Single-view 3D reconstruction of basketball scenes

Alexander Embiricos (embirico); Gabriel Poon (gpoon47)

March 19, 2014

Abstract

The introduction of tracking technology in sports, such as the SportVU system in the NBA, has provided profession analysts data with huge potential for advanced analysis. This data is unavailable to the general public due to the high cost required for installation and maintanence of the equipment. We propose a robust algorithm for camera calibration of NBA basketball scenes using a single image derived from television broadcast. We were able to achieve a high degree of accuracy of reprojected points for a large percentage of images seen from the video. This robust foundation for further development into an open-source 'SportVU' system.

1 Introduction

The proliferation of cameras and development of Computer Vision techniques has changed sports in significant ways. For example, the Hawk-Eye system (Hawk-Eye, 2014) determines the outcome of for out-of-bounds decisions in professional tennis and performs ball-tracking in other sports, and the 1st & Ten System (SportVision, 2014) places the first-down line on American Football TV broadcasts that has greatly enhanced the viewing experience for fans.

Computer Vision can also help teams develop new types of data to discover patterns that have evaded traditional means of analysis to discover exploitable advantages. The SportsVU system (Stats, 2013) developed by Stats LLC. has been employed by the National Basketball Association (NBA) for player, ball, and referee-tracking during all NBA games. Six cameras are placed high above the basketball courts, and provides the positional information of all recognizable moving objects on the court in 2D coordinates (with an extra dimension for the ball). This data has been used for quantitative analysis of behavior previously thought unfeasible to codify (e.g. Cervone et al. (2014); Maheswaran et al. (2014)), and appears to be revolutionizing analysis of the game.

Although SportVU provides an abundant amount of information, the data it collects is proprietary and is unavailable to the general public. One of the authors of this paper has had experience with this dataset, and believes there are fans who have the statistical knowledge to generate useful information from the data. The goal of this project is to create the foundation of a system that generates positional data from publicly available information of NBA games - TV broadcast of the games. Specifically, we want to create a system that can determine the 3D location of a point on the court in the image.

We propose a method that will automatically determine the homography from image coordinates to world coordinates. This homography can be determined from a single view image of the court,

Figure 1: Sample Image from TV feed of NBA game

such as Figure 1, following a set of procedures, which is outlined in Figure 2. A court mask - the area of the image that represents the court - is generated from the image. The top pixels of the mask is used to generate lines that represent the baseline and sidelines of the court. A second, more aggressive, court mask is generated, and is used in conjunction with the baseline and sidelines to determine the free throw and close paint lines. Finally we determine four intersecting points from perpendicular lines, which are then used to compute a homography.

2 Related Work

2.1 Existing Literature

Player tracking in sports is a topic explored in the literature since the mid 1990's (e.g. Intille and Bobick, 1995), and most of these approaches were based on multiple-camera models. Xu et al. (2005) tracked soccer players using multiple static cameras with overlapping views. Bebie and Bieri (1998) generated a 3D animation of a soccer game using two or more views of the court. The technology has matured to the extent that the NBA has adopted a 6-camera tracking system for its data collection.

Sports provides an avenue for single camera-view tracking because landmark points can be identified in the images. These landmarks have clearly defined locations in 3D space, which can be corresponded to image coordinates to determine a homography. Farin et al. (2003, 2005) determine landmarks in tennis views by utilizing white lines in the image view. They estimate line candidates by filtering white pixels in the image, and determine landmark points using these line candidates. Sports without regulated boundary line colors, like basketball, cannot benefit from this approach. Hu et al. (2011) circumvent this problem with a court masking approach. They utilize the fact that the color of the court surface is the dominant color of the image to filter out all pixels of the image that are not part of the court. Line candidates were estimated from this mask to find the landmark points. The algorithm outlined in this paper is largely based on Hu et al.'s work due to the many overlapping similarities.

2.2 Contributions of this Project

Previous literature covered applications of the method in relatively 'clean' images. For example, the approach Farin et al. (2005), Ohno et al. (2000) , etc. take to extract line candidates is by considering line candidates formed by white pixels in the camera feed. This approach would fail in any type of basketball broadcast video because line colors are not regulated in any basketball league, and will vary in color from court to court. Furthermore, as (picture of american football vs picture of nba court) demonstrates, there are more line candidates to consider, which increases the complexity of discovering points-of-interested that we would like to specify.

The current method is a further improvement from Hu et al. (2011) that perform 3D basketball scene reconstructions from single-views. The main difference is NBA basketball video feeds are much noisier than basketball videos from other leagues. First, an average NBA player occupies a much larger area on the image than players in any other league. Larger players mean more court features are occluded, which decreases the reliability of algorithms to locate particular points on the image. As a comparison, the average NBA player is around 6 foot 7 inches (Basketball-Reference.com, 2014), and the average height of Taiwanese women's basketball players, subjects of videos in Hu et al. (2011), is 10 inches shorter (USA, 2011). Second, there are many distracting visual elements in a TV broadcast of NBA games compared to the games or leagues studied in other papers.

Finally, NBA is without a doubt one of the most popular sports internationally. Many fans are excited about the potential for analysis of SportVU data, but are unable to obtain the proprietary dataset. Part of the goal of this project is to create a platfrom on which other NBA fans who have interest in Computer Vision can collaborate and develop an open-source version of SportVU using only publicly available information. The next steps would be to identify and track player and ball movements throughout the game with respect to the 3D coordinates we have identified. This can foster more intereset in NBA, and lead to more discoveries about the dynamics of the game beyond the constraints of team resources. Furthermore, the system is built for robustness, which permit application to other non-NBA basketball games such as the NCAA. This technology would allow college teams, which have a much smaller budget than professional basketball teams, to analyze opponents' strategies without paying scouts to travel to games, and to track multiple teams at the same time.

3 Technical Details

3.1 Algorithm Summary

The objective of the algorithm described in this paper is to calculate a homography between the a video image and the real world coordinates of a NBA-spec basketball court. As our algorithm relies on extracting features of the basketball court which are only visible at the sides of the court, we do not run it on shots which only show the center of the court, nor do we run it on frames of "action shots," which show closeups of plays at unconventional angles. Namely these features of interest are: 1) the sideline, 2) the baseline, 3) the freethrow box (see figure 3 for definitions).

Figure 2: Algorithm summary

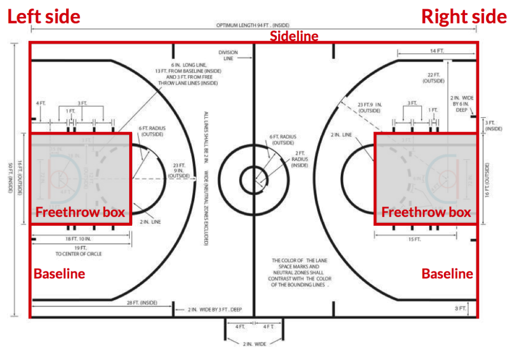

Figure 3: NBA spec court features

As shown in figure 2, we propose a pipeline which goes through several stages, taking us from a video file to a homography for each frame.

- We begin by extracting bitmap image files from the video file.
- Next, for any given frame, we compute the color histogram and use it to determine which color (and similar colors) are dominant (or most common) in the image. With some adjustment, this gives us the colorset of the basketball court.
- We use this colorset to create a binary image (aka "mask") distinguishing the basketball court from other parts of the image.
- We then run a Hough transform on the top points of this mask, which after some refinement, yields the sideline of the court, as well as the baseline of the court.
- Using our knowledge of these lines and a refined mask, we can extract two more lines around the freethrow box via Hough transform plus refinement.
- Now with four lines defined, we can define four points which have known coordinates in the image and in the model (see figure 4).
- Using these four points which share a common plane, we can estimate a homography from world coordinates to image coordinates.

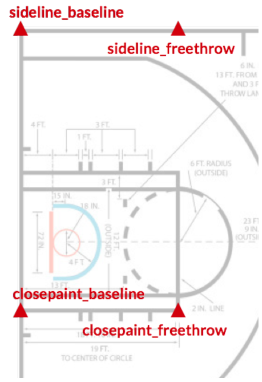

Figure 4: Four points used for homography estimation (shown for left side of court)

3.2 Algorithm Details

3.2.1 Video frame extraction

Our objective here is to extract specific frames from a source video file in BGR array form. (BGR refers to the Blue-Green-Red colorspace.)

This part simply utilizes OpenCV's `cv2.VideoCapture` utility. As the details of this are of little algorithmic interest, we leave the reader to see `src/extract_frames.py` for implementation details.

3.2.2 Dominant color detection

Our objective here is to determine which color values are most common in the image. On the assumption that, for the frames we run our algorithm on, the court fills most of the image, this set will include all of the color values that are contained by the court in our image.

The rationale behind dominant color detection is explained in section 2.1. In summary, we are unable to use Farin et al. (2003)'s method of line-pixel detection to extract features for homography estimation, due to the fact color difference between the court and the court lines in basketball is far less pronounced than in tennis or football.

We perform the dominant colorset calculation using the CbCr components of the YCbCr colorspace, which provides better results than using a more conventional colorspace such as BGR (Liu et al., 2006). The CbCr dimensions provide an accurate descriptor of color while remaining insensitive to lighting, which makes intuitive sense as we are ignoring "Y", which is the luminance component. The colorspace transformation can be simply performed using OpenCV's `cv2.cvtColor` utility.

Thus, in order to find the dominant colorset, we create a 2D color histogram of the image in CbCr space (in other words, we count the number of pixels with any given CbCr value). We can now define the "peak" P_1 of the histogram as the most common color appearing in n_1 pixels. We then construct the dominant colorset as the color values of the 4-connected region around P_1, only considering colors with counts greater than Tn_1. For our dataset, we found that setting the threshold $T = 0.02$ was an effective value.

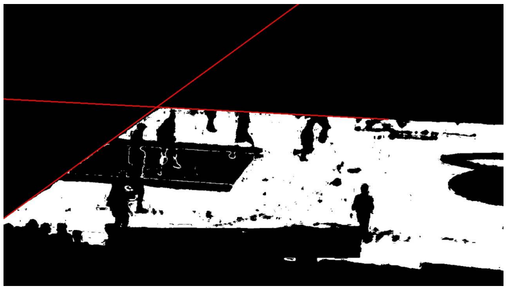

Figure 5: Court mask for detecting sideline and baseline (detected lines shown)

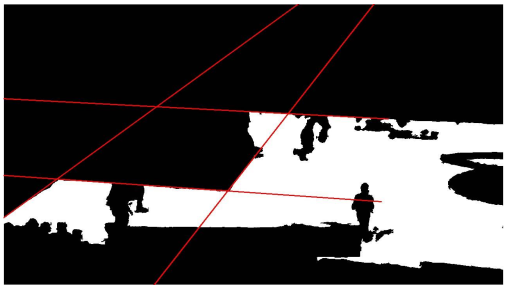

Figure 6: Court mask for detecting freethrow box (detected lines shown)

Unfortunately, the above algorithm as described does not work for some of the images in our dataset, due the audience taking up a large proportion of the image, and being of relatively homogenous color. A simple improvement to the above is to rely on our knowledge of basketball footage and to exclude the top and bottom pixels of the image, which roughly correspond to where the audience is, from consideration in the color histogram. We found that exluding the top 37.5% of the image, and the bottom 20.0% of the image allowed reliable results even under these conditions.

3.2.3 Court masking

Our objective in this section is to generate two court masks. In the first (see figure 5), we want to have the sideline and baseline of the court clearly defined for feature extraction. In the second (see figure 6), we want to have the freethrow box clearly defined for feature extraction.

To create the first mask, we simply create a new binary single-channel image where a pixel has a value of 1 iff its CbCR color value is contained in the dominant colorset (and a value of 0 otherwise). As shown in figure 5, the top edges of the white zone (1-values) are well defined.

To create the second mask which can be used to identify the freethrow box's bottom and center boundaries, we need to clean our image by filling holes in both the court space and the non-court space. We do this by first finding the external countours of the court space, and then filling this space with 1 values. For a second step we then find the external contours of the non-court space and then filling this space with 0 values. (Practically, this can be done by repeating the first step on an inverted image, and then reinverting the result.) As shown in figure 6, the bottom and center edges of the freethrow box (1-values) are mostly well defined.

3.2.4 Feature line detection

Our objective in this section is to identify four points in both the image and in the court model. We do this by calculating the intersection of the following 4 lines:

1. "Sideline": line that is collinear with the far-sideline
2. "Baseline": line that is collinear with the baseline
3. "Closepaint line": line that is collinear with the near closer-edge of the freethrow box
4. "Freethrow line": line that is collinear with the freethrow line

To find lines 1 and 2, we use the first mask found above. We begin by identifying pixels corresponding to the court border, which we define as the top nonzero-valued pixel in each column. We then feed these pixels into a standard hough transform with a low threshold (in order to be sure to find the baseline, which often has less votes than the sideline). Due to the low threshold, this results in close to 100 candidate lines (ρ_i, θ_i), many of which are near-collinear with our target line. The sideline is selected as the line with the most votes that has $\theta < 1.6$, and the baseline is selected as the line with the most votes that has $\theta \geq 1.6$.

To find lines 3 and 4, we use the second mask found above. We begin by using canny edge detection to find pixels to feed to hough. In order to reduce noise near the edges of the picture, we ignore all edges found in the top, bottom, left and right portions of the image. (See `hough.py` for exact proportions.) Then we use hough as before to detect lines parametrized by (ρ, θ). The freethrow line is selected as the line with the most votes that has similar θ and dissimilar ρ to the baseline. The closepaint line is then selected as the line with the most votes that has similar θ and dissimilar ρ to the sideline. (Again, see `hough.py` for exact definitions of "similar" and "dissimilar".)

3.2.5 Homography estimation

Our objective in this section is to use our mapping between 4 points in the image and 4 points in the model to estimate a homography between the image and model. We define the model based off measurements in figure 3, with the x-axis pointing from left to right of the figure, and the y-axis pointing from top to bottom of the figure. We define the plane of the basketball court as being at $z = 0$. In the image, we define the x-axis as pointing from left to right, and the y-axis as pointing from top to bottom.

We are looking for a homography H such that $P = HP_w$, where image point $P = (x, y, 1)^T$, and world point $P_w = (x_w, y_w, z_w, 1)^T$. As such H is a 3x4 matrix. However, since $z_w = 0$ for all points under consideration, we can work with a simpler definition of $P_w = (x_w, y_w, 1)^T$, resulting in H being a 3x3 matrix.

By writing out $P = HP_w$ in component terms for our 4 point correspondances, we can formulate

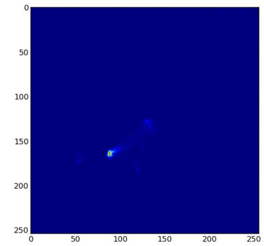

Figure 7: Color histogram used for dominant color detection

an equation constraining H (note that we make use of the normalization $h_{33} = 1$):

$$\begin{bmatrix} x_{w1} & y_{w1} & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_{w1} & y_{w1} & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & x_{w1} & y_{w1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{wn} & y_{wn} & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_{wn} & y_{wn} & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & x_{wn} & y_{wn} \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \\ 0 \\ \vdots \\ x_n \\ y_n \\ 0 \end{bmatrix}$$

4 Experimental Results

A representative color histogram used for dominant color detection is shown in figure 7. We note a main peak corresponding to the court colors (which are somewhat varied), as well as other much smaller peaks corresponding to the audience, freethrow box etc.

The most challenging component of this work was to estimate the image locations of four feature points. We assess our performance here by comparing the locations of the algorithmically computed points with the locations of points on the same image, chosen by hand. We manually selected feature points on 6 images in our dataset, and find a mean euclidean distance of only 8 pixels. We thus consider this a significant success.

In order to determine the robustness of the algorithm for the entire video feed, we calculated the percentage of images that we were able to find reasonable line candidates from our sample of images from a video feed. The algorithm performed reasonably well, i.e. line candidates were roughly matching actual court line positions, for 81% of the images, indicating that the algorithm is robust given the number of different 'shots' during a TV broadcast.

The homography estimation in section 3.2.5, produces results as shown in figure 8. We assess this section's performance by calculatin the average reprojection error, which is the average euclidean distance of reprojected points from eachother, and get a value of 23 pixels. However, we noticed that the reprojected points are consistently closer than the true coordinates together when the y-coordinate is greater, and consistently farther part when the y-coordinate is smaller. We tried several different methods of computing the homography, but the results remained the same.

Figure 8: Reprojection test: blue points are original points and red points are reprojected points

5 Conclusion

In this paper, we improved upon an existing algorithm by Hu et al. (2011) to calibrate cameras of an NBA court based on a single image. We demonstrated that the algorithm was able to perform with a high degree of accuracy despite a systematic error in the reprojected points. As a result of this work, we have demonstrated that it is possible to create a SportVU-like system using only publicly available information of NBA games, albiet limited to views where the sideline, baseline, free throw line, and close paint lines are visible.

References

Basketball-Reference.com (2014), "Nba league averages." URL `http://www.basketball-reference.com/leagues/NBA_stats.html`.

Bebie, Thomas and Hanspeter Bieri (1998), "Soccerman-reconstructing soccer games from video sequences." In *Image Processing, 1998. ICIP 98. Proceedings. 1998 International Conference on*, volume 1, 898–902, IEEE.

Cervone, Dan, Alexander DAmour, Luke Bornn, and Kirk Goldsberry (2014), "Pointwise: Predicting points and valuing decisions in real time with nba optical tracking data." In *MIT Sloan Sports Analytics Conference, 8th Annual*, MIT.

Farin, Dirk, Jungong Han, et al. (2005), "Fast camera calibration for the analysis of sport sequences." In *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*, 4–pp, IEEE.

Farin, Dirk, Susanne Krabbe, Wolfgang Effelsberg, et al. (2003), "Robust camera calibration for sport videos using court models." In *Electronic Imaging 2004*, 80–91, International Society for Optics and Photonics.

Hawk-Eye (2014), "About hawk-eye." URL http://www.hawkeyeinnovations.co.uk/page/about-hawk-eye.

Hu, Min-Chun, Ming-Hsiu Chang, Ja-Ling Wu, and Lin Chi (2011), "Robust camera calibration and player tracking in broadcast basketball video." *Multimedia, IEEE Transactions on*, 13, 266–279.

Intille, Stephen S and Aaron F Bobick (1995), "Closed-world tracking." In *Computer Vision, 1995. Proceedings., Fifth International Conference on*, 672–678, IEEE.

Liu, Yang, Dawei Liang, Qingming Huang, and Wen Gao (2006), "Extracting 3d information from broadcast soccer video." *Image and Vision Computing*, 24, 1146–1162.

Maheswaran, Rajiv, Yu-Han Chang, Jeff Su, Sheldon Kwok, Tal Levy, Adam Wexler, and Noel Hollingsworth (2014), "The three dimensions of rebounding."

Ohno, Yoshinori, Jun Miura, and Yoshiaki Shirai (2000), "Tracking players and estimation of the 3d position of a ball in soccer games." In *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, volume 1, 145–148, IEEE.

SportVision (2014), "1st & ten." URL http://www.sportvision.com/football/1st-ten%C2%AE-system.

Stats (2013), "Sportvu." URL http://www.stats.com/sportvu/sportvu.asp.

USA, World University Games (2011), "Women's basketball team capture gold over taiwan." URL http://www.wugusa.com/womens-basketball-team-capture-gold-over-taiwan/.

Xu, Ming, J Orwell, L Lowey, and D Thirde (2005), "Architecture and algorithms for tracking football players with multiple cameras." *IEE Proceedings-Vision, Image and Signal Processing*, 152, 232–241.