Evaluating Body Posture and Ball Trajectory to Determine Fast Pitch Softball Pitch Types

Alyssa Wolejko

March 24, 2014

Abstract

This document describes an instructional system to analyze a fast pitch softball pitcher's pitching motion and the location of the ball in the strike zone in order to determine the pitch type. Using a Microsoft Kinect sensor to evaluate a pitchers posture during their pitching motion and the ball trajectory of that pitch does this. The OpenNI open source API, which operates in conjunction with the Kinect, allows users to track the joints of a person's body and from there determine the angles of specific joints. The angles of certain joints through the pitching motion can determine the pitch type. A visual system tracking the ball can determine the pitch location in some predetermined strike zone. Taken together, these two sources of information allow us to compile a database that can be used as a statistics book for the pitcher intended to be used as a training and instructional tool for elite pre-collegiate or collegiate athletes.

1 Introduction

"It only takes one pitch to make a difference." -Unknown

Every pitcher has technique and skills learned through instruction to help them overcome batters. Pitching is considered the most challenging action in the sport of softball. One pitch can make or break a game. To be a successful pitcher, years of training and instruction are needed. Elite pitchers have between 3 and 6 different types of pitches that move in different directions, have different speeds, and require varied body postures to throw them. One of the ways that a pitcher can be evaluated is by recording pitching statistics via pitch tracking. Pitch tracking is used in softball to keep track of the pitcher's statistics regarding their pitch types, for instance, whether the pitches are strikes or balls according to the strike zone, or if the pitches are correctly located. Pitching statistics are used to help a pitcher develop their skills and accuracy of a certain pitch. If the pitch is thrown within the strike zone then it is a strike, otherwise it is a ball. For an elite pitcher, the correct location of a pitch type is just an important in evaluating accuracy. Accuracy in pitching can be defined as throwing a certain pitch type and having that pitch go to a consistent spot in the strike zone. The aim of this project is to accuately determine the pitch type from a pitcher's body posture and pitch location.

2 Background

Every pitch that a pitcher throws looks very similar to an untrained eye, however, there are distinct differences in body posture and pitch location in the strike zone. All of the different pitches in softball are based off of the generic fastball. Therefore, the information that the fastball gives us is not drastically different than the rest of the pitch types. The final 2-3 seconds of a pitchers pitching motion is the most telling when it comes to distinguishing between pitches. The three pitches that are being observed in this study are a curveball, rise ball, and an off-speed pitch. Each of these three pitches vary in speed, body angle relative to the plane of the ground, and the throwing hand position at the end of pitching motion. The hand position is assumed to be the most important characteristic.

2.1 Visual Differences between Pitch Types

2.1.1 Curveball

The pitcher is assumed to be a right-handed thrower. The pitcher has the ending body position with their torso just over their front knee (which on a right-handed pitcher is the torso over the left knee). The pitchers knees are pinched together with the right knee slightly in front of the left knee. This will also bring the left hip forward. Most importantly, the right hand will come across the body; finishing along the hip line.

2.1.2 Change-Up

The pitcher is assumed to be a right-handed thrower. The pitcher has the ending body position with their torso just behind their front knee (which on a right-handed pitcher is the torso just behind the left knee). The speed of this pitch is dramatically different from the other pitches (it is slower). Most importantly, the ending hand position is straight out in front of the pitcher with the palm facing down.

2.1.3 Riseball

The pitcher is assumed to be a right-handed thrower. The pitcher has the ending body position with their torso drastically behind their front knee (which on a right-handed pitcher is the torso well behind the left knee). The speed of this pitch is much faster than the other pitches that a pitcher throws. Most importantly, the ending hand position is up by the pitchers shoulder (bent at a right angle to a an acute angle).

2.2 Kinect Hardware and Software

2.2.1 Kinect Hardware

The Microsoft Kinect sensor is used to collect joint data for the experiment. This hardware has an array of sensors including an infrared sensor, RBG camera, 3D depth sensor, and multiple microphones. The Kinect can be connected, via USB, to a computer in order to collect data with the appropriate drivers.

Figure 1: Image of the Microsoft Kinect sensor [3]

2.2.2 Kinect Software

Primarily, we are going to use the OpenNI NITE library in conjunction with the Kinect hardware to collect joint data. OpenNI is the standard open source framework for 3D sensing. We are trying to track coordinates of joints in 3D space, so we will use this library. OpenNI has support for the Kinect sensor, therefore the Kinect will become a tool while using the OpenNI software. [11]

3 Related Work

Research similar to the proposed project concerns object tracking uses depth sensors similar to the Microsoft Kinect, for it's ability to track joints on a human skeleton in three dimensional space. Most of the research with posture tracking uses the MS Kinect OpenNI Skeletal Tracker; which is a way of tracking 15 joints of the body in 3D space, found in the OpenNI library. Instead of hardware being strapped to the athlete, these papers focus on the use of computer vision to obtain the results similar to those received through hardware attached to the body itself. There is an example of Kinect Skeletal Tracking being used in classifying football stances in a paper by Kato, Namiki, and Thepvilojanapong [4]. The Kinect OpenNI tracker is used to track up to 20 joints on the body and the angles between those joints can be computed to characterize the relationship of the joints to each other instead of in absolute space. The work by Monir, Rubya, and Rerdous describes the angular representation of skeletal data that is recorded by the Kinects OpenNI tracker in order to track the joints of a person's golf swing [6]. Once the information is gathered from the Kinect, in order to determine the pitch type a classifier is needed, such as the one mentioned in the paper by Kato, Namiki, and Thepvilojanapong [4]. A cross-correlation function is used in conjunction with a support vector machine (SVM) classifier to distinguish between football stances to determine whether the

player has appropriate form.

We can think of classifying pitch types as a type of dynamic gesture recognition. A pitching motion is a quick, fluid motion. A pitching motion requires a large amount of movement over a large area. During the duration of a pitch, we are keeping track of the joints in three dimensional space. This is different than static gestures; which are gestures that are held in place or have little to no movement. At Cornell, they built a program that would recognize military gestures and move a robot accordingly. The students working on the project used a Microsoft Kinect to record the training data and the live data to use a classifier on. Using a muti-instance classifier allows us to make distinctions between body movements. Scripts were made to extract the coordinate values for each joint, these XYZ values can be used to find the phi and theta values of the joint [1]. The phi and theta values are angles computed from a point in 3D space; in our case, the points that are being used are the joint coordinates in 3D space. They used both static and dynamic gestures to train their system.

The data mining and machine learning software, Weka [2], is being used to make these decisions that will ultimately classify the data it is given. Weka is capable of handling multiple instances. In our case, one pitching session contains multiple pitching motions. More importantly, one pitch is a series of joint data readings over time. Some of the classifiers that can be used are: CitationKNN [10], NaiveBayes [13], J48 [7], simple decision tree, etc. The Weka Explorer allows the systematic comparison of the predictive performance of Weka's machine learning algorithms on a collection of data sets. [12] So in this thesis we will use a multi-instance classifier to evaluate the dynamic gesture of a pitching motion in order to distinguish between pitch types.

4 Detecting and Classifying Pitches

We first need to recognize the pitch type, then recognize where the pitch goes within the strike zone. The overarching questions I seek to address in this thesis are: Can we accurately and automatically distinguish between fast pitch softball pitch types? Can we automatically determine the accuracy of a pitcher's pitches?

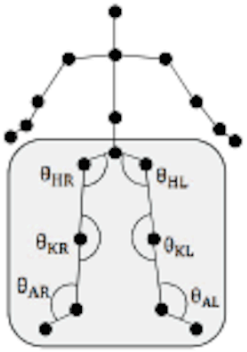

Fig. 1. Nine joints of interest.

Figure 2: Example of nine angular joints used to analyze football stances.[4]

To address these goals, we will create a computer program that can be trained to classify typical pitch types in order to evaluate the accuracy of a pitch. The system can be used as an instructional tool or simply a tool that is used to keep accurate pitching statistics. To achieve this, there are two main tasks:

- Classifying Posture: The posture of a pitcher can 'give-away' a certain pitch type. A pitching motion is the pitchers delivery of the ball to the batter from the pitching mound to home plate; one fluid motion constitutes a single pitch. By evaluating the pitcher's pitching motion we believe we can determine which pitch is being thrown. Specifically, it is our belief that the last couple seconds of a pitcher's motion is the most distinctive. When looking at posture, finding coordinates and angles of specified joints will prove useful. The angles between joints are computed from the 3D coordinate data of each joint. More emphasis should be put on finding the angles of the joints because that data will be more useful in specifying specific pitches. The relative positions of the joints are how the joints relate to each other instead of in absolute space. For example, the right arm can be bent at a 65 degree angle in 2 different pitching motions, however the absolute XYZ values may not be the same. The relative positions of the joints will also be more useful when trying to use the system on more than one pitcher because all elite pitcher's have the same, if not similar, pitching mechanics. Though the coordinates of joints are important, one pitcher's arm may be in a different coordinate plane than another pitcher's arm. The system will be more applicable to more athletes when angles of joints are used over joint

coordinates. We are going to collect data (via the Kinect) and see, using machine learning, if we can classify different pitch types. The break down on a pitcher's technique can be found in the APPROACH and METHODS section of this proposal.

- Pitch Location: Pitch location relative to the batter and home plate also plays a significant role in classifying pitch types. A pitch location is also specific to each pitch. For example, a curveball will have a location on the outside part of the strike zone and a riseball will have a location on the upper part of the strike zone. The body posture of a pitcher affects the location of the pitch. A riseball body posture will result in the pitch location of a riseball. Since a pitch location is the result of the body posture that the pitcher has, it can either confirm or falsify the hypothesis drawn from the pitcher's body posture as mentioned above. One way to approach the task of finding the pitch location is to apply visual object tracking to the ball exploiting its neon yellow color. By tracking the ball, it's position over a certain amount of time can be tracked into the strike-zone. If a pitch is within the strike zone it is considered a strike. If a pitch is outside the strike zone, it is considered a ball. We will attempt a more fine-grained analysis into nine zones. The strike zone is a rectangle with a height from a batter's knees to chest, and a width from one corner of home plate to the other. In order to be accurate, a rise ball should be in any of the top three zones (Figure 1: Zone1, Zone2, Zone3), a curve ball should be in the two outside-lower zones (Figure 1: Zone4, Zone7), and a change-up should be in any of the bottom three zones (Figure 1: Zone7, Zone8, Zone9).

5 Evaluation

The goal of the project is ultimately to evaluate the technique and posture of a pitcher's pitching motion. The system can be used as an instructional tool. To confirm the pitch types we must get the data from the Kinect; such as the coordinates and angles of the throwing arm and torso. The angles of joints can be computed from the three-dimensional coordinate data (recorded by the Kinect). We can get visual data from the Kinect as well by using the RGB camera built into the Kinect. Getting an image of the strike zone can aid in categorizing the pitch type. We can also compare the data to a catcher's evaluation of which

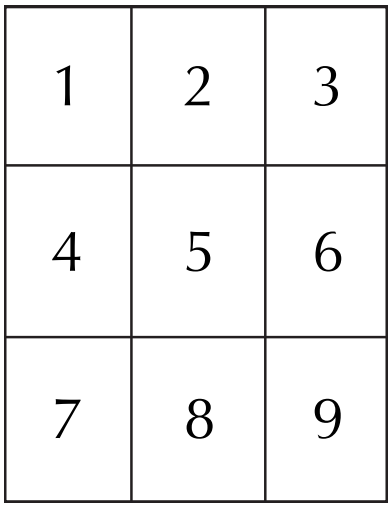

Figure 3: Example of a predetermined strike zone

pitch type it was and if it was supposed to be in that particular location.

6 Approach

This problem can be broken up into many tasks and subtasks. First, using the OpenNI tracker for Kinect and putting the data in a bag file using the Robot Operating System (ROS)[8] records detailed data about the joints of the body. ROS provides us with an interface to the Kinect; meaning it allows us to interact with the Kinect and the data that it records from the sensors. A bag file is a way of collecting data from the Kinect Sensor. The data must then be taken from the bag file and converted to a comma separated value (CSV) format so that Weka (machine learning software) [5] will be able to make instances out of this data. These instances can then be treated as independent instances of an event, or, in our case, collapsed into multi-instance bags where each bag represents a single pitch of a particular pitch type. The data within the CSV file, generated from the Kinect Sensor data, is passed through a classifier to determine the pitch type from the most repetitive and distinguishable features of a pitcher's pitching motion. Then, we will evaluate the data to determine the accuracy of the pitcher's pitches. The pitch location can aid in the identification of a pitch type as well. By taking a picture of the strike zone when the ball is within it,

the pitch location within the grid of the strike zone can be obtained.

Important Notes and Terminology A pitching motion is one fluid body movement when the pitcher delivers the ball to the batter standing at home plate. It takes about 7 seconds for one pitching motion to occur. The start of a pitching motion is indicated by a static body position being held for 3 seconds. This means that the position of the ball and the coordinate positions of the joints don't change over a time period of at least 2 seconds. The Kinect publishes at a rate of 30 hertz or frames per second. One frame is the collective coordinate information (x, y, z) of each joint being tracked at a particular moment in time. From the coordinate data, the spherical coordinates (theta and phi) can be calculated occurring at the same rate. Calculating the spherical data from the coordinate data is important when there is more than one athlete using the program because the throwing hand of different pitchers may be in different planes, but it is assumed that the relative positions of joints will be similar. Therefore if the program is trained on spherical data rather than coordinate data, it will make it possible to use the program on multiple athletes; since all elite pitchers have extremely similar pitching mechanics.

6.1 Visual Data Collection

To collect the data on each pitch, 2 different softball pitchers will perform their pitching motions for each of the three specified pitches in front of the Kinect sensor (sensor behind the pitcher, while the pitcher has their back to the sensor). For experimental purposes the Kinect was set up to the right side of the pitcher facing them, as well as at home plate facing the pitcher. When the side position and the front position were tested, both frequently lost track of the pitcher's joints. In the front position, the distance of about 30 feet affected the quality of the joint data and frequently lost track of the fast moving joints. From the side position, the joint movement was difficult to see because most of the joints aligned with one another. Initial experimentation tells us that this positioning of the sensor allows the camera to see dramatic variations in joint movement and have a visual of the strike zone. The three-dimensional coordinates of the joints in space are recorded and the spherical data of the angles is calculated from the XYZ values of each joint. Figure 4 is a visual representation of what is being classified.

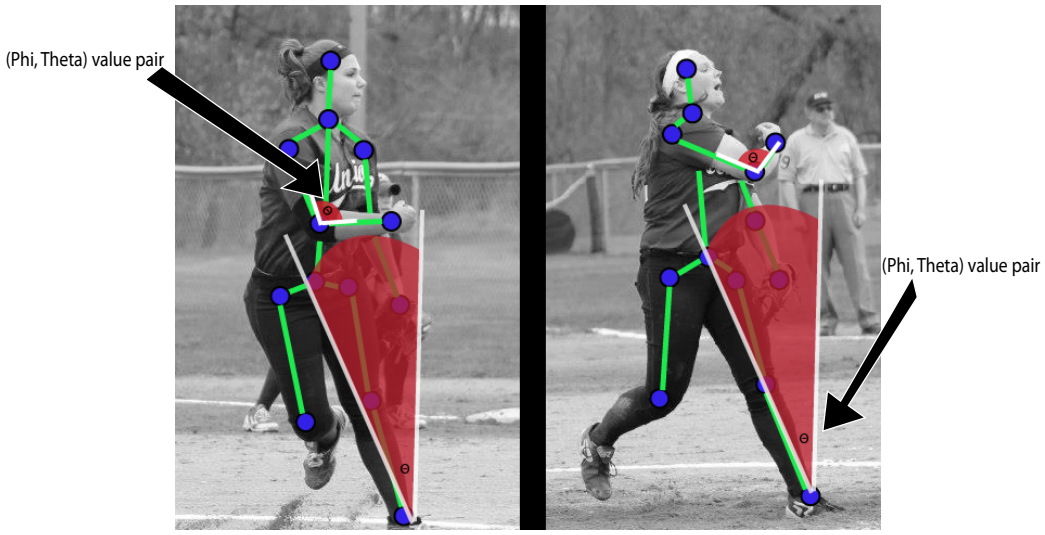

Figure 4: Coordinates and Angles of Joints

6.2 Formatting Data for the Classifier

The data collected by the Kinect must be formatted into a file format that is able to be interpreted by the machine learning software, Weka.

- CSV: A CSV file is comprised of data that is separated by commas. CSV is the acronym for 'comma-separated values'. This file format stores tabular data in plain-text form. Each attribute of the file is separted by commas. It is simple converting a CSV to an ARFF file because both have data that we consider to be attributes. However, Weka has a filter already for interpreting CSV files.

6.2.1 Structure of the CSV

A pitching session is the compilation of 20 pitching motions of the same pitch type being thrown. Each of the pitch types are referred to as a 'class'. We want to make a 'Weka Bag' for each instance of a pitching motion. The attributes are the x, y, z coordinates of each joint and the theta and phi (Θ, Φ) value pair for select joints. The values of these attributes are the float values that are published by Kinect. There are multiple frames that go into one instance because the publishing rate of the Kinect is 30 times per second. So if a pitching motion is about 7 seconds long and the Kinect publishes at a rate of 30 instances per second, then there are about 210 frames that make up one pitching motion. For example, there are 210 lines in the CSV file (each one being an instance of a pitch). In figure 3, an example of the compiled CSV file is shown.

bag_id	head_x	head_y	head_z	neck_x	neck_y	neck_z	
rise1	3.4939559	-0.1656096	0.39038289	3.51965177	-0.1635591	0.14555949	
rise1	3.51095589	-0.1666942	0.38621461	3.53004003	-0.1701876	0.15874247	...
rise1	3.51845589	-0.170685	0.38363646	3.53795999	-0.1747599	0.16473668	
.							
.							
.							
rise21	0.67692097	0.19632523	-0.2327173	0.74427188	0.05364816	-0.3690574	
rise21	0.67692091	0.19632524	-0.2327174	0.74427188	0.05364816	-0.3690574	...
curve1	3.30430567	0.05586334	0.32392481	3.34454711	0.03981554	0.16889079	
curve1	3.30480572	0.04691528	0.32778611	3.34144354	0.03120001	0.14642881	
.							
.							
.							
curve14	0.73423471	0.26018379	-0.2144263	0.83486958	0.12863351	-0.3501153	
curve14	0.73024967	0.26072241	-0.2103523	0.83277315	0.13132261	-0.3463738	
changeup1	2.90533992	-0.1240712	0.29465415	2.91739579	-0.1256008	0.08388426	...
changeup1	2.90811042	-0.1231387	0.29474292	2.91976078	-0.1247075	0.08453799	
.							
.							
.							

Figure 5: CSV file

In this example, the nominal attribute 'bag_id' will remain a consistent value for every frame that is in one pitching motion. This value will change when the next instance of the pitch type is started. There are X, Y, and Z values for each joint which are correspondingly labeled nominally by the joint name followed by an underscore and the coordinate. Currently, we are tracking 15 total joints (which is the maximum number of joints the OpenNI tracker tracks). There are also attributes that consist of theta (Θ) and phi (Φ) spherical

angles. This is calculated from the X, Y, and Z coordinate values by converting from Cartesian coordinates to spherical coordinates. A pitcher's basic technique and fundamentals are very similar to those of another elite pitcher's. Certain pitch types have the same body posture because it takes a certain posture to exert force on the ball. Therefore, two different elite pitcher's will have the similar body postures when throwing the same pitch types. See Figure 4 for a visual representation of how the same elbow angle can occur in different coordinate planes.

6.3 Multiple Instances

In each file of recorded data there are 20 repetitions of the 3 pitches that are being focused on; which comprises a session. Each pitch within a session is made up of multiple instances; a collection of instances makes one pitch. Each of these instance collections are organized by an identification number which means that there are several frames that a part of one singular pitch and that there are several pitches that are a part of one Kinect data file. By transforming these instance collections into a format that Weka can handle. One of the classifiers in Weka allows us to compress multiple instances into an instance that is considered the average instance value. By doing this, it accounts for the small variations in coordinates that might be in the recorded data. Therefore, all of the classification algorithms provided in Weka will be able to be used on the data because it was averaged into one instance. When using this averaging technique, we use a Simple Multi-Instance Filter before running a particular classifier algorithm.

6.4 Classification

This data can be passed through a classifier. We are trying to determine if the data is susceptible to classification. The classifier will be able to make decisions about a pitch by using the built in classifying algorithms. The most successful classifier performed on the data is the CitationKNN multi-instance classifier. The other classifiers that were tested needed to take in a single instance (the average instance talked about earlier). These classifiers did not perform as well as the CitationKNN classifer. The goal of classification is to make the classifier preform as optimally as possible without decreasing the accuracy significantly. We tried a variety of classifiers including: CitationKNN [10], NaiveBayes [13], J48 [7]. CitationKNN significantly

out performed all the others.

CitationKNN Background K-Nearest-Neighbor: an object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its k nearest neighbors. The neighbors are taken from a set of objects for which the correct classification is known. The algorithm for KNN, in the classification phase, k is a user-defined constant, and an unlabeled vector is classified by assigning the label which is most frequent among the k training samples nearest to that query point.

Experimenting with Classifiers CitationKNN has proved to be the best classifier to use with the dynamic gestures that we are trying to classify. According the table below, the CitationKNN classifier outpreformed the SimpleMI classifier everytime. SimpleMI is a classifier that allows us to compress multiple instances into an average of these instances. Doing this would allow us to use any type of classifier on the data; not just a multi-instance classifier. However, CitationKNN clearly does a better job at classifying than the SimpleMI. So we are posed with the question, "How can CitationKNN be optimized?, How can it run faster?".

Hypothesis: CitationKNN will preform faster on the data sets if the number of instances are reduced, however the accuracy may reduce.

- Data1 is the data produced from a recording of myself throwing 20 pitches of the 3 different pitches (curve, rise, changeup). Before each pitch in a session, there is a 2 second pause to signify the start of a pitch. There are 14400 total instances in Data1. When the data is reduced to 7200 instances, every other instance of the original Data1 was used.
- Data2 is the data produced from a recording of a second elite pitcher (not the author) throwing 20 pitches of the 3 different pitches (curve, rise, changeup). Before each pitch in a session, there is a 2 second pause to signify the start of a pitch. There are 14400 total instances in Data2. When the data is reduced to 7200 instances, every other instance of the original Data2 was used.

Below is a table of results on experiments done to determine the better classifier to use with the data

(CitationKNN versus SimpleMI):

Data	**Specifics**	**Instances**	**CitationKNN**	**SimpleMI-J48**
Data1	—	14400	100%	85%
Data1	reduced xyz*	14400	97%	80%
Data2	—	14400	100%	85%
Data2	reduced xyz*	14400	97%	80%
Data2 trained on Data1	—	14400	100%	98%
Data2 trained on Data1	reduced xyz*	14400	36%	23%
Data2 trained on Data1	4 thetaphi**	14400	100%	81%
Data2 trained on Data1	—	7200	43%	37%
Data2 trained on Data1	4 thetaphi**	7200	45%	38%

* torso (xyz), right_shoulder(xyz), right_elbow(xyz), right_hand(xyz), left_foot(xyz), right_knee(xyz), torso (θ, ϕ), elbow(θ, ϕ), shoulder(θ, ϕ), knee(θ, ϕ) ** torso_theta, torso_phi, elbow_theta, elbow_phi, shoulder_theta, shoulder_phi, knee_theta, knee_phi

In the table above, there is a comparison between the CitationKNN and J48 classifers. In the original files of collected data there are 14400 instances total. First, data set 1 was put through both classifiers using a 10-fold cross validation. A 10-fold cross validation breaks the training data into 10 sets of size $\frac{n}{10}$, trains on 9 datasets and tests on 1 dataset, then repeats 10 times and takes the mean accuracy. The 10-fold cross validation was preformed on both data sets that were recorded. In the attempt to prove the hypothesis of making the classifier perform faster, instances were taken out of the original data set. For example, the reduced XYZ values consisted of only a certain number of joints instead of the whole set of 15 joints. The torso (xyz), right_shoulder(xyz), right_elbow(xyz), right_hand(xyz), left_foot(xyz), right_knee(xyz), torso (θ, ϕ), elbow(θ, ϕ), shoulder(θ, ϕ), knee(θ, ϕ) were the specific joints chosen to see if reducing the amount of joints being tracked would impact the classification time. Reducing the number of joints also reduced the accuracy of both classifers being tested. According to the table, when one data set was classified there was about a 3% to 5% drop in accuracy. The drop in accuracy become more apparent when one of the data sets were used to classify the other. When one set is used as a training set for another, the classifier tries to correctly classify

the unknown data from one set using the known data from the other. Using Data1 as the training set, we correctly classified 100% of the instances with the CitationKNN classifer. However, CitationKNN still had a slow preformance time. So we reduced the XYZ values to only consist of torso (xyz), right_shoulder(xyz), right_elbow(xyz), right_hand(xyz), left_foot(xyz), right_knee(xyz), torso (θ, ϕ), elbow(θ, ϕ), shoulder(θ, ϕ), knee(θ, ϕ). The result from this reduction of joints resulted in a 36% accuracy rate (which is bad). The original data set was then reduced to just the theta and phi values (torso_theta, torso_phi, elbow_theta, elbow_phi, shoulder_theta, shoulder_phi, knee_theta, knee_phi) because of the hypothesis that states that the theta and phi values will become more important than the XYZ coordinate data. When the reduced data set is classified, via the training set, the CitationKNN classifier returns a 100% accuracy yet again, while the J48 classifier has an 83% accuracy rate. Reducing the number of joints and the number of instances of the data set was the next logical task to preform to see if the accuracy would decrease if the number of instances were significantly reduced. So we reduced the number of instances from 14400 to 7200 (half of the original number of instances) to see if the time of the classifier would reduce without losing accuracy. This was not the case since the accuracy of the classifier severely dropped to 45%. Preforming these experiments led us to the following conclusions...

6.4.1 Conclusions

The CitationKNN classifier preforms the best out of the classifiers offered. When using SimpleMI to compress a pitch into one single instance, too much accuracy is lost. Since the CitationKNN classifier takes a large amount of time to go through the data it's given, eliminating instances will make it preform faster. However, if too many instances are removed the accuracy decreases significantly.

7 Optimization Experiments

There are several ways that are thought to reduce the time it takes for the CitationKNN classifier to run. The following are examples to proposed experiments to be preformed:

1. Decrease the rate of which joint data is taken with the Kinect. Currently joint coordinates are being published 30 times per second. Start by decreasing this number to 15...and so on as long as the accuracy

percentage is retained.

2. Take the last 2-3 seconds of a pitching motion instead of the whole 6 seconds. This will determine if the last couple seconds are more useful than all of the data.

3. Distinguish the most important combinations of joints by trial and error method.

4. Convert all supposedly important joints to Phi and Theta values. Eliminate the not-as-important values as long as accuracy percentage is retained.

5. Use combinations of x,y,z coordinates and Phi/Theta values to try and retain accuracy.

8 Experiment Setup

A pre-determined strike zone is constructed on a large plywood board in a laboratory setting. The strike zone is a 3-by-3 grid that is about 2.5 ft. in width and 3 ft. in height. The strike zone is setup so that the bottom of the zone is at a batters knees (assuming the average height of the batter is 5'5" - 5'9") and reaches up to the letters of their jersey. The Kinect was set up at about 55ft from the plywood strike zone. On an actual field, the pitching mound is 63ft from home plate. However, for our purposes 55ft is a good distance from the strike zone to get accurate results and a good RGB image.

8.1 Image Evaluation

In terms of finding the location of the ball in a pre-determined strike zone, a picture of the strike zone can be taken and the location of the ball can be found. The RGB image of the strike zone can be pulled from the file that the Kinect sensors are being recorded to. Using an image of the strike zone replaces the challenge of tracking the ball, via object tracking, over a period of time. The position of the ball is only significant when it reaches the strike zone because the ending position will give us more information on which type of pitch it was. Using ROS we can get important data about the pitcher's body posture and pitch location from different nodes. Nodes are packages of open source code that can be subscribed to in order to get certain types of data. We will run several nodes to capture the RGB image, the joint coordinate

data, and the audio data from the Kinect sensors. The nodes that ROS is running are the RGB camera node, OpenNI skeletal tracker, and the audio common node. The audio data becomes extremely important when trying to find the correct moment to pull the RGB image from the file the Kinect recorded.

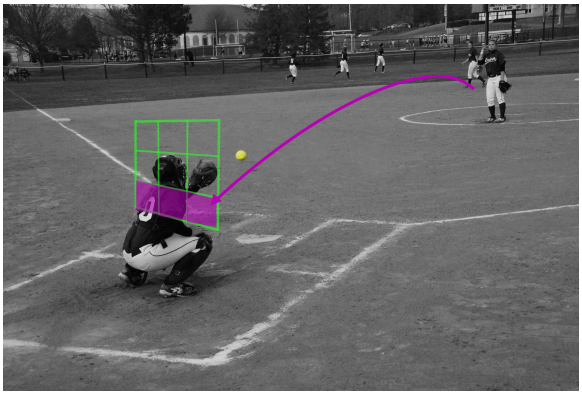

Figure 6: Changeup position in strike zone

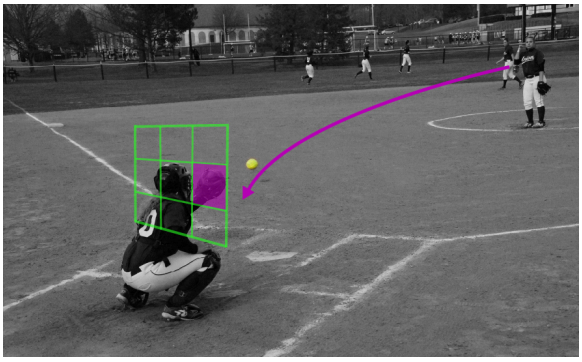

Figure 7: Curveball position in strike zone

Figures 6-8 illustrate the locations of the three different pitch types.

Audio data will replace the object tracking of the ball into the strike zone. The only position of the ball, that we can get the most information from, is the ending position of the ball in the strike zone. There is software that can record the audio data taken from a microphone and output it to a file. We are using the audio to identify the time when the ball is within the strike zone. When the ball reaches the strike zone, it makes a loud noise because of the catcher's glove (or in this experiment; a plywood board). From the file of

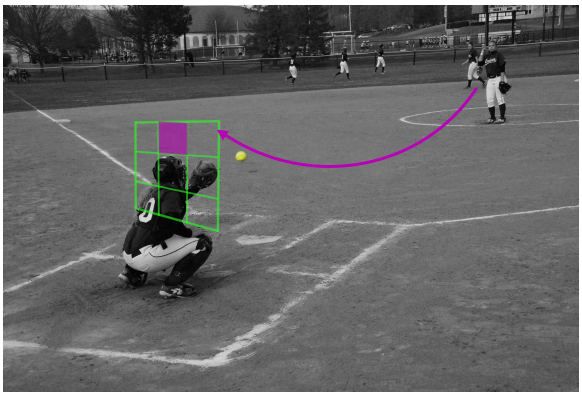

Figure 8: Riseball position in strike zone

audio data that is outputted, we can calculate the moment when the highest peak of sound occurred. We can then use the RGB camera to take a picture of this moment in time and evaluate the picture of the strike zone (to determine the location of the pitch).

8.1.1 Working with Audio Common

Audio common is a ROS package, used via the Robot Operating System, which allows us to record audio data to a file. Unfortunately, the Kinect sensor does not have reliable audio sensors, or a known method to extract the audio data from the microphone sensors on the Kinect itself. An external USB microphone had to be used in order to record usable audio data. The ROS node, audio common, allowed for the recording and playback of audio to a file. The audio data publishes to a separate file; which we can find the maximum level of volume and the corresponding timestamp at which the sound occurred. The peak sound occurs at the moment that the ball hits the plywood board; therefore if we want the RGB image, we pull the image from the file at the timestamp that the peak sound occurred.

8.1.2 Hough Transform Algorithms in OpenCV

Evaluating the image of the strike zone will allow us to determine the position of the ball within it. First we must find the edges of the strike zone grid and then find the 9 zones (as described previously) within the grid. This is one of finding if the pitch was in the correct zone for the pitch type it was previously

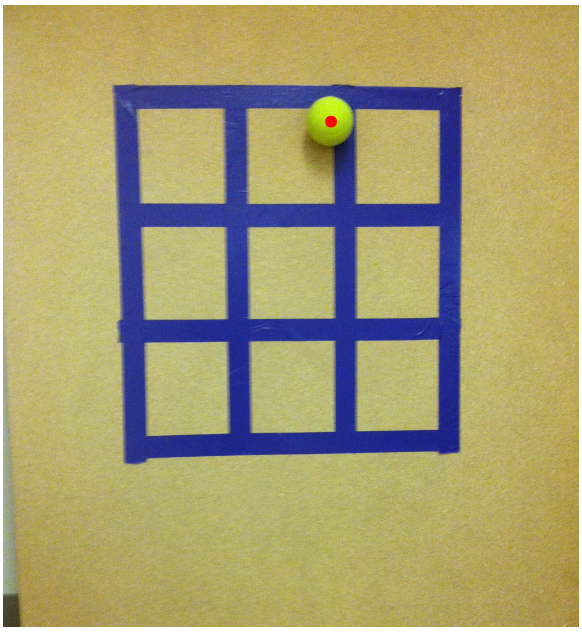

Figure 9: Image taken by Kinect

classifed for.

Hough Transform is feature extraction used in image analysis. These algorithms can be used on images with lines or edges and spherical objects. The edge detection and corner detection could be used to find the zones of the strike zone and the circle detection could find the ball within the strike zone. Therefore, the position of the ball within the grid (within one of the 9 zones of the grid) can be calculated.

Note: Implementation of these algorithms is included in Future Work section

9 Experiment Results

Initially, we worked on trying to classify pitching motions. A 10-fold cross validation breaks the training data into 10 sets of size $\frac{n}{10}$, trains on 9 datasets and tests on 1 dataset, then repeats 10 times and takes the mean accuracy. On a 10-fold cross validation, the CitationKNN algorithm has a 100 percent accuracy rate. This means that the classifier correctly classifies the different pitches 100 percent of the time. The CitationKNN algorithm returned 100 percent accuracy when all the attributes (xyz coordinates and Θ, Φ

Figure 10: USB microphone used to collect audio data [9]

pairs) were used and also when only the Θ and Φ values were used. The least amount of attributes used in the classification, the better. The system will run faster when there are fewer attributes.

9.1 Minimum Number of Instances Needed to Retain Accuracy

Hypothesis: Reducing the number of instances will improve the speed performance of the classifier Making the classifier preform more quickly was one of the main goals of classification. In order to reduce the time the classifier takes to classify the data, we must reduce the number of instances that the classifier has to classify. For example, if a 12,000-instance file takes 160 seconds to classify, then a 6,000-instance file should take about 80 seconds to classify. Even though we can make the classifier preform more quickly by reducing the number of instances, that doesn't necessarily mean that the accuracy of the classifier remains the same. Therefore the goal is to reduce the number of instances without reducing the accuracy of the classifier.

Hypothesis: The last couple seconds of a pitching motion is more distinguishable than the first couple seconds. The beginnings of different pitch types look the same because all pitch types require the basic mechanics of pitching. The last couple seconds of a pitching motion is when the pitcher puts the most force on the ball in order to make it move around the strike zone. The end of a pitcher's motion results in

the most physically distinctive posture. Therefore, the most important instances to analyze and put through the classifier are the last half/third of the instances that compose one pitch. (In the table below, removing from the last instances instead of first instances tests this hypothesis).

Below is a table of removing a certain number of instances and the results of the resulting accuracy:

Data	**Specifics**	**Instances**	**CitationKNN**
Data2 trained on Data1	3of240 instances removed	14220	100%
Data2 trained on Data1	5of240 instances removed	14100	100%
Data2 trained on Data1	15of240 instances removed	13800	100%
Data2 trained on Data1	25of240 instances removed	13500	100%
Data2 trained on Data1	50of240 instances removed	11400	100%
Data2 trained on Data1	80of240 instances removed	12000	100%
Data2 trained on Data1	140of240 instances removed	6000	93%
Data2 trained on Data1	100of240 instances removed	8400	100%
Data2 trained on Data1	120of240 instances removed	7200	98%
Data2 trained on Data1	110of240 instances removed	7800	98%
Data2 trained on Data1	105of240 instances removed	7800	100%
Data2 trained on Data1	first 105of240 instances	7800	83%
Data2 trained on Data1	first 150of240 instances	8940	96%

9.1.1 Conclusions

The minimum amount of instances able to retain 100% accuracy is 105 instances eliminated from each pitch. There are 240 total instances within each pitch, therefore $240 total - 105 removed = 135 instances$. An important finding to note is that the 105 instances were removed from the beginning of each pitch. When the first 150 instances of a pitch are used, the accuracy drops because the first instances do not have the most visually distinctive body posture. The beginning of pitches look the same because (as stated in the Introduction) all the different pitch types are based off of a pitcher's fastball. The body posture at the end of the pitch is what makes each pitch unique. If we are able to capture the end of a pitch in it's different

instances, and classify the pitches on this information, then we can classify the pitch type.

9.2 Eliminating Joints

Eliminating the number of joints will reduce the amount of data we have for each instance of a pitch. This again deals with the speed of the classifier. We want to use as few instances and as few joints as possible to run through the classifier without losing it's accuracy.

Hypothesis: The Theta and Phi values are just as important as the three-dimensional joint data. When using data recorded from two different pitchers, finding the angles of the joints becomes more important. The joints relation to each other will be quite similar between two different pitchers because elite pitchers have the same, well tuned, pitching mechanics. Even though the pitcher's arms might end up in two different planes according to the three-dimensional data, the angle between the joints remains consistent between the two.

Below is a table of testing the hypothesis by removing joints in order to determine which joints or angles are the most important:

Data	**Joints Removed**	**CitationKNN**	**Deduction**
Data2 trained on Data1	head	100%	not important
Data2 trained on Data1	head, neck	100%	not important
Data2 trained on Data1	head, neck, torso	100%	not important
Data2 trained on Data1	head, neck, torso, right_shoulder	100%	not important
Data2 trained on Data1	all xyz (only Φ Θ)	100%	Φ Θ important
Data2 trained on Data1	all xyz & shoulder Φ Θ	98%	Φ Θ important
Data2 trained on Data1	all xyz & torso Φ Θ	98%	Φ Θ important
Data2 trained on Data1	all xyz & elbow Φ Θ	98%	Φ Θ important
Data2 trained on Data1	all xyz & knee Φ Θ	98%	Φ Θ important

9.2.1 Conclusions

The four Φ Θ pairs are the most important; which confirms the hypothesis of relational space. The actual coordinates of the joints aren't as important as the angles of the joints. The four joints that retain 100% accuracy are right_shoulder Φ Θ, torso Φ Θ, left_knee Φ Θ, and right_elbow Φ Θ. These tests were executed on the 135 instances of each pitch.

10 Result Conclusions

Pitch types can be classified by focusing on body posture. Using 7800 instances (right_shoulder Φ Θ, torso Φ Θ, left_knee Φ Θ, and right_elbow Φ Θ) the dynamic gesture of a pitching motion (focusing on pitch types: curveball, rise ball, and change-up/off-speed) can be classified using the CitationKNN multi-instance classifier with 100% accuracy. All of the hypotheses made were experimented on and affirmed.

- Reducing the number of instances improved the speed performance of the classifier. The classifier initially finished classifying the data in about 4 minutes. When the instances were reduced to a little more than half of the original number of instances, the classifier finished in about 2 minutes. Therefore, reducing the number of instances reduced the amount of time it took for the classifier to finish classifying the data.

- The last couple seconds of a pitching motion are more distinguishable than the first couple seconds. The instances that were removed from the end of the pitch didn't have as high of an accuracy as the instances that were removed from the beginning. Therefore, the last couple seconds of a pitching motion is more distinguishable than the beginning seconds of a pitching motion.

- The Theta and Phi values are just as important as the three-dimensional joint data. The way the joints relate to each other became more important than how the joints related in three-dimensional space. By using the Phi and Theta values, we were able to classify multiple elite pitcher's body postures.

11 Future Work

The future work of this thesis includes the implementation of the Hough Transform Algorithms in order to find the location of the ball within the strike zone. This information would help, in conjunction with the results gotten from posture analysis, confirm the pitch type being thrown. Evaluating the speed of the ball, through object tracking over time, would also aid in the confirmation of an off-speed pitch (an off-speed pitch being slower than the other pitch types). Essentially, turning the Kinect into a radar gun is another step that could be taken to enhance this project. Finally, a real-time pitch evaluation could be implemented for pitcher's that are truly seeking a training aid to use.

References

[1] Garrett Bernstein, Dan Gallagher, and Nyk Lotocky. Robot recognition of military gestures. 2009.

[2] Eibe Frank, Mark Hall, Geoffrey Holmes, Richard Kirkby, Bernhard Pfahringer, IanH. Witten, and Len Trigg. Weka. In Oded Maimon and Lior Rokach, editors, *Data Mining and Knowledge Discovery Handbook*, pages 1305–1314. Springer US, 2005.

[3] Christopher Grant. 'kinect for windows' sdk coming this spring from microsoft @ONLINE, February 2011.

[4] K. Kato, Y. Namiki, N. Thepvilojanapong, N. Tetsutani, and Y. Ohta. Exploring analysis of football actions considering correlation of joints in the body. In *Networked Sensing Systems (INSS), 2012 Ninth International Conference on*, pages 1–2, 2012.

[5] Geoffrey Holmes Bernhard Pfahringer Peter Reutemann Ian H. Witten Mark Hall, Eibe Frank. The weka data mining software: An update. 2009.

[6] S. Monir, S. Rubya, and H.S. Ferdous. Rotation and scale invariant posture recognition using microsoft kinect skeletal tracking feature. In *Intelligent Systems Design and Applications (ISDA), 2012 12th International Conference on*, pages 404–409, 2012.

[7] Mrutyunjaya Panda and Manas Ranjan Patra. A comparative study of data mining algorithms for network intrusion detection. In *Emerging Trends in Engineering and Technology, 2008. ICETET'08. First International Conference on*, pages 504–507. IEEE, 2008.

[8] Morgan Quigley, Ken Conley, Brian P. Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. Ros: an open-source robot operating system. 2009.

[9] Apple Store. Apple store snowball microphone, March 2014.

[10] Jun Wang, Zucker, and Jean-Daniel. Solving multiple-instance problem: A lazy learning approach. In Pat Langley, editor, *17th International Conference on Machine Learning*, pages 1119–1125, 2000.

[11] OpenNI Source Website. Openni: The framework for 3d sensing, March 2014.

[12] Ian H. Witten, Eibe Frank, and Mark A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, Amsterdam, 3 edition, 2011.

[13] Harry Zhang. The optimality of naive bayes. *A A*, 1(2):3, 2004.