MobPitcher

A Mobile Sensor Based Throwing Speed Estimator

Final project for CICS 590U Spring 2019 Ubiquitous Computing
University of Massachusetts Amherst

Catherine Huang, Yen-Sung Chen

Abstract

The paper presents the design, building and testing of MobPitcher, an application aimed utilizing mobile sensors on smartphones to estimate the speed of pitching a ball into categories. With the availability of smartphone and other mobile devices such as smartwatch, most people have access to one and is using the sensors that comes with the device in daily activities. Unlike the common way of on mobile phone in using phone camera to capture images of the thrown object to estimate the speed, which has limitation in its setting, or purchasing radar speed gun that requires a separate equipment and may cost more than desired. We aimed to only utilize those sensors equip in most mobile devices, and with the help of machine learning trained model to come up with a solution to allow people to estimate the speed of how fast they throw a ball in any situation.

1. Introduction

We propose a method using mobile sensors to help user to know how fast they throw the ball. This is an improvement from the current method to capture speed of pitching because current method involve using speed gun, which may be expansive[, has limit such as range of detection, and need to setup the device to capture the speed. We found that previous works has been done on using inertial sensors to sense sporting events, However, to the best of our knowledge, the sensors were only for gesture recognition[1,2], for example, in being used to detect activity and postures such as whether or not the wearer is throwing a ball. Our design is to use the sensor to estimate the pitching speed, which will be a good replacement for a speed gun, considering the price and accessibility of the device, everyone with a smartphone will be able to use the app and keeps track of their throwing speed.

2. Experiments

In this section, we will discuss how we designed the usage of the device in order to collect data for training the model that will estimate the speed. As well as how the environment, user test cases and validation method were set up, alone with some statistic about our participants pool.

Equipments & Environment Setup

In order to collect the data generated from throwing actions, we utilize the accelerometer, gyroscope, and gravity sensors of smartphones. A Samsung Galaxy 7 is used by wrapping it on testers' arm with an armband. Then for the test environment setup. We decided to use

have the user stand close in front of a long wall, where we put markers on the wall to aid in calculating the validation/ground truth data. The distance between each marker is 3.3 feet apart from the right side of the marker to the right side of another marker, with the longest marked distance being 13 feet 2 inches long. An empty box is placed 15 feets away from the participants as a target to be thrown at. The extra distance is to prevent blocking the marker or the ball (Figure 2).

After setting up the marker on the wall and the target, a camera is setup opposite to the wall to capture the ball being thrown. The slow motion function with 240 fps of IPhone SE camera is used for tracking the route of ball for better calculating the velocity. The ball used is a bright green juggling ball about the size of a tennis ball. The color stands out and because we're conducting the experiment indoor, having a softer ball can lessen the probability of damaging the property or people.

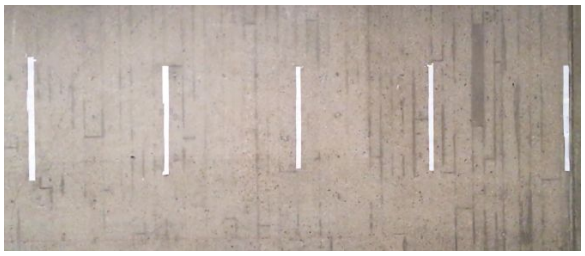

Figure 1: shows the marks on the wall and a frame of the video capture by the camera.

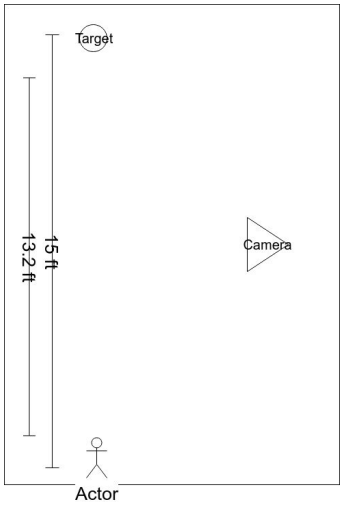

Figure 2: shows the setup of the testing environment.

User Tests & Data Collection

We had created an application on the phone for collecting the throwing data. Upon opening the app, the user will be prompt to enter information such as: ID(name, mostly initials), height (feet or meters), weight (pounds or kilograms). Figure 3 shows the user interface of the application. Then we separately recorded the gender, age and the dominant hand of the participant.

As mentioned, the application helps in collecting the throwing data, which upon pressing the start button, sensor data from Gyroscope and Accelerometer are collected in milliseconds and saved as text files. From the sensors, we focus on collecting data on linear acceleration, angular acceleration, gravitation, and pressure.

For the user tests, we asked the participants to strap the phone with the smartphone armband on their lower arm of their dominant arm, close to the wrist. Then we asked each participants to throw the ball at the target in 3 different poses (Above the head, Side way, and Submarine) with

various strength (normal, fastball, and random) in order to capture the ball moving in different velocity.

The participant is asked to first only use above the head pose to throw the ball for 9 times. The first 3 throws, they'll throw using normal strength, the next 3 throws will be using maximum strength, and the last 3 throws will be random strength determined by the participant. Then, the participant is asked to only throw the ball 9 more times from the side of their body, with the first 3 throws being normal strength, next 3 throws being the maximum strength, and the last 3 throws being randomly strength picked by the participant. The same process is repeated for the submarine pose. So, each participant ended up performing 27 throws.

A total of 243 throws were collected in two days time. During the data collection, participant can take breaks in between throws or freely moves their arm around. Because each throw is recorded separately. We asked the participant to click the start button right before throwing the ball, and then press stop button right after the throw.

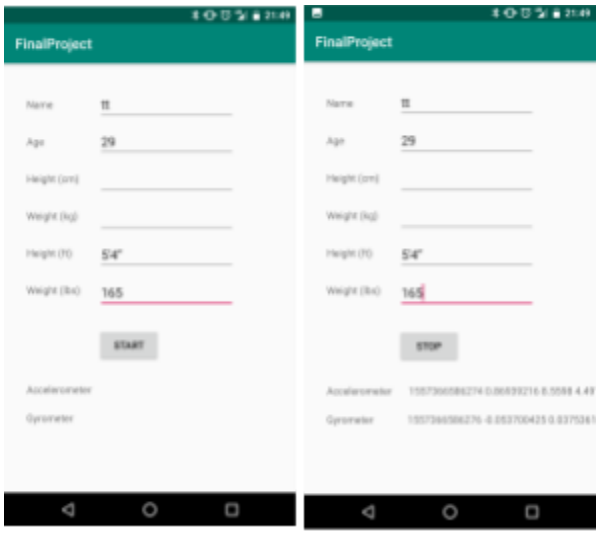

Figure 3: the user interface of the data collection application

Participant demographic

In total, we had collected data from 9 participants. The demographic of the participants is in the following table:

Gender	8 males	1 female
Age	Youngest: 22	Oldest: 29
Weight(lb)	Low: 132.2	High 176.3
Height(ft.)	Low: 5.4	high:5.9
Dominant Hand	Left: 1	Right: 8

Validation

For validation, we takes the slow motion videos capture by the IPhone SE, looks at the frame by frame changes of the ball to calculate the velocity of the ball. First the video is capture in 240fps so it contains a lot finer detail than using regular 60fps. Since we have setup markers along the wall with a known distance between each marker and the entire distance, this is used to calculate the change in distance of the ball by converting the pixel distance to real distance.

To get the location of the ball, Several image processing techniques are applied on slow motion videos to get the ground truth speeds. For each frame in video, two lines are drawn that match exactly the distance we've set in reality. Color filters, Gaussian filter, erosion and dilation are applied to extract the green ball on each frame. The real velocity of ball can be acquired by having the real distance divided by the elapsed time that green ball travels from one line to the other line.

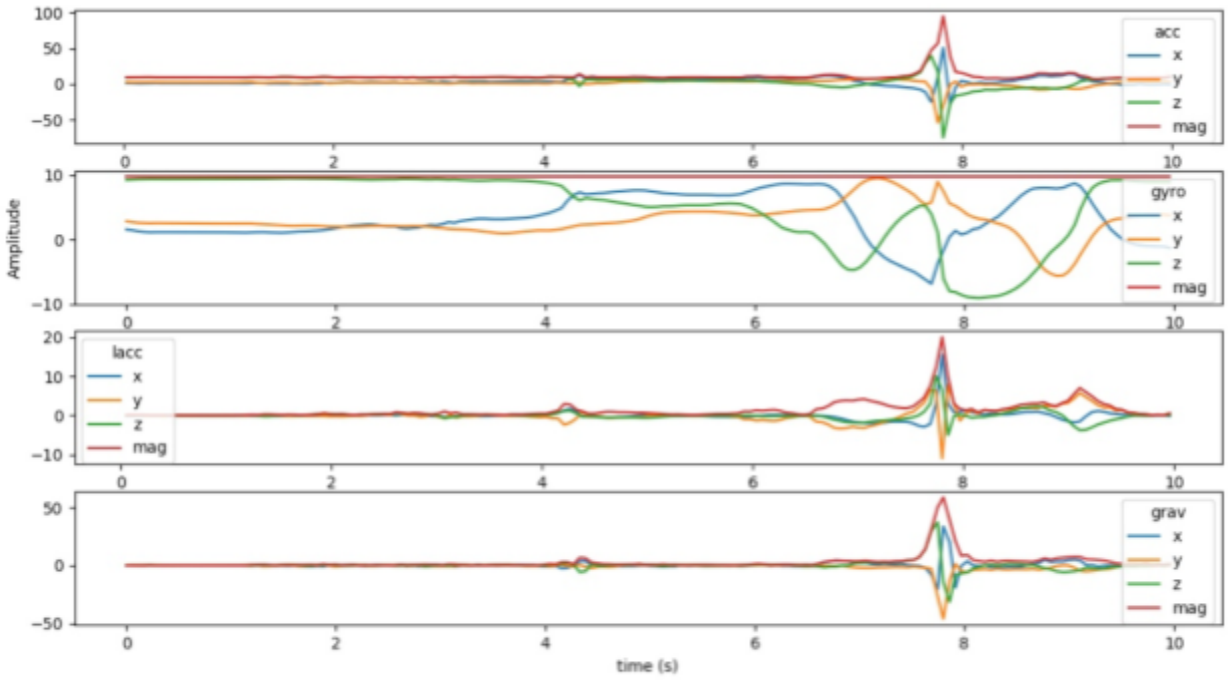

Figure 4: Raw signal data from sensors. From top to bottom: accelerometer, gyroscope, linear acceleration, gravity.

3. Dataset

A sample of throwing action is shown as Figure 4. From the figure, we can see that the signal shape of linear acceleration sensor and gravity sensor is similar to accelerometer. Therefore, only data of accelerometer and gyroscope are used for model training. For feature extraction, we mainly calculate time domain features listed as follow: mean, standard deviation, variance, mean absolute deviation (MAD), skew, Kurtosis, maximum, minimum and range. The sensor data as shown in Figure 4 doesn't have obvious routine waveform. Also, peaks in each signal locate differently in a time segment, which will result in different consequence in frequency domain. We doubt that extracting features in frequency domain have limited effects for training our model. As a result, the time domain features acquired from three different axes and magnitudes are utilized for model training.

4. Approach and Results

Our goal is to predict the velocity of object based on user's throwing actions. The regression method is used for training the speed prediction model. The dataset is first split into training set and validation set. KFold is applied on training set for cross validation and a model is measured with mean absolute error (MAE). We use decision tree regressor, k-nearest neighbor regressor and ridge regressor with different parameter settings. The decision tree regressor with maximum depth = 5 gives the lowest MAE. The result is shown as following table:

Regressor		MAE
Decision Tree	depth=3	6.0350
	depth=5	5.3950
	depth=7	6.7666
K-nearest neighbor	k=3	6.5837
	k=5	7.9736
	k=10	6.6385
Ridge	alpha=0.1	8.8827
	alpha=1	6.5606
	alpha=10	6.9065

5. Discussion and Improvements

Although the decision tree regressor with maximum depth = 5 gives the lowest MAE, there are several ways that can improve our results:

1. We acquired the ground truth speed by processing videos. However, we suffered from environmental limitation that we were bothered several times during recording ball throws, which forced us to move camera without pausing. The condition affects the accuracy of calculating the real velocity on certain frames.
2. The image frame on camera could suffer from image distortion or not capturing the ball within the frame, which may not give an accurate result for calculation. It would be better if we can use a speed gun for detecting object speed as second ground truth.
3. Not having a permanent experiment environment setup to collect all user data. This made the processing of ground truth data a lot more difficult because it relies on video data where the lighting of the different setting is different, and the space of the environment is different. Which prone to more error.
4. Due to the fact that most participant doesn't have experience with throwing the ball and aiming at the target, there are some throws among that ended up completely out of the frame of the camera. Resulting in lack of ground truth data for some throws.
5. Though we had 9 testers that each gave 27 throws, the usable data after raw data processing left not much. We found cases such as the route was not entirely captured by camera, which was counted as corrupted. It would be better if we can collect more throws and guarantee that the whole throwing process is recorded.
6. We only use basic time domain features such as mean, variance, skew, etc. It would be better if we can try more features.

6. Conclusion

Our work shows that velocity estimation of throwing objects is feasible. By using decision tree regression with maximum depth = 5, the model gives the lowest MAE. We believe the result can be further improved by addressing issues mentioned in Section 5. For future work, we plan to build an phone application with the model installed that can provide users with a

cheap solution in estimating throwing speed.

To access the data and code used in this project, please visit: https://drive.google.com/drive/folders/1C7knwWnLzyZ1WitEuTR_taHDhGfDjHPs?usp=sharing

Reference

1. Murray, Nick & Black, Georgia & Whiteley, Rod & Gahan, Peter & Cole, Michael & Utting, Andy & Gabbett, Tim. (2016). Automatic Detection of Pitching and Throwing Events in Baseball With Inertial Measurement Sensors. International Journal of Sports Physiology and Performance. 12. 1-18. 10.1123/ijspp.2016-0212.
2. Andrew, et al. “Detection of Throwing in Cricket Using Wearable Sensors.” *Sports Technology*, Routledge, 3 May 2017, research-repository.griffith.edu.au/handle/10072/48687.
3. Ball tracking: https://www.pyimagesearch.com/2015/09/14/ball-tracking-with-opencv/