Automating Analyses of Cal Water Polo

Take your best shot: using tracking data to quantify what makes the best water polo

Yanxiu (Christy) Chen, IEOR MEng, christy_chen@berkeley.edu
Seungwoo (Sean) Son, CS BA, seungwoo_son@berkeley.edu
Yinghao Wang, IEOR MEng, yinghaowang@berkeley.edu,
Kai Li, IEOR MEng, kaili@berkeley.edu
Youyang (Yang) Zhang, IEOR MEng, yangzyy@berkeley.edu
Mentor: Panna Felson, EECS PhD Candidate, panna@eecs.berkeley.edu

Project Background

- Currently, analyses of water polo data requires painstaking, manual labelling of objects (players and ball).
- Over the last eight years, companies like STATS have fueled an "analytics revolution" by providing player and ball tracking data in professional sports.
- Our project will automate object detections, and Cal Water Polo team can use our models to analyze scoring such as the best shots, and types of shots.
- Our goal is to improve Cal Water Polo team's performance.

Data Collection and Organization

More than 150K clips were sampled from 27 games, recorded from cal water polo competitions. Events (scoring moments) were extracted from each game video. The coordinations of players and the ball in each event were labeled manually by human and transferred to json files for future use. The features of our data are the bounding boxes around the players and ball. For our final demo project, we filmed the latest Cal water polo game video (Cal vs Stanford) for the demo.

Conclusion

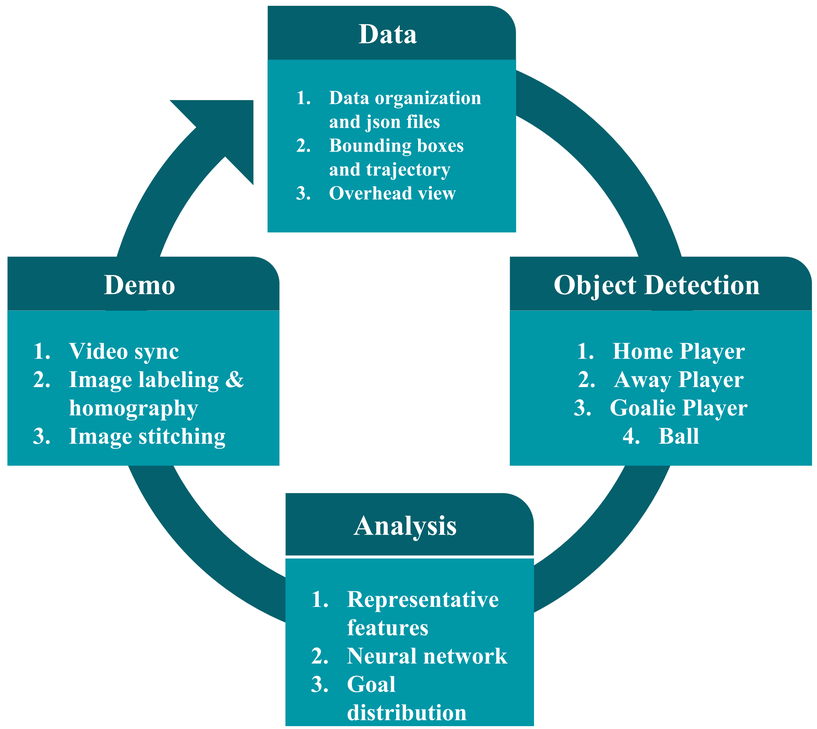
Our project will help Cal water polo team develop a more competitive scoring strategies by automating the process of object detection. Analyses will become more affordable and portable. We achieve promising per-frame detection performance of players (0.73 mean average precision) and the ball (0.62 AP). Through neural network, we can classify the shot types by inputting features including average speed and vertical direction. Our training accuracy can be as high as 90%.

Limitations of Project

In the classification model of shot type analyses, our features are not representative enough, thus the accuracy is not satisfying, given more time we will try more features and find out better input to train the model. For the demo, we only have two angels of the pool to stitch images. In the future, we could film with more angels to improve stitch accuracy.

Solution

We automate the analyses of Cal water polo with the following process. With the data we get, we train the Faster R-CNN object detector for future data analyses.



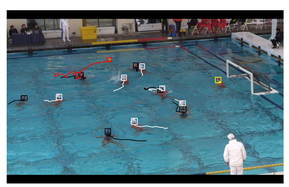


Image 1: Manually labelled objects

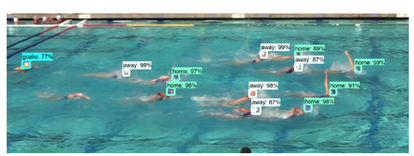


Image 2: Object detections

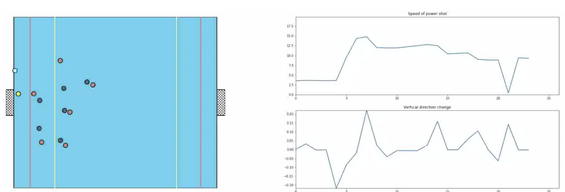


Image 3:Overview video with velocity and vertical direction curves

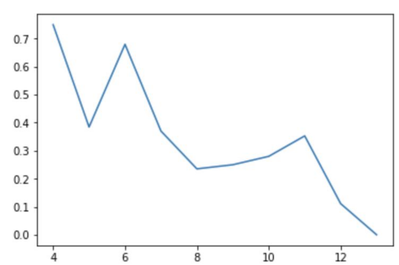


Image 4: Goal percentage by range

Approach

1. Data

Based on labels in json files, we created a model that draws bounding boxes and trajectories of players and the ball on each frame and merges all the frames together as a new video. We also designed a model to convert the camera 3D view to overhead view.

2. Object Detection

Bounding boxes are coordinates of x_min, x_max, y_min, y_max. To generate these bounding boxes, we trained a model with TensorFlow Object Detection. We decided to choose a pre-trained model between SSD mobile net and Faster R-CNN, as they have the highest accuracy. We ended up using the Faster R-CNN Model, because of its shorter training time.

3. Classification and goal distribution

Generate overview video with velocity curves of the ball for better review, extract features like average speed, vertical direction change and shot angles, and train neural network model to automatically classify the shot types.

Besides, we also get some pattern from those shots, such as scoring likelihood drops from 60% to 40% when shooting from the 2m line versus the 3m line.. Skip shots are 14% more likely to result in a goal than power shots.

4. Demo

First, we extract audio files from games clips and find the offset between these WAV files. Then, we trim these video clips based on the offset and extract every frame. Then, we calculate the homography between corresponding images and wrap the frames based on the calculated homography and stitch them by OpenCV. We repeat the last step and stitch all of the corresponding frames based on the same homography. Lastly, we can generate stitched video by combining all of the frames using FFmpeg.