Mid-term Project Report

Project Name: basketball predictor

Group Members:

Chentao Wang (cwang556@wisc.edu; NetID: cwang556);

Haozhe Luo (hluo45@wisc.edu; NetID: hluo45);

Xiaohan Sun (xsun233@wisc.edu; NetID: xsun233);

Houqi Li (hli492@wisc.edu; NetID: hli492).

1) Briefly explain what problem you are trying to solve.

We are planning to make a basketball predictor that can predict where the basketball will land and whether it will hit the basket. We want to solve this because as sports fans, we find that it's interesting to know if the basketball could shoot right into the basket before it lands. However, as human beings, our prediction is not that accurate. Therefore, we want to use the machine that has much better computing abilities to solve the problem. Our ultimate goal is to make the machine recognize basketball in a video and predict where the basketball will land when it just leaves the player's hands.

2) What is the current state-of-the-art?

- Applying Deep Learning to Basketball Trajectories & Applying Deep Bidirectional LSTM and Mixture Density Network for Basketball Trajectory Prediction
    - Yu Zhao, Rennong Yang, Guillaume Chevalier, Rajiv Shah, and Rob Romijnders used Deep Bidirectional LSTM and Mixture Density Network with RNN models to predict basketball trajectories by relying solely on the movement data. This method can predict a basketball shoot is a make or a miss. They have different models for different distances from the basket. The speed and the accuracy of the

trajectory of their model was said to outperform others. This method can help coaches and players to decide when and where to shoot.

- Two Dimensional Basketball Localization and Location Prediction
    - Jerry Giese and Matthew Wilson built a method to localize the trajectory of basketball shoots in two dimensional. The method contains image stabilization to get rid of camera shakes, Hough transforms to track the center of the ball, and then use the tracked coordinates of the center to predict the trajectory by using total least squares fit. For the tracking part, they used a Matlab function called "imfindcircles" which using Hough transform. They improved the method by adding priority to circles near the position of the ball from last frame and initializing with a manually-added position of the ball in the first frame. They used the first three to fifteen frames to fit a parabola to predict the trajectory. The method they used can predict the trajectory within three tenth of a second and do well with cluttered background. Several issues are: 1. It's just for 2D. 2. With low light, the basketball will become blurred and hard to predict. 3. Ignore the complexity of the shot will make or miss but just focus on if the parabola will cross the center of the hoop instead. Some statistical models may be needed.

3) Are you planning on re-implementing an existing solution, or propose your own new approach?**

- Right now we are re-implementing homework 2 solutions to track the ball in each frame and using physics formula to calculate the ball’s velocity (speed and direction). Then we

will use the velocity and the starting point to predict the trajectory and if it will miss or not. We also use what we did in hw2 to find the location of the board and the basket.

- We also notice that there are other methods such as the background subtraction method, which can also be used to track the moving object between several frames of a video. One research paper that is very similar to our project suggests using circular finding methods by MATLAB, which can also be useful. We will try them later and see which way will be the most ideal and fastest and if we can improve it for this specific scenario.

4) If you are proposing your own approach, why do you think existing approaches cannot adequately solve this problem? Why do you think you solution will work better?

- Not sure yet. We will need to look into more existing approaches and compare the performance and to see if we can improve based on them.

5) How will you evaluate the performance of your solution? What results and comparisons are you eventually planning to show? Include a time-line that you would like to follow.

- We will use another set of videos of shooting basketball to test our method. The set will contain both miss and score situations. We will first compare the predicted results and the real results to see the accuracy of our method. Then we will compare the time it takes to predict the trajectory and the real time of the shooting in the video, so we can tell if our method can have practical uses or not. Hopefully, if we can find other existing methods with public codes, we can also implement those methods and compare the accuracy and the running time with ours.
- Timeline:
    - Before Nov 10th: Finish building current prediction method and test

- Before Nov 4th: Take another set of videos for testing and finish the calculation of the trajectory using physics formula.
- Nov 4th - Nov 10th: Try out our methods and see if we can optimize the running time and the object detection algorithm to let it handle more general cases.
- Before Nov 25th: Try other existing methods and see if we can improve our owns.
- After Nov 25th: Collect results of our methods on the test set and prepare for the final report.

6) Problems we are facing:

- How to deal with the case that ball hits the board but jumps out of basket? Right now our solution is just to mark this a special case. We would like to learn how we can differentiate this case with a hit.
- Our code is very slow. To make this application helpful, we must speed up the algorithm. We are using Morphing and Condition checks, and we hope we could speed it up by setting up some thresholds. For example, we tried to add a threshold 10000 for area of each objects after morphing. Since original image is converted to labeled image, we first count the number of 1's in each object. Our ball approximately has 9000 pixels. If an object has more than 10000 pixels, it's not likely to be the ball. Then we don't need to compute its properties in our loop and go to next object. But still it takes 2 - 3 seconds to process each frame. We are trying to implement some syntax to replace for loop to improve the speed.

7) Process Tracking: kutt.it/534project