Sports Camera Calibration using Flexible Intersection Selection and Refinement

Hiroki Tsurusaki†, Keisuke Nonaka (member)†, Ryosuke Watanabe (member)†, Tomoaki Konno†, Sei Naito (member)†

Abstract Sports scene analysis is an important technology to quantify a player’s action and visualize game statistics. To realize such technology, camera calibration is required to recognize the player’s position from a video. In this paper, we propose an automatic camera calibration method by using intersection resorting and refinement. Our contributions are 1) flexible intersection selection and 2) intersection refinement to improve accuracy in calibration. A homography matrix is used to convert world coordinates to image coordinates for the calibration. Sports scenes can be estimated by using a priori information such as length and position of field lines and their intersections. Conventional methods using the field lines and intersections cannot realize sufficient calibration accuracy because the intersections are selected from the combination of horizontal and vertical lines. Moreover, displacement at the intersections occurs between the detected position and a real one on the input image. Our proposed method can solve these problems by flexible intersection selection and refinement. As a result, a player’s position in the real world is identified from the video by using the estimated homography matrix. Our experimental results show that the proposed method achieves higher accuracy than that by conventional methods.

Key words: camera calibration, field line detection, intersection refinement, sports image

1. Introduction

Sports video analysis requires recognition of a player’s position from a video.1)‒3) Camera calibration is necessary to associate world coordinates with image coordinates by estimating internal and external camera parameters. The calibration is employed for applications such as free-viewpoint video, which creates a 3D model from multiple camera videos.4)

The camera calibration method is classified into two categories.5) One is to use a calibration object5)‒7) and the other is to use a priori information (a priori information-based method).1)‒4)6)‒12)

The calibration object consists of two or three orthogonal planes, which realize high calibration accuracy because the features for calibration can be obtained easily from the calibration object. In a sports stadium, however, it is difficult to set the calibration object in the field during the game even though re-calibration is required due to an accidental touch of the camera position and settings. For this use-case, a short running time of the re-calibration is desirable.

The a priori information-based method can calibrate cameras without the calibration object. The calibration accuracy is, however, inferior to that of the calibration object-based method because the feature extraction is difficult.

For a sports video, the calibration method using the field lines as a priori information has been proposed.1)4)6)‒13) Yao et al.7) classified detected lines into horizontal and vertical directions to find intersections. In this method, calibration accuracy depends on the selected lines pattern. Since the search candidates of intersections are limited by the combination of the selected lines, the sufficient number of intersections required to estimate the homography matrix cannot be calculated. In this case, it is not possible to select intersections that will result in higher calibration accuracy. Moreover, when the detected lines are not aligned with the real field lines in the input image, displacement occurs between detected intersections and real intersections of the input image. This displacement affects the calibration accuracy.

In this paper, we propose a calibration method using a priori information for a sports stadium where it is difficult to set the calibration object. Moreover, we propose a method for flexible intersection selection and intersection refinement. The proposed flexible intersection selection is expected increase the number of intersection patterns in order to improve calibration accu-
racy for the synthesis of free-viewpoint video. The proposed intersection refinement improves the intersection position by re-detecting field lines around detected intersections, and re-calculating intersections. Since the large demands on the 3D model are the penalty area of soccer, the penalty area is assumed to be contained in the target video.

2. Related works

2.1 Overview of a priori information-based method

Much research has been conducted on the camera calibration method utilizing a priori information such as field lines and their intersections in a sports video. Chen et al.\(^4\) proposed a deep learning-based method for camera calibration. R.A. Sharma and Chen et al. also proposed deep learning-based methods for camera calibration,\(^4,12\) which achieve high calibration accuracy, but they require training datasets.

Homayounfar et al.\(^10\) proposed a calibration method using the vanishing points of the field. The calibration accuracy by Homayounfar’s method is inferior to that by the learning-based method because the homography matrix is estimated by two vanishing points. With only two vanishing points, small displacement has a significant effect on the calibration accuracy, and there is concern regarding significant performance degradation. On the other hand, Yao’s method\(^4\) uses many intersection patterns to estimate a homography matrix, so that even if some intersections have the displacement, the accuracy is guaranteed by other points.

Yao et al.\(^4\) proposed a calibration method using the following field template. Here, the field template is a model of field lines on a sports court whose physical length is generally standardized. First, a field area is extracted from an input image. Next, field lines are detected by probabilistic hough transform\(^15\) from the field area. Then, the intersections are determined by selecting some lines from all the detected lines. Corresponding intersections are selected from the field template. Homography matrices are then estimated by intersection pairs between the input image coordinate and the field template coordinate. Here, estimating the homography matrix is equal to estimating the internal and external camera parameter.\(^2\) Finally, a single matrix is determined based on certain criteria. We describe Yao’s method in detail in the next subsection because the proposed method is based on such method.

2.2 Calibration with probabilistic hough transform\(^4\)

(1) Field extraction
In many sports, the color of a field area consists of similar colors, which are distinguished from those of another region. The field area can be extracted based on the chromatic threshold. For example, a typical method is described as

\[
\text{mask}_{i,j} = \begin{cases} 
1, & I_{i,j}^{H,S,V} \in [\sigma_{H,S,V}^{\min}, \sigma_{H,S,V}^{\max}] \\
0, & \text{otherwise}.
\end{cases}
\]

Here, mask defines the field region as binary data. The mask values “1” and “0” mean the field region and the outside of the field region, respectively. \(i\) and \(j\) are indices for the vertical and horizontal pixel positions, respectively. \(\sigma_{H,S,V}^{\min}\) and \(\sigma_{H,S,V}^{\max}\) denote the upper and lower limit of H, S, and V components of an input image, respectively. \(I_{i,j}^{H,S,V}\) denotes H, S, and V components of the HSV color space. The mask image is divided into the continuous region of “0” or “1”. After that, the continuous region of “1” having the maximum number of pixels is defined as the field area.

(2) Line and intersection detection
White lines are detected as field lines by probabilistic hough transform\(^15\) from the field area. The detected lines are classified into horizontal and vertical directions. After that, horizontal and vertical lines are selected to identify intersections.\(^10\) When the line detection fails, a homography matrix is not calculated due to the lack of the number of intersections.

(3) Homography matrix estimation
The homography matrix projects \((x, y, z)\) to \((X, Y, Z)\), which is defined as

\[
\begin{bmatrix} 
x \\
y \\
z \\
\end{bmatrix} \sim \begin{bmatrix} h_{11} & h_{12} & h_{13} \\
h_{21} & h_{22} & h_{23} \\
h_{31} & h_{32} & h_{33} \\
\end{bmatrix} \cdot \begin{bmatrix} X \\
Y \\
Z \\
\end{bmatrix}.
\]

When two vectors are normalized by \(z\) and \(Z\), the number of unknown parameters of the homography matrix is eight. Therefore, the matrix can be estimated by four intersection pairs between an input image coordinate and a field template coordinate.

Although an intersection pair is easily obtained by classifying the detected lines into horizontal and vertical lines in the conventional method, the number of available intersection pairs is small. On the other hand, the number of available intersection pairs increases by using all lines, which introduces the difficulty of matching between an input image and the field template. In
the proposed method, a point matching method is proposed to solve the matching difficulty, which improves the calibration accuracy by the increase of the number of intersection pairs. Intersection refinement is also proposed to further improve the calibration accuracy.

3. Proposed method

Our calibration pipeline is shown in Fig. 1. Our proposed method extracts a field area from an input image, which is the same as Yao’s method. Lines are detected from the field image as described in section 3.1. After that, intersections are calculated from the detected lines as described in section 3.2. In order to estimate homography matrices, intersection pairs are generated by point matching method between the input image and the field template as described in section 3.3. This matching method is our main contribution. Many homography matrices are calculated from intersection pairs because of pair candidates. Finally, an appropriate matrix is determined from several candidates by the proposed evaluation described in section 3.4. As an additional method, both detected intersections and the homography matrix are refined in section 3.5.

3.1 Line detection and selection

It is difficult for the probabilistic hough transform to handle various scenes because many parameters are adjusted for each input image. Our proposed pipeline uses the Line Segment Detector\(^\text{14}\) (LSD), which does not require parameter adjustment because the detected lines are defined as \(L(\theta, \rho)\), where \(\theta\) and \(\rho\) denote angle and distance in polar coordinates. Some detected lines are merged because one long line is sometimes detected as many short lines due to noise. A merge condition is defined as follows.

- The absolute difference of \(\theta\) between each line is less than \(T_\theta\).
- The absolute difference of \(\rho\) between each line is less than \(T_\rho\).

Some lines satisfy the above conditions even if they are not on the same line. To avoid this situation, when the distance of each line is greater than \(T_y\), lines are not merged. Here, \(T_\theta\), \(T_\rho\), and \(T_y\) are thresholds, respectively. The \(W^0\) image is generated by drawing white lines on a black image followed by the dilation process to the white line. The \(W^0\) image is used for the homography matrix evaluation. The top \(N\) lines are selected in descending order by line length because the field lines are longer than other lines. When many lines are detected from one field line, only one line should be selected to avoid calculating the intersection in a similar position. Algorithm 1 shows the line selection algorithm in detail. ML and SL denote merged lines and selected lines. ML is sorted in descending order by line length. This algorithm selects lines from different lines in the input image.

After line selection, it is evaluated whether candidate lines lie on the white line of an input image. We evaluate selected lines (SL) by using the white line rate, which is defined as

\[
\text{WhiteLineRate} = \frac{\text{Number of pixels on the white line}}{\text{Number of all pixels}}.
\]  

(3)

When the selected line of the white line rate is less than 80%, this line is removed from SL.

3.2 Intersection detection

The intersections are obtained from \(N\) selected lines. When the selected lines overlay the field lines of an input image, all intersections are on the field lines. However, when the selected lines do not overlay the field lines of an input image, intersections are not on the field line. Such intersections are removed. The intersections after removal are referred to as \(P_{SL}\) in this paper.

3.3 Point matching with sort

In order to estimate a homography matrix, intersection points are matched between an input image and the field template. The recognition of an intersection position from the image is essentially difficult because valuable features of the intersection are insufficient. Due to computation power, we avoid the recognition but utilize the brute force approach with sort method-based pruning.

Algorithm 2 shows the proposed sorting algorithm.
Algorithm 2 Sorting Algorithm
1: for $i = 1$ to $4$ do
2:     if $\text{abs}(L_{\text{slope}}) > 1$ then
3:         $V_i$ sort in ascending order by $x$ coordinate.
4:         if $x$ coordinate is the same among $V_i$ then
5:             if $L_{\text{slope}} < 0$ then
6:                 $V_i$ sort in ascending order by $y$ coordinate.
7:             else
8:                 $V_i$ sort in descending order by $y$ coordinate.
9:         end if
10:     else
11:         $V_i$ sort in ascending order by $x$ coordinate.
12:     end if
13: else
14:     $V_i$ sort in ascending order by $y$ coordinate.
15:     if $y$ coordinate is the same among $V_i$ then
16:         if $L_{\text{slope}} < 0$ then
17:             $V_i$ sort in ascending order by $x$ coordinate.
18:         else
19:             $V_i$ sort in descending order by $x$ coordinate.
20:     end if
21: else
22:     $V_i$ sort in ascending order by $y$ coordinate.
23:     end if
24: end for
25: return $V_i$, $V_i'$

$V_i$ and $V_i'$ denote points from the input image and the field template, respectively. $L_{\text{slope}}$ denotes a slope of the longest line among detected lines. $V_i$ is sorted based on the absolute value of $L_{\text{slope}}$. By using sorted $V_i$, the number of combination patterns is reduced from $M! \times pC_M$ to $pC_M$. Here, $M$ and $P$ are the number of selected intersections and all intersections calculated by SL, respectively.

When shooting a soccer video of a half court or penalty area, it is difficult for the proposed sorting algorithm to recognize whether the soccer video is on the left or right side. Therefore, we provide the side information for each input image. “the side information” indicates whether the camera shoots the right-side or left-side of the field. The number of patterns is further reduced from $pC_M$ to $pC_M$ by using the half field template.

3.4 Homography matrix estimation and evaluation

The final homography matrix is determined so that the value of the evaluation function can be maximized. A binary image is generated by projecting the field template onto a black image based on the estimated homography matrices. The evaluation function of the homography matrix is defined by

$$E = \frac{C(\text{AND}(W^0, W^n)) - \alpha C(\text{XOR}(W^0, W^n))}{C(W^0)}.$$ (4)

$W^n$ denotes the binary image generated by the $n$th homography matrix. AND$(a, b)$, XOR$(a, b)$, and $C()$ denote the function of logical multiply, exclusive disjunction of the image $a$ and $b$, and counting non-zero pixels, respectively. $C(\text{AND}(W^0, W^n))$ evaluates whether the white line of $W^0$ is identical to that of $W^n$. When all pixels of $W^n$ are white, $C(\text{AND}(W^0, W^n))$ becomes a high value. To avoid this, $C(\text{XOR}(W^0, W^n))$ is added to evaluate non-white line pixels.

3.5 Intersection refinement

The positions of the calculated intersections are sometimes different from the real positions of intersections in the input image because detected lines are not aligned with the field lines. We assume the difference is not so large on image coordinates but it significantly affects the accuracy of the estimated homography matrix. In order to reduce the difference, we recalculate the position of the estimated intersections with a small patch-wise strategy as described next.
Fig. 2 The process of intersection refinement. White and color lines denote real field lines and detected lines, respectively. (1) If the absolute difference of $\theta_{l_{SL1}}$ and $\theta_{l_{DL}}$, which are theta of $l_{SL1}$ and $l_{SL2}$, is less than $\tau$, $l_{SL2}$ is classified in $G_1$. (2) Normal lines are merged, and dashed lines are not merged. (3) Refined intersection is calculated by two center position lines ($CL_1$ and $CL_2$) between the top two lines of length among $G_1$ and $G_2$.

A patch with $127 \times 127$ pixels is extracted from an input image where the center position of the patch is on the intersection. From our assumption, most of the patches include the real intersection and lines. The thinning process shown in (1) to (6) is applied to ensure that the patch contains white lines only:

1. bilateral filtering\(^{17}\)
2. binarizing to 0 or 255
3. $3 \times 3$ median filter
4. dilation process
5. thinning process by Zhan et al.\(^{18}\)
6. gaussian filtering

As a result, only white lines are included in the patch defined as the thinned line patch. The thinning process has some parameters. In our experiment, the parameters of the bilateral filter are $\sigma_r = 3$, $\sigma_d = 8$, and $T_{x,y} = \text{avg} + 7$, where avg is the average of $13 \times 13$ pixel values from the input image at $(x,y)$ as the center position.

The field lines from the patch are re-detected to refine the intersection. LSD is applied to a processed patch by the thinning process to detect the lines. The detected lines with LSD are defined as DL. We assume here that a line in DL is likely to represent the real white field line better than the one in SL because the patch (small size image) contains a few noises, e.g. other objects. Note that the LSD detects the line along a boundary between the white line and field, not the center of the white line. In other words, one white line has at least two detected lines along both sides of the white line. Let $l_{SL1}, l_{SL2}$ in SL be the two lines containing a certain intersection point in $P_{SL}$ determined in section 3.2 (a blue point in Fig. 2 (1)). In addition, a line in the DL detected in the patch by the above process is denoted by $l_{DL}$. After the line re-detection, a new intersection is calculated following three steps. Note that we describe only the case of $l_{SL1}$ in detail, but it can easily be applied to the case of $l_{SL2}$.

1. Line classification
2. Line merge
3. Intersection calculation

Figure 2 shows the process of intersection refinement. (1) A similarity between $l_{SL1}$ and $l_{DL}$ is calculated as follows: $S_{SL1,DL} = |\theta_{l_{SL1}} - \theta_{l_{DL}}|$, where the $\theta_{SL1}$ and $\theta_{DL}$ are the angle of $l_{SL1}$ and $l_{DL}$, respectively. If the $S_{SL1,DL}$ is smaller than a certain threshold value $\tau$, the $l_{DL}$ is classified into a class $G_1$ considering the line is a better candidate to replace the $l_{SL1}$. We repeat this classification for all the $l_{DL}$ in DL.

(2) Although the $l_{DL}$s are good representations of the real white field line in a patch, these lines are fragmented even if these are along the same real field line. We therefore merge these fragmented lines if they seem to belong to the same real field line. To perform the merge, we simply calculate a distance $d$ among the endpoint of all the lines in $G_1$. If the $d$ between two lines is less than the threshold, we merge the two lines as one line. Then, after the merge, we select the top two lines in $G_1$ as $l_{SL1,DL}^1$ and $l_{SL1,DL}^2$.

(3) As described above, the LSD detects the boundary between the white line and the field on both sides of the white line. We therefore calculate the center be-
Table 1 The parameters of field extraction.

|   | H  | S  | V  |
|---|----|----|----|
| σ_{min} | 40 | 70 | 5  |
| σ_{max} | 90 | 255| 255|

Table 2 The parameters of line merge, selection, and refinement.

| value | T_{o1} | T_{o2} | T_{o3} | T_{o4} | r  |
|-------|--------|--------|--------|--------|----|
| 0.05  | 15     | 0.26   | 10     | 5     |

between \( l_{SL1,DL}^1 \) and \( l_{SL1,DL}^2 \) as one line \( CL_1 \). This \( CL_1 \)
represents the exact position of the white line (not the boundary) and helps us to find
the exact intersection. By obtaining the \( CL_2 \) in the same way and using the
\( CL_1 \) and \( CL_2 \), we can calculate a new intersection and refine the position of the \( P_{SL} \) by the new one if it is included in the patch. After intersection refinement, the homography matrix is recalculated.

4. Experimental results

4.1 Experimental conditions

In our experiments, the World Cup dataset is used as an input image with the ground-truth. This dataset includes 186 test images, 209 training images, and ground-truth of the homography matrix from 20 soccer scenes. We use 33 images from test images as input images which contain the “penalty area” because the proposed method needs more than four points to estimate a homography matrix. Intersection over Union (IoU) is defined as

\[
\text{IoU} = \frac{\text{Area}(\text{intersection})}{\text{Area}(\text{union})}
\]

The bold type indicates the best results.

Table 3 Comparison of each calibration method on the number of high IoU and mean IoU images.

| method        | High IoU images | mIoU |
|---------------|-----------------|------|
| Yao et al. [12] | 2/33            | -    |
| Chen et al. [12] | 25/33          | 0.973|
| Our method    | 26/33           | 0.975|

is realized by Chen’s method while it fails with Yao’s and our proposed method since the latter methods require at least four intersection points. This limitation is relaxed for our target application, since the image is assumed to always contain the penalty area.

The parameters, shown in Table 1, are common and derived for the 33 images, experimentally. These parameters, however, need to be adjusted by the user for each soccer video, because, for example, the color of the field is different for each stadium and country.

4.2 Objective evaluation

Figure 3 shows the mechanism of IoU evaluation. The IoU value is calculated from the projection results, which are obtained by projecting an input image to the field template using \( H^{-1} \) and \( H^{GT}_{-1} \). \( H \) and \( H^{GT} \) denote an estimated and ground-truth homography matrix. The IoU is defined as

\[
\text{IoU} = \frac{C(\text{AND}(I^{GT}_{\text{pred}},I^{GT}_{\text{pred}})))}{C(\text{OR}(I^{GT}_{\text{pred}},I^{GT}_{\text{pred}})))}
\]

where \( I^{GT}_{\text{pred}} \) and \( I^{GT} \) denote the projection results using \( H^{-1} \) and \( H^{GT}_{-1} \). \( \text{OR}(a,b) \) is the function of the logical sum between image \( a \) and \( b \).

When IoU is greater than 0.95, we define it as a high IoU image. Figure 4 shows a high IoU image (IoU \( \geq 0.95 \)) and a low IoU image (IoU < 0.95). When
IoU is greater than or equal to 0.95, the displacement between real field lines and the projected result is significantly small. Hence, it is defined as the high IoU image. Table 3 shows the number of high IoU images and mean IoU (mIoU) of each method. mIoU is calculated from 25 common high IoU images by the proposed method and that of Chen et al., respectively. Table 3 shows that the number of high IoU images of Yao et al. is extremely low. The reason is that Yao’s method needs to adjust the parameters for each input image. Our proposed method provides a slightly better performance on both criteria despite no learning. The maximum IoU improvement by our method reaches 0.03, which can be seen in the subjective quality improvement described in section 4.3. Large improvements are noticeable for zoomed out cases.

An evaluation of the running time is omitted in this paper because the proposed method obviously requires a longer running time than that by Yao’s method due to the large number of intersection pairs. It is noted that running time of the proposed method will become practical in the future by GPU implementation because the most time consuming part is suitable for parallel processing although the current CPU implementation does not draw this advantage.

### 4.3 Subjective evaluation

Figure 5 shows the projection results. The top and bottom rows of each method show different projection results that are obtained by projecting the field template to an input image (T2I) and an input image to the field template (I2T), respectively. In Yao et al. and our method, the red points in T2I indicate selected intersections. The light blue lines denote projected lines by the estimated homography matrix.

Figure 6 shows over-detection results of the real line extraction, which results in the fault of camera calibration. This is because many white pixels other than the field line in the field image result in false white line detection since the proposed method utilizes pixel color for white line extraction.

Subjective quality is confirmed based on an overlap of field lines of an input image and the field model that is denoted by red lines. The displacement by Yao et al. is larger than that by others. Our scheme can maintain smaller displacement compared to Chen’s scheme. In Table 3 and Fig. 5, even though the difference of mIoU between Chen et al. and our method is small, the different displacement is observed in the projection result.

When IoU is higher than 0.9, it is difficult to identify the subjective difference in I2T by the comparison of real field lines and the field template. A comparison of T2Is is important to evaluate calibration accuracy in order to analyze the small displacement. Although the IoU results of the proposed method and Chen et al. are comparable, the displacement of T2Is by the proposed method is smaller than that by Chen et al. The projection result by the proposed method is the highest performance on subjective evaluation. It is noted that the required accuracy is two or three pixels in the input image (T2I image). Here, both the input image (T2I image) and the free-viewpoint video are supposed to have the same resolutions. Therefore, the generated free-viewpoint video has less than two or three pixels’ displacement of the player’s position, which is negligible in the video representation when both actual and virtual cameras have the same distance from the player’s position.

### 4.4 Evaluation of the line selection and intersection refinement

In this section, we further discuss the performance of the proposed intersection refinement described in section 3.5. Figure 7 shows the line detection result by LSD (left) and the line selection result (right). The colored line in the LSD result includes undesirable lines, which are obtained by something like the object boundary, aside from the real field lines. They are yellow lines showing the longest line in the selected lines. When we select non-field lines in an input image, calibration accuracy becomes inferior. Our proposed the line selection selects real field lines from the detected line by the line selection algorithm. Furthermore, each line is selected from different field lines, which avoids calculating the intersection at similar positions.

Table 4 shows the objective evaluation results of our calibration method. The result is compared to the method without intersection refinement. Although mIoU performance is the same, the number of high IoU images is increased by IR. Therefore, IR can improve the accuracy for low IoU images. The IR provides better performance on both criteria. Figure 8 shows subjective results of the calibration method with intersection refinement. The top image is an example of suc-

### Table 4  Objective evaluation of intersection refinement (IR). mIoU is calculated by 18 high IoU images.

| method  | High IoU images | mIoU |
|---------|----------------|------|
| Without IR | 18/33          | 0.974|
| With IR   | 26/33          | 0.974|
The projection results by (a) Yao et al., (b) Chen et al., and (c) our method. In (a), (b), and (c), the upper and lower rows show projecting the field template to an input image (T2I) and an input image to the field template (I2T), respectively. The red points are shown as selected intersection points. The light blue lines denote projected lines by an estimated homography matrix.

The overdetection result of real line extraction (left) and failure result of camera calibration (right).

Line detection (left) and selection results (right). In the line detection result, each color line shows the detected lines by LSD. The pink and yellow lines of the right-side result show the selected line by our proposed method. Selected lines by the proposed line selection method are selected mostly from real lines.

The IR reduces the displacement between projected lines and real field lines. As a result, IR improves the calibration performance. This improvement is observed in other input images. On the other hand, the projection result without IR is a
failure. When intersections have the displacement, the projected lines also have the displacement. Although our proposed method is performed without parameter trainings, our experimental results show competitive performance compared with Chen’s learning-based method.

5. Conclusion

In this paper, we proposed a calibration method with flexible intersection selection and intersection refinement. The proposed method can select more effective intersection pairs than that by the calibration method with probabilistic hough transform. Intersection refinement improves the calibration performance because refined intersections are located on the real intersection of the input image. As a result, the performance of our proposed method showed higher accuracy and smaller displacement than that by the state-of-the-art method.

The proposed method has a problem in that some input images are evaluated as being less than 0.7 of IoU because the real line extraction extracts lines other than real lines. Accordingly, it fails to evaluate a homography matrix. As future work, we intend to improve the real line extraction so that only real lines are extracted. Furthermore, we intend to implement our proposed method in GPU to reduce the running time.

References

1) D. Farin, S. Krabbe, W. Effelsberg, and P.H.N. de With, “Robust camera Calibration for Sport Videos using Court Models,” SPIE Storage and Retrieval Methods and Applications for Multimedia 2004, Vol. 5307, pp. 80–91, 2004.
2) A. Ekin and A.M. Tekalp “Automatic soccer video analysis and summarization,” IEEE Trans. on Image Processing, Vol.12, No.7, pp.796–807, 2003.
3) S. Takahashi and M. Haseyama, “A method of Important Player Extraction Based on Link Analysis in Soccer Videos,” ITE Trans. on MTA, Vol.5, No.2, pp.42–48, 2017.
4) Q. Yao, A. Kubota, K. Kawakita, K. Nonaka, H. Sankoh, and S. Naito, “Fast camera self-calibration for synthesizing Free Viewpoint soccer Video,” 2017 IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 1612–1616, 2017.
5) Z. Zhang, “A Flexible New Technique for Camera Calibration,” IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 22, No.11, pp.1330–1334, 2000.
6) Z. Zhang “Camera Calibration with One-Dimensional Object,” IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol.26, No.7, pp.892–899, 2004.
7) L. Wang, F.C. Wu, and Z.Y. Hu “Multi-Camera Calibration with One-Dimensional Object under General Motions,” IEEE International Conference on Computer Vision, 2007.
8) G.A. Tomas, “Real-Time Camera Calibration using Sports Pitch Markings,” Journal of Real-Time Image Processing, Vol.2, No.2–3, pp.117–132, 2007.
9) H. Sankoh, S. Naito, M. Harada, T. Sakata, and M. Minoh, “Free-viewpoint Video Synthesis for Sports Scenes Captured with a Single Moving Camera,” ITE Trans. on MTA, Vol.3, No.1, pp.48–57, 2015.
10) N. Homayounifar, S. Fidler, and R. Urtasun, “Sports Field localization via Deep Structured Models,” 2017 IEEE Conference on Computer Vision and Pattern Recognition, pp.5212–5220, 2017.
11) R.A. Sharma, B. Bhat, V. Gandhi, and C.V. Jawahar, “Automated Top View Registration of Broadcast Football Videos,” 2018 IEEE Winter Conference on Applications of Computer Vision, 2018.
12) J. Chen and J.J. Little “Sports Camera Calibration via Synthetic Data,” 2019 IEEE/ICVF Conference on Computer Vision and Pattern Recognition Workshops, 2019.
13) G. Sudhir, J.C.M. Lee, and A.K. Jain, “Automatic classification of tennis video for high-level content-based retrieval,” IEEE International Workshop on Content-Based Access of Image and Video Database, 1998.
14) R.G.V. Gioi, J. Jakubowicz, J.M. Morel, and G. Randall, “LSD: a Line Segment Detector,” Image Processing On Line, pp.35–55, 2012.
15) N. Kiryati, Y. Elder, and A. M. Bruckstein, “A probabilistic Hough transform,” Pattern Recognition, Vol 24, No.4, pp.303–316, 1991.
16) Q. Yao, H. Sankoh, K. Nonaka, and S. Naito, “Automatic Camera Self-Calibration for Immersive Navigation of Free Viewpoint Sports Video,” IEEE International Conference on Multimedia Signal Processing, 2016.
17) C. Tomasi and R. Manduchi, “Bilateral Filtering for Gray and Color Image,” IEEE International Conference on Computer Vision, pp.839–846, 1998.
18) T.Y. Zhang and C.Y. Suen, “A Fast Parallel Algorithm for Thinning Digital Patterns,” Communications of the ACM, pp.236–239, 1984.
Tomoaki Konno received B.E. and M.E. degrees from Tohoku University in 2007 and 2009, respectively. He joined KDDI Corporation in 2009. He is currently a research engineer of the Ultra-Realistic Communications Laboratory in KDDI Research, Inc., Japan. His research interests include free viewpoint video and virtual reality. He is mainly engaged in system architecture design and development in free viewpoint video.

Sei Naito received B.E., M.E., and Ph.D degrees from Waseda University in 1994, 1996, and 2006, respectively. He joined Kokusai Densin Denwa Corporation (currently KDDI Corporation) in 1996. He is currently an executive director in charge of the Media ICT division of KDDI Research Inc., Japan.