ZeroToHero

A Punch Recognition & Quality Assessment System

By

LIAM O'SHEA

Supervised By

Dr. Sion Hannuna & Professor Majid Mirmehdi

Department of Computer Science
UNIVERSITY OF BRISTOL

A dissertation submitted to the University of Bristol in accordance with the requirements of the degree of BACHELOR OF SCIENCE in the Faculty of Engineering.

MAY 2014

Abstract

In ZeroToHero we present a real time punch recognition and quality assessment algorithm which has the potential to bring specialist boxing coaching into a home environment. Human poses were recorded using the Kinect & Microsoft SDK, an already widespread, low cost consumer device with high accessibility. We investigated the efficacy of a number of dimensionality reduction techniques when applied to pose sequences including CCA, LLE, Hessian LLE, Laplacian Eigenmaps, LTSA and PCA. We found that PCA was most suitable for facilitating time series segmentation since we could consistently obtain a sinosidal-like sequence. This enabled us to perform automatic punch segmentation on the reduced dimensionality pose-time series and extract useful features from the original unsmoothed reduced dimensionality data set. Segmentation was achieved by smoothing the projection coefficients for the first principal components before extracting local maxima from across the time series. Each maxima was then inspected by a set of heuristics which determine whether it signifies the start of a punch or is just a local maxima that still exists after smoothing. Finally, the optimal number of principal components retained and the suitability of a variety of classifiers was empirically assessed including SVM, Neural Networks(NN) and Decision trees(DT). On a 6 class problem of punch classification ZeroToHero achieved accuracy results of **94.90%** with NN and **93.22%** accuracy using SVM and DT. On a two-class problem of good and bad jabs an accuracy of **98.60%** was achieved using NN, **97.83%** by SVM and **91.21%** by DT.

AUTHOR'S DECLARATION

I declare that the work in this dissertation was carried out in accordance with the requirements of the University's Regulations and Code of Practice for Research Degree Programmes and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the author.

SIGNED: ..... DATE: .....

Table of Contents

List of Tables

Table **Page**

List of Figures

CHAPTER 1

INTRODUCTION

Computer aided coaching has revolutionised training approaches and been a key driver in improving sports performance at the highest level.[3]Professional athletes can now use advanced augmented coaching to accurately and reliably collect measurements that can then used as indicators of performance.[21] This gives the coaching teams unique insight which they use to tailor training programs to give the athlete a goal to focus on and improve. Huge injections of research funding in preparation for the London Olympics such as The Elite Sport Performance Research in Training (ESPRIT) Programme, meant that UK Sport was allocated £2,000,000 while the Engineering and Physical Sciences Research Council were allocated £6,119,249.[18] [40] This money was spent on enhancing elite performance training with technology [14] [39] which allowed British Olympians to take advantage of these training methods and return with a medal record making it the most successful team since 1908.[43] These hi-tech training methods were used by Team GB Boxing [4] and have started to become incorporated in the USA Boxing team.[20]

However despite the recent success, sports technology is still in its infancy, with the majority of devices being expensive, specialist and proprietary[17]. There is an array of consumer devices such as the Fitbit, Nike+ Fuelband, Garmin Vivofit and Jawbone that function as general activity trackers but there is currently nothing that offers specialist training or coaching advice. The purpose of ZeroToHero was to use an already widespread, low cost consumer device to bring specialist boxing coaching to everyone. The low-cost is especially crucial since most boxing clubs struggle with funding, rely on volunteers and traditionally have members that are the young and least wealthy in society. ZeroToHero explores the ability of the Kinect to act as a electronic boxing trainer, offering advice on punch and stance performance.

1.1 Goals

1. To recognise that a punch is being thrown
2. To successfully classify this punch
3. To classify between a good and bad punch

1.1.1 Scope

For the course of this project the 'orthodox' stance will be used, meaning only those who are right handed will benefit from this system initially. However once the concept is proven it would be trivial to adjust this for 'southpaws' since the principles and foundations are exactly the same. The only difference will be that the right hand side of the body will be the more dominant and will be physically closer to the Kinect instead of the left hand side.

CHAPTER 2

BACKGROUND & RESEARCH

This chapter describes and explains boxing concepts which are important to understanding the goal of the project. It gives an overview of the types of punches a boxer must execute along with common errors associated with those. It also reviews relevant literature in areas covered by this thesis and explores the current cutting edge possibilities. Finally I present an overview of the potential techniques that can be used for my project after analysing a variety of different techniques for their suitability based on the literature review and cutting edge methods.

2.1 Boxing Background

Boxing requires an incredible amount of co-ordination and timing as well as the ability to rapidly execute punches in a controlled and precise manner. Unlike professional boxing which is 12 x 3 minute rounds an amateur bout is 3 x 2 minute rounds which changes the dynamics of the contest. Amateur boxing relies on a points scoring system since there is often insufficient time for knockouts, requiring amateur boxers to rely on the mastery of technique and proper form. For example, dropping a guard for a split second can open you up to an experienced boxer and could spell disaster.

As someone who has boxed for over 5 years and captain of the University of Bristol's Amateur Boxing Club (UOBABC) I understand the difficulties of developing good technique and how much time and experience is required from a coach to develop a new boxer. This one-on-one time is incredibly valuable but also expensive and for the large majority very hard to get. ZerotoHero's aim is to be able to identify different types of punches and to offer feedback on the quality of the movement. This will bring some much needed expert advice to a beginner who can practice in the comfort of their own home.

I am using my own experience and that of local professionals, coaches and local legend Denis Stinchcombe MBE, the centre director of Riverside Youth Project and Registrar for the Western Counties for the Amateur Boxing Association.

2.1.1 Motivation

This research is borne out of a desire to improve access and cost to boxing coaching which are problems I have encountered first hand through the University Boxing Club. In a wider context it could be used in developing countries where physical access to coaches with the required expertise may be difficult as well as local clubs in the UK. Every year UOBABC takes in new members that are total beginners. We spend an enormous amount of time and effort helping them learn the basics and encourage people to practice at home. The problem from a boxer's perspective is that it is incredibly hard to spot your own faults, especially without experience. If it was possible to practice at home with the benefits of coaching it would bring massive improvements to a trainees ability. The current most effective way to train as an individual is to stand in front of a mirror and observe yourself while shadow boxing. It could also be used as a way to introduce younger children to the sport since the Kinect is incredibly popular in that demographic and so is a good choice from an inclusion perspective.

2.2 Boxing Technique

For the scope of this project I am going to focus on the most common orthodox stance. There are tens of slight variations on each punch but I am going to focus on the core foundations and important principles from which these can be built.

2.2.1 Stance

The most fundamental building block of boxing is the stance, that is how you hold and position your body as well as the placement & orientation of your feet. A good stance is crucial since it allows the boxer to be well balanced and light on their feet, allowing fast movement in any direction as well as the ability to quickly duck, weave, slip and bob and lay back to avoid punches. It is also crucial for offence since the power from punches come from the transfer of weight from one leg to another which requires a very specific twisting hip movement. Often beginners forget this crucial step and so I'm hoping to use this unique trait to help me judge quality later on. A successful stance should have the following characteristics:

- Left foot forward, right foot back with a distance slightly wider than shoulder width with a 45 degree angle twist.
- Right heel off the ground at all times with weight distribution mostly on your back leg.

- Chin tucked down.
- Right hand on the right hand side of your chin, left hand should be a few inches in front of the left side of the face.
- Elbows tucked in to protect the torso section.

2.2.2 Punches

Jab

The elbow should stay tucked in while the left fist extends with palms facing inwards before twisting your wrist at the last moment. The natural thing to do is extend the punch with palms facing down, unfortunately this immediately makes the elbow stick out which allows the opponent to easily see you are about to throw a punch (telegraphing) while opening up your body for a counter attack. The punch should also finish so your arm is fully extended which helps to extend your reach and protect your chin before speedily returning it to the guard position.
Target Characteristic: Elbow movement

Cross

The cross is designed as your heavy straight punch and as such is slower but more powerful. To get a snappy and powerful punch it is important to transfer your weight rapidly from your back leg to your front leg, twisting your hips.
Target Characteristic: Twisting of the hip
Target Characteristic: Distribution of weight to the front foot

Hooks

Your elbow should be raised to shoulder height and your fist and shoulder should be at 90 degrees to each other. A transfer of weight between the front foot and back with the twisting of the hip is essential.
Target Characteristic: Elbow being raised to parallel
Target Characteristic: Hip twist resulting in weight transfer from front to back

Uppercuts

This required the fighter to crouch down into the squat position and throw a punch vertically upwards, with the aim of striking the opponent's chin. Target Characteristic: Sufficient crouching before releasing the punch
Target Characteristic: Directly vertical punch, keeping guard close at all times.

Pose

The above descriptions should illustrate that boxing involves the movement of the entire body and hence it is vital that any system developed considers the whole pose not just hand movement.

2.3 Kinect

This section provides some background information about Microsoft Kinect that is important for understanding the features and limitations of Kinect Analysis. According to Microsoft, the Kinect has worldwide sales of approximately 28 million units and contains an RGB camera, an infrared (IR) emitter and an IR depth sensor as well as a multi-array microphone. The interaction space of the Kinect is limited by the field of view of the Kinect cameras. The Kinect has a 43°vertical by 57°horizontal field of view. The Kinect sensor can be tilted using a built-in tilt motor. Tilting the Kinect increases the interaction space by +27 and −27 degrees. The Kinect sensor provides sensor data in form of data streams. It can capture audio, color and depth data. In addition, it can process the depth data to generate skeleton data. Therefore, the Kinect offers four different data streams that can be accessed: audio stream, colour stream, depth stream and skeleton stream. The streams can deliver at most 30 frames per second (FPS) using a resolution of 640×480 which drops to 12 FPS with a resolution of 1280×960.

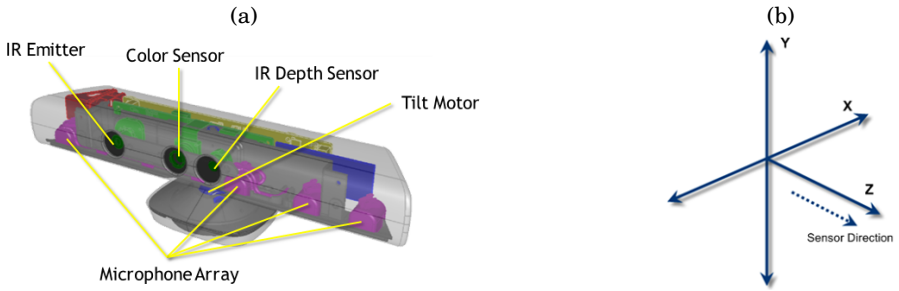

FIGURE 2.1. (a) Kinect (b) Kinect Dimensions

Depth & Infrared Stream

The depth sensor generates invisible IR light to determine an object's depth from the sensor. The primary use for the IR stream is to improve external camera calibration using a test pattern observed from both the RGB and IR camera to more accurately determine how to map coordinates from one camera space to another. [27] The NUI API uses the depth stream to detect the presence of humans in front of the sensor.[8] Skeletal tracking is optimized to recognize users facing the Kinect, so sideways poses provide some challenges because parts of the body are not visible to the sensor. These attributes are a good fit for boxing as both personal assessment and deliberate practice are traditionally done by shadow boxing face on looking at a mirror.

Kinect	Array Specifications
Viewing angle	43° vertical by 57° horizontal field of view
Vertical tilt range	±27°
Frame rate (depth and color stream)	30 frames per second (FPS)
Audio format	16-kHz, 24-bit mono pulse code modulation (PCM)
Audio input characteristics	A four-microphone array with 24-bit analog-to-digital converter (ADC) and Kinect-resident signal processing including acoustic echo cancellation and noise suppression
Accelerometer characteristics	A 2G/4G/8G accelerometer configured for the 2G range, with a 1° accuracy upper limit.

FIGURE 2.2. Kinect Specifications

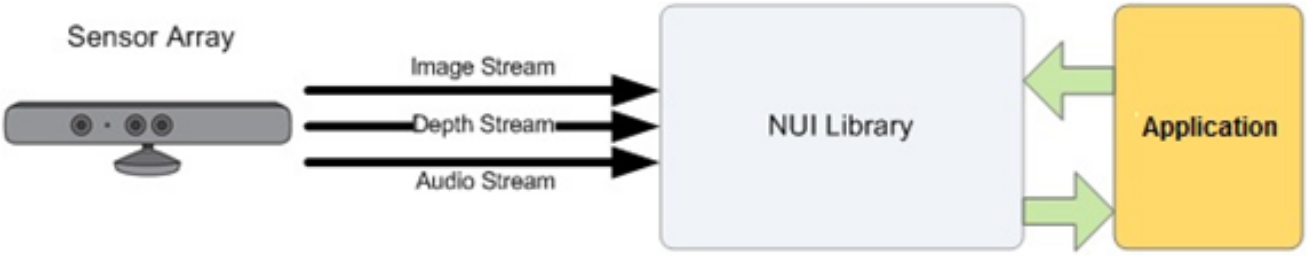

FIGURE 2.3. Kinect Streams

2.3.1 Skeleton & Joint Tracking

The Kinects default tracking mode can track up to 20 joints per skeleton providing the subject is standing relatively face on and are fully visible to the sensor. Although the Kinect is capable of different modes (e.g. sitting) this is not relevant in the context of my project. Each skeleton frame contains the position of each joint as well as information about the tracking quality. Joints can have one of three different tracking states, Not Tracked (0), Inferred (1) and Tracked (2), this flag is useful as an indicator of the quality of the measurements you are receiving for a particular joint. When possible, tracked joints are used to help calculate the position of those joints that cannot be directly tracked hence the ability to infer joints. The Kinects default tracking mode is designed to track people who are standing and fully visible to the sensor. The default range requires skeletons to be at least 80 centimetres away from the device to be tracked properly.

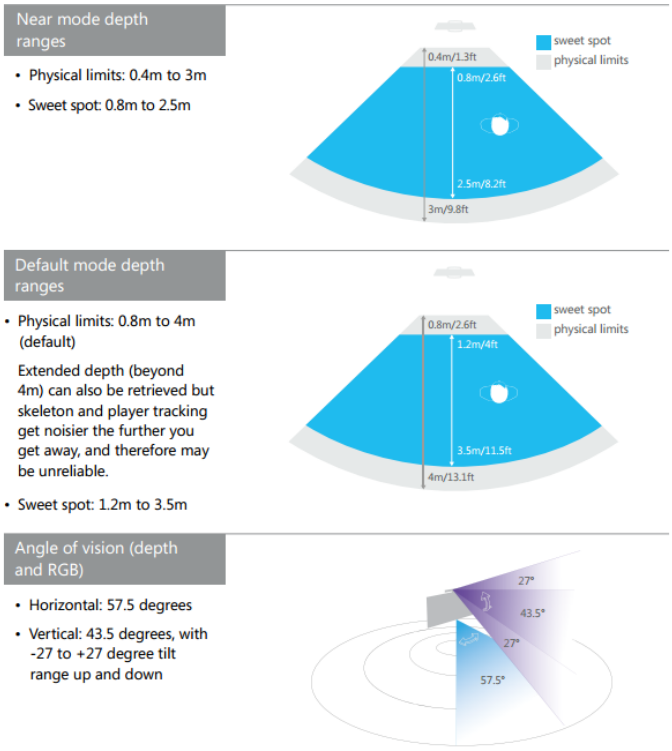

FIGURE 2.4. Kinect Depth Specifications.

2.4 Literature Review

The Kinect is currently a rich area of research in its own right and it straddles the fields of Image Processing and Computer Vision as well as Human computer Interaction which are exciting and popular areas of research. Recognising punches is an example of activity recognition in which there has been a fairly large body of work using the Kinect since it was released to PC in 2012 by Microsoft.

With the development of such a low cost device researchers have shifted their focus onto action analysis and gesture recognition based on depth maps. I wanted to see what was possible with the Kinect at the cutting edge of research so I consulted the most recent papers from the conference on Computer Vision and Pattern Recognition (CVPR), International Conference on Computer Vision (ICCV) & European Conference on Computer Vision (ECCV) as well as an array of other resources to find the most relevant and useful research.

Examples of current work are a customisable hand gesture recognition interface [7], physical therapy instruction [11][10], Karate instruction [6] and quality assessment of the reach and grasp of stroke survivors [41].

Oya Çseliktutan developed a graph-based method for aligning two dynamic skeleton sequences with the aim of action recognition and 'objective quantification of goodness' of said action.

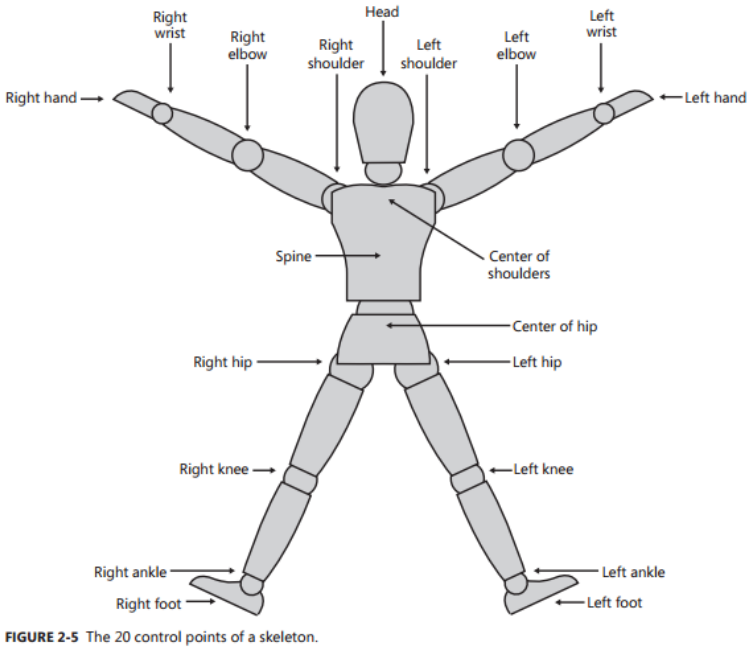

FIGURE 2-5 The 20 control points of a skeleton.

FIGURE 2.5. Kinect Joint Tracking Skeleton.

[9] Depth maps are used similarly to the real-time skeleton tracking algorithm developed by Shotton. [37]

In 2013 a controlled study of 16 people who were blind or who had low vision was run to test the usefulness of Eyes-Free Yoga: An Exergame Using Depth Cameras for Blind & Low Vision Exercise. The purpose of this was to teach the participants yoga poses using audio feedback and received a positive response from participants.[33] The study took the Kinect joint positions and calculated joint angles while using heuristics for each pose. However the study only measured success using 6 unique static poses that were required to be held for an extended period of time. In contrast boxing requires fast repeated movements with many of the movements sharing similarities which make them harder to differentiate. Therefore I did not find this useful for evaluating the feasibility of my project.

The Kinect has also been used to evaluate salsa dancers performance with comparison to a professional dancer in real-time[2][36]. A score was achieved by adding three different metrics, one from the correlation coefficient of quaternions, another using joint velocities and finally a "3D flow Error," calculated from frame vectors. Quaternoin correlation was then used to estimate the time-shift between two dancing sequences.

The experimental results were encouraging, with most of the scores consistent with real-life rankings with the exception of a few poor results due to bad skeleton calibration and tracking. This work draws strong parallels in what I am trying to achieve and demonstrated that the Kinect was a viable option for my work.

Work on disc throwing performance[45] also showed some promise with limited success with the lower ability groups improving their movement. This approach simply used joint angles to measure 5 different phrases of the throw and compare that to joint angle rules. I found this to be overly simplistic in comparison to the project I am trying to achieve although it did indicate that joint angles may be a useful technique for judging movements.

Xia took each joint position and represented this using a histogram. [24] The joint positions were converted from Cartesian co-ordinates to spherical co-ordinates and splitting the sphere into several grids. A sequence of poses was recorded and represented as histograms before using Hidden Markov Models for classification. Some of the concepts used in this work were adapted from Li [42] in particular the 'bag of 3D points' used to characterise poses.

Previous implementations have used joint weighting which have been shown to increase recognition performance with weighted dynamic warping[35] and decision forests [23]. Joint weighting is also used with a newly proposed skeletal representation 'Sequence of most informative joints (SMIJ)', where at each time instance the most informative skeletal joints are selected depending on the action being taken [30]. In this work each sequence was divided into small temporal sequence (frames), taking into account joints with maximum variance and represented these 'most informative joints' with a histogram for action recognition.

Raptis proposed a different skeleton representation which was designed for 'recognition robustness under noisy data'[32] alongside a cascaded correlation-based classifier and a distance metric produced by dynamic time warping, that will match the performed gesture to the closest matching gesture.[22] PCA is applied on only the torso joint and used to estimate the orientation of the skeleton which allows specific relevant features to be extracted.

The most recent research in real-time action recognition methods have been built on random decision forests.[23] [29] Miranda's 2012 paper[28] and Raptis's paper [32] both use SVM's to identify key poses before passing them to the forest to classify. A comparative summary of 'model-based method for body pose estimation and tracking'[25] show SVM to be a popular and effective technique used in 2013.

Automated segmentation

Using RGB-**d** video I have encountered very few methods for automatic action quality assessment although a few approaches have been proposed based on skeleton tracking. For example Bianco and Tisato recognize Karate moves based on skeletal joints.[6]

For each strike a triplet of joints are manually selected and can then be represented by the angles of the joints. K-means clustering is used to obtain a set of key poses and Dynamic Time Warping attempts to align sequences of poses. The distance measure from DTW is then used as a performance evaluation. However the quality of this paper did not seem to me to meet the same standards as many of the others reviewed and the poses used were very static and very different to each other. Given this I would expect a good classification performance since there should not

be too many common features. Since boxing has a much more ‘closed’ guard and the punches share many similarities I am not convinced that k-means clustering will produce good results.

2.5 Existing Products

UFC Personal Trainer

The closest commercial product is ‘UFC Personal Trainer’ a very broad commercial Xbox game aimed at introducing people to UFC. After evaluating this product it became clear to me that it’s main focus is exercise regimes rather than technical fighting. Therefore it does not offer the preciseness or technical fighting focus that I require. In this game your technically wrong punches will still register and several movements are unrealistic.

Kinect sports boxing game

This game has very poor punch discrimination and does not require any sort of real boxing ability. The goal here is usually to punch towards the Kinect controller as quickly as possible. It fails to recognise properly thrown hooks and uppercuts and translate those into the game. Games such as these give people the illusion that punch recognition is solved which is far from the truth. It is likely that this game uses a combination of heuristics and hand tracking which is acceptable in a game designed to be fun for a total beginner but has no place when actually assessing or recognising real boxing technique. As discussed earlier it is crucial that the whole pose is considered not just the writs tjoints.

Fighters Uncaged

This game has generally poor reviews from reviewers, with most complaints involving the reliability of the Kinect to accurately measure fighting moves. [19] “When the fights actually start, pulling off moves becomes a series of desperate flails, trying to get the game to recognize your actions.”[16] and “The game fails to register most of the movements”, “The idea was great for Kinect, but something went horribly wrong along the way. Kinect is supposed to register your every kick and punch, but it only catches about the half of them.”, “the Kinect control is lazily implemented.”[26]
Since this was meant to be ‘flagship’ game for the Kinect it is unlikely the development team decided to ‘lazily implement’ their tracking algorithm; it’s much more likely that in fact the problem is just very hard.

Literature & Product Conclusion

From my research and product comparisons I concluded that the Kinect was a viable option for my project especially since it has an easily accessible skeleton stream and is worth pursuing.

Crucially however there has been no research specifically into the area of boxing with the Kinect which brings it's own unique challenges. Boxers are trained to be fast, well guarded and to give very little away in their movements, especially punches. Therefore many of the punches and poses are very similar since the goal is to be naturally evasive. This will make segmenting punches and giving useful quality metrics challenging in it's own way unlike say a discus throw. A boxers stance is often quite 'closed' so I anticipate challenges tracking obscured joints and the overall accuracy of inferred joint positions. However I could see from the above studies that hip movement could be effectively tracked which was my main concern when evaluating the Kinect since all boxing moves rely on the hip rotation.

It is clear that this will not be an easy project. Many years of Microsoft research have gone into the current Kinect but surprisingly games for their flagship console fail to provide tracking accuracy that satisfies consumers. Furthermore no commercial products exist that incorporates proper boxing technique into a game, suggesting there may be limitations with the hardware that prevent this or that it it simple difficult to do. However the research in this area has shown sufficient promise for me to combine my boxing and computer science knowledge to work on this problem.

2.6 Dimensionality

Curse of Dimensionality

The curse of dimensionality, first discussed by Richard E Bellman in his book Dynamic Programming[31] is the term for a set of problems that occur when using high dimensionality data. As dimensionality increases as does the search space, resulting in the available data become sparse. In order to obtain accurate, reliable and statistically sound results the total amount of data required can grow exponentially in relation to the dimensionality. Likewise the organisation and searching of non-reduced data becomes difficult, with space and search time dependent on data volume. Searching and organising data also relies on the ability to group instances into groups that share similar characteristics, unfortunately if the data appears to be dissimilar due to sparseness it can prevent grouping strategies.

Algorithms that can successfully deal with high dimensionality data typically will have high time complexity and will not always produce more accurate results than algorithms that work on the lower dimensionality data. Therefore it is sensible to look at some dimensionality reduction techniques which might produce better results.

Furthermore since the long-term goal is to give feedback with live data, speed is of the essence. Therefore dimensionality reduction is an important component required to decrease the processing time on any input.

2.6.1 Dimensionality Reduction

Dimensionality Reduction is the process of reducing the number of variables under consideration for any given problem. For example, each frame from the Kinect is represented by 60 data points: the x,y,z co-ordinates of the 20 skeleton joints. Considering the Kinect is capable of 30 FPS that is 1800 data points per second of movement. With long sequences of recordings this could become a huge amount of data to process, most of which could be represented in a reduced dimensionality. Furthermore the range of poses exhibited by a boxer is fairly limited and so it it likely that they can be more compactly represented.

2.6.2 Principal Component Analysis (PCA)

Principal Component Analysis is a statistical procedure that transforms a set of observations of potentially correlated variables into a set of linearly uncorrelated variables called principal components. The number of principal components should always be less than or equal to the number of original values with the first principal component having the largest possible variance. Each following component will attempt to represent as much variance in the data as possible. In my case I will be looking to reduce my 60 data points per frame into a low dimensionality set that will help me to uniquely identify punches. As PCA is a linear method it is easy to project

poses back into their own original space and visually inspect the original and the projected one to see what data has been lost. (See Fig 3.7)

Eigenvectors & Eigenvalues

The eigenvector of a matrix M is a vector V such that $M \times V$ yields a constant multiple of V, the multiplier of which is denoted by λ.

$$Av = \lambda v$$

λ is known as the eigenvalue of M which corresponds to v. Any multiple of an eigenvector will have the same eigenvalue as as original eigenvector.

The principal components produced by PCA correspond to the principal eigenvectors of the covariance matrix which are calculated by eigenvalue decomposition. The principal eigenvectors are those with the highest eigenvalues and so are easily deduced.

Covariance Matrix & Optimisations

If the data is represented in a column vector each with a finite variance then the covariance matrix $\sum$ is the matrix whose (i,j) entry is the covariance

$$\sum_{\mathbf{ij}} = COV(X_i, X_j) = \mathbf{E}[(X_i - \mu_i)(X_j - \mu_j)]$$

$$\mathbf{X} = \begin{bmatrix} X_i \\ . \\ . \\ X_n \end{bmatrix}$$

Where:

$$\mu_i = E(X_i)$$

and

$$\mu_j = E(X_j)$$

The is equivalent to

$$\mathbf{N} = \frac{1}{n-1} \sum_{\mathbf{i=1}}^{\mathbf{n}} E[(X - E[X])(X - E[X])^T]$$

Where **N** is the covariance matrix, **n** is the number of frames. $\frac{1}{n-1}$ has been added to make the estimate unbiased.

The number of data points in each frame is 60 and the Kinect is capable of 30 FPS which over 120 seconds produces 216,000 data points to represent that sequence. A matrix of this size is costly and slow to find eigenvectors for. Let us say that 60n is the number of data points for frame and m is the number of frames. Take X as size (d x 60n) and X^T as (60n x d) meaning $X^T X$ is of size (60n x 60n), while XX^T is of size (d x d). It is however known that the eigenvalues for $X^T X$ are the same as for XX^T which is useful since this is of a much smaller size and hence less costly.[38] The eigenvectors (v_i) of $X^T X$ are related to the eigenvectors (U_i) of XX^T by the equation

$$V_i = X^T U_i$$

with V_i needing to be normalised afterwards. This method can reduce the eigenvalues and eigenvectors produced from a maximum possible N^2 for X^TX to M values for X^TX.

The M eigenvalues of XX^T will be identical to the first M largest eigenvalues from X^TX. This is an important performance step since the goal of my system is to run in real time and this will reduce the time taken.

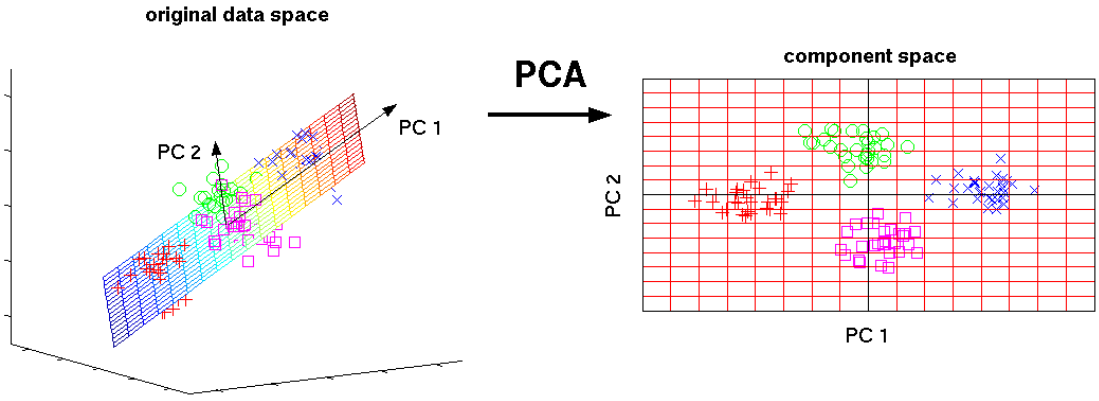

FIGURE 2.6. PCA example showing the 1st and 2nd largest **eigenvalues** in the original space before being converted into the new component space **eigenspace?**

2.7 Manifold Learning

Manifold learning is a Non-Linear Dimensionality Reduction process that transforms high dimensionality data that typically requires multiple dimension to represent it which is difficult to interpret. To simplify the data it is possible to assume that the data lies on an embedded non-linear manifold within the high-dimensionality space. Assuming the manifold has low enough dimensions it can then be visualised in this lower-dimensionality space.

2.7.1 Diffusion Maps

Diffusion maps are a non-linear and relatively new technique developed in 2006 by Ronald R. Coifman and Stephane Lafon.[13] The aim of a diffusion map is to provide a framework for finding meaningful geometric descriptions of data sets. Diffusion maps are capable of turning high dimensionality data into low dimensional structure. Unlike other dimensionality reduction techniques like PCA, diffusion maps attempt to discover the underlying manifold, a lower dimensional constrained surface containing the data. Diffusion maps are based on defining a Markov chain on the graph of the data. By performing this for a set number of time steps a measure for the proximity of data points is obtained, which is used to define a diffusion distance. Pairwise diffusion distance are retained as well as possible in the dimensionally reduced data.

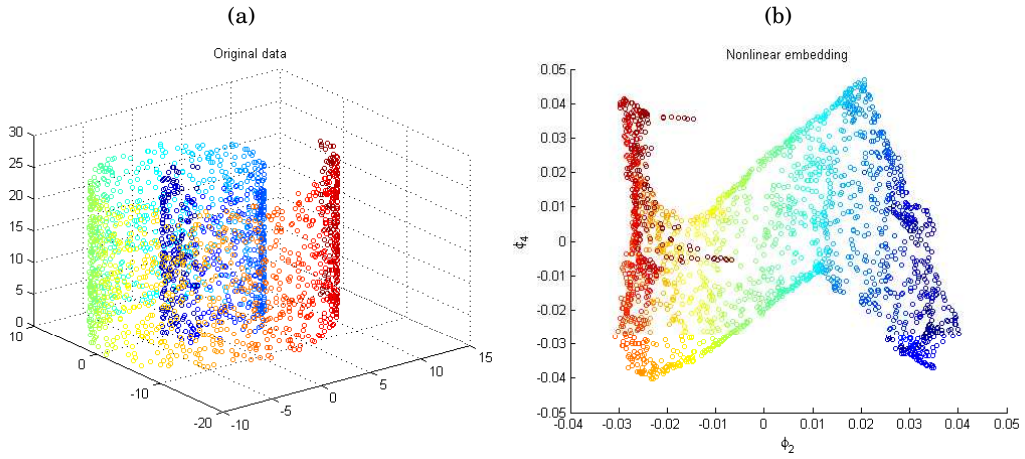

FIGURE 2.7. (a) Original example data (b) Non-linear embedding

2.7.2 Locally Linear Embedding (LLE)

In LLE a data manifold is constructed by finding a set of the nearest neighbours for each point[34]. Together they are used to compute a set of weights that describes each point as a linear combination of its neighbours. Finally, it uses an eigenvector-based technique to find the low-dimensional embedding of points, such that each point is still described with the same linear combination of its neighbours. Furthermore, the preservation of local properties allows for successful embedding of non convex manifolds. LLE tends to handle non-uniform sample densities poorly because there is no fixed unit to prevent the weights from drifting as various regions differ in sample densities.

2.7.3 Laplacian Eigenmaps

Laplacian Eigenmaps find a low-dimensional data representation by preserving local properties of the manifold.[5] Similar to LLE, a graph is built from neighbourhood information, with the distance between a point and it's K nearest neighbour is minimised. A weighting system is used such that in the low-dimensionality space the distance from a point to it's nearest neighbour is more significant to the cost function than other nearby points. Put simply the closer a neighbour is to the selected data point the heavier its weighting. The goal overall is to minimise the cost function based on the graph information to ensure that points close together in the high dimensionality data remain so after the reduction, preserving local distances.

2.7.4 Local Tangent Space Alignment (LTSA)

LTSA is based on the intuition that when a manifold is correctly unfolded, all of the tangent hyperplanes to the manifold will become aligned.[46] In other words there will exist a linear mapping from high dimensionality data to a local tangent space which will have a linear mapping to a low-dimensionality data point. It begins by computing the k-nearest neighbours of every point. It computes the tangent space at every point by computing the d-first principal components in each local neighbourhood. It then optimizes to find an embedding that aligns the tangent spaces.

2.7.5 Curvilinear Component Analysis (CCA)

CCA is a learning algorithm that starts with larger distances before iterating to smaller ones.[15] It looks to output a configuration of points that preserves the original distances as much as possible while focusing on the smaller distances.

The large distance information will be overwritten by the smaller distance information unless a conflict occurs. The stress function of CCA is related to the sum of Bregman divergences which aim to generalise squared euclidean distances so they all share the same properties. If compromises between the larger and smaller distance information but be made the stress function determines this.

2.8 Segmentation Methods

Automatic segmentation is a crucial step in this project since it will be used too find the beginning and end of each punch. If this is used on a larger scale then much greater data sets will need to be used and collated from multiple sources. If these can be processed and automatically segmented this will remove any manual work required and make this a truly useful system. I will look at several methods that might aid me in finding meaningful characteristics to separate each punch.

2.8.1 Dynamic Time Warping (DTW)

Dynamic time warping is a time series comparison method that takes two temporal sequences, often which vary in time or speed and tries to calculate an optimal match. The two sequences are aligned and a similarity score is produced. This technique can be used on any data that can be represented in a linear fashion and has been successfully applied to fields such as signature recognition, voice recognition, partial shape matching and gait analysis. For example in gait analysis, similarities could be detected in the walking pattern of two people even though their speed and acceleration may be difference. If a punch sequence is similar enough to others of the same type and different enough from other types of punch this could prove a useful method

of recognising the start and end of a punch as well as the type of punch. Specifically a sliding temporal window in the punch stream could be compared to known punch templates.

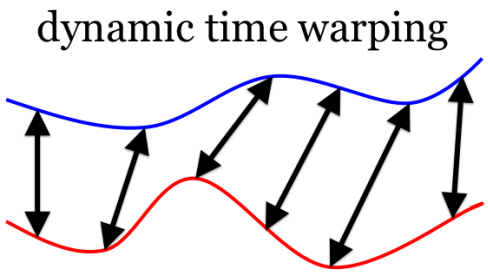

FIGURE 2.8. Dynamic time Warping visualisation

2.8.2 Fourier Transformation (FT)

The Fourier Transform is a mathematical transformation that can decompose a time series from the time domain and transform it into the frequency domain. Any wave-like functions can be represented as a combination of simple sine waves by breaking down the original series into the sum of a set of oscillation functions, sines and cosines. The series is split into a magnitude spectrum which will comprise of complex exponentials and a phase spectrum which will encode angles in radians. The Fourier series of a function $f(x)$ is given by:

$$f(x) = \frac{1}{2}a_0 + \sum_{n=1}^{\infty} a_n cos(nx) + \sum_{n=1}^{\infty} b_n sin(nx)$$

Where:

$$a_0 = \frac{1}{\pi}\int_{-\infty}^{\infty} f(x)dx$$

$$a_n = \frac{1}{\pi}\int_{-\infty}^{\infty} f(x)cos(nx)dx$$

$$a_n = \frac{1}{\pi}\int_{-\infty}^{\infty} f(x)sin(nx)dx$$

Here the cosine coefficient is a_n and the sine coefficient is b_n.

A property worth noting is that a shift in the time domain corresponds to a linear change in phase but does not alter the magnitude spectrum. Since I have demonstrated that the projection coefficients for the first principal component produces a continuous sinusoidal-like function I can use the discrete-time Fourier transform (DFTD) to obtain the Fourier coefficients. It is possible that these coefficients will be unique enough for each punch class and so it can be used to infer which class a new punch belongs to. I could also look at this for feature extraction. (See p30)

2.9 Classification Methods

The efficacy of a number of machine learning methods were empirically assessed, the following will give a outline of these techniques.-

2.9.1 Support Vector Machines (SVM)

A Support Vector Machine is a kernel-based method that is effective for highly dimensional datasets. The SVM uses a kernel to calculate the scalar product of two feature vectors in a high dimensional feature space. The decision function uses the hyper-planes, defined by the Support vectors, to classify the data. Only significant samples are taken for use as support vectors so that a high variance will make less of a different to the accuracy of the model. This is well suited to a recognition task as the same class of punch will be thrown slightly differently depending on body size, tiredness and expertise.

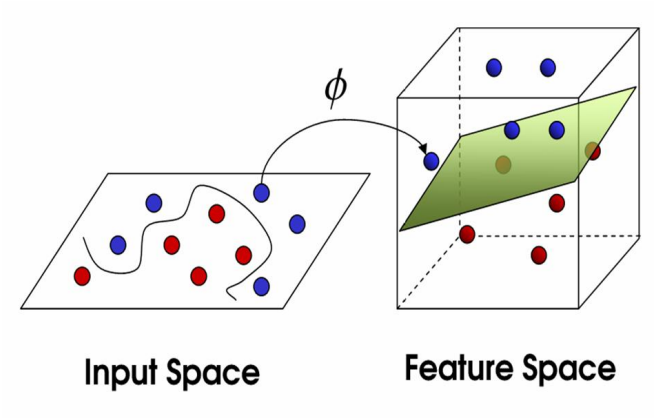

FIGURE 2.9. Principle of Support Vector Machines (SVM)

2.9.2 Neural Networks

Neural networks are models based on the parallel processing of information similar to that of the brain. A NN can be configured for different applications such as clustering, pattern recognition, Dynamic Time series and curve fitting. They are able to derive meaning from complex data that human beings would be unable to notice and that other techniques will fail on. Crucially the networks are capable of approximating non-linear functions unlike DTW. Neural networks learn by examples in the form of training data and are adaptive based on a system of weightings calculated by % error. Other characteristics are their ability to self organise and the ability to work in real-time if sufficient parallelism is supported.

Below is a model of a simple neuron with many inputs and a single output. This is based on the biological models of a single neuron where a certain injection of current will decide if the

neuron should fire. The 'Integrate & Fire' model is as follows:

$$C_m \frac{dv}{dt} = -G_l(V - E_l) + I_e$$

With time constraint:

$$\tau_m = \frac{C_m}{G_l}$$

And membrane resistance:

$$R_m = \frac{1}{G_l}$$

IF:

$$V > V_t h'$$

THEN: Generate a spike.

$$V = V_r eset$$

In this scenario the neuron has two modes of operation, testing and training. In the training mode the neuron can be trained to fire depending on a particular input pattern. In the testing mode when a recognised input pattern is detected the associated output from the training becomes the output. If the pattern is not recognised in the list of taught input patterns then it tries to identify that which is closest to it (which it does recognise) and produce an output matching that.

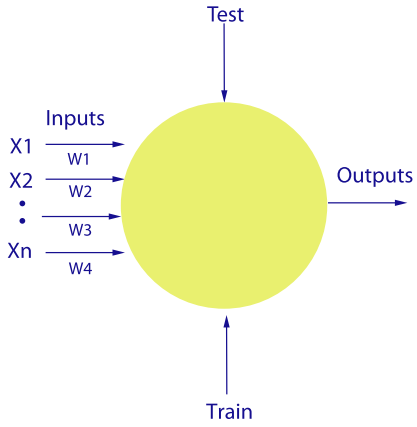

FIGURE 2.10. A model of a single Perceptron

Back Propagation (BP)

Neural networks (NN) work by connecting a series of single neurons into a learning network. Back Propagation is a learning algorithm whereby the network behaves similarly to above but with the added complexity that each input is given a weighting. This weighting is calculated iteratively by working out the error for each neuron like so: $Error_a = Ouput_a(1-Output_a)(Target_a - Output_a)$ which put simply is the difference between the correct output and the actual output for neuron A.

Since BP networks learn by example they must be given labelled training data which is used to correctly calculate the weights given to each input to give the required output. Since BPNN are ideal for pattern recognition and mapping tasks it could be suitable for use on punch sequences.

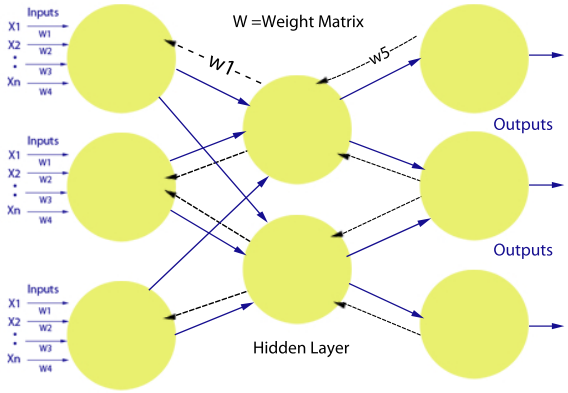

FIGURE 2.11. A model of a Neural Network using the Back-Propagation algorithm.

2.9.3 Decision Trees & Random Forest

A predictive tree like model which maps observations about an object to reach a conclusion about its target value. In this model the 'leaves' represent class labels while the branches themselves represent conjunctions of features that will lead to the class labels. Put simply each condition in an internal node while each outcome is an external node. Information gain is the difference between the initial entropy and the new entropy after following a branch along the decision tree. Since the goal of any machine learning technique is to achieve a low value of entropy to make accurate predictions, the decision tree is constructed such that each branch has the maximum possible information gain. The data is split by each feature that has the maximum information gain recursively for each branch.

Bagging is a method of assigning a measure of accuracy to sample estimates. We sample from an approximating distribution and try to approximately calculate the properties of the estimator based on this. We measure a statistic from a sample of the population and then use this to say something about the whole population. For example, we might say that for our set of data a subset is used to determine a class. If any sample in the entire population follows the same tree rules then it will be labelled as in that class. We have used bootstrapped samples in our classifiers as they have been constructed using random sampling with replacement. This method is designed to improve the accuracy of machine learning algorithms, helping to reduce variance and hence over fitting.

Random forest is an averaging algorithm which produces a diverse set of classifiers by introducing randomness in the classifier construction. Each tree is built from a bootstrapped

sample from the training set. During construction the node splits are chosen based on the best split among a random subset of features, not the best split among all the features. Due to the randomness of this algorithm we expect it to have a slight bias compared to other decision tree models, however due to the averaging we expect a lower variance.

These Ensemble methods combine the predictions of multiple trees to come to a consensus.

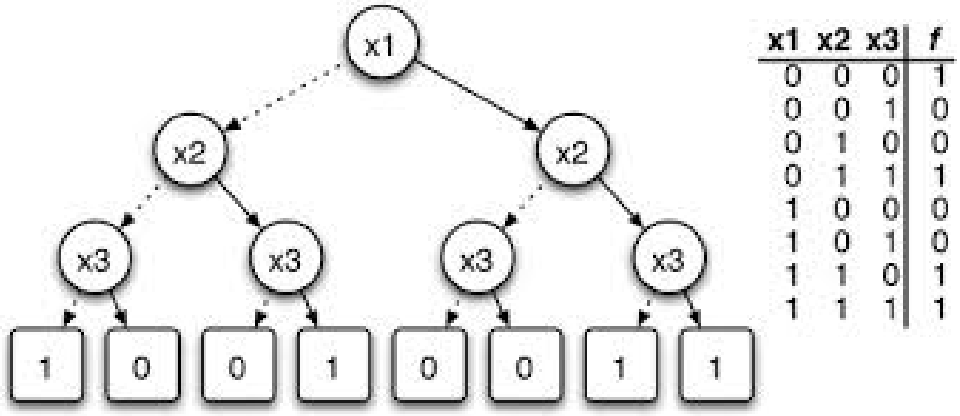

FIGURE 2.12. Decision Tree Diagram

2.10 Punch Quality Assessment

I will be measuring quality in reference to a 'ground truth' dataset recorded from local professionals. My aim is to create rules that are capable of detecting basic mistakes and provide feedback. I will start by targeting the characteristics as mentioned in the earlier background chapter, for example throwing a jab with the elbow sticking out instead of tucked is a classic beginners mistake. At the very least I expect to be able to classify a mixed set of punches as 'good' and 'bad'.

CHAPTER 3

MY APPROACH

This chapter will discuss each step of the implementation stage such as the collection of skeleton data, dimensionality reduction comparison, punch segmentation, punch classification and quality assessment.

3.1 Data collection

Once I had submitted my study to the University Ethics Committee,s I visited 'Bristol Boxing Gym' and 'Broad Plain Boys Club' to record punch sequences from a variety of different experience levels and body types. In some cases I instructed the participants to try to throw the punches as precisely as possible, paying attention to good form. In others I asked them to make intentional beginner mistakes, such as allowing the left elbow to stick out when throwing the jab. I also asked the participants to try to keep their punches evenly spaced out when throwing them and to stand in approximately the same location as each other.

Skeletal data is recorded using the Kinect and stored as a space separated text file, with each line corresponding to one timeframe. The structure of a line is:

$$trackingflag\ x_0\ y_0\ z_0\ \ trackingflag\ x_1\ y_1\ z_1 \ldots\ trackingflag\ x_{19}\ y_{19}\ z_{19}$$

where x_i, y_i, z_i are the x,y,z coordinates representing the position of the ith joint. A large value that could not be a Kinect measurement is used as a flag to signify the beginning of a newline. The tracking_flag is an integer which describes the status of the joint:
Joint not tracked = 0, Joint position inferred = 1, Join position tracked = 2.
If the joint is not tracked the position is set to (-10000, -10000, -10000) and is not used. All joint's positions are measured relative to the camera, which is considered to be at position (0,0,0).

Joint Number	Joint Name
0	NUI_SKELETON_POSITION_HIP_CENTER
1	NUI_SKELETON_POSITION_SPINE
2	NUI_SKELETON_POSITION_SHOULDER_CENTER
3	NUI_SKELETON_POSITION_HEAD
4	NUI_SKELETON_POSITION_SHOULDER_LEFT
5	NUI_SKELETON_POSITION_ELBOW_LEFT
6	NUI_SKELETON_POSITION_WRIST_LEFT
7	NUI_SKELETON_POSITION_HAND_LEFT
8	NUI_SKELETON_POSITION_SHOULDER_RIGHT
9	NUI_SKELETON_POSITION_ELBOW_RIGHT
10	NUI_SKELETON_POSITION_WRIST_RIGHT
11	NUI_SKELETON_POSITION_HAND_RIGHT
12	NUI_SKELETON_POSITION_HIP_LEFT
13	NUI_SKELETON_POSITION_KNEE_LEFT
14	NUI_SKELETON_POSITION_ANKLE_LEFT
15	NUI_SKELETON_POSITION_FOOT_LEFT
16	NUI_SKELETON_POSITION_HIP_RIGHT
17	NUI_SKELETON_POSITION_KNEE_RIGHT
18	NUI_SKELETON_POSITION_ANKLE_RIGHT
19	NUI_SKELETON_POSITION_FOOT_RIGHT

3.2 Exploring Hand Movement & Periodicity

My first step was to organise the data in such a way that it could be usefully visualised so that I could comprehend the nature of the data that needed to be processed. I began by recording a sequence of 5 jabs, being careful to evenly time the punches before manually isolating the left wrist joint over time. In this instance the timing was important as my aim was to produce a sinusoidal-like sequence with five full 'cycles' demonstrating the movement of the left hand over time.**(Fig 3.1)** This result was promising since I could now confirm that punches could be represented by a periodic sequence. As a proof of concept isolating a single joint was a sensible first step however this cannot be reliably used to perform action recognition since it would be a single point of failure. If for example the joint was obscured or the Kinect mis-tracked the joint you would lose the ability to recognise an action. Since a boxing punch is based on the transference of weight and the rotation of the hip the whole pose must be considered to ensure other periodic aspects are not missed. The use of dimensionality reduction offers increased robustness against these types of problem since it allows the entire skeleton to be represented in low dimensionality data and as such will not be as sensitive to noise or failures to track.

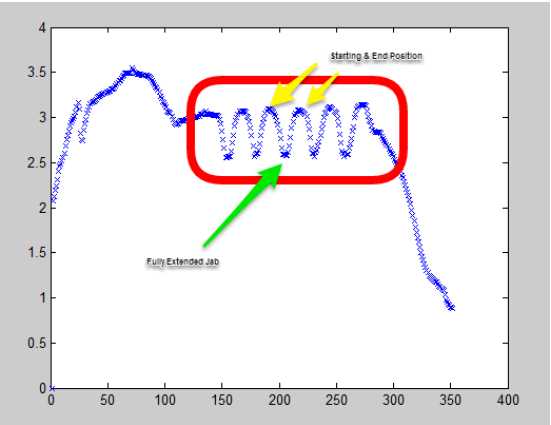

FIGURE 3.1. This shows the left wrist joints movement in the z direction over time. The x-axis represents time (kinect frames) and the y-axis the distance of the fist from the Kinect

3.3 Pose Representation

Next I needed to find a meaningful way of visualising the entire skeleton over a period of time, since I then knew that the pose as a whole needed to be represented and assessed. I naïively began by plotting the entire data set, which produced a reasonably jagged signal with two troughs followed by a large peak before levelling of again. **Fig 3.7 (a)**. At this point I realised attempting to plot the skeleton data in this way would not be useful, but as a test I differentiated the raw data to produce a velocity as opposed to distance. My hope was that since velocity is position invariant it would be a better choice than using the raw co-ordinates. Unfortunately I found that the velocity introduced more noise and due to this it was clear that it was going to be much more difficult to segment later on.

Realising that for now I needed to look at individual joints I took the central hip joint and plotted its movement in the z direction over time as well as plotting the differential of that to give me the velocity. As you can see from **Fig 3.7 (b)**, the velocity data was much more useful and produced a reasonably smooth periodic sequence. Since the initial recording was 10 jabs one after the other I expected a periodic, cyclical signal to be produced.

Since PCA was a familiar method I implemented it in Matlab using three principal components and plotted the first projection coefficient for the first principal component with both the distance and velocity data. From this we can see that the distance data was slightly jagged with a local maxima which might obstruct attempts to automatically segment punches. The velocity data produced a smoother series but introduced unwanted noise towards the beginning and end of a sequence. **Fig 3.2**

Next I plotted all three projection coefficients for the three principal components for both distance and velocity. **(Fig 3.8)** We can see for the first projection coefficient both distance and

velocity are similar with distance being slightly more jagged at the peaks of the punching period and velocity being more jagged at the start and end. For the second they are both fairly similar but with distance having less peaks and troughs and with only one dramatic spike at 250 as opposed to two at 50 and 275. For the third the depth comes out ahead with a smoother signal which again has less 'spikes' (which will make it hard to segment.) Although by eye it looks like the depth data under PCA produces a more useful periodic signal neither are particularly convincing so I will have to perform a more precise comparison.

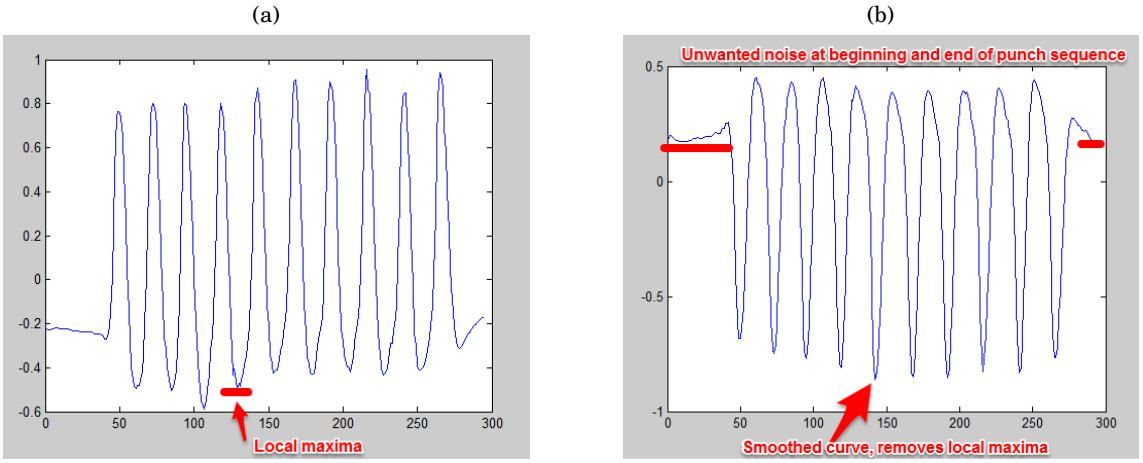

FIGURE 3.2. A plot of the projection coefficient for the first principal component for a jab over time for (a) Distance, (b) Velocity

3.4 Dimensionality Reduction (DR) Comparison

Each dimensionality reduction method was evaluated on its ability to produce useful, smooth and sinusoidal output to aid automatic segmentation. Automatic segmentation is a crucial step in the project since on a larger scale much greater data sets will need to be used and collated from multiple sources. If these can be processed and automatically segmented this will remove any manual work required and make this a truly useful system. I used the Matlab Dimensionality Toolbox and the 'intrinsic_dim' function that performs an estimation of the intrinsic dimensionality of a dataset X given a specific method. This was calculated using six different methods, maximum-likelihood estimation (MLE), correlation dimension estimation (CorrDim), nearest neighbourhood, packing numbers and Geodesic minimum spanning tree (GMST) and eigenvalue estimation. These results were then combined and averaged to produce a dimension **d** to be used as an argument for LLE, Hessian LLE, Laplacian, LTSA, CCA and PCA (see background). The lower dimensionality data was then plotted and analysed. The projection coefficient of the first principal component, plotted for multiple dimensionality reduction methods

shows that PCA gives the most useful results, followed closely by CCA. If we then look at the second projection coefficient we can see that PCA performs second to LTSA but LTSA performs poorly for the first component. This makes PCA the obvious choice as. Although PCA is simpler than more modern techniques, it actually produced the most consistently smooth cyclical signal making it a sensible choice.

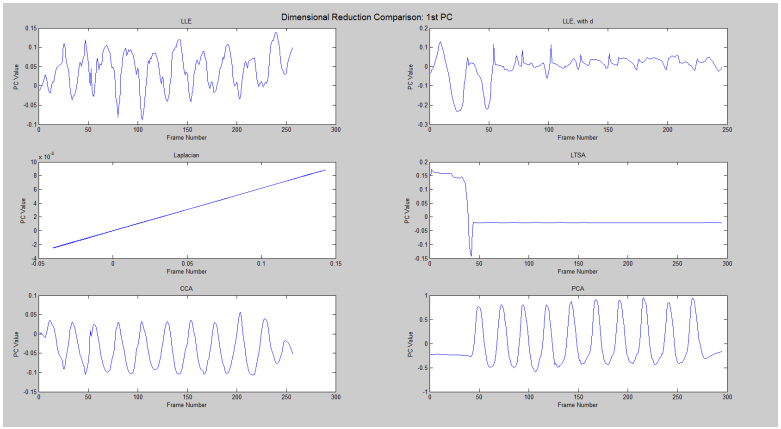

FIGURE 3.3. The projection coefficient for the first principal component as produced by a variety of DR techniques over time

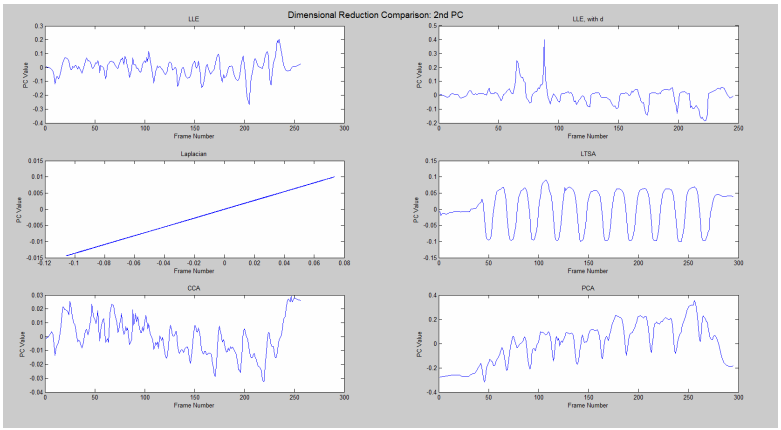

FIGURE 3.4. The projection coefficient for the second principal component as produced by a variety of DR techniques over time

3.4.1 Pose Reconstruction

Another benefit of using PCA is that it is possible to reconstruct the original pose using the equation

$$y = U * a + Xm$$

where U is the eigenvectors, a is the projection coefficients and Xm the mean of the original data. Since we are going from low dimensionality data back to high dimensionality data it is inevitable there will be some reconstruction error. However by plotting this reconstruction along with the original it is possible to visualise how closely the principal components generated represent the original. This is crucial because the decomposition can be verified to make sure that important information is not lost during the reduction process.

3.5 Punch Segmentation Algorithm

After failing to find a way to automatically segment punches using existing methods it became apparent that a custom algorithm will be needed for this task, which can be broken down into several steps.

1. Perform PCA on raw data.
2. Smooth the principal coefficients.
3. Find the local maxima and minima of the first principal coefficient.
4. Remove erroneous minima/maxima using heuristic rules.
5. Take a number of evenly spaced samples from between two maximum points.
6. Fetch the unsmoothed PCA data that correspond to the time the samples were taken.
7. Train classifier on new data.

Each frame representing 20 joint positions is reduced from 60 dimensions to 3 principal coefficients per frame. Next the principal coefficients are smoothed to remove any 'wrong' minima/maxima which would prevent the automated segmentation of punches. All remaining maxima are then passed through a set of heuristic rules that checks the location of a maxima point relative to its neighbourhood points, to determine if it is legitimately the start of a punch. Once this has been achieved a number of evenly spaced samples are taken from between two maximum points resulting in each individual punch being sampled. As the sequence is temporal each sample relates to a time which is then used to fetch the unsmoothed principal coefficients at that time. It is necessary to use the unsmoothed data for classification since this data has been overly-smoothed as its purpose was segmentation. As a result the unique shape and characteristics of each sequence are lost and therefore it no longer accurately represents a pose. Certain elements of the pose such as the movement of the hip joint which would correspond to a rotation in the pose would be crucial to successfully classify punches. Put simply, without using the unsmoothed

data the classifier would fail to find distinct features for each class of punch and so would be unable to classify effectively. **Fig 3.6**

Heuristic Rules

As the punch sequence is sinusoidal-like every maxima should correspond to the beginning of a punch (and the end of the previous one) so it is known that the start of every punch should fall within a certain range of amplitude. This allowed me remove a lot of false positives that had not been removed in smoothing by simply thresholding. Each type of punch has its own unique signature so a different threshold is applied depending on the punch, although there are some common rules that are applied to the left or right hand side of the body. Once this is done each maxima is looked at relative to its neighbours to try to determine if it is likely to be the beginning of a punch. I used the Pythagorean theorem to calculate the distance between a maxima and its neighbours; if the distance is too small it is clear that one of the points is erroneous and that only one of these will be necessary for segmentation. Likewise if a neighbour is too far away it becomes obvious that one of the maxima is incorrect and needs to be removed.

3.6 Classification

Once the data has been successfully segmented and reduced I began to experiment with a range classifiers, as referenced in the background section.

3.6.1 Feature Extraction (PCA)

Once the data has been successfully segmented into individual punches we can begin to extract features that can be used to train a classifier. Twelve to fifteen evenly spaced samples are taken for each individual punch, depending on the classifier to be used later. Each sample corresponds to a point in time which is used to extract the required number of principal coefficients for each point. Taking three coefficients as an example we would have $36 - 45$ features for each punch. Once all the different punch sequences are sampled for each type of punch we can use this data to train a classifier. A multi-class SVM, decision tree, Random Forest and neural network were tested.

3.6.2 Feature Extraction (DTW)

After using DTW I found that there was not enough difference between the various segmented punches to give me an effective way to segment. Comparing jabs to other jabs yielded similarity values from $0 - 1.589$ with an average difference of 0.499, comparing jabs to crosses yielded similarity values from $0 - 2.113$ with the average 0.716. Due to the large spread of values and

similarity in values between different classes of punches there was not enough of a consistent difference in the score to be able to extract features or have an indication of the type of punch.

3.6.3 Fast Fourier Transform (FFT)

I investigated the possibility of running a FFT on the raw data as well as each segmented punch that was obtained from using PCA and my own segmentation algorithm. The range for the coefficients was $0-(63.030-0i)$ with a spread across all of the different types of punches. Unfortunately this meant the coefficients obtained in both cases proved to be a poor differentiator between punches due to the coefficients being so variable even within the same punch types.

3.6.4 Neural Networks (NN)

I used NNs extensively since my initial attempts at classification produced very promising results compared to the other classifiers even when the punch segmentation algorithm was in development resulting in fairly noisy input. I discovered that when using between 1-3 projection coefficients the network size performed better when smaller with 10 Neurons, with 5 not being able to handle the complexity of the data and 20 beginning to over-fit the data to the training set. **Tab 3.1**. By the end of the project when using between 5-7 eigenvectors with a large training set 25 Neurons was the optimal network size.

n-neurons	5	10	15
Run 1	11.76%	11.76%	14.71%
Run 2	8.82%	2.94%	20.59%
Run 3	14.71%	20.59%	20.59%
Run 4	29.84%	8.82%	11.76%
Run 5	17.65%	5.88.96%	5.88%
Avg	**83.44%**	**90.00%**	**85.29%**

TABLE 3.1. Punch classification error of different size neural networks on a six class problem. (Jab, Cross, Left-Hook, Right-Hook, Left-Uppercut, Right-Uppercut)

3.6.5 Support Vector Machines (SVM)

I tested SVM as a technique on a two-class problem and concluding that this technique worked well and was worth pursuing. Unfortunately Matlab does not support a multi-class SVM so I used LIBSVM a support vector machine library. I choose this since the end goal was to have a real-time system and this particular part was implemented in c which dramatically increased performance compared to others that were tried.

After trying several variants of SVM, I discovered the most effective used a radial basis kernel and probability estimates to determine the class that each each input belongs to. It also performed

best on smaller data sets with ten samples (per principal coefficient) **Table 3.2**. However with the larger datasets and increased number of eigenvectors used later in this project I found fifteen to be optimal.

n-samples	5	10	15	20
Run 1	88.13%	93.48%	82.43%	84.78%
Run 2	94.65%	95.65%	82.61%	86.96%
Run 3	92.48%	86.97%	97.83%	93.48%
Run 4	87.13%	86.96%	91.30%	84.78%
Run 5	90.30%	86.96%	89.13%	76.09%
Avg	**85.74%**	**95.65%**	**88.60%**	**85.20%**

TABLE 3.2. Punch classification accuracy of different numbers of samples for SVM on a six class problem using a reduced dataset (Jab, Cross, Left-Hook, Right-Hook, Left-Uppercut, Right-Uppercut)

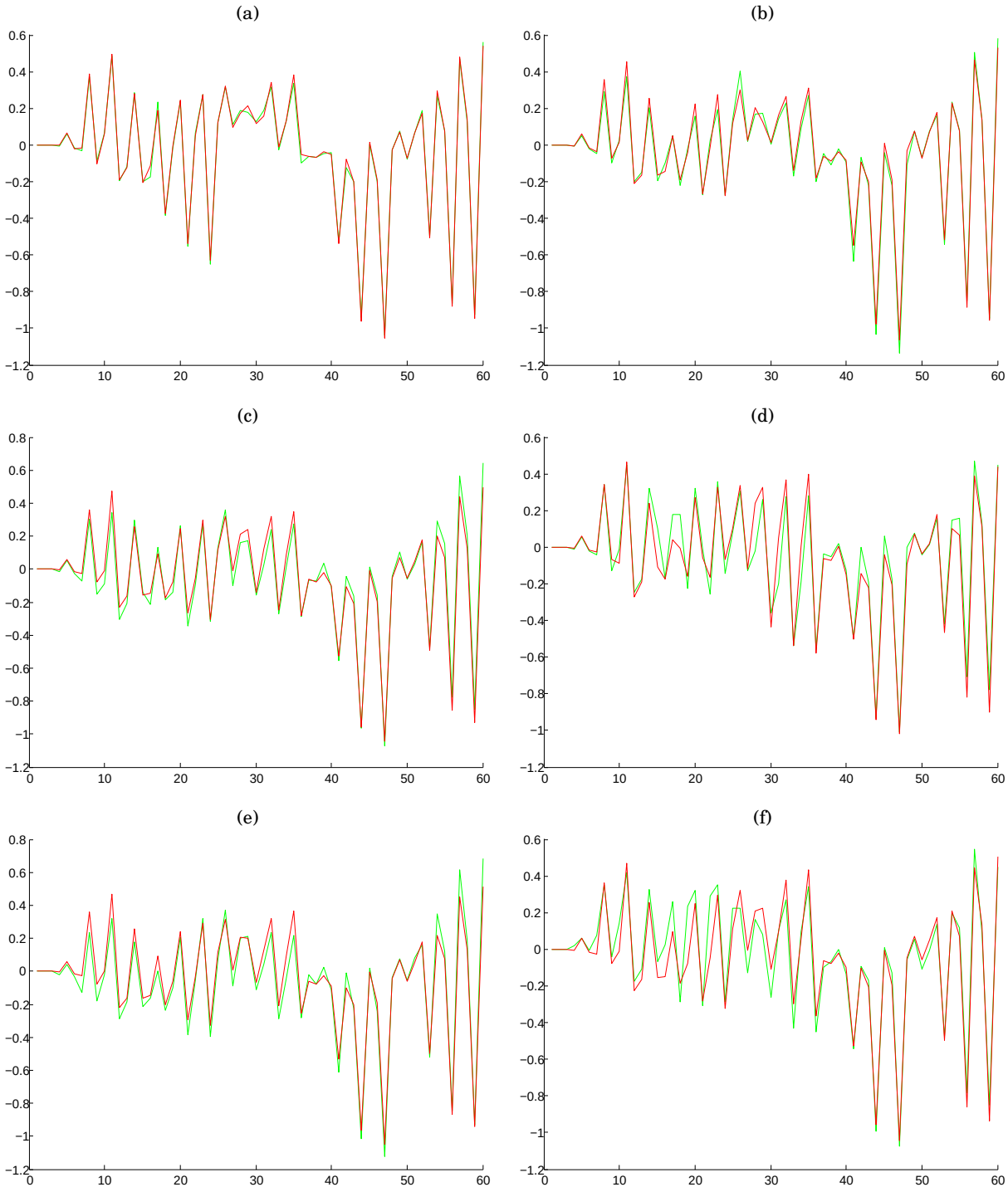

FIGURE 3.5. Each graph represents a single pose with the original data in green and the reconstructed data in red obtained from back propagation. We can see from the reconstruction how much data has been lost as a result of dimensionality reduction. **(a)** Jab **(b)** Cross **(c)** Left-Hook **(d)** Right-Hook **(e)** Left-Uppercut **(f)** Right-Uppercut

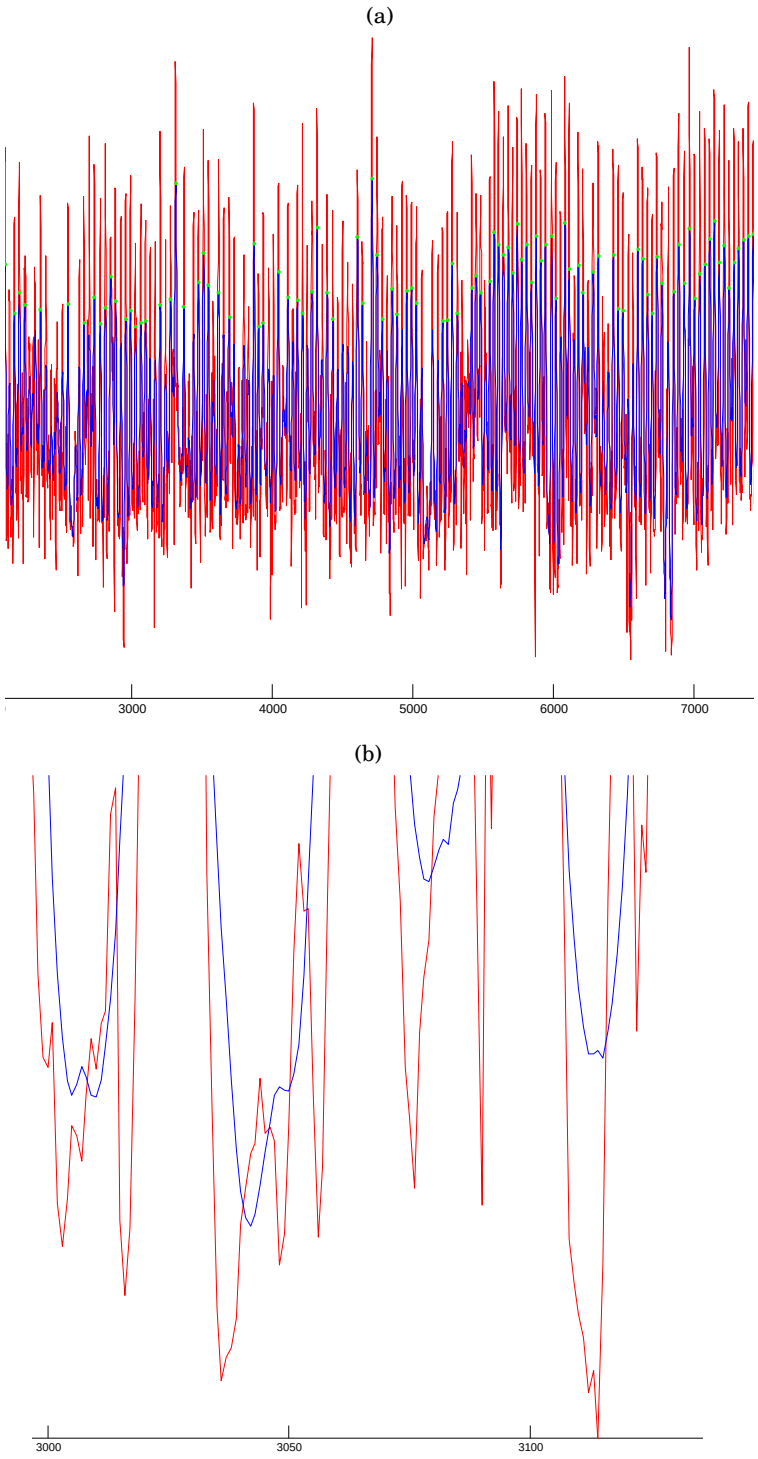

FIGURE 3.6. An overlay of the original principal coefficients (red) and smoothed principal coefficient (blue) for a right uppercut. (a) Displayed is an entire punch sequence; notice that the smoothed sequence covers approximately $\frac{2}{3}$rds the area than that of the original, demonstrating a substantial amount of information loss. Also notice the maxima marked on the smoothed curve correspond to peaks on the unsmoothed sequence. (b) Displayed is a closer look at the half-way point of four individual hooks. Observe the unique shape of the hooks as they increase and decrease several times as they tend to and then away from the minima. Witness the effect smoothing has on this unique characteristic, this would be a very useful feature for a classifier.

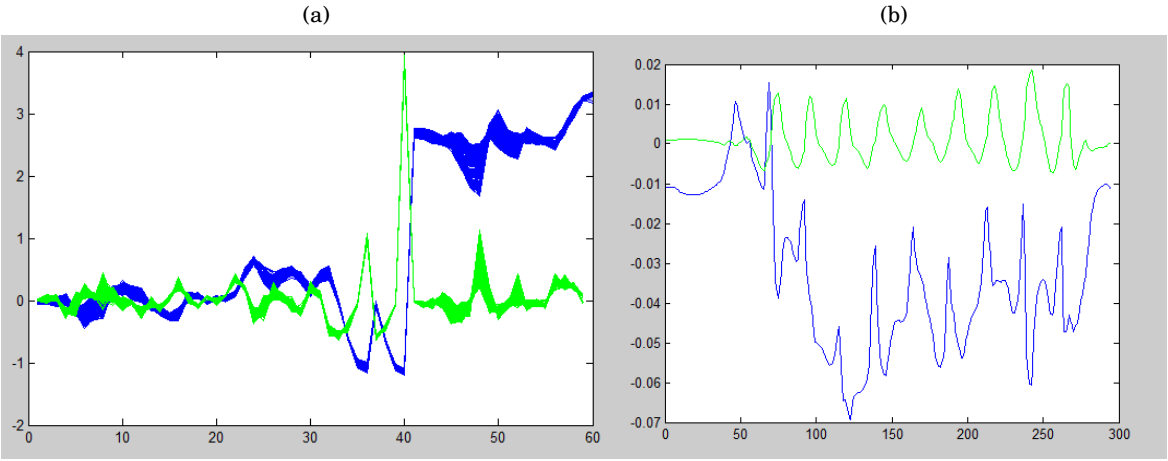

FIGURE 3.7. Distance data is shown in blue and velocity data shown in green. (a) All skeleton data has been plotted, with the x-axis corresponding to a joint co-ordinate. There are 20 joints each represented by 3 co-ordinates (x,y,z) hence the range 0-60. We can see a spike at 40 which corresponds to joints 15 and onwards which are the right hip, right knee, right ankle and right foot. The y-axis represents distance from the Kinect in the z direction which explains the spike since in a boxing stance the right hand side of the body should be further away. (b) Hip Joint over Time

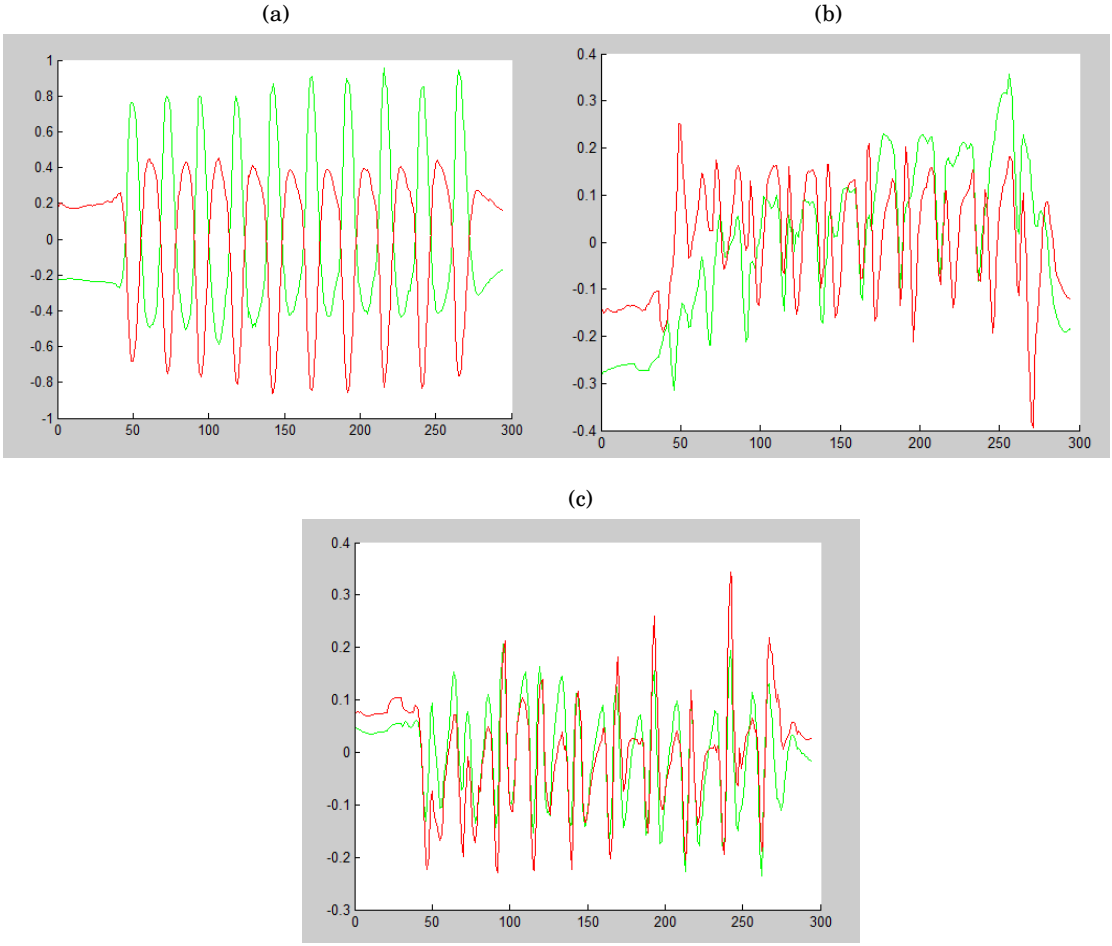

FIGURE 3.8. These figures shows the projection coefficient for the first principal component for distance data (green) and velocity data(red). (a) first projection coefficient (b) second projection coefficient (c) third projection coefficient

CHAPTER 4

RESULTS, EVALUATION & CONCLUSION

This chapter will present and discuss the results produced by the classifiers discussed in the 'My Approach' section. Other applications for this work will be suggested and difficulties encountered during the project will be highlighted. Further work can then be discussed within this context.

Accuracy & Precision

Any classification percentages given in this report have been obtained from the average of 5 consecutive results to give an accurate representation of classifier performance. Precision is defined as the degree to which measurements under the same conditions show the same results. Therefore all values have been rounded to two decimal points since I believe this is more than precise enough for my purposes.

4.1 Final results & Discussion

Below are the final results for the three best performing classifiers which highlight how the number of eigenvectors chosen can effect accuracy.

Figure 4.1 shows a summary of all my results pertaining to a different number of eigenvectors and it's effect on different classifiers. With the exception of using three eigenvectors where all classifiers suffered an accuracy reduction, there follows a general trend that as the number of eigenvectors increases as does the accuracy. This is intuitive since more eigenvectors means the data is represented with increased dimensionality and as such we would expect the SVM to have a small increase in accuracy given more significant data. However since each successive eigenvector represents less variance in the data than the one previously they become a less

Classifier	Eigenvectors Used	Accuracy
SVM	1	72.88%
Decision Tree	1	74.58%
Neural Network	1	74.60%
SVM	2	81.36%
Decision Tree	2	83.05%
Neural Network	2	86.40%
SVM	3	76.27%
Decision Tree	3	79.66%
Neural Network	3	79.70%
SVM	4	86.44%
Decision Tree	4	93.22 %
Neural Network	4	91.50%
SVM	5	**93.22**%
Decision Tree	5	91.53%
Neural Network	5	**94.90**%
SVM	6	**93.22**%
Decision Tree	6	**93.22**%
Neural Network	6	76.30%
SVM	7	91.52%
Decision Tree	7	91.52%
Neural Network	7	**94.90**%

TABLE 4.1. Punch classification accuracy of SVM, Neural Network and Decision Tree on a six class problem. (Jab, Cross, Left-Hook, Right-Hook, Left-Uppercut, Right-Uppercut)

and less useful way of representing data. As the number of eigenvectors increase as does the complexity of classification, since a classifier now has to train on a larger data set which is more costly to compute, resulting in slower running times. If too many eigenvectors are used the classifier will be forced to consider them during training, which for the less significant EVs could result in noise being introduced, making the classifier more prone to over fitting. It will also make feature extraction more difficult. It is unusual for all three classifiers to experience a drop in accuracy when an extra eigenvector is added, especially one that represents such a large variance in the data. For data this complex I find it surprising that the third eigenvector did not represent meaningful data, however, I can only infer that it instead represents some variance in the data that does not relate to a change in boxing pose.

SVMs and Neural networks performed excellent on a two-class problem between good and bad jabs. In this case the 'bad' jabs were thrown with the elbow sticking out since this is a target characteristic discussed earlier in the background chapter. Given this accuracy we can see that ZeroToHero is capable of distinguishing between very similar poses with only one joint position being significantly different. If this project was extended it is easy to see how this could be used

Classifier	Eigenvectors Used?	Accuracy
SVM	6	97.83%
Decision Tree	6	91.21%
Neural Network	6	**98.60**%

TABLE 4.2. Punch classification accuracy of SVM, Neural Network and Decision Tree on good & bad jabs

on the other punches to target common beginner mistakes.

Neural Networks

The neural networks performed excellently, with a surprising drop when using the sixth eigenvector. However, after looking at the behaviour of the other classifiers we can see that it did not change the accuracy of SVM, for neural networks the 7th EV did not change the accuracy. This suggest to me that similar to the 3rd EV it represents some variation that has nothing to do with boxing pose and as such introducing it only reduced accuracy. I believe the decision tree fared better in this case due to the fact that it is very robust to errors, both in classification and training.

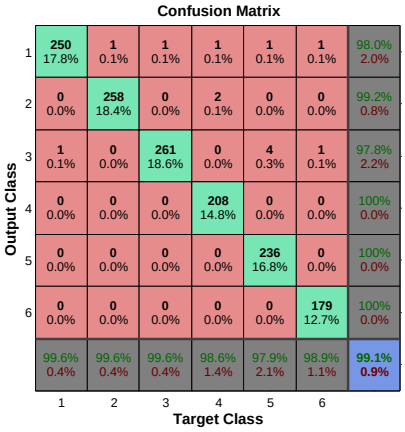

FIGURE 4.1. Neural network confusion matrix on training set

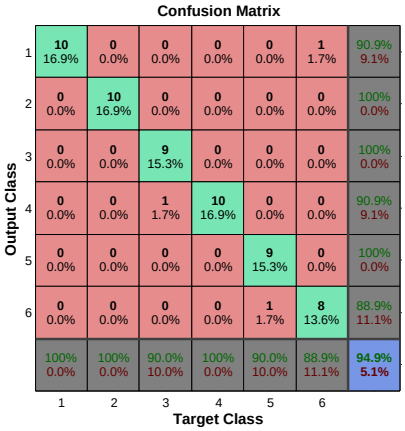

FIGURE 4.2. Neural network confusion matrix on a test set

4.2 Difficulties

I found this project to be an excellent way to combine my interest in boxing with my Computer Science background. It was however a difficult project and I ran into a few problems during research and development. One of the first difficulties is that skeleton data is can be incredibly noisy. After collecting datasets from both local clubs and the university boxing club I discovered that it was incredibly difficult, with my current system, to link these sequences together and to then use them as one large training set. Even though I had taken precautionary measures such as asking all of my subjects to stand in approximately the same spot during recording, even slight variations in distance from the Kinect combined with different body types resulted in unusable data. Different body types affected the speed at which punches were thrown since a short stocky person or a particularly tall person might be slower than average. This resulted

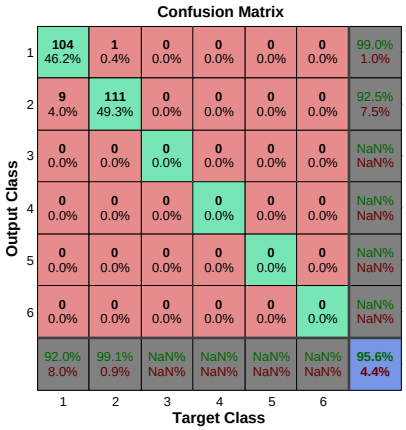

FIGURE 4.3. Neural network confusion matrix on a two-class problem

in a different time series but this can be successfully dealt with by the current implementation. However I found differences in distance from the Kinect during recording made a big difference, as did skeletal height. As arm length tends to be proportional to skeletal height, those with longer arms had a shorter distance between themselves and the kinect after the punch has been delivered. Conversely those with shorter arms had a longer distance. Since the current system uses skeletal joint co-ordinates the projection coefficient for the first component is heavily influenced by distance from the Kinect. Unsurprisingly this meant my data was un-normalised, which made it difficult to spot patterns that could have been used to classify the punches. It also made it almost impossible to automatically segment since the projection coefficients for the first principal components were quite different even within punches of the same class.

4.2.1 Normalisation

In an attempt to normalise my data for future recording sessions I took the the central hip joint and used this as as the skeletal centroid, with it being given position (0,0,0). This resulted in all joints being given positions relative to the hip joint. 'Movement' has two components; the global translation of the skeleton and movement in the pose itself. By normalising on the hip joint the global movement is removed which is useful since we are only interested in a change of pose. Although this method seemed to reduce noise and improve my classification, I still had the problem that the first principal component was heavily influenced by distance from the Kinect.

4.2.2 Further Work & Improvements

My first improvement would be to develop an algorithm for normalising skeleton data so that data from multiple recording sessions and distances from the Kinect could be used together as one dataset. Using the above hip normalisation technique it would be possible to look at the

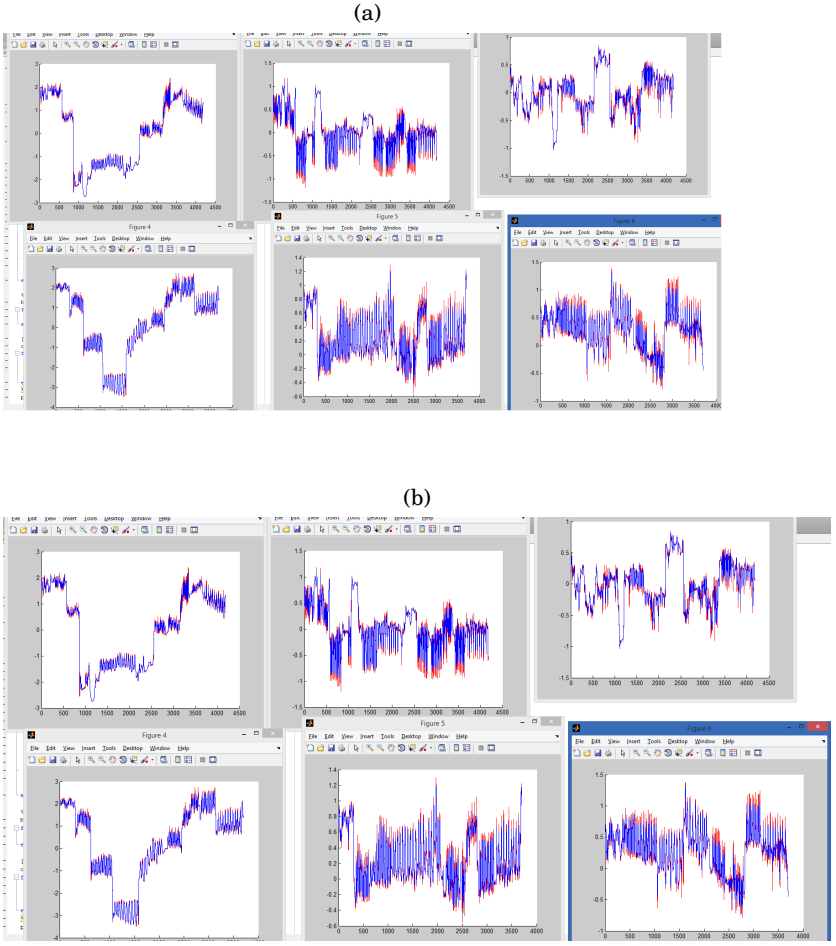

FIGURE 4.4. Twelve different un-normalised punch sequences, two for each punch. Given that these are sequences of the same punch you would hope them to be reasonably similar, however, we can clearly see the difference that distance from the Kinect can make.

distance from each joint to the centroid and normalise those values based on a 'normal' base value. This would be useful in accounting for different lengths of limbs and body types.

I would also like to implement a system where distance from the Kinect was unimportant, meaning the first principal component would not be heavily influenced by distance. This could be by using velocity and implementing a better segmentation algorithm.

I could also look at using quaternions and their associated rotation matricies instead of joint position.

Although the system is fast enough to perform in real time it is currently not 'live' since data

is recorded and then analysed, this would be the next step.

Finally I would like to assess the quality of every punch, currently we can classify a punch and tell if it is a good or bad punch. I would link both classifiers together, possibly using an ensemble method so each punch is recognised and then assessed on it's quality based on the classification.

4.2.3 Other Applications

Makaton is language program that uses gestures, body language, signs and symbols to communicate, designed for individuals that struggle to communicate via speech.[12] [44] For Makaton to work effectively you must be facing those you are trying to communicate with while repeating simple gestures with your hands. This shares similarities with this system since you must also be relatively face-on and the movement is periodic. I believe this system could be adapted relatively quickly and easily to recognise Makaton gestures. For similar reasons this could also be used to recognise British Sign Language (BSL) although this would be more complicated than the very simple gestures that Makaton relies on.

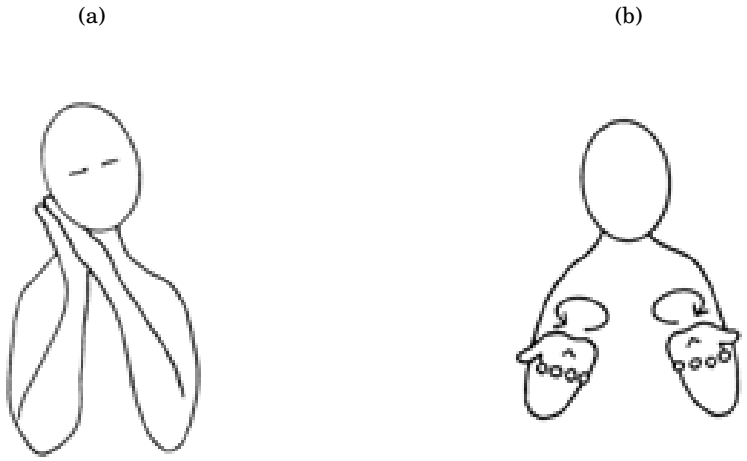

FIGURE 4.5. Examples of simple gestures that make up the Makaton language

ZeroToHero could also be used to classify any atomic activity[1] which can be temporally segmented such as kicking, punching, jumping as well as sitting and standing.

It could be used in the field of Human-Computer interaction for simple gesture recognition provided the gestures were designed with Kinect's limitations in mind. A simple semaphore-type system would be ideal since the gestures would be reasonably easy to recognise provided they were repeated.

Finally my system could be incorporated into pre-existing martial arts trainers to add additional functionality or to simple improve their current punch recognition algorithm.

Conclusion

I have presented a real time punch recognition and quality assessment algorithm which has the potential to bring specialist boxing coaching into a home environment. Human poses were recorded using the Kinect & Microsoft SDK, an already widespread, low cost consumer device with high accessibility. I investigated the efficacy of a number of dimensionality reduction techniques when applied to pose sequences including CCA, LLE, Hessian LLE, Laplacian Eigenmaps, LTSA and PCA. I found that PCA was most suitable for facilitating time series segmentation since I could consistently obtain a sinosidal-like sequence. This enabled me to perform automatic punch segmentation on the reduced dimensionality pose-time series and extract useful features from the original unsmoothed reduced dimensionality data set. Segmentation was achieved by smoothing the projection coefficients for the first principal components, before extracting local maxima from across the time series. Each maxima was then inspected by a set of heuristics which determine whether it signifies the start of a punch or is just a local maxima that still exists after smoothing. Finally, the optimal number of principal components retained and the suitability of a variety of classifiers were empirically assessed including SVM, Neural Networks(NN) and Decision trees(DT). On a 6 class problem of punch classification, ZeroToHero achieved accuracy results of **94.90%** with NN and **93.22%** using SVM and DT. On a two-class problem of good and bad jabs, an accuracy of **98.60%** was achieved using NN, **97.83%** by SVM and **91.21%** by DT.

BIBLIOGRAPHY

[1] J. K. AGGARWAL AND M. S. RYOO, *Human Activity Analysis : A Review*, (2007).

[2] D. S. ALEXIADIS, P. KELLY, P. DARAS, N. E. O'CONNOR, T. BOUBEKEUR, AND M. B. MOUSSA, *Evaluating a dancer's performance using kinect-based skeleton tracking*, Proc. 19th ACM Int. Conf. Multimed. - MM '11, (2011), p. 659.

[3] S. T. AWARDS, *Best training technology*. `http://www.sportstechnologyawards.com/sm_training.html`, 2014.

[4] D. S. BBC, *Usa boxing goes high tech in training in colorado springs*. `http://gazette.com/usa-boxing-goes-high-tech-in-training-in-colorado-springs/article/1500163`, May 2013.

[5] M. BELKIN AND P. NIYOGI, *Laplacian eigenmaps for dimensionality reduction and data representation*, Neural Comput., (2003).

[6] S. BIANCO AND F. TISATO, *Karate moves recognition from skeletal motion*, 2012.

[7] E. B. C. KESKIN AND L. AKARUN, *A unified framework for concurrent usage of hand gesture, shape and pose*, CVPR - 2012.

[8] D. CATUHE, *Programming with the kinect for windows software development kit*.

[9] O. ÇELIKTUTAN, C. B. AKGÜL, C. WOLF, AND B. SANKUR, *Graph-Based Analysis of Physical Exercise Actions*, (2013).

[10] C. CHANG, B. LANGE, AND M. ZHANG, *Towards pervasive physical rehabilitation using Microsoft Kinect*, Pervasive . . . , (2012).

[11] S.-F. C. CHANG, YAO-JEN AND J.-D. HUANG., *"a kinect-based system for physical rehabilitation: A pilot study for young adults with motor disabilities."*, ICIAR - 2011.

[12] M. T. M. CHARITY, *About makaton*. `https://www.makaton.org/aboutMakaton/`.

[13] R. R. COIFMAN AND S. LAFON, *Diffusion maps*, Appl. Comput. Harmon. Anal., 21 (2006), pp. 5–30.

[14] E. . P. S. R. COUNCIL, *New technology will help improve athletes' performance*. http://www.epsrc.ac.uk/newsevents/news/2009/Pages/improveathletesperf.aspx, 2009.

[15] P. DEMARTINES AND J. HERAULT, *Curvilinear component analysis: a self-organizing neural network for nonlinear mapping of data sets.*, IEEE Trans. Neural Netw., 8 (1997), pp. 148–54.

[16] J. DEVRIES, *Fighters uncaged review | ign*. http://www.ign.com/articles/2010/11/09/fighters-uncaged-review, 2010.

[17] E. F. DYNAMICS, *"elliott fight dynamic gloves"*. http://elliottfightdynamics.com/, 2014.

[18] ENGINEERING AND P. S. R. COUNCIL, *Details of grant - esprit with pervasive sensing*. http://gow.epsrc.ac.uk/NGBOViewGrant.aspx?GrantRef=EP/H009744/1, 2009-2015.

[19] GAMESRADER, *Fighters uncaged review | gamesradar*. http://www.gamesradar.com/fighters-uncaged-review/, 2011.

[20] GAZETTE, *Usa boxing goes high tech in training in colorado springs*. http://www.bbc.co.uk/news/technology-18735629, May 2013.

[21] T. GUARDIAN, *London 2012 olympics: How athletes use technology to win medals*. http://www.theguardian.com/sport/2012/jul/04/london-2012-olympic-games-sport-technolog 2012.

[22] K. KAEWPLEE, N. KHAMSEMANAN, AND C. NATTEE, *Muay Thai Posture Classification using Skeletal Data from Kinect and k-Nearest Neighbors*.

[23] C.-C. C. L. XIA AND J. K. AGGARWAL., *A decision forest based feature selection framework for action recognition from rgb-depth cameras.*, ICIAR - 2013.

[24] ——, *View invariant human action recognition using histograms of 3d joints.*, IEEE CVPR - 2012.

[25] M. B. MANJUATHA, B. P. PRADEEP, AND S. Y. SANTHOSH, *Survey on Skeleton Gesture Recognition Provided by Kinect*, (2014), pp. 8475–8483.

[26] METACRITIC, *Fighters uncaged critic reviews for xbox 360*. http://www.metacritic.com/game/xbox-360/fighters-uncaged/critic-reviews, Jan. 2011.

[27] MICROSOFT, *Infrared stream - msdn*. http://msdn.microsoft.com/en-us/library/jj663793.aspx, 2012.

[28] L. MIRANDA, T. VIEIRA, D. MARTINEZ, T. LEWINER, A. W. VIEIRA, AND M. F. M. CAMPOS, *Real-Time Gesture Recognition from Depth Data through Key Poses Learning and Decision Forests*, 2012 25th SIBGRAPI Conf. Graph. Patterns Images, (2012), pp. 268–275.

[29] L. MIRANDA, T. VIEIRA, D. MARTÍNEZ, T. LEWINER, A. W. VIEIRA, AND M. F. M. CAMPOS, *Online gesture recognition from pose kernel learning and decision forests*, Pattern Recognit. Lett., 39 (2014), pp. 65–73.

[30] C. R. K. G. V. R. B. R. OFLI, FERDA, *Sequence of the most informative joints (smij): A new representation for human skeletal action recognition*, 2012.

[31] R. E. B. P. U. PRESS, *"dynamic programming"*.

[32] M. RAPTIS, D. KIROVSKI, AND H. HOPPE, *Real-time classification of dance gestures from skeleton animation*, Proc. 2011 ACM SIGGRAPH/Eurographics Symp. Comput. Animat. - SCA '11, (2011), p. 147.

[33] K. RECTOR, C. L. BENNETT, AND J. A. KIENTZ, *Eyes-Free Yoga : An Exergame Using Depth Cameras for Blind & Low Vision Exercise*, Int. ACM SIGACCESS Conf., (2013).

[34] S. T. ROWEIS AND L. K. SAUL, *Nonlinear dimensionality reduction by locally linear embedding.*, Science, 290 (2000), pp. 2323–6.

[35] T. T. T. S. CELEBI, A. S. AYDIN AND T. ARICI, *Gesture recognition using skeleton data with weighted dynamic time warping*, VISAPP - 2013.

[36] R. T. M. G. P. K. D. M. P. D. A. D. S. ESSID, D. ALEXIADIS AND N. E. O'CONNOR, *An advanced virtual dance performance evaluator*, ICASSP - 2012.

[37] 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*, Cvpr 2011, (2011), pp. 1297–1304.

[38] M. TURK AND A. PENTLAND, *Eigenfaces for recognition*. `http://www.cse.unr.edu/~bebis/MathMethods/PCA/case_study_pca1.pdf`, 1991.

[39] R. C. UK, *Cutting edge 2012: The research behind sport*. `http://www.rcuk.ac.uk/media/CuttingEdge2012/#bike`, 2012.

[40] ——, *Memorandum from research councils uk (rcuk) in response to the house of lords inquiry into sports and exercise science and medicine.* `http://www.rcuk.ac.uk/RCUK-prod/assets/documents/submissions/SportsExercise.pdf`, 2012.

[41] N. L. M. B. T. R. V. VENKATARAMAN, P. TURAGA AND S. L. WOLF., *Attractor-shape for dynamical analysis of human movement: Applications in stroke rehabilitation and action recognition*, HAU3D'13 - 2012.

[42] Z. Y. Z. W. Q. LI AND Z. C. LIU, *Action recognition based on a bag of 3d points.*, CVPR4HB10 - 2010.

[43] WIKIPEDIA, *Great britain at the 2012 summer olympics*. http://en.wikipedia.org/wiki/Great_Britain_at_the_2012_Summer_Olympics, 2014.

[44] ——, *Makaton*. http://en.wikipedia.org/wiki/Makaton, 2014.

[45] K. YAMAOKA, M. UEHARA, T. SHIMA, AND Y. TAMURA, *Feedback of Flying Disc Throw with Kinect and its Evaluation*, Procedia Comput. Sci., 22 (2013), pp. 912–920.

[46] Z. ZHANG AND H. ZHA, *Nonlinear Dimension Reduction via Local*, in Intell. Data Eng. Autom. Learn., 2003, pp. 477–481.