CameramanVis: Where the Camera Should Look

Jianhui Chen

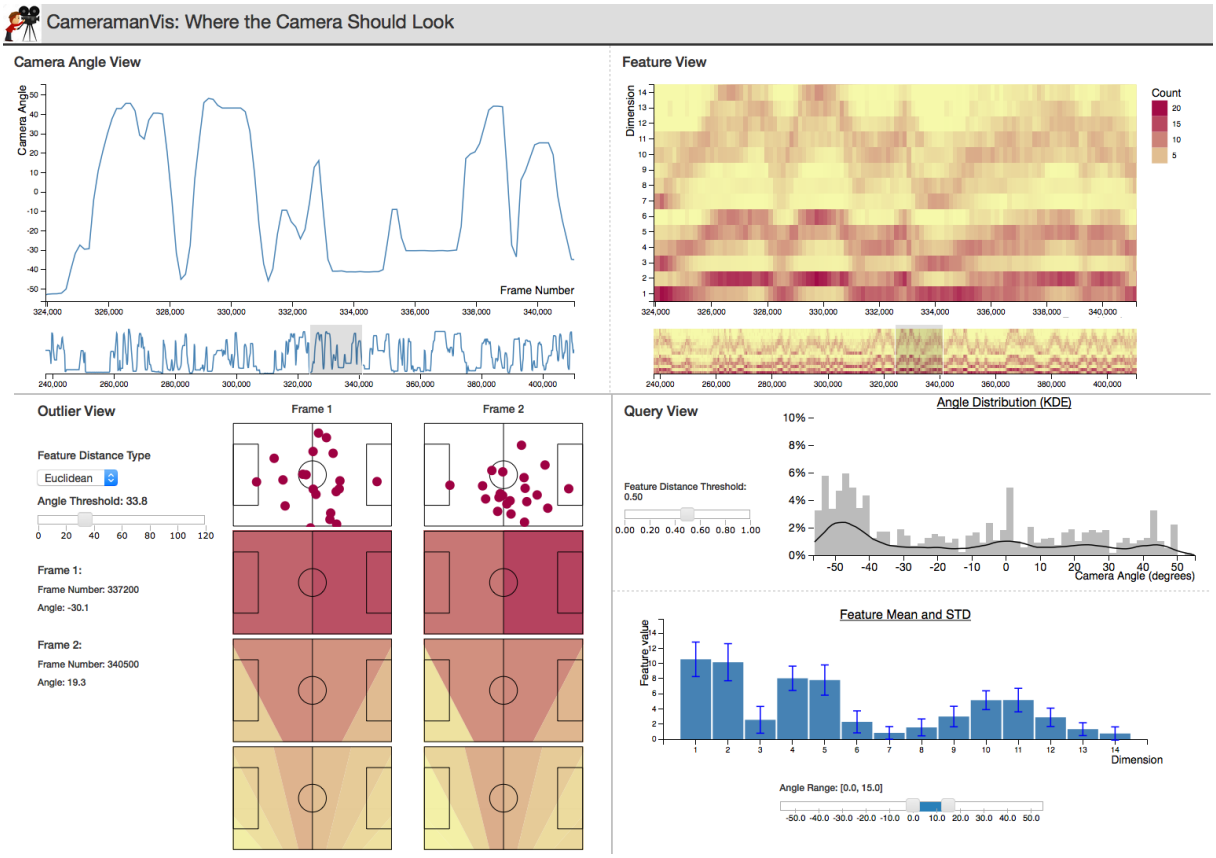

Fig. 1. CameramanVis visualizes correlation between the camera angle and player locations for soccer games. On this screen, users can browse the camera angles (top left) and features (top right) using brushes. A line char and a heat map will show the detailed view. On the bottom left, the outlier view shows outlier pairs when users set an angle threshold for filtering data. The outliers are depicted using side-by-side player locations (red dots) and multiple color maps on the soccer field. On the bottom right, the query view shows camera angle distribution using gray histogram and black line chart. It also shows feature distribution using blue bar chart with error bars when users set the query inputs using sliders.

Abstract— This paper introduces CameramanVis, an interactive visualization system for visualizing correlation between camera angles and player locations for soccer games. The system supports computer vision researchers to browse the soccer data, to identify camera angle and player location patterns, and to identify and compare outliers. The system visualizes camera angle, player locations and their correlation using three multi-form views. The system uses linked-highlighting to coordinate multiple views. In CameramanVis, we propose an intuitive spatial soccer field method to visualize features which are derived from player locations. Some preliminary user feedback suggests that CameramanVis is an effective tool for exploring correlation between camera angle and player locations. For long term sports analysis, we plan to use CameramanVis as a research prototype to investigate camera control, and multi-camera planning.

1 Introduction

Sports data is pervasive in science, entertainment and business. Camera angle is very important for capturing most interesting events in sports. Camera planning aims for predicting where the camera should look by understanding the environment. Algorithms for camera planning have been investigated for a variety of scenarios from scripted cooking shows and college lectures to team sports. It requires machine learning techniques to design features that describe the scene. For example, the mimicking system [5] models camera planning as supervised learning. It also requires computer vision techniques as the learning examples are estimated from videos of human operators. Although camera planning has been successfully applied to basketball games [4], applying the similar technique to soccer games encounters a number of challenges. For example, players are sparsely distributed in larger areas so that the patterns of player locations are less repeatable than the patterns in basketball games.

This paper proposes CameramanVis to visualize the correlation between camera angles and player locations for soccer games. By visu-

- *Jianhui Chen is with the University of British Columbia. E-mail: jhchen14@cs.ubc.ca.*

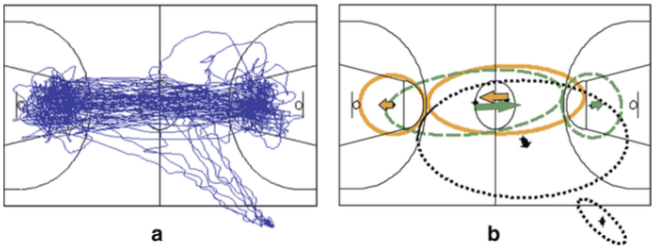

Fig. 2. (a) Trajectory of the team centroid; (b) Gaussian mixture models for the three phases of the game. The arrows show the direction and magnitude of the velocity component of the flow vector. Figure from [10].

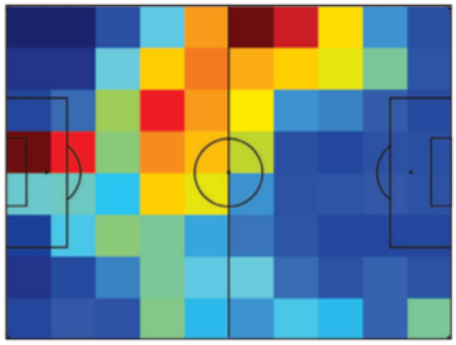

Fig. 3. Ball occupancy map over a half match for a team attacking left to right. The frequency of ball occupancy is coded by color on the soccer field. Figure from [3].

alizing the soccer data, we can have a better understanding of the data so that we can improve the camera planning algorithm. Fig. 1 shows the screen shot of CameramanVis. CameramanVis has three views: global view, outlier view and query view. It supports data analysis for computer vision researchers who work in automatic camera planning for soccer games.

In the CameramanVis project, we mainly did followings:

- Design and implement CameramanVis using D3 and other web techniques.
- Design a spatial soccer field feature visualization method which is very intuitive to users.
- Conduct a preliminary user study and obtain positive user feedback.

In the next section, we will present related work. Then we will introduce data and task abstraction, the proposed solution, and implementation details. Finally, we will present the result, discussion and conclusion.

2 Related Work

CameramanVis relates to both the visualization of sports and the visualization of camera data.

2.1 Visualizing sports data

In sports analysis, player trajectory and occupancy map are widely used for visualizing interesting objects such as players and balls. For example, Fig. 2 (a) shows that lines are overlaid on the basketball court to visualize the trajectory of players. Fig. 2 (b) shows that ellipses visualize the the derived game phases such as offensive play, defensive play, and time out. In Fig. 3, a heat map visualizes the ball occupancy on a soccer field.

Shooting data visualization has been extensively studied. For example, Goldsberry [7] proposed a more sophisticated heat map to quantify the shooting range of basketball players. Pileggi and others [11] introduced radial heat map shown in Fig. 4 to visualize the distance of shots from the net. The radial heat map uses a series of concentric rings surrounding the attacking goal representing cumulative bins of shots.

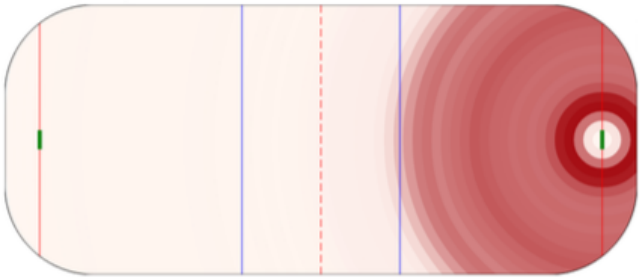

Fig. 4. Radial heat map conveys information about shot length. The darkest red ring represents the densest shot. Figure from [11].

Fig. 5. The camera trajectory in an 8 km route. Only the x, y positions of the camera is visualized using blue line. The reference is from Google map. Figure from [1].

Researcher have proposed visualization systems on soccer [9], baseball [6], and tennis [12]. However, their work is not directly related our work as ours is to analyze a specific sports broadcasting problem instead of a sports visualization system.

2.2 Visualizing camera data

Cameras have two types of parameters: intrinsic and extrinsic parameters. Camera data visualization usually refers to the visualization of extrinsic parameters which are the 3D position and orientation such as Euler angles. Because camera parameters are measured relatively to reference coordinates (such as world coordinates), a reference object is necessary except a default coordinate is assumed.

When the number of camera goes up to hundreds, only the dominantly varying parameters are visualized. For example, Fig. 5 shows the trajectory of a camera mounted on a self-driving car. The car traveled about 8 km in a city. The z position of the camera is assumed constant because the variance of the height is much smaller than the variance of x, y positions of the camera. Also, the orientation of the camera is fixed with respect to the car. In this case, the dominating varying parameters are x, y positions of the camera. As a result, a 2D line overlaying on a geography map effectively visualizes the camera. So, identifying and eliminating invariant camera parameters is very important to effective visualize large number of cameras.

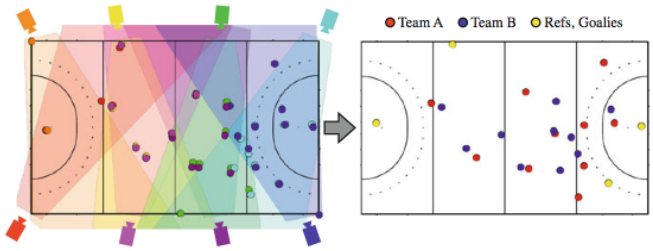

Fig. 6. Multiple camera views of field hockey game. Left: the view frustums are visualized by different colors. Right: different teams are represented by color circles. Figure from [2].

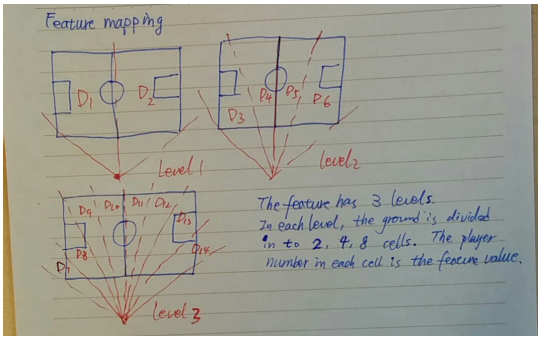

Fig. 7. Sketch of spatial soccer field feature. The feature has three scales. In each scale, the soccer court is radially divided into 2, 4 and 8 cells. The number in each cell is the feature value.

2.3 Visualizing both sports and camera data

Visualizing both sports and camera has been used to illustrate data-fusing from multiple camera in sports applications. Multiple cameras are able to cover the whole playing ground with sufficient resolutions. Fig. 6 left shows eight cameras looking at a field hockey game. The projected view frustums of cameras are visualized by colored polygons. Meanwhile, Fig. 6 right shows the players position in the playing ground using circles with different colors.

A number of visualization methods in CameramanVis are inspired by previous work. For example, the spatial soccer field feature visualization method is inspired by Fig. 6 left.

3 Data and Task Abstractions

3.1 Dataset

The dataset is from my research project. It was collected from a semi-professional soccer game. The original data size is 172, 800 frames (48 minutes). The data used in CameramanVis is uniformly down-sampled from the original data to 1, 142 frames for speeding up the algorithm. The data type is a table. Each item has the following attributes.

- **Frame Number** (sequential): a non-negative integer that represents time.
- **Camera Angle** (diverging) : camera pan angle. The range is in $[-56^o, 55^o]$ with 0 in the middle. Negative angle means that the camera looks at the left side of the soccer court.
- **Player Location** (spatial): player locations on the playing ground. It is a set of 2D locations in the rang of 115×75 yards.
- **Feature** (quantitative): quantized player locations in one frame. It is a 14 dimension multi-scale vector. It has 3 scales, and each scale divides the soccer field into 2, 4, and 8 sub-regions, respectively. The value in each dimension is the number of players located in corresponding sub-regions. Fig. 7 shows the sketch of the feature.

System	CameramanVis
What:Data	Table form. Each item has 3 main attributes.
What:Derived	Multi-scale spatial feature.
Why	Browse, discover, identify and compare
How:Encode	Line chart, color map, spatial color map, bar chart, and error bar
How:Facet	Overview + detailed views, linked-highlighting
How:Reduce	Filtering, aggregate
How:Manipulate	Focus + context
Scale:	About a half soccer match, sampled frames is about 1,500 frames

Table 1. Analysis of cameramanVis solution idiom.

3.2 Users and Tasks

We designed CameramanVis with the goal of supporting computer vision researchers to explore soccer data in the length about one hour. The typical users are graduate students, professors and research scientists in camera planning and sports analysis areas.

CameramanVis must support researchers in **browsing** the whole dataset, **discovering** correlation between the camera angle and player locations. For example, CameramanVis shall help researchers to answer questions such as:

- What is the angle distribution for a particular set of data?
- Are there repeatable patterns in the angle distribution?
- What is the player distribution for a particular range of camera angles?

CameramanVis is also designed to **identify** outlier pairs and **compare** these outliers.

Formally, CameramanVis mainly supports three tasks:

- Task one: identify and compare outliers
  Input: a sequence of data and an angle threshold
  Output: two frames that have the smallest feature distance but their angle distance is larger than the threshold
  Complexity: O(dN^2) in which d is dimension of features and N is the size of data
- Task two: identify camera angle patterns
  Input: a reference feature and a distance threshold
  Output: camera angle distribution of all data whose distances to the reference feature is less than the threshold
  Complexity: O(N)
- Task three: identify feature patterns
  Input: a range of camera angle
  Output: feature distribution
  Complexity: O(N)

4 Solution

The analysis of CameramanVis is summarized in Table 1. CameramanVis is intended to explore the correlation between the camera angles and player locations. To support tasks in Sec. 3.2, the solution is split into three views:

- **Global View** : For browsing whole data set, filtering data for outlier detection and selecting a reference feature for identifying angle patterns.
- **Outlier View**: For identifying and comparing outlier.
- **Query View** : For identifying angle and feature patterns.

4.1 Global view

Global view contains camera angle view and feature view.

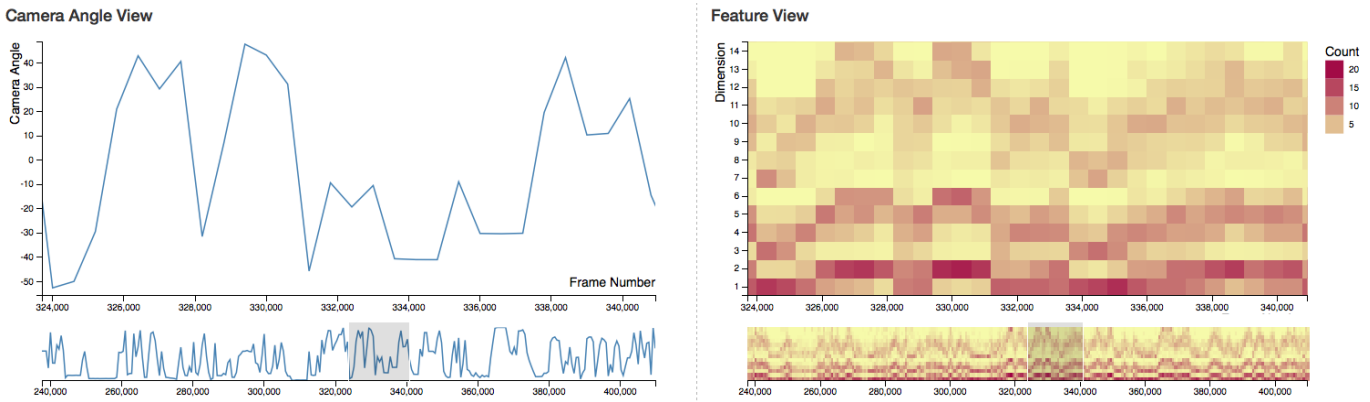

Fig. 8. Global view. The left sub-view shows camera angles versus frame number, and the right sub-view shows features versus frame number. In both views, users can use the brush to select a sub-sequence of data to explore the details.

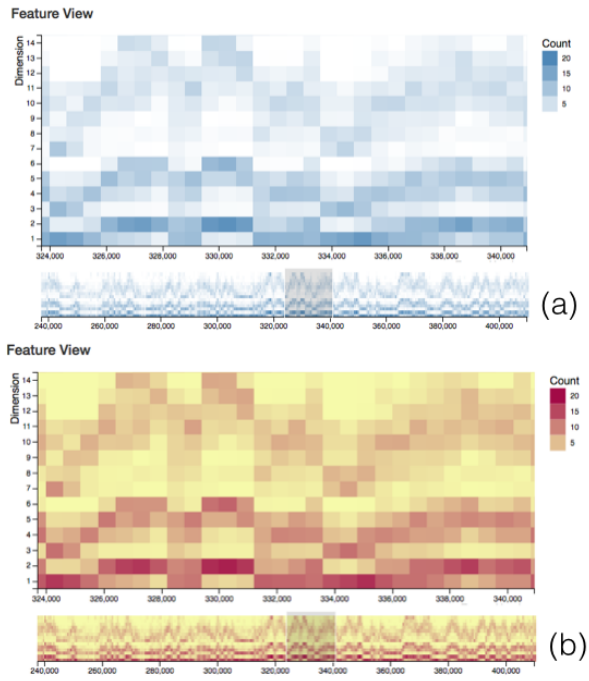

Fig. 9. (a) The white and steel blue were used in the initial heat map, (b) the light yellow and dark red are used in the final heat map. The final heat map avoids color confusing between the heat map and the background.

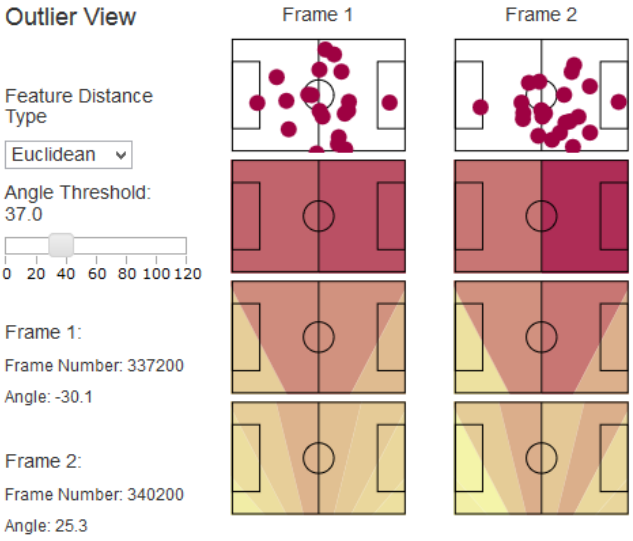

Fig. 10. Outlier view. This view shows side-by-side comparison of outlier pairs. The dark red circles are player locations. From the multi-scale soccer heat map, one can find the two frames are very similar in feature space. However, the corresponding camera angles are very different: the frame 1 is -30.1^o, and the frame 2 is 25.3^o.

4.1.1 Camera angle view

Fig. 8 left shows the camera angle view. It follows overview first, details on demand principle. It has two sub-views: the overview on the bottom and the detailed view on the top. There is gray block brush in the overview which is used to select the data for detailed view. The brush support expanding, shrinking and shifting range selection. A line chart visualizes camera angles.

4.1.2 Feature view

Fig. 8 right shows the feature view. The feature view shares most of the design idiom with camera angle view except that it uses a heat map to visualize the feature. The value of feature is coded using saturation as the saturation channels are automatically interpreted as ordered by our perceptual system.

Alternative heat map At first, I tried $[white, steel\ blue]$ color range as shown in Fig. 9 (a) for the heat map because the steel blue is consistent with the line chart in the camera angle view. However, the white color is confusing as the background color is also white. After trying several color ranges, $[light\ yellow, dark\ red]$ is used in the final implementation. This color range avoids the color confusing problem, and the saturation clearly encodes the player count.

4.2 Outlier view

Outlier view supports the task one outlier detection. Fig.10 shows the screen shot of the outlier view. It has three columns. The left column has a combo box and a slider bar to set parameters. At the moment, we do not have clear idea of which distance measurement is most suitable for multi-scale features. We provide "Euclidean" and "Manhanttan" distance selection so that users can decide which distance measurement is better. The bottom of the left column uses simple text output to show the frame number and angle of outliers.

The second and third columns are the main visualization part of the outlier view. We use side-by-side comparison of small multiple views to illustrate the difference between two frames. The views from top to bottom are player location, scale one, scale two and scale three of features. The feature is visualized by composing colors on the soccer

Fig. 11. The pyramid method using Matlab color map. The feature values are coded by rainbow color. The locations of players are coded by the orders from the left to the right. The scale of the feature is coded by a pyramid shape.

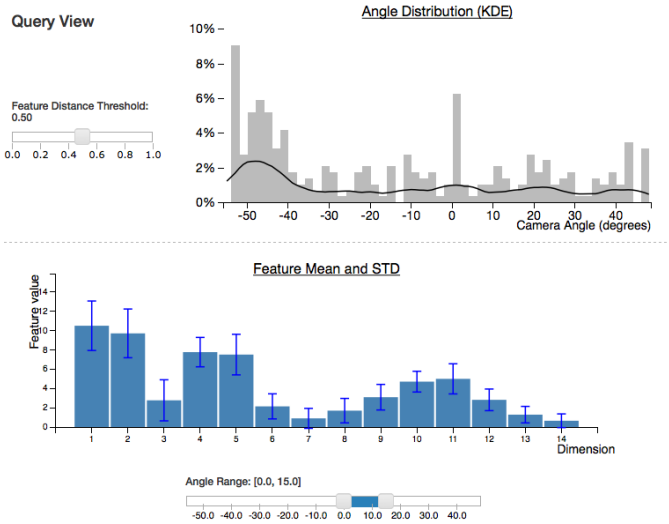

Fig. 12. Query view. The top sub-view shows camera angle distribution using histogram and kernel density estimation (KDE). The bottom sub-view shows feature distribution using the bar chart and the error bar. The KDE line shows the angle distribution has a peak around -50^o and uniform distribution in other places. The error bar shows that the dimension 7 and dimension 14 are unstable because they have relatively larger standard deviation.

field. The color is consistent with the color in feature view.

We have considered two methods to visualize the feature.

Spatial soccer field method Fig. 7 shows the sketch of our spatial soccer field method for visualizing features. It illustrates how each dimension maps to the spatial locations in the soccer field. Fig. 10 shows the visualization result for real data. Because this method composes color information on the soccer field, we call this method as spatial soccer field method.

Pyramid method Fig. 11 shows the pyramid feature visualization method. The mimicking system [5] used this method to visualize features in basketball games. We call this method as pyramid method as it looks like a pyramid. The feature value is coded by a rainbow color. The spatial locations of a group of squares are used to code the feature location on the soccer field.

The spatial soccer field Method is more intuitive as player locations are directly coded on the soccer field. It is also memorable to users. However, we found the pyramid method is more compact than the spatial soccer field method. We would like to use both of them in the system in the future.

4.3 Query view

The query view has two sub-views: query angle view on the top and query feature view on the bottom. Fig. 12 shows an example of query view.

The query angle view shows the angle distribution after user select a reference feature in the feature view and a normalized feature

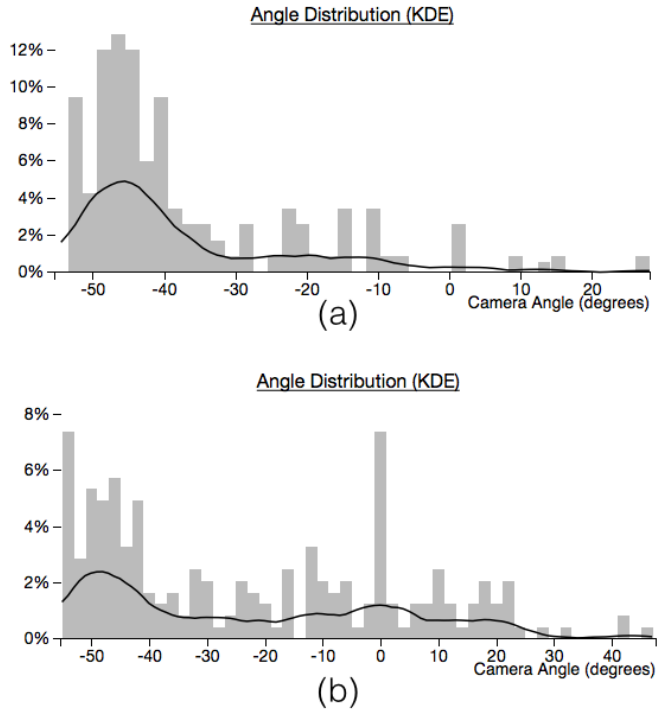

Fig. 13. Two angle distribution patterns. (a) One peak, (b) Two peaks. It means the one-to-one supervised learning is not suitable for the whole dataset.

distance threshold using a slider bar. We use "mouse click" instead of "mouse over" in selecting the reference feature because we found "mouse over" will be mistakenly triggered when users move mouse from one view to another view. The histogram and line chart (KDE) are superimposed to visualize the angle distribution of filtered data. They are complementary. The histogram discretely represents the distribution of data so that it provides more details. On the other hand, KDE represents the same data using a continuous line so that it clearly shows the patterns in the data. For example, Fig. 13 shows two patterns we found using CameramanVis: one-peak pattern and two-peak pattern. The two-peak pattern illustrates that the data has one-to-many mapping from the feature space to camera angle space. It reminds the researchers that one-to-one mapping method such supervised regression is not adequate to model the complex soccer data.

In the query feature view, the data is filtered by an angle range slider bar. The feature distribution is represented by mean value and standard deviation for each dimension of features. Bar char and error bar visualize the mean value and standard deviation, respectively.

4.4 Linked-Highlighting

There are two linked-highlighting between different views. The first one is selecting a sequence of data int the camera angle view for outlier detection. The second one is mouse clicking a feature in the feature view as a reference feature for exploring angle distribution. There was another linked high-lighting between the two brushes in the camera angle view and the feature view. It was removed in the finalized version as we found it irritated other operators.

5 Implementation

Libraries CameramanVis is implemented in JavaScript using D3, d3 slider and jQuery libraries. We use Bootstrap to structure the UI elements. We also use errorbar.js to draw error bars.

Data preparation The original data is from my research project. We generate the data using Matlab for CSV files and C++ program for JSON files.

Structure of code I implemented the following files:

- *cameraManVis.html* contains the html code for three views. It is adapted from VibBiz [14].
- *cmv_data.js* handles all data-related scripts such as loading and pre-computing the distance table.
- *cmv_global.js* contains global view scripts.
- *cmv_outlier.js* contains outlier view scripts. Specifically, *draw_space_feature* function draws spatial soccer field feature.
- *cmv_query.js* contains query view scripts.
- *cmv_main.css* contains most of CSS code.

These files are the major parts of whole visualization components.

How to draw spatial soccer filed feature using D3

- Draw a soccer field using circle and rectangles in a SVG. The size of the SVG is the same as the size of the soccer field.
- Draw arcs using D3. The center of the arc is along the center line of the soccer field. The start angle and end angle is determined by the dimension of the feature.
- Fill the arc area with mapped color from feature values, and the arc areas outside of the SVG will be automatically clipped as D3 only draw arc inside of the SVG.

6 Results

Scenario of Use John is a computer vision graduate student working on a robotic camera planning project. He wants to analyze how human operators control the camera to improve his automatic system.

First, John opens the CameramanVis with pre-loaded data. The system gives him the global view in Fig. 8. He browses and compares camera angles and features in the side-by-side views. He feels like the correlation between the two views is reasonable in general. He thinks it might be helpful to only look a short sequence of the data. He selects a short sequence in the camera angle view using the brush at the bottom. The main camera angle view immediately shows detailed pan angles. He also browses the feature in the same way and feels that the data so far is reasonable.

Then, John moves to the outlier view to check whether there are outlier pairs in the data. When he adjusts the angle threshold using a slider, the outlier pair is immediately identified (Fig. 10). By looking at the side-by-side small-multiples and the camera angles, John instantly finds that the outliers are somehow beyond his expectation. The player locations and features show that the two frames are very similar in feature space, however, the angular difference is more than 50^o. Why does it happen? He switches the feature distance type from the "Euclidean" distance to the "Manhattan" distance using a combo box selection. The outlier view shows similar result. He makes a record of the frame numbers for further study.

Next, he moves to query view to identify patterns of the pan angles and the features distribution. He wants to know the variance of the human operators given similar features. So, he selects one feature from the feature view using mouse click and adjusts the feature distance threshold. The camera angle distribution is immediately shown in the query view (Fig. 12 top). He identifies the distribution has a peak around -50^o and has uniform distribution in other places. He thinks it is because the feature distance threshold is too large. So he decreases the threshold value from 0.5 to 0.2, then the peak value appears around 0^o. He thinks the 0^o is a reasonable value as the players are around the center of the soccer field.

Finally, John adjusts the camera angle range using a range slier bar (Fig. 12 bottom). He moves the slider bar to the most left side and the most right side. The chart bar shows the mean and standard deviation of feature in each dimension. The two feature distributions have opposite patterns as shown in Fig. 14. It confirms his expectation that the camera angle follows the player positions in general.

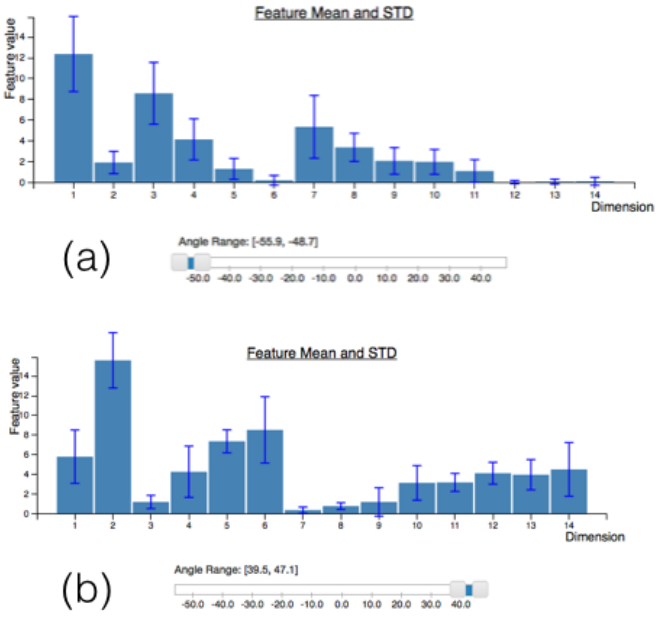

Fig. 14. Feature distribution. (a) When the camera looks at the most left of the soccer field, (b) When the camera looks at the most right of the soccer field. The bar chart shows the features has opposite patterns. The patterns confirm the fact that the camera angle follows the player positions in general.

John has a clear idea of the new collected data after further exploring camera angle distribution using different features.

Informal User Feedback I showed the intermediate version of CameramanVis to one of my fellow graduate students whose major is robotics. First, I briefly introduced the purpose of the system, the three views, and how to use the brush and slider bars. Then, she used the system for several minutes. Next, I asked her suggestions for the system. She said she likes the system in general. Then she gave following suggestions:

1. The outlier comparison should be column-by-column, but row-by-row.
2. Add legend in angle distribution and feature distribution otherwise users can not understand the histogram and bar chart.

I accept the suggestion 1, and the current version uses column-by-column comparison. I have considered the suggestion 2 but did not implement the suggestion in the current version as I think too much visual information in small areas is not a good idea.

I also showed the current version of CameramanVis to my supervisor who is a senior computer vision professor. He knows details of my research project. As we had limited conversation time, I explained the system using a recorded video. During my demon, he said he likes the side-by-side outlier comparison. He also immediately pointed out that the dimension 14 in the feature is unstable because the standard deviation is relatively larger than the mean value. I explained that it may be caused by the uneven distribution of player locations on the soccer field. After the demonstration, he supports my idea to send a refined version of CameramanVis to our partners.

Although the user study is taken with two persons informally, we believe CameramanVis is a promising system to explore the complex correlation between the camera angle and player locations.

7 Discussion

Overall, we believe CameramanVis is an excellent beginning of visualizing soccer data for camera planning. The system achieves many of its goals.

Our informal user feedback suggests that researchers intend to use CameramanVis. As the users are professional in the research area, they can immediately make decisions by comparing the visualized pattern and the pattern what they think should be.

Due to lack of time, CameramanVis has some limitations.

Limitations First, the system does not scale well to densely sampled data. The system is slow when sampled data number is large than 1,500. The negative effect sparsely-sampled data is that the line char is not smooth as shown in Fig. 8 left. One of the bottle neck of speed is the "brush" in D3, and data filter computation.

Second, some features are not fully implemented in CameramanVis. For example, the camera angle output is the text. It is better to visualize the camera angle using colored areas as in Fig. 6. Also, the reference feature is not highlighted after mouse clicking in the task two. Users can not remember which feature is used as a reference feature. And there are several unsolved bugs in the current system. For example, the histogram in the angle query view is occasionally incorrect when the feature distance threshold is smaller than a particular value.

Third, the current system only has the front end of the web application. It only runs on a local machine, which restricts its deployment.

Fourth, the user study is preliminary. To make the system more useful to other researchers, more user feedback is necessary.

Lessons Learned This project has been a valuable learning experience. We learned more than what we expected at the beginning of the project.

First, discussion is important. A number of major features of CameramanVis, such as spatial soccer feature, multiple distance measurement, and linked-highlighting are finalized after discussion with the course lecturer and fellow students.

Second, dividing the whole system to sub-systems makes implementation faster. This project gives my first web programming experience. To make the project work properly, we divide the system into 5 sub-systems and implement them separately. The standalone D3 examples in the Internet provide very useful prototype for our subsystems.

Third, applying the system to large data is more difficult than we expected. Because optimizing the speed not only takes time, but also requires deep understanding of web programming, we have to give up the original idea of densely sampling the original data. The desired sampling step is 30, but the current sampling step is 150.

8 Conclusion and Future Work

We presented CameramanVis, a useful visualization system that support browsing and identifying outliers, patterns from broadcasting soccer data for researchers. We employed linked-highlighting to coordinate multiple views in the system. We met the challenging goal of using CameramanVis for large densely sampled time series data. We conducted an informal user study, and users are enthusiastic about the system.

CameramanVis is our first step on visualizing the sports data for the purpose of algorithm design. In the future, we would like to combine CameramanVis with machine learning visualization methods such as [13, 8] to bridge the gap between data visualization and algorithm design.

References

[1] H. Badino, D. Huber, and T. Kanade. Real-time topometric localization. In *Robotics and Automation (ICRA), IEEE International Conf.*, pages 1635–1642, 2012.

[2] A. Bialkowski, P. Lucey, P. Carr, S. Sridharan, and I. Matthews. Representing team behaviours from noisy data using player role. In *Computer Vision in Sports*, pages 247–269. 2014.

[3] A. Bialkowski, P. Lucey, P. Carr, Y. Yue, S. Sridharan, and I. Matthews. Identifying team style in soccer using formations learned from spatiotemporal tracking data. In *Data Mining Workshop (ICDMW), IEEE International Conf.*, pages 9–14, 2014.

[4] I. Chant. Robotic cameras learn to follow basketball. `http://spectrum.ieee.org/tech-talk/robotics/robotics-hardware/robot-cameras-taping-basketball`, 2015. [Online; accessed 16-Dec-2015].

[5] J. Chen and P. Carr. Mimicking human camera operators. In *Applications of Computer Vision (WACV), IEEE Winter Conf.*, pages 215–222, 2015.

[6] C. Dietrich, D. Koop, H. T. Vo, and C. T. Silva. Baseball4d: A tool for baseball game reconstruction & visualization. In *Visual Analytics Science and Technology (VAST), IEEE Conf.*, pages 23–32, 2014.

[7] K. Goldsberry. Courtvision: New visual and spatial analytics for the nba. In *MIT Sloan Sports Analytics Conf.*, 2012.

[8] K. Lau. Random forest ensemble visualization. `http://kenlau177.github.io/Indented-Agg-Tree/`, 2014. [Online; accessed 16-Dec-2015].

[9] C. Perin, R. Vuillemot, and J.-D. Fekete. Soccerstories: A kick-off for visual soccer analysis. *Visualization and Computer Graphics, IEEE Trans.*, 19(12):2506–2515, 2013.

[10] M. Perše, M. Kristan, S. Kovačič, G. Vučkovič, and J. Perš. A trajectory-based analysis of coordinated team activity in a basketball game. *Computer Vision and Image Understanding*, 113(5):612–621, 2009.

[11] H. Pileggi, C. D. Stolper, J. M. Boyle, and J. T. Stasko. Snapshot: Visualization to propel ice hockey analytics. *Visualization and Computer Graphics, IEEE Trans.*, 18(12):2819–2828, 2012.

[12] T. Polk, J. Yang, Y. Hu, and Y. Zhao. Tennivis: Visualization for tennis match analysis. *Visualization and Computer Graphics, IEEE Trans.*, 20(12):2339–2348, 2014.

[13] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *ECCV*, pages 818–833. 2014.

[14] K. Zhang and S. Hasti. Vibviz. `http://www.cs.ubc.ca/~kzhang2/VibViz/`, 2014. [Online; accessed 16-Dec-2015].