Two Dimensional Basketball Localization and Location Prediction

Matthew Wilson
mbowiewi@stanford.edu

Jerry Giese
jerry.giese@stanford.edu

***Abstract*—In two dimensions, we accurately predicted the trajectory of a basketball using approximately ten images each taken less taken less than three tenths of a second after a shooter a free throw. Our trajectory predictions were made using automatically stabilized handheld video footage of a shooter releasing a free throw. Although we used only the first few images of the shooter releasing the basketball to make our path predictions, we tracked the basketball in each image of the shot in order to compare the predicted trajectory of the ball to the true trajectory of the ball. Extending our analysis to three dimensions would enable a viewer to reliably predict the outcome of a free throw using information about the basketball's location from only the first third of the balls path to the hoop.**

I. Introduction

Given a handful of images of a basketball player releasing a free throw, one may have sufficient information to determine the result of the shot. In this project we attempt to accurately predict the path of a basketball shot using as little information about the shot as possible. For simplicity we assume the camera is aligned perpendicularly to the ball's plane of motion. Under this assumption, the ball's path is a parabola (rather than a perspectively distorted parabola) and can be described by two coordinates. In two dimensions, we were able to predict the trajectory of the ball with a high degree of accuracy using only a small set of images taken only tenths of a second after a shooter releases the basketball. Although this project only considers two dimensions, many of the techniques used in this project may be generalized to three dimensions as would be necessary for a prediction of the outcome a real shot.

Our project is interesting because it can be thought of as a prototype for a shot-prediction system, which could be used during broadcasts of basketball games or other sporting events. More generally, our project could be adapted for any application in which one is interested in tracking an object to predict its motion.

From an image processing perspective, this project required us to determine the position of the basketball with a high degree of accuracy. To accomplish this task we had to stabilize handheld video footage of the shooter, and develop an algorithm to track the motion of the basketball. Once the basketball was localized in space, we predicted its path using Newton's laws of motion, and total least squares. Videos of the shooter were taken using the camera of an iPhone 5S.

TABLE I
Camera specifications [3]

Resolution	1920x1080 pixels
Frame rate	30 FPS
Color channels	3 (RGB)
Aperture	F2.2

II. Methods

Overview: The main components of the image processing algorithm are shown in figure 1. First, image stabilization is performed to remove the effects of camera shake. This was done by using cross correlation, finding the correlation peaks, and using the peaks to determine image frame offsets. Then, the ball center is tracked throughout the video frames. Ball tracking relied on the Hough transform. Last, the tracked ball center coordinates are used to predict the trajectory of the ball. For this step, we fit a parabolic arc using a total least squares fit.

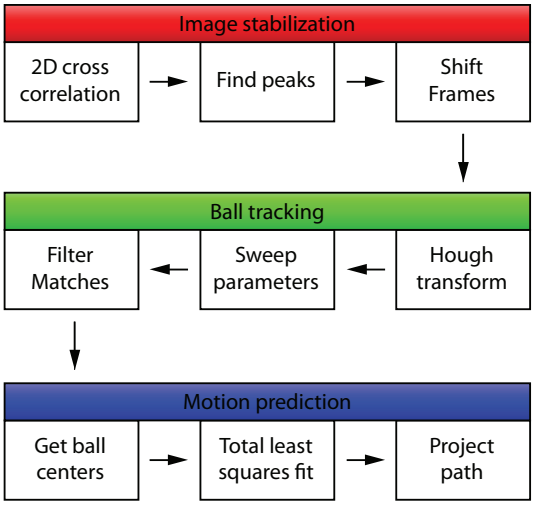

Fig. 1. An overview of our path prediciton method.

Equipment: To record the video, we used the front camera of an iPhone 5s. The camera's video capture specifications are listed in table I.

Image Stabilization: Because cameras are generally not stationary, we decided to hold the camera rather than rigidly mount it to a tripod. This resulted in videos with a slight

amount of camera shake. We observed a frame alignment variability of around 2-3 pixels between frames. Because the object we are tracking has a diameter on the order of 30-40 pixels, this camera shake would result in significant path projection errors. In order to combat this problem, we decided to first perform image stabilization.

The image stabilization algorithm used in this project utilizes 2D cross correlation. This method is computationally fast, but does not account for camera rotation. We observed that most of the instability in the video was due to translation and not rotation, therefor we predicted that translational frame alignment would be reasonably accurate for this project while maintaining fast computation.

To further increase computation speed, we only consider small windows within the image to calculate the cross correlation as shown by the blue boxes in figure 2.

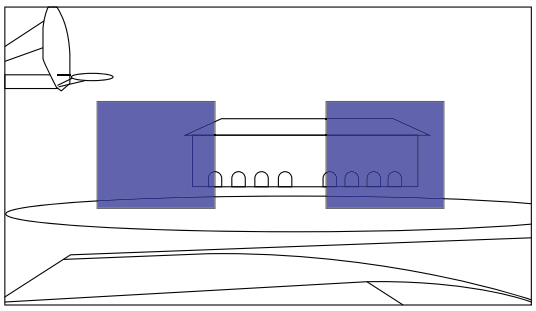

Fig. 2. Subspaces of the frame used for cross correlation.

The cross correlation of two windows are used in the determination of frame translation. This was done to prevent frame tracking glitches caused by objects moving through a window of consideration. The window resulting in the highest correlation peak is used as the valid tracking frame.

To minimize computation time, we only perform cross correlation over a small range in which we assume a maximum frame to frame shift distance. But, in order to prevent cumulative tracking error which could be caused by correlated one frame with the previous frame, all tracking is done with reference to the initial video frame. Because frame drift generally increases over time, we would normally have to increase the convolution distance to match the image drift over the span of the entire video. To counter this problem, we enabled the tracking windows to follow the tracked images as shown in figure 3. This allowed us to calculate the cross correlation over small shifts while being able to track frames over an arbitrary distance.

Ball Tracking: After the handheld video was stabilized we found the position (in pixels) of the basketball in each frame of the free throw footage. We found the ball in all frames of the footage (rather than just the first few) because we wanted to map out the measured trajectory of the ball in order to compare it to our predicted trajectory.

In order to track the position of the basketball, we experimented with two different tracking algorithms. Both tracking algorithms relied on imfindcircles, Matlab's function to find

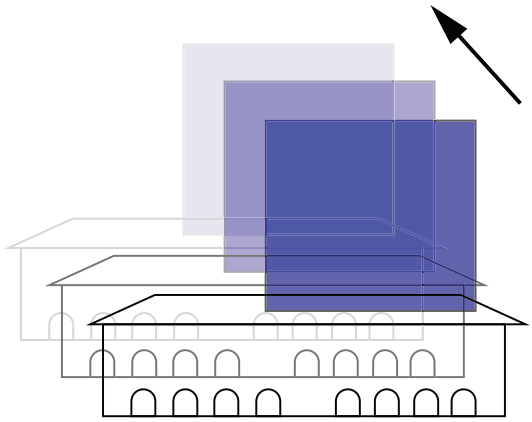

Fig. 3. Illustration of the tracking subspace frame drift.

circles in an image using the Hough transform. The first of the two algorithms, despite its simplicity performed surprisingly well. After estimating the minimum and maximum ball radius in pixels, we simply called imfindcircles with an appropriate value of the sensitivity parameter found by trial and error. In a well-lit scene with minimal background clutter, this approach located the ball with surprising reliability. However, this simplistic approach failed when tracking conditions were less than ideal.

The second tracking algorithm we used also relied on Matlab's circle finding method, but did so in a more sophisticated way. The second tracking algorithm only searched a small patch of the image known to contain the ball. The sensitivity parameter passed to Matlab's circle finding method was iteratively increased from a relatively low value until the method found a circular object in the image patch known to contain the ball. If two circular objects were found in the same small patch of the image, then the object closest to the ball's previous location was assumed to be the ball. To initialize the algorithm, the location of the ball in the first frame after the shot was released had to be manually input. These changes greatly increased the robustness of the tracking algorithm to background clutter, and decreased tracking computation time.

Trajectory Calculation: After we have determined the position of the ball in at least three frames, we may fit a parabola to the observed path. When calculating the predicted trajectory of the basketball, it is important to note that our tracking algorithms return position estimates which are equally likely to contain error in the horizontal and vertical position. For this reason, the standard least squares fit of a parabola to the data is not appropriate [2]. Instead, we use total least squares to fit a parabola to the data. Also, notice that because our tracking algorithm finds the position of the ball in each video frame, we may easily adjust the number of points used to fit the parabola in order to compare the trajectories predicted by different numbers of ball location measurements.

III. Results

In two dimensions, we predicted the trajectory of a basketball using the first three to fifteen frames of a video showing

a shooter releasing the ball. Figure 4 shows two sequences of images illustrating our trajectory prediction algorithm by comparing our predicted trajectories to the observed ball locations. The first pair of images in the sequence shows our prediction made using three frames of video. Each successive pair of images in the sequence shows our trajectory prediction made using two more video frames than the previous image.

As the figure shows, our prediction improved as we considered more video frames. The prediction using only three frames of video, the minimum amount needed to fit a parabola to the data, was very unreliable. That said, relatively accurate predictions can be made with as few as five frames of video. In order to predict the trajectory of the ball with sufficient accuracy to determine the outcome of the shot, we found that we needed approximately ten frames of video, which corresponds to three tenths of a second. During a typical basketball free throw, the ball takes approximately one second from the time the ball leaves the shooter's hand to the time it arrives at the hoop. Therefore, using our method, we found that the two-dimensional trajectory of a basketball can be predicted well enough to determine the outcome of the shot using position measurements taken during the first third of the shot's journey toward to hoop.

IV. Discussion

Shot Outcome Prediction: In this analysis, we did not use the predicted path of the ball to predict the outcome of the shot due to time constraints. Even in two dimensions, predicting the outcome of a shot given the hoop location and a predicted trajectory is a more challenging problem than it may seem at first glance. It is not simply a question of whether or not the parabolic path of the ball comes within a certain tolerance of the center of the hoop. For example, paths with very little arc may intersect the center of the hoop, but will often result in a miss because the ball will not clear the front of the rim. For the shot to go in, the ball must clear the front of the rim, and fall into the hoop before reaching the back of the rim. Furthermore, to accurately predict the outcome of shots whose path intersections the hoop, some sort of statistical model (requiring many shots of training data) would likely be required.

Extension to Three Dimensions: Although we considered only two dimensions in this project, our analysis can be extended to three dimensions in different two ways. The first is to use a second camera angle. The most simplistic two camera setup uses one camera aligned (approximately) perpendicularly to the ball's plane of motion, and one camera placed behind the shooter aligned parallel to the ball's plane of motion. With this setup, a shot is classified as a make if and only if the shot could have gone in according to both camera angles. We had hoped to perform this particular three-dimensional analysis, but could not due to time constraints.

Although two cameras could certainly be used to extend our analysis to three dimensions, a three dimensional analysis is theoretically possible using only one camera angle. Such an analysis would consider perspectively distorted parabolas when fitting the path of the ball [1]. This would require more fit parameters, which require more ball position data to accurately determine.

Poor Filming Conditions: Our ball tracking algorithm is quite robust to background clutter as can be seen in figure 4. As the figure shows, our tracking algorithm performs well with trees, buildings, and other clutter in the background. Although the tracking algorithm still performs well with a cluttered background, the amount of time needed to find the ball increases. This is because more iterations are needed before the sensitivity parameter of Matlab's circle finding method reaches the point that a circle can be distinguished from the background.

Although our two dimensional shot path prediction system is fairly robust to background clutter, it does not perform well in low light conditions. This is because the frames of video taken by the iPhone 5S camera showed a blurred basketball when the lighting was poor. A blurred basketball is much harder to detect because it is less circular. Furthermore, a blurred basketball is almost impossible to localize sufficiently to extrapolate its path based on only a few points.

Color Channel Isolation: In an attempt to increase the effectiveness of the ball finding algorithm, we attempted to implement a preprocessing step to increase the contrast of the ball using information about its color. To do this, we took the red channel magnitude and divided it by the sum of the magnitudes of all three color channels. A normalized example is shown in figure 5. This step does make the ball more visually recognizable. One issue with this method is that a significant amount of noise can be introduced in darker areas, such as in the trees, because of the division by small pixel intensities. This creates a lot of edge artifacts that causes issues in the Hough circular edge finder algorithm. This resulted in ball radii and center locations having small errors which significantly reduced the path projection accuracy.

Fig. 5. Enhancement of ball contrast using color channel manipulation

Acknowledgments

We would like to thank Hany Farid, Bernd Girod, and David Chen for their guidance during this project.

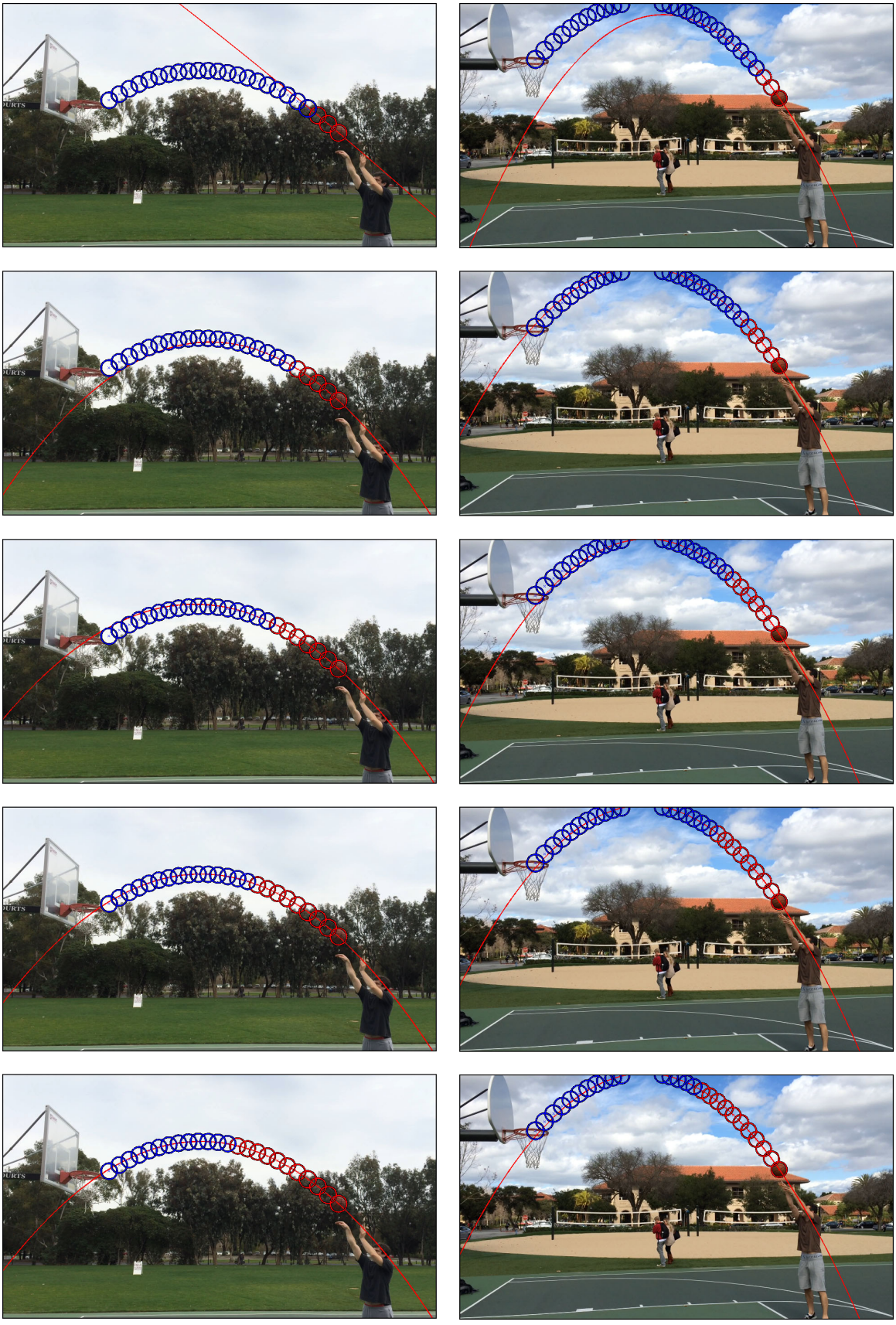

Fig. 4. Two sequences of images comparing the observed and predicted trajectory of a basketball. Each of the two sequences of images shows the same observed ball trajectory along with predicted trajectories made using different numbers of video frames (3,5,7,9,11, respectively). The red circles are the observed ball locations used to make the location prediction, and the blue circles are the ball locations observed after the prediction was made. The location prediction is the parabola shown in red. The shot on the left hit the front of the rim, while the shot on the right went in.

REFERENCES

[1] Valentina Conotter, James F. O'Brien, and Hany Farid. "Exposing Digital Forgeries in Ballistic Motion". IEEE Transactions on Information Forensics and Security, 7(1):283 296, February 2012.

[2] I. Markovsky and S. Van Huffel, Overview of total least squares methods. Signal Processing, vol. 87, pp. 2283-2302, 2007.

[3] P. M. Ferenczi, Review: How does Apple's new iPhone 5s perform as a camera?, March 2012, http://connect.dpreview.com/post/7518611407/apple-iphone5s-smartphone-camera-review.

APPENDIX

TABLE II
MEMBER TASKS

Task	Member(s)
Record videos	Jerry and Matt
Frame tracking algorithm	Jerry
Ball tracking algorithm	Matt
Ball path estimation	Matt
Color channel enhancement function	Jerry
Make poster	Jerry and Matt
Write report	Jerry and Matt