemannian centre of mass required 22min. While, the average running time for MACH-3 on the ETH80 is only 3.74s. This indicates the additional advantage of MACH over the intrinsic approaches such as Geo-NN when only a small number of training data is available.

Comparisons between MACH and SPD-Sc. In all experiments, our proposed method MACH achieved much better performance than SPD-Sc, especially on Brodatz (10.4 percentage points better). We conjecture that the improvement could be attributed to the restricted condition on the weights w_i, as well as the convex hull optimisation that based on each separate class.

5.6 Summary

In this chapter, we presented a Manifold Convex Hull (MACH) framework for classification tasks on SPD points. To solve the optimisation problem posed when performing convex hull analysis, we studied three different solutions MACH-1, MACH-2 and MACH-3. MACH-1 served as the intrinsic solution, whilst MACH-2 and MACH-3 are the approximated solutions that possess lower computa-

tional load, as well as competitive accuracy compared to MACH-1. Experiments were performed on four computer vision applications where our proposed methods showed superior performance than several recent methods. Chapter 6 continues by proposing the landmark manifold that following the Component-3 in the manifold scheme (refer to Figure 3.1).

Chapter 6

Landmark Manifold to Recognise Facial Emotions

Intellectuals solve problems,
geniuses prevent them.

Albert Einstein

Chapter Summary: *Manifold approaches can be potentially sped up by using lower-dimensional manifold points. In this chapter, we propose to use the landmark locations to model an efficient manifold model for facial emotion recognition.*

6.1 Overview

Automatically recognising facial emotions has drawn increasing attention in computer vision. Facial landmark based methods are one of the most widely used approaches to perform this task. However, these approaches do not provide good performance. Thus, researchers usually tend to combine more information such as textural and audio information to increase the recognition rate. In this chapter we propose a novel method, here called the landmark manifold, which shows the possibility to achieve competitive performance by facial landmark information alone. Through experiments on the well-known dataset: marked Cohn-Kanade extended facial emotion dataset (CK+) [101], we show that with accurate facial landmarks, our simple approach is fast to run and can achieve competitive performance compared with enormously expensive methods.

6.2 Introduction

In this chapter, we study manifold methods for emotion recognition. Facial emotions play an important role in our daily life and human beings can distinguish different emotions without much effort. Also, it is an important component for various applications such as Human-Computer Inter-

actions (HCI) and automated video annotations [82]. However, automatically recognising the facial emotions is still a challenging problem for computer systems.

Facial emotions usually are categorised into six classes: angry, happy, sad, disgust, fear and surprise [101]. In the past decade, a number of different approaches have been proposed to tackle this problem [95, 97, 101, 156]. Broadly speaking, there are two common approaches to deal with the representations: geometric shape based methods and appearance based methods. Geometric shape based methods use facial landmark information alone. The facial landmarks could be estimated using methods such as: Active appearance model (AAM) [34], Constrained Local Model (CLM) [35], and Supervised Descent Method (SDM) [162]. However, it is usually difficult to get reliable performance using facial landmark information alone in many complex scenarios. For instance, poor illumination or high face pose angle could severely affect the system due to errors introduced by the estimated facial landmarks. On the other hand, the appearance based methods use texture information such as Scale Invariant Feature Transform (SIFT) and Local Binary Patterns (LBP) [99, 107]. After the features have been extracted, various learning methods are employed to select discriminative features. Some classification schemes such as Support Vector Machine (SVM), Linear Discriminant Analysis (LDA) and Sparse Representation Classifier (SRC) could be then applied on the selected features.

It is noted that facial emotions are mainly generated by the movements of the facial muscles. This leads to changes in the facial landmark locations. Thus, capturing the change of facial landmarks could be useful in addressing the facial emotion recognition. In this chapter, we propose to model the set of facial landmarks estimated from each frame within a video as a point on a Riemannian manifold. Riemannian manifolds have been used to address various challenging vision applications such as face recognition and pedestrian detection [142, 144].

Employing Riemannian manifolds to address the facial emotion recognition problem has been used in several previous works [96, 97]. The best performance is achieved by Liu *et al.* in [97], where three different Riemannian manifolds were used by modelling features extracted from Histogram of Oriented Gradients (HOG) and SIFT features. Then, combined multiple kernels that calculated from the three manifolds were employed. We note that despite the excellent results achieved by this method, the computational cost is expensive for large datasets.

6.3 The Proposed Landmark Manifold

In this section, the proposed method, denoted Landmark Manifold, is presented. Our method comprises two main steps. First, we extract the facial landmarks for each video frame. Then we model these landmarks onto SPD manifolds and Grassmann manifolds. Finally, for landmark on SPD manifold, we conduct the emotion recognition task by applying the Log-Euclidean metric with Linear SVM. On the other hand, for landmark on Grassmann manifold, kernel SVM with the kernel defined in Eqn. (2.13) is employed. We note that kernel calculation on Grassmann is much cheaper than SPD. Thus, we opt to use the kernel approach for Grassmann manifold.

6.3.1 Facial Landmark Estimation

A number of face alignment methods have been proposed to track the facial landmarks for emotion recognition, including the classic Active Appearance Model (AAM) [34], Constrained Local Model (CLM) [35], and Supervised Descent Method (SDM) [162]. Recently, it has been shown that the Coarse-to-Fine Shape Searching (CFSS) achieved the state-of-the-art performance compared to the other facial landmarking methods. In our work, we consider these two methods: SDM and CFSS. Besides the state-of-the-art method, CFSS, we opt to consider SDM because it is currently widely used for face alignment [38].

Supervised Descent Method (SDM)

SDM is a cascaded regression method. We define a shape $\boldsymbol{s} \in \mathbb{R}^{2n}$ as a set of n landmark coordinates, $\boldsymbol{s} = [x_1, y_1, \cdots, x_n, y_n]$. The goal of SDM is to estimate a shape $\boldsymbol{s}$ as close as possible to its ground truth landmarks, $\boldsymbol{s}^*$:

$$\min \|\boldsymbol{s} - \boldsymbol{s}^*\|^2 \,. \tag{6.1}$$

In general, the cascaded regression face alignment methods start from an initial shape $\boldsymbol{s^0} \in \mathbb{R}^{2n}$ and refine the shape by applying a cascade of regressors through T stages. At the t-th stage, shape-indexed features $\boldsymbol{f^t}$ are extracted from the image $\boldsymbol{I}$ based on the shape estimated at the previous iteration $\boldsymbol{s^{t-1}}$. Each regressor determines an update, $\triangle \boldsymbol{s} \in \mathbb{R}^{2n}$ based on the shape-indexed features. The current shape of the t-th stage $\boldsymbol{s^t}$ is updated by adding the update $\triangle \boldsymbol{s}$ to the shape of the previous iteration $\boldsymbol{s^{t-1}}$ via:

$$\boldsymbol{s^t} = \boldsymbol{s^{t-1}} + \triangle \boldsymbol{s} \,. \tag{6.2}$$

SDM uses linear regression and local Scale-Invariant Feature Transform (SIFT) features on local patches centred on the current estimated landmark positions to determine the update, $\triangle \boldsymbol{s}$. Thus, Eqn. (6.2) is then replaced by:

$$\boldsymbol{s^t} = \boldsymbol{s^{t-1}} + \mathrm{R}^{t-1}\, \phi^{t-1} + \mathrm{b}^{t-1} \,, \tag{6.3}$$

where R^{t-1} and b^{t-1} are a sequence of generic descent directions and bias terms respectively; ϕ^{t-1} represents a feature vector obtained by concatenating SIFT features around currently estimated landmarks $\boldsymbol{s^{t-1}}$. The R^t and b^t can be learned by minimising:

$$\min_{\mathrm{R}^t, \mathrm{b}^t} \|\triangle \boldsymbol{s}^* - \mathrm{R}^t\, \phi^t - \mathrm{b}^t\|^2 \,, \tag{6.4}$$

where $\triangle \boldsymbol{s}^*$ is the difference between current shape and ground truth shape $\triangle \boldsymbol{s}^* = \boldsymbol{s}^* - \boldsymbol{s^t}$.

Coarse-to-Fine Shape Searching (CFSS)

In order to make the face alignment method more robust, Zhu *et al.* proposed a Coarse-to-Fine Shape Searching method (CFSS) [170]. The central idea of CFSS is to search a shape sub-region at each

stage in a coarse-to-fine manner so that the estimated shape can avoid being trapped by the local optima if given a poor initialization.

The search is within a shape space $\boldsymbol{S} = \{\boldsymbol{s_1}, \boldsymbol{s_2}, ..., \boldsymbol{s_N}\}$, which contains N candidate shapes obtained from the training set. In a coarse-to-fine manner, the searching is performed through $l = 1, ..., L$ stages. At each l-th stage, the goal is to find a finer shape sub-region represented by a sub-region centre $\bar{s}_{(l)}$ and shape probability distribution $P^R_{(l)}$ (represents the scope of the sub-region).

Each l-th stage starts from sampling N_l initial shapes from the shape space $\boldsymbol{S}$ according to the probability learned from the previous stage $P^R_{(l-1)}$. Then the N_l initial shapes are regressed to ground truth shapes in a cascaded manner. The cascaded regression method used here is similar to SDM. After learning the regressors and obtaining the regressed N_l candidate shapes, the sub-region centre $\bar{s}_{(l)}$ is estimated by;

$$\bar{s}_{(l)} = \sum_{j=1}^{N_l} w^j \boldsymbol{s}^i_T \,, \tag{6.5}$$

where the weight w^j of each candidate shape is estimated by the dominant set approach.

At the last stage sub-region centre $\bar{s}_{(L)}$ is the final estimated shape.

6.3.2 Modelling Landmarks on SPD Manifolds

For each video, we extract a group of facial landmark features $\mathcal{S} = \{\boldsymbol{s_1}, \boldsymbol{s_2}, ..., \boldsymbol{s_n}\}$, n is the number of frames. $\boldsymbol{s_i}$ is represented by p landmarks, $\boldsymbol{s_i} = \{x_1, y_1, x_2, y_2, ..., x_p, y_p\}^\top$. Then we compute the covariance matrix by:

$$\boldsymbol{C} = \frac{1}{n-1} \sum_{i=1}^{n} (\boldsymbol{s_i} - \bar{\boldsymbol{s}}) (\boldsymbol{s_i} - \bar{\boldsymbol{s}})^\top \,, \tag{6.6}$$

where $\bar{\boldsymbol{s}}$ is the mean vector of the facial landmarks. The covariance matrix $\boldsymbol{C}$ can be regarded as points on the SPD manifold. A number of machine learning methods have been proposed to address classification problem on SPD manifolds [31, 144]. In this work, we use a simple-yet-effective method comprising the linear SVM in conjunction with the Log-Euclidean metric. This method is fast to run and shows extremely good performance for emotion recognition.

Technically, to employ the linear SVM, we first map the SPD points from the SPD manifold onto the tangent space at the identity matrix (*i.e.,* the Log-Euclidean space) via: $\boldsymbol{A} = \log(\boldsymbol{C})$. This essentially flattens the manifold into the Euclidean space. Following Pennec *et al.* [112], a vectorised feature vector can be extracted as:

$$\begin{aligned} \mathrm{Vec}(\boldsymbol{A}) = [&a_{1,1}, \sqrt{2}a_{1,2}, a_{2,2}, \sqrt{2}a_{1,3}, \sqrt{2}a_{2,3}, a_{3,3}, \\ &..., \sqrt{2}a_{1,d}, ..., \sqrt{2}a_{d-1,d}, a_{d,d}]^\top \,. \end{aligned} \tag{6.7}$$

These features are then fed into the linear SVM. We note that the Euclidean distance of the above features is the same as the Log-Euclidean metric presented in Eqn. (2.6).

6.3.3 Modelling Landmarks on Grassmann Manifolds

Modelling the frames to one video as a group of linear spaces have been shown to be superior on a number of video-based analysis such as object recognition [68] and face recognition [47]. The group of linear spaces can be regarded as points on the Grassmann manifold. Also, Begelfor *et al.* [14] exploited affine invariant clustering of shapes by using the Grassmann manifold structure.

Specifically, the descriptor of each video is first formed as: $\mathcal{S} = \{\boldsymbol{s_1}, \boldsymbol{s_2}, ..., \boldsymbol{s_n}\}$, where n is the number of frames and each $\boldsymbol{s_i}$ is represented by p landmark points, $\boldsymbol{s_i} = \{x_1, y_1, x_2, y_2, \ldots, x_p, y_p\}^\top$. As shown in [37], the pairwise landmark distances provided more robust information for the image-based emotion recognition. Therefore, we model the dynamic changes of these pairwise distances from each frame on the Grassmann manifold. Specifically, we applied the Singular Value Decomposition (SVD) on the following matrix $\boldsymbol{W}$:

$$\begin{aligned} \boldsymbol{W} &= \{\boldsymbol{w_1}, \boldsymbol{w_1}, ..., \boldsymbol{w_n}\}, \\ \boldsymbol{w_i} &= \left[|x_1 - x_2|, ..., |x_{n-1} - x_n|, |, y_1 - y_2|, ..., |y_{n-1} - y_n|\right]^\top. \end{aligned} \tag{6.8}$$

The SVD operation is to get the group of subspaces $\boldsymbol{U}$:

$$\boldsymbol{W} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^\top, \tag{6.9}$$

where $\boldsymbol{U} = \{\boldsymbol{u}_1, ..., \boldsymbol{u_d}\}$ is the top d eigenvectors and is a point on the Grassmann manifold. For the consideration of speed and performance, the kernel SVM with projection kernel (refer to Eqn. (2.13)) is applied.

6.4 Experimental Results

In the experiments, we use the Cohn–Kanade extended facial emotion (CK+) dataset [101] to perform a number of evaluations and analyse the efficacy of our proposed landmark manifold method. We further provide discussions on the estimated landmark locations given by CFSS and SDM.

6.4.1 Dataset Description

The CK+ dataset (refer to Figure 6.2 for examples) is widely used to benchmark the emotion recognition methods. The database contains 123 different subjects and 593 frontal sequences. Among these, 118 subjects are labelled with the seven emotion categories (anger, contempt, disgust, fear, happy, sad and surprise). The sequences were annotated with 68 landmark points. The key-frames within each video sequence were manually labelled, while the remaining frames were estimated by AAM. Thus, we call these original landmarks AAM landmarks. Following the work of Lucey *et al.* [101], the leave-one-subject-out protocol is used in our experiments.

6.4.2 Experimental Results with Original Landmarks

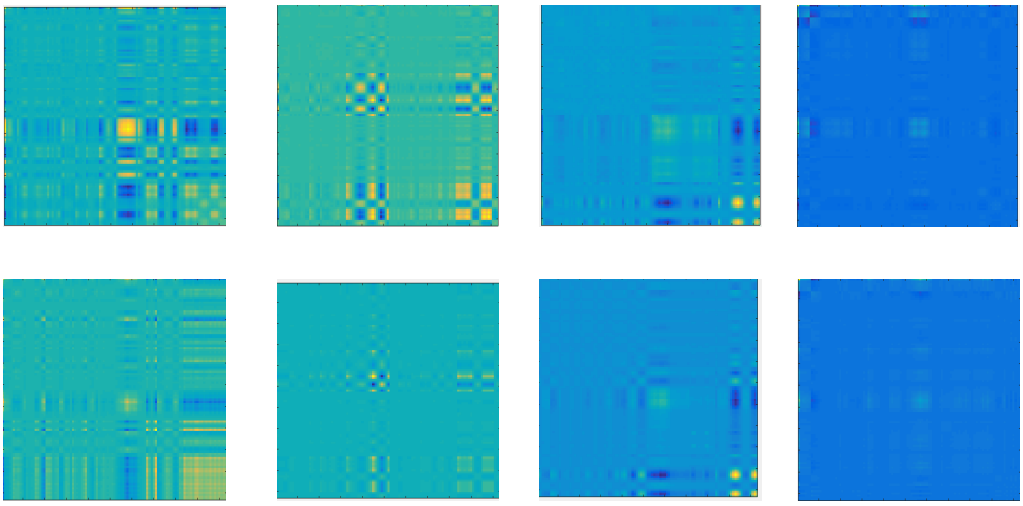
In this experiment, we use the CK+ dataset with the original CK+ landmarks [101]. First, we model the landmarks onto the SPD manifold through Eqn. (6.6). Then we project the SPD points onto the log-Euclidean space and vectorise them using Eqn. (6.7). We train a multi-class SVM using the leave-one-subject-out protocol. The variations of our proposed method are named in the following style: *landmarking method-manifold used*. For instance, AAM-SPD is the method using AAM modelled in SPD.

The results for each emotion category are shown in Figure 6.3. The overall and average accuracy are listed in Table 6.2. In particular, the average accuracy is computed by the mean of each category accuracy. Whilst, the overall accuracy is determined by the percentage of videos that are correctly classified over all tests. We found that the proposed method achieves nearly perfect performance for the high energy emotions such as 'happy', 'disgust' and 'surprise' (*i.e.,* the ones exhibiting large displacements of the facial landmarks). Even for the complex emotions such as 'fear' and 'sad', the performance is significantly better than the best performance reported by Lucey *et al.* [101] (65.2% vs 88% and 68.0% vs 82.14% for 'fear' and 'sad', respectively). It is noted that the best performance from [101] used the shape and textural information based on same landmarks. However, our superior performance is obtained by only using the shape information. These results suggest that the proposed manifold landmark captures important information to perform the facial emotion recognition.

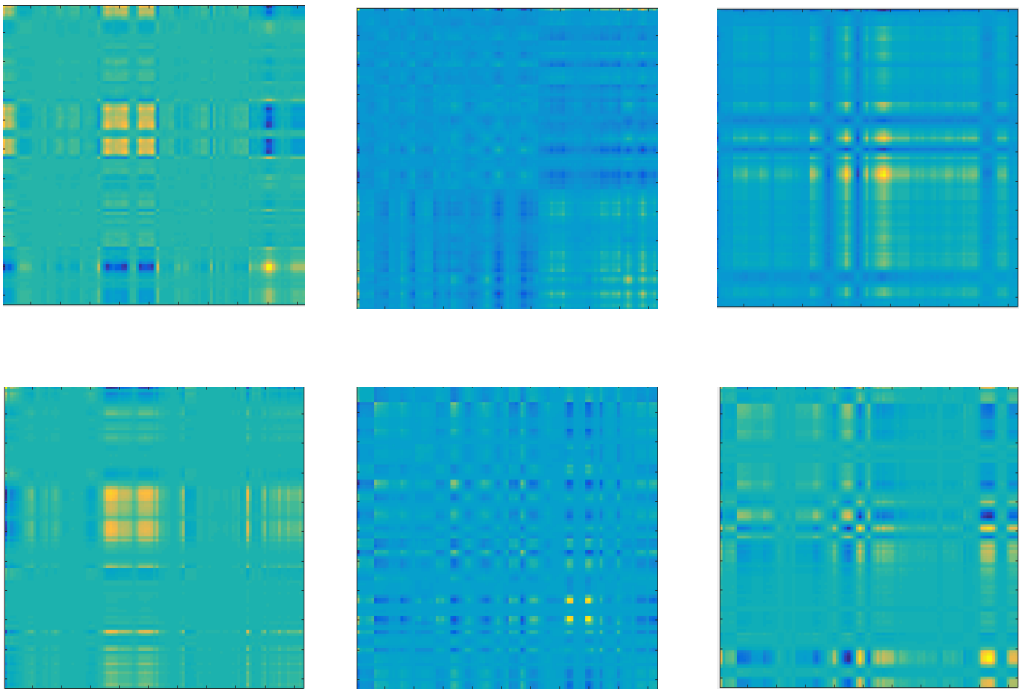
We further contrast our system to single manifold methods discussed by Liu *et al.* [97]. In particular, the SPD features are extracted from the textural features (*i.e.,* either HOG or SIFT). We call these methods HOG-SPD and SIFT-SPD. As can be seen from Table 6.1, the proposed AAM-SPD outperforms both methods.

Our proposed AAM-SPD method also achieved much better performance than the recent methods: HHM [156], ITBN [156], 3DCNN [95], 3DCNN-DAP [95] and MCF [33]. Compared to the state-of-the-art performance from Liu *et al.* [97] based on fusion of six different manifold kernels and two textural features (HOG+SIFT-SPD+G+Gaussian), our simple landmark model achieves competitive recognition rates. To indicate the effectiveness of our proposed AAM-SPD features, we show some intermediate visualisation results for different emotions on CK+ dataset in Figure 6.1. We use the full range of colours to display the SPD features generated using AAM landmarks. Note that in Figure 6.1a and Figure 6.1b, images from a same column are with same emotion labels, but from different subjects. From the visualisation, we found that the AAM-SPD features are quite discriminative for the emotion recognition task.

It is noteworthy to mention that our landmark manifold has a much smaller number of parameters. In addition, the Riemannian kernel computation is expensive especially for large datasets. For example, in the test stage, computing the RBF kernel used in [97] requires $O(N_{tr}N_{te}d^3)$, where N_{tr} is the number of training points, N_{te} is the number of test points and d is the dimensionality of the manifold points. Their best performance [97] is achieved by computing 12 kernel matrices in total. However, in our AAM-SPD, the main computation cost is the Log-Euclidean projection ($\log(\boldsymbol{C})$),



(a) Emotion class: Disgust, Happy, Surprise, Angry (from left to right). Images from a same column are with same emotion labels, but from different subjects.



(b) Emotion class: Contempt, Sadness, Fear (from left to right). Images from a same column are with same emotion labels, but from different subjects.

Figure 6.1: Intermediate visualisation results for AAM-SPD features on CK+ dataset.

which is $O((N_{te}d^3)$. The average time to test one video using AAM-SPD is only 0.15s in Matlab on an Intel Core 2 Duo processor running at 3 GHz.

We also report the results of modelling the original landmarks on the Grassmann manifold (AAM-G) (refer to section 6.3.3 for details). Compared to the AAM-SPD, the AAM-G accuracy for 'contempt', 'fear' and 'sad' shown in Figure 6.4, is much worse. However, we note that the average and overall accuracy of AAM-G are still better than textural-based Grassmann manifold methods: HOG-

Figure 6.2: Emotion examples from CK+ Dataset [101]: Disgust, Happy, Surprise, Angry, Sadness, Fear (from left to right, top to bottom). Note that we only show six emotion examples and the copyrights belong to their respective owner Jeffrey Cohn.

	Angry	Contempt	Disgust	Fear	Happy	Sad	Surprise
Angry	91.11	0	6.67	0	0	2.22	0
Contempt	0	94.44	0	0	0	5.56	0
Disgust	0	0	100	0	0	0	0
Fear	4	0	0	88	8	0	0
Happy	0	0	0	0	100	0	0
Sad	14.29	0	3.57	0	0	82.14	0
Surprise	0	1.2	0	0	0	0	98.8

Figure 6.3: Percentage recognition rates on the SPD manifold with original landmarks (AAM-SPD).

G and SIFT-G [97] listed in Table 6.1. This again demonstrates that modelling the facial landmark on Riemannian manifolds is much more effective than modelling textural features on the manifolds.

	Angry	Contempt	Disgust	Fear	Happy	Sad	Surprise
Angry	91.11	0	4.44	0	0	4.44	0
Contempt	5.56	77.78	0	0	0	11.11	5.56
Disgust	0	0	98.31	1.69	0	0	0
Fear	4	0	0	72	16	4	4
Happy	0	0	0	0	100	0	0
Sad	17.86	3.57	3.57	0	0	75	0
Surprise	0	1.2	0	0	0	0	98.8

Figure 6.4: Percentage recognition rates on the Grassmann manifold with original landmarks (AAM-G).

Table 6.1: The comparisons between the proposed landmark manifold with recent state-of-the-art face emotion recognition methods.

Methods	Average Acc	Overall Acc
AAM [101]	83.3	88.3
HMM [156]	83.5	n/a
ITBN [156]	86.3	88.8
3DCNN [95]	78.0	85.9
3DCNN-DAP [95]	87.9	92.4
MCF [33]	92.7	n/a
Riemannian manifold-based methods		
HOG-SPD [97]	91.63	n/a
SIFT-SPD [97]	89.4	n/a
HOG-G [97]	83.6	n/a
SIFT-G [97]	84.4	n/a
HOG+SIFT-SPD+G+Gaussian [97]	**94.8**	**96.6**
Proposed landmark manifolds		
AAM-SPD	**93.5**	**95.7**
AAM-G	87.6	92.7

Table 6.2: Recognition rates on SPD and Grassmann manifold with different landmarks methods.

Methods	Average Acc	Overall Acc
SDM-SPD	78.9	86.9
CFSS-SPD	81.9	88.7
CFSS+SDM-SPD	82.7	89.3
AAM(mouth)+SDM-SPD	87.4	91.4
AAM(mouth)+CFSS-SPD	88.2	92.4
AAM-SPD	**93.5**	**95.7**
SDM-G	82.4	89.6
CFSS-G	84.8	89.9
CFSS+SDM-G	86.1	91.5
AAM-G	**87.6**	**92.7**

6.4.3 Experimental Results with CFSS and SDM Landmarks

When dealing with 'in-the-wild' data, estimating accurate facial landmarks is a challenging problem. Misalignments often happen when using the facial alignment approaches to estimate the facial landmark locations from the uncontrolled real-world videos. To that end, in this section, we study how our proposed landmark manifold would perform when provide with the inaccurate facial landmark locations. We perform the experiments using the landmarks estimated by the recent state-of-the-art face alignment methods such as CFSS and SDM.

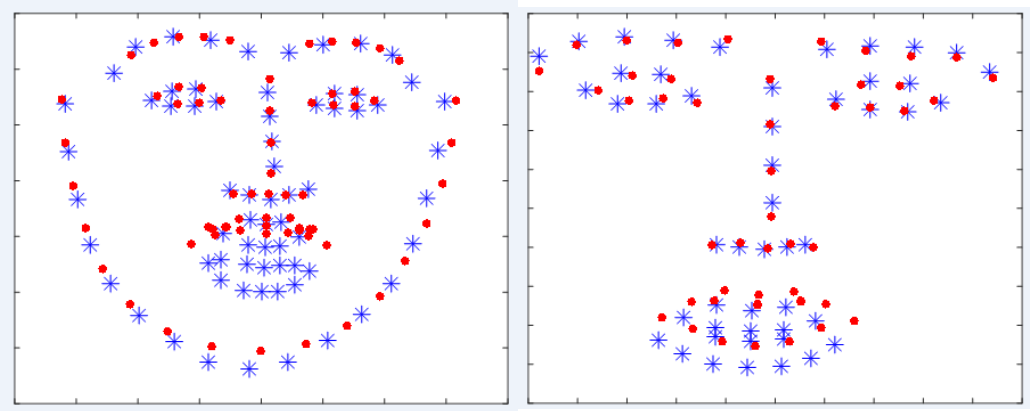


Figure 6.5: Examples of landmark errors by CFSS (left image) and SDM (right image). The 'red dot' points are the original AAM landmarks, 'blue x-mark' points are the CFSS and SDM landmarks.

SPD Landmark Manifolds

We first study the SPD manifold with CFSS and SDM landmarks. As shown in Table 6.2, inaccurate facial landmarks severely affect the performance of the SPD manifold (*i.e.,* from 93% to 78% and 81%, for CFSS-SPD and SDM-SPD, respectively). This is probably because the SPD manifold is sensitive to the errors of the landmark locations.

To further boost the performance of our proposed landmark manifold, we use the fusion landmark locations from CFSS and SDM for each frame $\boldsymbol{s}_i$, $i = 1, 2, ..., n$:

$$\boldsymbol{s_i} = \{\lambda * s_{CFSS,i} + (1 - \lambda) * s_{SDM,i}\}, \tag{6.10}$$

where the $s_{CFSS,i}$ and $s_{SDM,i}$ are the landmark locations generated by CFSS and SDM separately. We denote the fusion method as CFSS+SDM-SPD. The performance shows a slight increase. Upon closer examination, we found that the landmark points are not correctly estimated around the mouth by CFSS and SDM (refer to Figure 6.5 for examples). To further verify this hypothesis, we performed experiments by replacing the mouth landmark points of CFSS and SDM with the corresponding points from AAM. The average accuracy increases from 81.9% to 88.2% for CFSS-SPD, from 78.9% to 87.4% for SDM-SPD. Thus, to get excellent performance, the proposed SPD model requires more accurate landmark locations.

Grassmann Landmark Manifolds

In this section, we study the Grassmann manifold with CFSS landmarks and SDM landmarks. The results of CFSS-G and SDM-G are reported in Table 6.2, Figure 6.6 and 6.7. Again, we see a performance decrease is shown when inaccurate facial landmarks are given.

From Table 6.2, we found that the fusion landmark manifold (CFSS+SDM-G) actually can achieve competitive performance to the AAM-G. These results suggest that it is possible to improve the performance of the proposed landmark manifold by combining multiple landmarks obtained from various methods.

	Angry	Contempt	Disgust	Fear	Happy	Sad	Surprise
Angry	86.67	0	6.67	0	4.44	0	2.22
Contempt	0	77.78	0	5.56	0	16.67	0
Disgust	1.69	0	93.22	3.39	0	0	1.69
Fear	8	0	4	68	4	0	16
Happy	0	0	0	0	98.55	0	1.45
Sad	10.71	0	10.71	0	0	71.43	7.14
Surprise	1.2	0	0	1.2	0	0	97.59

Figure 6.6: Percentage recognition rates on the Grassmann manifold with CFSS landmarks.

	Angry	Contempt	Disgust	Fear	Happy	Sad	Surprise
Angry	73.33	4.44	8.89	0	2.22	6.67	4.44
Contempt	11.11	72.22	5.56	0	5.56	0	5.56
Disgust	1.69	0	94.92	0	0	1.69	1.69
Fear	4	0	0	72	4	0	20
Happy	0	0	0	0	100	0	0
Sad	17.86	7.14	3.57	0	3.57	67.86	0
Surprise	0	1.2	0	0	0	0	98.8

Figure 6.7: Percentage recognition rates on the Grassmann manifold with SDM landmarks.

	Angry	Contempt	Disgust	Fear	Happy	Sad	Surprise
Angry	86.67	2.22	8.89	0	2.22	0	0
Contempt	11.11	83.33	0	0	5.56	0	0
Disgust	1.69	0	98.31	0	0	0	0
Fear	0	0	0	64	20	0	16
Happy	0	0	0	0	100	0	0
Sad	7.14	0	14.29	0	3.57	71.43	3.57
Surprise	1.2	0	0	0	0	0	98.8

Figure 6.8: Percentage recognition rates on Grassmann manifold with the fusion landmarks.

6.5 Summary

In this chapter, we found that the dynamic information from the facial landmarks plays a significant role on the emotion recognition task. To that end, we proposed a method named the landmark manifold that utilises this information. When using accurate facial landmarks obtained from each video frame on CK+ dataset, our proposed method achieved competitive performance to the recent state-of-the-art methods which employ multiple manifold models and much more complex kernelised learning algorithms. Thus, our proposed landmark manifold possessed a lower computational complexity without significant accuracy loss. To deal with the errors generated from the inaccurate estimated facial landmarks, we showed that it was possible to fuse multiple facial landmarks estimated by various methods.

Chapter 7

Conclusions and Future Work

> The true sign of intelligence is not knowledge but imagination.
>
> Albert Einstein

Chapter Summary: *Current manifold approaches with superior performance suffer from intensive computational cost, especially when the dataset is large. This thesis explores three avenues to address the problem and proposes various methods which have much lower computational complexities, whilst still maintaining excellent performance in the examined computer vision applications. Future avenues of exploration include optimised projection functions, extension to other types of manifolds and learning improved lower-level features.*

7.1 Thesis Summary and Conclusions

Manifold models have been proven to have superior performance in various computer vision applications. Despite a steady progress in developing manifold approaches for image and video analysis, there are still some remaining challenges. Unlike the Euclidean space, manifolds naturally possess nonlinear structures, which makes the operations such as computing geodesics and learning models suffer from intensive computational cost. This thesis proposed three different paths, random projection framework, convex hull framework and landmark manifold, to reduce the computational complexities of manifold approaches without significant accuracy loss for image and video analysis. These results have significant benefits in computer vision applications in terms of classification and clustering scenario.

Chapter 1 briefly introduced the manifolds and our research problems. Also, the research questions, goals and contributions were listed in this chapter.

Chapter 2 elaborated the background theory of manifold geometries, as well as the related work in manifold approaches for computer vision.

In Chapter 3, we presented a comprehensive scheme for manifold approaches, illustrated in Figure 3.1. Currently, some manifold approaches suffer from high computational load when the dataset is extremely large. Based on this scheme, we discussed several different directions to reduce the computational cost, while still maintaining the excellent accuracy in image and video analysis. Each direction can be covered by a research question that is listed in below Section 1.3.1.

Illustrated by the scheme in Figure 3.1, given the manifold features, the learning approaches can be broadly categorized into intrinsic methods and extrinsic methods. Extrinsic methods first map the manifold to a Euclidean space, and then apply Euclidean-based learning logarithm in the new space. One of the most popular extrinsic methods is using kernel functions, which usually achieve excellent performance. However, kernel-based learning algorithms such as kernel k-means suffer from high computational cost, especially for large datasets. Due to the issues with the current extrinsic methods, the first research question posed is:

Q1: *Is it possible to devise a mapping function that maps manifold data points into a Euclidean space wherein computational complexity is considerably reduced? If so, can the manifold structure be well-preserved in the newly created space produced by such a mapping function?*

The answer to this question is: Yes, it is possible to devise a mapping function. In addition, the manifold structure could be potentially preserved. More specifically, the thesis adapted the random projection concept in Euclidean space into the manifold. According to the Johnson-Lindestrauss's Lemma (refer to Lemma 4.3.1), the topological structure from the original space could be preserved up to some points. Equipped with this lemma, Chapter 4 proposed a random projection framework for manifold points. The core part of this framework is the random hyperplane generation. In this chapter, we described three hyperplane generation algorithms: KGRP, KORP and KPCA-RP. Through experiments on several computer vision applications, it has been verified that the proposed framework significantly reduced the computational complexity, but still maintained excellent performance.

On the other hand, there is a school of thought to research intrinsic methods on the manifold. Geodesic Nearest Neighbour is one of the most popular intrinsic methods for classification tasks. However, the computational complexity suffers a linear growth with the number of training data. Driven by these concerns, we pose the second research question:

Q2: *Is it possible to improve the performance and computational time of the Geodesic Nearest Neighbour by utilising the intrinsic classifiers on manifolds?*

The answer to this question is: Yes, it is possible to improve the performance and computational time of the Geodesic Nearest Neighbour. Inspired by the development of classifiers in the Euclidean counterpart, we noted that nearest convex hull classifiers that can improve the performance of Nearest Neighbour through modelling each class using all the convex combinations of training points. Thus, we extend the convex hull concept in Euclidean space into the manifold. More specifically, in Chapter 5, we proposed a manifold convex hull to tackle the classification tasks on SPD manifolds. This is a novel mathematical framework with the definition of the convex combinations and the nearest convex hull distance over the SPD manifolds. MACH-1 was proposed to solve the optimisation method

using the intrinsic structures, which is slow. To speed up the computations, MACH-2 and MACH-3 were proposed whose accuracies are competitive with MACH-1. The experimental results on several different computer vision applications showed that the proposed manifold convex hull outperforms other intrinsic methods and some state-of-the-art methods.

Further examination of the manifold scheme, we found that the dimensionality of the manifold points also has significant impact on the computational load of the following learning process. To that end, one can apply dimensionality reduction techniques to obtain low-dimensional manifold points. Alternatively, one can use features with small dimensions to form the manifold model. We focused on the second method that was more direct to research. This led to the third research question:

Q3: *Can we reduce the dimensionality of the manifold points by finding discriminative lower-level features with low dimensionality to generate the manifold model?*

The answer to this question is: Yes, we can find discriminative lower-level features with low dimensionality to generate the manifold model. This leads to a more efficient lower-dimensional manifold model. To evaluate the performance, we performed an application-based study in this thesis. More specifically in Chapter 6, we revisited the state-of-the-art manifold method for emotional recognition. We found that their method [97] used complex and high-dimensional features (SIFT and HOG) to form three different manifold models. Then, multiple kernels were used to obtain the final recognition results. This resulted in intensive computational load. In our work, we identified that it is possible to use the simple landmark locations as the lower-level features to form only one manifold model. This achieved competitive performance with the multiple kernel methods proposed in [97]. This chapter therefore answered the research question Q3 by showing that it is possible to use effective features to form a lower-dimensional manifold for the emotion recognition application without obvious accuracy loss. The lower-dimensional manifold led to lower computational complexities.

7.2 Future Work

To make the manifold approaches more applicable in computer vision, it is necessary to tackle the computational complexity issues and therefore develop fast and accurate approaches. The research questions presented in this thesis and the approaches proposed lead us to several promising future directions. In this section, we outline a few directions for our future work.

The random projection framework proposed in Chapter 4 showed that it is possible to project the manifold points onto a discriminative Euclidean space and use linear learning methods to tackle the computer vision tasks. Due to the properties of random projection, the errors of projection functions we used in this chapter are bounded but not minimised. In the future work, it will be ideal to develop optimised projection functions that can map manifold points onto a Euclidean space, where the classification/clustering error is minimised. Another direction can be the development of the intrinsic random projection framework for manifolds. To that end, one needs to research the intrinsic projection definitions and how to generate the intrinsic random projection hyperplanes on manifolds.

The manifold convex hull proposed in Chapter 5 showed excellent performance in the examined computer vision applications. This approach focused on SPD manifold. One direction for the future work are the extension work on the nearest convex hull classifiers on types of manifolds such as Grassmann manifolds. To that end, one need to develop the solutions to the optimisation problems based on metrics on Grassmann manifolds. To decrease the computational complexity, one can consider to develop the hashing techniques based on the weights obtained by the nearest convex hull framework.

In Chapter 6, we focused on a specified application– emotional recognition. Manifold approaches showed superior performance in recognising the facial emotions due to its ability to capture the dynamic changes on the faces. However, the question remains open on what are the best features to form the manifold model for emotion recognition. We showed that the simple landmark locations can form a low-dimensional manifold that is computationally efficient for the emotion recognition. However, this landmark manifold is sensitive to the landmark location errors. Thus, it did not perform well when some landmark estimation methods were used. Fortunately, the performance would increase if one used the fusion landmark features formed from different estimations. Further work is needed on the feature learning algorithms that can minimise the landmark errors. Another possible direction will be the research on other types of lower-level features with small dimensions to generate the manifold model. How to find the best lower-level features for different manifold model is still an open question.

7.3 Concluding Remarks

In the recent years, there has been a large growth in the amount of vision data. Classifying or clustering data into different categories helps people to better understand the characteristics of the data. The recent advances of computer vision have proven that the intrinsic data structure lies in a nonlinear space. The manifold geometry provides an effective tool to capture the intrinsic structure of the vision data, which leads to superior classification/clustering [124]. For example, utilising the manifold geometry can achieve high success in pedestrian detection and tracking [142]. Effective face recognition algorithms have been developed using manifold representations [60]. Despite the progress in manifold approaches in computer vision, the computational load issues still remain a significant barrier to applications that require real time performance. This thesis presented three paths to reduce the computational load of manifold approaches.

The first path is to devise a mapping function that maps the manifold points onto a Euclidean space wherein the computational load is reduced (pondered in research question 1). We addressed this by proposing the random projection framework in Chapter 4. The second path is to extend the effective classifiers originally developed for Euclidean space into the manifold and devise efficient solutions to ensure a low computational load (pondered in research question 2). We addressed this by proposing the convex hull framework in Chapter 5. The last path considered in this thesis, is to use effective lower-level features with low dimensionality to generate the manifold model, which would lead to less computational load when processing the manifold points (pondered in research question

3). We addressed this by studying manifold approaches on facial emotion recognition and proposing the landmark manifold model in Chapter 6.

After reducing the computational load of manifold approaches without significant accuracy loss, the vision data can be more easily analysed using manifold approaches. Accurate, as well as fast, manifold approaches enhance the classification and clustering of similar images/videos, which in turn enables effective automated solutions for computer vision applications required real-time performance.

Bibliography

[1] https://www.youtube.com/yt/press/statistics.html, accessed on 2nd Nov. 2014.

[2] M. F. Abdelkader, W. Abd-Almageed, A. Srivastava, and R. Chellappa, "Silhouette-based gesture and action recognition via modeling trajectories on riemannian shape manifolds," *Computer Vision and Image Understanding*, vol. 115, no. 3, pp. 439–455, 2011.

[3] P.-A. Absil, R. Mahony, and R. Sepulchre, "Riemannian geometry of grassmann manifolds with a view on algorithmic computation," *Acta Applicandae Mathematica*, vol. 80, no. 2, pp. 199–220, 2004.

[4] D. Achlioptas, "Database-friendly random projections: Johnson-lindenstrauss with binary coins," *Journal of Computer and System Sciences*, vol. 66, no. 4, pp. 671–687, 2003.

[5] A. Alavi, A. Wiliem, K. Zhao, B. C. Lovell, and C. Sanderson, "Random projections on manifolds of symmetric positive definite matrices for image classification," in *Winter Conference on the Applications of Computer Vision*, 2014, pp. 301–308.

[6] A. Alavi, Y. Yang, M. Harandi, and C. Sanderson, "Multi-shot person re-identification via relational stein divergence," in *International Conference on Image Processing*, 2013, pp. 3542–3546.

[7] R. Anirudh and P. Turaga, "Geometry-based symbolic approximation for fast sequence matching on manifolds," *International Journal of Computer Vision*, vol. 116, no. 2, pp. 161–173, 2016.

[8] R. I. Arriaga and S. Vempala, "An algorithmic theory of learning: Robust concepts and random projection," in *Annual Symposium on Foundations of Computer Science*, 1999, pp. 616–623.

[9] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache, "Log-euclidean metrics for fast and simple calculus on diffusion tensors," *Magnetic Resonance in Medicine*, vol. 56, no. 2, pp. 411–421, 2006.

[10] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache, "Geometric means in a novel vector space structure on symmetric positive-definite matrices," *SIAM journal on matrix analysis and applications*, vol. 29, no. 1, pp. 328–347, 2007.

[11] M. Baktashmotlagh, M. Harandi, B. C. Lovell, and M. Salzmann, "Discriminative non-linear stationary subspace analysis for video classification," *International Conference on Machine Learning*, vol. 36, no. 12, 2014.

[12] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Transactions on Mathematical Software*, vol. 22, no. 4, pp. 469–483, 1996.

[13] L. Bazzani, M. Cristani, A. Perina, M. Farenzena, and V. Murino, "Multiple-shot person re-identification by hpe signature." in *International Conference on Pattern Recognition*, 2010, p. 2.

[14] E. Begelfor and M. Werman, "Affine invariance revisited," in *Computer Vision and Pattern Recognition*, 2006, pp. 2087–2094.

[15] J. R. Beveridge, B. A. Draper, J.-M. Chang, M. Kirby, H. Kley, and C. Peterson, "Principal angles separate subject illumination spaces in ydb and cmu-pie," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 2, pp. 351–363, 2009.

[16] E. Bingham and H. Mannila, "Random projection in dimensionality reduction: applications to image and text data," in *International Conference on Knowledge Discovery and Data Mining*, 2001, pp. 245–250.

[17] E. G. Birgin, J. M. Martínez, and M. Raydan, "Algorithm 813: Spg software for convex-constrained optimization," *ACM Transactions on Mathematical Software*, vol. 27, no. 3, pp. 340–349, 2001.

[18] A. Blum, "Random projection, margins, kernels, and feature-selection," in *Subspace, Latent Structure and Feature Selection*. Springer, 2006, pp. 52–68.

[19] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Annual Workshop on Computational Learning Theory*, 1992, pp. 144–152.

[20] C. Boutsidis, A. Zouzias, and P. Drineas, "Random projections for k-means clustering," in *Advances in Neural Information Processing Systems*, 2010, pp. 298–306.

[21] J. Bullas. (2012) 48 significant social media facts, figures and statistics plus 7 infographics. Http://www.jeffbullas.com/2012/04/23/48-significant-social-media-facts-figures-and-statistics-plus-7-infographics/.

[22] R. H. Byrd, J. C. Gilbert, and J. Nocedal, "A trust region method based on interior point techniques for nonlinear programming," *Mathematical Programming*, vol. 89, no. 1, pp. 149–185, 2000.

[23] B. Caputo, E. Hayman, and P. Mallikarjuna, "Class-specific material categorisation," in *International Conference on Computer Vision*, vol. 2, 2005, pp. 1597–1604.

[24] R. Caseiro, P. Martins, J. F. Henriques, F. S. Leite, and J. Batista, "Rolling riemannian manifolds to solve the multi-class classification problem," in *Computer Vision and Pattern Recognition*, 2013, pp. 41–48.

[25] H. Cevikalp and B. Triggs, "Face recognition based on image sets," in *Computer Vision and Pattern Recognition*, 2010, pp. 2567–2573.

[26] A. B. Chan and N. Vasconcelos, "Probabilistic kernels for the classification of auto-regressive visual processes," in *Computer Vision and Pattern Recognition*, vol. 1, 2005, pp. 846–851.

[27] A. B. Chan and N. Vasconcelos, "Modeling, clustering, and segmenting video with mixtures of dynamic textures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 5, pp. 909–926, 2008.

[28] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[29] C.-C. Chen, M. S. Ryoo, and J. K. Aggarwal, "UT-Tower dataset: aerial view activity classification challenge," http://cvrc.ece.utexas.edu/SDHA2010/Aerial_View_Activity.html, 2010.

[30] S. Chen, A. Wiliem, C. Sanderson, and B. C. Lovell, "Matching image sets via adaptive multi convex hull," in *Winter Conference on Applications of Computer Vision*, 2014, pp. 1074–1081.

[31] A. Cherian and S. Sra, "Riemannian sparse coding for positive definite matrices," in *European Conference on Computer Vision*, 2014, pp. 299–314.

[32] A. Cherian, S. Sra, A. Banerjee, and N. Papanikolopoulos, "Jensen-bregman logdet divergence with application to efficient similarity search for covariance matrices," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 9, pp. 2161–2174, 2013.

[33] S. W. Chew, S. Lucey, P. Lucey, S. Sridharan, and J. F. Conn, "Improved facial expression recognition via uni-hyperplane classification," in *Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 2554–2561.

[34] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active appearance models," *International Conference on Machine Learning*, no. 6, pp. 681–685, 2001.

[35] D. Cristinacce and T. F. Cootes, "Feature detection and tracking with constrained local models." in *BMVC*, vol. 2, no. 5. Citeseer, 2006, p. 6.

[36] S. Dasgupta, "Experiments with random projection," in *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, 2000, pp. 143–151.

[37] M. Day, "Exploiting facial landmarks for emotion recognition in the wild," *arXiv preprint arXiv:1603.09129*, 2016.

[38] F. De la Torre, W.-S. Chu, X. Xiong, F. Vicente, X. Ding, and J. F. Cohn, "Intraface," in *IEEE International Conference on Automatic Face and Gesture Recognition*, 2015.

[39] K. G. Derpanis and R. P. Wildes, "Classification of traffic video based on a spatiotemporal orientation analysis," in *Winter Conference on Applications of Computer Vision*, 2011, pp. 606–613.

[40] I. S. Dhillon, Y. Guan, and B. Kulis, "Kernel k-means: spectral clustering and normalized cuts," in *International Conference on Knowledge Discovery and Data Mining*, 2004, pp. 551–556.

[41] D. L. Donoho *et al.*, "High-dimensional data analysis: The curses and blessings of dimensionality," *AMS Math Challenges Lecture*, pp. 1–32, 2000.

[42] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern classification*. John Wiley & Sons, 2012.

[43] T. Duncan, "Stochastic systems in riemannian manifolds," *Journal of Optimization theory and applications*, vol. 27, no. 3, pp. 399–426, 1979.

[44] J. C. Dunn, *A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters*. Taylor & Francis, 1973.

[45] E. Elhamifar and R. Vidal, "Sparse manifold clustering and embedding," in *Advances in Neural Information Processing Systems*, 2011, pp. 55–63.

[46] A. Ess, B. Leibe, and L. Van Gool, "Depth and appearance for mobile scene analysis," in *International Conference on Computer Vision*, 2007, pp. 1–8.

[47] M. Faraki, M. Harandi, and F. Porikli, "More about VLAD: a leap from euclidean to riemannian manifolds," in *Computer Vision and Pattern Recognition*, jun 2015, pp. 1704–1716.

[48] M. Faraki, M. T. Harandi, A. Wiliem, and B. C. Lovell, "Fisher tensors for classifying human epithelial cells," *Pattern Recognition*, vol. 47, no. 7, pp. 2348–2359, 2014.

[49] M. Faraki, M. Palhang, and C. Sanderson, "Log-euclidean bag of words for human action recognition," in *IET Computer Vision*, 2014.

[50] M. Filippone, F. Camastra, F. Masulli, and S. Rovetta, "A survey of kernel and spectral methods for clustering," *Pattern Recognition*, vol. 41, no. 1, pp. 176–190, 2008.

[51] P. T. Fletcher, J. Moeller, J. M. Phillips, and S. Venkatasubramanian, "Horoball hulls and extents in positive definite space," in *Algorithms and Data Structures*, 2011, pp. 386–398.

[52] P. Frankl and H. Maehara, "The johnson-lindenstrauss lemma and the sphericity of some graphs," *Journal of Combinatorial Theory, Series B*, vol. 44, no. 3, pp. 355–362, 1988.

[53] S. Gao, I. W.-H. Tsang, and L.-T. Chia, "Kernel sparse representation for image classification and face recognition," in *European Conference on Computer Vision*, 2010, pp. 1–14.

[54] C. E. Ginestet, A. Simmons, and E. D. Kolaczyk, "Weighted frechet means as convex combinations in metric spaces: Properties and generalized median inequalities," *Statistics & Probability Letters*, vol. 82, no. 10, pp. 1859–1863, 2012.

[55] N. Goel, G. Bebis, and A. Nefian, "Face recognition experiments with random projection," in *Defense and Security*. International Society for Optics and Photonics, 2005, pp. 426–437.

[56] K. Guo, P. Ishwar, and J. Konrad, "Action recognition in video by sparse representation on covariance manifolds of silhouette tunnels," in *Recognizing Patterns in Signals, Speech, Images and Videos*. Springer, 2010, pp. 294–305.

[57] K. Guo, P. Ishwar, and J. Konrad, "Action recognition using sparse representation on covariance manifolds of optical flow," in *Advanced Video and Signal Based Surveillance*, 2010, pp. 188–195.

[58] J. Hamm and D. D. Lee, "Extended grassmann kernels for subspace-based learning," in *Advances in neural information processing systems*, 2008, pp. 601–608.

[59] J. Hamm and D. D. Lee, "Grassmann discriminant analysis: a unifying view on subspace-based learning," in *International Conference on Machine Learning*, 2008, pp. 376–383.

[60] J. Hamm and D. D. Lee, "Grassmann discriminant analysis: a unifying view on subspace-based learning," in *International Conference on Machine Learning*, 2008, pp. 376–383.

[61] M. Harandi, R. Hartley, C. Shen, B. Lovell, and C. Sanderson, "Extrinsic methods for coding and dictionary learning on grassmann manifolds," *International Journal of Computer Vision*, vol. 114, no. 2-3, pp. 113–136, 2015.

[62] M. Harandi and M. Salzmann, "Riemannian coding and dictionary learning: Kernels to the rescue," in *Computer Vision and Pattern Recognition*, June 2015.

[63] M. Harandi, C. Sanderson, C. Shen, and B. C. Lovell, "Dictionary learning and sparse coding on grassmann manifolds: An extrinsic solution," in *International Conference on Computer Vision*, December 2013.

[64] M. T. Harandi, M. Salzmann, and R. Hartley, "From manifold to manifold: geometry-aware dimensionality reduction for spd matrices," in *Computer Vision*. Springer, 2014, pp. 17–32.

[65] M. T. Harandi, M. Salzmann, S. Jayasumana, R. Hartley, and H. Li, "Expanding the family of grassmannian kernels: An embedding perspective," in *European Conference on Computer Vision*, 2014, pp. 408–423.

[66] M. T. Harandi, C. Sanderson, R. Hartley, and B. C. Lovell, "Sparse coding and dictionary learning for symmetric positive definite matrices: A kernel approach," in *European Conference on Computer Vision*, 2012, pp. 216–229.

[67] M. T. Harandi, C. Sanderson, R. Hartley, and B. C. Lovell, "Sparse coding and dictionary learning for symmetric positive definite matrices: A kernel approach," in *European Conference on Computer Vision*, 2012, pp. 216–229.

[68] M. T. Harandi, C. Sanderson, S. Shirazi, and B. C. Lovell, "Graph embedding discriminant analysis on grassmannian manifolds for improved image set matching," in *Computer Vision and Pattern Recognition*. IEEE, 2011, pp. 2705–2712.

[69] M. T. Harandi, C. Sanderson, S. Shirazi, and B. C. Lovell, "Kernel analysis on grassmann manifolds for action recognition," *Pattern Recognition Letters*, vol. 34, no. 15, pp. 1906–1915, 2013.

[70] M. Harandi, C. Sanderson, A. Wiliem, and B. Lovell, "Kernel analysis over Riemannian manifolds for visual recognition of actions, pedestrians and textures," in *Winter Conference on Applications of Computer Vision*, 2012, pp. 433–439.

[71] P. Hobson, B. C. Lovell, G. Percannella, M. Vento, and A. Wiliem, "Benchmarking human epithelial type 2 interphase cells classification methods on a very large dataset," *Artificial Intelligence in Medicine*, 2015.

[72] X. Hong, H. Chang, S. Shan, X. Chen, and W. Gao, "Sigma set: A small second order statistical region descriptor," in *Computer Vision and Pattern Recognition*, 2009, pp. 1802–1809.

[73] Y. Hu, A. S. Mian, and R. Owens, "Sparse approximated nearest points for image set classification," in *Computer Vision and Pattern Recognition*, 2011, pp. 121–128.

[74] Z. Huang, R. Wang, S. Shan, and X. Chen, "Projection metric learning on grassmann manifold with application to video based face recognition," in *Computer Vision and Pattern Recognition*, 2015, pp. 140–149.

[75] A. K. Jain, "Data clustering: 50 years beyond k-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010.

[76] S. Jayasumana, R. Hartley, M. Salzmann, H. Li, and M. Harandi, "Kernel methods on the riemannian manifold of symmetric positive definite matrices," in *Computer Vision and Pattern Recognition*, 2013, pp. 73–80.

[77] S. Jayasumana, R. Hartley, M. Salzmann, H. Li, and M. Harandi, "Kernel methods on riemannian manifolds with gaussian rbf kernels," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 12, pp. 2464–2477, 2015.

[78] S. Jayasumana, M. Salzmann, H. Li, and M. Harandi, "A framework for shape analysis via hilbert space embedding," in *International Conference on Computer Vision*, 2013, pp. 1249–1256.

[79] W. B. Johnson and J. Lindenstrauss, "Extensions of lipschitz mappings into a hilbert space," *Contemporary Mathematics*, vol. 26, no. 189-206, p. 1, 1984.

[80] H. Karcher, "Riemannian center of mass and mollifier smoothing," *Communications on pure and applied mathematics*, vol. 30, no. 5, pp. 509–541, 1977.

[81] M. Kloft, U. Brefeld, S. Sonnenburg, and A. Zien, "ℓ_p-norm multiple kernel learning," *Journal of Machine Learning Research*, vol. 12, pp. 953–997, 2011.

[82] I. Kotsia and I. Pitas, "Facial expression recognition in image sequences using geometric deformation features and support vector machines," *IEEE Transactions on Image Processing*, vol. 16, no. 1, pp. 172–187, 2007.

[83] B. Kulis and K. Grauman, "Kernelized locality-sensitive hashing," *International Conference on Machine Learning*, vol. 34, no. 6, pp. 1092–1104, 2012.

[84] E. Kushilevitz, R. Ostrovsky, and Y. Rabani, "Efficient search for approximate nearest neighbor in high dimensional spaces," *SIAM Journal on Computing*, vol. 30, no. 2, pp. 457–474, 2000.

[85] J. Lawson and Y. Lim, "Monotonic properties of the least squares mean," *Mathematische Annalen*, vol. 351, no. 2, pp. 267–279, 2011.

[86] B. Leibe and B. Schiele, "Analyzing appearance and contour based methods for object categorization," in *Computer Vision and Pattern Recognition*, 2003, pp. II–409.

[87] N. E. Leonard and P. S. Krishnaprasad, "Motion control of drift-free, left-invariant systems on lie groups," *IEEE Transactions on Automatic Control*, vol. 40, no. 9, pp. 1539–1554, 1995.

[88] A. D. Leow, I. Yanovsky, N. Parikshak, X. Hua, S. Lee, A. W. Toga, C. R. Jack, M. A. Bernstein, P. J. Britson, J. L. Gunter *et al.*, "Alzheimer's disease neuroimaging initiative: a one-year follow up study using tensor-based morphometry correlating degenerative rates, biomarkers and cognition," *Neuroimage*, vol. 45, no. 3, pp. 645–655, 2009.

[89] P. Li, T. J. Hastie, and K. W. Church, "Very sparse random projections," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data mining*, 2006, pp. 287–296.

[90] X. Li, W. Hu, Z. Zhang, and X. Zhang, "Robust visual tracking based on an effective appearance model," in *European Conference on Computer Vision*. Springer, 2008, pp. 396–408.

[91] X. Li, W. Hu, Z. Zhang, X. Zhang, M. Zhu, and J. Cheng, "Visual tracking via incremental log-euclidean riemannian subspace learning," in *Computer Vision and Pattern Recognition*. IEEE, 2008, pp. 1–8.

[92] D. Lin, S. Yan, and X. Tang, "Pursuing informative projection on grassmann manifold," in *Computer Vision and Pattern Recognition*, vol. 2. IEEE, 2006, pp. 1727–1734.

[93] T. Lin, H. Zha, and S. U. Lee, "Riemannian manifold learning for nonlinear dimensionality reduction," in *European Conference on Computer Vision*, 2006, pp. 44–55.

[94] J. Liu and N. Zheng, "Gait history image: a novel temporal template for gait recognition," in *International Conference on Multimedia and Expo*, 2007, pp. 663–666.

[95] M. Liu, S. Li, S. Shan, R. Wang, and X. Chen, "Deeply learning deformable facial action parts model for dynamic expression analysis," in *Asian Conference on Computer Vision*. Springer, 2014, pp. 143–157.

[96] M. Liu, S. Shan, R. Wang, and X. Chen, "Learning expressionlets on spatio-temporal manifold for dynamic facial expression recognition," in *Computer Vision and Pattern Recognition*, 2014, pp. 1749–1756.

[97] M. Liu, R. Wang, S. Li, Z. Huang, S. Shan, and X. Chen, "Video modeling and learning on riemannian manifold for emotion recognition in the wild," *Journal on Multimodal User Interfaces*, pp. 1–12, 2015.

[98] X. LIU, A. SRIVASTAVA, and K. GALLIVAN, "Optimal linear representations of images for object recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 5, pp. 662–666, 2004.

[99] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[100] G. Lu, Y. Zhou, X. Li, and M. Kudo, "Efficient action recognition via local position offset of 3d skeletal body joints," *Multimedia Tools and Applications*, vol. 75, no. 6, pp. 3479–3494, 2016.

[101] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews, "The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression," in *Computer Vision and Pattern Recognition Workshops*. IEEE, 2010, pp. 94–101.

[102] Y. M. Lui and J. R. Beveridge, "Grassmann registration manifolds for face recognition," in *European Conference on Computer Vision*. Springer, 2008, pp. 44–57.

[103] Y. M. Lui, J. R. Beveridge, and M. Kirby, "Action classification on product manifolds," in *Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 833–839.

[104] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to information retrieval*. Cambridge University Press, 2008, vol. 1.

[105] M. Moakher, "A differential geometric approach to the geometric mean of symmetric positive-definite matrices," *SIAM Journal on Matrix Analysis and Applications*, vol. 26, no. 3, pp. 735–747, 2005.

[106] G. Nalbantov, P. Groenen, and C. Bioch, "Nearest convex hull classification," Econometric Institute Research Papers, Tech. Rep., 2006.

[107] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern Recognition*, vol. 29, no. 1, pp. 51–59, 1996.

[108] N. R. Pal and K. Sarkar, "What and when can we gain from the kernel versions of c-means algorithm?" *IEEE Transactions on Fuzzy Systems*, vol. 22, no. 2, pp. 363–379, 2014.

[109] Y. Pang, Y. Yuan, and X. Li, "Gabor-based region covariance matrices for face recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 7, pp. 989–993, 2008.

[110] X. Pennec, "Intrinsic statistics on riemannian manifolds: Basic tools for geometric measurements," *Journal of Mathematical Imaging and Vision*, vol. 25, no. 1, pp. 127–154, 2006.

[111] X. Pennec, P. Fillard, and N. Ayache, "A riemannian framework for tensor computing," *International Journal of Computer Vision*, vol. 66, no. 1, pp. 41–66, 2006.

[112] X. Pennec, P. Fillard, and N. Ayache, "A riemannian framework for tensor computing," *International Journal of Computer Vision*, vol. 66, no. 1, pp. 41–66, 2006.

[113] T. N. Phyu, "Survey of classification techniques in data mining," in *International MultiConference of Engineers and Computer Scientists*, 2009, pp. 18–20.

[114] F. Porikli, O. Tuzel, and P. Meer, "Covariance tracking using model update based on lie algebra," in *Computer Vision and Pattern Recognition*, vol. 1. IEEE, 2006, pp. 728–735.

[115] J. Qing, H. Huo, and T. Fang, "Nearest convex hull classifiers for remote sensing classification," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 37, pp. 589–594, 2008.

[116] T. Randen and J. H. Husoy, "Filtering for texture classification: A comparative study," *International Conference on Machine Learning*, vol. 21, no. 4, pp. 291–310, 1999.

[117] J. Rice, *Mathematical statistics and data analysis*. Cengage Learning, 2006.

[118] T. Sakai and A. Imiya, "Fast spectral clustering with random projection and sampling," in *Machine Learning and Data Mining in Pattern Recognition*, 2009, pp. 372–384.

[119] A. Salaheldin, S. Maher, and M. El Helw, "Robust real-time tracking with diverse ensembles and random projections," in *International Conference on Computer Vision Workshops*, 2013, pp. 112–120.

[120] A. C. Sankaranarayanan, P. K. Turaga, R. G. Baraniuk, and R. Chellappa, "Compressive acquisition of dynamic scenes," in *European Conference on Computer Vision*, 2010, pp. 129–142.

[121] M. W. Schmidt, E. Berg, M. P. Friedlander, and K. P. Murphy, "Optimizing costly functions with simple constraints: A limited-memory projected quasi-newton algorithm," in *International Conference on Artificial Intelligence and Statistics*, 2009.

[122] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation*, vol. 10, no. 5, pp. 1299–1319, 1998.

[123] W. R. Schwartz and L. S. Davis, "Learning discriminative appearance-based models using partial least squares," in *SIBGRAPI*, 2009, pp. 322–329.

[124] H. S. Seung and D. D. Lee, "The manifold ways of perception," *Science*, vol. 290, no. 5500, pp. 2268–2269, 2000.

[125] J. Shawe-Taylor and N. Cristianini, *Kernel methods for pattern analysis*. Cambridge University Press, 2004.

[126] Q. Shi, S. Shen, R. Hill, and A. van den Hengel, "Is margin preserved after random projection?" in *International Conference on Machine Learning*, 2012, pp. 591–598.

[127] S. Shirazi, M. T. Harandi, C. Sanderson, A. Alavi, and B. C. Lovell, "Clustering on grassmann manifolds via kernel embedding with application to action analysis," in *International Conference on Image Processing*, 2012, pp. 781–784.

[128] R. Sivalingam, D. Boley, V. Morellas, and N. Papanikolopoulos, "Tensor sparse coding for region covariances," in *European Conference on Computer Vision*, 2010, pp. 722–735.

[129] R. Sivalingam, D. Boley, V. Morellas, and N. Papanikolopoulos, "Tensor sparse coding for region covariances," in *European Conference on Computer Vision*, 2010, pp. 722–735.

[130] R. Sivalingam, V. Morellas, D. Boley, and N. Papanikolopoulos, "Metric learning for semi-supervised clustering of region covariance descriptors," in *International Conference on Distributed Smart Cameras*. IEEE, 2009, pp. 1–8.

[131] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *CRCV-TR-12-01, 2012*, 2012.

[132] S. Sra, "Positive definite matrices and the s-divergence," in *SIAM Journal on Matrix Analysis and Applications*, 2013.

[133] S. Sra, "Positive Definite Matrices and the S-Divergence," *Proceedings of the American Mathematical Society*, 2015.

[134] C. H. Suryanto, H. Saigo, and K. Fukui, "Protein clustering on a grassmann manifold," in *Pattern Recognition in Bioinformatics*, 2012, pp. 71–81.

[135] P. Terán and I. Molchanov, "The law of large numbers in a metric space with a convex combination operation," *Journal of Theoretical Probability*, vol. 19, no. 4, pp. 875–898, 2006.

[136] D. Tosato, M. Farenzena, M. Spera, V. Murino, and M. Cristani, "Multi-class classification on riemannian manifolds for video surveillance," in *European conference on computer vision*. Springer, 2010, pp. 378–391.

[137] D. Tosato, M. Spera, M. Cristani, and V. Murino, "Characterizing humans on riemannian manifolds," *International Conference on Machine Learning*, vol. 35, no. 8, pp. 1972–1984, 2013.

[138] L. W. Tu, *An introduction to manifolds*. Springer, 2011.

[139] P. Turaga, A. Veeraraghavan, A. Srivastava, and R. Chellappa, "Statistical computations on grassmann and stiefel manifolds for image and video-based recognition," *International Conference on Machine Learning*, vol. 33, pp. 2273–2286, 2011.

[140] P. Turaga and R. Chellappa, "Locally time-invariant models of human activities using trajectories on the grassmannian," in *Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 2435–2441.

[141] P. K. Turaga, A. Veeraraghavan, and R. Chellappa, "From videos to verbs: Mining videos for events using a cascade of dynamical systems," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[142] O. Tuzel, F. Porikli, and P. Meer, "Region covariance: A fast descriptor for detection and classification," in *European Conference on Computer Vision*, 2006, pp. 589–600.

[143] O. Tuzel, F. Porikli, and P. Meer, "Human detection via classification on riemannian manifolds," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2007, pp. 1–8.

[144] O. Tuzel, F. Porikli, and P. Meer, "Pedestrian detection via classification on riemannian manifolds," *International Conference on Machine Learning*, vol. 30, no. 10, pp. 1713–1727, 2008.

[145] C. Udriste, *Convex functions and optimization methods on Riemannian manifolds*. Springer Science & Business Media, 1994, vol. 297.

[146] A. Veeraraghavan, A. K. Roy-Chowdhury, and R. Chellappa, "Matching shape sequences in video with applications in human movement analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 12, pp. 1896–1909, 2005.

[147] S. S. Vempala, *The random projection method*. American Mathematical Soc., 2004, vol. 65.

[148] R. Vemulapalli, J. K. Pillai, and R. Chellappa, "Kernel learning for extrinsic classification of manifold features," in *Computer Vision and Pattern Recognition*, 2013, pp. 1782–1789.

[149] P. Vincent and Y. Bengio, "K-local hyperplane and convex distance nearest neighbor algorithms," in *Advances in Neural Information Processing Systems*, 2001, pp. 985–992.

[150] G. Walsh, A. Sarti, and S. Sastry, "Algorithms for steering on the group of rotations," in *American Control Conference*. IEEE, 1993, pp. 1312–1316.

[151] R. Wang, H. Guo, L. S. Davis, and Q. Dai, "Covariance discriminative learning: A natural and efficient approach to image set classification," in *Computer Vision and Pattern Recognition*, 2012, pp. 2496–2503.

[152] R. Wang, S. Shan, X. Chen, and W. Gao, "Manifold-manifold distance with application to face recognition based on image set," in *Computer Vision and Pattern Recognition*. IEEE, 2008, pp. 1–8.

[153] T. Wang and P. Shi, "Kernel grassmannian distances and discriminant analysis for face recognition from image sets," *Pattern Recognition Letters*, vol. 30, no. 13, pp. 1161–1165, 2009.

[154] Y. Wang and G. Mori, "Human action recognition by semilatent topic models," *International Conference on Machine Learning*, vol. 31, no. 10, pp. 1762–1774, 2009.

[155] Z. Wang, N. Djuric, K. Crammer, and S. Vucetic, "Trading representability for scalability: adaptive multi-hyperplane machine for nonlinear classification," in *SIGKDD*, 2011, pp. 24–32.

[156] Z. Wang, S. Wang, and Q. Ji, "Capturing complex spatio-temporal relations among facial muscles for facial expression recognition," in *Computer Vision and Pattern Recognition*, 2013, pp. 3422–3429.

[157] D. S. Watkins, *Fundamentals of matrix computations*. John Wiley & Sons, 2004, vol. 64.

[158] A. Wiliem, R. Vemulapalli, and B. C. Lovell, "Explicit discriminative representation for improved classification of manifold features," *Pattern Recognition Letters*, 2016.

[159] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *International Conference on Machine Learning*, vol. 31, no. 2, pp. 210–227, 2009.

[160] Y. Wu, B. Wu, J. Liu, and H. Lu, "Probabilistic tracking on riemannian manifolds," in *International Conference on Pattern Recognition*. IEEE, 2008, pp. 1–4.

[161] Y. Xie, J. Ho, and B. Vemuri, "On a nonlinear generalization of sparse coding and dictionary learning," in *Journal of Machine Learning Research: Workshop and Conference Proceedings*, vol. 28, no. 3, 2013, pp. 1480–1488.

[162] X. Xiong and F. Torre, "Supervised descent method and its applications to face alignment," in *Computer Vision and Pattern Recognition*, 2013, pp. 532–539.

[163] S. Yan and X. Tang, "Trace quotient problems revisited," in *European Conference on Computer Vision*. Springer, 2006, pp. 232–244.

[164] C. Ye, J. Liu, C. Chen, M. Song, and J. Bu, "Speech emotion classification on a riemannian manifold," in *Advances in Multimedia Information Processing*. Springer, 2008, pp. 61–69.

[165] S. Yu, D. Tan, and T. Tan, "A framework for evaluating the effect of view angle, clothing and carrying condition on gait recognition," in *International Conference on Pattern Recognition*, vol. 4, 2006, pp. 441–444.

[166] C. Yuan, W. Hu, X. Li, S. Maybank, and G. Luo, "Human action recognition under log-euclidean riemannian metric," in *Asian Conference on Computer Vision*, 2010, pp. 343–353.

[167] C. Yuan, W. Hu, X. Li, S. Maybank, and G. Luo, "Human action recognition under log-Euclidean Riemannian metric," in *Asian Conference on Computer Vision*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2010, vol. 5994, pp. 343–353.

[168] S. Zheng, J. Zhang, K. Huang, R. He, and T. Tan, "Robust view transformation model for gait recognition," in *International Conference on Image Processing*, 2011, pp. 2073–2076.

[169] X. Zhou and Y. Shi, "Nearest neighbor convex hull classification method for face recognition," in *International Conference on Computational Science*. Springer, 2009, pp. 570–577.

[170] S. Zhu, C. Li, C. Change Loy, and X. Tang, "Face alignment by coarse-to-fine shape searching," in *Computer Vision and Pattern Recognition*, 2015, pp. 4998–5006.