Gesture Recognition of Tennis Biomechanics

A Thesis Proposal

Submitted to:

Temple University Graduate Board

In Partial Fulfillment

Of the Requirement for the Degree

MASTER OF SCIENCE

IN ELECTRICAL ENGINEERING

Victor Espinoza

December 2017

1. Introduction 4
2. Motivations 5
3. Research Objectives 6
4. Background 6
4.1 Kinect v2 Background 6
4.2 Tennis Background 9
4.3 Pattern Recognition Techniques 10
4.3.0 Support Vector Machines 10
4.3.1 Random Forest Algorithm 13
4.3.2 Hidden Markov Model 15
4.3.4 Multi-layer Perceptron 17
5. Methods 18
5.1 Protocol Design 18
5.2 Signal Processing 20
5.3 Accuracy Monitoring 21
6. Implementation 22
6.1 Random Forest Implementation 22
6.2 SVM Implementation 23
6.3 HMM implementation 24
6.4 MLP Implementation 26
7. Proposed Work Overview 27
7.1 Proposed Work Timeline 29

Abstract

The purpose of this study is to create a gesture recognition system that interprets motion capture data of a tennis player to determine which biomechanical aspects of a tennis swing best correlate to a swing efficacy. For our learning set this work proposes recording 50 collegiate tennis athletes of similar competency with the Microsoft Kinect performing standard tennis swings in the presence of different targets. With the acquired data we anticipate extracting biomechanical features that correlate to ball trajectory with proper technique and test them as inputs to our designed classifiers. The proposed work implements two machine learning algorithms to classify ball trajectory that have shown reliability in previous gesture recognition applications, the Hidden Markov Model (HMM) and Support Vector Machine (SVM).

In attempt to achieve results more independent of our selected sensor's accuracy which dictates over extracted features, we implement a featureless deep learning approach with the use of a multi-layer perceptron (MLP). We expect our classifying results to favor feature driven machine learning algorithms over the MLP deep learning approach. If we implement initial conditions to the MLP derived from reliable features, we expect the MLP to come close in matching the performance of the feature dependent approaches and gradually outperform them as sample size increases.

1.0 Introduction

The goal of this work is to use machine learning and computer vision to identify which biomechanical movements are best correlated with effective tennis swings based on motion-capture data of subjects playing tennis. In this work we aim to develop a set of machine learning algorithms that will learn proper technique of the three standard tennis swings; forehand, backhand, and serve. Within each of these three tennis swings, we will aim for the algorithm to learn how to properly hit a tennis ball to the left and right side of the tennis court.

In practice, these general directions, or ball trajectories, are referred to as hitting the ball "down-the-line" and "cross-court" when speaking of forehands and backhands. When speaking of the tennis serve, the two common ball trajectories are referred to as "down-the-'T'" or "out-wide". These common ball trajectories will be explained more in depth in the background section. As for the serve portion of the study, this work will analyze the biomechanics of how to serve to two different general directions. Once we have collected skeletal tracking data of skilled tennis players performing these swings, we can extract biomechanical features such as joint angles and distal landmarks to develop ~~a~~ machine learning algorithm that will classify these swings and correlate them to ball trajectories.

In order to obtain a proper learning data set, we will choose highly skilled tennis athletes to perform the mentioned tennis swings in the presence of different targets that demonstrate good tennis practice. In this work, we propose using the Microsoft Kinect v2 as the motion capturing tool. Previous research and applications have shown the Kinect's spatial measurement limitations and how it compares to a laboratory grade motion capture system.

2.0 Motivation

In recent years, a variety of new technologies have been introduced to tennis that have changed the game. Technologies like “Hawk-Eye” which track the ball’s trajectory and landing have started to diminish the need of official line judges during tennis matches since it can determine whether the ball landed in or out. Not only has this made line-calling more accurate and reliable, it has also opened up a field of more in-depth statistical analysis of the sport. This multi-camera system uses triangulation to track the tennis ball up to 2.6 mm accuracy [1] and has attracted research institutions and companies like IBM and SAP to develop statistical analysis software on professional match play. Ultimately, the work proposed here will develop a statistical model based on biomechanical features of tennis players while drawing a bridge to the field of study that focuses on the trajectory of the ball.

In addition, this work could also contribute to the development of tennis teaching software. If the Kinect’s skeletal tracking algorithm is accurate enough, future work could use this data to classify a subject’s swing technique as proper or poor. Private tennis lessons, work towards to perfecting a customer’s swing technique. Furthermore, a professional tennis instructor will provide tips and detailed adjustments to correct technical mistakes, but not to the extent of instruction that requires precision down to the millimeter. Consequently, we anticipate that the accuracy of the Kinect’s body tracking will be sufficient for the proposed work since millimeter precision is not vital to the instruction or the learning of standard tennis swings.

This work could also contribute to making tennis video games or simulators more realistic. The value of our results will be significant to applications that refer to tennis and that include motion capture. In addition, the skeletal tracking data collected of highly skilled players will record the

current biomechanical approach of advanced players as it changes overtime as players and equipment technology evolve.

3.0 Research Objectives

The overall objective of the proposed work is to determine if a computer can correlate biomechanics to swing efficacy based on the skeletal tracking data obtained with the Kinect. More specifically, to determine if we are able to design a set of algorithms that are capable of learning the correct techniques of a proper forehand, backhand, and serve with a desired target. With this main objective in mind, we can derive specific aims.

The success of this study will require us to accomplish the following set of goals:

A. Determine features of body movement that are correlated to a tennis player's swing
B. Design an algorithm that will be able to predict the tennis ball's general trajectory from body tracking data obtained from the Kinect v2.

We will exploit the Kinect's skeletal tracking data with various discrete-time signals processing techniques and pattern recognition techniques in order to meet these goals.

4.0 Background

4.1 Kinect v2 Background

The Microsoft Kinect v2 comprises of three different sensors; depth sensor, infrared sensor, and an RGB camera, which do not match in resolution or physical location. Microsoft has developed a proprietary machine learning algorithm to detect up to six human bodies with the Kinect using these three layers of data streams [2]. This algorithm is what makes this motion capture technology a markerless approach. The skeletal tracking data is encapsulated in the "body

frame" in which we can obtain 3-dimensional data of 25 different body joint centers of each body at a frame rate of up to 30 fps. The joint data includes x, y, z components and. In The Kinect-estimated joint centers are demonstrated in figure 4.1.0

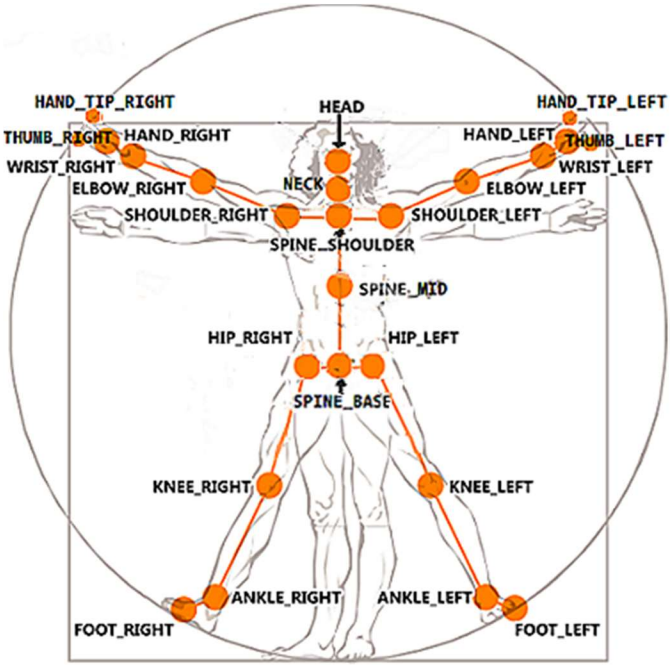


fig. 4.1.0 [3]

As previously mentioned in the introduction, section 1.0, it is important to be mindful of the Kinect's skeletal tracking's accuracy, and how this accuracy will affect our machine learning algorithm design.

Studies show that the Kinect v2's accuracy and frame rate is outperformed by multi-camera laboratory grade motion capture systems, but it is reliable enough to be considered a valid clinical measurement tool [4]. Although the Kinect v2 is a low grade motion capture system, displaying low accuracy especially in ankle and feet detection [4], it is portable, low-cost, and does not require calibration or the use of markers on subjects like a Vicon system of 2 mm accuracy at 100 fps. Without the constraints of markers and a research lab setting, subjects that

will partake in our study will be able perform as freely as they would during match play as long as they stay in the Kinect's field of vision and within four meters of the sensor.

The Kinect v2 outputs 3-dimensional space coordinates for 25 joints of each body tracked with reference to the actual device. In more detail, the origin (point 0,0,0) of the 3 dimensional coordinate system obtained by the Kinect is the actual location of the Kinect as shown in figure 4.1.1. The *z* component represents the orthogonal distance from a joint to the Kinect. The *y* component represents the orthogonal distance from the joint to the floor plane, and the *x* component entails the distance of the joint that extends to the right or left of the sensor. With the discussion of the coordinate system, it also important to note that the floor plane is also encapsulated within the Kinect's body frame. This floor detection is also a component of Microsoft's proprietary skeletal tracking since the *y* coordinates are dependent on it.

Since the position of the subject with respect to the Kinect will not be constant for every swing across all subjects, we will have to measure distances and angles with respect to the subject's own body. Additionally, we can manipulate the data by removing parts of the *x* and z components of the 3D coordinates in a way to relocate the body coordinates of into a fixed position for each frame of the data collected.

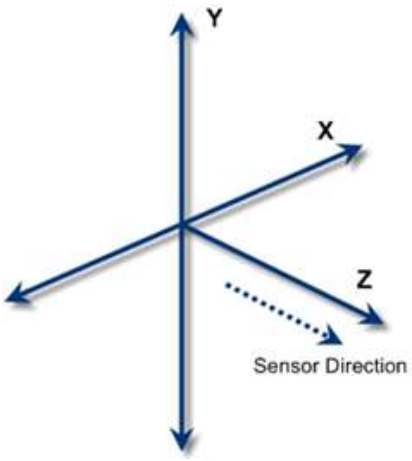


fig. 4.1.1 [5]

4.2 Tennis Background

Generally speaking, an intermediate tennis player is able to hit the three standard tennis swings; forehand, backhand, and serve. This level of player should also have control over the general direction of the ball from all of these previously mentioned swings. These directions refer to the ball's trajectory with respect to the subject who hit the ball. The forehand and backhand, often referred to as groundstrokes, can be struck "down-the-line" or "cross-court". The serve can also be struck in two general targets, "down-the-T", which refers to the center of the court, and "out-wide" which pertains to aiming the serve at the singles side line that belongs to that service box. The clarification of theses swings and associated directions are important since they can vary depending on the dominant hand of the subject and on which side of the court the subject who is hitting the ball from. This set of tennis swings and general directions create a general baseline for tennis knowledge and ability, as well as a better understanding of the objective of this work.

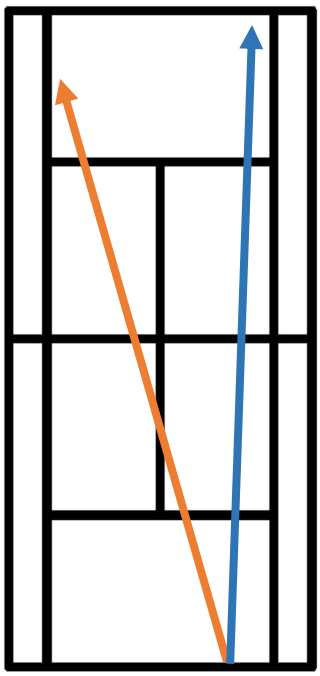


Fig 4.2.0 Forehand down-the-line (blue) crosscourt (orange) for a right-handed player

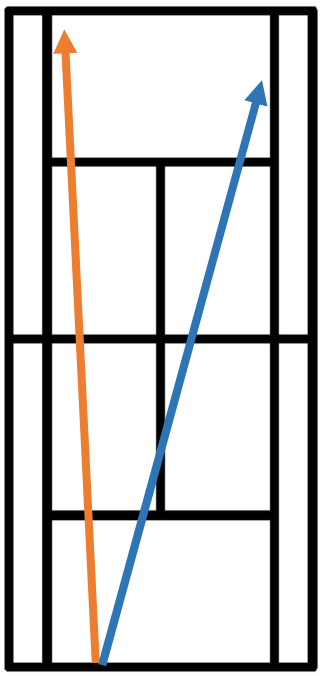


Fig 4.2.1 Backhand down-the-line (orange) crosscourt (blue) for a right-handed player

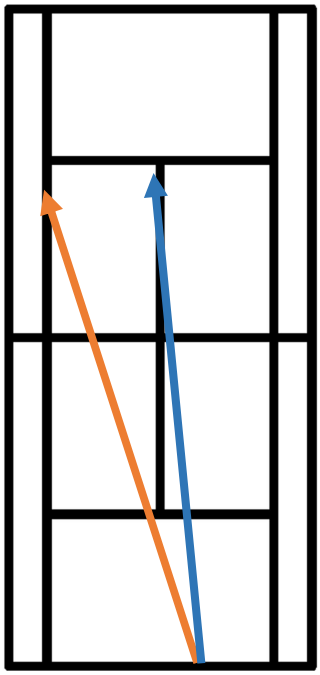


Fig 4.2.2 Deuce side serve down-the-T (blue) out-wide (orange)

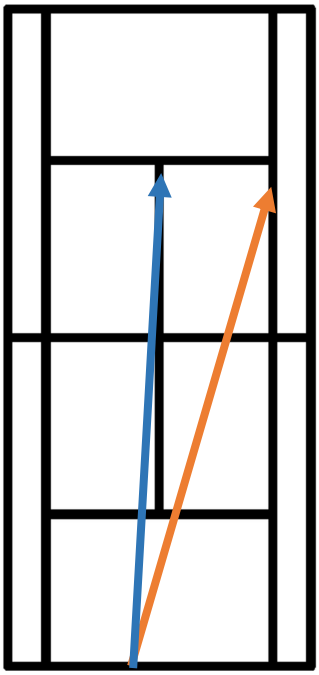


Fig 4.2.3 Ad Side down-the-T (blue) out-wide (orange)

4.3 Pattern Recognition Techniques

In this section we will introduce the pattern recognition approaches that we anticipate using in the proposed work. We will consider two methods that are feature dependent, support vector machines and hidden Markov models. The feature independent approach in this work is the multi-layer perceptron.

4.3.0 Support Vector Machines

Support vector machines are a supervised machine learning technique used to classify data between two classes. SVMs are essentially the use of a decision hyperplane of dimension (p -1) to classify p-dimensional data points in the feature space. This decision hyperplane is designed based on a subset of the training data points referred to as *support vectors* which lie close to

estimated boundary between the two classes; the remaining training data samples becomes essentially ignorable.

The design procces of an SVM classifier starts by considering the following basic classifier in slope intercept form.

$$f(x) = \beta x + b \ = 0 \qquad \text{Eq. 4.3.0.1}$$

If we impose the constraints of equation 4.3.0.2 where $y_j = \pm 1$ as the classification variable, and x_j denotes data points, we can define the decision hyperplane becomes by finding the optimal value of β. The data points that are chosen as the support vectors satisfy equation 4.3.0.3

$$y_j(\beta x_j + b) \geq 1 \qquad \text{Eq. 4.3.0.2}$$

$$y_j f(x_j) \geq 1 \qquad \text{Eq. 4.3.0.3 simplified form of 4.3.0.2}$$

$$y_j(\beta x_j + b) = 1 \qquad \text{Eq. 4.3.0.4 for all support vectors}$$

To optimize our classifier with the given constraints we solve for optimal β using LaGrange multipliers. The form of our optimized classifier becomes

$$f(x) = \sum_{j=1}^{N} \alpha_j y_j \, x_j \cdot x_i + b \qquad \text{Eq. 4.3.0.5}$$

If the classes are not linearly separable, the training data are transformed to a higher dimension where the data can be separated by a linear surface. Kernel functions can be used to avoid the need for explicitly mapping each data point into a higher dimensional space. This use of a kernel function leads us to the final form of the classifier shown in Eq. 4.3.0.6

$$f(x) = \sum_{j=1}^{N} \alpha_j y_j K(x_j, x_i) + b$$ Eq. 4.3.0.6

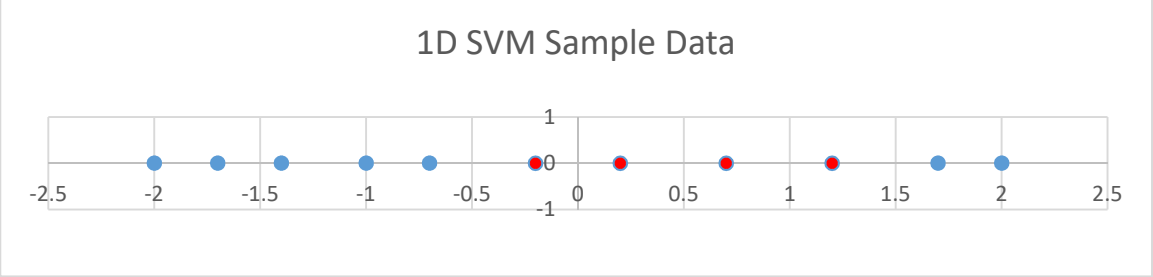
Commonly used kernel functions in practice are

Eq. 4.3.0.7 Linear Kernel $k(x, y) = x_i x_j + c$

Eq. 4.3.0.8 RBF Kernel $k(x, y) = e^{-\gamma \left|\left|x_i - x_j\right|\right|^2}$

Eq. 4.3.0.9 Polynomial Kernel $k(x, y) = (x_i x_j + c)^d$

Fig. 4.3.0.0



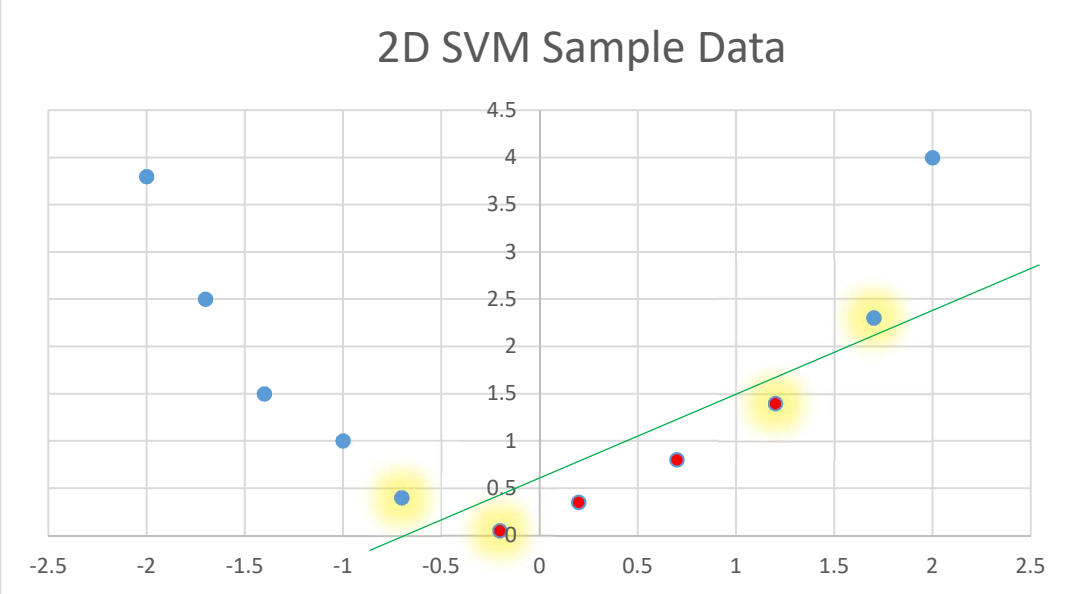


Fig. 4.3.0.1

4.3.1 Random Forest Algorithm

In order to comprehend the random forest approach we will briefly go over the decision tree, which is the building block of the random forest algorithm. The decision tree is a classification technique that consists of a set of questions, and decides a path that will ultimately lead to the classification of the data. The questions can be thought of as nodes or visualized as a point where branches of the tree split. As data travels up a finite number of nodes it will ultimately end up in a leaf, where the data is classified. An example of a decision tree classifier is demonstrated in figure 4.3.1.1 of the given sample data found in figure 4.3.1.0

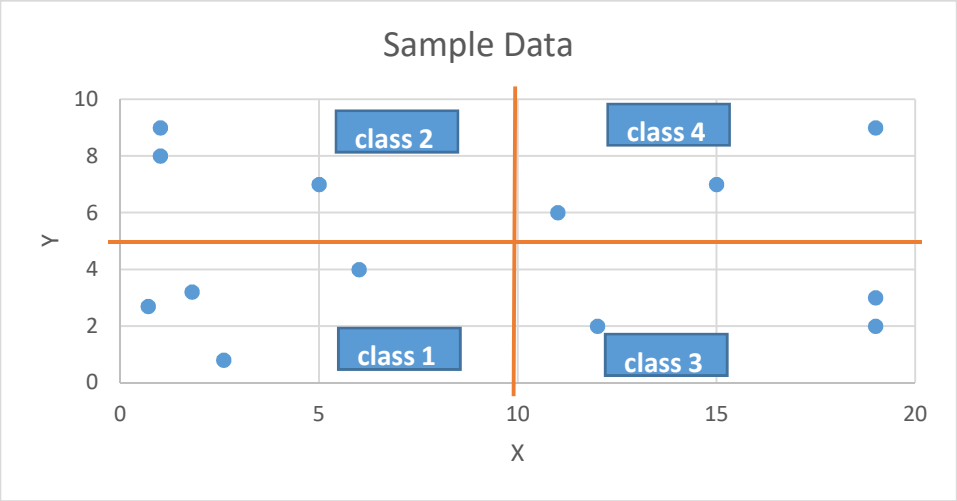


Figure 4.3.1.0 Sample Data

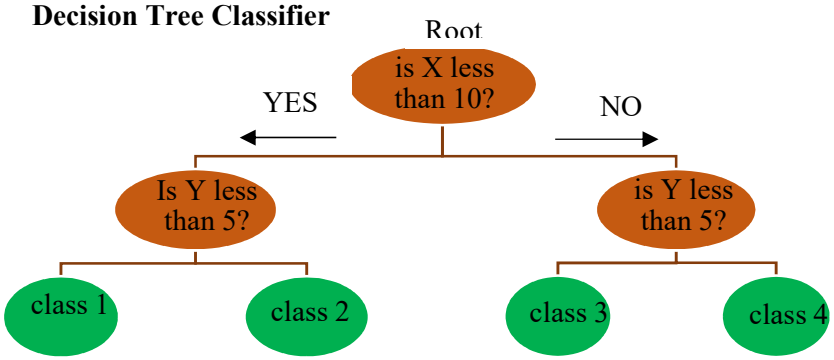


Figure 4.3.1.1

Although this is a practical and computationally inexpensive approach it is easy to overfit the data, meaning that complex trees will not classify non-training data very well.

In order to quantify for the efficacy of the trees we will implement ways to measure the impurity in the classifications. Two common methods to quantify this impurity in classification regions are by measuring entropy and the Gini index, shown in eq. 4.3.1.0 and 4.3.1.1 respectively. In the worst scenario, the entropy measurement will return a value of 1, indicating that the probability of the classes w_j in that split or region is equally likely. On the other hand, the Gini index takes the sum of the square of the probability of each class in the region or split and subtracts this sum from 1. If two classes are equally likely in a region or split, the Gini index would return a value of 0.5.

$$i(N) = -\sum_j P(w_j) \log_j P(w_j) \qquad \text{Eq. 4.3.1.0 Entropy}$$

$$i(N) = 1 - \sum_j P^2(w_j) \qquad \text{Eq. 4.3.1.1 Gini Index}$$

As the number of features of the training data increases, the decision tree becomes greater in depth, indicating more nodes and more classification leafs that will decrease in variance. In order to account for these weaknesses in decision trees, it is common to practice pruning. Pruning entails setting a predetermined maximum depth, or path from root to leaf, for the decision tree, as well as setting a minimum number of samples of training data per leaf, or class. In addition, we introduce the random forest algorithm as an "ensemble" machine learning method in which we deploy various weak learners, individual decision trees. This algorithm is essentially a set of multiple decision trees that can be used for classification and regression. In use as a classification method, the data sample will go through each of the decision trees while we keep count of the classifications, leafs, that point to each class in the whole forest. The random forest will

ultimately classify the new data object based on the leaf with the most votes. The general workflow of random forests is depicted in figure 4.3.1.2

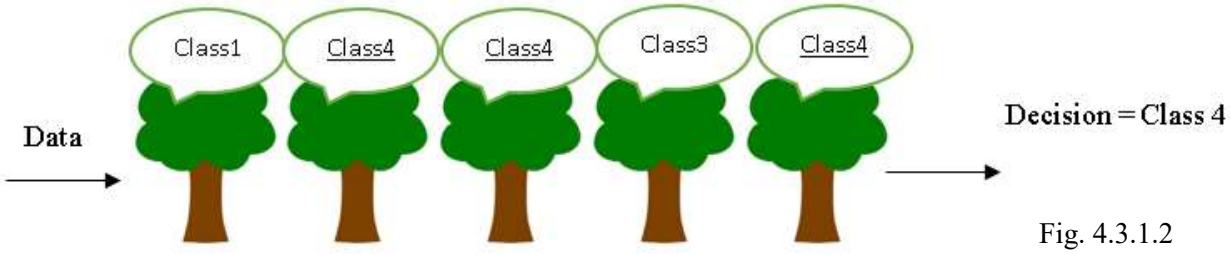


Fig. 4.3.1.2

4.3.2 Hidden Markov Model

Hidden Markov models, HMMs, are a well-established technique in machine learning with applications involving sequences of data like speech recognition and handwriting recognition. HMMs are represented by a graph of finite number of states, which are connected by transitions. At each of these states, there are two sets of probabilities; transition probabilities and emission probabilities. Transition probabilities represent the chances of transferring to the next state while the emission probabilities represent the chance of emitting an output, or symbol, at the current state. Figure 6.2 demonstrates an example of a network of 3 hidden states denoted by w_i, with transition probabilities a_{ij}, and emission probabilities b_{jk} of emitting discrete symbol v_k. HMMs are a double stochastic process, the red component of the figure indicates the visible, or observable, portion of the model, while the black components of the model indicate the hidden, non-observable part. These transition and emission probabilities create a transition matrix and emission matrix respectively and are denoted in equations 4.3.2.0 and 4.3.2.1. Equation 4.3.2.2 displays the probabilities of starting at each state, which are referred to as the initial probabilities. An HMM is considered ergodic if every element of matrix A is a non-zero value. In other words, each state in the model is reachable from any other states in a finite number of transitions.

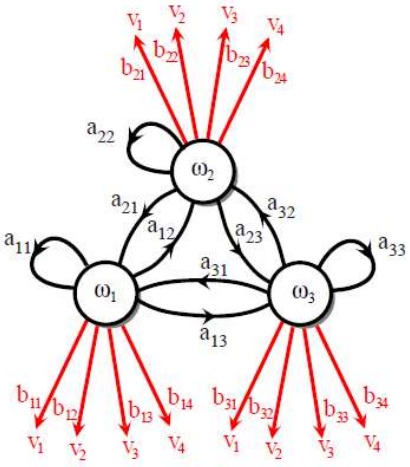


figure 6.2.0 [6]

Eq. 4.3.2.0 $A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$ Eq. 4.3.2.1 $B = \begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ b_{31} & b_{32} & b_{33} & b_{34} \end{bmatrix}$

$$\Pi = [\pi_1 \quad \pi_2 \quad \pi_3] \qquad \text{Eq. 4.3.2.2}$$

HMMs are expressed in the form of equation 6.2.3

$$\lambda = (A, B, \Pi) \qquad \text{Eq. 4.3.2.3}$$

If we apply this machine learning technique to sequential data in time, like speech or motion capture data, we can focus on a subset of HMM's that is non- ergodic. This type of HMM, known as the Left-Right Model or Bakis model, is more suited for sequential data. Since it is not possible to go back in time, the transition probability of states become the option to stay at the current state or move on to the next one as depicted in figure 4.2.1.

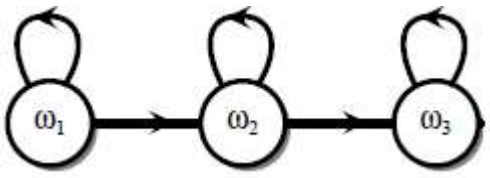


Fig. 4.3.2.3

In this case, the transition matrix becomes $A = \begin{bmatrix} a_{11} & a_{12} & 0 & 0 \\ 0 & a_{22} & a_{23} & 0 \\ 0 & 0 & 0 & a_{33} \end{bmatrix}$

While the initial state probabilities becomes $\Pi = [1 \quad 0 \quad 0]$

4.2.2 Multi-layer Perceptron

Since all previously mentioned machine learning algorithms take calculated biomechanical features from Kinect based estimations as inputs, we heavily rely on the limited accuracy of the sensor's skeletal tracking. In this section we introduce a common deep learning approach, the multilayer perceptron, which can take raw data as input. The building block of the multilayer perceptron, commonly referred to as MLP or feed-forward neural network, is the neuron displayed in orange in figure 4.2.2.0. The input layer consists of all the x_m nodes, the hidden layer is made up of the computational unit, and output layer is comprised of $\hat{y}$. Output $\hat{y}$ is computed with the use of an activation function. The activation function is typically designed to return a binary value based on a threshold, or a probability value ranging from 0 and 1.

$$\hat{y} = \varphi\left(\sum_{i=1}^{m}(w_i x_i) + bias\right) \qquad \text{Eq. 4.3.2.4}$$

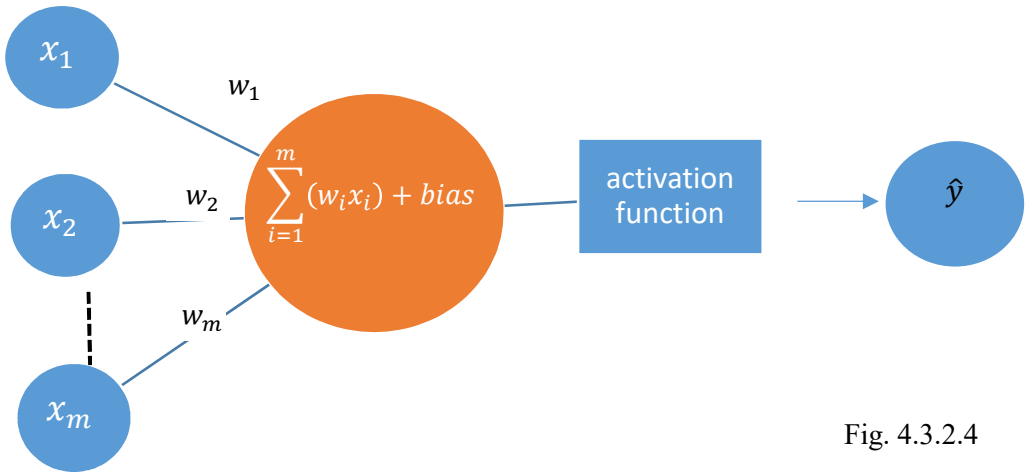


Fig. 4.3.2.4

The training process consists of two phases, the feedforward phase and the back propagation phase. In the feedforward mode, the weights are estimated or randomized and the input is introduced to the network and are computed in the forward direction. The output $\hat{y}$ is then compared to the true value of y and the error is calculated between the two using a cost function $J(w)$. The backpropagation phase then takes this computed error and adjust the weights of the network based on gradient descent at a learning rate η as shown in equation 4.3.2.5.

$$\Delta w = -\eta \frac{\partial \mathrm{J(w)}}{\partial w} \qquad \text{Eq. 4.3.2.5}$$

The ultimate goal of the training is to go through each training set, an epoch, and solve for the optimal values of each weight to minimize the error.

5.0 Methods

In this section we describe how we plan on collecting data from highly skilled athletes performing the three standard tennis swings and managing the raw extracted motion capture data.

5. 1 Protocol Design

The protocol for this study will start off by setting the Kinect v2 on a tripod located on the corner of the tennis court. This corner is defined by the baseline and the outer edge of one of the two doubles alleys, and will vary depending on which direction the subject is facing while executing a tennis groundstroke. On the opposite side of the net, we will set up a grid of 8.4x7.5 feet rectangles. A cone will be set in the center of one these rectangles which will define the target rectangle. Once the target is set and the subject has warmed up, we will give the subject 10 attempts try to hit the target while the research team records the landing of each attempt in the defined grid. If a subject misses by not making contact with the ball or by hitting the ball into the

net, they will be granted one supplementary attempt. Each attempt will consist of the co-investigator hitting a ball at slow pace from the opposite side net to the subject who will be waiting for the ball at the baseline. The ball will be fed to the subject in such a way that the subject will be able to strike the ball while staying the Kinect's field of vision while the Kinect is recording. There will be six different targets set up for the forehand portion of the study, as well for the backhand portion of the study. Similarly, the subject will be given 10 attempts for each target.

For the serve portion of the study, a different size grid will be set up consisting of 3.5x3.5 feet squares and will be set on a service box diagonally across from the subject. A cone will then be placed at the center of one of the rectangles to define the target rectangle. Starting from the "deuce side" of the court, the subject will stand 3 feet from the center of the baseline and take 10 attempts at hitting the target while performing their serve motion. For each side of serve position, "deuce" and "ad" side, the subject will be required to hit two different targets from each of these serving sides. From each of the two sides, one target will be located in the corner of the service box that is closest to the center of the court, the second target will be located on the opposite corner of the box close to one of the doubles alley. By placing targets in opposite sides of the courts we can expect to be able to denote biomechanical features that correlate to the different trajectories of the tennis ball. Figure 5.1.0 below displays the grid with reference to a regulation size tennis court, the targets of the study, and the locations where the Kinect would be placed for recording depicted by the blue dots.

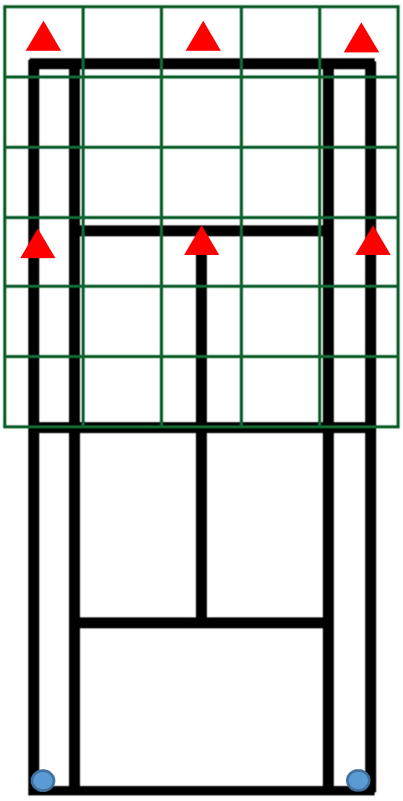


Fig. 5.1.0 Ground stroke portion of the study where the red triangles represent targets

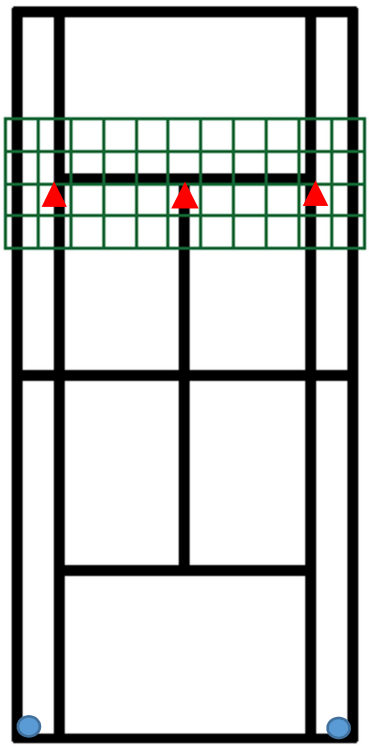


Figure 5.1.1 Serve portion of the study

5.2 Signal Processing

The Kinect's frame rate can fluctuate below the advertised 30 frames per seconds. This variable frame rate is not controllable by the user and mainly relies on the hardware components of the sensor and the user's computer resources. In order to compensate for fluctuation in frame rate, the Kinect raw data will be interpolated up to the advertised frame rate of 30 frames per second. And then it will be passed through a 4^{th} order Butterworth filter. Once the data is interpolated and filtered, we will gather the data of the actual swing. Taking into consideration different swing speed across subjects, we have determined that the dominant side wrist of the participant has denotable maxima and minima that indicate the start and end of the each swing. More

specifically, the Y component of the wrist joint signal demonstrates that tennis athletes accelerate their dominant arm by swinging in a more vertical direction.

Since the position of the subject with respect to the Kinect will not be constant for every swing across all subjects, we will have to measure distances and angles with respect to the subject's own body. Alternatively, we can manipulate the data by removing part of the *x* component of the 3D coordinates in a way to relocate the body coordinates into a fixed position for each frame of the data collected

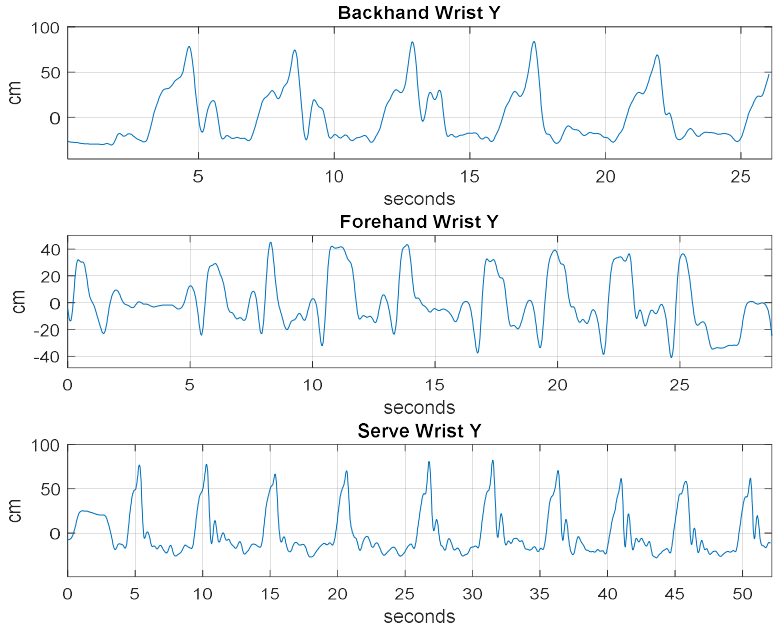


Fig 5.2.0

5.3 Accuracy Monitoring

In order to monitor the accuracy of the Kinect skeletal tracking, the research team will clinically derive each subject's ulna, humerus, femur and tibia prior to running the protocol and compare with the calculated Kinect measurements. These segments lengths will be hand-measured by a single investigator using the following proximal and distal landmarks: acromion process to

lateral humeral epicondyle (humerus); radial head to radial stylus (ulna); greater trochanter to lateral femoral condyle (femur); palpated lateral joint space to lateral malleolus (tibia). The Kinect derived measurements will be obtained by calculating the Euclidian distance between the mentioned joints.

$$\boldsymbol{d_{fr}} = \sqrt{\sum_{\boldsymbol{dim=1}}^{\boldsymbol{3}} \left(\boldsymbol{joint_{1_{dim}}(fr) - joint_{2_{dim}}(fr)}\right)^{\boldsymbol{2}}}$$

Equation 5.3.0 Euclidian Distance

6.0 Implementation

In this section we will describe how the proposed work will feed the acquired data from section 5.0 to the machine learning techniques mentioned in 4.0 and apply them to a set hypothesized biomechanical features in order to meet our objectives. The features we hypothesize will determine the direction of the ball is the angular displacement of the subject's dominant-side hip and shoulder. In other words, we predict that the angular displacements of the two mentioned joints will have a correlation to the amount of change in direction required to hit a present target. We also predict that the angles created by the subjects' ulna and humerus during the swing are correlated to the distance between the subject and the target as well as the direction. More specifically, the mentioned angle will become more obtuse throughout the swing as the subject aims for a target that is further away from him or herself.

6.1 Random Forest Implementation

In this work, the random forest algorithm will be used to classify the window of skeletal tracking data into one of six swing classes; right-handed or left-handed forehand, backhand, or serve. We propose making a set of weak learners, of maximum depth of five splitting nodes. The ultimate

goal of our random forest design is to reduce the entropy or Gini index on our training data to achieve optimal classification efficacy.

6.2 SVM Implementation

Once we have classified the input window as a forehand, backhand, or serve we will predict the trajectory of the swing. In this work SVMs will be used to predict trajectory of the ball. We anticipate that the biomechanics of the same swing in the presence of two different targets will display overlap in the feature space and will likely not be linearly separable. Moreover, if noticeable features exist to classify ball trajectory we anticipate these variances to be relatively small and will lead us to rely heavily on the accuracy of the Kinect Sensor.

To quantify the angle between the ulna and humerus in to a single feature value, we will measure the average radial distance between the dominant side hip to the dominant side wrist throughout the swing by calculating the Euclidian distance, equation 5.3.0. This average distance return a single value that will correlate the angle since the distance will increase as the elbow angles increase. In order to compensate for variance in arm length across subjects, we plan on normalizing the mentioned distance by a factor of subject height. As for the quantification of the angular velocity of hips and shoulders we plan on calculating angle differential with respect to time along the Kinect's y-axis. This processes essentially simplifies to calculating the angle between each frame of the swing while dividing by the sampling rate of 1/30 seconds. Figure 7.2.0 denotes the rotation of the hips in the Kinect's XZ plane where the blue denotes the frame after the black frame. The midpoint between hips and between shoulders will be defined by *spine base* and *spine shoulder* points respectively.

$$angular\ velocity = \frac{d\theta}{dt} \qquad \text{Eq.6.2.0}$$

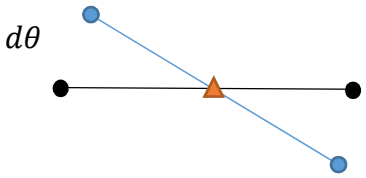


Fig. 6.2.0

6.3 HMM Implementation

Since HMMs are a double stochastic process, we can apply this to our application if consider the measurable stochastics process the human action, and the underlying stochastic process as the knowledge or strategy behind it [7] . HMM implementation leads to 3 processes [8]:

1. **Evaluation** – given the feature observation sequence $O = O_1\ O_2\ ...\ O_T$ and model λ = (A, B, Π), we will compute the probability of the observation sequence given the model, P(O| λ), using the forward algorithm.

   The forward algorithm is defined as:

$$\alpha_j(t) = P(O_1\ O_2..O_T|\lambda) \quad \text{Eq. 6.3.1}$$

$$\alpha_j(1) = \pi_j b_{jO_1} \quad \text{Eq. 6.3.2}$$

$$\alpha_j(t+1) = [\sum_{i=1}^{N} \alpha_i(t)a_{ij}]\, b_j(O_{t+1}) \quad 1 \le t < T-1\ \&\ 1 \le j \le N \quad \text{Eq. 6.3.3}$$

$$P(\mathrm{O}|\,\lambda) = \sum_{i=1}^{N} \alpha_i(T) \quad \text{Eq. 6.3.4}$$

2. **Training** – Given a set of training sequences of a feature of T samples long $O = O_1\ O_2..O_T$, find matrix A and B. In this work we will use the Baum-Welch algorithm to estimate and adjust the A and B matrices to create our model and maximize P(O| λ). We will also experiment with number of states ranging from 3 to 10. In order to understand

the Baum-Welch algorithm, also referred to as the forward-backward algorithm, it is also necessary to introduce the backward algorithm shown in equation 6.3.5.

$$\beta_i(t) = [\sum_{j=1}^{N} a_{ij} b_j(O_{t+1})]\ \beta_{t+1}(j) \quad t = T-1, T-2, \dots, 1\ \ \&\ 1 \le j \le N \qquad \text{Eq. 6.3.5}$$

Using the forward and backward variables we can derive the probability of being in state j at time t.

$$\gamma_{ij}(t) = \frac{\alpha_j(t-1) a_{ij} b_{ij} \beta_i(t)}{\mathrm{P(O|}\,\lambda)} \qquad \text{Eq. 6.3.6}$$

Ultimately, we can estimate each individual component of the A and B matrix using equations 6.3.7 and 6.3.8 respectively.

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T} \gamma_{ij}(t)}{\sum_{t=1}^{T} \sum_{k=1}^{c} \gamma_{ik}(t)} \qquad \text{Eq. 6.3.7}$$

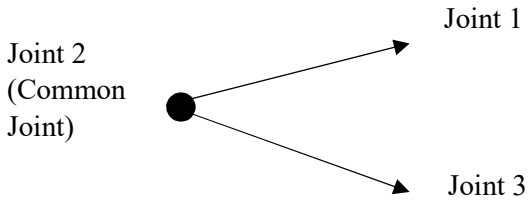
$$\hat{b}_{ij} = \frac{\sum_{t=1}^{T} \sum_{t=1}^{l} \gamma_{jl}(t)}{\sum_{t=1}^{T} \sum_{k=1}^{l} \gamma_{jl}(t)} \qquad \text{Eq. 6.3.8}$$

3. **Decoding** – To determine the most probable state path $Q = Q_1\ Q_2 .. Q_T$, given output sequence $O = O_1\ O_2 .. O_T$ and model λ we will use Viterbi algorithm works best for Left Right bounded models [9]. This most likely single path of length T becomes

$$\delta_j(t) = \max(Q_1\ Q_2 .. Q_T = j, O_1\ O_2 .. O_T | \lambda) \qquad \text{Eq. 6.3.9}$$

In order to create an input for this model we will take into consideration three angles from the dominant side of the subject as a function of time. These three angles will be calculated using three joints from the skeletal data and equation 3.7.1. For each calculated angle, we will need the three joints with one common joint to create two vectors.

Fig. 6.3.1



$$\emptyset_i = \cos^{-1}\left(\frac{u \cdot v}{||u||\,||v||}\right) \qquad \text{eq. 6.3.10}$$

Angle	Joint 1	Joint 2	Joint 3	Sequence
$\emptyset$	Shoulder	Elbow	Wrist	$\emptyset_1, \emptyset_2, \ldots . \emptyset_T$
θ	Elbow	Shoulder	Neck	$\theta_1, \theta_2, \ldots \theta_T$
ω	Elbow	Shoulder	hip	$\omega_1, \omega_2, \ldots \omega_T$

With the calculated angle sequences we can create the following sequence input, V, for our model by concatenating the three sequences as shown in equation 6.3.11.

$$O = [\, \emptyset_1, \emptyset_2, \ldots . \emptyset_T, \theta_1, \theta_2, \ldots \theta_T, \omega_1, \omega_2, \ldots \omega_T \,] \quad \text{Eq. 6.3.11}$$

We will train the two different models per swing; one cross-court model and one down the line model for the ground strokes, and one out-wide and down-the-T model for the serve.

6.4 MLP implementation

The MLP does not require any feature extraction from our end so we plan on feeding all 25 joint center 3D data of a fixed swing window. We plan on allocating one neuron per joint in the hidden, such that each neuron will have 3 inputs. In addition we will use a sigmoid function

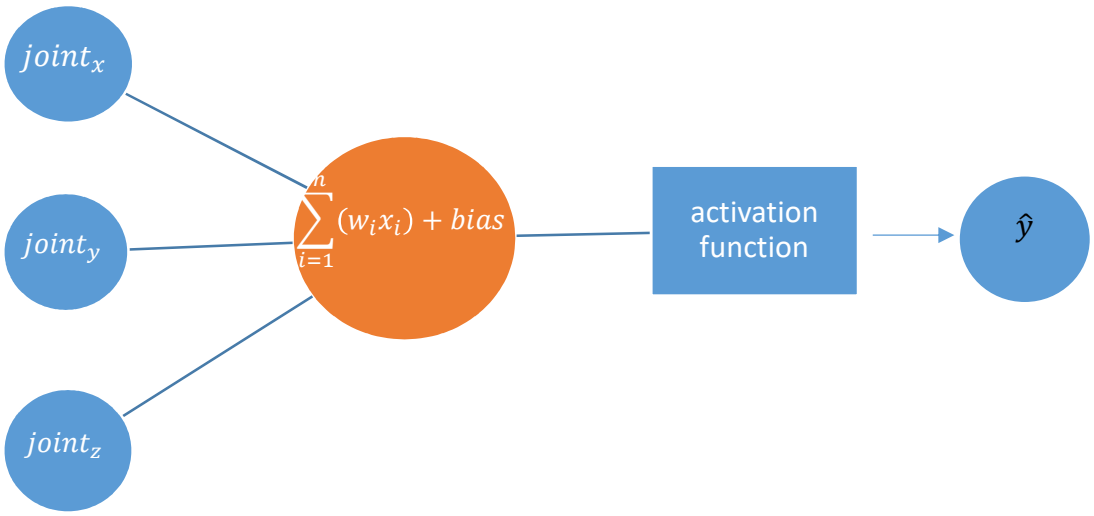
(equation 6.4.0) as the activation function, but will also experiment hyperbolic tangent (equation 6.4.1).

$$\varphi(x) = \frac{1}{1 + e^{-x}}$$

Eq. 6.4.0 sigmoid activation function

$$\varphi(x) = \frac{e^{x} - e^{-x}}{1 + e^{-x}}$$

Eq. 6.4.1 hyperbolic tangent activation function



7.0 Proposed Work Overview

In this work we propose using the random forest algorithm to classify the skeletal tracking data of a tennis athlete as a forehand, backhand, or serve. Once the swing is classified into one of these three classes, we plan on using support vector machines in the biomechanical feature space to predict the general direction of the tennis ball. Since we expect to encounter overlap in data of the same swing that differs in ball trajectory, we plan on separating the two sub-classes using a

maximum margin classifier and a kernel function. Previous research indicates that the linear kernel was more efficient in gesture classification than RBF (radial basis function) kernel [10] but we will investigate in other kernel functions if results are not satisfactory. Nevertheless, SVMs allow us to use an unlimited amount of dimensions in the feature space, so we expect to find one, or a pair of features that make the ball trajectory distinguishable with the Kinect and its limited accuracy.

In addition, we are proposing the use of a hidden Markov model, a Bakis Model more specifically, an approach that will take into consideration a biomechanical feature as a function of time to develop a model for each of three tennis swings, in each general direction. In more detail, we will train 6 models, $\lambda = (A, B, \Pi)$, where $\Pi = [1\ 0\ ..\ 0]$. As our deep learning approach, we will develop a multilayer perceptron in which we will pass the interpolated skeletal data so that we will not have to rely on the accuracy of the Kinect since we will not be calculating any features.

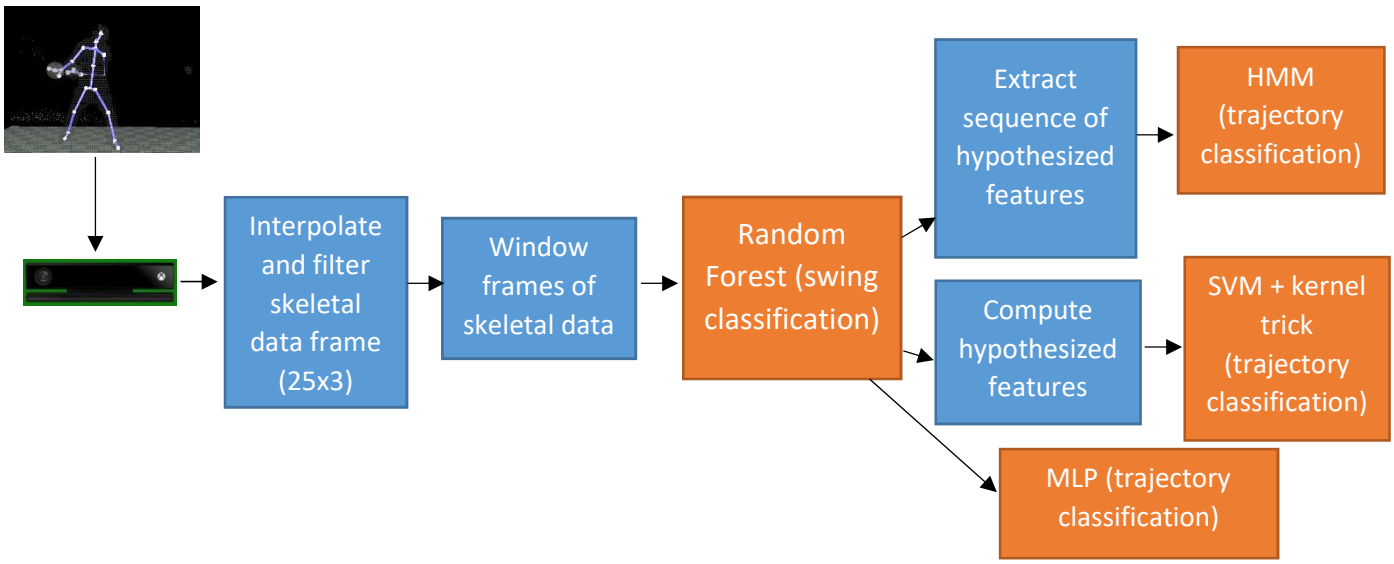


Fig. 7.0 The design cycle of the proposed work.

7.1 Proposed Work Timeline

The data acquisition software that will record athletes with the sensor has been written and will be optimized to efficiently and reliably find "zero crossings" that indicate the start of each swing to facilitate the annotation process. This work will be allocated four weeks for data collection and annotating the data. The IRB has approved for a maximum of 50 participants in this project. Each individual session will last 30 minutes, which equals to a maximum amount of 25 hours of data collection excluding travel time and setup time. Once the all the data has been collected, we will allocate 70% of the raw data to the training set and 30% to the evaluation set. Before starting the design of the mentioned machine learning algorithms, we will perform a principal component analysis on the hypothesized features as a baseline test to work efficiently. Seven weeks will be allocated to the design and optimization of classifying algorithms. Lastly five weeks will be allocated of analyzing results and defense preparation.

References

[1] D. Whiteside and M. Reid, "Spatial characteristics of professional tennis serves with implications for serving aces: A machine learning approach," *J. Sports Sci.*, vol. 35, no. 7, pp. 648–654, Apr. 2017.

[2] "BodyFrame Class." [Online]. Available: https://msdn.microsoft.com/en-us/library/windowspreview.kinect.bodyframe.aspx. [Accessed: 14-Dec-2017].

[3] "JointType Enumeration." [Online]. Available: https://msdn.microsoft.com/en-us/library/microsoft.kinect.jointtype.aspx. [Accessed: 14-Dec-2017].

[4] K. Otte *et al.*, "Accuracy and Reliability of the Kinect Version 2 for Clinical Measurement of Motor Function," *PLOS ONE*, vol. 11, no. 11, p. e0166532, Nov. 2016.

[5] "Coordinate Spaces." [Online]. Available: https://msdn.microsoft.com/en-us/library/hh973078.aspx. [Accessed: 14-Dec-2017].

[6] R. . Duda and P. E. Hart, *Pattern Classification*, 2nd ed. Wiley, 2000.

[7] J. Yang, Y. Xu, and C. S. Chen, "Hidden Markov model approach to skill learning and its application in telerobotics," in *[1993] Proceedings IEEE International Conference on Robotics and Automation*, 1993, pp. 396–402 vol.1.

[8] S. Mitra and T. Acharya, "Gesture Recognition: A Survey," *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.*, vol. 37, no. 3, pp. 311–324, May 2007.

[9] H. Hiyadi, F. Ababsa, C. Montagne, E. H. Bouyakhf, and F. Regragui, "A depth-based approach for 3D dynamic gesture recognition," in *2015 12th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, 2015, vol. 02, pp. 103–110.

[10] S. Bhattacharya, B. Czejdo, and N. Perez, "Gesture classification with machine learning using Kinect sensor data," in *2012 Third International Conference on Emerging Applications of Information Technology*, 2012, pp. 348–351.