Sport Analytics

Justin Schubeck, Alex Liu, Erman Nghonda, Christophe Bobda and Jack Judy
Department of Electrical and Computer Engineering,
University of Florida, Gainesville, 32611, FL, USA
jschubeck@ufl.edu, aliu1@ufl.edu, enghonda@ufl.edu, cbobda@ece.ufl.edu, jack.judy@ufl.edu

***Abstract*—In this paper we present an end to end method for calibrating cameras, detecting a basketball's position, and projecting multiple-camera data into a 3D space for analysis. Information, in the form of 3D coordinates, enables the user to analyze the metrics of position, speed, and trajectory of a basketball free throw. Tracking basketball movement in video frames is of high importance to elite athletes when analyzing and extracting information about performance. Having a system to perform shot analysis could be embedded into training or expanded into a prediction and feedback system for tracking progress.**

***Index Terms*—calibration, detection, triangulation**

I. Introduction

Sport analytics using computer vision could fill the crucial gap of intuitive film review in any sport. In basketball, computer vision and advancing technology can allow free throw trajectories to be tracked and analyzed with a much higher level of autonomy than previously possible. This method of free throw analysis could allow players to master shot trajectories with far less time and effort involved. Computer vision eliminates the need for team trainers or players to manually review film of each free throw countless times to determine its trajectory and accuracy. This innovation could replace a process that consumes hours of valuable training time.

Using cameras, a computer, and computer vision software, it is possible to analyze a free throw with a high level of autonomy in the matter of minutes. The resulting analysis can be presented through plots and statistics such as velocity or height. The main objectives of sport analytics using computer vision are: multi-angle data collection, ball detection/tracking, and 3D triangulation. Multi-angle data collection involves using cameras placed at different locations around the basketball court to capture the same shot from different angles. Ball detection locates the ball in each camera's video, producing the coordinates for the center of the ball. The background environment of each camera angle varied, so multiple detection methods were incorporated. Both of the first two objectives aid the third, 3D triangulation, which introduces a high level of accuracy to the detected ball location and the ability to model free throw trajectory in 3D space. 3D triangulation of the basketball location, though more complex, proves far more accurate and less susceptible to angular distortion than analysis in 2D.

In the following sections, the methods used to properly calibrate the cameras, detect the ball, and triangulate the ball location will be discussed. Additionally, description and evaluation of the results of the free throw analysis will be given.

II. Methods

The following pipeline seen in Fig. 1 overviews an implementation to detect, track, and project a basketball's 3D position during a free throw.

A. Data Collection

The data set consisted of six total free throws on a standard outdoor basketball court. Three of those free throws were successful, and three free throws were unsuccessful. All cameras were recording videos on a tripod at 66 inches high and were never moved at any point during the data collection stage. Three cameras were used: the iPhone Xs on the left angle recording on 1080p at 30fps, the iPhone X on the center angle recording on 4K at 30fps, and the iPhone 7s Plus on the right angle recording on 4K at 30fps. The videos were then aligned in time and cropped to keep only relevant information. The player and ball were not cut out of the frame at any time.

B. Camera Calibration

Camera calibration consists of using a known 3D coordinate system and corresponding 2D image coordinates to produce an intrinsic camera matrix and distortion coefficients for each camera used to capture the data set. The 3D coordinate system used is a representative mapping from dimensions in the real world to a user-defined coordinate system. 2D image coordinates, on the other hand, are absolute pixel coordinates corresponding to the various 3D real world coordinates as seen by the camera. The user defined 3D coordinate system followed a 100 units to 70 inches ratio and can be seen in Fig. 2. The corresponding 2D labels for one camera angle can be seen in Fig. 3.

A camera intrinsic matrix allows 3D coordinates to be transformed into 2D coordinates on an image. This matrix can be constructed by obtaining the focal lengths in each direction,(f_x, f_y), and the image centers,(c_x, c_y). An intrinsic camera matrix is then defined as: [1]

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

Distortion coefficients can also be extracted for each camera to correct any radial, k_n, or tangential, p_n, distortion in

Camera Calibration									
Camera	f_x	f_y	c_x	c_y	k_1	k_2	p_1	p_2	k_3
iPhone Xs	874.2	871.2	464.1	284.7	0.340	-2.212	0.011	-0.008	7.100
iPhone X	834.0	830.8	483.6	270.5	0.272	-0.904	0.003	0.006	0.429
iPhone 7s Plus	849.3	847.6	483.9	268.8	0.349	-3.001	0.005	0.000	10.718

TABLE I
CALIBRATION VALUES

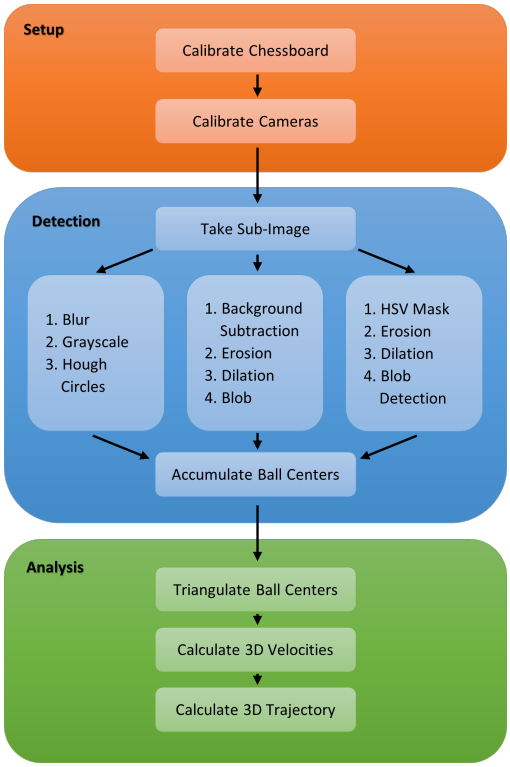

Fig. 1. Pipeline

an image. It is common to obtain camera intrinsic matrices and distortion coefficients using chessboard calibration, as it provides a consistent grid pattern with known dimensions. The

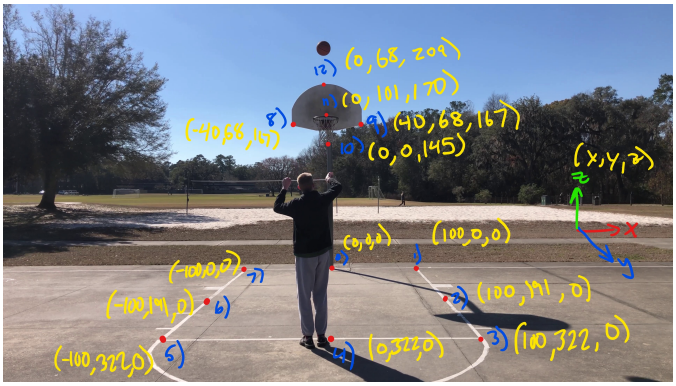

Fig. 2. User-defined 3D coordinate system

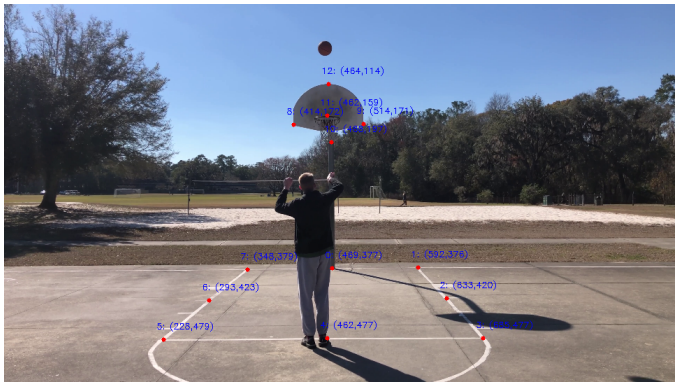

Fig. 3. 2D pixel values corresponding to 3D points

distortion coefficients were defined as: [1]

$$d = \begin{bmatrix} k_1 & k_2 & p_1 & p_2 & k_3 \end{bmatrix} \tag{2}$$

To begin calibration, each camera recorded a video of the chessboard on the same video settings used to record the data set. Ten unique frames were then taken from each video to use as inputs for the camera's calibration. A result of this calibration can be seen in Fig. 4. Outputs of the chessboard calibration include the intrinsic camera matrix and distortion coefficients. With the intrinsic parameters, 3D coordinate points, and 2D pixel coordinates, the rotation matrix, R, and translation vector, t, can be found for each camera [4]. These values will be used later in the pipeline for triangulation.

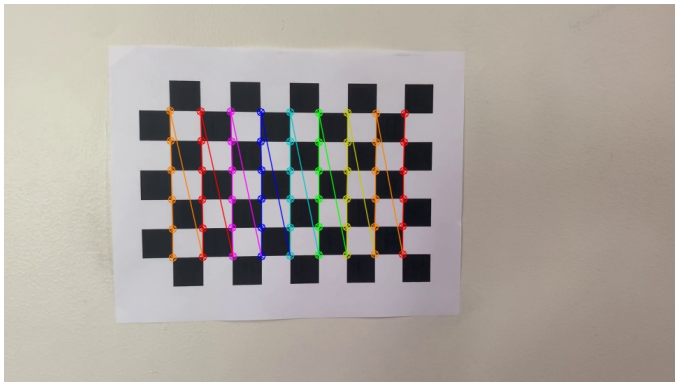

Fig. 4. Instance of chessboard calibration input image

C. Ball Detection

For detection, the input video was processed frame by frame. Each frame was resized to 960 by 540 pixels to reduce

the dimensionality before performing unique preprocessing steps for each detection method. Detection was performed on a 100 by 100 pixel subset of the original frame to reduce error and remove unnecessary parts of the image. This bound was then updated to be centered around the middle of the detected circle in order to ensure the ball would be visible in the next sub-image. If multiple circular objects were detected, the bound would center around the object closest to the ball's previous position. If no circular objects were detected, the bound would not update.

The primary goal of the ball detection stage was to isolate, detect, and locate the center of the basketball in each frame for all camera angles. To achieve this without a deep machine learning model, boosting was used to convert multiple average detectors into a stronger one. Three different detection methods were used:

1) Hough Circles: The preprocessing to complete Hough circle detection included applying a Gaussian blur with a 3 by 3 pixel kernel and converting the image to gray scale. The blur helped remove any background noise from the image. The Hough Transform used the equation of a circle and the gradient information of edges to detect circles that fit user-defined parameters [3].

2) Blob Detection with Background Subtraction: The preprocessing to complete blob detection with a background subtracted mask included applying a binary threshold on the background subtracted image, erosion with a 5 by 5 pixel kernel, and dilation with a 5 by 5 pixel kernel. Blob detection was then applied with various size and shape parameters to detect circular objects from the mask.

3) Blob Detection with Color Mask: The preprocessing to complete blob detection with a color mask included converting the image to HSV, binarization of the image with a color range, erosion with a 5 by 5 pixel kernel, and dilation with a 5 by 5 pixel kernel. Blob detection was then applied with various size and shape parameters to detect circular objects from the mask.

After the detection stage, each method had either detected a circular object or had not. If no method detected any circular object, then a 'None' or missing value would be stored in the list of ball centers. Otherwise, the priority of each method would determine which ball center values would be stored for later use in triangulation. The priority of the methods was Hough circles, blob detection with background subtraction, then blob detection with color mask. Fig. 5 visualizes the overall frame, along with the detection results of each method.

D. Triangulation

The cameras used to capture the free throw produced video that was in a 2D plane. Therefore, with the use of ball detection methods on the videos, only 2D positions of the ball could be obtained. 2D coordinates could not accurately represent an object's location in 3D space unless the camera is positioned perfectly at a 90 degree angle and has no distortion. In the case of free throws, this means the basketball's trajectory could not be accurately tracked or analyzed with 2D coordinates alone. The solution to this issue is to obtain 3D ball center coordinates from the various camera angles used to capture the free throw. 3D ball center coordinates of the ball's location throughout the entire free throw would allow for the recreation of its trajectory in 3D space. This 3D recreation offers drastically higher accuracy for ball trajectory than the use of 2D coordinates.

In order to obtain these 3D ball center coordinates, triangulation was used. In triangulation, 3D coordinates are recreated using an accurate relationship between points in 3D space and corresponding projection points in 2D space [4]. Triangulation is performed by using four components from each camera angle: the intrinsic camera matrix generated during camera calibration, the rotation matrix, the translation vector, and the 2D camera plane coordinates (pixels) [2]. In the pipeline, if more than two cameras detected the ball, then the two angles that missed detection the least were used to perform triangulation for that frame, as there were no built in multi-camera triangulation functions available.

The equation below models the relationship between points in 3D space and corresponding points in 2D space [2]. (u, v) is a 2D image coordinate from a video frame. (X, Y, Z) is a 3D coordinate recreated during triangulation.

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (3)$$

1) Rotation and Translation: Rotation and translation are the extrinsic parameters of a camera. Their task is to relate a camera coordinate system to a real world coordinate system. Rotation and translation are both captured using three parameters. For rotation, these three parameters can be thought of as equivalent to yaw, pitch, and roll [4]. The calculation of rotation and translation also takes into account the distortion coefficients, adjusting appropriately. The rotation is represented by a 3 x 3 matrix, while the translation is represented by a 3-element vector [4]. Even though the rotation matrix contains 9 elements, it only contains three parameters and thus 3 degrees of freedom [4]. The reason for a 3 x 3 rotation matrix is to allow for easy matrix multiplication once it is concatenated with the translation vector. The concatenated matrix multiplied by the camera intrinsic matrix is called the projection matrix, $K[R|t]$, since it projects a point in the real world coordinate system to a point in the camera coordinate system.

Using the projection matrix, it was possible to view 3D point and shapes in 2D space and visually observe the accuracy of projection matrices. Fig. 6 shows the perimeter of the 3D free throw box on the court surface. The 3D points on the perimeter were projected to the 2D center camera image plane. The result represents an accurate relationship between the coordinate system of the center camera and the real world coordinate system, which is important for 3D reconstruction.

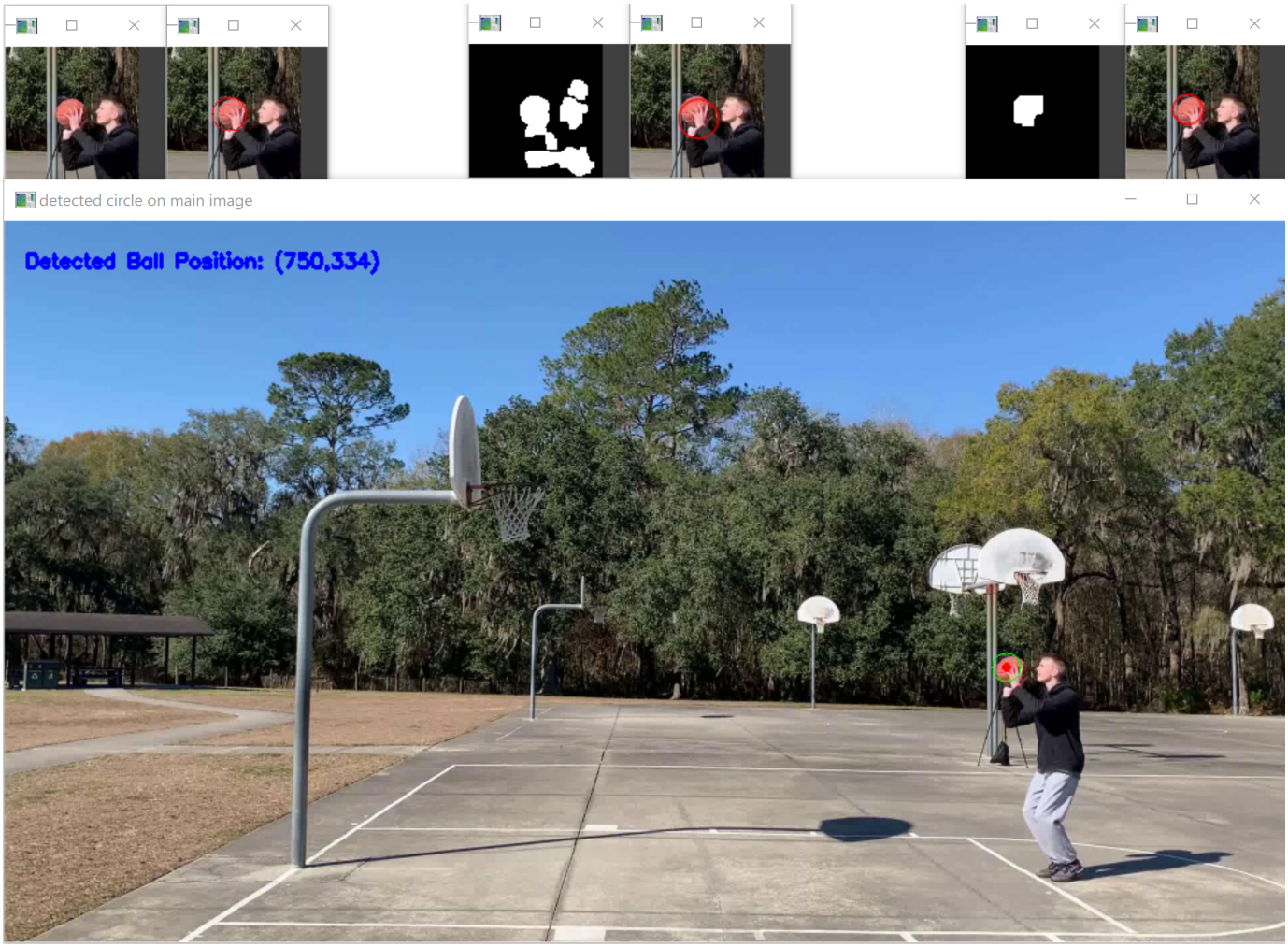

Fig. 5. Instance of ball detection on frame: (Hough, Background Subtraction, Color Mask)

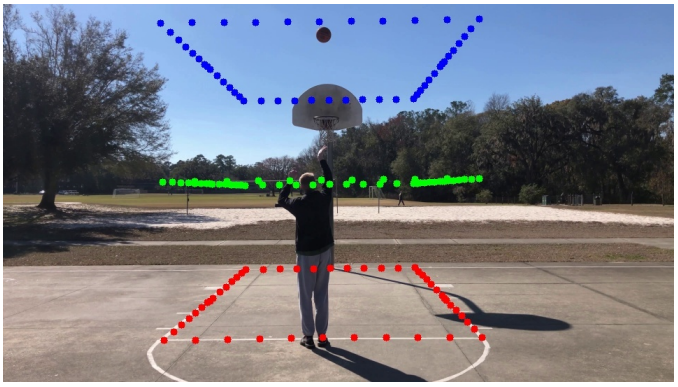

Fig. 6. Projection of 3D free throw box onto the 2D center camera plane

2) 3D Reconstruction: In 3D reconstruction, the intrinsic and extrinsic camera matrix are multiplied, creating the transformation between the real world coordinate system and the camera coordinate system.

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K[R|t] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{4}$$

With the use of the 2D point from the camera plane and the projection matrix, the triangulated 3D point can be obtained. In practice, triangulation is able to be accurately performed by using just two camera angles. Triangulation would require the intrinsic and extrinsic parameters, as well as the 2D image coordinates, from all camera angles used. One of the common triangulation methods available is direct linear transform, or DLT. The DLT method uses linear homogeneous equations to project 2D coordinates to 3D coordinates.

III. Results and Discussion

Upon obtaining the list of triangulated ball centers using video from the left, center, and right camera angles, many metrics were calculated to analyze the shot of the ball. The total velocity of the ball between each pair of frames was calculated, as the distance in real world coordinates was

estimated and frame rate was known. If the ball was not triangulated in a specific frame, then a velocity calculation was not possible at that time. The two consecutive frames where the ball was detected were then used, along with the time between those two frames.

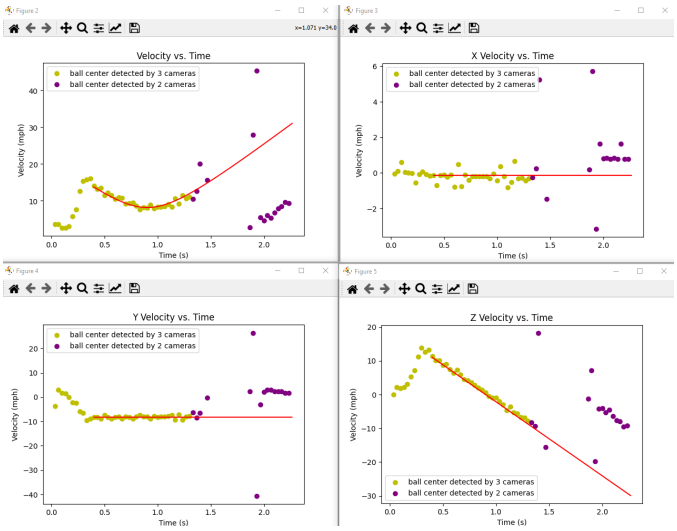

Fig. 7. Instance of velocity analysis

The velocity of the ball in each direction was also calculated. This was found in a similar manner to the total velocity, except only using directional data from each axis respectively. The velocity plots can be seen in Fig. 7.

From the directional velocities, a trajectory was estimated in Fig. 8 for the ball using the following projectile motion physics equations [5]:

$$X_c = x_0 + V_x t \tag{5}$$
$$Y_c = y_0 + V_y t \tag{6}$$
$$Z_c = z_0 + V_z t + gt^2/2 \tag{7}$$

(X_c, Y_c, Z_c) is the 3D coordinate of the ball center (x_0, y_0, z_0) is the chosen initial position of the ball, (V_x, V_y, V_z) is the directional velocity, t is time since the initial position point of time, and g is the gravitational acceleration.

Instantaneous velocities to initialize these equations were chosen at a point in time where the ball had left the player's hands. As can be seen in the directional velocity plots in Fig. 7 there was variance in the data. This variance could have been from marginal errors in ball center detection, triangulation, or calibration. With these marginal initialization errors came a larger final error in the projectile motion equations over time, hence the red trajectory diverging from the true ball position over time. From the user-defined 3D coordinate system, key features of the data set, such as the court and basket, were plotted in addition to the triangulated ball centers, as can be seen in Fig. 8. The height of the ball over time was also plotted to analyze arch of the player's shot in Fig. 9.

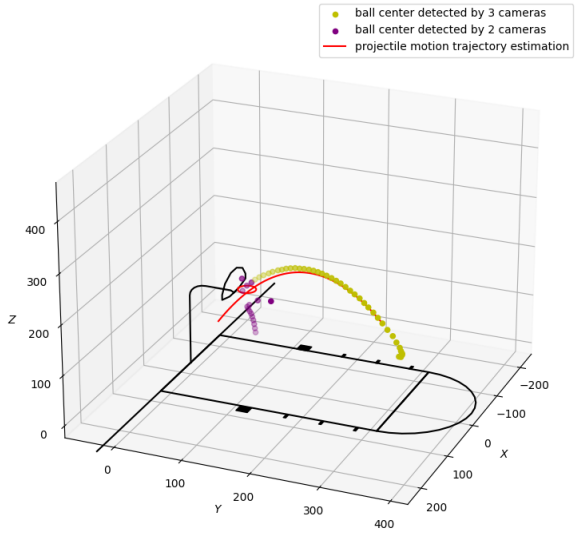

Fig. 8. Instance of triangulated centers and estimated trajectory

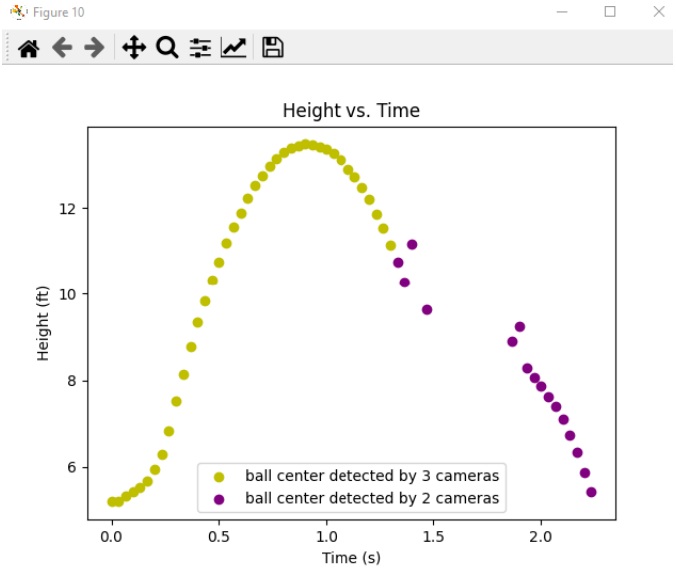

Fig. 9. Instance of triangulated ball height

IV. Conclusion

In this paper we present a pipeline for 3D sport analysis of basketball free throws from two or more video cameras. The flow included data collection, camera calibration, ball detection, triangulation, and data analysis. Future improvements could include increasing the amount of data in order to perform machine learning in the ball detection stage. Most of the error in the current pipeline came from the triangulation, which was the result of imperfect ball detection. The simplistic detection methods work extremely well on contrasting and clear backgrounds like the bright, blue sky but can perform worse on a noisy background like trees in the wind. A machine learning detection stage could also extend into any type of ball tracking, as area would not be as critical a factor like it is on

simplistic detection methods.

Overall, the purpose of the project was executed as expected, and the trajectory estimation was accurate enough to visualize whether the ball will hit inside or outside of the basketball rim. The position and velocity analysis also followed closely with the theoretical equations used. Several extensions could be made to this project in the future such as pose estimation to detect when the player is shooting and releasing the ball, automating whether the shot was made based on the ball movement after it approaches the rim area, or applying the provided analysis to other sports with a ball.

REFERENCES

[1] OpenCV 3.0.0-dev documentation, Camera Calibration and 3D Reconstruction, OpenCV, Nov. 10, 2014. Accessed on: Apr. 7, 2022. [Online]. Available: https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_calib3d/py_calibration/py_calibration.html

[2] FDX Labs, Calculate X, Y, Z Real World Coordinates from Image Coordinates using OpenCV, Apr. 2019. Accessed on: Apr. 7, 2022. [Online]. Available: https://www.fdxlabs.com/calculate-x-y-z-real-world-coordinates-from-a-single-camera-using-opencv/

[3] OpenCV, Hough Circle Transform. Accessed on: Apr. 7, 2022. [Online]. Available: https://docs.opencv.org/4.x/da/d53/tutorial_py_houghcircles.html

[4] R. Szeliski, 'Computer vision algorithms and applications', 2011. [Online]. Available: http://dx.doi.org/10.1007/978-1-84882-935-0.

[5] E. Ribnick, S. Atev, kal N. P. Papanikolopoulos, 'Estimating 3D positions and velocities of projectiles from monocular views', IEEE Transactions on Pattern Analysis and Machine Intelligence, t. 31, tx. 5, oo. 938–944, 2009.