1-1-2014

Grassmann Learning for Recognition and Classification

Sherif Azary

Grassmann Learning for Recognition and Classification

Ph.D. Dissertation

by

Sherif Azary

B. THOMAS GOLISANO COLLEGE OF COMPUTING AND INFORMATION SCIENCES

DEPARTMENT OF COMPUTING AND INFORMATION SCIENCES-PHD

ROCHESTER INSTITUTE OF TECHNOLOGY

ROCHESTER, NEW YORK

Date:

Monday, April 28, 2014

Dissertation Advisor:

Dr. Andreas Savakis, Professor, Department of Computer Engineering, RIT

Dissertation Committee:

Dr. Nathan D. Cahill, Associate Professor, School of Mathematical Sciences, RIT

Dr. Shanchieh J. Yang, Associate Professor & Department Head, Department of Computer Engineering, RIT

Dr. Richard Zanibbi, Associate Professor, Department of Computer Science, RIT

Defense Chair:

Dr. Dhireesha Kudithipudi, Associate Professor, Department of Computer Engineering, RIT

B. THOMAS GOLISANO COLLEGE OF

COMPUTING AND INFORMATION SCIENCES

ROCHESTER INSTITUTE OF TECHNOLOGY

ROCHESTER, NEW YORK

CERTIFICATE OF APPROVAL

The Ph.D. Degree Dissertation of Sherif Azary has been examined and approved by the dissertation committee as complete and satisfactory for the dissertation requirement for Ph.D. degree in Computing and Information Sciences

_____

Dr. Andreas Savakis, Advisor (date)

_____

Dr. Nathan D. Cahill, Member (date)

_____

Dr. Shanchieh J. Yang, Member (date)

_____

Dr. Richard Zanibbi, Member (date)

_____

Dr. Dhireesha Kudithipudi, Defense Chair (date)

_____

Dr. Pengcheng Shi, Ph.D. Program Director (date)

B. Thomas College of Computing and Information Sciences

Abstract

Computational performance associated with high-dimensional data is a common challenge for real-world classification and recognition systems. Subspace learning has received considerable attention as a means of finding an efficient low-dimensional representation that leads to better classification and efficient processing. A Grassmann manifold is a space that promotes smooth surfaces, where points represent subspaces and the relationship between points is defined by a mapping of an orthogonal matrix. Grassmann learning involves embedding high dimensional subspaces and kernelizing the embedding onto a projection space where distance computations can be effectively performed.

In this dissertation, Grassmann learning and its benefits towards action classification and face recognition in terms of accuracy and performance are investigated and evaluated. Grassmannian Sparse Representation (GSR) and Grassmannian Spectral Regression (GRASP) are proposed as Grassmann inspired subspace learning algorithms. GSR is a novel subspace learning algorithm that combines the benefits of Grassmann manifolds with sparse representations using least squares loss ℓ^1-norm minimization for improved classification. GRASP is a novel subspace learning algorithm that leverages the benefits of Grassmann manifolds and Spectral Regression in a framework that supports high discrimination between classes and achieves computational benefits by using manifold modeling and avoiding eigen-decomposition. The effectiveness of GSR and GRASP is demonstrated for computationally intensive classification problems: (a) multi-view action classification using the IXMAS Multi-View dataset, the i3DPost Multi-View dataset, and the WVU Multi-View dataset, (b) 3D action classification using the MSRAction3D dataset and MSRGesture3D dataset, and (c) face

recognition using the ATT Face Database, Labeled Faces in the Wild (LFW), and the Extended Yale Face Database B (YALE).

Additional contributions include the definition of Motion History Surfaces (MHS) and Motion Depth Surfaces (MDS) as descriptors suitable for activity representations in video sequences and 3D depth sequences. An in-depth analysis of Grassmann metrics is applied on high dimensional data with different levels of noise and data distributions which reveals that standardized Grassmann kernels are favorable over geodesic metrics on a Grassmann manifold. Finally, an extensive performance analysis is made that supports Grassmann subspace learning as an effective approach for classification and recognition.

Acknowledgements

I would like to express my deepest gratitude to my advisor, Dr. Andreas Savakis, for his guidance, support, patience, mentorship, and encouragement in making this research possible. Dr. Andreas Savakis has always been accommodating to my schedule and encouraged me to complete my research while I worked full-time and moved away from Rochester, NY to further my career. Without this encouragement I would not be where I am today and I am grateful. Dr. Andreas Savakis and Dr. Shanchieh J. Yang have also provided financial support for my PhD research and I thank them for believing in me.

I would also like to thank my committee and defense chair, Dr. Nathan D. Cahill, Dr. Shanchieh J. Yang, Dr. Richard Zanibbi, and Dr. Dhireesha Kudithipudi for their support and advice in making this dissertation possible. They have always been accommodating to my schedule and extremely supportive to make this research productive and stimulating. It has been an honor. I would like to thank Dr. Pengcheng Shi for accepting me into the Computing and Information Science Ph.D. program and believing in me to complete my research as a part-time student with a full-time job.

I would like to thank my colleagues, Dr. Grigorios Tsagkatakis and Dr. Raymond Ptucha, for their friendship, academic advice, and words of encouragement. I also cannot express enough appreciation to Joyce Hart, Kathryn Stefanik, and Lorrie Jo Turner for their academic support.

I would like to thank my brother and his wife, Hani and Mary Azary, for their continuous words of encouragement and being great friends. I would like to thank my brother in law, Johnny Girgis, and my good friend, Pete Henen, for their friendship and humor. I would like to thank my parents, Wahid and Samia Azary, for their patience, love, and raising me with a passion for problem solving and math. I owe the success of my education and my career to them. Finally, I would like to thank my wife, May, for always being there for me, and for her love and understanding, and for being my best friend.

Table of Contents

1 Introduction

The automatic recognition of human actions a fundamental but challenging task in computer vision research for a wide variety of applications including autonomous surveillance, law enforcement, health care monitoring systems, and human computer interfacing. Automatic face recognition is another important task for many applications. The main challenge of such systems is their ability to classify in unconstrained environments. Images of human actors can vary by their sizes, shapes, poses, occlusions, viewpoint variations, noise, and lighting. Additionally, action classification systems would need to account for action execution speed requiring spatio-temporal representations that are invariant to such factors.

The most common approaches to classification involve extracting meaningful features from images or video and applying statistical or machine learning tools to make classification decisions. Optimal action representations are those that can capture both the spatial structure of an activity and its temporal structure over time. While many features can represent spatial and temporal domains independently, there are spatio-temporal features that are capable of representing both domains, such as space-time interest points and 3D Harris corner detectors. Such features are well-suited for challenging applications such as multi-view and 3D action classification systems. Within these domains are a wide variety of representations involving normalization, invariance, and exhaustive search. Similarly, face image representations are expected to be robust enough to distinguish between a wide range of human subjects and under unconstrained conditions such as variations in illumination and facial expressions. Local binary patterns and local ternary patterns are among the most popular face image representations.

Methodologies that can account for the statistical and geometric properties of high dimensional representations have proven to be extremely valuable in deriving meaningful

information. Principal component analysis (PCA) is a common dimensionality reduction method based on the eigenvectors of the covariance matrix. Although fast, PCA does not maintain geometry and local structuring of high dimensional data. Manifold learning techniques have been developed to handle non-linear dimensionality reduction. Manifold learning involves reducing high dimensional data to a lower dimensional space while optimally preserving the local geometries from the high dimensional information. An ideal mapping should be fast, preserve clustering, and account for occlusions and outliers. There are many dimensionality reduction algorithms that are powerful enablers of robust classification and in this dissertation the benefits and drawbacks of many of these methods are discussed. As an alternative, sparse representations are methods of finding sparse solutions that are useful in a variety of applications including classification.

Grassmann learning is a dimensionality reduction algorithm where subspaces are mapped as points onto a smooth and curved surface where distances between subspaces are geodesic. The main advantage of Grassmann learning over traditional manifold learning methods is that high dimensional feature representations may not typically lie on a Euclidean space. Grassmann learning maps subspaces onto points based on orthogonal constraints, promoting high between-class discrimination by their geometrical structuring, and accounting for missing data through subspace spanning. Grassmann kernelization embeds subspaces onto a projection space where distance computations can be effectively performed.

In this dissertation, representations for action classification and face recognition systems are explored in Chapter 2. Spatio-temporal surface descriptors for multi-view and 3D action classification systems are presented using radial distance measures and 3D joint descriptors for multi-view and 3D action classification. These surfaces have proven to be effective at

representing actions while being invariant to time, scale, and localization. These spatio-temporal surface representations motivated the development of more robust motion surface representations. Motion surfaces, proposed in this dissertation, have proven to be very effective representations for describing where motion exists in a scene and how motion evolves over time. Motion history surface (MHS) and motion depth surface (MDS) descriptors are suitable for activity representations for multi-view and 3D depth action sequences.

In Chapter 3 dimensionality reduction algorithms including principal component analysis, multidimensional scaling, local linear embedding, and linear extensions of graph embedding are discussed and evaluated. The benefits and drawbacks of these methods are identified including time complexities. Grassmann learning and its benefits towards action classification and face recognition in terms of accuracy and performance are investigated in Chapter 4. Grassmann learning in a kernelized principal component analysis framework is defined and evaluated. In Chapter 5, Grassmannian Sparse Representation (GSR) is proposed as a Grassmann inspired subspace learning algorithm. GSR is a novel subspace learning algorithm that combines the benefits of Grassmann manifolds with sparse representations using least squares loss ℓ^1-norm minimization for improved classification. Sparse representations are introduced as a method for finding sparse solutions for underdetermined systems. Images and video sequences can be encoded using sparse representations to be more easily interpretable and classification using least squares loss ℓ^1-norm minimization shows to be suitable for classification at the cost of poor computational performance. This framework is extended into a Grassmann learning framework through GSR. The high cost of poor performance through GSR encouraged the pursuit of a faster learning framework. Grassmannian Spectral Regression (GRASP) is introduced in Chapter 6. GRASP is a novel subspace learning algorithm that leverages the benefits of

Grassmann manifolds and Spectral Regression in a framework that supports high discrimination between classes and achieves computational benefits by using manifold modeling and avoiding eigen-decomposition.

In Chapter 7, the classification accuracies and performance of all previously discussed learning methods including GSR and GRASP are presented for computationally intensive action and face datasets. An in-depth analysis of Grassmann metrics is applied on high dimensional data with different levels of noise and data distributions revealing that standardized Grassmann kernels are favorable over Grassmann geodesic metrics in a Grassmann space. GSR and GRASP are compared against existing sparse representations, manifold learning, and Grassmann learning methodologies. An extensive performance analysis is made that support Grassmann subspace learning through GSR and GRASP as effective approaches for classification and recognition over state-of-the-art approaches. The dissertation concludes in Chapter **Error! Reference source not found.**.

1.1 Contributions

In this section the contributions made in this dissertation are explicitly defined. The first is the definition of radial distances and radial distance surfaces as action representations. Such surfaces have shown to be suitable representations for multi-view action classification. This work was extended to handle 3D action sequences using 3D joint surface descriptors. This led to the evolution of motion surfaces where motion history surfaces (MHS) and motion depth surfaces (MDS) are proposed as descriptors that can accurately represent motion in multi-view and 3D action classification applications.

The main contributions of this dissertation are the definition of Grassmannian Sparse Representations and Grassmannian Spectral Regression for high classification accuracy and computational performance. With this, an extensive evaluation is made on Grassmann metrics which is not found at this level of depth in the existing literature. Through experiments and evaluation, this dissertation exposes the benefit of using Grassmann kernels with robust classifiers over geodesic metrics using kernel standardization. Additionally, a thorough time complexity evaluation is made on all learning methods.

2 Representations for Action Classification and Face Recognition

2.1 Action Representations

Weinland et al. [1] discuss a broad range of spatial, temporal, and spatio-temporal approaches for addressing action classification problems. Spatial action representations attempt to describe the spatial structure of actions. Body models [2], body pose estimations [3], kinematic joint models [4], and stick figures [5] tend to be intuitive and descriptive, but may require significant training and computational resources. Spatial parametric image features include contour/silhouette representations [6], optical flow [7], and motion history images/motion energy images [8]. Such features do not require body part labeling or tracking, but are computationally intensive because of high dimensional data representations and difficulties with occlusions. Spatial statistical approaches are based on the statistics of local features, such as features detected using the Harris corner detector [9]. Local feature descriptors can be classified using Bag of Features [10], Support Vector Machines (SVM's) [11], local Principal Component Analysis (local PCA) [12], and Manifold Learning - e.g. supervised locality preserving projections (sLPP) [13]. The main benefits of spatial statistical representations are that they are not relying on body part labeling, silhouette extraction, and localization. However, such representations are usually unordered and of varying sizes making it difficult to use with classifiers.

Temporal representations of human actions identify the temporal structure of an action and are categorized into action grammars [14], action templates [15], and temporal statistics [16]. Action grammars identify an action by a set of action primitives. Given a set of all action primitives, an action grammar acts as a function to learn the transitions between those primitives. A popular method for identifying action primitive transitions is the use of Hidden Markov Models (HMM). Many action recognition systems utilize action grammars with HMM's

including Kruger and Grest [17] and Chakraborty et al. [18]. Action grammars are highly modular but require manual structuring making action grammars impractical for systems intended to classify a large set of action classes. Action templates are a combination of action primitives into one larger representation. Pattern matching is usually applied to compare actions to a collection of action templates in a database. Junejo et al. [19] propose a view independent approach to action recognition on 2D video sequences using Self Similarity Matrices (SSM). Their approach captures temporal histograms of gradient orientations in the spatial domain and concatenates the features descriptors into one large local SSM feature vector descriptor. This feature vector descriptor is an action template. Yao et al. [20] collect action pose templates as a combination of Histogram of Gradient (HoG) features and Histogram of Optical Flow (HoF) features. These templates are classified using Support Vector Machines (SVM's). Action templates are known to be effective and discriminative, but do not have a built-in mechanism to account for temporal segmentation. Temporal statistics find statistical patterns of actions in the temporal domain such as identifying frequent features over time.

Spatio-temporal representations are those that can describe an action structure in both the spatial and temporal domains. One of the earliest spatio-temporal feature descriptors was introduced by Laptev and Lindeberg [21] by extending on the Harris Corner Detector algorithm to detect space-time interest points that can be used to represent motion-based activities. Other spatio-temporal interest points include cuboids using temporal Gabor filters [22], Harris 3D detectors as a 3D extension of the Harris corner detector for detecting significant local variations on both space and time [23], and Hessian detectors that are scale and affine invariant across the space and time domains [24]. Vili et al. [25] introduce dynamic texture descriptors to describe human movement. A human action is represented as a volume in XYT space and Local Binary

Patterns are used to extract histogram features in the XT and YT spaces. An interesting action representation that inspired the action descriptor used in this dissertation is the spatio-temporal action surfaces covered by Souvenir and Parrigan [26]. The 2D Radon transform was applied on each frame of an action and converted to a 1D signal called the R-Transform. A surface was created as a sequence of these signals called the RXS surface. These surfaces are then scaled down to a standard time interval while preserving action information supporting the concept of spatio-temporal invariance.

2.1.1 Action Representations for Multi-View and 3D Applications

Autonomous action classification systems can be restricted by visual sensor constraints or benefit from their physical positions in a scene. Multiple view recognition systems tend to use standard RGB cameras, while 3D cameras used in the gaming industry can provide both color and depth information. An overview of multi-view and 3D action classification systems are discussed in this section.

Weinland et al. [1] explain that viewpoint independence is commonly addressed by normalization, invariance, or exhaustive search. View normalization is based on correcting the current view through a transformation to a canonical view. This approach is taken in the work of Gkalelis et al. [27] who use multi-view posture vectors of synchronized frames along with a combination of circular shifting Discrete Fourier Transforms to determine the posture of an individual relative to the current view. Ding et al. [28] present a pose-normalization algorithm using random forest embedded active shape models to map 2D features into a 3D corresponding space. Similarly, Iosifidis et al. [29] applied morphological operations on binary body masks of the torsos of individuals and extrapolated from the ratio of the width and the height of the torso

along with centroid movements the relative posture of the body with respect to the current view. Drawbacks of this approach are the scale and body shape dependency of the torsos as well as expected physical translation across a scene to calculate the posture position. Iosifidis et al.'s [30] later work approached the issues of torso and translation dependencies by creating multiple view binary masks. Bodor et al. [31] used image based rendering to reconstruct views that would be suitable for classifiers. Silhouettes from multiple cameras were captured and projected into a 3D space so that the 3D motion path could be determined. These motion paths were then used to determine the orthogonal views needed for classifiers. However, this system assumes linear motion paths, so activities such as *turning around* and *punching* are not expected to be easily classified.

A view-invariant matching approach depends on finding common features across multiple views. Popular view-invariant feature representations are Self-Similarity Matrices (SSM) [19], which represent distances between action representations, and Cross Ratios (CR) [32] which determine common interest points across multiple action frames. View normalization methods are based on a single transformation for body orientation and view invariance. In comparison, SSM and CR methods, which ignore transformation dependent features, also perform an exhaustive search over all possible transformations to identify matching pairs. These methods are categorized as view-invariant and exhaustive. Holte et al. [33] propose view-invariant 3D feature descriptors based on motion information which are the 3D Motion Context (3D-MC) and the Harmonic Motion Context (HMC). Motion vectors computed from 2D action sequences are extended to 3D flow using pixel to vertex correspondences which are combined to create 3D motion vector fields. A combination of 3D-MC, which is a 3D extension of general shape context, and HMC, which is a spherical representation of weighted sums of spherical harmonics,

are used to provide a view-invariant representation of an action. Normalized correlation coefficients between the test and training action sequences are used to classify actions.

The recent availability of cost-effective depth cameras, such as the Microsoft Kinect sensor, provides a significant advantage, as depth images can facilitate body posture estimation and action classification. Benefits over traditional image sensors include automatic background segmentation, limb identification and invariance to illumination, color, and texture. Shotton et al. [34] used depth data to calculate kinematic joint positions using spatial mode distributions along with randomized decision forests. Their approach is invariant to pose, body shape, and clothing. Similarly Schwarz et al. [35] used depth cameras to identify points on a human with a maximal geodesic distance from the body center of mass, along with optical flow to make predictions on joint tracking while considering occlusions. Beyond kinematic joint tracking, recent research has extended to understanding gestures and actions from depth maps using action graphs [36], statistical analysis on actionlets [37], and Hidden Markov Models [38] [39].

2.2 Face Representations

Face image representations are encodings that describe facial images and, ideally, should be robust enough to distinguish between human subjects. Eigenfaces [40] is an approach based on finding principal components of face images that linearly project the image space to a low dimensional feature space. Although effective under ideal lighting conditions, frontal pose and neutral facial expressions, eigenfaces are not robust and outliers from varying lighting conditions, view angles, and expressions can result in undesired classification errors. Fisherfaces [41] maintain the Euclidean structure while maintaining high between class discrimination and

being less sensitive to lighting and expressions. Laplacianfaces [42] preserve the local structure of the image space and detects the face manifold structure.

A challenge for Eigenfaces, Fisherfaces, and Laplacianfaces is robustness to lighting conditions and facial expressions. Tann and Triggs [43] identify three categories for dealing with these factors which are appearance-based, normalization-based, and feature-based methods. Appearance-based methods require building a large training set that covers varying illumination conditions and expressions. Normalization-based methods involve the normalization techniques such as histograms. This included gamma correction, Difference of Gaussian (DoG) Filtering, and contrast equalization. Figure 2-3 shows eight different subjects under varying illumination conditions and their corresponding normalization in the second row. The illumination invariant approach illustrated was proposed by Tann and Triggs [43] using gamma corrections, DoG filtering, masking, and contrast equalization.

Feature-based methods identify illumination and expression invariant features. One such example is Local Binary Patterns (LBP) which has proven to be effective for texture representations while being highly discriminative and invariant to global gray-level transformations for lighting invariance. LBP is based on thresholding image pixel neighborhoods and encoding a binary pattern. The original LBP method applies an operator on each pixel of an image which thresholds the neighboring pixels at the value of the central pixel. The result of this operator is an image patch with an 8-bit code. An example of a basic LBP operator and its resulting 8-bit code is shown in Figure 2-1. The central pixel with value 77 is analyzed with a 3x3 window. Any neighboring pixel values greater than 77 are assigned a binary value of 1. Any that are less than 77 are assigned a binary value of 0. After applying the LBP operator, the binary encoding is an 8-bit value read from the top left neighbor clockwise

around the central pixel. The encoding is considered uniform if there is at most one transition from 0 to 1 or 1 to 0 (i.e.: 1110001). The resulting encoding in the example provided is not uniform because there are two transitions for 0 to 1 and 1 to 0. The uniform properties of image patches are useful for histograms that identify uniform and non-uniform patterns.

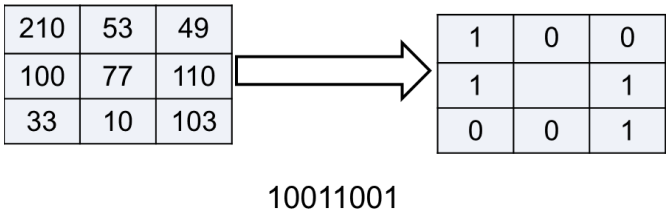


Figure 2-1: The LBP operation and the resulting 8-bit encoding of the central pixel [43].

LBP is popular for face image representations [44] [45] and there are many extensions. Ojala et al. [46] propose a scale and rotation invariant extension of LBP. The work in [47] proposes patch-based descriptors using three-patch and four-patch binary patterns. The main disadvantage of LPB's is the lack of sensitivity to noise. Local Ternary Patterns [43] is an extension of LBP that accounts for robustness to noise and weak illumination gradients by using a three valued code instead of a binary code. Values within a certain tolerance are assigned a value of 0, values above the tolerance are assigned a value of 1, and values below the tolerance are assigned a value of -1. LTP encodings are demonstrated in the Figure 2-2. Figure 2-3 illustrates sample face images from the YALE database which have been illumination normalized. The third and fourth rows show the result LBP and LTP images for those illumination normalized faces.

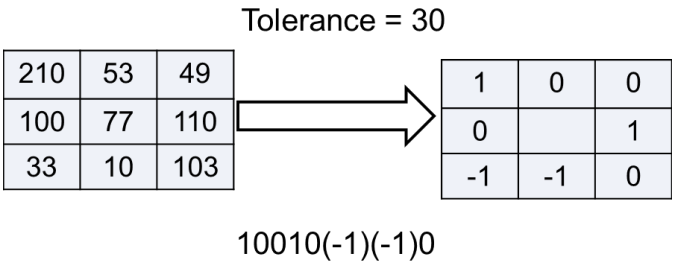


Figure 2-2: The LTP operation and the resulting 8-bit encoding of the central pixel [43].

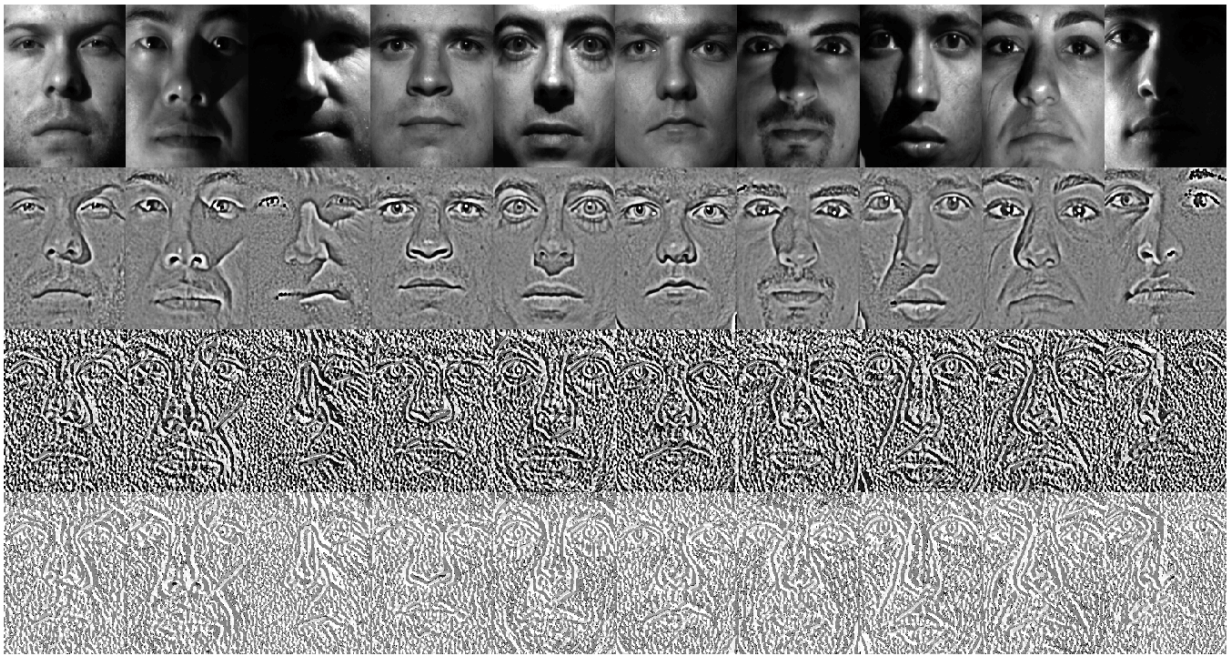


Figure 2-3: Normalization based processing of face images from the YALE face database under different lighting conditions. The top row shows the original images and the second row shows the resulting illumination normalized face images. The third row shows the LBP image representations. The fourth row shows the LTP image representations.

2.3 Radial Distance Representations for Action Recognition

The first contribution in this dissertation is the definition of radial distance surfaces [13] as efficient feature representations. Locality Preserving Projection (LPP), a manifold learning technique, was used for learning low dimensional representations of action primitives to recognize activities across multiple views. To adapt the action classification problem for 3D

depth maps, 3D joint descriptors [48] are also proposed. Radial distances, radial distance surfaces, 3D joint surfaces, and the manifold learning framework are presented in this section.

2.3.1 Radial Distance Measures

Radial distances are features based on distances from a centroid to the outer contour of a silhouette. A manifold learning framework was used for obtaining low dimensional representations of action primitives that can be used to recognize activities across multiple views. For each frame of an entire activity video, silhouettes were represented by binary images after background subtraction. To efficiently describe a silhouette in some detail while maintaining robustness to noise, radial distances were defined from the silhouette centroid to the farthest contour at various angle increments, so that they capture the entire signature over 360 degrees. Radial distances of a silhouette are illustrated in Figure 2-4.

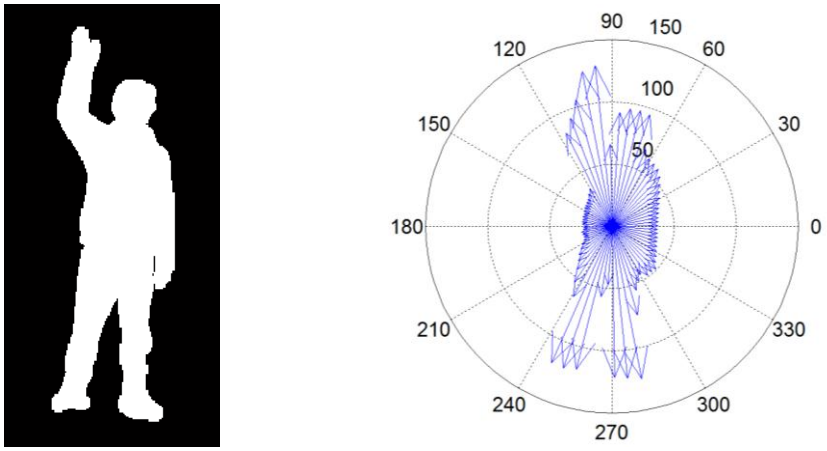


Figure 2-4: An example of (left) a silhouette of a subject performing a *waving* action and (right) the corresponding radial distances from the origin to the contour boundaries over 360 degrees.

Connected components are identified along with their corresponding areas, bounding box regions, and centroids. During the training phase, the largest detected object was cropped and

processed, since it was assumed that there was only one individual conducting an activity at a time and the largest connected component in a frame was that individual. During the testing phase, the system did not make such assumptions and could process multiple individuals in a single scene. By cropping the detected connected region, the region $I(x,y)$ could be processed while preserving the characteristic of scale and localization invariance since the size and location of the silhouette could be ignored.

Once a bounding box was established along with the centroid of the silhouette, the binary silhouette image was converted to a contour plot. The Euclidean distance from the (x,y) centroid to the (x,y) bounds of the contour over 360 degrees in increments of 5 degrees using Equation (1) could then be determined, where $I(x,y)$ is the silhouette image, θ is the angle of the radial distance vector between the centroid and the contour, and r is the radial distance.

$$I(x,y) \rightarrow r(\theta)$$
$$r(\theta) = \sqrt{(x_{centroid-}x(\theta)_{contour})^2 - (y_{centroid-}y(\theta)_{contour})^2} \quad (1)$$

This resulted in 72 radial measures that could be used to form a 2D signal describing the radial distance measures of a silhouettes' contour between 0 and 360 degrees as shown in Figure 2-5b. To further preserve scale invariance the radial magnitude is normalized using Equation (2) and is illustrated in Figure 2-5c.

$$r'(\theta) = \frac{r(\theta)}{\max_\theta(r(\theta))} \quad (2)$$

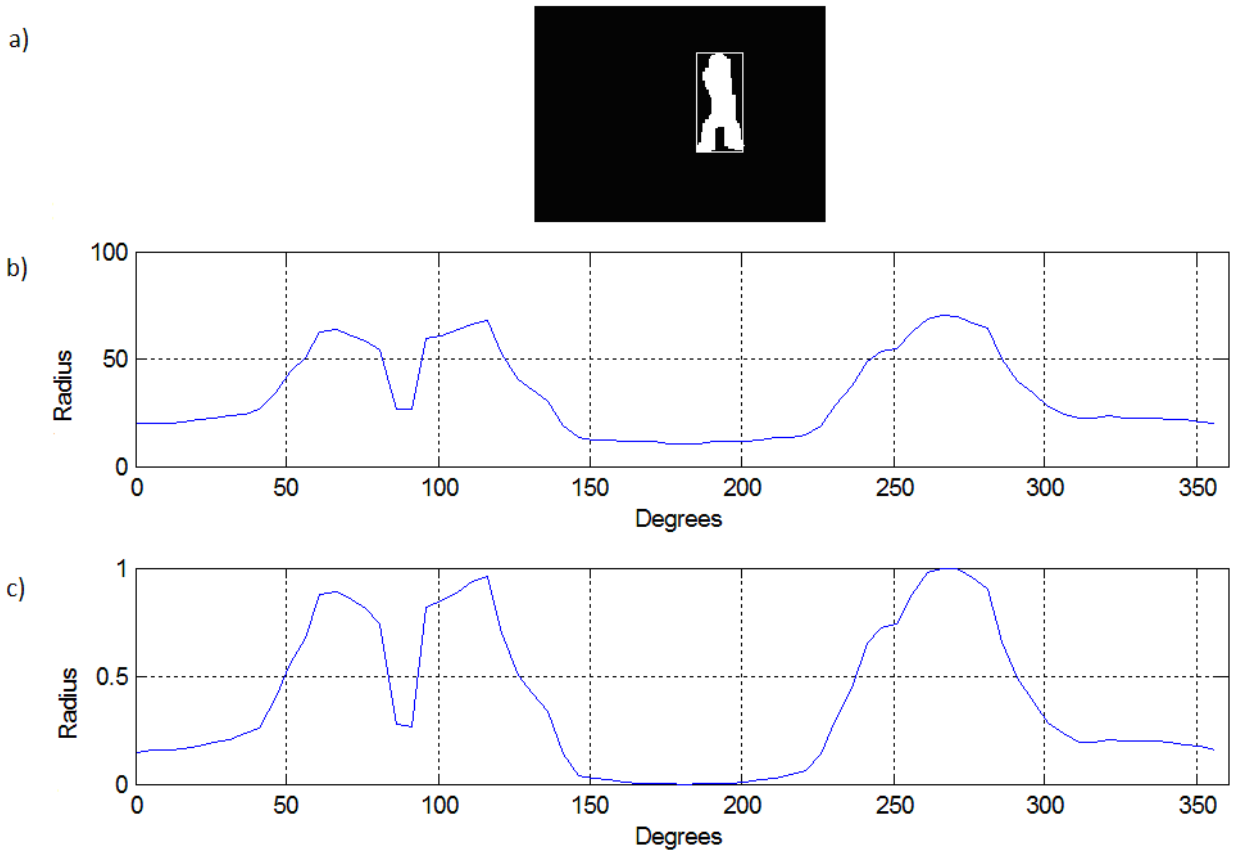


Figure 2-5: An example of (a) a bounding box around a silhouette, (b) the corresponding radial measure plot over 72 evenly distributed angles, and (c) the normalized signal. The two peaks between 50 and 150 degrees represent the outline of the legs of the individuals and the peak at 265 degrees represents the detection of the individuals head.

2.3.2 Radial Distance Surfaces

To formulate a radial distance based spatio-temporal action descriptor, time was added as an additional parameter. The radial distance approach was applied on a single instance of time and combined into a surface. An instance of a cropped region defined by $I(x, y)$ could be defined as a function with a temporal parameter $I(x, y, t)$. In [26], the R-Transform of each frame of an action was combined to form the RXS surface that described the entire activity over time. In our process we followed a similar approach by creating a radial distance surface that could also

describe an activity over time. Equations (1) and (2) were enhanced to include time as a parameter resulting in Equations (3) and (4).

$$I(x, y, t) \rightarrow r_t(\theta)$$

$$r_t(\theta) = \sqrt{\left(x_{t,centroid} - x(\theta)_{t,contour}\right)^2 - \left(y_{t,centroid} - y(\theta)_{t,contour}\right)^2} \tag{3}$$

$$r'_t(\theta) = \frac{r_t(\theta)}{\max_{\theta,t}(r_t(\theta))} \tag{4}$$

By incorporating time, the 2D signals defining an instance in time became a 3D surface defined by radial magnitude, angle, and time as shown in Figure 2-6. As previously mentioned an action is not executed in a fixed amount of time. The same individual *bending down* in one scene might take six seconds in one trial and take ten seconds in another trial. The system must support time invariance and this can be done by normalizing the time axis of the surface.

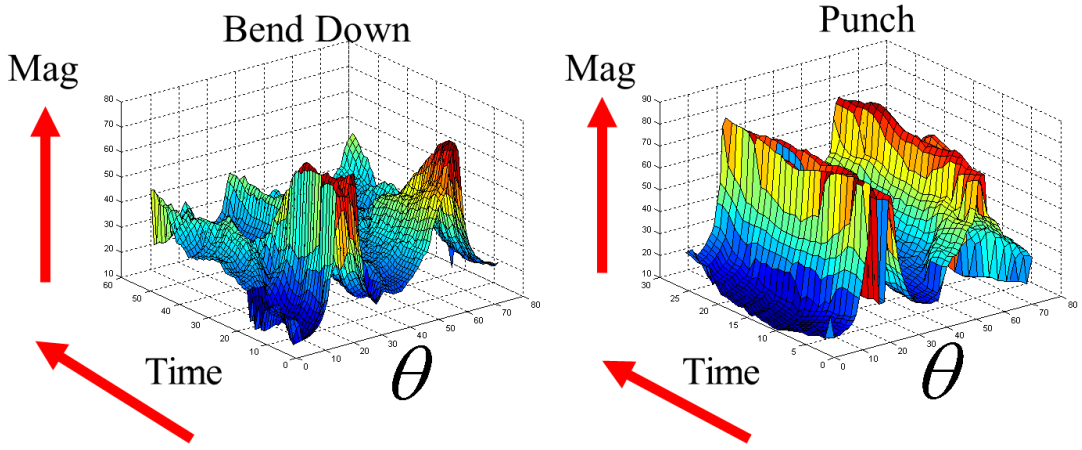


Figure 2-6: An example of a 3-D surface plot defining the punching action from three different camera views from the IXMAS dataset.

Locality Preserving Projections (LPP) is a linear dimensionality reduction algorithm that computes a lower dimensional representation of data from a high dimensional space. It is a linear approximation of the nonlinear Laplacian Eigenmap and is discussed in Section 3.4. In our work [13], LPP was used to evaluate radial distance surfaces on the IXMAS multi-view dataset (Section 7.2.2). In the experimental evaluation, one manifold was used to represent all actions of all views. This only required one transformation to reduce our large input data set using LPP. As a result this form of multi-view training is equivalent to viewpoint independence based on exhaustive searching as discussed by Weinland et al. [1]. The approach requires a training dictionary with enough action representations to represent multiple views to make accurate classification decisions.

Figure 2-7 shows the 3D embedding of high dimensional radial distance surface actions. Ten actions were trained using manifold learning reducing 5,000 dimensions down to only three dimensions for visual illustration. As shown in Figure 2-7, there are clear separations between activities independently of the view in a 3D space. *Point*, *kick*, *Bend* down, and *stand-up* were the most discriminative but the remaining actions, although clustered, do overlap with each other using this framework.

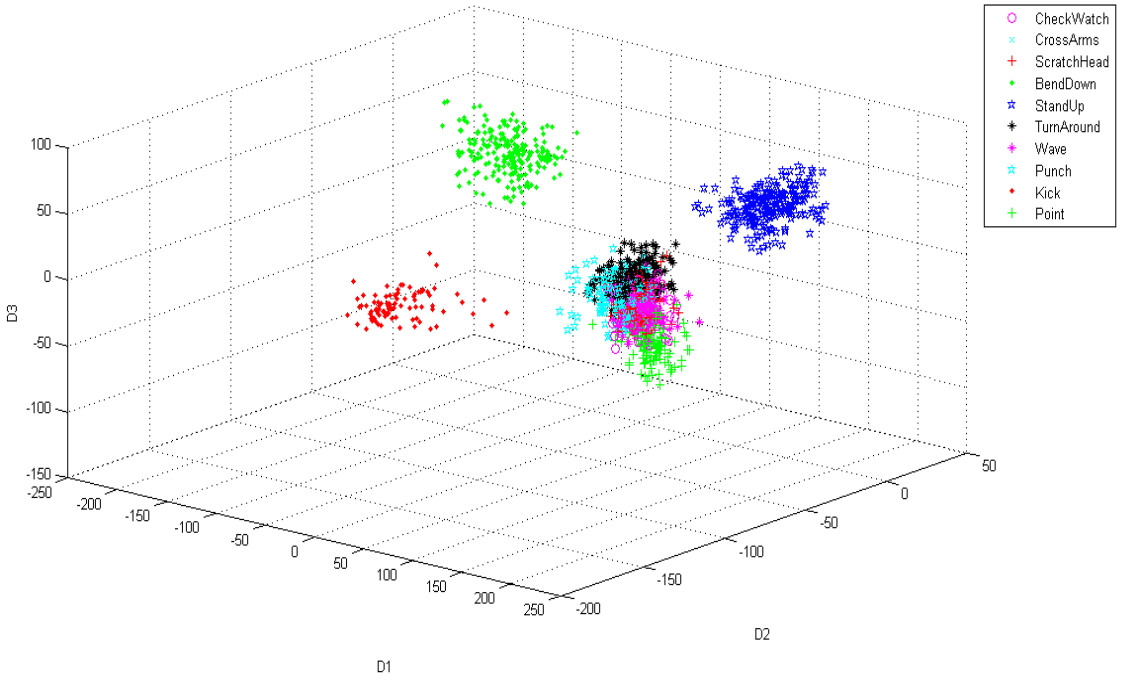


Figure 2-7: The 3D embedding for trained activities from the IXMAS dataset. The actions are *Check Watch*, *Cross Arms*, *Scratch Head*, *Bend Down*, *Stand Up*, *Turn Around*, *Wave*, *Punch*, *Kick*, and *Point*.

	Check Watch	Cross Arms	Scratch Head	Bend Down	Stand Up	Turn	Wave	Punch	Kick	Point
Check Watch	**0.80**		0.20							
Cross Arms		**0.91**				0.09				
Scratch Head			**0.92**				0.08			
Bend Down				**1.00**						
Stand Up					**1.00**					
Turn Around	0.13	0.04	0.04			**0.79**				
Wave							**1.0**			
Punch							0.08	**0.92**		
Kick									**1.0**	
Point							0.10			**0.90**

Table 1: Confusion matrix for 1-NN with a 92.48% average accuracy using LPP and radial distance surfaces on the IXMAS dataset.

Table 1 shows the confusion matrix results after testing new data against the trained data using leave one subject out cross validation. Using the 1-nearest neighbor classifier, the overall

accuracy was 92.48% with *turn around* being the most difficult action to classify. The accuracy with 3 nearest neighbor and 5 nearest neighbor is 93.23% and 93.98% respectively.

Overall the results look promising with the highest recognition rates using the 5-nearest neighbor classifier. The biggest challenge is finding a clear separation between similar activities. For example, *scratch head* and *wave* can be confused because both actions require the act of raising an arm towards the head and, since the viewpoint the action is being captured from is not fixed, there is potential for confusion. The classification of the *turning around* action has a high error rate because the radial distance measure is not effective in capturing useful information of this action over time. With actions such as *punching* and *kicking*, the radial distance surface plot indicates a significant change while the *turning around* surface plot is not as descriptive.

2.3.3 3D Joint Descriptor Surfaces

Radial distance features collect descriptive information for 2D images, but do not take advantage of the information provided by the depth dimension in the 3D depth maps. For example, actions such as a *forward punch* (punch towards the camera) are poorly described by the silhouette, but are described much better with depth data. The 3D joint coordinates, that are available through the Microsoft Kinect interface software, were selected to capture the depth dimension. The 3D joint coordinates were calculated using the approach proposed by Shotton et al. [34], where 3D positions of body joints are predicted from a single depth camera using randomized decision forest classifiers for body part labeling. Specifically, mean-shift is used to classify each pixel in an image using spatial mode distribution along with the randomized decision forests to propose 3D joint positions. The approach is invariant to pose, body shape, and clothing.

The Microsoft Research 3D Dataset (MSRAction3D) (Section 7.2.4) includes the 3D joint data comprised of 20 coordinates of joint positions in a frame along with their corresponding depth value and confidence level. The joint positions include the locations of hands, wrists, elbows, shoulders, the head, the shoulder center, the spine, the hip center, the hips, the knees, the ankles, and feet. These kinematic coordinates are captured into a feature vector after the joint coordinates are subtracted from the center torso of the human to define relative data and account for localization invariance. The difference between the coordinates and the torso coordinates are then normalized to define features which are scale invariant.

Figure 2-8 shows examples of joint positions on sample frames of a subject performing a *tennis swing* action. The 2D coordinates of the joints are normalized individually from the depth values and the coordinates and the depth data are vectorized into a 1D feature vector of 60 features (20 x-values, 20 y-values, and 20 depth values).

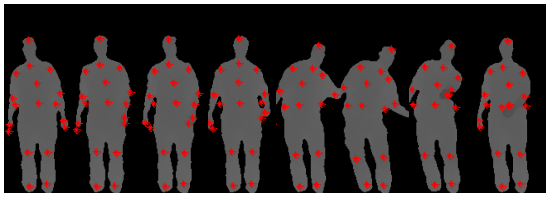


Figure 2-8: Video depth sequences from the MSRAction3D dataset with 3D joint tracking on a subject executing a *tennis swing* action.

The 3D joint descriptors only represent spatial structures of an instance of time of human pose. Temporal structuring is necessary to capture the description of an entire action, but it is known that actions can vary in execution time. To account for time variations we created surface plots from the feature descriptors which capture the entire action and normalize the surface descriptor in the time domain. This creates surface descriptors that are invariant to activity

execution time. Our approach is presented in [48] and is inspired from our earlier work in [13]. Figure 2-9 presents an example of a 3D joint tracking surface representing a *tennis swing* action.

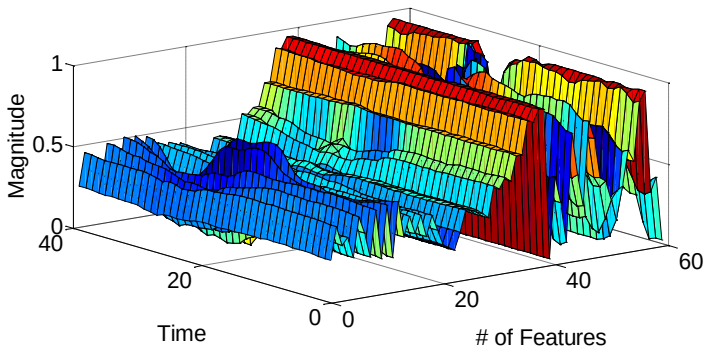


Figure 2-9: The normalized 3D joint tracking surface for a *tennis swing* action from the MSRActrion3D dataset.

2.3.4 Radial Distance Surfaces and 3D Joint Surface Descriptor Evaluation

In [48], an evaluation was made using radial distance surfaces, 3D joint surfaces, and a combined larger representation of both descriptors as one representation. LPP was used as the manifold learning method with a nearest neighbor classifier on the MSRAction3D dataset consisting of depth map sequences. There are ten subjects of varying shapes and sizes performing twenty actions two to three times at various speeds. The dataset actions are listed in Table 2 which is organized in the same experimental setup as [36]. The 20 actions were divided into three subsets consisting of 8 actions each. Additionally, we also tested against the entire set of activities (subset 4). The subsets 1 and 2 were designed to group activities with similar movements while the third subset was designed to group actions that are more likely to be error prone due to their similarities. In our experiments, we considered cross validation through random selection and training and testing with half of the data samples as well as leave one subject out training.

Subset 1	Subset 2	Subset 3	Subset 4
Hor. Arm Wave Hammer Forward Punch High Throw Hand Clap Bend Tennis Serve Pickup & Throw	High Arm Wave Hand Catch Draw X Draw Tick Draw Circle Two Hand Wave Forward Kick Side Boxing	High Throw Forward Kick Side Kick Jogging Tennis Swing Tennis Serve Golf Swing Pickup & Throw	All Actions

Table 2: The MSRAction3D action subsets used for action classification experiments.

	Radial Distance	3D Joint Tracking	Radial Distance & 3D Joint Tracking
	Cross Validation		
Subset 1	78.31%	84.34%	**87.95%**
Subset 2	74.47%	77.66%	**78.72%**
Subset 3	91.58%	98.95%	**98.95%**
Subset 4	65.09%	**73.71%**	73.28%
	Leave One Subject Out		
Subset 1	89.01%	77.65%	**92.34%**
Subset 2	73.73%	74.45%	**80.01%**
Subset 3	85.05%	91.70%	**92.98%**
Subset 4	67.00%	76.14%	**76.49%**

Table 3: Action classification accuracy on the MSRAction3D dataset. Cross validation and leave one subject out testing were used with radial distance measures, 3D joint tracking, and a combined descriptor.

As presented in Table 3, the combination of radial distance surfaces with 3D joint tracking meets or exceeds the classification accuracy of either radial distances or 3D joint tracking independently. In subset 1 where actions were grouped because of their similarities, we achieve 92.34% accuracy using leave one subject out which indicates that the approach is strongly invariant to individual size, shape, location in a scene, and action execution time which is what our approach was intended to address. Furthermore, our approach performs extremely well on

subset 3 which was intended to evaluate similar activities. This demonstrates that manifold learning on descriptor surfaces are strong in classifying similar activities and are therefore highly discriminative.

Through cross validation the most problematic action to classify for 3D joint tracking was *draw X* which frequently got confused with *horizontal arm wave*. For radial distances *forward punch* was frequently confused with *horizontal arm wave* which is understandable since the radial distances are similar between these depth related actions. The combined descriptor faces challenges distinguishing between *hammer* and *tennis serve* as well as between *draw X* and *horizontal arm wave*. When our training set became larger using the leave one subject out approach the most challenging action to classify was *draw tick* which frequently got confused with *hammer* and *forward punch*.

2.4 Motion Images as Action Descriptors

The next contribution is the formulation of spatio-temporal motion surfaces that can be adapted for multi-view and 3D action classification applications. To avoid the complexity involved with body part labeling and tracking, motion images are utilized as temporal templates. The advantages of motion images include simple representations that provide good performance, ability to represent the direction of motion in a scene, and ability to identify where motion exists in a scene. Motion images are extended to represent 3D motion for 3D action classification. These feature representations can be defined into a spatio-temporal descriptor through surfaces similar to radial distance surfaces. This section presents motion images, motion history surfaces, and motion depth surfaces.

2.4.1 Motion Energy Images and Motion History Images

Motion history images are the primary spatial parametric features used in this dissertation for action classification systems. Proposed by Davis and Bobick [49], Motion History Images (MHI's) are temporal templates that are capable of describing where motion exists in a scene and how the motion is evolving over time. The MHI features are based on Motion Energy Images (MEI's) which offer a binary representation of where motion occurs in a scene. It is an indicator of motion over time. Given a video frame $I(x, y, t)$, calculate a binary image $D(x, y, t)$ as the difference image between $I(x, y, t)$ and $I(x, y, t \pm \Delta)$ where Δ is a time offset. The binary MEI $E_\tau(x, y, t)$ is defined as:

$$E_\tau(x, y, t) = \bigcup_{i=0}^{\tau-1} D\,(x, y, t - i) \tag{5}$$

where τ is the temporal extent of the action. This equation captures motion across τ. An example of MEI's is shown in the second row of Figure 2-10.

MHI's capture how motion changes over time in addition to where motion changes over time. The MHI descriptor $H_\tau(x, y, t)$ is defined as:

$$H_\tau(x, y, t) = \begin{cases} \tau \;\; if\; \psi(x, y, t) = 1 \\ \max(0, H_\tau(x, y, t-1) - \alpha) \; o.w. \end{cases} \tag{6}$$

where τ describes the initial motion response, the decay operator is regulated by α, and $\psi(x, y, t)$ is an update function. There are many variants of update functions [8] including background subtraction, image differencing, and optical flow. Sample motion history images in Figure 2-10 are shown using a background subtraction update function. The MHI shows more recent motion appearing brighter than older motion. A main advantage of MHI is that the results represent the direction of motion.

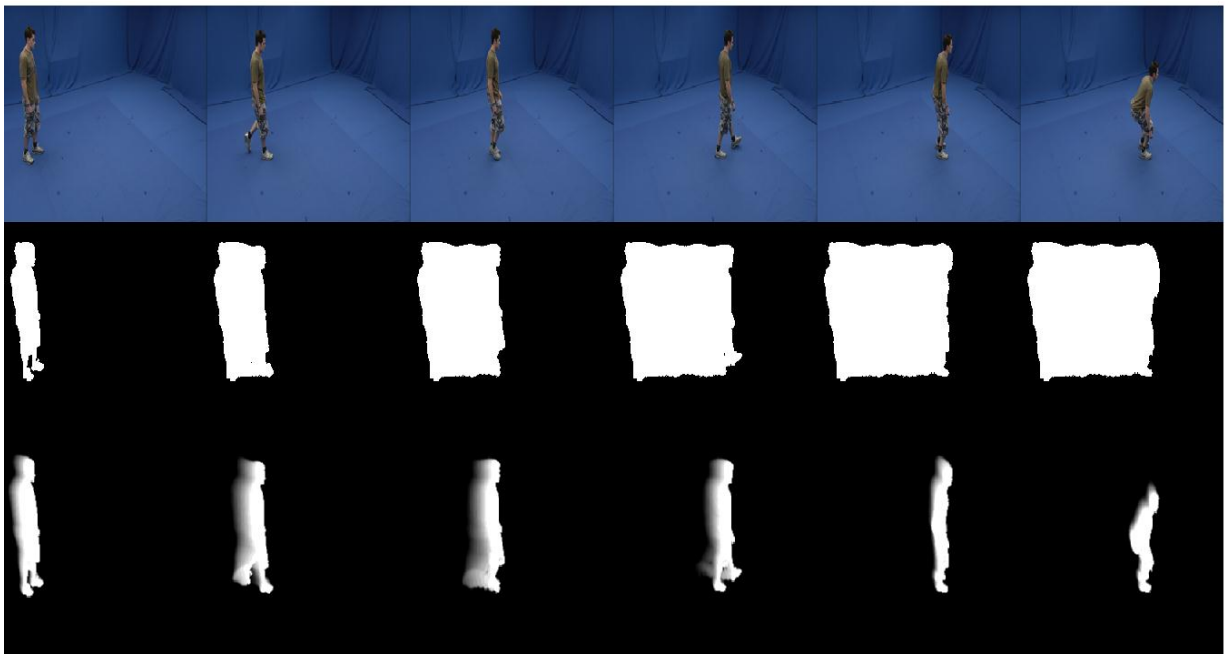


Figure 2-10: A subject from the i3DPost Multi-View dataset walking across a scene and then sitting. The second row shows the corresponding Motion Energy Images and the third row shows the corresponding Motion History Images with $\tau = 7$.

2.4.2 Motion History Surfaces

The MHI descriptor is useful in identifying spatial and temporal structuring of actions, however, the MHI representations in their current form do not easily allow for comparisons between various actions. Actions vary in terms of the time of execution making it difficult to formulate an action classification method. Furthermore, human subjects executing such actions can vary in size and their style in performing actions. It is desired to formulate an action template as one large representation of an action of a fixed size that can be invariant to scale, position in a scene, and action execution time.

To do so, spatio-temporal action surfaces are composed from MHI primitives that can account for these factors. Regions of interest (ROI) of a scene are identified where the motion occurs eliminating the issue of localization. To preserve invariance to human sizes each ROI is

resized using bicubic interpolation. Figure 2-11 demonstrates an example frame of a subject walking and the resulting fixed size representation of that frame.

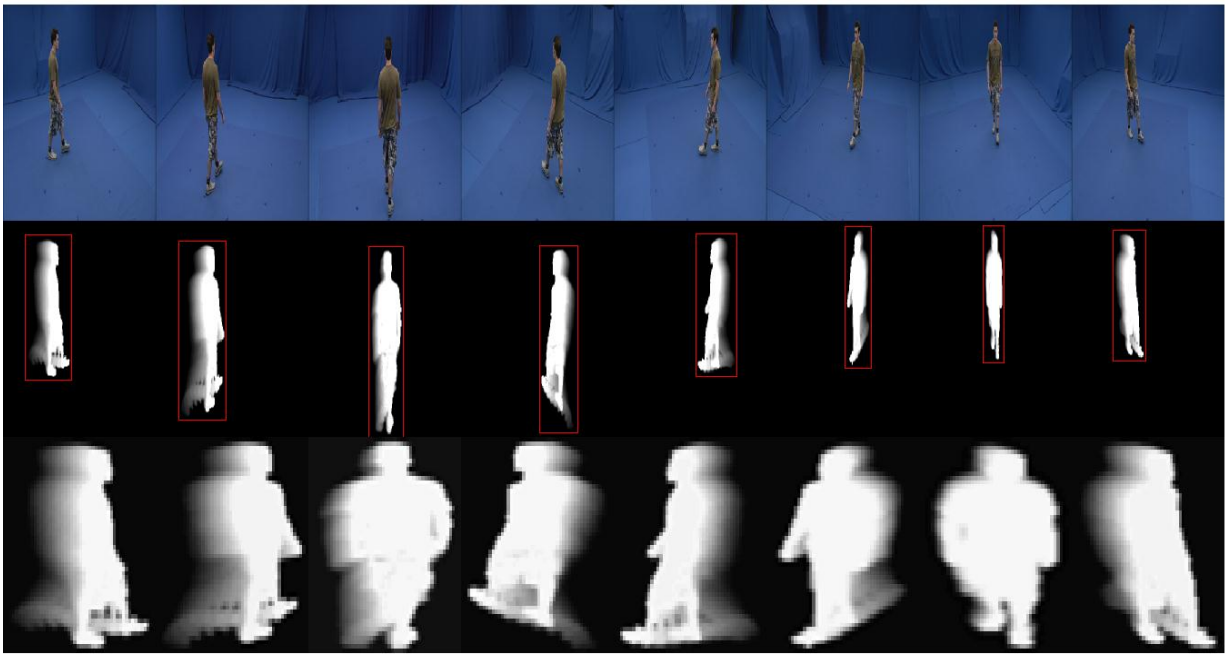


Figure 2-11: An instance of time of an i3DPost multi-view scene of an individual walking in MHI form. The top row shows the original frame. The second row shows the bounding boxes around the region of interest. The bottom row shows a fixed size representation of that same subject.

These fixed size action primitives offer spatial representations but do not identify any temporal structuring beyond the MHI representation of each action primitive at one instance of time. To formulate spatio-temporal action templates, we collect entire action sequences and concatenate the MHI descriptors to form motion history surfaces. In this formulation, the motion history surfaces become spatio-temporal action templates. These surfaces are normalized using Equation (7) to encourage minimum scale variations while preserving relative frame information. Action surfaces can vary due to the execution time of an action by an individual. To account for time invariance, these surfaces are resized using bicubic interpolation. Azary and Savakis propose motion history surfaces in [50] for multi-view action classification systems. Radial distance surfaces combined with skeletal tracking are considered for 3D action classification

systems in [48]. Spatio-temporal action surfaces for an individual walking across multiple views are shown in Figure 2-12.

$$H_\tau{}' = \frac{H_\tau}{max_{x,y,t}(H_\tau)} \tag{7}$$

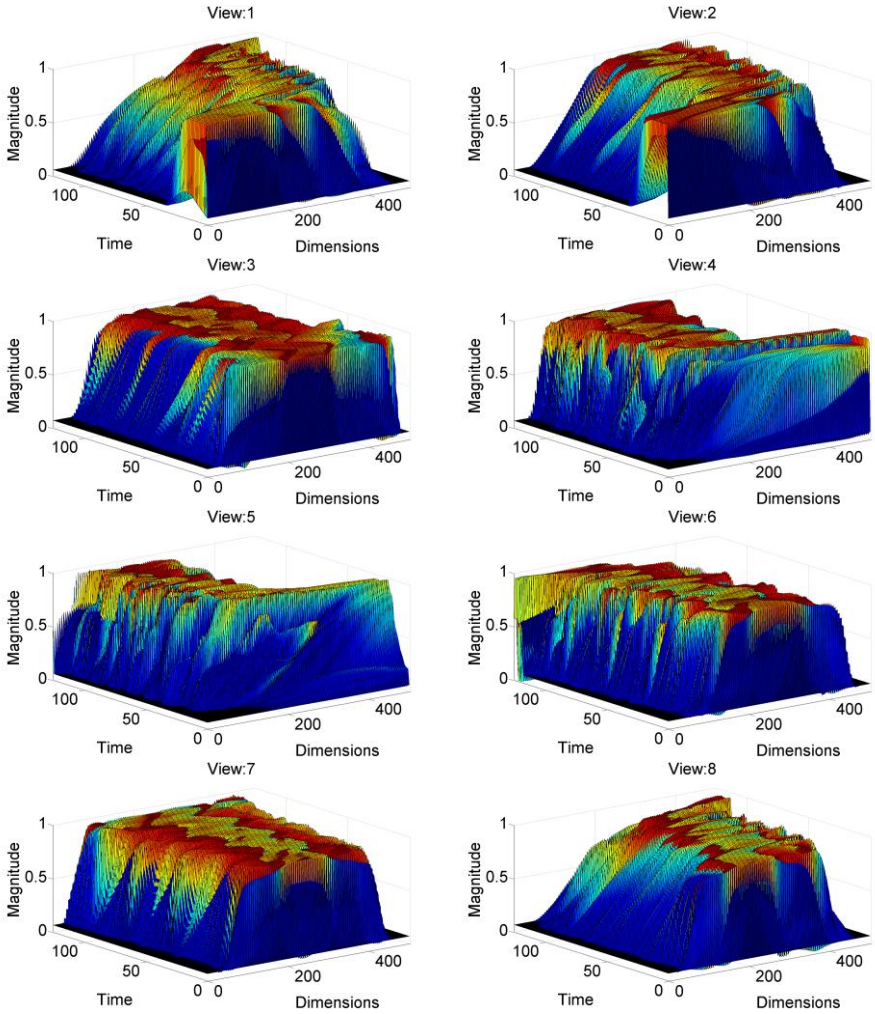


Figure 2-12: Spatio-temporal motion history surfaces for eight views of an individual walking from the i3DPost dataset.

2.4.3 Motion Depth Surfaces

For 3D video sequences, we use Motion Depth Surfaces (MDS's) by incorporating the additional dimension of depth. Assuming $I(x, y, t)$ represents a depth value at pixel (x, y) for time t, we define a motion depth image (MDI) as follows:

$$MDI_\tau(x, y, t) = \begin{cases} I(x, y) \ \ if \ D(x, y, t) = 1 \\ \max(0, MDI_\tau(x, y, t-1) - \alpha) \end{cases} \quad (8)$$

This formulation permits us to capture motion activity in the depth direction as well as within a frame. We concatenate each MDI to create a motion depth surface (MDS) that represents spatio-temporal motion with built-in depth motion. As was done with MHS, these surfaces were scaled to a fixed size to account for variations in the timing of actions and to ensure that the number of dimensions of each action descriptor remains consistent and its size is manageable.

Examples of subjects executing a *horizontal arm wave* and a *forward punch* from the MSRAction3D dataset shows how the direction of depth is incorporated into the MDS descriptor as shown in Figure 2-13. Similarly, Figure 2-14 shows a comparison of an MHS and an MDS description of the ASL gesture for *Green* from the MSRGesture3D dataset.

Horizontal Arm Wave

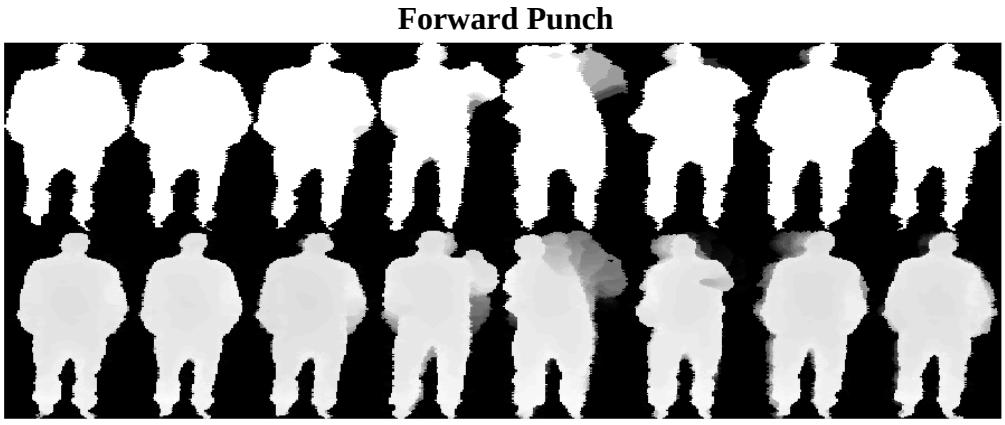
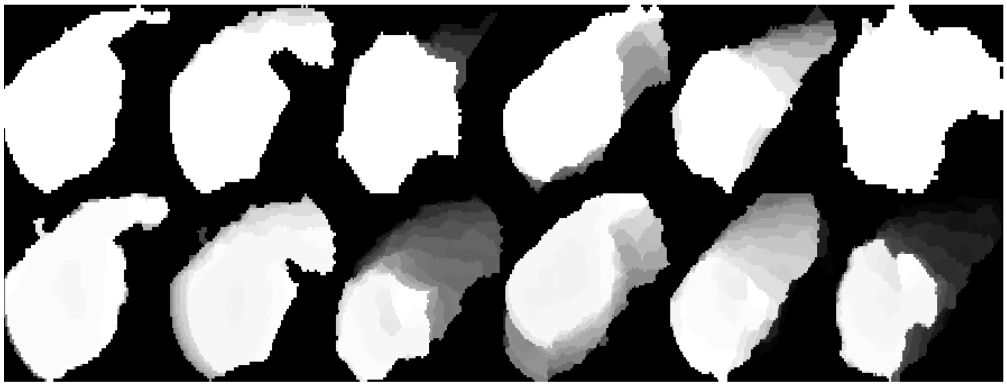


Figure 2-13: A comparison of MHI (top rows) with MDI (bottom rows) for subjects performing a horizontal arm wave action and a forward punch from the MSRAction3D dataset.

a)



b)

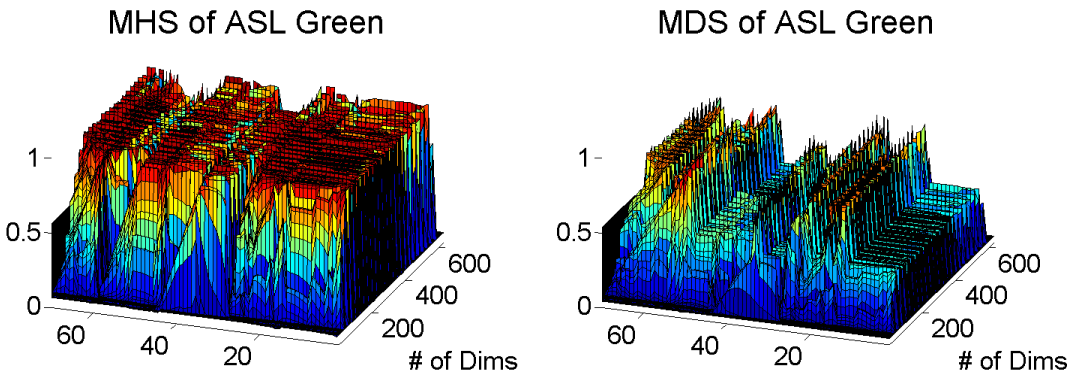


Figure 2-14: (a) Sample frames of the ASL sign for *Green* from the MSRGesture3D dataset. The top row frames show MHI's and the bottom row frames show MDI's. (b) The corresponding MHS and MDS.

3 Dimensionality Reduction Methodologies

The high dimensional data that represent an action or a face can become overwhelming when dealing with a large number of data samples. A common challenge for real-world classification and recognition systems is the computational performance associated with processing high-dimensional data. Subspace learning dimensionality reduction addresses the issue of high dimensional data by finding an efficient low-dimensional representation. The following sections focus on dimensionality reduction methods including principal component analysis (PCA), metric multidimensional scaling (MDS), local linear embedding (LLE), and linear extensions of graph embedding (LGE).

3.1 Principal Component Analysis

Principle Component Analysis (PCA) is a widely used methodology for reducing the dimensions of complex and/or noisy data sets to extract relevant information that can be beneficial in describing the data. It is a linear technique that projects data along the directions of maximal variance. PCA has been employed for action classification systems in several works including [51] and [52]. In this section, PCA is overviewed, including its benefits and limitations, as outlined in [53] and [54]. For data sets with large number of samples n the information can be computationally expensive to process. PCA aims to reduce noise and redundancy while preserving the global structure of the high dimensional data [55] by preserving the maximal variance. Given n-samples of data $\boldsymbol{X}$, each of m-dimensions, PCA provides a way to calculate a lower dimensional representation $\boldsymbol{Y}$ of the higher dimensional data through a transformation $\boldsymbol{Y' = P'X}$. To solve for this transformation, PCA calculates a square covariance matrix. PCA solves for the principal components of all samples by calculating the eigenvectors of the

covariance matrix to identify principal components of maximal variance. An alternative approach to finding the eigenvectors involves Singular Value Decomposition (SVD).

PCA is a linear method for extracting linear features based on maximal variance. When the data set is a representation of non-linear features, the principal components may not be effective in simplifying the data set successfully. For an illustrative example, four 3D shapes are presented in Figure 3-1. The first row shows the original 3D representations: swiss roll, Gaussian, twin peaks, and intersect. The colors identify classes associated with each sample. The 2D representations after applying principal component analysis are shown in the second row. PCA performs well in representing the swiss roll and Gaussian surfaces, but the other more complex shapes do not show a consistent pattern of within class clustering.

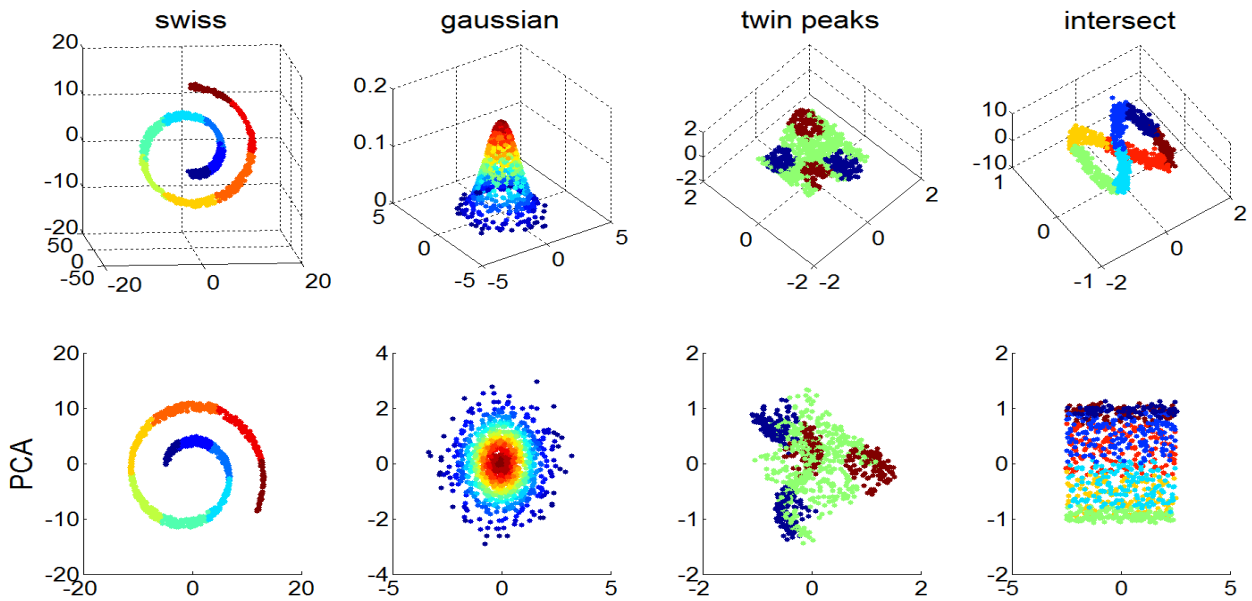


Figure 3-1: PCA dimensionality reduction examples including the swiss roll, Gaussian, twin peak, and intersection.

Given n as the number of data samples and p as the number of classes, the covariance matrix computation of PCA has a time complexity of $O(p^2 n)$. The eigenvalue decomposition has a time complexity of $O(p^3)$. Therefore, PCA has a time complexity of $O(p^2 n + p^3)$ [56].

3.2 Metric Multidimensional Scaling

Metric Multidimensional Scaling (MDS) is a linear technique for dimensionality reduction based on proximity data analysis. MDS attempts to define a distance measure between data in the high dimensional space that would be preserved in a lower dimensional space and is a good identifier of clustering patterns. MDS has been used for a wide variety of applications including stock market analysis [57], wireless sensor network localizations [58], and protein binding predictions [59].

Given a data set $\boldsymbol{X} = \{\boldsymbol{X_1}, \boldsymbol{X_2}, \dots, \boldsymbol{X_n}\}$ for which each element in the data set resides in a high dimensional space B such that $\boldsymbol{X}_i \in \mathbb{R}^B$, MDS will solve for a lower dimensional representation set $\boldsymbol{Y} = \{\boldsymbol{Y}_1, \boldsymbol{Y}_2, \dots, \boldsymbol{Y}_n\}$ in space b such that $\boldsymbol{Y}_i \in \mathbb{R}^b$ and $b \ll B$. This mapping is approximated by the distances between samples following $\|\boldsymbol{X}_i - \boldsymbol{X}_j\|$ [60]. A square dissimilarity matrix is created which measures the distance between each pair of elements in the high dimensional space as demonstrated in Equation (9) with $D_{ii} = 0$ and $D_{ij} > 0$.

$$\boldsymbol{D} = \begin{bmatrix} D_{11} & \cdots & D_{1j} \\ \vdots & \ddots & \vdots \\ D_{i1} & \cdots & D_{ij} \end{bmatrix} \tag{9}$$

The Minkowski distance metric [61] shown in Equation (10) is a general distance measure between elements where n is the number of data samples. This equation is transformed to the City-Block metric [62] and the Euclidean distance metric when $r = 1$ and $r = 2$ respectively.

Such distance measures can be used as proximity measures in the high dimensional space depending on the application.

$$D_{ij} = \left[\sum_{k=1}^{n} \left|\boldsymbol{X}_{ik} - \boldsymbol{X}_{jk}\right|^{r}\right]^{\frac{1}{r}} \tag{10}$$

Given the dissimilarity matrix $\boldsymbol{D}$, the MDS problem becomes a minimization problem for which we desire a transformation that will minimize the error of the distances in a lower dimensional space. To do this we use the following stress function as a least squares criterion. The stress function $S_D(\boldsymbol{X}_1, \boldsymbol{X}_2, \dots, \boldsymbol{X}_n)$ in Equation (11) measures the deviation between the distance D_{ij} and the target distance $\left\|\boldsymbol{X}_i - \boldsymbol{X}_j\right\|$.

$$S_D(\boldsymbol{X}_1, \boldsymbol{X}_2, \dots, \boldsymbol{X}_n) = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n}\left(D_{ij} - \left\|\boldsymbol{X}_i - \boldsymbol{X}_j\right\|\right)^2} \tag{11}$$

Equation (12) shows the minimization function that minimizes the stress over all points while finding the transformation that will reduce the number of dimension from B to b such that $b \ll B$ [63].

$$\min_{Y} \sum_{i=1}^{n}\sum_{j=1}^{n}\left(\left\|\boldsymbol{X}_i - \boldsymbol{X}_j\right\|^2 - \left\|\boldsymbol{Y}_i - \boldsymbol{Y}_j\right\|^2\right)^2$$
$$\min_{Y} \sum_{i=1}^{n}\sum_{j=1}^{n}\left({D_{ij}}^{X} - {D_{ij}}^{Y}\right)^2 \tag{12}$$

The method of minimization with metric multidimensional scaling is an eigenvalue problem. The distance matrix $\boldsymbol{D}^X$ is converted to a matrix of inner products $\boldsymbol{X}'\boldsymbol{X}$ which reduces Equation (12) to Equation (13).

$$\min_{Y} \sum_{i=1}^{n}\sum_{j=1}^{n}\left(\boldsymbol{X}_i'\boldsymbol{X}_j - \boldsymbol{Y}_i'\boldsymbol{Y}_j\right)^2 \tag{13}$$

The eigenvectors $\boldsymbol{V}$ of $\boldsymbol{X}'\boldsymbol{X}$ are used to solve for the top m eigenvalues, $\boldsymbol{\lambda}$. The coordinates are transformed from high dimensional space B with $\boldsymbol{X}_i \in \mathbb{R}^B$ to lower dimensional space b with $\boldsymbol{Y}_i \in \mathbb{R}^b$ following Equation (14). Figure 3-2 shows the 2D representations of 3D shapes that were reduced through MDS using a Euclidean distance metric ($r = 2$).

$$\boldsymbol{Y} = \boldsymbol{\lambda}^{1/2}\boldsymbol{V}' \tag{14}$$

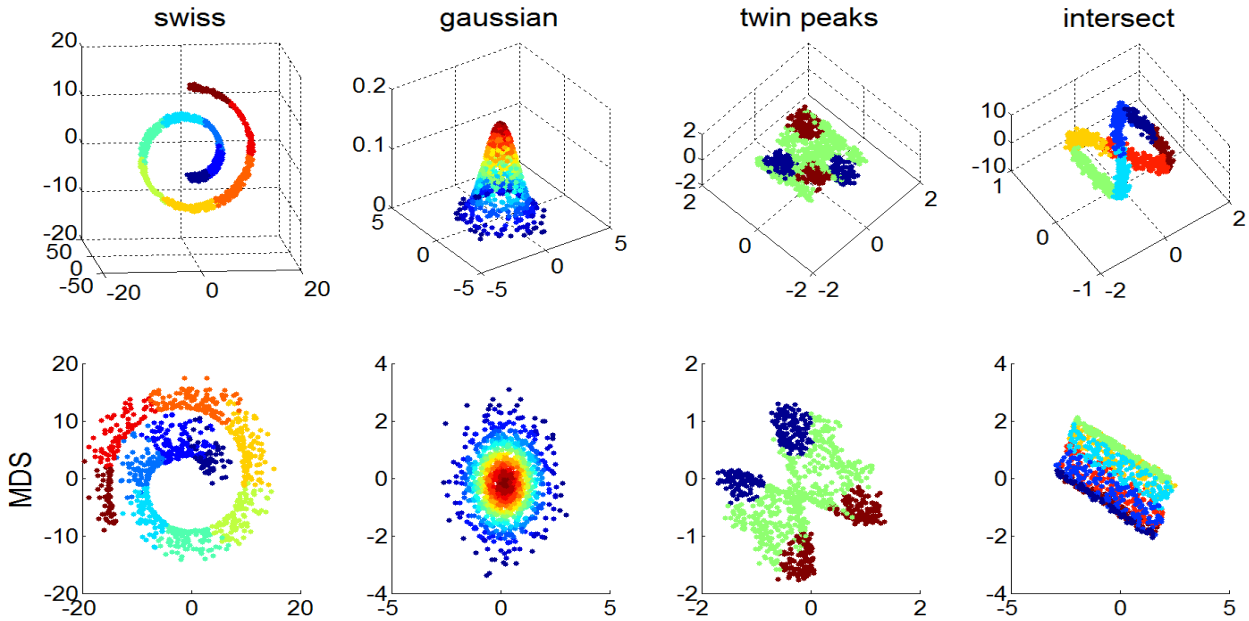


Figure 3-2: MDS dimensionality reduction examples including the swiss roll, Gaussian, twin peak, and intersection.

The lower dimensional representation of the training data only represents the original high dimensional training data and it is unclear how to map new testing data samples. For this reason, MDS is ideal for proximity and cluster analysis, but is not ideal for systems requiring the classification of new data samples. The computational complexity of MDS is $O(n^3)$ [64].

3.3 Locally Linear Embedding

Locally Linear Embedding (LLE) is an unsupervised eigenvector method for dimensionality reduction that preserves the embedding of high dimensional data through maximal discrimination in the lower dimensional space. The result is a preservation of the underlying

structure of the manifold. LLE has been used for a wide variety of applications including the mapping of DNA gene expressions [65] and super resolution [66].

Given a data set $\boldsymbol{X} = \{\boldsymbol{X}_1, \boldsymbol{X}_2, \dots, \boldsymbol{X}_n\}$ for which each element in the data set resides in a high dimensional space D such that $\boldsymbol{X}_i \in \mathbb{R}^D$, LLE maps $\boldsymbol{X}$ to a lower dimensional representation new data set $\boldsymbol{Y} = \{\boldsymbol{Y}_1, \boldsymbol{Y}_2, \dots, \boldsymbol{Y}_n\}$ for which each element in $\boldsymbol{Y}$ resides in a lower dimensional space d such that $\boldsymbol{Y}_i \in \mathbb{R}^d$ and $d \ll D$. LLE uses multiple stages for this mapping. First, it computes the nearest neighbors of each data point $\boldsymbol{X}_i$. Then, it constructs a weight matrix $\boldsymbol{W}_{ij}$ between all data points $\boldsymbol{X}_i$ that represent the local linear geometry. Weights are assigned a value of zero for the pairs that are not considered nearest neighbors. Nearest neighbor weights are computed in a manner that can best reconstruct each data point from its neighbors in the lower dimensional space. This is accomplished by establishing measurement of reconstruction errors based on the cost function of Equation (15). This cost function identifies how well each $\boldsymbol{X}_i$ can be linearly constructed from its nearest neighbors $\boldsymbol{X}_{N(1)} \dots \boldsymbol{X}_{N(k)}$ [67].

$$\varepsilon(W) = \sum_{i=1}^{n} \left| \boldsymbol{X}_i - \sum_{j=1}^{k} \boldsymbol{W}_j{}^{(i)} \boldsymbol{X}_{N(j)} \right|^2 \tag{15}$$

This cost function is designed to ensure invariance to rotation and scale [68]. The constraint $\sum_{j=1}^{k} \boldsymbol{W}_j{}^{(i)} = 1$ ensures the sum of the weights between $\boldsymbol{X}_i$ and all selected neighbors will sum to 1 and be invariant to translation. The cost function can then be treated as a constrained least squares problem to solve for the optimal weights. These weights represent the local linear geometry of the patches since they were determined by assigning weights of the nearest neighbors of $\boldsymbol{X}_i$. Given the optimal weights, the final step of the LLE algorithm is to compute the lower dimensional neighborhood-preserving mapping $\boldsymbol{Y}$ based on the selected weights using the following cost function.

$$\Phi(Y) = \sum_{i=1}^{n} \left| \boldsymbol{Y}_i - \sum_{j=1}^{k} W_j^{(i)} \boldsymbol{Y}_{N(j)} \right|^2 \tag{16}$$

Constraining $\sum_i \boldsymbol{Y}_i = 0$ and $1/N \sum_i \boldsymbol{Y}_i' \boldsymbol{Y}_i = I$ results in the following cost function.

$$\Phi(Y) = \sum_{i=1}^{n} |(\boldsymbol{I} - \boldsymbol{W})\boldsymbol{Y}_i|^2 = tr(\boldsymbol{Y}'\boldsymbol{M}\boldsymbol{Y}) \tag{17}$$

where $\boldsymbol{M} \in R^{N \times N}$ and $\boldsymbol{M} = (\boldsymbol{I} - \boldsymbol{W})'(\boldsymbol{I} - \boldsymbol{W})$. The final step of LLE is to compute the bottom non-zero eigenvalues of matrix $\boldsymbol{M}$.

Similar to MDS, the lower dimensional representation of the training data only represents the original high dimensional training data and it is unclear how to map new testing data samples. For this reason, LLE is ideal for preserving the embedding of high dimensional data through maximal discrimination and analyzing clustering patterns, but not ideal as a method for learning and classifying new test data. Figure 3-3 illustrates the 2D mappings of 3D shapes. Notice how the shapes are clustered in patches. The time complexity of LLE is a sum of searching the nearest neighbors $O(Dn^3)$, computing the reconstruction weights $O(Dnk^3)$, and computing the eigenvalues $O(kDn^3)$ [69], where n is the number of data samples, D is the number of dimensions in the high dimensional space, and k is the number of nearest neighbors.

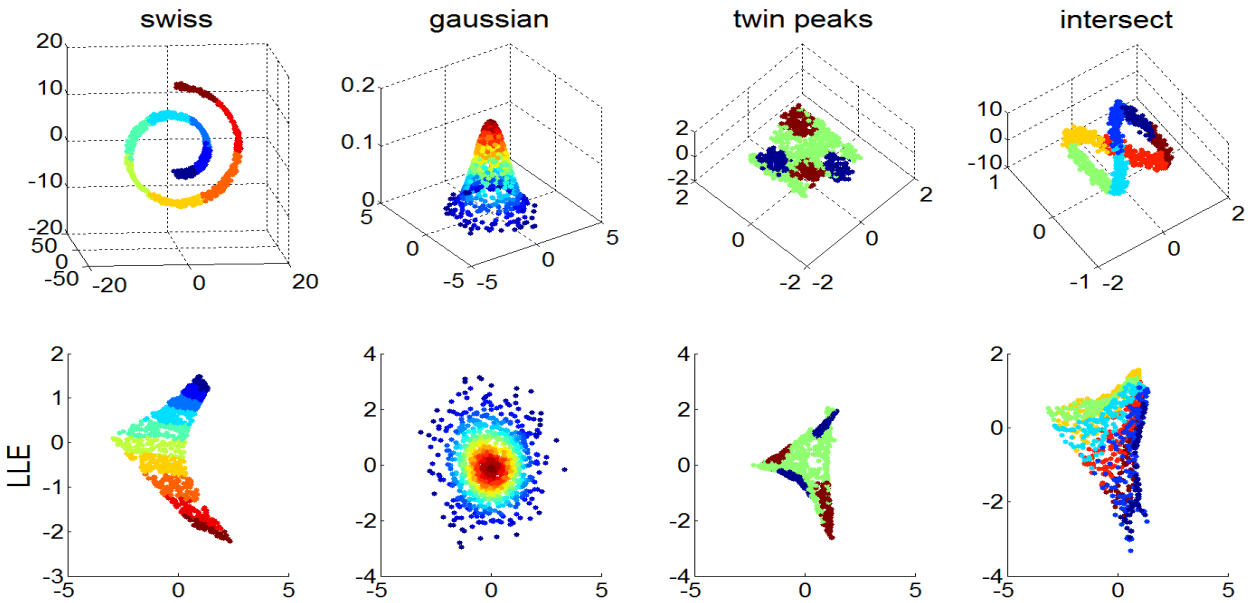


Figure 3-3: LLE dimensionality reduction examples including the swiss roll, Gaussian, twin peak, and intersection.

3.4 Linear Extensions of Graph Embedding

Linear Extensions of Graph Embedding (LGE) methods are eigen-based linearized techniques to solve linear approximations of non-linear systems through dimensionality reduction. They include linear discriminant analysis (LDA) [70], locality preserving projections (LPP) [71], and neighborhood preserving embedding (NPE) [72]. Manifold learning by Locality Preserving Projections (LPP) preserves local neighborhood information and was first reported for action classification systems in the work of Wang and Suter [73]. Linear Discriminant Analysis (LDA) was applied to action classification in [74].

The LGE family of linear dimensionality reduction algorithms computes a lower dimensional representation of data from a high dimensional space, while preserving the local structure of the input data [55]. LGE solves transformations from a high dimensional space to a lower dimensional space which preserve local neighborhood information. LPP is a linear approximation of the nonlinear Laplacian Eigenmap [71] [75] and a generalization of LDA. An

action represented in a lower dimensional space is spatially close to other actions in the same manner as in the higher dimensional space.

Nonlinear methods such as LLE, Isomap, and Laplacian Eigenmaps reveal the relationship of training data samples along a manifold by learning the global structure of such manifolds and finding mutual relationships among the training data samples [76]. However, since these methods model data with nonlinear approaches, the lower dimensional representation of the training data only represents the original high dimensional training data and it is unclear how to map new testing data samples, as explained by He and Niyogi [71]. LGE methods are linear algorithms and can map new test data making these algorithms more effective and faster than the previously mentioned techniques [71] [76].

Suppose we are given a set of training data with n points such as $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n\}$ in space $\mathbb{R}^D$ where $\mathbb{R}^D$ is the high-dimensional space of the original data set of D dimensions. The objective is to find a transformation matrix $\boldsymbol{A}$ that can map $\boldsymbol{x}_i$ to $\boldsymbol{y}_i$ with $\{\boldsymbol{y}_1, \dots, \boldsymbol{y}_n\}$ in space $\mathbb{R}^d$ for which $d \ll D$, as shown in Equation (18), while preserving local neighborhood information. The representation $\boldsymbol{y}_i$ of data in d-dimensional space is obtained by a transformation of higher dimensional data $\boldsymbol{x}_i$ in D dimensional space. LGE solves for this transformation through a graph embedding framework.

$$\begin{aligned} \boldsymbol{y'}_i &= \boldsymbol{A'x_i} \\ \boldsymbol{A} &\in \mathbb{R}^{D \times d} \end{aligned} \tag{18}$$

The first step of the LGE algorithm is to form the adjacency graph between nodes. Given $\boldsymbol{G}$ as a graph with n nodes, an edge is assigned between nodes i and j if $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are close to each other. Two variations of determining the closeness between nodes are the *k*-nearest neighbor and

ε-ball [77]. The *k*-nearest neighbor approach is to select the *k* closest points to $\boldsymbol{x}_i$. The ε-ball approach is to find points that satisfy Equation (19) given a parameter ε.

$$\left\|\boldsymbol{x_i} - \boldsymbol{x_j}\right\|^2 < \varepsilon \tag{19}$$

Given the adjacency graph $\boldsymbol{G}$ weights are assigned to detected edges on a separate weight matrix $\boldsymbol{W} = \left(w_{ij}\right)_{n\times n}$. For unconnected nodes, a weight of zero is assigned while for connected nodes, weights can be determined using two variations. The first is the Simple-Minded approach for which a weight is automatically assigned a unitary value if two nodes are connected as shown in Equation (20).

$$w_{ij} = \begin{cases} 1 \; if \; \boldsymbol{x}_i \; and \; \boldsymbol{x}_j \; belong \; to \; the \; same \; class \\ 0 \; otherwise \end{cases} \tag{20}$$

The second variation is based on similar, but distinguishable weight matrices that are specific to LDA, LPP, and NPE as discussed by Cai et al. [78]. For example, the LDA weight matrix is defined by Equation (21) where n is the number of samples associated with classes $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. By doing so, LDA accounts for within-class scattering.

$$w_{ij} = \begin{cases} 1/n \; if \; \boldsymbol{x_i} \; and \; \boldsymbol{x_j} \; belong \; to \; the \; same \; class \\ 0 \qquad\qquad otherwise \end{cases} \tag{21}$$

LPP utilizes the heat kernel approach as shown in Equation (22) for which a weight can be calculated given a parameter of *t* [71] [75]. This weight matrix allows for linear projective mappings which preserve the neighborhood structure of a data set. The equation comes from the studies of heat dispersion of solids and liquids.

$$w_{ij} = e^{\frac{\left\|\boldsymbol{x_i} - \boldsymbol{x_j}\right\|^2}{t}} \tag{22}$$

NPE conststructs a weight matrix based on k-nearest neighbors. The weight matrix is estimated by finding weights that minimize the residual sum of squares for reconstructing each

$\boldsymbol{x}_i$ from its k nearest within-class neighbors given the objective function and constraint of Equation (23) [72].

$$\arg\min_{w_{ij}} \sum_i \left\| \boldsymbol{x_i} - \sum_j w_{ij} \boldsymbol{x_j} \right\|^2$$
$$\text{subject to } \textstyle\sum_j w_{ij} = 1\,, \;\; j = 1,2,\dots,k \tag{23}$$
$$k = \#\, of\ neighbors$$

Given the weight matrix, the diagonal degree matrix $\boldsymbol{D}$ whose elements are the sums of the columns of $\boldsymbol{W}$ is solved in Equation (24). The diagonal degree matrix is a diagonal matrix where each diagonal value identifies how many edges each vertex has. This means that higher values of $\boldsymbol{D}_{ii}$ are more connected to other vertices and therefore more significant.

$$\boldsymbol{D}_{ii} = \sum_j w_{ij} \tag{24}$$

The Laplacian Matrix $\boldsymbol{L}$ is also solved by subtracting the adjacency weight matrix from the diagonal degree matrix as shown in Equation (25).

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W} \tag{25}$$

An objective function used to solve for the optimal map $\boldsymbol{y}$ from the graph is defined in Equation (26). This objective function preserves local neighborhood structuring by ensuring that if i and j are close, than $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$ are also close.

$$\arg\min_{\boldsymbol{A}} \sum_{ij} \left(\boldsymbol{y'_i} - \boldsymbol{y'_j}\right)^2 w_{ij} \tag{26}$$

Given the linear relationship $\boldsymbol{y'_i} = \boldsymbol{A'x_i}$, the objective function is reduced to that of Equation (27) through algebraic formulation.

$$\frac{1}{2}\sum_{ij} \left(\boldsymbol{y'_i} - \boldsymbol{y'_j}\right)^2 w_{ij} = \frac{1}{2}\sum_{ij} \left(\boldsymbol{A'x}_i - \boldsymbol{A'x}_j\right)^2 w_{ij} \tag{27}$$

$$\boldsymbol{Y'(D-W)Y = A'X(D-W)X'A}$$

$$\boldsymbol{Y'LY = A'XLX'A}$$

The objective functions are transformed into the minimization problems of Equation (28). Constraining $\boldsymbol{Y'DY} = 1$ and $\boldsymbol{A'XDX'A} = 1$ removes arbitrary scaling, promotes a unique solution, and reduced the minimization problems to:

$$\begin{aligned} \min_{\boldsymbol{Y}} \frac{\boldsymbol{Y'LY}}{\boldsymbol{Y'DY}} &\rightarrow \min_{\boldsymbol{Y'DY=1}} \boldsymbol{Y'LY} \\ \min_{\boldsymbol{A}} \frac{\boldsymbol{A'XLX'A}}{\boldsymbol{A'XDX'A}} &\rightarrow \min_{\boldsymbol{A'XDX'A=1}} \boldsymbol{A'XLX'A} \end{aligned} \tag{28}$$

The final stage of the LGE algorithm is to form the Eigenmaps. This is done by solving for the eigenvectors and eigenvalues in Equation (28). For $\min_{\boldsymbol{Y}} \boldsymbol{Y'LY}$, $\boldsymbol{Y}$ is a column of vectors which are the solutions of the equation ordered according to their eigenvalues $\lambda_0<\lambda_1<\ldots<\lambda_{l-1}$. The optimal $\boldsymbol{Y}$ is given by the minimum eigenvalue solution to the generalized eigenvalue problem in Equation (29).

$$\boldsymbol{LY = \lambda DY} \tag{29}$$

Similarly, $\boldsymbol{A}$ is a column of vectors which are the solutions of the equation ordered according to their eigenvalues $\lambda_0<\lambda_1<\ldots<\lambda_{l-1}$ [71]. The optimal $\boldsymbol{A}$ is given by the minimum eigenvalue solution to the following generalized eigenvalue problem in Equation (30).

$$\boldsymbol{XLX'A = \lambda XDX'A} \tag{30}$$

Note that these are two separate generalized eigenvalue problems for which the eigenvalues are not the same. The optimal transformation matrix $\boldsymbol{A}$ is then used to map high dimensional data into a lower dimensional space following the linear relationship $\boldsymbol{y'_i = A'x_i}$. Figure 3-4

demonstrates improvements over PCA for the previous clustering example and illustrates how LGE methods preserve neighborhood information.

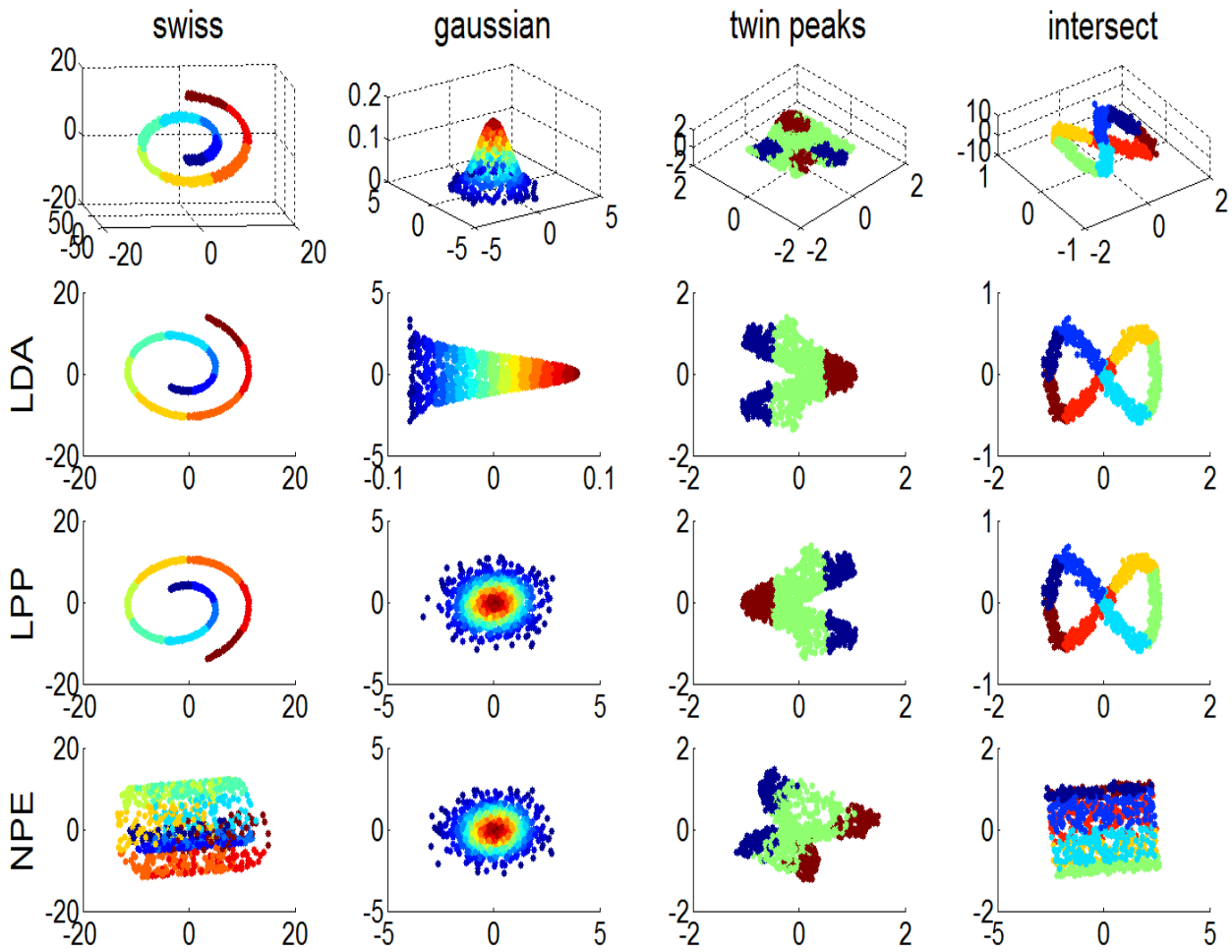


Figure 3-4: LDA, LPP, and NPE dimensionality reduction examples including the swiss roll, Gaussian, twin peak, and intersection.

All 2D embeddings show a preservation of neighborhood classes as their corresponding 3D shapes. LDA, LPP, and NPE mappings vary based on the constructed weight matrix and NPE is not as effective as LDA or LPP for class separation and clustering for the swiss roll and intersect surfaces. This is because NPE relies on a weight matrix based on k-nearest neighbors and the weight matrix cannot account for the intersection of multiple classes. LGE has a cubic

complexity of order $O\left(\frac{3}{2}n^2D + \frac{9}{2}n^3\right)$ where n is the number of data samples and D is the number of dimensions for each feature, where $D > n$ [78].

4 Grassmann Learning

Another approach for deriving meaningful information from high dimensional data is to find low-dimensional representations through linear subspaces using Riemann and Grassmann manifolds. A manifold is a topological space embedded in a high dimensional Euclidean space $\mathbb{R}^D$, such that each manifold point has a neighborhood homeomorphic to a Euclidean space of dimension $m < D$ [79]. A Riemannian manifold $R(M, g)$, is a differentiable manifold M with a smoothly varying inner product g on a tangent space at each point, p. Each point on a Riemannian manifold is essentially a vector space composed of tangent vectors of all possible curves passing through each point p [80]. This property makes a Riemannian manifold a naturally smooth and curved surface where geodesic metrics can be applied. Riemannian manifolds are an alternative over traditional manifolds where high dimensional feature representations do not typically lie on a Euclidean space. Harandi et al. [81] demonstrated improvements in discrimination accuracy by embedding data onto Riemannian manifolds and applying LPP on Riemannian pseudo kernels for the applications of gesture recognition, person re-identification, and texture classification.

Grassmann manifolds $G(m, D)$, a subset of Riemannian manifolds, are manifolds where distances between subspaces can be measured by principal angles. They are the set of m-dimensional linear subspaces of R^D [82]. Grassmann manifolds offer a computation advantage by allowing subspaces to be represented as individual points, they promote high class discrimination by their geometrical structuring, and they account for missing data through subspace spanning. Shigenaka et al. [83] present the Grassmann Distance Mutual Subspace Method (GD-MSM) and Grassmann Kernel Support Vector Machines (GK-SVM) for improved face recognition in comparison to MSM and SVM alone. Park and Savvides [84] adopted

Grassmann kernels into Kernel Principal Component Analysis (KPCA) for face recognition. Turaga et al. [85] embedded representations on Grassmann manifolds and used probability density functions to estimate classes on noisy data with applications on face recognition, shape matching, shape retrieval, and multi-view systems. Hamm and Lee [82] proposed Grassmann kernelized linear discriminant analysis (GDA) for face recognition and object categorization. Similarly, Harandi et al. [79] proposed a Grassmann based graph embedding framework for action analysis.

4.1 Grassmann Framework

Given n training samples in $\boldsymbol{C} \in \mathbb{R}^D$, solve for m unit vector representations of each class where m is the number of samples of each class. Unit vector representations are determined through singular value decomposition (SVD), such that:

$$\begin{gathered} \boldsymbol{C}_{D\times m} = \boldsymbol{U}_{D\times D}\boldsymbol{S}_{D\times m}\boldsymbol{V'}_{m\times m} \\ \boldsymbol{U'U} = \boldsymbol{I}, \qquad \boldsymbol{V'V} = \boldsymbol{I} \end{gathered} \tag{31}$$

where $\boldsymbol{U}_{D\times D}$ is an orthogonal matrix whose columns are the eigenvectors of $\boldsymbol{CC'}$ and $\boldsymbol{V}_{m\times m}$ is the transpose of an orthogonal matrix whose columns are the eigenvectors of $\boldsymbol{C'C}$. The diagonal matrix $\boldsymbol{S}_{D\times m}$ contains the singular values in descending order. With the orthogonal matrix $\boldsymbol{U}_{D\times D}$ define a unit vector $\boldsymbol{u}_{1\times D}$ representation of each sample with an imposed orthogonal constraint. The unit vectors of each k-class are grouped into an orthonormal matrix $\boldsymbol{Y}_{D\times m}$. The span of the orthonormal matrix $\boldsymbol{Y}_{D\times m}$ represents a subspace of a class on a Grassmann manifold. If the columns of $\boldsymbol{Y}$ span a vector $\boldsymbol{u}$, then $\boldsymbol{u}$ can be classified to that subspace. The distances between

subspaces can be measured by their principal angles. A visual overview of the Grassmann framework is shown in Figure 4-1.

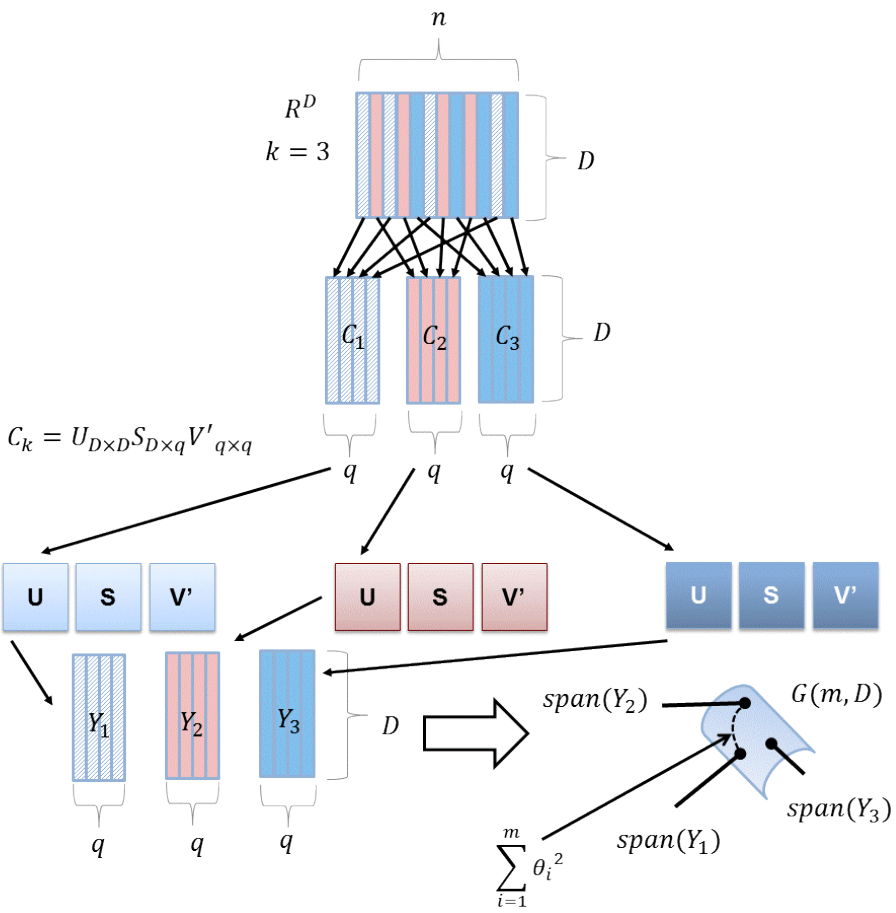


Figure 4-1: This figure demonstrates the mapping of three classes from a Euclidean space onto a Grassmann manifold. The span of the orthonormal matrix Y represents a subspace as a single point on a Grassmann manifold. The geodesic distance between subspaces, $d(Y_i, Y_j) = \sum_{i=1}^{m} \theta_i^2$, is a function of principal angles.

There are many benefits to using Grassmann manifolds. The span of orthonormal matrices embedded as single points promotes high between-class discrimination and promotes within-class clustering. It also allows for directly comparing two subspaces, which is computationally cheaper than measuring all distances between individual elements [82]. Embedding points on a Grassmann manifold has a complexity of $O(Dm^2)$ where D is the number of dimensions and m

is the number of subspaces [86]. Additionally, Grassmann manifolds fill in missing information through linear spans of subspaces.

For an illustrative example, Figure 4-2 shows original 3D shapes and their corresponding orthogonal embedding where each class is plotted separately. The classes in each example are clustered and are separable by their principal angles from other classes, and capable of being compared to other subspaces by geodesic metrics. In all examples, classes are clustered into planes that cross through the origin at different angles. The amount of separation is identifiable by the difference of principal angles between classes.

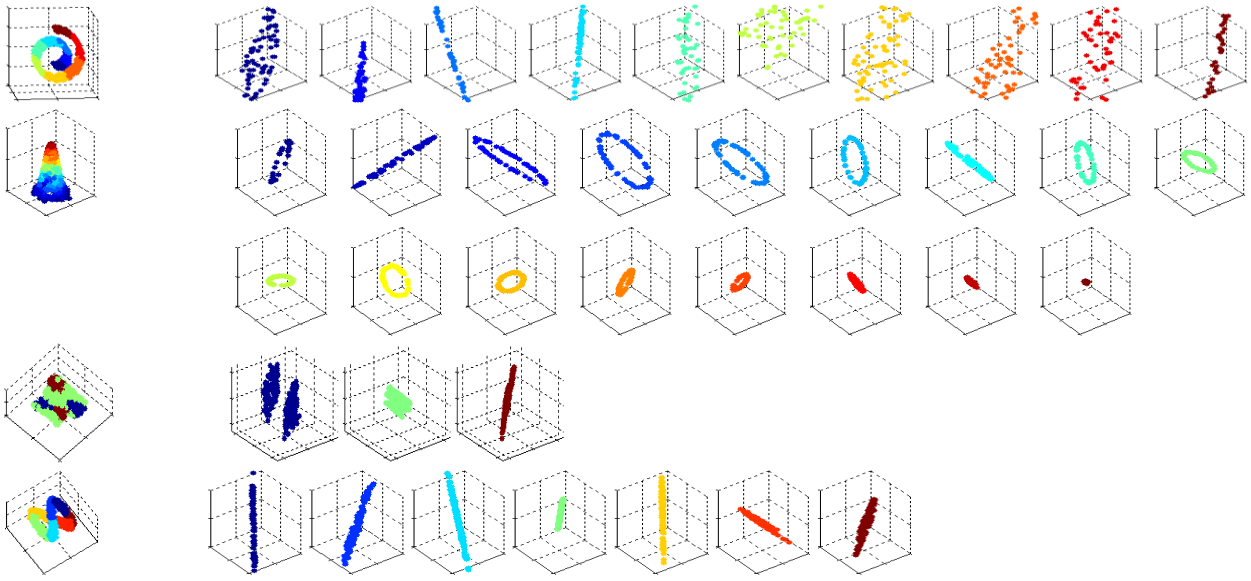


Figure 4-2: The transformation of data from a 3D Euclidean space to their orthogonal embedding. Each class clusters into planes that cross through the origin and classes are separated from each other based on their principal angles.

4.2 Grassmannian Metrics

Given the span of two subspaces $\boldsymbol{Y_1}$ and $\boldsymbol{Y_2}$, a similarity measure between them is a measure based on principal angles $\boldsymbol{\theta} = [\theta_1, \dots, \theta_m]$. The principal angle between two orthonormal matrices is determined by:

$$cos\theta_b = \max_{\substack{\boldsymbol{u}_b \in span(\boldsymbol{Y}_1) \\ \boldsymbol{v}_b \in span(Y_2)}} (\boldsymbol{u}_b' \boldsymbol{v}_b)$$

$$s.t.\ \boldsymbol{u}_b' \boldsymbol{u}_b = 1, \quad \boldsymbol{v}_b' \boldsymbol{v}_b = 1 \tag{32}$$

$$\boldsymbol{u}_b' \boldsymbol{u}_i = 0, \quad \boldsymbol{v}_b' \boldsymbol{v}_i = 0$$

$$(i = 1, \dots, b-1)$$

This is equivalent to solving for the principal angles using SVD such that:

$$\boldsymbol{Y_1}' \boldsymbol{Y_2} = \boldsymbol{USV}'$$

$$diag(\boldsymbol{S}) = (cos\theta_1, \dots, cos\theta_m) \tag{33}$$

Shigenaka et al. [83] and Hamm and Lee [82] define similarity metrics based on principal angles as shown in Equations (34) through (39). Each similarity measure has their benefits and drawbacks. For example, any measure based on all principal angles will balance class discrimination and robustness to noise. Measures based on the smallest principal angle θ_1 tend to be more robust to noise and less discriminative. Measures based on the largest principal angle θ_m tend to be discriminative and less robust to noise.

$$\textit{Projection: } d(\boldsymbol{Y}_i, \boldsymbol{Y}_j) = (m - \textstyle\sum_{i=1}^{m} cos^2\theta_i)^{1/2} \tag{34}$$

$$\textit{Binet-Cauchy: } d(\boldsymbol{Y}_i, \boldsymbol{Y}_j) = (1 - \textstyle\prod_i cos^2\theta_i)^{1/2} \tag{35}$$

$$\textit{Max Correlation: } d(\boldsymbol{Y}_i, \boldsymbol{Y}_j) = (1 - cos^2\theta_1)^{1/2} \tag{36}$$

$$\textit{Min Correlation: } d(\boldsymbol{Y}_i, \boldsymbol{Y}_j) = (1 - cos^2\theta_m)^{1/2} \tag{37}$$

$$\textit{Procrustes: } d(\boldsymbol{Y}_i, \boldsymbol{Y}_j) = 2\left(\textstyle\sum_{i=1}^{m} sin^2\frac{\theta_i}{2}\right)^{1/2} \tag{38}$$

$$\textit{Geodesic: } d(\boldsymbol{Y}_i, \boldsymbol{Y}_j) = \textstyle\sum_{i=1}^{m} {\theta_i}^2 \tag{39}$$

$$\textit{Mean Distance: } d(\boldsymbol{Y}_i, \boldsymbol{Y}_j) = \frac{1}{m}\textstyle\sum_{i=1}^{m} sin^2\theta_i \tag{40}$$

4.3 Grassmannian Kernels

Grassmann manifolds are naturally smooth and curved surfaces. The geometrical characteristics and structuring of Grassmann manifolds are discussed in [87], [88]. With this smooth characteristic, the distance between two subspaces is geodesic. Grassmann kernels provide a means to simplify subspace metrics so that geodesic computations are avoided. Three common Grassmann kernels are projection kernels, canonical correlation kernels, and Binet-Cauchy kernels. In this dissertation, projection kernels are used since they have proven to be the most effective.

4.3.1 Grassmann Projection Kernels

A projection kernel $\boldsymbol{k}_p$ maps an isometric embedding from the Grassmannian space to a projection space. A projection metric is used to calculate the distance between subspaces by measuring the principal angles, $\boldsymbol{\theta} = [\theta_1, \dots, \theta_m]$. The principal angle between two orthonormal matrices is determined by:

$$cos\theta_b = \max_{\substack{\boldsymbol{u}_b \in span(\boldsymbol{Y}_1) \\ \boldsymbol{v}_b \in span(Y_2)}} (\boldsymbol{u}_b' \boldsymbol{v}_b)$$
$$s.t.\ \boldsymbol{u}_b' \boldsymbol{u}_b = 1, \quad \boldsymbol{v}_b' \boldsymbol{v}_b = 1, \quad \boldsymbol{u}_b' \boldsymbol{u}_i = 0, \quad \boldsymbol{v}_b' \boldsymbol{v}_i = 0 \tag{41}$$
$$(i = 1, \dots, b-1)$$

The principal angle is related to the projection metric by:

$$d_p(\boldsymbol{Y}_1, \boldsymbol{Y}_2) = \left(\sum_{i=1}^{k} sin^2\theta_i \right)^{\frac{1}{2}} = \left(m - \sum_{i=1}^{k} cos^2\theta_i \right)^{\frac{1}{2}} \tag{42}$$

This allows for Euclidean distance metrics between two subspaces from isometric embeddings. The projection of two matrices $\boldsymbol{Y}_1$ and $\boldsymbol{Y}_2$ as defined by proposition 1 of Hamm and Lee [82]:

$$K_p(\boldsymbol{Y}_1, \boldsymbol{Y}_2) = tr[(\boldsymbol{Y}_1\boldsymbol{Y}_1')(\boldsymbol{Y}_2\boldsymbol{Y}_2')] = \|\boldsymbol{Y}_1{}'\boldsymbol{Y}_2\|_F^2 \quad (43)$$

The projection kernel can be calculated as the Frobenius norm which is $\|\boldsymbol{Y}_1{}'\boldsymbol{Y}_2\|_F^2$, the square root of the sum of the absolute squares of $\boldsymbol{Y}_1{}'\boldsymbol{Y}_2$. Grassmann kernels require kernel-based methods for classification such as PCA, LDA, etc., as reported in Turaga et al. [89]. Grassmann learning with projection kernels have a time complexity of $O\left(\frac{n(n-1)}{2}(Dm^2)\right)$.

4.4 Grassmannian Principal Component Analysis

A major challenge associated with feature representations, such as motion history surfaces or histograms of local ternary patterns, is high dimensional data representations. The volume of data can be difficult to handle, especially as the number of samples and classes are large. Such data representations can be filled with outliers, noise, redundant data, and are extremely expensive to process in their current high dimensional format. For this reason, subspace learning methods are explored to reduce these action representations to a form that can be processed and analyzed. The motivation for using Grassmann learning is because of its unique characteristics to promote high class discrimination through smooth and curved surfaced, and its ability to improve performance by embedding spans of orthonormal matrices as individual points. A general overview of Grassmann learning is illustrated in Figure 4-3 for face recognition. Local ternary pattern histograms of face image are embedded onto a Grassmann space where projection kernels are created for training and testing as a function of the principal angles between subspaces. The kernels are used for manifold learning of lower dimensional representations. In this section, we consider PCA in combination with Grassmann learning and define Grassmann kernel principal component analysis (GPCA).

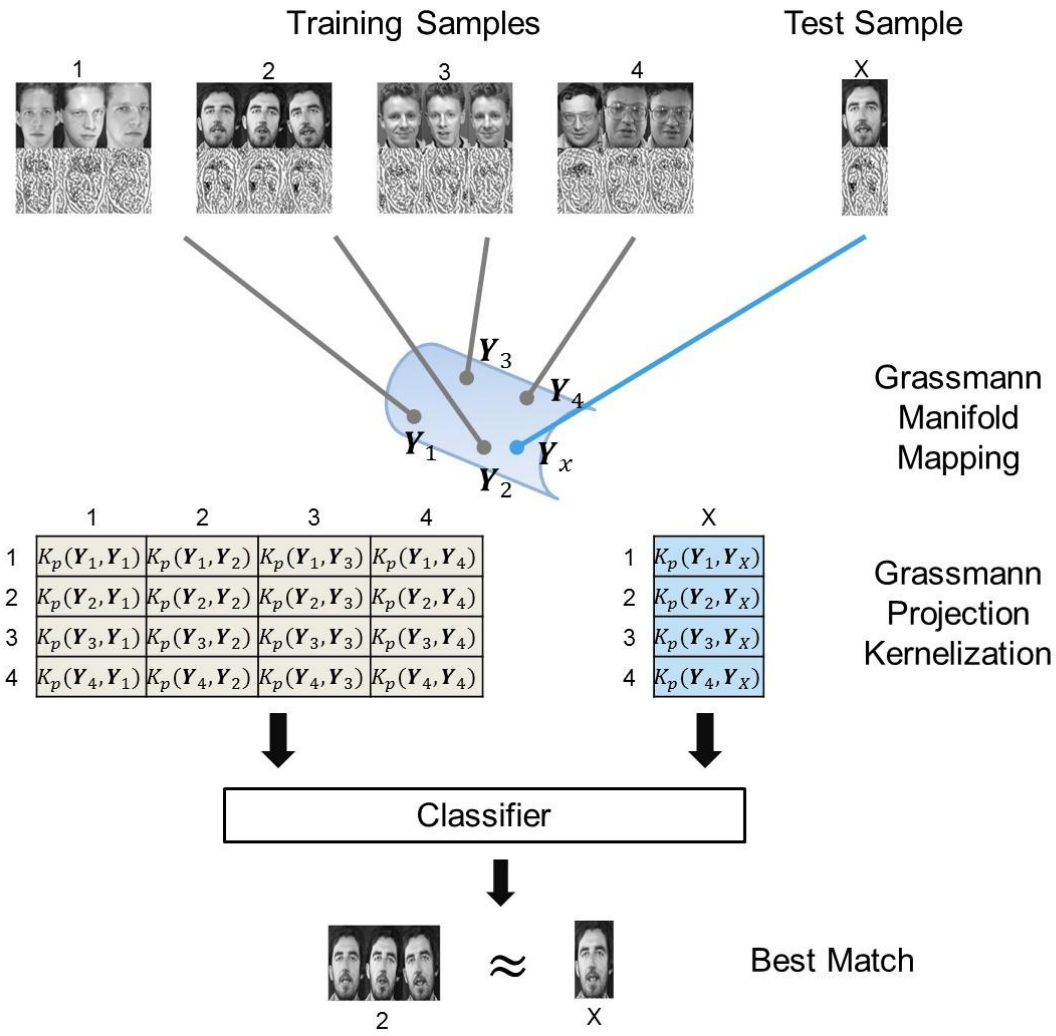


Figure 4-3: An overview of Grassmannian based classifiers for face recognition. For face recognition, local ternary pattern histograms are derived from face images and mapped onto a Grassmann space. Training and testing Grassmann kernels are constructed and processed through manifold learning or sparse representations.

As discussed in Section 3.1, PCA is a linear method for extracting linear features. When processing non-linear features the principal components determined by maximal variance are typically not effective in simplifying the data set successfully. Kernel PCA (KPCA) [90] has proven to be more effective at extracting non-linear structures from data and is well suited for non-linear features. Mika et al. [91] utilize KPCA using Gaussian kernels for denoising and

reconstruction of hand writing characters. Liu [92] used a Gabor based kernel in KPCA for facial expression recognition.

Park and Savvides [84] proposed Multifactor Grassmann Manifolds (MGM) which are a combination of Grassmann manifolds with multi-linear subspace methods included GPCA. PCA identifies principal components of maximal variance by calculating the eigenvectors of the covariance matrix. GPCA identifies non-linear features of high dimensional data by forming the covariance matrix of a Grassmann kernel, and then calculating the eigenvectors of the covariance matrix to identify principal components of maximal variance. The benefit to using Grassmann kernels is due to representations that map an isometric embedding from the Grassmannian space to a projection space while promoting high discrimination. The principal components determined from a Grassmann kernel covariance matrix respect non-linear feature subspaces and high between-class separability. Figure 4-4 shows the 2D embedding of 3D shapes after applying GPCA including a variation in subspace sizes of a single point.

As the subspace sizes increase we see a more clear separation of classes. However, the class separation does not appear to be discriminative enough for classification and recognition systems. Even the three class twin peak shape shows difficulty in distinguishing between the maroon and blue class although there is a clear separation with the green class.

Given that Grassmann learning has a complexity of $O\left(\frac{n(n-1)}{2}(Dm^2)\right)$ with projection kernels and that PCA has a complexity is $O(p^2n + p^3)$, the GPCA time complexity is expected to be $O\left(\frac{n(n-1)}{2}(Dm^2 + p^2m + p^3)\right)$ where n is the number of data samples, D is the number of dimensions, m is the number of Grassmann subspaces, and p is the number of classes.

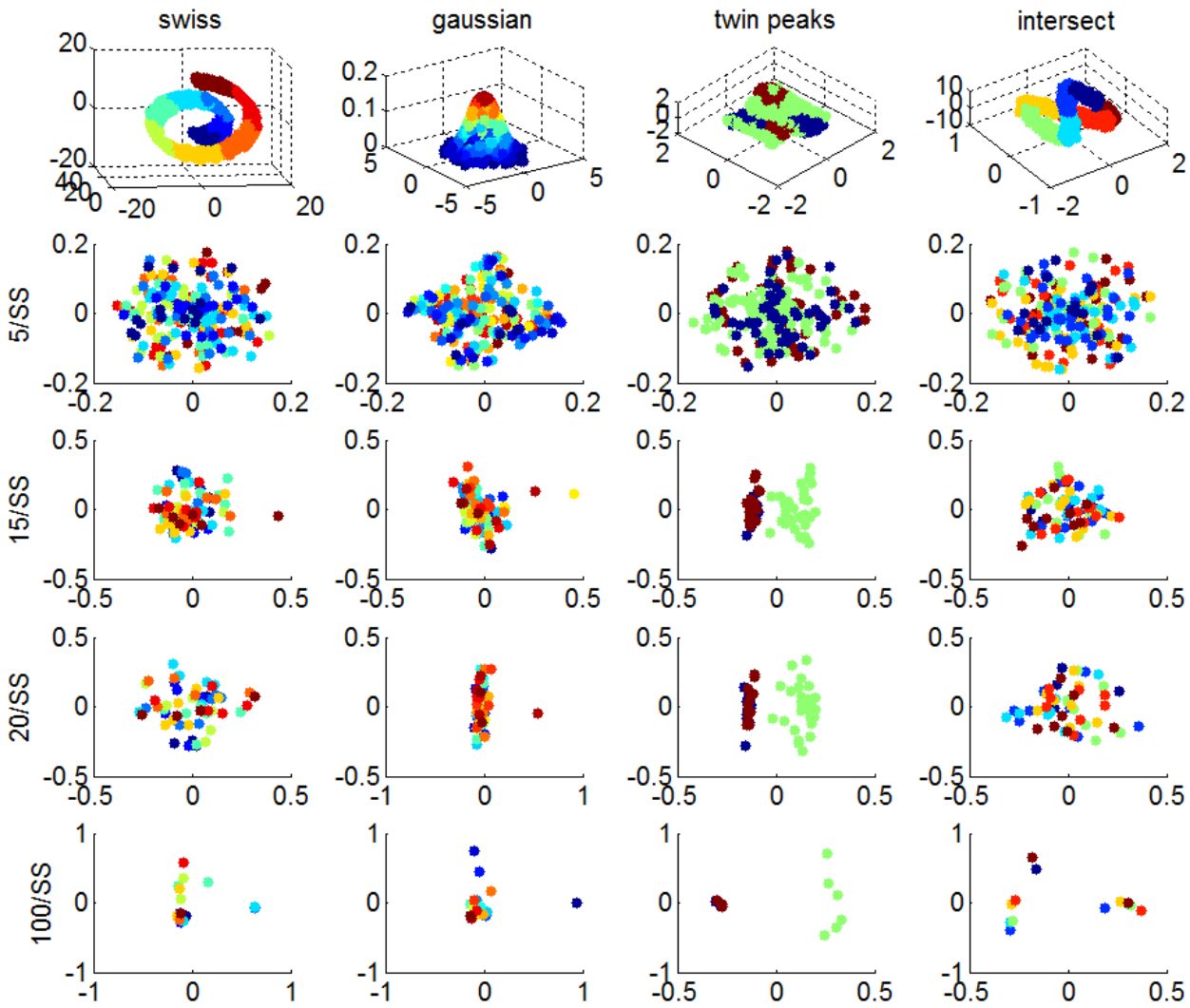


Figure 4-4: The 2D embedding of 3D shapes after applying GPCA. The first row shows the original 3D shapes. The remaining rows show the embedding when each sub space is composed of 5, 15, 20, and 100 data samples per subspace respectively.

5 Grassmannian Sparse Representations

In this chapter, the sparse representations framework and its applications towards classification is presented. Sparse representations are followed up with the formal definition of Grassmannian Sparse Representations (GSR), a subspace learning algorithm that combines the benefits of Grassmann manifolds with sparse representations using least squares loss ℓ^1-norm minimization for improved classification. GSR is another major contribution in this dissertation. This section begins with a background on sparse representations and concludes with a formal definition of GSR.

5.1 Sparse Representations

Another recent development for finding lower dimensional representations is sparse representations. The term *sparse* is a measurable property of a vector associated with the number of non-zero entries contained in that vector. In many real-world systems, data is often sparsely represented, which means that a small portion of a data representation can describe the entire system, and would be beneficial in reducing high-dimensional data. The theory stems from the *Pareto Principle*, a phenomenon that in any population contributing to some common effect only a few members of the population actually contributes to the majority of the effect [93]. This phenomenon can be observed in a wide variety of applications including economics [94], biology [95], and social networks [96]. Sparse representations are a method for finding sparse solutions for underdetermined systems. In computer vision applications, images or video sequences can be encoded using sparse representations to be more easily interpretable and much faster to process. Sparse representations have been used for face recognition [97], super-resolution [98], denoising [99], and image classification [100].

Sparse representation methods have also been utilized for action classification frameworks. Zhang et al. [101] use sparse representations and Bag of Words of spatio-temporal feature descriptors which are projected into a lower dimensional space using PCA and apply ℓ^1-minimization to classify actions. Liu et al. [102] use motion context descriptors to represent frame description and motion context and find sparse representations.

Another recent trend showing success is the interaction of dimensionality reduction methods with sparse representations for improved classification and recognition. Ptucha and Savakis [103] defined a framework for facial expression recognition that combined LGE with K-SVD, an iterative sparse coding technique utilizing singular value decomposition similar to k-means clustering. Lu et al. [104] propose a framework for super-resolution which combines sparse coding with spectral graph processing to learn the geometrical structure of training data. Zheng et al. [105] proposed a sparse coding objective function method that imposes a graph Laplacian regularizer to solve for sparse representations while also accounting for geometrical structures. Experiments were applied for clustering analysis for facial expressions and object classification with a higher rate of success than sparse coding alone. A major drawback with sparse representation classification methods is the issue of run-time performance and memory utilization. The theoretical complexity is difficult to analyze although studies suggest that ℓ^1-norm minimization in the Lasso formulation has an exponential worse case complexity [106] [107]. For this reason, sparse representation classification for high dimensional recognition and classification systems are not ideal.

Given a matrix $\boldsymbol{D}_{m\times n} = [\boldsymbol{D}_1, \boldsymbol{D}_2, \dots, \boldsymbol{D}_p]$ representing an over-complete dictionary of n-action samples, each of m-dimensions, with p separate action classes and a test sample $\boldsymbol{x}$, a linear representation is defined as:

$$\boldsymbol{x} = \boldsymbol{D}\boldsymbol{a} \tag{44}$$

where $\boldsymbol{a}_0 = [0, \ldots, 0, a'_p, 0, \ldots, 0] \in R^n$ is a sparse coefficient vector whose entries are all zero except for those associated with the *p*th action class. Corruption and occlusions can complicate the action classification process affecting the coefficient vector representation [108] by either providing no unique solution or allowing many solutions. Least squares minimization approaches can be used to address the issue. If there is a large number of action classes p, the coefficient representation is naturally sparse [101] and ideally we can find the sparsest solution using ℓ^0-norm minimization:

$$\hat{\boldsymbol{a}} = \arg min \|\boldsymbol{a}\|_0 \ s.t. \ \boldsymbol{x} = \boldsymbol{D}\boldsymbol{a} \tag{45}$$

where $\|\boldsymbol{a}\|_0$ counts the number of non-zeros in vector $\boldsymbol{a}$. However, the system is underdetermined and finding the sparsest solution is NP-hard. ℓ^2-norm minimization or Euclidean norm is a least squares minimization approach based on:

$$\hat{\boldsymbol{a}} = \arg min \|\boldsymbol{a}\|_2^2 \ s.t. \ \boldsymbol{x} = \boldsymbol{D}\boldsymbol{a} \tag{46}$$

where $\|\boldsymbol{a}\|_2^2 = \sum_i \boldsymbol{a}_i^2$. ℓ^2-norm minimization assumes that the best-fit curve has a minimal sum of squared deviations from a dataset [109]. Advantages of ℓ^2-norm minimization are that the solution to the problem is performed easily and the result is always unique. However, an issue with ℓ^2-norm minimization is that the approach assumes a normal distribution which may not be the case for collected data due to noise and errors in the dataset resulting in outliers [110]. ℓ^2-norm minimization utilizes all available examples in order to identify the solution. If the solution $\hat{\boldsymbol{a}}$ is sparse enough, ℓ^0-norm minimization is equal to that of ℓ^1-norm minimization [101] [111]:

$$\hat{\boldsymbol{a}} = \arg min \|\boldsymbol{a}\|_1 \ s.t. \ \boldsymbol{x} = \boldsymbol{D}\boldsymbol{a} \tag{47}$$

where $\|\boldsymbol{a}\|_1 = \sum_i |\boldsymbol{a}|$. ℓ^1-norm minimization promotes sparse solutions and can be reformed as a convex linear programming optimization method. Furthermore, ℓ^1-norm minimization is an effective technique for solving underdetermined systems of linear equations [112] and concentrates on few non-zero coefficients making the approach robust with built-in outlier detection.

There are many methods for ℓ^1-norm minimization, and in this paper we focus on the least squares loss method with regularization:

$$\widehat{\boldsymbol{a}} = \arg min \|\boldsymbol{D}\boldsymbol{a} - \boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{a}\|_1 \;\; s.t. \;\; \boldsymbol{x} = \boldsymbol{D}\boldsymbol{a} \tag{48}$$

where λ is ℓ^1-norm regularization parameter which is used to achieve sparser solutions. When a problem solution is known to be sparse, an applied penalty through regularization provides low variance feature selection, improved approximations, and more interpretable solutions [113]. This is apparent in Figure 5-1 showing the reconstruction coefficients when $\lambda = 0$ and $\lambda = 300$. When $\lambda = 0$, the problem is reduced to an ℓ^2-norm minimization problem.

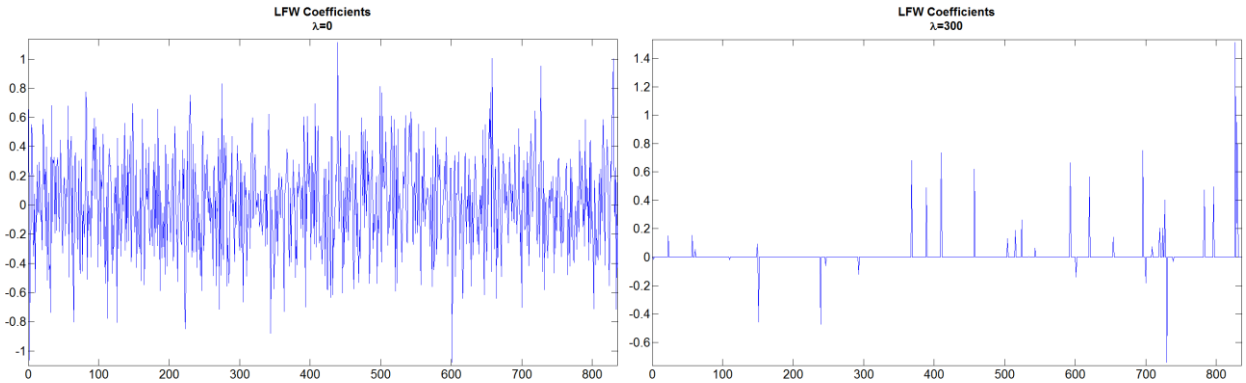


Figure 5-1: These plots show the reconstruction coefficients from the LFW dataset using least squares loss method with regularization following Equation (48). The left plot shows the coefficients when $\lambda = 0$, an ℓ^2-norm minimization problem. The right plot shows the coefficients when $\lambda = 300$.

Given the sparse coefficient vector $\hat{\boldsymbol{a}}$, minimum reconstruction error can be used to classify a test sample to class p. Minimum reconstruction is a preferred classification heuristic because it preserves the linear structure of face and action representations by utilizing all non-zero coefficients [97] for reconstruction. Minimum reconstruction is done by reconstructing a sample from each class and comparing them against the reconstructed sample from all classes using Equation (49) to minimize the residuals. The smallest residual identifies the class p.

$$p^* = \underset{i = 1:p}{\arg min} \left\| \boldsymbol{D}\hat{\boldsymbol{a}}^{\boldsymbol{i}} - \boldsymbol{x} \right\|_2 \tag{49}$$

5.2 3D Action Classification Using Sparse Spatio-Temporal Feature Representations

In this section we present the incorporation of sparse representations for 3D action classification as presented in [114]. Our goal is to define feature descriptors which represent an over-complete dictionary of human actions from depth data, meaning that the dimension of the feature vector is larger than the dimension of the input. We selected two distinct feature descriptors for comparison and evaluation, kinematic 3D joint surfaces (Section 2.3.3) and raw depth data. For raw depth surfaces, we utilized features extracted from raw depth data by determining the largest connected object in the scene and defining a bounding box around that region of interest. The raw data is read from the scene, scaled to a constant feature size, and normalized to obtain a feature descriptor that is invariant to scale and localization. As was done for 3D joint surfaces, to account for variance in action execution time the raw depth surface features were resized to a fixed length using bicubic interpolation. Figure 5-2 shows the resulting descriptor plot for raw

depth data for one instance of time of a subject executing a *waving* action. Figure 5-3 shows the resulting raw 3D action surface for that same action.

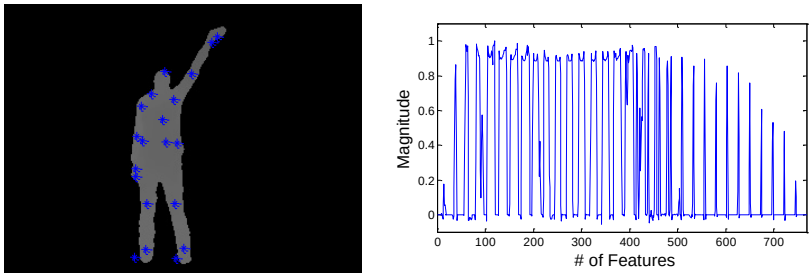


Figure 5-2: Example frame of a test subject performing a *waving* action in 3D space with kinematic coordinates from the MSRAction3D dataset. The plot shows the depth surface descriptor for that frame instance.

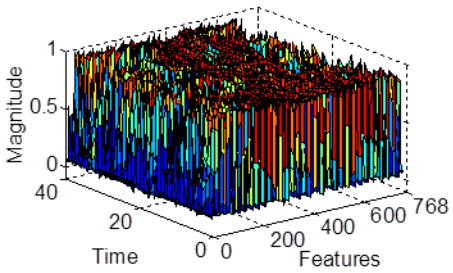


Figure 5-3: The action surface plot for raw 3D depth information of 768 features across 40 frames of a subject executing a *waving* action from the MSRAction3D dataset.

The MSRAction3D dataset (Section 7.2.4) was used for our experiment with the same experimental setup described in Section 2.3.4. Twenty actions were divided into three subsets consisting of eight actions each as presented in Table 2. Additionally, we test against the entire set of. The subsets 1 and 2 were designed to group activities with similar movements while subset 3 was designed to group actions that are more dissimilar, and therefore more suitable for sparser solutions. 2-fold cross validation (2FCV) was used where we randomly select half the subjects for testing and half the subjects for training, and additionally we train and test on both sets allowing for each action sample to be used for either training or validation on each fold. To

ensure large and over-complete dictionaries, we also experiment with leave one out cross validation (LOOCV) where each test subject is validated against the remaining subjects and repeated for all subjects until all subjects have been used for training and testing. The results of our approach are presented in Table 4 and Table 5 for the kinematic joint feature descriptor and the raw depth data extrapolated from the 3D video sequences using ℓ^1-norm minimization and ℓ^2-norm nearest neighbor.

Subset	Cross Validation Approach	ℓ^1-norm Minimization	ℓ^2-norm Nearest Neighbor
Subset 1	LOOCV	**80.73%**	80.21%
	2FCV	**77.66%**	76.60%
Subset 2	LOOCV	77.11%	**78.78%**
	2FCV	73.17%	**75.61%**
Subset 3	LOOCV	**93.89%**	89.29%
	2FCV	**91.58%**	89.47%
Subset 4	LOOCV	72.11%	**72.32%**
	2FCV	63.23%	**73.54%**

Table 4: Results from the MSRAction3D dataset using kinematic joint feature descriptors and cross validation methods, ℓ^1-norm minimization and nearest neighbor.

Subset	Cross Validation Approach	ℓ^1-norm Minimization	ℓ^2-norm Nearest Neighbor
Subset 1	LOOCV	**67.79%**	61.76%
	2FCV	**74.47%**	69.15%
Subset 2	LOOCV	**84.50%**	71.50%
	2FCV	**84.15%**	67.07%
Subset 3	LOOCV	**82.37%**	74.99%
	2FCV	**88.42%**	86.32%
Subset 4	LOOCV	**71.05%**	58.82%
	2FCV	**76.23%**	71.75%

Table 5: Results from the MSRAction3D dataset using raw depth feature descriptors and cross validation methods against ℓ^1-norm minimization and nearest neighbor.

As suspected, the best action classification accuracies came from subset 3 because the dissimilarity between the grouped actions naturally encourages sparser solutions. We also find that with kinematic joint descriptors, ℓ^1-norm minimization does not drastically outperform ℓ^2-norm minimization. This indicates that the normal distribution and utilization of all available

training examples is sufficient and that the kinematic descriptor is not sparse enough to accurately classify actions. However, the kinematic joint descriptor is very powerful descriptor for accurate action classification.

When examining the raw depth data feature descriptor, we begin to see that the natural sparse representation of each action sequence results in an improvement over nearest neighbor classification. We obtain an accuracy of 76.23% on all 20 3D video actions using 2FCV which performs 5.18% better than LOOCV, indicating that the training dictionaries are over-complete without training a majority of the action samples. This is even more apparent when noticing that in almost all cases 2FCV's outperform LOOCV for both ℓ^2-norm minimization and ℓ^1-norm minimization.

5.3 Grassmann Learning with Sparse Representations

Grassmannian Sparse Representations (GSR) is proposed in this dissertation as a framework which combines Grassmannian kernels and sparse representations using least squares loss. The benefits of GSR include improved computational efficiency by reducing the coefficient reconstruction vector size, high with-in class integration along with high between-class separability promoted by Grassmann manifolds, and efficient representations promoted by ℓ^1-norm minimization. The motivation is to combine computational efficiency and high class discrimination, promoted by the structure of Grassmann manifolds, with efficient data representation promoted by ℓ^1-norm minimization.

We construct a training projection kernel $\boldsymbol{K}_{train}$ of size $m_{train} \times m_{train}$, as a kernel mapping of all data elements between each other, where m_{train} is the number of training subspaces. Similarly we construct a testing projection kernel $\boldsymbol{K}_{test}$ of size $m_{train} \times m_{test}$, which maps training subspaces to testing subspaces, where m_{test} is the number of testing

subspaces. With this configuration, kernels can be introduced into the least squares loss function with regularization of Equation (50) such that:

$$\begin{gathered}\hat{\boldsymbol{a}} = \arg min \|\boldsymbol{K}_{train}\boldsymbol{a} - \boldsymbol{K}_{test}(i)\|_2^2 + \lambda\|\boldsymbol{a}\|_1 \\ s.t.\ \boldsymbol{K}_{test} = \boldsymbol{K}_{train}\boldsymbol{a},\ i = [1, \dots, m_{test}]\end{gathered} \tag{50}$$

where $\boldsymbol{K}_{train}$ is the training projection kernel, $\boldsymbol{K}_{test}$ is the testing kernel, $\boldsymbol{a}$ is the coefficient vector, and m_{test} is number of test elements which is equal to the number of testing subspaces. The objective function above promotes sparse solutions through ℓ^1-norm minimization, an effective technique for solving underdetermined systems of linear equations with outlier detection, and promotes class discrimination through Grassmannian manifolds. It should also be noted that either individual elements or a group of elements may be treated as a single subspace through Grassmann learning depending on the application.

With the reduction from a high dimensional space to training and testing kernels, classification can be carried out using minimum reconstruction to identify which Grassmann embedded subspace class is most associated with a new Grassmann embedded test sample. Given the coefficient vector $\hat{\boldsymbol{a}}$ determined from Grassmann kernels, minimum reconstruction can be used to classify a test sample by reconstructing a sample from each class from projected Grassmann points and comparing them against the reconstructed sample from all classes of projected Grassmann points using:

$$\begin{gathered}p^* = \underset{j\,=\,1:p}{\arg min} \left\|\boldsymbol{K}_{train}\hat{\boldsymbol{a}}^{\boldsymbol{j}} - \boldsymbol{K}_{test}(i)\right\|_2 \\ s.t.\quad i = [1, \dots, m_{test}]\end{gathered} \tag{51}$$

There are many benefits of the GSR framework. Fast high dimensional data reduction is achieved through linear derivations of weighted isometric embeddings from a Grassmann space to a Euclidean space. The Grassmannian component of the algorithm supports high between

class discrimination because these manifolds have smooth structure and can fill in missing data through linear spanning.

The sparse representation component of the algorithm is representing a linear combination of basis vectors from Grassmann kernels rather than high dimensional data input. This automatically incorporates the benefits of Grassmann learning in a sparse coding framework. Additionally, regularization can easily be incorporated to solve for the sparse reconstruction coefficients. Grassmann subspaces can represent an entire class and the number of sparse reconstruction coefficients can reduce to the number of classes in the classification system. For multi-view action systems, a single action class, independent of the viewpoint, can be represented as a single point on a Grasssmann space. Multiple trials of the same 3D action class can also be represented as single points. Face images of one subject of varying illuminations and expressions can be represented as a single point. These reductions simplify reconstruction and will reduce the computation load.

Grassmann learning has a squared time complexity of $O(n^2Dm^2)$ with projection kernels [79] where n is the number of samples, m is the number of subspaces in the Grassmann space, and D is the number of dimensions of each input sample. Sparse representation classification has a theoretical exponential complexity. However, in the GSR framework the time complexity would be exponential on a Grassmann kernel and can therefore perform the fastest when an entire class is represented as a single point on a Grassmann space.

6 Grassmannian Spectral Regression

In this chapter spectral regression and its applications towards classification is presented. Spectral regression is followed up with the formal definition of Grassmannian Spectral Regression (GRASP), a subspace learning algorithm which leverages the benefits of Grassmann manifolds and Spectral Regression in a framework that supports high discrimination between classes and achieves computational benefits by using manifold modeling and avoiding eigen-decomposition. GRASP is the next major contribution in this dissertation.

6.1 Spectral Regression

While eigen-based linear subspace approaches are effective at learning linear and non-linear representations of data, recent efforts have emerged towards least squares frameworks because of drawbacks associated with eigen-formulations. DelaTorre [115] suggests that eigen-decomposition results in normalization factors and inaccuracies with rank deficient matrices, and proposes a least-squares weighted kernel reduced rank regression (LS-WKRRR). Cai et al. [78] encourage the avoidance of eigen-decomposition because of computational inefficiencies and introduces Spectral Regression for regularized subspace learning. Based on regression and spectral graph analysis, this approach enables regularization which is not as simple to do with eigen-decomposition.

Spectral Regression (SR) [78] is a regularized subspace learning approach that overcomes the disadvantages of eigen-based approaches in terms of inefficiencies in execution time performance, memory allocation, and regularization. With the LGE framework the minimization problem for $\boldsymbol{Y}$ is $\min_{\boldsymbol{Y}} \frac{\boldsymbol{Y'LY}}{\boldsymbol{Y'DY}}$ and the minimization problem for $\boldsymbol{A}$ is $\min_{\boldsymbol{A}} \frac{\boldsymbol{A'XLX'A}}{\boldsymbol{A'XDX'A}}$. Constraining

$\boldsymbol{Y'DY} = 1$ and $\boldsymbol{A'XDX'A} = 1$ allows for the problems to be generalized to the minimization problems $\min_{\boldsymbol{A}} \boldsymbol{Y'LY}$ and $\min_{\boldsymbol{A}} \boldsymbol{A'XLX'A}$ respectively. These are also equivalent to the maximization problems $\max_{\boldsymbol{Y}} \boldsymbol{Y'WY}$ and $\max_{\boldsymbol{A}} \boldsymbol{A'XWX'A}$ corresponding to their maximum eigenvalues:

$$\boldsymbol{WY} = \boldsymbol{\lambda DY} \tag{52}$$

$$\boldsymbol{XWX'A} = \boldsymbol{\lambda XDX'A} \tag{53}$$

The eigenvalues $\boldsymbol{\lambda}$ for Equation (52) and the eigenvalues $\boldsymbol{\lambda}$ for Equation (53) are distinct. Given the linear relationship $\boldsymbol{Y'} = \boldsymbol{A'X}$ and that $\boldsymbol{A}$ is the eigenvectors of Equation (53), the Spectral Regression framework redefines $\boldsymbol{Y}$ to be the eigenvectors of Equation (52) so that the eigenvalues $\boldsymbol{\lambda}$ of both eigen-problems are the same. To solve for the eigenvectors $\boldsymbol{A}^*$ efficiently the spectral regression approach follows a two-step iterative process outlined below:

(1) Solve for $\boldsymbol{Y}$ in Equation (52).

(2) Solve for the eigenvectors $\boldsymbol{A}^*$ corresponding to the maximum eigenvalue of Equation (53) that satisfies $\boldsymbol{Y'} = \boldsymbol{A'X}$, using least squares regression and the equation below where $\boldsymbol{y_i}$ is the i^{th} element of $\boldsymbol{Y}$.

$$\boldsymbol{A}^* = \arg\min_{\boldsymbol{A}} \sum_{i=1}^{n} (\boldsymbol{A'x_i} - \boldsymbol{y'}_i)^2 \tag{54}$$

The minimization problem could be underdetermined with many possible solutions. To account for this, regularization can be used with parameter α regulating the amount of shrinkage, and an applied penalty on the norm of $\boldsymbol{A}$, where $\|\boldsymbol{A}\|^2$ is an ℓ^2 norm:

$$\boldsymbol{A}^* = \arg\min_{\boldsymbol{A}} \left(\sum_{i=1}^{n} (\boldsymbol{A'x_i} - \boldsymbol{y_i})^2 + \alpha \|\boldsymbol{A}\|^2 \right) \tag{55}$$

A class value is assigned by performing classification in the lower dimensional space using k-Nearest Neighbor (k-NN) or another classifier. Other types of regularizers can be incorporated, which demonstrates the flexibility of regularized subspace learning for adaptation to various applications.

Spectral Regression is known to be more effective for smaller class problems [78]. Figure 6-1 shows spectral regression dimensionality reduction on the same four shapes in our example with a varying regularization parameter, α.

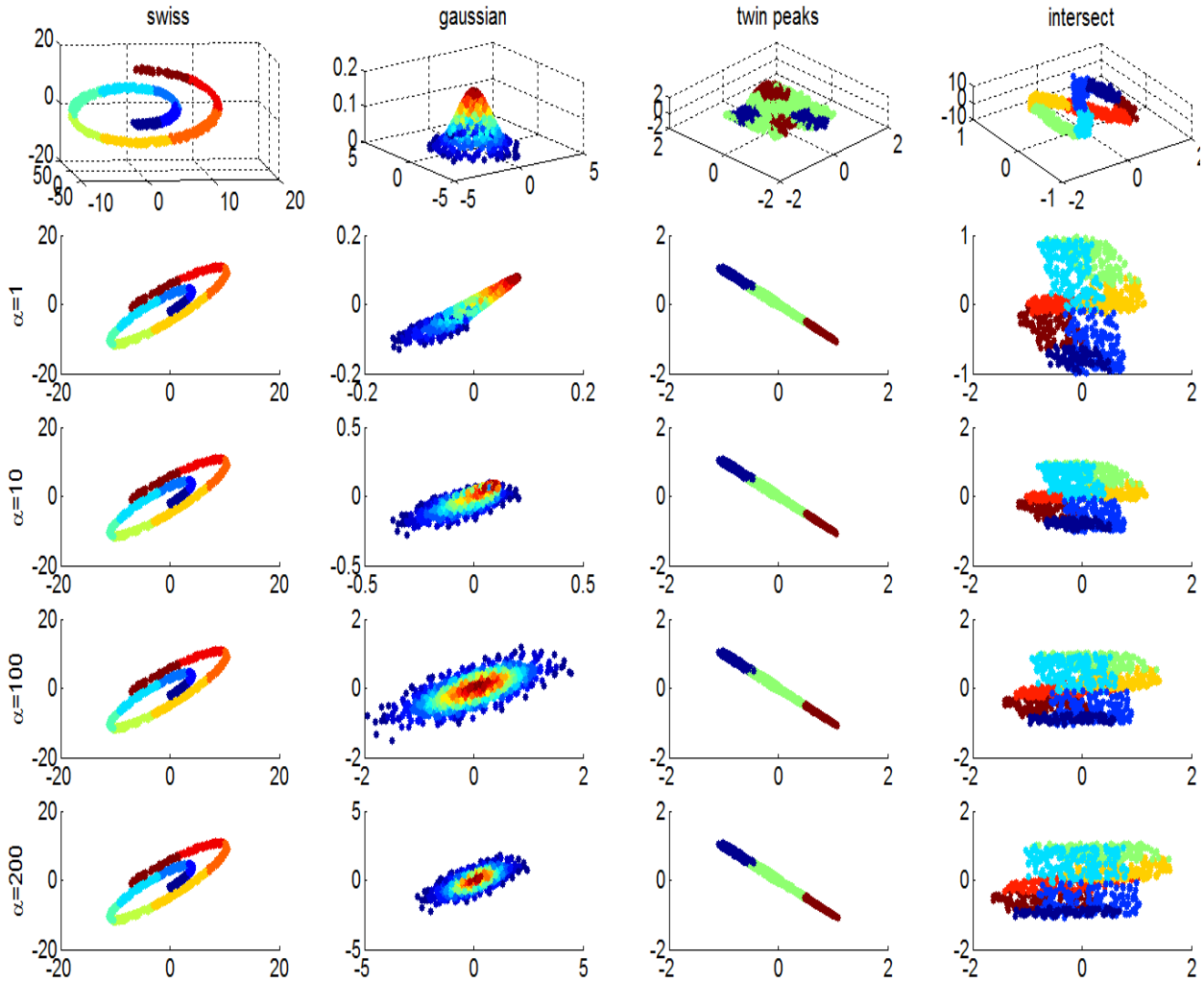


Figure 6-1: Spectral regression dimensionality reduction examples including the swiss roll, Gaussian, twin peak, and intersection with a varying regularization parameter, α.

With Spectral Regression, the Gaussian surface is embedded in a different yet separable manner compared to PCA and LPP when α is small. As the α increases the 2D embedding of the Gaussian shape takes form and appears more separable. Meanwhile, the intersect shape appears more separable when α is small. Regularization appears to have minimal impact on the swiss roll and twin peaks shapes. The twin peaks class separation is closely clustered in comparison to PCA and LPP. The swiss roll and intersect shapes class discrimination is degraded in comparison to LPP because of the larger number of classes involved.

While LGE has a cubic complexity, analysis of computation complexities finds that Spectral Regression has a linear complexity of $O(2csnD)$ where n is the number of data samples, D is the number of dimensions for each feature such that $D > n$, c is the number of classes, and s is the number of iterations in the least squares framework [78].

6.2 Grassmann Learning with Spectral Regression

Grassmannian Spectral Regression (GRASP) combines Grassmann manifolds with Spectral Regression in a framework that is computationally efficient, offers improved class separability, supports regularization, and does not require eigen-decomposition. The important benefit of GRASP is improved classification performance due to high within class integration along with high between-class separability promoted by Grassmann manifolds, along with a drastic improvement in computational performance achieved by manifold modeling and avoiding eigen-decomposition. There are two problems with eigen-decomposition subspace learning. First they add a level of computational complexity as suggested by DelaTorre [115]. Secondly, such algorithms do not easily incorporate regularization.

To begin, we construct training projection kernels $\boldsymbol{K}_{train}$ of size $m_{train} \times m_{train}$, as a kernel mapping of all data elements, where m_{train} is the number of training subspaces. Similarly we construct testing projection kernels $\boldsymbol{K}_{test}$ of size $m_{train} \times m_{test}$, which map training subspaces to testing subspaces, where m_{test} is the number of testing subspaces. These kernels map the Grassmannian space to a projective space. The objective is to find a transformation matrix $\boldsymbol{A}$ that maintains the linear relationship and preserves neighborhood information between the training Grassmannian kernel $\boldsymbol{K}$ and the lower dimensional representation $\boldsymbol{Y}$:

$$\boldsymbol{Y}' = \boldsymbol{A}'\boldsymbol{K} \tag{56}$$

This can be accomplished through the spectral regression framework. Given the eigen-problems $\boldsymbol{WY} = \boldsymbol{\lambda DY}$ and $\boldsymbol{KWK}'\boldsymbol{A} = \boldsymbol{\lambda KDK}'\boldsymbol{A}$, redefine $\boldsymbol{Y}$ to be the eigenvectors so that the eigenvalues $\boldsymbol{\lambda}$ of both eigen-problems are the same. The eigenvectors $\boldsymbol{A}^*$ can be solved by the following two step process:

(1) Solve for $\boldsymbol{Y}$ in $\boldsymbol{WY} = \boldsymbol{\lambda DY}$

(2) Solve for the eigenvectors $\boldsymbol{A}^*$ corresponding to the maximum eigenvalue of $\boldsymbol{KWK}'\boldsymbol{A} = \boldsymbol{\lambda KDK}'\boldsymbol{A}$ that satisfies $\boldsymbol{Y}' = \boldsymbol{A}'\boldsymbol{K}$.

We use least squares regression by introducing Grassmann kernels into the least squares loss function with regularization to promote a unique solution such that:

$$\boldsymbol{A}^* = \arg\min_{\boldsymbol{A}} \left(\sum_{i=1}^{P} (\boldsymbol{A}' k_i - \boldsymbol{y}_i)^2 + \lambda \|\boldsymbol{A}\|^2 \right) \tag{57}$$

$$\boldsymbol{K} = \left[k_1, \ \ldots, k_P\right], \qquad \boldsymbol{Y} = [\boldsymbol{y}_1, \ \ldots, \boldsymbol{y}_P]'$$

where P is the number of subspaces on the Grassmann manifold, $\{k_i\}_{i=1}^{P} \in G(m, D)$. This formulation [116] allows for least squares regularization of an isometric embedding in Grassmann space instead of a high dimensional Euclidean space.

$\boldsymbol{K}$ can be any type of kernel and in this dissertation projection kernels are used. A weighted representation of the projection kernels and canonical correlation kernels was proposed in [82], such that $\boldsymbol{K} = \alpha \boldsymbol{K}_p + \beta \boldsymbol{K}_{cc}$, where α regulates the projection kernel and β regulates the canonical correlation kernel. The eigenvectors $\boldsymbol{A}^*$ gives a linear method of reducing the kernel data such that $\boldsymbol{Y}' = \boldsymbol{A}'\boldsymbol{K}$. It is then possible reduce the dimensions of the training and testing kernels following:

$$\begin{aligned} \boldsymbol{Y}_{train} &= \boldsymbol{A}'\boldsymbol{K}_{train} \\ \boldsymbol{Y}_{test} &= \boldsymbol{A}'\boldsymbol{K}_{test} \end{aligned} \tag{58}$$

With the reduced training and testing kernels, classification can be carried out using k-NN to classify a test subspace. Since each training subspace represents an entire class, only one nearest neighbor (1-NN) classification is required because each training class is represented as a single point on a Grassmann space. There are many benefits of the GRASP framework. The spectral regression component of the algorithm allows for regularization to quickly converge to a unique solution while avoiding the computational burden of eigen-based approaches. Fast high dimensional data reduction is achieved through linear derivations of weighted isometric embeddings from a Grassmann space to a Euclidean space. The Grassmannian component of the algorithm supports high between class discrimination because these manifolds have smooth structure and can fill in missing data through linear spanning.

Figure 6-2 demonstrates the 2D embedding of 3D shapes with various subspace sizes. This example illustrates how the number of samples decreases and class clustering improves as the

number of samples per subspace increase. When compared to GPCA there is a more clear class separation as the subspace sizes become larger.

Grassmann learning has a complexity of $O\left(\frac{n(n-1)}{2}(Dm^2)\right)$ with projection kernels. Given that Spectral Regression has a linear computational complexity of $O(2csnD)$, GRASP would require $O\left(\frac{n(n-1)}{2}(Dm^2 + 2c^2s)\right)$ or $O(n^2Dm^2)$ operations. This is because spectral regression is applied on Grassmann projection kernels where the number of data samples n is equal to the number of classes c, and the number of dimensions D of each class is a scalar. This is equal to the graph embedding discriminant analysis squared complexity of $O\left(\frac{n(n-1)}{2}(Dm^2 + m^3)\right)$ or $O(n^2Dm^2)$ operations [79] when $m \ll D$ and $n \ll D$. The difference is minimal for small input action classification systems. However, as the number of samples n increases, so does the number of inputs n in each subspace m. As the inputs get larger, GRASP would maintain its performance while Grassmann graph embedding techniques would require more operations.

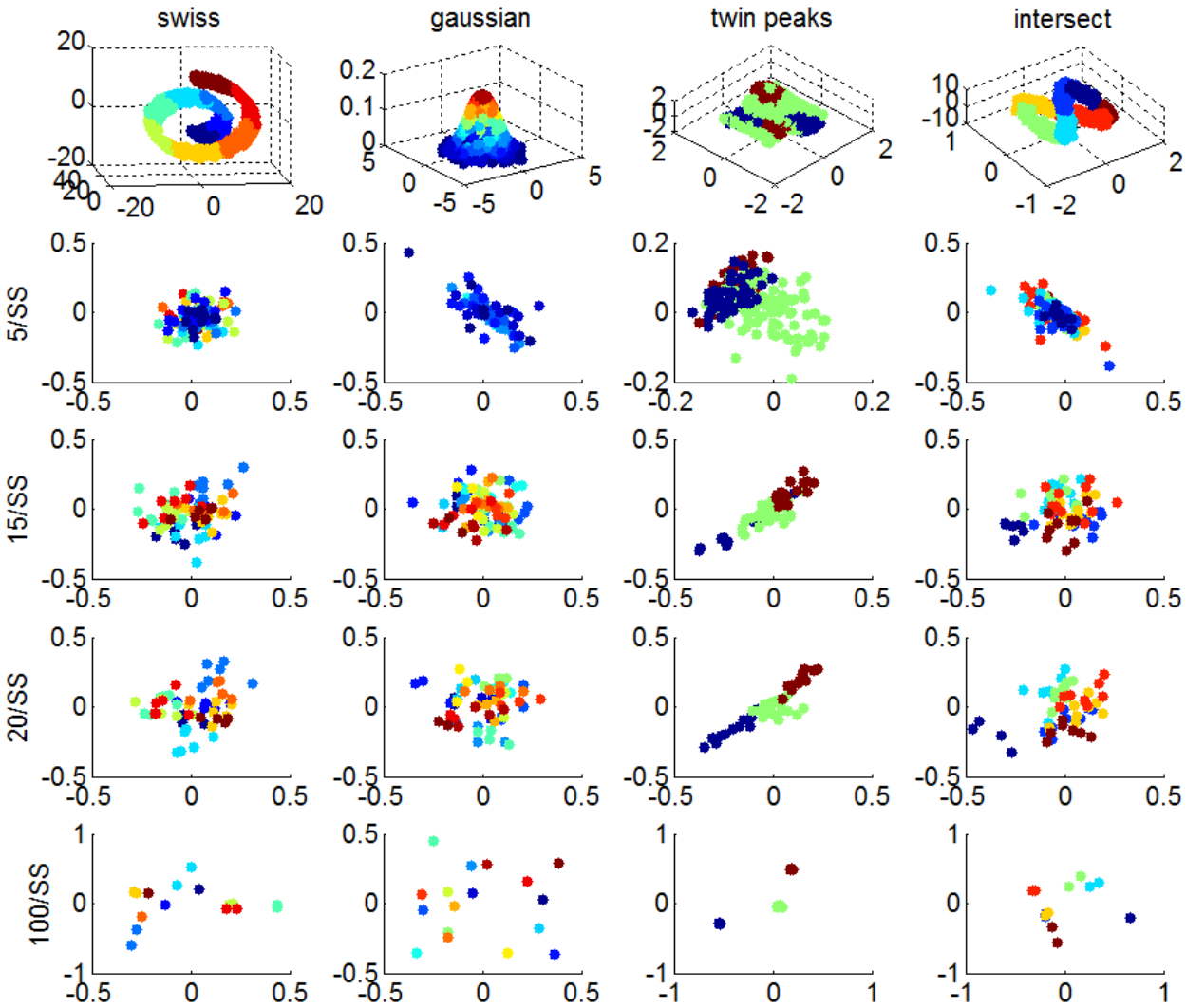


Figure 6-2: The 2D embedding of 3D shapes after applying GRASP. The first row shows the original 3D shapes. The remaining rows show the embedding when each sub space is composed of 5, 15, 20, and 100 data samples per subspace respectively.

7 Experimental Setup and Analysis

In this section the experimental setup of GSR and GRASP is presented. The focus is on multi-view action classification, 3D action classification, and face recognition. Motion history surface (MHS) descriptors [116] were used for multi-view datasets and motion depth surface (MDS) descriptors [117] were used for 3D datasets. Local Ternary Pattern descriptors [43] were used for facial recognition datasets.

7.1 Evaluation Assumptions

A few assumptions are presumed for the work presented in this dissertation. It is assumed that all actions provided in the action datasets have segmented silhouettes obtained through an existing approach. All action and face datasets provide images without major occlusions. This means that actors are visible in a scene throughout most of the time that an action is being conducted without purposely being blocked from the view of the camera. For face images, actors with glasses and under extreme illumination variations are expected. For action classification, the most complex scenes performed in all the datasets are interactions between individuals which are provided by the i3DPost dataset. It is also assumed that the action classification systems being evaluated are segmented in time, i.e. we are given the starting and ending time of an action. In other words the systems used do not automatically apply temporal segmentation.

7.2 Datasets

The datasets used for experimentation are (a) multi-view action datasets using the i3DPost Multi-View dataset, the IXMAS Multi-View dataset, and the WVU Multi-View dataset; (b) 3D action datasets using the MSRAction3D dataset and MSRGesture3D dataset; and (c) face image datasets using the ATT dataset, LFW dataset, and Yale Extended Face dataset.

7.2.1 i3DPost Multi-View Human Action Dataset (i3DPost)

The i3DPost multi-view human action dataset [27] provides synchronized multiple views of individuals performing action sequences. The dataset consists of synchronized high definition images of 8 views performed by 8 people executing 12 actions.

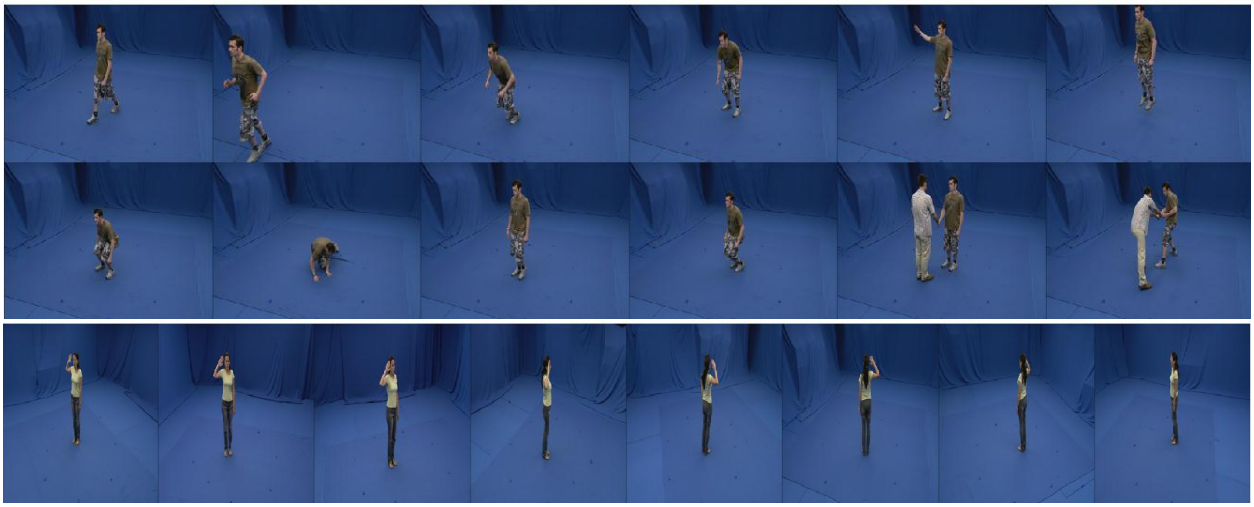


Figure 7-1: These are sample frames from the i3DPost multi-view dataset. The top group of images show a sample of all 12 actions from one view. The bottom group of images show multiple views of one instance of time of a wave action.

Each action is performed over 125 frames. The actions include individual actions such as *walk, run, jump, bend, hand-wave,* and *jump in place*. The dataset also includes action combinations where multiple actions are executed in the same sequence, which are *sit-stand up, run-fall, walk-sit,* and *run-jump-walk*. Finally, the dataset also include interactions between two

individuals, which are *handshake* and *pull*. The images are provided in a high-resolution color format in PNG files and also include background images for image differencing, camera calibration parameters for 3D reconstruction, and 3D mesh models.

7.2.2 INRIA Xmas Motion Acquisition Sequences (IXMAS)

The INRIA Xmas Motion Acquisition Sequences (IXMAS) dataset was presented by Weinland et al. [118] and was created in 2005 including extracted silhouettes. The dataset offers 390x291 pixels resolution images in PNG/BPM formats. There are five synchronized views captured at 50FPS of ten subjects executing 14 actions between 2 and 3 trials each. The fifth view is a top view and was ignored in the experiments. The actions include *Check Watch, Cross Arms, Scratch Head, Sit Down, Get Up, Turn Around, Walk, Wave, Punch, Kick, Point, Pick Up, Throw (overhand),* and *Throw (underhand).* Underhand throwing was excluded from the experiments because there were 75% less underhand throwing samples than all other action samples.

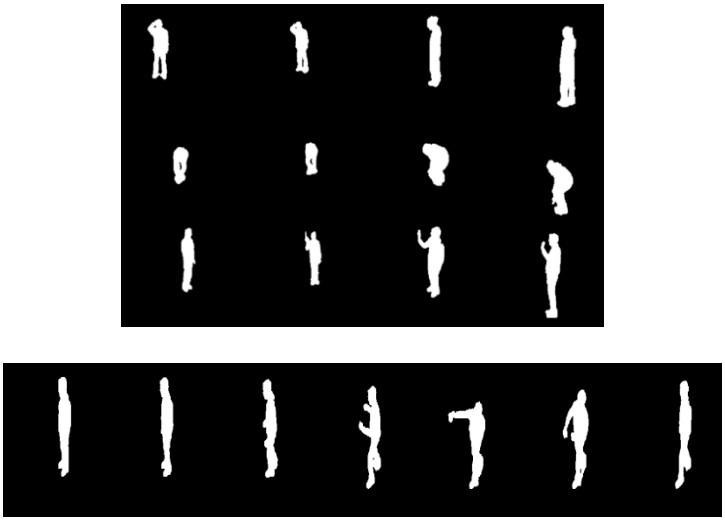


Figure 7-2: Samples from the IXMAS dataset. The top images show four views of an individual executing *Scratch Head, Pick Up,* and *Wave*. The bottom images show one view of one subject performing a *Punch*.

7.2.3 West Virginia University Multi-View Action Dataset (WVU)

The West Virginia University (WVU) multi-view dataset [119] provides 8 views of 5 subjects performing 12 actions executed at 20FPS and a resolution of 640x480 pixels. The actions include *Standing Still, Nodding head, Clapping, Waving 1 hand, Waving 2 hands, Punching, Jogging, Jumping Jack, Kicking, Picking, Throwing,* and *Bowling*. The standing still action was excluded because the motion history surface descriptors expect motion. The action sequences are not consistently synchronized over all views as can be seen in Figure 7-3 and extracting the silhouettes from this dataset were challenging because of variations in lighting in various images.

Figure 7-3: Sample frames from the WVU dataset. The top group of images show multiple views of one instance of time of a subject performing a *two handed wave*. The bottom group shows a subject performing *jumping jacks*.

7.2.4 Microsoft Research Action 3D Dataset (MSRAction3D)

The Microsoft Research Action 3D (MSRAction3D) Dataset [36] consists of depth map sequences recorded with a depth sensor at 15 FPS and 320×240 pixel resolution. There are ten subjects performing twenty actions two to three times for a total of 567 depth map sequences. The dataset actions are: *high arm wave, horizontal arm wave, hammer, catch, tennis swing, forward punch, high throw, draw X, draw tick, tennis serve, draw circle, hand clap, two hand*

wave, side boxing, golf swing, side boxing bend, forward kick, side kick, jogging, and *pick up and throw*. No corresponding RGB information is available, however 3D joint positions are available. All silhouettes have been segmented as demonstrated in the sample action frames of Figure 7-4. Figure 7-5 illustrates sample depth frames with kinematic joint identifiers.

Figure 7-4: Sample depth frames from the MSRAction3D dataset showing a *forward punch* action.

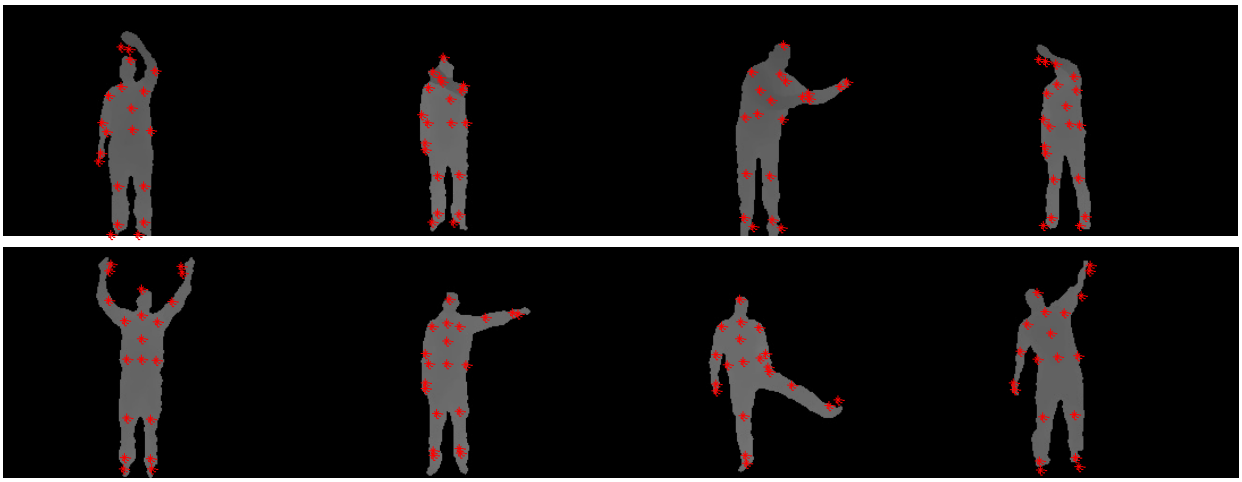


Figure 7-5: Sample frames from the MSRAction3D dataset with plotted kinematic joints of a high arm wave, horizontal arm wave, golf swing, draw X, two-hand wave, side boxing, side kick, and a tennis serve.

7.2.5 Microsoft Research Gesture3D Dataset (MSRGesture3D)

In the Microsoft Research Gesture3D (MSRGesture3D) dataset [120] there are ten people performing 12 American Sign Language (ASL) gestures which represent *Z, J, Where, Store, Pig, Past, Hungary, Green, Finish, Blue, Bathroom, and Milk*. There are between two and three

gesture trials for each subject with a total of 336 image files. The dataset consists of depth map sequences recorded with a depth sensor at 10 FPS and resolution of 106×160 pixels. The dataset contains some dead frames and we applied interpolation to correct for the dead frames when applicable. The sample frames for the ASL for the letters *J* and *Z* are shown in Figure 7-6.

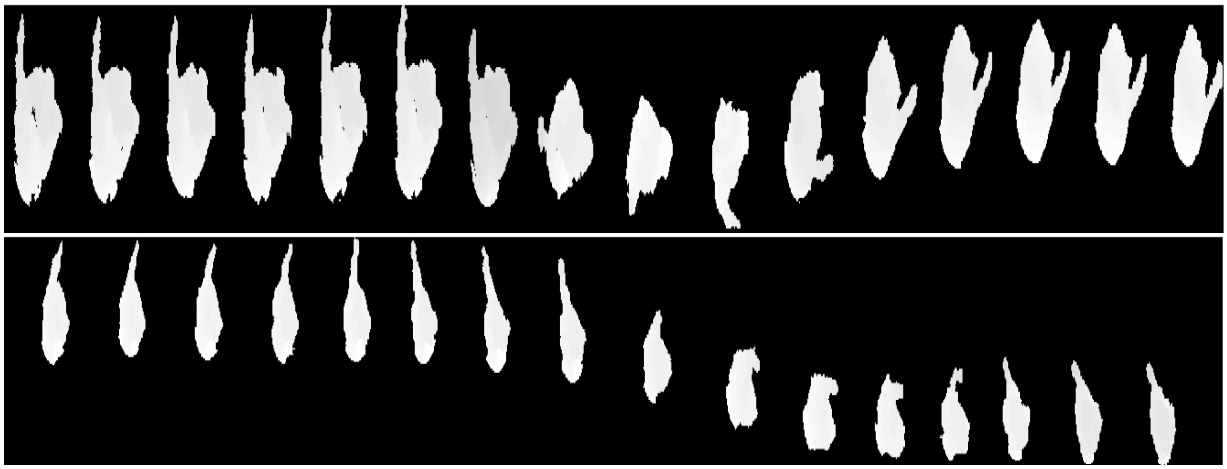


Figure 7-6: Sample depth frames from the MSRGesture3D dataset showing the ASL for *J* (top) and ASL for *Z* (bottom).

7.2.6 Database of Faces from AT&T Laboratories (ATT)

The database of faces from AT&T laboratories (ATT) [121] is a collection of faces images from 40 subjects with 10 face images per subject. There are a total of 400 face images in PGM file format. Each face image is 92x112 pixels and all images are grayscale. The face images of each subject can vary by pose, lighting, facial expressions, and facial details such as glasses as demonstrated in Figure 7-7. Figure 7-8 shows a sample face image of each of the 40 subjects. All subjects and all faces images are used in our experiments.

Figure 7-7: Face images for two subjects from the database of faces from AT&T laboratories. The face images of the first subject contain images with and without glasses. The second subject face images vary by expression.

Figure 7-8: Sample face images from each of the 40 subjects from the database of faces from AT&T laboratories.

7.2.7 Labeled Faces in the Wild (LFW)

The Labeled Faces in the Wild (LFW) dataset [122] is a face database with 5749 individuals and 13,233 total face images collected from the web. The images are cropped and are in PGM file format. The subjects vary by many parameters including pose, lighting, expression, background, race, ethnicity, age, gender, clothing, hairstyles, camera quality, color saturation, and focus. In

our experiments, we used all subjects who have at least 20 face images. We did not exceed the use of 30 face images per subject. Therefore, 62 subjects were used for face recognition with a total of 1,673 total face images. Figure 7-9 shows multiple face image samples of two subjects. These samples illustrate face images that vary in terms of expressions, pose, and illumination. Figure 7-10 shows a sample image from each of the 62 subjects used for evaluation.

Figure 7-9: Face images for Donald Rumsfeld (top) and Hans Blix (bottom) from the labeled faces in the wild database.

Figure 7-10: One face image sample of each of the 62 subjects from the labeled faces in the wild database.

7.2.8 Extended Yale Face Database B (YALE)

The Yale Face Database and Extended Yale Face Database B (YALE) [123] were combined for a collection of 38 individuals and 2,424 total face images in PGM file format. Each subject has approximately 9 poses and 64 illumination conditions. None of the subjects wear glasses but subjects do vary by race, ethnicity, and gender. Figure 7-11 shows 65 sample faces images of one subject that vary by illumination. Notice how the subjects eyes changes as the illumination is varied. Figure 7-12 shows one face image sample of each of the 38 subjects used for evaluation.

Figure 7-11: 65 face images for one subject which vary by illumination.

Figure 7-12: One face image sample of each of the 38 subjects used from the Yale Extended B dataset.

7.3 Grassmann Similarity Measure Analysis

The next major contribution is the evaluation of Grassmann measures on all datasets to compare against Grassmann learning methods including GPCA, GLDA, GLPP, GSR, and GRASP. The purpose of this section is to identify the various Grassmann metrics that measure distances between Grassmann points and to identify the benefits and drawbacks.

Large Grassmann subspaces are expected to reduce the processing time since the span of these subspaces are represented as individual points on a Grassmann manifold. This characteristic is demonstrated in Figure 7-13 which shows the separation by principal angles between each action class for the i3DPost, IXMAS, WVU, and MSRAction3D datasets in a Grassmann space using the geodesic metric $d(\boldsymbol{Y}_i, \boldsymbol{Y}_j) = \|\boldsymbol{\theta}\|_2$.

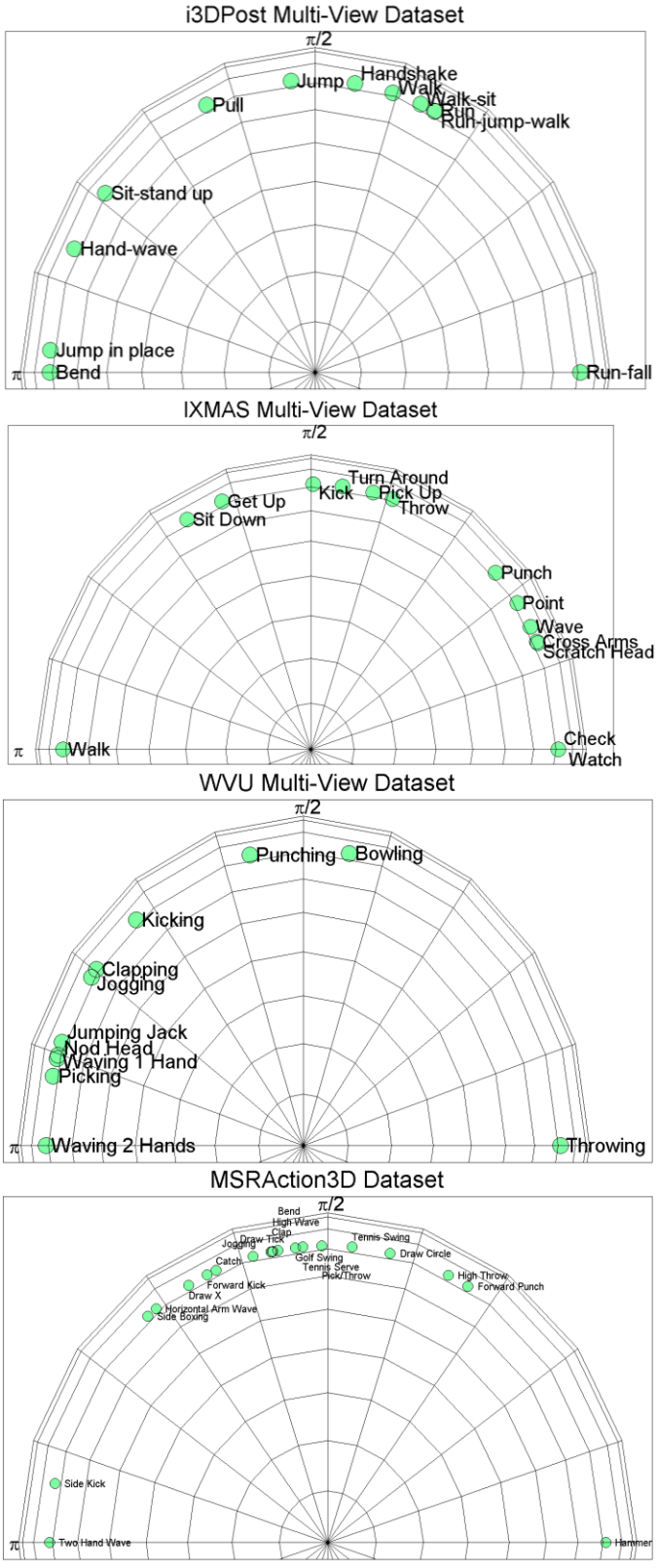


Figure 7-13: The principal angles in a Grassmann space between action classes for the i3DPost, IXMAS, WVU, and MSRAction3D datasets.

For all datasets, similar actions are correlated through Grassmann learning. For the i3DPost dataset relative to the *Run-fall* action, Grassmann learning identifies the actions *Walk*, *Walk-sit*, *Run-jump-walk*, and *Run* to be clustered and closest to *Run-fall*. *Run-fall* is farthest from *Bend*. Grassmann learning also promotes between class discrimination. It is apparent that mobility actions (those that involve movement across a scene) group together in the first quadrant while immobile actions group together in the second quadrant. For the IXMAS dataset relative to the *Check Watch* action, the actions *Scratch Head*, *Cross Arms*, and *Wave* are closest and are actions where the actor uses their arms. Actions *Punch* and *Point* are also closely correlated in a Grassmann space. The *Walk* action is clearly dissimilar from all the other actions. For the WVU dataset the action *Throwing* is closest to *Bowling* and *Punching* which are conceptually similar and farthest from *Waving 2 Hands*. For the MSRAction3D dataset *Hammer* is closest to *Forward Punch* and *High Throw* which are also very similar and farthest from *Two Handed Wave*.

Constraints of identifying closeness relationships through orthogonal mappings can also enforce unwanted relationships. For the i3DPost example, a *Hand wave* is considered closest to *Sit-stand up* and *Jump in place* which is not naturally correlated but is learned that way due to orthogonal constraints imposed on a Grassmann space for all actions relative to each other. Overall, Grassmann learning using the span of orthonormal matrices embedded as single points do show effort to promote high between-class discrimination and promote within-class clustering. This demonstrates the advantage of Grassmann learning in the GSR and GRASP frameworks for increasing the between class separability while decreasing the with-in class separability.

	Grassmann Measures	i3DPost	IXMAS	WVU	MSR Action3D	MSR Gesture3D	ATT	LFW	Yale Extended B
Single	Projection	91.28%	80.71%	62.86%	73.79%	88.10%	99.00%	55.42%	98.89%
	Binet-Cauchy	91.28%	80.71%	62.86%	73.79%	88.10%	99.00%	55.42%	98.89%
	Max Correlation	91.28%	80.71%	62.86%	73.79%	88.10%	99.00%	55.42%	98.89%
	Min Correlation	91.28%	80.71%	62.86%	73.79%	88.10%	99.00%	55.42%	98.89%
	Procrustes	91.28%	80.71%	62.86%	73.79%	88.10%	99.00%	55.42%	98.89%
	Geodesic	91.28%	80.71%	62.86%	73.79%	88.10%	99.00%	55.42%	98.89%
	Mean Distance	91.28%	80.71%	62.86%	73.79%	88.10%	99.00%	55.42%	98.89%
All	Projection	**94.79%**	**98.46%**	78.18%	74.23%	87.50%	**100.00%**	70.16%	**100.00%**
	Binet-Cauchy	93.75%	97.69%	78.18%	62.56%	77.50%	72.50%	67.74%	**100.00%**
	Max Correlation	84.38%	90.00%	67.27%	44.31%	67.50%	26.25%	39.52%	**100.00%**
	Min Correlation	**94.79%**	89.23%	72.72%	**78.06%**	**92.50%**	**100.00%**	**97.58%**	**100.00%**
	Procrustes	**94.79%**	**98.46%**	**80.00%**	70.25%	80.00%	**100.00%**	69.35%	**100.00%**
	Geodesic	93.75%	**98.46%**	**80.00%**	69.63%	78.33%	**100.00%**	**97.58%**	**100.00%**
	Mean Distance	**94.79%**	**98.46%**	78.18%	74.23%	87.50%	**100.00%**	68.55%	**100.00%**

Table 6: The Grassmann similarity measures between subspaces on a Grassmann manifold. The first group shows similarity measures where each test action sample is a unique point on a Grassmann manifold and there is one principal angle between each subspace. The second group shows the similarity measures when test samples of the same class are grouped and represented as a single point on a Grassmann space. There are multiple principal angles between subspaces for the second group.

Table 6 shows the Grassmann similarity measures using the techniques outlined in Equations (34) through (40). All training inputs of the same action class represent a single point on a Grassmann space. This means that the number of training subspaces m_{train} is equal to the number of action classes p. For testing, two separate experiments were run. In the first case, each test sample is treated as a single point on a Grassmann space and labelled as "Single". This means that the number of testing subspaces m_{test} is equal to the number of test samples n, ($m_{test} = n$). In the second experiment, all test inputs of the same class were grouped into one subspace and labelled as "All" ($m_{test} \ll n$). This is ideal for systems where multiple test samples are classified simultaneously, such as multiple views, multiple trials of an unknown action, or multiple face images of a single unknown subject.

Given Equations (34) through (40) for all experiments in the "Single" setup, there is exactly one principal angle since each subspace $\boldsymbol{Y}_{D\times q}$ has only one test sample ($q = 1$). With exactly

one principal angle between subspaces, all Grassmann measures will have equivalent classification results. The ATT and Yale Extended B dataset have the highest classification accuracies. The ATT dataset is fairly clean with 10 similar face images per subject for 40 subjects. The Yale Extended B dataset has a high amount of face images per subject and only 38 subjects. The LTP descriptor for face recognition is clearly capable of representing face images in a discriminative manner. The LFW dataset is more challenging because of the uncontrolled environment for capturing the face images off of the web with a larger amount of test subjects. The WVU dataset has the lowest classification accuracies for the action datasets which is attributed to the high levels of noise due to lighting inconsistencies and multi-view synchronization issues.

For the "All" setup, classification accuracies increase in comparison to the "Single" setup. This indicates that Grassmann learning does fill in missing data through linear spanning and is more robust when the number of points on a Grassmann manifold is small and the number of samples representing those points is large. Metrics that classify well on the i3DPost dataset utilize all principal angles and a similar pattern emerges with the IXMAS and WVU datasets indicating that the metrics have a good balance of robustness to noise and class clustering. The minimum correlation metric performs best on the MSRAction3D, MSRGesture3D, and LFW datasets indicating that the largest principal angle is the most effective for classification. This means that the input data of these datasets are highly clustered. The Yale Extended B dataset classifies perfectly independent of the Grassmann measure being evaluated. This is attributed to the large Grassmann subspaces that represent a single point on a Grassmann space. The larger the subspaces become, the more discriminative the distances between other classes. The ATT

dataset classifies the worse for the max correlation metric. This indicates that the distribution of LTP image data are highly discriminative.

This evaluation identifies that not one specific Grassmann measure is ideal for all datasets. For example, the minimum correlation measure is ideal for the i3DPost, MSRAction3D, MSRGesture3D, and LFW datasets. However, the minimum correlation measure is also the worse classifier for the IXMAS dataset. Grassmann measures are dependent on the distribution of the high dimensional data [82]. Kernels can be standardized and therefore kernelization provides a way of avoiding functions based on principal angles that are dependent on data distributions and can be processed using kernel-based methods [89]. This is a motivation to apply kernelization and evaluate GSR and GRASP.

7.4 Grassmann Kernel Standardization

As previously mentions, data distributions affect the classification accuracies of geodesic metrics in a Grassmann space. The next contribution in this dissertation is the proposal and justification Grassmann kernel standardization to ignore variations between individual Grassmann points when subspace sizes vary. Assume Grassmann kernels follow a Gaussian distribution $f(\boldsymbol{x}) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$ as shown in the red curve of Table 7. A kernel would have a non-zero mean μ and a non-unitary standard deviation σ. If a kernel follows a standard normal distribution $f(\boldsymbol{x}) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$ as shown in the blue curve, data would be centered with $\mu = 0$ and $\sigma = 1$.

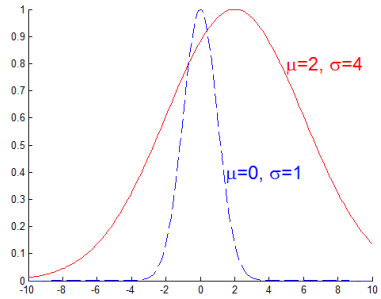
	No Normalization	$f(\boldsymbol{k}) = \boldsymbol{k}$
	Zero mean	$f(\boldsymbol{k}) = \boldsymbol{k} - \mu$
	Zero mean and unit variance	$f(\boldsymbol{k}) = \dfrac{\boldsymbol{k} - \mu}{\sigma}$

Table 7: Centered and Standard Normal Distributions

In Figure 7-14 the 3D embeddings of four 3D shapes with corresponding 2D embeddings of zero mean is shown. The result is the centered distribution of the data.

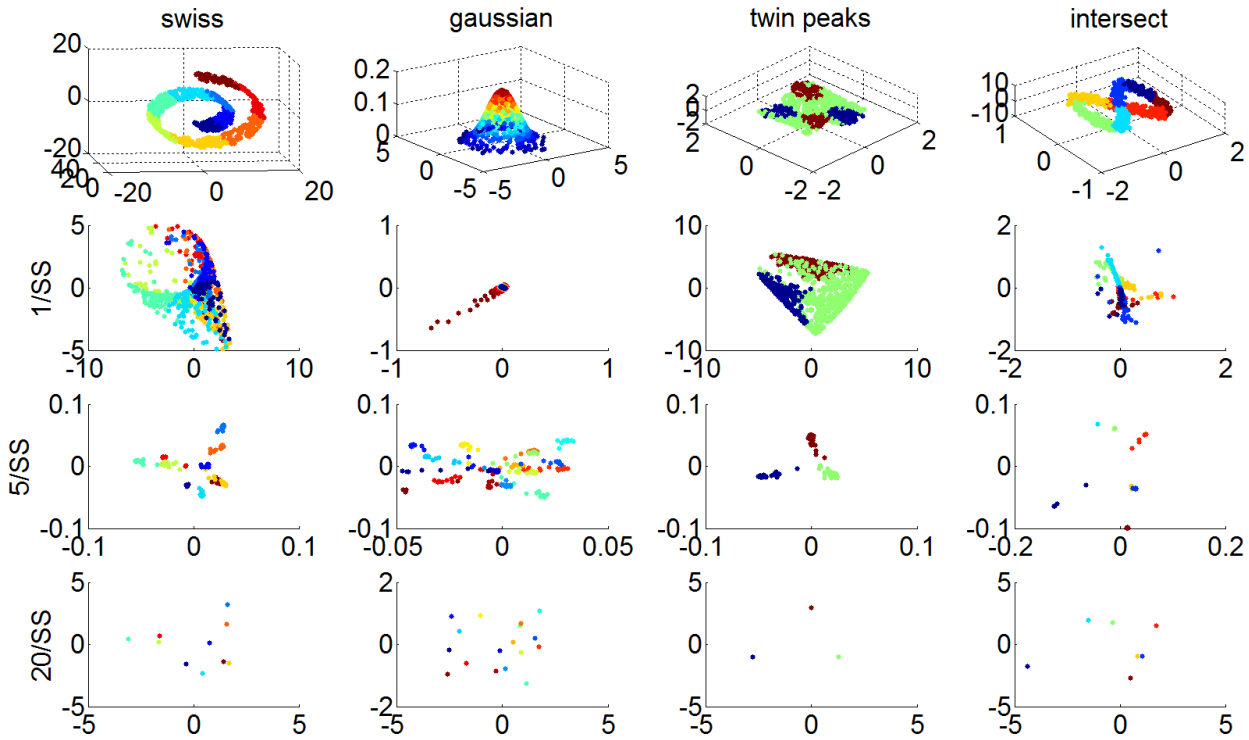


Figure 7-14: GRASP embeddings after normalizing the Grassmann kernels ($\mu = 0$)

In the Figure 7-15 the identical distribution after centering the Grassmann kernels and dividing by the standard deviation to standardize the kernel are shown. The embedding look identical except the embedding are centered and scaled.

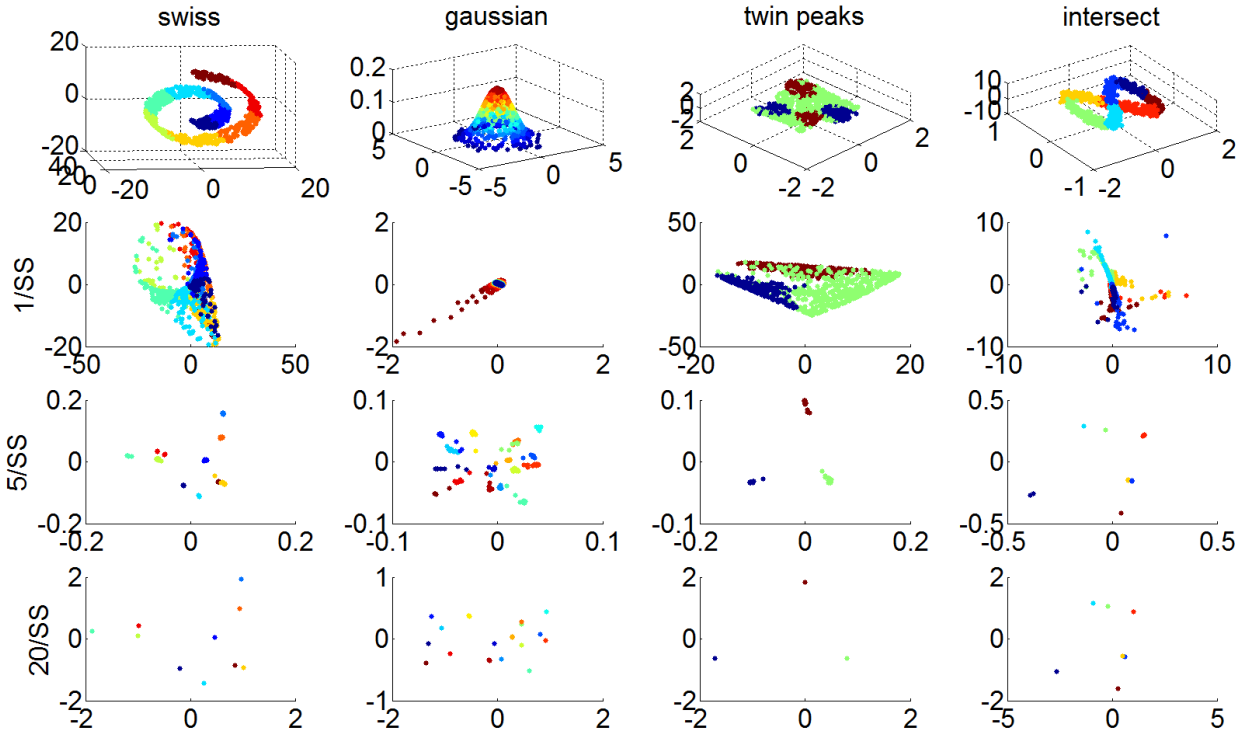


Figure 7-15: GRASP embedding after normalizing the Grassmann kernels and $(\mu = 0, \sigma = 1)$

Kernel standardization on GRASP and GSR allows for learning algorithms such as spectral regression and sparse representations to be effective while ignoring Grassmann point distribution variations. Figure 7-16 and Figure 7-17 presents results on the impact of kernel normalization on GRASP and GSR and the classification accuracies when varying training and testing subspace sizes evaluated on the multi-view datasets. The patterns for GRASP and GSR are consistent for all evaluated datasets. When centering the Grassmann kernels without dividing by the standard deviation, the best classification accuracy is achieved when maintaining a consistent subspace

size for the training and testing kernels. Meanwhile, as the subspace sizes vary between the training and testing kernels, a significant drop in classification accuracy is observed. For example, the IXMAS dataset has a classification accuracy of 90.85% when the subspace sizes are equal to 3. However, when the training subspace size is set to 110 and testing subspace size is set to 3, the classification accuracy drops down to 7.69%. This setup is suitable for applications such as multi-view surveillance systems or systems where there are identical subspace sizes for training and testing. When centering the Grassmann kernels while dividing by the standard deviation, the best classification accuracies are obtained when the training subspace sizes are large. Results for the i3DPost dataset show an 86.72% classification accuracy when training subspace sizes are 56 samples per subspace while testing subspace sizes are one sample per subspace.

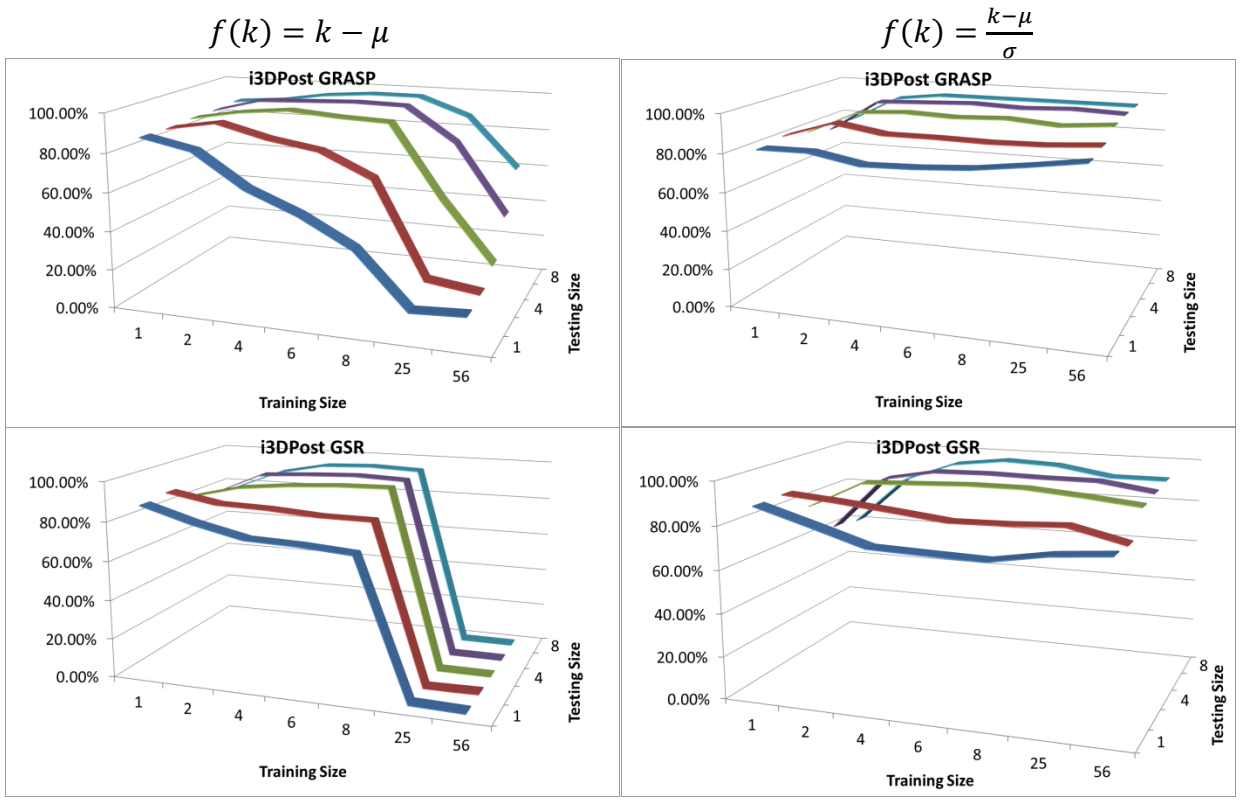


Figure 7-16: i3DPost GRASP (top row) and GSR (bottom row) classification accuracies without (left) and with (right) kernel standardization.

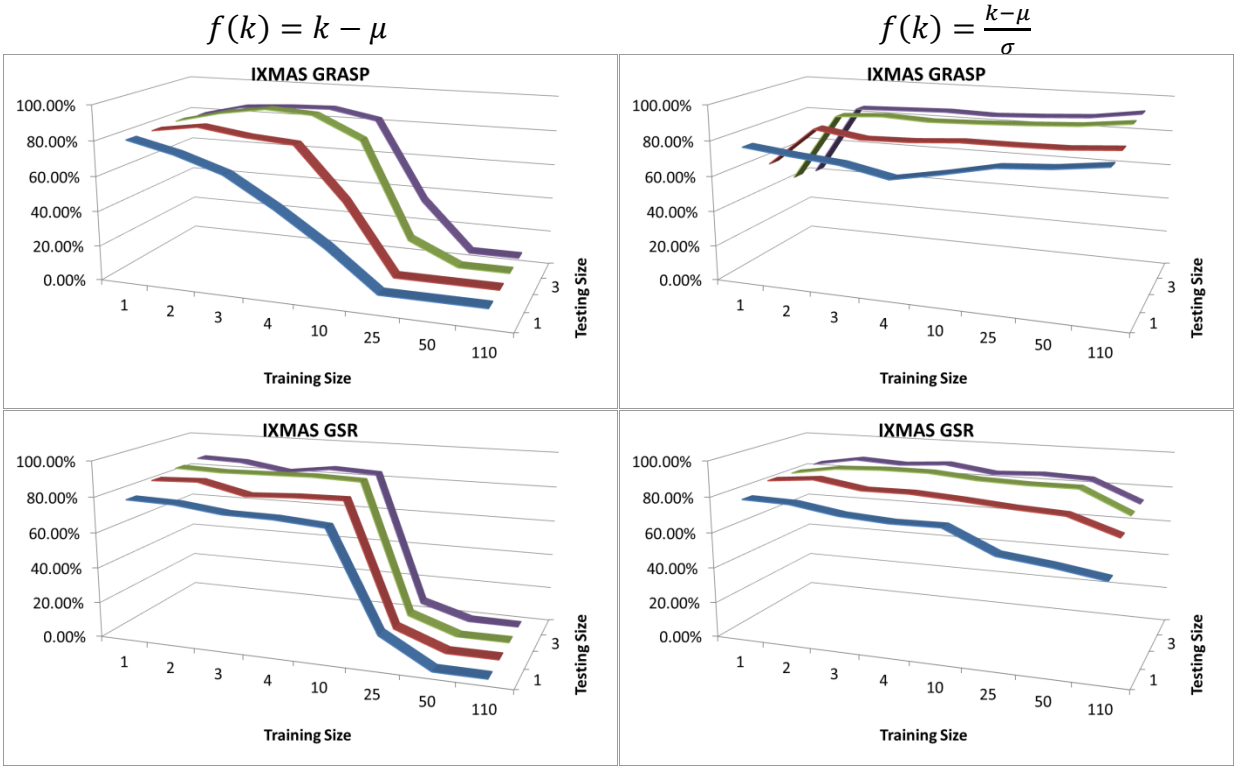


Figure 7-17: IXMAS GRASP (top row) and GSR (bottom row) classification accuracies without (left) and with (right) kernel standardization.

This interesting observation suggests that Grassmann methods do not need to be restricted to fixed subspace sizes if Grassmann kernels are standardized. This also supports the motivation for using Grassmann kernel based manifold learning over Grassmann metrics in a Grassmann space. Equations (34) through (39) presented Grassmann metrics and each metric has their benefits and drawbacks based on the level of noise and the data distribution. The utilization of Grassmann kernels which can be standardized overcomes the Grassmann metric dependencies on noise and distributions. The manipulation of Grassmann kernels in this manner would be ideal for applications where test samples can vary such as single view surveillance systems or 3D action classification while maintaining very large training subspace sizes.

7.5 Sparse Representation Analysis

Figure 7-18 shows the sparse coefficients and corresponding residuals determined through sparse representations on the i3DPost, MSRAction3D, and LFW datasets for the classification of one test sample.

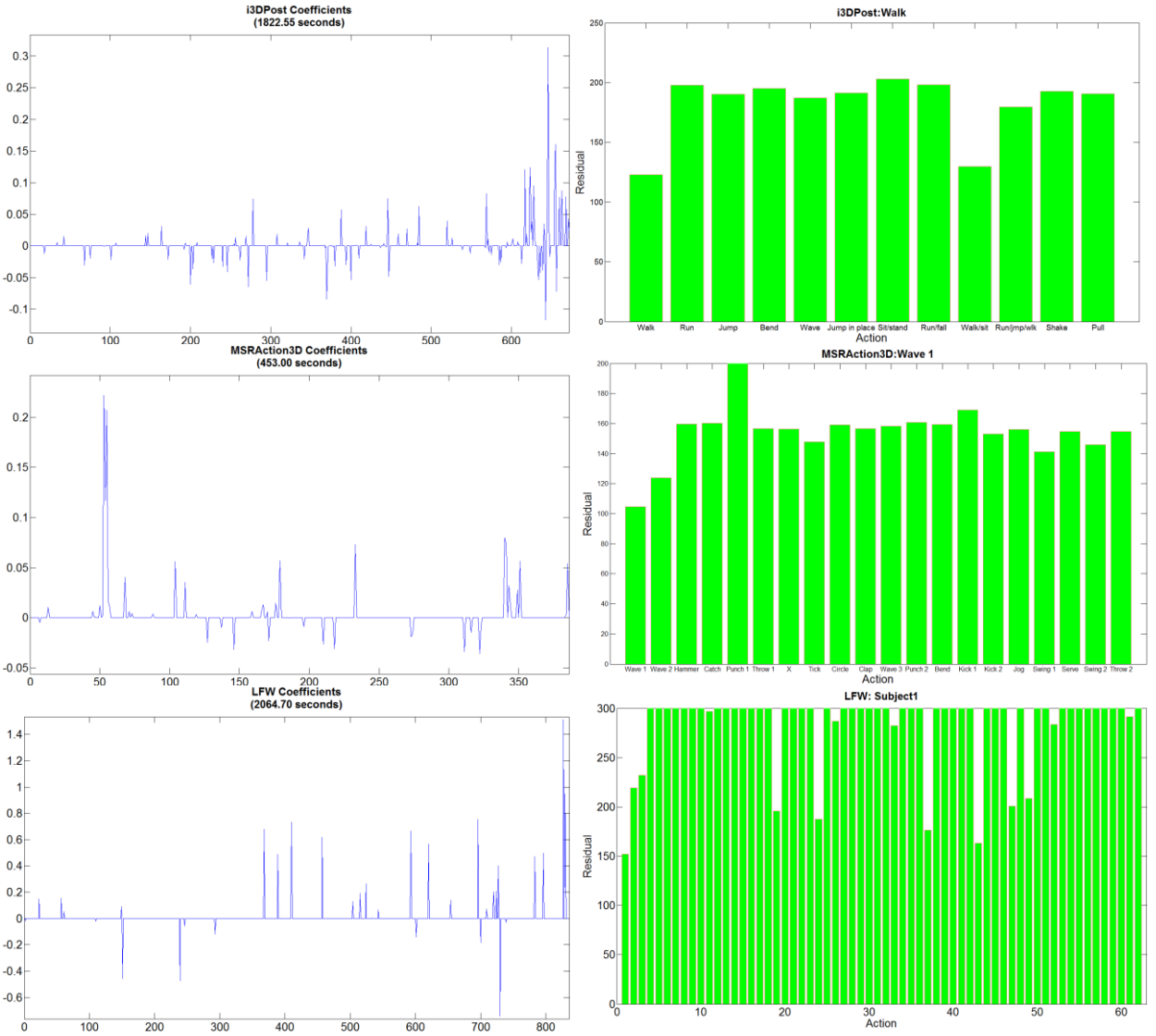


Figure 7-18: The coefficient vectors and corresponding residuals determined through Sparse Representations for one test sample from the i3DPost, MSRAction3D, and LFW datasets. The test samples were *Walk* for i3DPost, *Wave 1* for MSRAction3D, and *Subject 1* for LFW.

The non-zero coefficients from the sparse coefficient vector were used to reconstruct a sample from each class in the dictionary. The residuals are calculated using Equation (49) . For each evaluation the first action class or face image was used for testing. This means the smallest residual is expected to be the first class from each dataset.

Figure 7-18 shows that all classes were correctly classified since the smallest residual from each bar chart is the first class. For i3DPost, *Walk* has the smallest residual from all the action classes with *Walk/Sit* trailing as the second smallest. We also observe that the largest residual is *Sit/Stand* which can identify the most orthogonal class to the *Walk* action. Similar patterns can be observed with the MSRAction3D dataset. *Wave 1* which is a high arm wave was correctly classified and the second trailing action is *Wave 2* which is a horizontal arm wave. Sparse representations identify that the most orthogonal action to the high arm wave is *Punch 1* which is a forward punch. For the LFW dataset, the residuals identify subject 1 to be correctly classified. The trailing subject was subject 43 and the most orthogonal subject was subject 62. Figure 7-19 shows the face images corresponding to the subject identifiers. The face on the left is the test image. The face image in green to the right shows that the test image was correctly classified to the right subject. Subject 43 is the second most similar subject to subject 1. Subject 62 in red is the most different subject to the test subject.

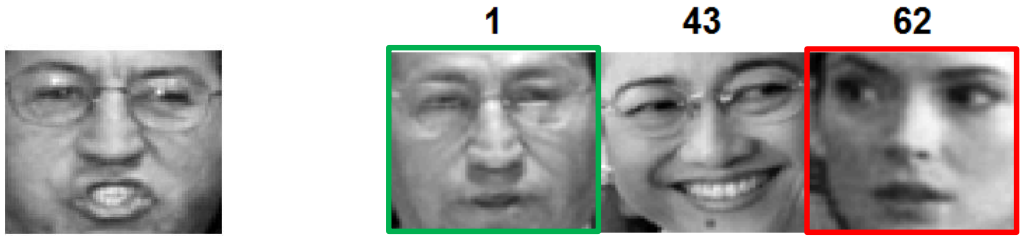


Figure 7-19: The face image test sample is shown on the left. Sparse representations identify subject 1 to be the most similar, subject 43 to be the second most similar and subject 62 to be the most different.

The residuals for each evaluation are very good at identifying classes and orthogonal classes. However, the processing times needed to obtain the coefficient vectors and apply minimum reconstruction are extremely slow as indicated by Figure 7-18. This is because the coefficient vector sizes are equal to the number of test inputs for each experiment and with exponential time complexities the processing time is extremely high.

7.6 Grassmannian Sparse Representation Analysis

Figure 7-20 shows the sparse coefficients and corresponding classification times for one subject in the same datasets of Section 7.5 using GSR. All samples of a single class are represented as a single point on a Grassmann space, resulting in coefficient vectors of a size equal to the number of classes. GSR eliminates the need for additional mapping between a large coefficient vector to its corresponding action class and as a result the minimum reconstruction method is simplified. Although still not ideal for real-time performance, the performance and classification accuracies have considerably improved when comparing to sparse representations on high dimensional data. We see that the i3DPost evaluation took 1822.55 seconds to process using sparse representations and only 0.56 seconds to process through GSR. Similar speed ups can be observed on the remaining datasets.

Another observation is the residuals determined through GSR. The residuals for each evaluation are more apparent for identifying a class and are stronger indicators of the most suitable class. The remaining classes have similar and higher residuals. This is because the Grassmann learning component of GSR has managed to find orthonormal mappings that promote within class clustering and between-class discrimination. This demonstrates that GSR is capable of reducing coefficient vectors while maintaining high classification accuracy.

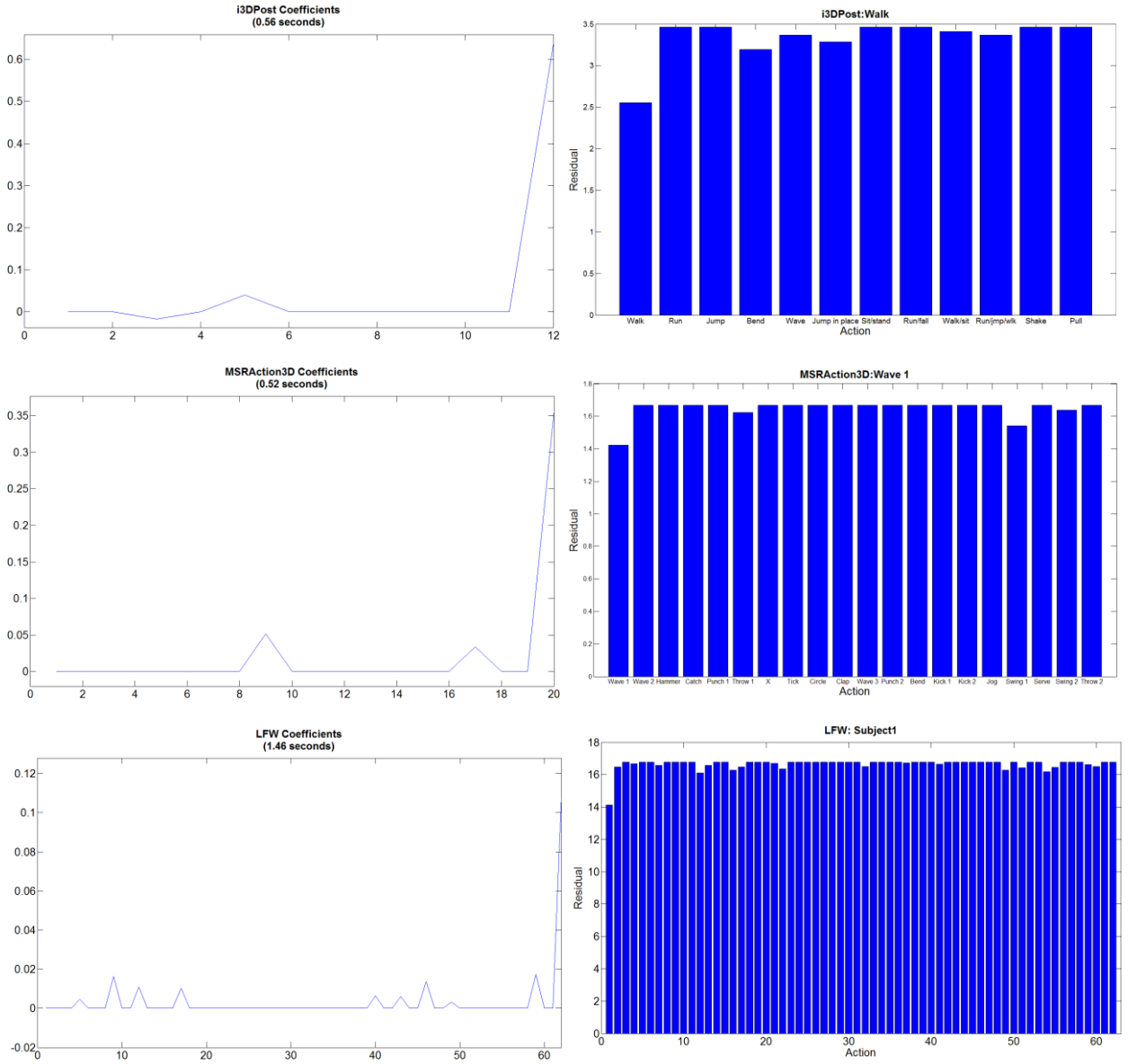


Figure 7-20: The calculated coefficient vector representations and classification times through GSR for one subject of each action dataset and one fold of the LFW face dataset. The coefficient vector sizes have reduced down to the number of classes in comparison to sparse representation classification. The datasets analyzed are the i3DPost, MSRAction3D, and LFW datasets.

7.7 Classification and Performance Results and Analysis

Experiments based on Euclidean ℓ-2 norm, PCA, LDA, LPP, NPE, and Spectral Regression were classified using k-NN with k=3. Combinations of Grassmann kernel methods including GPCA

[84], GLDA [82], GLPP [79] [81], and GRASP [116] were classified using k-NN with k=1, since each Grassmann point represents a single class. Sparse representations and GSR [117] were classified using minimum reconstruction. Computational processing times for each algorithm were also captured. All action experiments were based on leave one subject out cross validation (LOOCV). All face recognition experiments were evaluated using 2-fold cross validation (2FCV). In all experiments, all inputs from each database were used unless otherwise noted in the descriptions of the datasets.

The classification accuracy results shown in Table 8 and Figure 7-21 show experimental classification results using various algorithms on the three multi-view datasets, two 3D datasets, and three face datasets. The "Single" and "All" references identify whether test samples for m_{test} were treated as single subspaces ($m_{test} = 1$) or subspaces composed of all available test samples associated with a class. The latter is ideal for systems where multiple test samples are classified simultaneously, such as multiple views or multiple trials of an unknown action, or multiple samples of the sane unknown face.

	Method	i3DPost	IXMAS	WVU	MSR Action3D	MSR Gesture3D	ATT	LFW	Yale Extended B
	k-NN (k=3)	79.30%	63.22%	40.87%	65.93%	74.11%	92.00%	31.54%	94.16%
	PCA	79.30%	66.17%	42.94%	66.12%	74.11%	92.00%	31.54%	94.16%
	LDA	78.78%	57.91%	38.54%	76.00%	82.14%	**99.50%**	43.20%	92.98%
	LPP	**80.73%**	64.72%	41.67%	76.91%	78.87%	**99.50%**	38.65%	99.14%
	NPE	73.96%	63.76%	42.10%	77.03%	81.55%	99.00%	21.78%	96.63%
	Sparse Rep.	79.56%	**74.28%**	**48.41%**	**79.54%**	**83.33%**	99.25%	**69.53%**	**99.35%**
	Spectral Reg.	77.99%	57.71%	38.82%	77.00%	80.06%	99.00%	43.97%	99.02%
Single	GPCA	90.36%	44.94%	63.14%	76.07%	87.50%	**99.00%**	55.09%	98.85%
Single	GLDA	90.76%	78.01%	62.66%	53.70%	86.90%	**99.00%**	31.93%	98.76%
Single	GLPP	90.36%	79.38%	63.14%	76.07%	87.80%	**99.00%**	55.09%	98.85%
Single	GSR	**91.41%**	79.19%	60.29%	**76.17%**	87.20%	**99.00%**	53.92%	98.85%
Single	GRASP	90.49%	**80.76%**	**63.76%**	75.04%	**88.39%**	**99.00%**	**57.53%**	**98.89%**
All	GPCA	92.71%	77.69%	80.00%	72.54%	85.83%	**100.00%**	66.94%	**100.00%**
All	GLDA	90.63%	90.77%	80.00%	57.70%	85.83%	**100.00%**	58.87%	**100.00%**
All	GLPP	92.71%	93.08%	80.00%	72.54%	85.83%	**100.00%**	67.74%	**100.00%**
All	GSR	**95.83%**	96.92%	**90.91%**	**77.17%**	**87.50%**	**100.00%**	**96.77%**	**100.00%**
All	GRASP	94.79%	**97.69%**	81.82%	75.13%	**87.50%**	**100.00%**	83.87%	**100.00%**

Table 8: The classification accuracies for various algorithms evaluated on multi-view action datasets, 3D action datasets, and face recognition datasets.

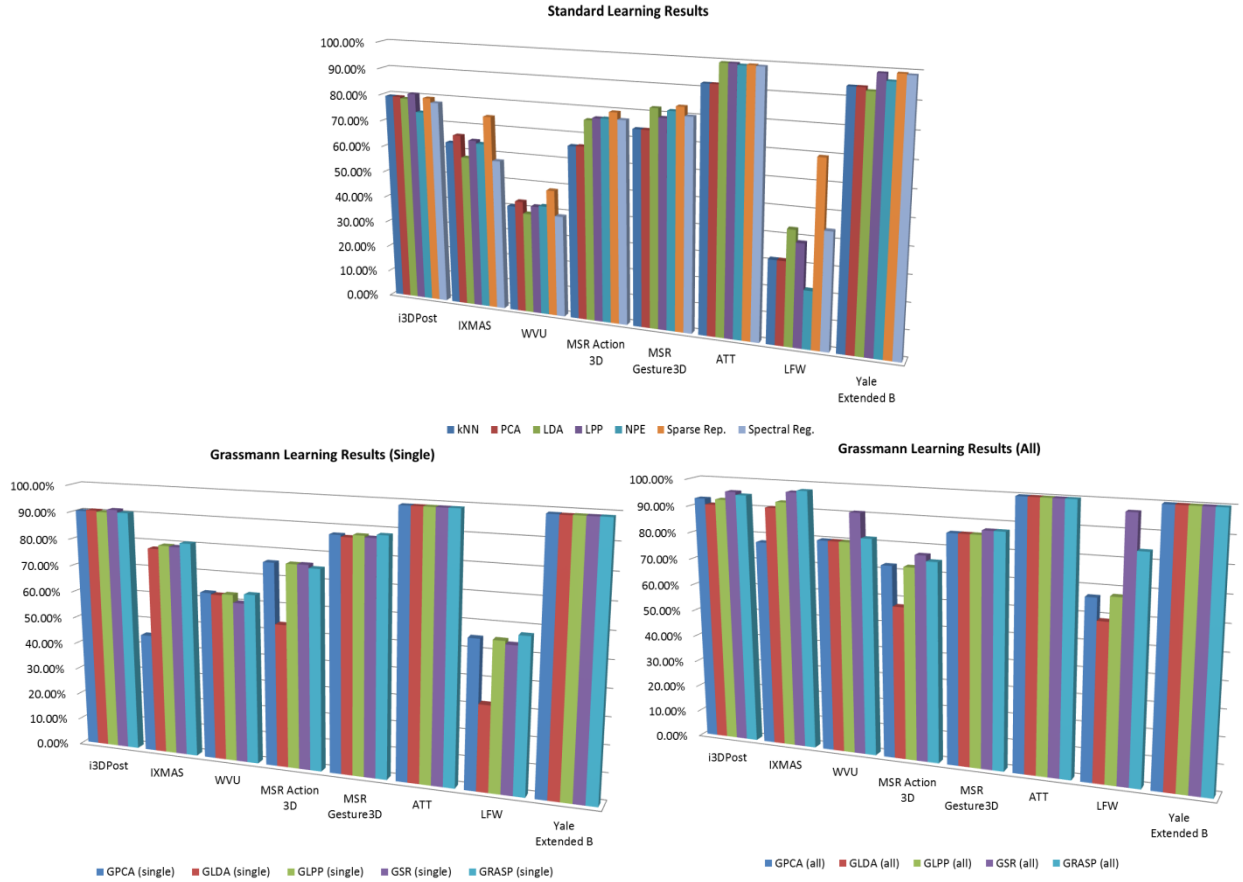


Figure 7-21: Classification charts for standard learning, Grassmann learning with single test inputs, and Grassmann learning with all test inputs of one class.

In Table 8, the best classification accuracies for each dataset are highlighted in bold for test subspaces representing a single action input (labeled Single) and for an entire set of actions (labeled All). Visualization of these results is shown in Figure 7-21. The IXMAS, WVU, and LFW datasets are clearly the most challenging datasets to evaluate. The IXMAS dataset is challenging because there are more classes than subjects. The WVU dataset is challenging for the same reason and because of high levels of noise due to lighting inconsistencies and multi-view synchronization issues. The LFW dataset is challenging because of the high amount of unconstrained face images collected from different sources off the web. The results from non-Grassmann based methods show that sparse representations are the most effective for high

classification results with GSR accurately classifying 96.77% on the LFW dataset. This is a 12.9% lead over GRASP and a 29.03% leaver over GLPP.

The Grassmann based algorithms are much better at classifying actions and recognizing faces than the non-Grassmann algorithms, with GRASP and GSR performing at or near the top for the single and all cases. All Grassmann based methods have very similar classification accuracies when test subspaces are represented as single inputs but there are cases when certain methods classify poorly. GPCA is a poor learning method for the IXMAS dataset while GLDA is a poor learning method for the MSRAction3D and LFW datasets. When test subspaces represent an entire class as a point on a Grassmann space, GRASP and GSR have an advantage over GPCA and the graph embedding frameworks. For the less challenging datasets including ATT and Yale Extended B, all standard learning and Grassmann methods classify extremely well. GRASP has a slight edge over GSR on the IXMAS dataset. GRASP and GSR classify the same on the MSRGesture3D dataset. However, GSR has shown to have the best classification accuracy for the most challenging datasets including WVU and LFW.

Figure 7-22 and Figure 7-23 shows the confusion matrices with single test subspaces and all test samples of one class in a subspace. The GSR classification results are shown on the i3DPost dataset and the GRASP classification results are shown on the IXMAS dataset. The confusion matrices identify high within-class clustering and high between class discrimination. For the GSR results in the "single" case, we see acceptable levels of errors. For example, *Walk* is misclassified 11 times with *Walk-sit*. *Run-jump-walk* is misclassified 8 times with *Walk*. When a Grassmann point represents an entire test class in the "all" case, classification errors are minimal. Factoring in that this system is classifying actions from multiple views proves that GSR is robust, efficient, and accurate. Similar patterns are noticeable on the i3DPost dataset

using GRASP. In the "single" case, *Scratch Head* is confused 19 times with *Wave*. *Punch* is confused 15 times with *Point*. In the "all" case classification errors are minimized.

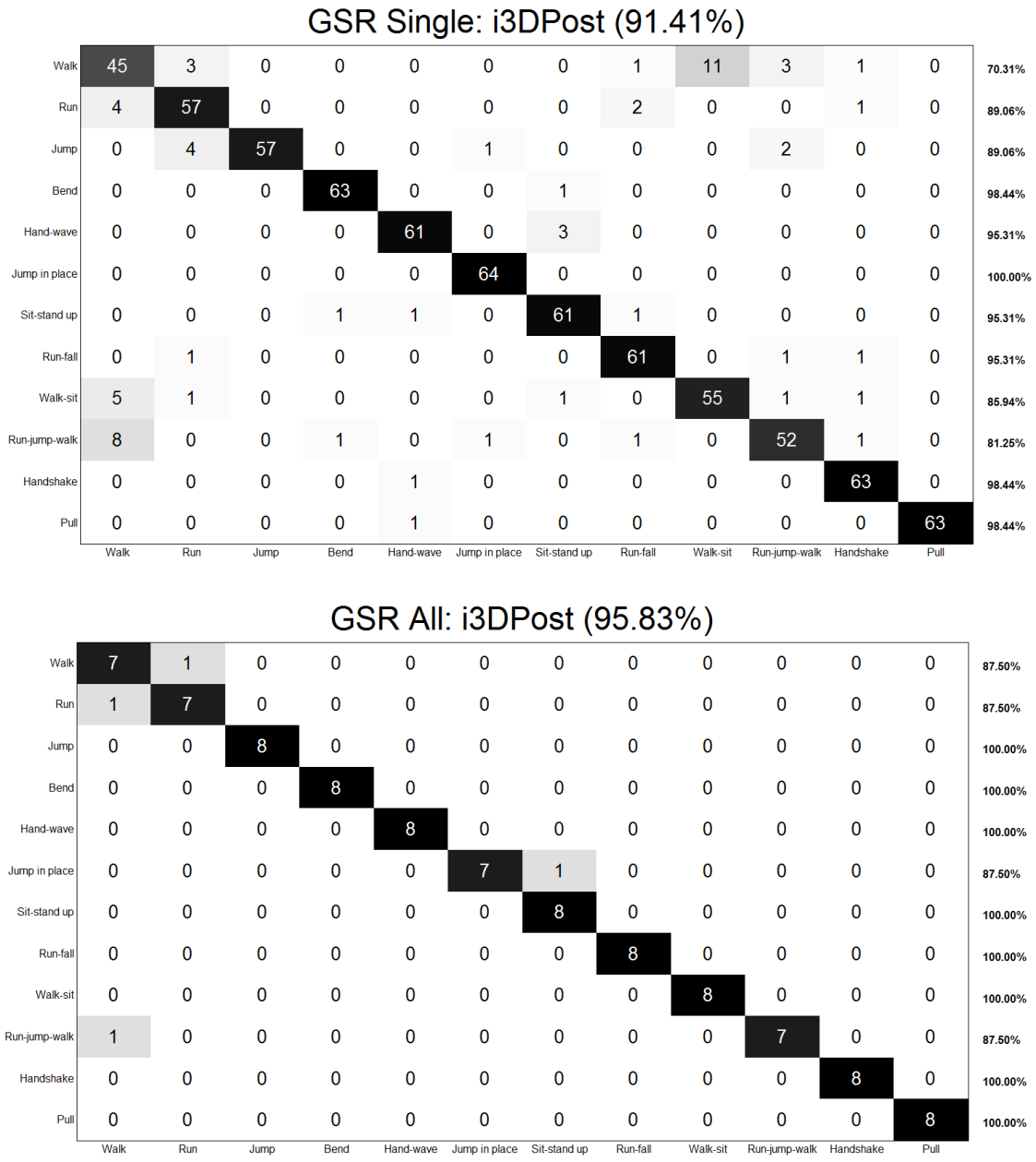


Figure 7-22: Confusion matrices that show the classifications made through GSR on the i3DPost dataset. The classifications were made with single test subspaces (top) and all test elements of one class in one single subspace (bottom).

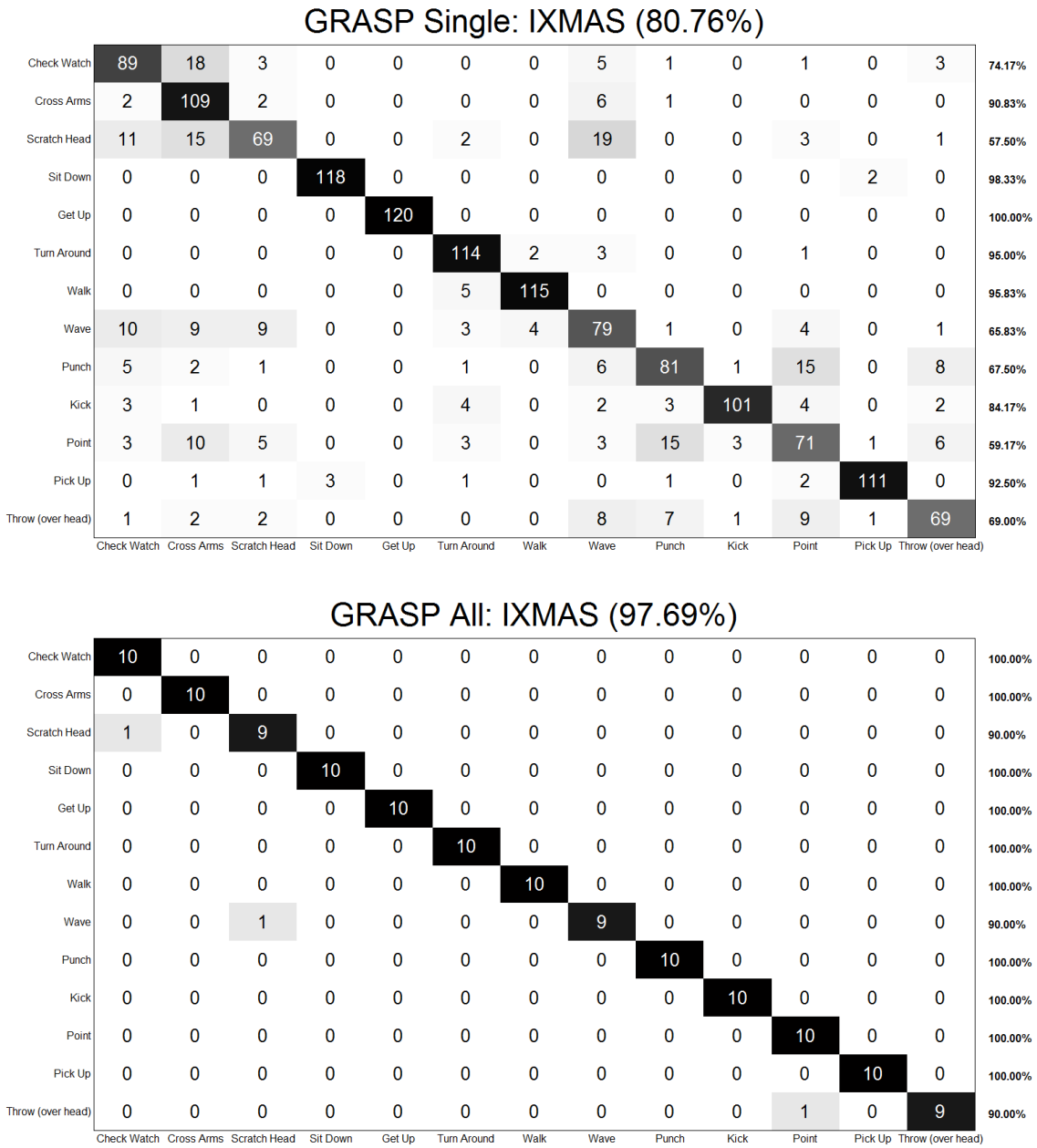


Figure 7-23: Confusion matrices that show the classifications made through GRASP on the XIMAS dataset. The classifications were made with single test subspaces (top) and all test elements of one class in one single subspace (bottom).

Table 9 shows the execution performance times for all algorithms on all datasets. The processing times in green identify the fastest processing times for each group of evaluations. The processing times in red identify the slowest processing times for each group of evaluations. Although sparse representations with minimum reconstruction are very good for classification

they are also the slowest performance with estimated exponential complexities. When sparse representations are applied in a Grassmann framework through GSR, classification times were drastically improved to exceed standard classification methods. The drastic classification improvement can be attributed to the sizes of the sparse coefficient vectors. Sparse representations on individual test samples mean larger sparse coefficient vectors. The larger the coefficient vectors, the more likely classification errors can be made due to the extreme high dimensions of input samples and variability between inputs. Through Grassmann learning entire classes can be embedded as single points in a Grassmann space. This promotes within class clustering and between class discrimination. When combined with the sparse representation framework, classification accuracy and performance is improved.

Classification results through GSR are still not fast enough for real-time applications. Figure 7-24 shows the performance charts for various Grassmann learning methods where GSR is clearly slower. GRASP was proposed as a fast performing classification framework to overcome the performance drawbacks of GSR. Spectral regression frameworks tend to be the fastest by avoiding eigen-decomposition. Manifold learning with Grassmann frameworks show considerable improved processing times compared to standard methods. This is because the points embedded on a Grassmann manifold represent trained action subspaces rather than individual training samples. In both the single element subspaces and all element subspaces, GRASP has a slight edge for computational performance over graph embedding frameworks. However, because graph based learning and spectral regression are being applied on Grassmann kernels which have already reduced the high dimensional data of the original inputs, the computational advantage of GRASP is relatively small. A more significant improvement in performance can be observed as the number of classes increase.

	Method	i3DPost (sec)	IXMAS (SEC)	WVU (sec)	MSR Action3D (sec)	MSR Gesture3D (sec)	ATT (sec)	LFW (sec)	Yale Extended B (sec)
	k-NN (k=3)	407.59	1440.09	3585.82	216.88	78.45	3.63	603.41	448.02
	PCA	93.13	266.71	133.05	37.39	25.58	0.53	245.48	393.30
	LDA	20.61	49.05	27.09	10.83	6.21	0.29	6.34	22.91
	LPP	21.65	50.28	26.38	9.99	6.83	0.33	5.95	24.55
	NPE	23.29	52.66	19.07	8.66	6.48	0.38	6.19	21.89
	Sparse Rep.	20015.38	22573.86	22336.66	4024.79	2300.78	70.57	5188.19	2313.62
	Spectral Reg.	11.62	19.22	16.74	6.17	4.03	0.25	4.12	17.54
Single	GPCA	0.14	0.29	0.41	0.16	0.17	0.31	0.90	1.11
Single	GLDA	0.13	0.35	0.31	0.16	0.09	0.08	0.69	1.24
Single	GLPP	0.13	0.33	0.32	0.16	0.09	0.08	0.54	1.57
Single	GSR	10.33	60.95	17.86	26.52	3.74	3.94	103.47	52.96
Single	GRASP	0.13	0.24	0.30	0.09	0.06	0.08	0.88	1.22
All	GPCA	0.03	0.07	0.02	0.07	0.04	0.06	0.10	0.06
All	GLDA	0.03	0.06	0.04	0.09	0.04	0.02	0.07	0.03
All	GLPP	0.03	0.05	0.06	0.10	0.04	0.02	0.05	0.13
All	GSR	3.01	8.64	3.05	10.27	1.51	2.24	15.24	4.83
All	GRASP	0.02	0.03	0.01	0.04	0.03	0.02	0.07	0.05

Table 9: The classification performance in seconds for various algorithms evaluated on multi-view action datasets, 3D action datasets, and face datasets.

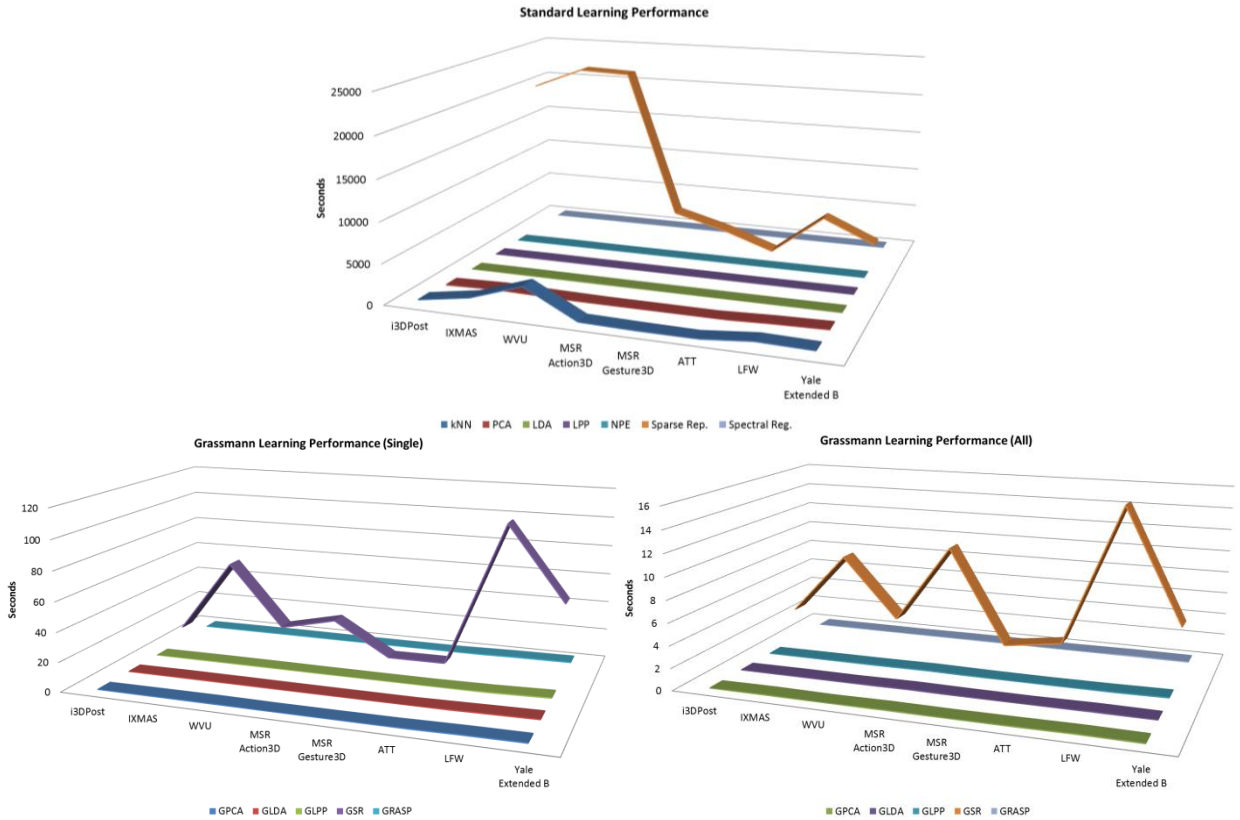


Figure 7-24: Performance charts for Grassmann learning with single test inputs, and Grassmann learning with all test inputs of one class.

When comparing GSR and GRASP against each Grassmann metric in Table 6, we see that GSR and GRASP are consistent in meeting classification accuracies depending on the distribution of the data in the datasets. For example, when comparing the minimum correlation metric results against GSR, we see that IXMAS classification through GSR has improved by 7.69% while meeting or slightly trailing in the other datasets. Meanwhile metrics that classify well on the IXMAS dataset classify extremely poor on the LFW dataset. Kernelization through GSR and GRASP eliminate the dependency on the data distributions by projecting Grassmann points onto a projective space where kernel standardization can be applied. Overall, Grassmann measures are ideal when data distribution and noise levels are known while GSR is ideal when data distribution and noise levels are unknown.

7.8 Comparison to State-Of-The-Art Methods

In this section we compare the classification accuracies of GSR and GRASP against state-of-the-art methods on the i3DPost, IXMAS, MSRAction3D, MSRGesture3D, ATT, and YALE datasets. The WVU dataset has not been thoroughly evaluated by many state-of-the-art methods and the few papers that have evaluated this dataset have not provided sufficient information regarding experimental setup to allow for a direct comparison. The intended use of LFW, as presented by Huang et al. [122], is for evaluating the matching of face pairs. Given a pair of face images, methods which use LFW output match probabilities rather than hard decisions. GSR and GRASP require training sets and could not be accurately compared to the LFW methodology.

GSR and GRASP results are presented in the "Single" and "All" Grassmann subspace configurations. The most comparable configuration to the state-of-the-art methods is the "Single" Grassmann subspace configuration because individual test samples are compared

against a trained set rather than groups of unknown test samples of the same class being compared against a trained set. All results identified in bold indicate the top performing methods excluding results presented for the "All" Grassmann subspace configurations which are expected to classify better than the "Single" Grassmann subspace configuration.

7.8.1 i3DPost Multi-View Human Action Dataset (i3DPost)

For the i3DPost dataset, all comparisons are based on LOOCV. Gkalelis et al. [27] introduced the i3DPost dataset and applied fuzzy vector quantization with linear discriminant analysis for human movement recognition and report their results when classifying five actions. Iosifidis et al. [30] used fuzzy vector quantization with artificial neural networks and fuzzy vector quantization with linear discriminant analysis [29] for action recognition evaluated on eight actions. Azary and Savakis [50] used sparse representations on motion history surfaces with a minimum reconstruction residual classifier and ran experiments on variations of action subsets. Holte et al. [33] used view-invariant 3D motion based vector fields from 3D Motion Context (3D-MC) and the Harmonic Motion Context (HMC) as action representations. Karali and ElHelw [124] combine motion history of skeleton volumes and temporal change in bounding volume utilizing logistic model trees, Mahalanobis distances, and linear discriminant analysis.

Action Subset Label	Action List
All Actions	*Walk, Run, Jump, Bend, Hand-wave, Jump in place, Sit-stand up, Run-fall, Walk-sit, Run-jump-walk, Handshake, Pull*
11 Actions	*Walk, Run, Jump, Bend, Hand-wave, Jump in place, Sit-stand up, Run-fall, Walk-sit, Run-jump-walk, Handshake*
10 Actions	*Walk, Run, Jump, Bend, Hand-wave, Jump in place, Sit-stand up, Run-fall, Walk-sit, Run-jump-walk*
8 Actions	*Walk, Run, Jump, Bend, Hand-wave, Jump in place, Sit-stand up, Run-fall*
6 Actions	*Walk, Run, Jump, Bend, Hand-wave, Jump in place*
5 Actions (E1)	*Walk, Run, Jump, Bend, Jump in place*
5 Actions (E2)	*Walk, Jump, Bend, Hand-wave, Jump in place*

Table 10: Action subsets for the i3DPost dataset as reported in the works of [27], [29], [30], [33], [50], and [124].

Method	All Actions	11 Actions	10 Actions	8 Actions	6 Actions	5 Actions (E1)	5 Actions (E2)
GSR (Single)	**91.41%**	88.92%	87.50%	91.60%	92.71%	92.50%	97.19%
GRASP (Single)	90.49%	**90.06%**	**90.47%**	92.97%	**93.49%**	**93.13%**	96.25%
GSR (All)	**95.83%**	89.77%	88.75%	**92.19%**	**95.83%**	95.00%	**100%**
GRASP (All)	94.79%	**96.59%**	**96.25%**	90.63%	**95.83%**	**97.50%**	**100%**
Gkalelis et al. [27]						90.00%	
Iosifidis et al. [30]				**95.50%**			
Azary and Savakis [50]	87.37%		86.72%	93.16%	92.97%	89.06%	
Holte et al. [33]			80.00%		89.58%		**97.50%**
Karali and ElHelw [124]		89.00%					
Iosifidis et al. [29]				90.88%			

Table 11: The classification results of state-of-the-art approaches in the i3DPost dataset.

The subset action list is shown in Table 10 and the corresponding classification results are shown in Table 11. The comparison shows that Grassmann based methods classify better except for the eight action experimental setup and the five action experimental setup (E2). For the eight action experimental setup, Iosifidis et al. [30] outperformed GSR and GRASP with fuzzy vector quantization and artificial neural networks. Holte et al.'s [33] 3D-MC and HMC methods outperformed GSR and GRASP by 0.31% and 1.25% respectively.

7.8.2 INRIA Xmas Motion Acquisition Sequences (IXMAS)

For the IXMAS dataset, all comparisons are assumed to be based on LOOCV. Wu et al. [125] uses multiple kernel learning with augmented features (AFKML) to fuse spatio-temporal and local appearance features. Liu and Shah [126] used maximization of mutual information (MMI) clustering with support vector machines. Yan et al. [127] build 4D action feature models to encode shapes of actors from multiple views. Orrite et al. [128] used histograms of normalized optical flow. Karali and ElHelw's [124] method classifies best when comparing against "Single" Grassmann subspace configurations. However, their results exclude an entire action set of "throw".

Method	Number of Views	Excluded Actions	Classification Results
GSR (Single)	4	Excludes under hand throw	79.19%
GRASP (Single)	4	Excludes under hand throw	80.76%
GSR (All)	4	Excludes under hand throw	**96.92%**
GRASP (All)	4	Excludes under hand throw	**97.69%**
Wu et al. [125]	4	Excludes all throw	88.20%
Liu and Shah [126]	4	Excludes underhand throw	82.80%
Yan et al. [127]	4	Excludes throw and point	78.00%
Karali and ElHelw [124]	Unknown	Excludes all throw	**88.48%**
Orrite et al. [128]	Unknown	Unknown	73.30%

Table 12: The classification results of state-of-the-art approaches in the IXMAS dataset.

7.8.3 Microsoft Research Action 3D Dataset (MSRAction3D)

For the MSRAction3D dataset, the list of action subsets used in the experiments are presented in Table 13. These subsets have been consistently used as baselines in existing literature. Subset 1 and Subset 2 group actions with similar characteristics. Subset 3 groups actions that are dissimilar. The full set is introduced in this dissertation as a new baseline and includes all actions.

Subset 1	Subset 2	Subset 3	Full Set
Hor. Arm Wave	High Arm Wave	High Throw	
Hammer	Hand Catch	Forward Kick	
Forward Punch	Draw X	Side Kick	
High Throw	Draw Tick	Jogging	All Actions
Hand Clap	Draw Circle	Tennis Swing	
Bend	Two Hand Wave	Tennis Serve	
Tennis Serve	Forward Kick	Golf Swing	
Pickup & Throw	Side Boxing	Pickup & Throw	

Table 13: Action subsets for the MSRAction3D dataset.

To compare against state-of-the-art results, the same experimental setup is carried out, where twenty actions were divided into three subsets consisting of eight actions. The Subsets 1 and 2 were designed to group activities with similar movements while Subset 3 was designed to group actions that are more dissimilar. As in the work of Li et al. [36] and many existing publications, three types of tests were conducted as follows: training with 1/3 of the training samples and testing with 2/3 of the samples, training with 2/3 of the samples and testing against

1/3, and training with half of the samples and testing against the other half. Cross validation was not used, and without knowing which samples were used for testing and training for each test, we compare against the same experimental setup with random samples selected for training and testing. However, LOOCV results are also presented where each test subject is validated against the remaining subjects and repeated for all subjects until all subjects have been used for training and testing. Since the average classification results over the three subsets are commonly presented, the average results are also presented for GSR and GRASP.

The work of Li et al. [36] use action graphs to model the dynamics of the actions and a Bag of Features (BoF) to encode the action and classify test samples against a training set. Yang et al. [129] extracted histograms of oriented gradients (HOG) from depth motion maps. Yang and Tian [130] propose applying PCA and normalization computed channels of depth data which they call Eigenjoints. Wang et al. [131] present a pose estimation algorithm and exploit spatio-temporal pose structures.

Table 14 presents classification results when 1/3 of the data samples were trained and 2/3 of the data samples were tested. The results indicate that the method proposed by Xia et al. [132] is the most effective. Their approach uses histograms of kinematic joint positions which are projected into a lower dimensional space using linear discriminant analysis and classified based on visual word clustering.

Subset	Subset 1	Subset 2	Subset 3	Average of Subsets[1 2 3]	Full Set
GSR (Single)	96.12%	91.03%	93.65%	93.60%	86.36%
GRASP (Single)	93.75%	92.57%	92.84%	93.05%	**87.52%**
GSR (All)	100%	100%	100%	100%	100%
GRASP (All)	100%	100%	100%	100%	100%
Li et al. [36]	89.50%	89.00%	96.30%	91.60%	N/A
Yang and Tian [130]	94.70%	95.40%	97.30%	92.47%	N/A
Yang et al. [129]	97.30%	92.20%	**98.00%**	95.83%	N/A
Xia et al. [132]	**98.47%**	**96.67%**	93.47%	**96.20%**	N/A

Table 14: The classification results of state-of-the-art approaches on the MSRAction3D dataset with 1/3 randomly trained samples, 2/3 randomly tested samples.

Table 15 presents classification results when 2/3 of the data samples were trained and1/3 of the data samples were tested. GRASP classified with a 100% accuracy on Subset 1 while Xia et al. [132] classifies the best for Subset 2 and Yang et al. [129] classifies the best on Subset 3. Yang and Tian [130] maintain the best balance of classification results on all three subsets with an average of 97.77%. When comparing the average results for this experimental setup, GRASP and GSR slightly trail the leading methods by 0.51% and 0.65% respectively.

Subset	Subset 1	Subset 2	Subset 3	Average of Subsets[1 2 3]	Full Set
GSR (Single)	97.92%	95.42%	98.44%	97.26%	**94.00%**
GRASP (Single)	**100%**	92.92%	98.44%	97.12%	93.54%
GSR (All)	100%	100%	100%	100%	100%
GRASP (All)	100%	100%	100%	100%	100%
Li et al. [36]	93.40%	92.90%	96.30%	94.20%	N/A
Yang and Tian [130]	97.30%	98.70%	97.30%	**97.77%**	N/A
Yang et al. [129]	98.70%	94.70%	**98.70%**	97.37%	N/A
Xia et al. [132]	98.61%	**97.92%**	94.93%	97.15%	N/A

Table 15: The classification results of state-of-the-art approaches on the MSRAction3D dataset with 2/3 randomly trained samples, 1/3 randomly tested samples.

Table 16 presents classification results when 1/2 of the data samples were trained and1/2 of the data samples were tested. It is observed that for this configuration GRASP and GSR classify the best with 95.63% and 95.13% respectively.

Subset	Subset 1	Subset 2	Subset 3	Average of Subsets[1 2 3]	Full Set
GSR (Single)	96.58%	**95.31%**	93.49%	95.13%	92.72%
GRASP (Single)	**97.72%**	94.35%	94.62%	**95.63%**	91.23%
GSR (All)	100%	100%	100%	100%	100%
GRASP (All)	100%	100%	100%	100%	100%
Li et al. [36]	72.90%	71.90%	79.20%	74.67%	N/A
Yang and Tian [130]	74.50%	76.10%	**96.40%**	82.33%	N/A
Yang et al. [129]	96.20%	84.10%	94.60%	91.63%	N/A
Wang et al. [131]	Not Provided	Not Provided	Not Provided	90.22%	N/A
Ellis et al. [133]	Not Provided	Not Provided	Not Provided	65.70%	N/A
Xia et al. [132]	87.98%	85.48%	63.46%	78.97%	N/A

Table 16: The classification results of state-of-the-art approaches on the MSRAction3D dataset with 1/2 randomly trained samples, 1/2 randomly tested samples.

Table 17 presents classification results using LOOCV and only comparing GSR and GRASP. This is because the cross validation method is not presented in any existing literature

on the MSRAction3D dataset. However, the results are important because they identify that action recognition is more challenging when the same subjects are excluded from the training set. Table 14 through Table 16 show average classification results in the 90th percentile. However, LOOCV proves to be more challenging with Grassmann based classification results in the 80th percentile range. GSR proves to be more effective than GRASP for the subsets and when evaluating the full set of all actions.

Subset	Subset 1	Subset 2	Subset 3	Average of Subsets[1 2 3]	Full Set
GSR (Single)	**81.67%**	**81.15%**	**87.88%**	**83.57%**	**76.17%**
GRASP (Single)	80.53%	80.35%	86.74%	82.54%	75.04%
GSR (All)	78.45%	80.72%	**89.50%**	82.89%	77.17%
GRASP (All)	76.44%	**82.81%**	88.96%	82.74%	75.13%

Table 17: The classification results of state-of-the-art approaches on the MSRAction3D dataset using leave one subject out cross validation.

7.8.4 Microsoft Research Gesture3D Dataset (MSRGesture3D)

For the MSRGesture3D dataset, all ASL gestures are evaluated using LOOCV and reported in Table 18. The results indicate that Grassmann learning in the "Single" and "All" Grassmann subspace configurations trail state-of-the-art approaches between 0.5% to 3.33%. The second leading method is by Zhang and Tian [134] who present edge enhanced depth motion maps that can be classified with kernelized support vector machines. The leading method is presented by Oreifej and Liu [135] who present a 4D descriptor based on depth, time, and spatial coordinates using histograms of normal orientations.

Subset	All Gestures
GSR (Single)	87.20%
GRASP (Single)	87.50%
GSR (All)	87.20%
GRASP (All)	87.50%
Kurakin et al. [120]	87.70%
Wang et al. [136]	88.50%
Oreifej and Liu [135]	**92.45%**
Yang et al. [129]	89.20%
Zhang and Tian [134]	90.53%

Table 18: The classification results of state-of-the-art approaches on the MSRGesture3D dataset using leave one subject out cross validation.

7.8.5 Database of Faces from AT&T Laboratories (ATT)

For the ATT dataset, 2FCV results are reported for GSR and GRASP and compared against the state-of-the-art methods listed in Table 19. The "Single" Grassmann subspace configuration of GSR and GRASP outperform all methods by a range of 12.36% to 0.13%. The closest competitive method is presented by Faraji and Qi [137] who present neutrosophic set preprocessing for noise removal and facial feature enhancement along with kernel Fisher linear discriminant analysis (KFLDA) and Tan and Triggs (TT) discriminant method. The next closest competitive method is reported at 98.53% by Liu et al. [138] using a method called spherical marginal Fisher analysis. This method is an extension of marginal Fisher analysis.

Subset	Classification Results
GSR (Single)	**99.00%**
GRASP (Single)	**99.00%**
GSR (All)	100%
GRASP (All)	100%
Yang et al. [139]	96.00%
Cai et al. [140]	96.35%
Faraji and Qi [137]	98.87%
Xu et al. [141]	96.50%
Gumus et al. [142]	95.30%
Choi et al. [143]	86.64%
Fernandes and Bala [144]	96.00%
Liu et al. [138]	98.53%

Table 19: The classification results of state-of-the-art approaches on the ATT dataset using 2-fold cross validation.

7.8.6 Extended Yale Face Database B (YALE)

For the YALE dataset, 2FCV results are reported for GSR and GRASP and compared against the state-of-the-art methods listed in Table 20. The results indicate that GRASP is the top performer with GSR trailing by only 0.04%. The closest competitive method is presented by Fernandes and Bala [144] with a classification accuracy of 97.50% using regularized linear discriminant analysis with probabilistic reasoning models. Cai et al. [140] report classification results of 95.17% using orthogonal Laplacian-faces (OLF).

Subset	Classification Results
GSR (Single)	98.85%
GRASP (Single)	**98.89%**
GSR (All)	**100%**
GRASP (All)	**100%**
Yang et al. [139]	84.24%
Cai et al. [140]	95.17%
Choi et al. [143]	82.13%
Fernandes and Bala [144]	97.50%
Liu et al. [138]	86.13%

Table 20: The classification results of state-of-the-art approaches on the YALE dataset using 2-fold cross validation.

7.9 Benefits and Limitations of Grassmann Learning

As previously explained, there are many benefits to using Grassmann manifolds including promoting high between-class discrimination and within-class clustering, computational advantages, and accounting for missing information through linear spans of subspaces. Grassmann learning can be used for various classification and recognition problems including action, face, and object classification. Grassmann learning has proven to be effective when large amounts of training information is available and subspaces are well represented by large amounts of data samples on a Grassmann manifold. Grassmann learning is difficult to use in an unsupervised framework without class labeling. A better understanding of data clustering on

Grassmann manifolds is necessary to explore and implement unsupervised Grassmann learning methods. Grassmann learning has also shown to be less discriminative for large class systems with subspaces represented by a small number of data samples.

8 Conclusions

The benefits of Grassmann learning for processing high dimensional data and easing computation loads were explored. This dissertation began by discussing high dimensional representations and radial distance surfaces were proposed. Such surfaces were found to be scale invariant, localization invariant, and time invariant for multi-view action classification. This was justified through manifold learning with LPP. However, the results indicate that the approach is not robust in terms of promoting high between-class discrimination and requires an exhaustive dictionary of action representations across multiple views. The next contribution in this dissertation is the definition of motion history surfaces (MHS) and motion depth surfaces (MDS) based on spatio-temporal considerations. These high dimensional surfaces were evaluated with dimensionality reduction algorithms including PCA, LGE, Spectral Regression, Grassmann learning, and Sparse Representations.

For sparse representations, we presented a novel approach to action classification of 3D video sequences using sparse representations of spatio-temporal kinematic joint features and raw depth features which are invariant to scale and localization. We created over-complete dictionaries and took advantage of the sparse nature of the feature descriptors to classify actions using least squares loss ℓ^1-norm minimization with parameter regularization. We found that the representations of raw depth features are naturally sparser than kinematic joint features as a result of comparing ℓ^1-norm minimization with ℓ^2-norm nearest neighbor classification.

Understanding the benefits and drawbacks of these various learning techniques allowed for the next major contribution of this dissertation with the GSR and GRASP frameworks. These methods are intended to improve classification accuracies and improved run-time performance. An extensive evaluation of GSR and GRASP was made for the applications of action

classification and face recognition. Beyond the GSR and GRASP framework, another major contribution is the observation of standardizing Grassmann kernel distributions and its impact on classification accuracies using GSR and GRASP. We discovered that standardization allows for the best results when there is variation in subspace sizes between Grassmann points.

8.1 Future Work

There are many research opportunities to explore within the framework of GSR and GRASP methodologies and beyond. Applications such as object recognition and super-resolution can be explored through GSR and GRASP. However, GSR and GRASP are supervised learning algorithms and are not suited for clustering analysis. Gruber and Theis [145] have found improved clustering patterns when applying k-means clustering on Grassmann manifolds. Similar and more recent work using k-means clustering on Grassmann manifolds was also observed by Shirazi et al. [146] with a potential to improve action classification accuracies. The understanding of clustering patterns on a Grassmann manifold can give rise to unsupervised learning algorithms that can also account for high between class discrimination and high within-class clustering. Grassmann learning can be incorporated into clustering methods such as MDS, LLE, and Isomap for improved clustering and it would be interesting to see the benefits and drawbacks of such Grassmann clustering approaches. Beyond Grassmann clustering, Grassmann classifiers for face sequence recognition using SVM's are presented in the work of Shigenaka et al. [147]. Similarly, Vemulapalli et al. [148] present a general framework for SVM classifiers on Riemannian manifolds using kernel learning approaches. There is opportunity to explore the effectiveness of SVM classifiers on Grassmann manifolds.

9 References

[1] D. Weinland, R. Ronfard and E. Boyer, "A survey of vision-based methods for action representation, segmentation and recognition," *Computer Vision and Image Understanding*, 2011.

[2] H. Ghasemzadeh, V. Loseu and R. Jafari, "Collaborative Signal Processing for Action Recognition in Body Sensor Networks: A Distributed Classification Algorithm Using Motion Transcripts," in *In Proc. 9th ACM/IEEE Int. Conf. Inf. Process.*, 2010.

[3] K. Raja, I. Laptev, P. Perez and L. Oisel, "Joint pose estimation and action recognition in image graphs," in *18th IEEE International Conference on International Conference on Image Processing (ICIP)*, 2011.

[4] D. Weinland, E. Boyer and R. Ronfard, "Action Recognition from Arbitrary Views using 3D Exemplars," in *IEEE ICCV*, 2007.

[5] S. Maji, L. Bourdev and J. Malik, "Action recognition from a distributed representation of pose and appearance," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[6] Y. Wang and Z. Zhang, "View-invariant action recognition in surveillance videos," in *First Asian Conference on Pattern Recognition (ACPR)*, 2011.

[7] H. Imtiaz, U. Mahbub and M. A. R. Ahad, "Action recognition algorithm based on optical flow and RANSAC in frequency domain," in *Proceedings of SICE Annual Conference (SICE)*, 2011.

[8] M. A. R. Ahad, J. Tan, H. Kim and S. Ishikawa, "Action recognition by employing combined directional motion history and energy images," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2010.

[9] C. Harris and M. Stephens, "A combined corner and edge detector," in *Alvey Vision Conference*, 1988.

[10] A. P. B. Lopes, R. S. Oliveira, J. M. d. Almeida and A. d. A. Araujo, "Comparing alternatives for capturing dynamic information in Bag-of-Visual-Features approaches applied to human actions recognition," in *IEEE International Workshop on Multimedia Signal Processing (MMSP)*, 2009.

[11] J. Liu, J. Yang, Y. Zhang and X. He, "Action Recognition by Multiple Features and Hyper-Sphere Multi-class SVM," in *20th International Conference on Pattern Recognition (ICPR)*, 2010.

[12] Y. Ke and R. Sukthankar, "PCA-SIFT: A more distinctive representation for local image descriptors," in *Computer Vision and Pattern Recognition*, 2004.

[13] S. Azary and A. Savakis, "View Invariant Activity Recognition with Manifold Learning," in *Advances in Visual Computing*, *Lecture Notes in Computer Science*, Las Vegas, Nevada, 2010.

[14] M. S. Ryoo and J. K. Aggarwal, "Recognition of composite human activities through context-free grammar based representation," in *Computer Vision and Pattern Recognition*, 2006.

[15] B. Yao and S.-C. Zhu, "Learning deformable action templates from cluttered videos," in *IEEE 12th International Conference on Computer Vision*, 2009.

[16] A. F. Bobick and J. W. Davis, "The recognition of human movement using temporal templates," *Pattern Analysis and Machine Intelligence*, vol. 23, no. 2, pp. 257-267, 2001.

[17] V. Krüger and D. Grest, "Using Hidden Markov Models for Recognizing Action Primitives in Complex Actions," in *Image Analysis*, Springer Berlin Heidelberg, 2007, pp. 203-212.

[18] B. Chakraborty, O. Rudovic and J. Gonzalez, "View-Invariant Human-Body Detection with Extension to Human Action Recognition using Component-Wise HMM of Body Parts," in *Automatic Face & Gesture Recognition*, 2008.

[19] I. N. Junejo, E. Dexter, I. Laptev and P. Perez, "View-Independent Action Recognition from Temporal Self-Similarities," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011.

[20] B. Yao, Z. Liu, X. Nie and S.-C. Zhu, "Animated Pose Templates for Modelling and Detecting Human Actions," in *IEEE Pattern Recognition and Machine Intelligence*, 2013.

[21] I. Laptev and T. Lindeberg, "Space-time interest points," in *ICCV*, Nice, France, 2003.

[22] J. Gall, A. Yao, N. Razavi, L. V. Gool and V. Lempitsky, "Hough Forests for Object Detection, Tracking, and Action Recognition," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011.

[23] I. Laptev, "On Space-Time Interest Points," in *International Journal of Computer Vision*, 2005.

[24] G. Willems, T. Tuytelaars and L. V. Gool, "An Efficient Dense and Scale-Invariant Spatio-Temporal Interest Point Detector," in *IEEE International Conference on ECCV*, 2008.

[25] K. Vili, Z. Guoying and P. Matti, "Texture based description of movements for activity analysis," in *VISAPP*, 2008.

[26] R. Souvenir and K. Parrigan, "Viewpoint Manifolds for Action Recognition," in *EURASIP Journal on Image and Video Processing*, 2009.

[27] N. Gkalelis, H. Kim, A. Hilton, N. Nikolaidis and I. Pitas, "The i3DPost multi-view and 3D human action/interaction Database," in *Conference for Visual Media Production*, 2009.

[28] L. Ding, X. Ding and C. Fang, "Continuous Pose Normalization for Pose-Robust Face Recognition," in *IEEE Signal Processing Letters*, 2012.

[29] A. Iosifidis, N. Nikolaidis and I. Pitas, "Movement Recognition Exploiting Multi-View Information," in *IEEE Conference on Multi Media Signal Processing (MMSP)*, 2010.

[30] A. Iosifidis, A. Tefas, N. Nikolaidis and I. Pitas, "Multi-view human movement recognition based on fuzzy distances and linear discriminant analysis," in *Computer Vision and Image Understanding*, 2012.

[31] R. Bodor, A. Drenner, D. Fehr, O. Masoud and N. Papanikolopoulos, "View-independent human motion classification using image-based reconstruction," *Image and Vision Computing*, vol. 27, no. 8, pp. 1194-1206, 2009.

[32] K. Huang, Y. Zhang and T. Tan, "A Discriminative Model of Motion and Cross Ratio for View-Invariant Action Recognition," in *IEEE Transactions on Image Processing*, 2012.

[33] M. B. Holte, T. B. Moeslund, N. Nikolaidis and I. Pitas, "3D Human Action Recognition for Multi-View Camera Systems," in *International Conference of 3D Imaging, Modeling, Processing, Visualization and Transmission*, 2011.

[34] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman and A. Blake, "Real-Time Human Pose Recognition in Parts from Single Depth Images," in *CVPR*, 2011.

[35] L. A. Schwarz, A. Mkhitaryan, D. Mateus and N. Navab, "Estimating human 3D pose from Time-of-Flight images based on geodesic distances and optical flow," in *FG*, 2011.

[36] W. Li, Z. Zhang and Z. Liu, "Action recognition based on a bag of 3D points," in *CVPRW*, 2010.

[37] J. Wang, Z. Liu, Y. Wu and J. Yuan, "Mining Actionlet Ensemble for Action Recognition with Depth Cameras," in *Computer Vision and Pattern Recognition (CVPR)*, 2012.

[38] E.-J. Weng and L.-C. Fu, "On-Line Human Action Recognition by Combining Joint Tracking and Key Pose Recognition," in *Intelligent Robots and Systems (IROS)*, 2012.

[39] A. Mansur, Y. Makihara and Y. Yagi, "Inverse Dynamics for Action Recognition," in *Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2012.

[40] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of cognitive neuroscience*, vol. 3, no. 1, pp. 71-86, 1991.

[41] P. N. Belhumeur, J. P. Hespanha and D. J. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection," *Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 711-720, 1997.

[42] X. He, S. Yan, Y. Hu, P. Niyogi and H.-J. Zhang, "Face Recognition Using Laplacianfaces," *Pattern Analysis and Machine Intelligence*, vol. 27, no. 3, pp. 328-340, 2005.

[43] X. Tan and B. Triggs, "Enhanced Local Texture Feature Sets for Face Recognition Under Difficult Lighting Conditions," *Image Processing*, vol. 19, no. 6, pp. 1635-1650, 2010.

[44] T. Ahonen, A. Hadid and M. Pietikainen, "Face description with local binary patterns: Application to face recognition," *Pattern Analysis and Machine Intelligence*, vol. 28, no. 12, pp. 2037-2041, 2006.

[45] G. Zhang, X. Huang, S. Z. Li, Y. Wang and X. Wu, "Boosting local binary pattern (LBP)-based face recognition," in *Advances in biometric person authentication*, 2005.

[46] T. Ojala, M. Pietikainen and T. Maenpaa, "Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns," *Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971-987, 2002.

[47] L. Wolf, T. Hassner and Y. Taigman, "Descriptor based methods in the wild," in *Workshop on Faces in'Real-Life'Images: Detection, Alignment, and Recognition*, 2008.

[48] S. Azary and A. Savakis, "A spatiotemporal descriptor based on radial distances and 3D joint tracking for action classification," in *19th IEEE International Conference on Image Processing (ICIP)*, Orlando, Florida, 2012.

[49] J. W. Davis and A. F. Bobick, "The Representation and Recognition of Action Using Temporal Templates," in *IEEE Conference on Computer Vision and Pattern Recognition*, 1997.

[50] S. Azary and A. Savakis, "Multi-view action classification using sparse representations on Motion History Images," in *IEEE Western New York Image Processing Workshops (WNYIPW)*, Rochester, NY, 2012.

[51] S. Ali and M. Shah, "Human action recognition in videos using kinematic features and

multiple instance learning," *Pattern Analysis and Machine Intelligence,* vol. 32, no. 2, pp. 288-303, 2010.

[52] H. J. Seo and P. Milanfar, "Action Recognition from One Example," *Pattern Analysis and Machine Intelligence,* vol. 33, no. 5, pp. 867-882, 2011.

[53] J. Shlens, "A Tutorial on Principal Component Analysis," Salk Institute for Biological Studies, La Jolla, CA, 2005.

[54] L. I. Smith, "A Tutorial on Principal Components Analysis," 2002. [Online]. Available: http://kybele.psych.cornell.edu/~edelman/Psych-465-Spring-2003/PCA-tutorial.pdf.

[55] L. Qiaoa, S. Chen and X. Tan, "Sparsity Preserving Projections with Applications to Face Recognition," *Pattern Recognition,* vol. 43, no. 1, pp. 331-341, 2010.

[56] D. Fradkin and D. Madigan, "Experiments with random projections for machine learning," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2003.

[57] M. A. A. Cox, "Analysis of stock market indices through multidimensional scaling," *Journal of Statistical Computation and Simulation,* vol. 83, no. 11, pp. 2015-2029, 2013.

[58] Y. Ding, D. Yang and G. Han, "Multidimensional Scaling-Based Localization Algorithm for Wireless Sensor Network with Geometric Correction," *Journal of Networks,* vol. 9, no. 3, pp. 582-587, 2014.

[59] H. Mogi and Y. Taguchi, "Protein binding prediction using non-metric multidimensional scaling method," in *Bioinformatics and Biomedicine Workshops (BIBMW)*, 2011.

[60] A. Buja, D. F. Swayne, M. L. Littman, N. Dean and H. Hofmann, "Interactive Data Visualization with Multidimensional Scaling," 2004.

[61] M. Steyvers, "Multidimensional Scaling," Encyclopedia of Cognitive Science, 2002.

[62] J. B. Kruskal, "Nonmetric multidimensional scaling: a numerical method," *Psychometrika,* vol. 29, no. 2, pp. 115-129, 1964.

[63] A. Ghodsi, "Dimensionality Reduction A Short Tutorial," Department of Statistics and Actuarial Science, University of Waterloo, Waterloo, Ontario, Canada, 2006.

[64] T. Yang, J. Liu, L. McMillan and W. Wang, "A fast approximation to multidimensional scaling," in *ECCV Workshop on Computation Intensive Methods for Computer Vision (CIMCV)*, 2006.

[65] B. Li, C.-H. Zheng, D.-S. Huang, L. Zhang and K. Han, "Gene expression data classification using locally linear discriminant embedding," *Computers in Biology and Medicine,* vol. 40, no. 10, pp. 802-810, 2010.

[66] H. Chang, D.-Y. Yeung and Y. Xiong, "Super-resolution through neighbor embedding," in *Computer Vision and Pattern Recognition*, 2004.

[67] D. d. Ridder and R. P. Duin, "Locally Linear Embedding for Classification," in *Pattern Recognition Group*, The Netherlands, 2002.

[68] L. K. Saul and S. T. Roweis, "An Introduction to Locally Linear Embedding," 2001. [Online]. Available: https://www.cs.nyu.edu/~roweis/lle/papers/lleintro.pdf. [Accessed 4 February 2012].

[69] J. Yin, D. Hu and Z. Zhou, "Growing locally linear embedding for manifold learning," *Journal of Pattern Recognition,* vol. 2, no. 1, pp. 1-16, 2007.

[70] S. Mika, G. Ratsch, J. Weston, B. Scholkopf and K.-R. Muller, "Fisher Discriminant Analysis with Kernels," in *IEEE Signal Processing Society Workshop* , 1999.

[71] X. He and P. Niyogi, "Locality Preserving Projections," in *Advances in Neural Information Processing Systems*, 2003.

[72] X. He, D. Cai, S. Yan and H.-J. Zhang, "Neighborhood Preserving Embedding," in *IEEE ICCV*, 2005.

[73] L. Wang and D. Suter, "Learning and Matching of Dynamic Shape Manifolds for Human Action Recognition," in *IEEE Transactions on Image Processing*, 2007.

[74] P. Liu, J. Wang, M. She and H. Liu, "Human action recognition based on 3D SIFT and LDA model," in *IEEE Robotic Intelligence In Informationally Structured Space (RiiSS)*, 2011.

[75] M. Belkin and P. Niyogi, "Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering," in *Advances in Neural Information Processing Systems*, Vancouver, British Columbia, Canada, 2002.

[76] Y. Tang and R. Rose, "A study of using locality preserving projections for feature extraction in speech recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2008.

[77] R. Pless and R. Souvenir, "A Survey of Manifold Learning for Images," in *IPJS Transactions on Computer Vision And Applications*, 2009.

[78] D. Cai, X. He and J. Han, "Spectral Regression for Efficient Regularized Subspace Learning," in *IEEE International Conference on Computer Vision (ICCV)*, 2007.

[79] M. T. Harandi, C. Sanderson, S. Shirazi and B. C. Lovell, "Graph Embedding Discriminant Analysis on Grassmannian Manifolds for Improved Image Set Matching," in *IEEE CVPR*, 2011.

[80] 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.

[81] M. T. Harandi, C. Sanderson, A. Wiliem and B. C. Lovell, "Kernel Analysis over Riemannian Manifolds for Visual Recognition of Actions, Pedestrians and Textures," in *IEEE Workshop on Applications of Computer Vision (WACV)*, 2012.

[82] J. Hamm and D. D. Lee, "Grassmann Discriminant Analysis: a Unifying View on Subspace-Based Learning," in *Int. Conf. Machine Learning (ICML)*, 2008.

[83] R. Shigenaka, B. Raytchev, T. Tamaki and K. Kaneda, "Face Sequence Recognition Using Grassmann Distances and Grassmann Kernels," in *World Congress on Computational Intelligence*, 2012.

[84] S. W. Park and M. Savvides, "The Multifactor Extension of Grassmann Manifolds for Face Recognition," in *IEEE Automatic Face & Gesture Recognition*, 2011.

[85] P. Turaga and A. Veeraraghavan, "Statistical analysis on Stiefel and Grassmann manifolds with applications in computer vision," in *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2008.

[86] Q. Rentmeesters, P. A. Absil and P. V. Dooren, "An efficient particle filtering technique on the Grassmann manifold," in *Acoustics Speech and Signal Processing (ICASSP)*, 2010.

[87] J. M. Lee, "Introduction to smooth manifolds," in *Springer*, 2002.

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

[89] P. Turaga, A. Veeraraghavan and R. Chellappa, "Statistical analysis on Stiefel and Grassmann manifolds with applications in computer vision," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[90] B. Schölkopf, A. Smola and K.-R. Müller, "Kernel principal component analysis," in *Artificial Neural Networks*, 1997.

[91] S. Mika, B. Scholkopf, A. Smola, K.-R. Müller, M. Scholz and G. Rätsch, "Kernel PCA and De-Noising in Feature Spaces," in *NIPS*, 1998.

[92] C. Liu, "Gabor-based kernel PCA with fractional power polynomial models for face recognition," in *Pattern Analysis and Machine Intelligence*, 2004.

[93] J. M. Juran, "The non-Pareto Principle: Mea culpa," *Quality Progress*, pp. 8-9, May 1975.

[94] J. D. Farmer and J. Geanakoplos, "Power laws in economics and elsewhere," Santa Fe Institute, Santa Fe, NM, 2008.

[95] G. B. West, "The Origin of Universal Scaling Laws in Biology," New York, Oxford University Press, 1999.

[96] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel and B. Bhattacharjee, "Measurement and analysis of online social networks," in *IMC '07 Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, New York, NY, 2007.

[97] J. Wright, A. Yang, A. Ganesh, S. Sastry and Y. Ma, "Robust Face Recognition via Sparse Representation," in *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2009.

[98] F. Qiu, Y. Xu, C. Wang and Y. Yang, "Noisy image super-resolution with sparse mixing estimators," in *4th International Congress on Image and Signal Processing (CISP)*, 2011.

[99] L. Bao, W. Liu, Y. Zhu, Z. Pu and Magnin, "Sparse representation based MRI denoising with total variation," in *9th International Conference on Signal Processing (ICSP)*, 2008.

[100] Y. Zuo and B. Zhang, "General image classification based on sparse representation," in *9th IEEE International Conference on Cognitive Informatics (ICCI)*, 2010.

[101] J. Zhang, Y. Wang, J. Chen and Q. Li, "Sparse Representation for Action Recognition," in *3rd International Congress on Image and Signal Processing (CISP2010)*, 2010.

[102] C. Liu, Y. Yang and Y. Chen, "Human Action Recognition using Sparse Representation," in *IEEE International Conference on Intelligent Computing and Intelligent Systems (ICIS)*, 2009.

[103] R. Ptucha and A. Savakis, "Joint Optimization of Manifold Learning and Sparse," in *10th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, 2013.

[104] X. Lu, Y. Yuan and P. Yan, "Alternatively Constrained Dictionary Learning for Image Superresolution," in *Computer Vision and Pattern Recognition (CVPR)*, 2012.

[105] M. Zheng, J. Bu, C. Chen, C. Wang, L. Zhang, G. Qiu and D. Cai, "Graph Regularized Sparse Coding for Image Representation," *IEEE Transactions on Image Processing*, vol. 20, no. 5, pp. 1327-1336, 2011.

[106] J. Mairal and B. Yu, "Complexity Analysis of the Lasso Regularization Path," in *arXiv*, 2012.

[107] B. Gaartner, M. Jaggi and C. Maria, "An exponential lower bound on the complexity of regularization paths," *arXiv*, vol. 0903, no. 4817, 2009.

[108] J. Wright, Y. Ma, J. Maira, G. Sapiro, T. S. Huang and S. Yan, "Sparse Representation for Computer Vision and Pattern Recognition," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 1031 - 1044, 2010.

[109] S. J. Miller, "The Method of Least Squares," Mathematics Department Brown University, Providence, RI, 2006.

[110] S. Bektaş and Y. Şişman, "The comparison of L1 and L2-norm minimization methods," in *International Journal of the Physical Sciences (IJPS)*, 2010.

[111] D. L. Donoho, M. Elad and V. Temlyakov, "Stable recovery of sparse overcomplete representations," in *IEEE Transactions on Information Theory*, 2005.

[112] D. L. Donoho and Y. Tsaig, "Fast Solution of L1-norm Minimization Problems When the Solution May Be Sparse," Stanford CA, 94305, Department of Statistics, Stanford University, 2006.

[113] M. Schmidt, "Least Squares Optimization with L1-Norm Regularization," University of British Columbia, 2005.

[114] S. Azary and A. Savakis, "3D Action Classification Using Sparse Spatio-temporal Feature Representations," in *Advances in Visual Computing*, 2012.

[115] F. D. l. Torre, "A Least-Squares Framework for Component Analysis," in *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2012.

[116] S. Azary and A. Savakis, "Grassmannian Spectral Regression for Action Recognition," in *Advances in Visual Computing, Lecture Notes in Computer Science*, Rethymnon, Crete, Greece, 2013.

[117] S. Azary and A. Savakis, "Grassmannian Sparse Representations and Motion Depth Surfaces for 3D Action Recognition," in *CVPR Workshop on Human Activity Understanding from 3D Data*, 2013.

[118] D. Weinland, R. Ronfard and E. Boyer, "Free Viewpoint Action Recognition using Motion History Volumes," in *Computer Vision and Image Understanding*, 2006.

[119] S. Ramagiri, R. Kavi and V. Kulathumani, "Real-time multi-view human action recognition using a wireless camera network," in *Distributed Smart Cameras (ICDSC)*, 2011.

[120] A. Kurakin, Z. Zhang and Z. Liu, "A Real Time System for Dynamic Hand Gesture Recognition with a Depth Sensor," in *European Signal Processing Conference*, 2012.

[121] F. S. Samaria and A. C. Harter, "Parameterisation of a stochastic model for human face identification," in *Applications of Computer Vision*, 1994.

[122] G. B. Huang, M. Ramesh, T. Berg and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," University of Massachusetts, Amherst, 2007.

[123] K.-C. Lee, J. Ho and D. Kriegman, "Acquiring linear subspaces for face recognition under variable lighting," *Pattern Analysis and Machine Intelligence*, vol. 27, no. 5, pp. 684-698, 2005.

[124] A. Karali and M. ElHelw, "Motion history of skeletal volumes and temporal change in bounding volume fusion for human action recognition," in *Multimodal Pattern Recognition of Social Signals in Human-Computer-Interaction. Springer Berlin Heidelberg*, 2013.

[125] X. Wu, D. Xu, L. Duan and J. Luo, "Action recognition using context and appearance distribution features," in *Computer Vision and Pattern Recognition*, 2011.

[126] J. Liu and M. Shah, "Learning human actions via information maximization," in *Computer Vision and Pattern Recognition*, 2008.

[127] P. Yan, S. M. Khan and M. Shah, "Learning 4d action feature models for arbitrary view action recognition," in *Computer Vision and Pattern Recognition*, 2008.

[128] C. Orrite, P. Monforte, M. Rodriguez and E. Herrero, "Human Action Recognition under Partial Occlusions," in *Pattern Recognition and Image Analysis*, 2013.

[129] X. Yang, C. Zhang and Y. Tian, "Recognizing actions using depth motion maps-based histograms of oriented gradients," in *Proceedings of the 20th ACM international conference on Multimedia*, 2012.

[130] X. Yang and Y. Tian, "Eigenjoints-based action recognition using naive-bayes-nearest-neighbor," in *Computer Vision and Pattern Recognition Workshops*, 2012.

[131] C. Wang, Y. Wang and A. L. Yuille, "An approach to pose-based action recognition," in *Computer Vision and Pattern Recognition*, 2013.

[132] L. Xia, C.-C. Chen and J. K. Aggarwal, "View invariant human action recognition using histograms of 3d joints," in *Computer Vision and Pattern Recognition Workshops*, 2012.

[133] C. Ellis, S. Z. Masood, M. F. Tappen, J. J. L. Jr and R. Sukthankar, "Exploring the trade-off between accuracy and observational latency in action recognition," *International Journal of Computer Vision*, vol. 101, no. 3, pp. 420-436, 2013.

[134] C. Zhang and Y. Tian, "Edge Enhanced Depth Motion Map for Dynamic Hand Gesture Recognition," in *Computer Vision and Pattern Recognition Workshops*, 2013.

[135] O. Oreifej and Z. Liu, "Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences," in *Computer Vision and Pattern Recognition*, 2013.

[136] J. Wang, Z. Liu, J. Chorowski, Z. Chen and Y. Wu, "Robust 3d action recognition with random occupancy patterns," in *Computer Vision–ECCV*, 2012.

[137] M. R. Faraji and X. Qi, "An effective neutrosophic set-based preprocessing method for face recognition," in *Multimedia and Expo Workshops*, 2013.

[138] R. Liu, C. Jia, E. Pang, M. Qu, S. Pang and Z. Yu, "Global and Local Information Based Spherical Marginal Fisher Analysis for Face Recognition," *Journal of Information and Computational Science*, vol. 10, no. 4, p. 1025–1034, 2013.

[139] J. Yang, D. Zhang, A. F. Frangi and J.-y. Yang, "Two-dimensional PCA: a new approach to appearance-based face representation and recognition," *Pattern Analysis and Machine Intelligence*, vol. 26, no. 1, pp. 131-137, 2004.

[140] D. Cai, X. He, J. Han and H.-J. Zhang, "Orthogonal laplacianfaces for face recognition," *Image Processing*, vol. 15, no. 11, pp. 3608-3614, 2006.

[141] Y. Xu, D. Zhang, J. Yang and J.-Y. Yang, "A two-phase test sample sparse representation method for use with face recognition," *Circuits and Systems for Video Technology*, vol. 21, no. 9, pp. 1255-1262, 2011.

[142] E. Gumus, N. Kilic, A. Sertbas and O. N. Ucan, "Evaluation of face recognition techniques using PCA, wavelets and SVM," *Expert Systems with Applications,* vol. 37, no. 9, pp. 6404-6408, 2010.

[143] Y. Choi, T. Tokumoto, M. Lee and S. Ozawa, "Incremental two-dimensional two-directional principal component analysis (I (2D) 2 PCA) for face recognition," in *Acoustics, Speech and Signal Processing* , 2011.

[144] S. Fernandes and J. Bala, "Performance Analysis of PCA-based and LDA-based Algorithms for Face Recognition," *nternational Journal of Signal Processing Systems,* vol. 1, no. 1, pp. 1-6, 2013.

[145] P. Gruber and F. J. Theis, "Grassmann Clustering," in *European Signal Processing Conference*, 2006.

[146] 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 *Internation Conference on Image Processing (ICIP)*, 2012.

[147] R. Shigenaka, B. Raytchev, T. Tamaki and K. Kaneda, "Face Sequence Recognition Using Grassmann Distances and Grassmann Kernels," in *World Congress on Computational Intelligence*, 2012.

[148] R. Vemulapalli, J. K. Pillai and R. Chellappa, "Kernel Learning for Extrinsic Classification of Manifold Features," in *Computer Vision and Pattern Recognition*, 2013.