Pan-tilt-zoom SLAM for Sports Videos

Jikai Lu[1,2]
lujikai@zju.edu.cn
Jianhui Chen[1]
jhchen14@cs.ubc.ca
James J. Little[1]
little@cs.ubc.ca

[1] Department of Computer Science, University of British Columbia, Vancouver, Canada
[2] College of Computer Science and Technology, Zhejiang University, Hangzhou, China

Abstract

We present an online SLAM system specifically designed to track pan-tilt-zoom (PTZ) cameras in highly dynamic sports such as basketball and soccer games. In these games, PTZ cameras rotate very fast and players cover large image areas. To overcome these challenges, we propose to use a novel camera model for tracking and to use rays as landmarks in mapping. Rays overcome the missing depth in pure-rotation cameras. We also develop an online pan-tilt forest for mapping and introduce moving objects (players) detection to mitigate negative impacts from foreground objects. We test our method on both synthetic and real datasets. The experimental results show the superior performance of our method over previous methods for online PTZ camera pose estimation.

1 Introduction

Visual simultaneous localization and mapping (SLAM) estimates camera poses and environment (*e.g.* landmarks) using stream videos [1, 11, 27, 46], usually assuming that the camera is calibrated and has general movement. While a variety of approaches have been published [2, 10, 13, 39, 44, 45], very few SLAM methods [23] have been explicitly designed for pan-tilt-zoom (PTZ) cameras, which have fixed locations but change pan, tilt angles and zoom levels. These cameras provide detailed views of dynamic scenes and are widely used in sports broadcasting. A robust pan-tilt-zoom SLAM system is beneficial for object tracking and autonomous broadcasting [5, 19, 28, 41]. These real applications motivate our work.

Our work closely relates to panoramic SLAM systems [9, 29]. These systems assume rotation-only camera motion and track the camera in three degrees of freedom (DoF). Because no parallax is observed, a robust estimation of depth is almost impossible. Researchers have developed many methods to overcome this problem. For example, Pirchheim and Reitmayr [29] use plane-induced homographies to represent the cameras between keyframes. Possegger *et al.* [31] use cylindrical panoramic images to evaluate trajectories of moving objects. Unlike previous methods, we propose to represent the dynamic part of a PTZ camera using pan, tilt angles and focal length. First, this representation has the smallest number of degrees of freedom. As a result, the estimation is more robust than over-parameterized methods when outliers exist. Second, the estimation is practically useful for robotic PTZ

[1] The majority of work was performed when Jikai Lu visited UBC as a Mitacs intern.

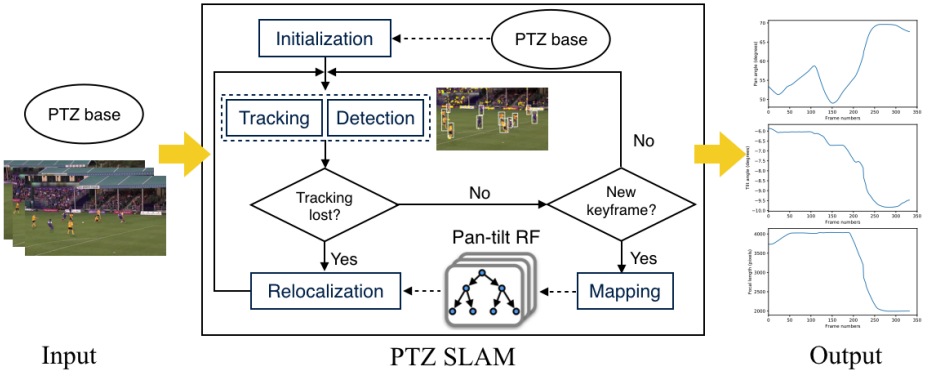

Figure 1: System overview. Given a PTZ base and the first camera pose, our system outputs the camera's pan, tilt and focal length for sports videos. Inside the system, we track camera poses and detect players. We also maintain a map by dynamically adding keyframes to a pan-tilt random forest (RF). When the camera tracking is lost, we relocalize the camera using the pan-tilt forest.

cameras which usually have three degrees of control signals (speed of pan, tilt angles and zoom levels).

Sports videos are challenging for robust camera tracking in long sequences. Sports cameras move very fast so that images can be severely blurred, causing difficulty for feature matching/tracking. Also, moving objects (*e.g.* the players) occlude static objects in the background and inject many outliers for feature tracking. In these situations, sports camera tracking is more challenging than surveillance cameras tracking in which cameras usually do not have very rapid rotations.

To overcome these challenges, we propose to use a novel camera model [7] for tracking, extending the two-point method [7] from a single frame to multiple frames. When camera location and base rotation are known, the two-point method provides the minimum-configuration solution for PTZ cameras. It was proved to be more accurate than homography-based methods. To deal with players, we also integrate player detection results with the SLAM system by discarding keypoints that are on player bodies.

Figure 1 shows the system overview. The system assumes that the camera location and the first-frame camera pose are known. The system outputs a sequence of camera poses (pan, tilt and focal length) from image streams. In the system, the *detection* component detects player bounding boxes using an off-the-shelf object detection method [34]. The *tracking* component estimates the optimal solution for pan, tilt and focal length based on the extended Kalman filter (EKF) [11]. When the tracking is lost, the *relocalization* component recovers the camera pose using an online random forest which is gradually updated by the *mapping* component.

Our main contributions are: (1) develop a PTZ SLAM system by extending the two-point method [7] to sequential data and integrating player detection; (2) demonstrate the superior performance of the system on both synthetic and real datasets compared with strong baselines.

2 Related Work

Image mosaicing and panoramic SLAM: Image mosaicing estimates the relative pose of the camera when each image is captured [3, 24]. To handle rotation-only camera motion, panoramic SLAMs [15, 30] use local panoramic images for mapping. For example, Lisanti *et al*. [23] developed a PTZ camera tracking system. They first build an offline map using multi-scale reference frames. In the online process, the camera parameters are initialized by hardware (camera actuator) and are optimized by 2D image feature matching. They use 2D coordinates in the reference frames as landmarks. Our method and [23] are designed for sports and surveillance applications, respectively. Our method is significantly different from [23] in two points. First, we build an online map and use one reference frame to initialize the system. Second, we use rays as landmarks for tracking and mapping, which facilitates the EKF SLAM system.

Combining object detection with SLAM systems is an emerging trend because of the fast development of object detection [38, 43, 47]. For example, Sun *et al*. [38] developed a motion removal approach that filters out moving objects. Zhong *et al*. [47] developed a robotic vision system that integrates SLAM systems with an object detector to make the two functions mutually beneficial. Our method follows this trend and uses an off-the-shelf object detector [34] without training.

PTZ camera calibration: PTZ camera calibration is a fundamental task for surveillance systems and sports applications [6, 12, 33, 40]. Previous methods [16, 32, 37] first manually annotate several reference images. Then, they calibrate other images by finding correspondences between a testing image and the reference images. Reference-frame based methods are straightforward but lack of temporal coherence. As a result, they fail when fewer correspondences are available (*e.g*. in blurred images).

Researchers proposed different camera parameterization methods for PTZ calibration. For example, Hayet *et al*. [17] first estimate a homography matrix with the reference coordinates. Then, they decompose the pan-tilt-zoom parameters from the estimated homography matrix. The camera model is essential for sequential calibration. Recently, Chen *et al*. [7] proposed a novel PTZ camera model. They decompose the projection matrix into two parts. The first part is the camera location and the base (*e.g*. tripod) rotation. This part is fixed and can be estimated by pre-processing. The second part is the pan, tilt and focal length that dynamically change over time. This camera model shows superior performance in single frame calibration. Our work extends [7] from single image calibration to multiple (sequential) image calibration and tracking.

3 Method

We propose a pan-tilt-zoom SLAM system based on the extended Kalman filter and visual feature tracking. The input of our system is the camera location, the tripod base rotation and the camera pose of the first frame. The output is the camera poses of the image sequence.

3.1 Preliminaries

EKF-SLAM [11] uses a probabilistic feature-based map that represents the current estimates of the camera and all landmarks with uncertainty. The map is represented by a state vector $\hat{\mathbf{x}}$ and a covariance matrix Σ. The state vector $\hat{\mathbf{x}}$ is the stacked state estimates of the camera and landmarks $\hat{\mathbf{x}} = [\hat{\mathbf{x}}_v, \hat{\mathbf{y}}_1, \hat{\mathbf{y}}_2, ...]^\mathsf{T}$. The covariance matrix Σ is the uncertainty of these estimates.

EKF-SLAM has three basic components of *tracking*, *mapping* and *relocalization*. In tracking, the system first uses velocity models to predict camera poses and the projection of landmarks in the image. Then, it tracks point features from the previous frame to the current frame and updates the camera pose and landmarks. In mapping, the system selects frames that have good tracking quality and different camera poses as keyframes. Then, the camera poses and landmarks of keyframes are optimized using bundle adjustment. Finally, the landmarks are memorized as a global map which can be in the form of keyframes, point clouds and neural networks. The relocalization component estimates the camera pose of a lost frame using the map. It either matches the current image against keyframes with known poses, or establishes 2D-to-3D correspondences between keypoints in the current image and landmarks in the scene to estimate camera poses. These three components work together to estimate camera poses robustly.

3.2 Camera Model and Ray Landmark

We use a PTZ camera model from [7]:

$$\mathtt{P} = \underbrace{\mathtt{K}\mathtt{Q}_\phi\mathtt{Q}_\theta}_{\text{PTZ}}\underbrace{\mathtt{S}[\mathtt{I}|-\mathbf{C}]}_{\text{prior}}, \tag{1}$$

in which $\mathbf{C}$ is the camera location in the world coordinate and $\mathtt{S}$ is the base rotation with three degrees of freedom. $\mathtt{Q}_\theta$ and $\mathtt{Q}_\phi$ are rotation matrices for pan and tilt angles. $\mathtt{K}$ is the camera matrix in which the focal length f is the only free parameter because we assume square pixels, a principal point at the image center and no lens distortion.

There are three coordinates involved with this projection matrix: the world coordinate, the tripod coordinate and the image coordinate (see Figure 2). First, world points are translated by $\mathbf{C}$ and rotated by $\mathtt{S}$ to the tripod coordinate. Because the tripod is fixed in practice, this *prior* part $\mathtt{S}[\mathtt{I}|-\mathbf{C}]$ can be pre-estimated. Second, tripod points are rotated and projected to the image coordinate by the *PTZ* part $\mathtt{K}\mathtt{Q}_\phi\mathtt{Q}_\theta$. In sports cameras, this part changes over time and is the focus of this work. Starting here, we refer to the pan, tilt angles and focal length as *camera pose*.

In PTZ-SLAM, the camera state $\hat{\mathbf{x}}_v$ comprises pan, tilt and focal length and their velocities $\hat{\mathbf{x}}_v = [\theta, \phi, f, \delta_\theta, \delta_\phi, \delta_f]^\mathsf{T}$. A landmark state $\mathbf{r}_i = [\theta_i, \phi_i]^\mathsf{T}$ is a ray in tripod coordinates. The projection of a ray $\mathbf{r}_i$ to an image is:

$$\mathtt{P}_{ptz}(\mathbf{r}_i) = \mathtt{K}\mathtt{Q}_\phi\mathtt{Q}_\theta \begin{bmatrix} \tan(\theta_i) \\ -\tan(\phi_i)\sqrt{\tan^2(\theta_i)+1} \\ 1 \end{bmatrix}, \tag{2}$$

in which we represent the ray as a 3D point in the plane of $Z = 1$.

When the camera pose is known, we back-project a pixel location $\mathbf{p}_i = [x_i, y_i]^\mathsf{T}$ to a ray by:

$$\begin{cases} \theta_i = \arctan(\dfrac{X}{Z}) \\ \phi_i = \arctan(-\dfrac{Y}{\sqrt{X^2+Z^2}}) \end{cases}, \tag{3}$$

where:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \mathtt{Q}_\theta^{-1}\mathtt{Q}_\phi^{-1}\mathtt{K}^{-1}\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}. \tag{4}$$

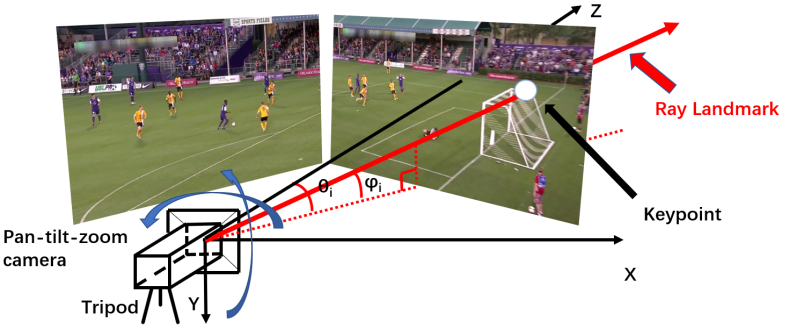

Figure 2: The coordinate system and ray landmarks. The camera pose is represented by pan, tilt and focal length in the tripod coordinate. We use a ray (red line) to represent a landmark in the scene. Best viewed in color.

This parameterization has the minimum number of degrees of freedom. We will show that our parameterization outperforms other methods such as homography-based methods in tracking. Moreover, our method can take advantage of multi-frame pixel-to-ray correspondences in the relocalization process and thus is more robust than keyframe-based methods.

3.3 Online Pan-tilt Forest

We develop an online pan-tilt forest method as the global map for camera relocalization. Unlike previous use of pan-tilt forests [7], our method dynamically updates the forests with new keyframes, greatly speeding up the mapping process. We first introduce pan-tilt forests [7] in brief. Then, we describe the online learning process of the random forest.

A pan-tilt forest is a regression model that predicts the landmark by its appearance in the image:

$$\hat{\mathbf{r}} = h(\mathbf{v}), \tag{5}$$

where $\mathbf{v}$ is the feature vector (*e.g.* SIFT descriptor) that describes the image patch. In training, $\{(\mathbf{v},\mathbf{r})\}$ are paired training examples. In testing, the ray $\hat{\mathbf{r}}$ is predicted by the learned regressor $h(\cdot)$.

A random forest is an ensemble of randomized decision trees. In a tree, the n^{th} internal node splits examples S_n to sub-trees by maximizing of the information gain:

$$E(S_n,\pi_n) = \sum_{(\mathbf{x},\mathbf{r})\in S_n} (\mathbf{r}-\bar{\mathbf{r}})^2 - \sum_{j\in L,R} \Big(\sum_{(\mathbf{x},\mathbf{r})\in S_n^j} (\mathbf{r}-\bar{\mathbf{r}})^2 \Big), \tag{6}$$

where $\bar{\mathbf{r}}$ indicates the mean value of $\mathbf{r}$ for all training samples reaching the node, π_n is the learned tree parameters. Note that left and right subsets S_n^j are implicitly conditioned on the parameter π_n. Here we omit the subscript $\mathbf{p}$ of $\mathbf{r_p}$ and $\mathbf{x_p}$ for notation convenience. The optimization encourages examples that have similar ray angles to be in same sub-trees.

Training terminates when a node reaches a maximum depth of D or contains too few examples. In a leaf node, we save the mean values of samples reaching this leaf node $(\bar{\mathbf{v}}_l, \bar{\mathbf{r}}_l)$. During testing, a sample traverses a tree from the root node to a leaf node. The leaf outputs the $\bar{\mathbf{r}}_l$ as the prediction. As a result, a random forest produces multiple ray candidates for a single pixel location. We keep all candidates and choose the one that minimizes the reprojection error in the camera pose optimization process.

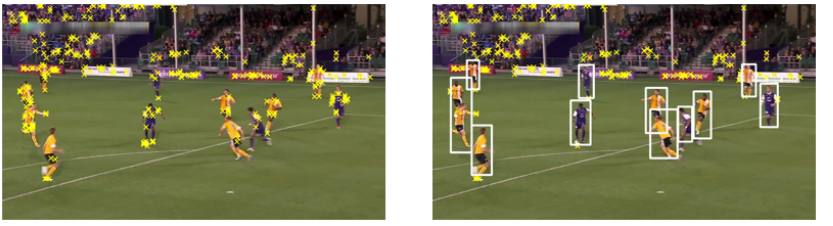

Figure 3: Player detection. Left: keypoints without player detection; right: keypoints with player detection. Best viewed in color.

Online learning: During the tracking, keyframes become available in sequential order. We adapt the online random forest method from [35] for the training. We assume a forest $h_t(.)$ is available, for example, by training on previous t keyframes. We develop two ways to add new examples S_{t+1} to the forest: update an existing tree or add a new tree. First, we estimate the ratio of correct predictions on S_{t+1} using $h_t(.)$. The correctness is measured by thresholding the angular errors (0.1°). When the correctness is higher than a threshold (50%), we update one randomly selected decision tree in $h_t(.)$ by adding new examples S_{t+1}. Otherwise, we add a new tree which is trained on S_{t+1} and a random set of examples from previous keyframes. We find the online training decreases the training time without decreasing the prediction accuracy.

Camera pose optimization: From the pan-tilt forest, we obtain a set of pixel locations and predicted ray pairs $\{(\mathbf{p},\hat{\mathbf{r}})\}$. The camera pose optimization is to minimize the re-projection error:

$$\{\theta,\phi,f\} = \arg\min_{\mathtt{P}} \sum_i \|\mathbf{p}_i - \mathtt{P}(\hat{\mathbf{r}}_i)\|^2. \tag{7}$$

Because the predictions from the pan-tilt forest may still have large errors, we use the RANSAC method [14] to remove outliers. Moreover, we use the two-point method [7] to estimate the initial camera pose inside of the RANSAC method.

3.4 Player Detection

We integrate player detection with the SLAM system. First, we detect objects using a pre-trained Faster R-CNN network [34] on PASCAL VOC dataset. Then, we extract bounding boxes of the "person" category and remove the keypoints that are inside of the bounding boxes in tracking/mapping. As a result, we remove most of keypoints that are on player bodies while only sacrifice a few background keypoints. Figure 3 shows an example of keypoints with/without player detection.

4 Experiments

We conducted experiments on one synthetic dataset and two real datasets.

Basketball dataset: This dataset has one image sequence (1280 × 720 at 60 fps) from a high school basketball match [5]. The sequence length is 3,600 frames and the pan angle range is about $[-30^o, 20^o]$. This dataset is challenging in terms of dynamic objects, fast camera motion and repeated background objects.

Soccer dataset: This dataset has one image sequence (1280 × 720 at 30 fps) from a United Soccer League (USL) soccer game [25]. The sequence length is 333 and the pan

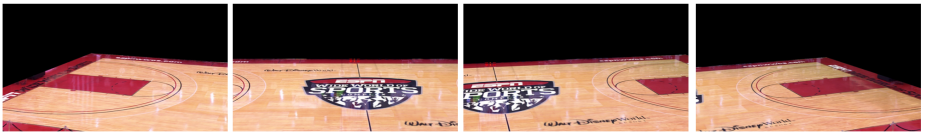

Figure 4: Synthetic image examples.

Sequence		Reprojection error (pix.)					
		EKF-H			EKF-PTZ (ours)		
Seq. ID	Velocity	Mean	Median	Max	Mean	Median	Max
1	0.02	0.1	0.1	0.1	0.1	0.1	0.2
2	0.83	0.4	0.4	0.7	0.3	0.1	1.1
3	0.70	1.0	0.3	16.0	0.3	0.3	0.5
4	0.08	2.1	2.2	3.9	0.7	0.7	1.3

Table 1: EKF-H vs. our method for camera tracking. The (mean angular) velocity (degrees per second) is measured from ground truth.

Outlier	Correctness (%)	
(%)	Keyframe	RF (ours)
10	98.0	100
20	89.3	100
30	79.0	100
40	67.7	100
50	47.0	99.7

Table 2: Keyframe-based method vs. ours for relocalization.

angle range is about $[50^o, 70^o]$. This dataset is challenging in terms of texture-less playing ground, fast camera motion and narrow field-of-view. Moreover, the focal length dramatically changes from 4,000 pixels to 2,000 pixels in about 2 seconds.

On both datasets, the ground truth is manually annotated at each frame independently. The reprojection error is about 1.5 and 2 pixels for the basketball dataset and the soccer dataset, respectively. The PTZ base parameters are pre-estimated and fixed in the experiment.

Error metric: For our method, we report the mean and standard deviation of errors for pan, tilt and focal length. We also visualize estimated camera trajectories to evaluate the smoothness of the results.

We compare our method with four baselines. For a fair comparison, player detection results are also used in all baselines except for baseline 1.

Baseline 1: We remove the player detection component in our system and keep the remaining settings the same.

Baseline 2: We adapt a 3D EKF-based SLAM method [11] to the PTZ camera model. The method only tracks playing-ground landmarks at $Z = 0$. The non-playing-ground points are not used as we cannot estimate their 3D locations from a pure-rotation camera.

Baseline 3 (EKF-H): We implement an online version of the method [23] which uses an off-line mapping for PTZ camera tracking. In this baseline, all the landmarks are located on the plane (the mosaic plane in [23]) that is extended from the first frame. In tracking, we first use RANSAC to estimate the homography from the current frame to the mosaic plane. Then, EKF is used to update homography and landmarks based on observations. We denote this method by EKF-H because it essentially replaces our camera model by a homography.

Baseline 4 (off-line): We implement a reference frame based method [5]. We manually select five reference frames that cover the whole scene. In testing, each frame finds the most-similar frame from the reference frames with SIFT feature matching. Then, the camera pose is estimated by frame-to-frame keypoint matching. This baseline is off-line as it needs a set of references with annotated camera poses before tracking.

4.1 Synthetic Data Experiments

We conducted experiments on a synthetic basketball dataset to verify the robustness of our method. Specially, we want to show that our camera model (Section 3.2) and pan-tilt forest

Method	Basketball			Soccer		
	Pan (°)	Tilt (°)	FL (pix.)	Pan (°)	Tilt (°)	FL (pix.)
Baseline 1	0.50 ± 0.46	0.02 ± 0.02	31.95 ± 25.37	0.28 ± 0.32	0.10 ± 0.11	110.58 ± 62.66
Baseline 2	0.48 ± 0.41	0.05 ± 0.03	38.44 ± 23.58	-	-	-
Baseline 3	3.63 ± 3.03	0.34 ± 0.27	395.40 ± 303.88	0.74 ± 0.99	0.13 ± 0.13	179.33 ± 150.45
Baseline 4 (Off-line)	0.10 ± 0.61	0.06 ± 1.31	16.63 ± 96.65	0.04 ± 0.05	0.02 ± 0.02	36.01 ± 41.46
Our method	0.17 ± 0.14	0.05 ± 0.03	19.28 ± 18.68	0.08 ± 0.07	0.08 ± 0.07	63.70 ± 61.66

Table 3: Quantitative results. The metric is mean absolute errors compared with the human annotation. FL is short for focal length. Although baseline 4 is slightly better than ours, it needs an off-line initialization and has much larger maximum errors on basketball dataset (details in the text).

(Section 3.3) are superior to homography-based and keyframe-based methods, respectively. First, we use multiple (about 30) calibrated images to generate a top-view image of the playing ground with detailed textures. Then, we synthesize sequences of images by warping the court image using pre-defined camera poses. Finally, we run our method on the generated images. Figure 4 shows four image examples from the synthetic sequences.

We test the camera tracking accuracy on four sequences. Each sequence has 600 frames (10 seconds) and has various angular velocities. We run our method and baseline 3 (EKF-H) on these sequences. In testing, we turn off the relocalization of both methods. Then, we report the reprojection errors which are computed by projecting rays to images using ground truth cameras and estimated cameras. Table 1 shows the mean angular velocity (degrees per second) in each sequence and the mean, median and maximum reprojection errors (pixels). Our method achieves lower reprojection errors than the EKF-H method. It demonstrates the advantage of our camera model. Moreover, we found it is hard to tune the parameters in EKF-H (*e.g.* the uncertainty of homography). On the other hand, it is easier to set the uncertainty of pan-tilt-zoom parameters in our method because they are physically meaningful.

We also compare the robustness of our relocalization method with traditional keyframe-based method [45] using a 3,600 frame sequence. The keyframe-based method uses the nearest keyframe to relocalize the lost frame. In the testing, the same keyframe selection method is used to make sure the same set of keyframes are used for both methods. Random outliers are added to the keypoints in both methods. Table 2 shows the percentage of correctly relocalized testing images. An estimated camera is "correct" when it is within 2° angular error of the ground truth. This accuracy is sufficient to restart the tracking system [36]. Our method achieves much higher correctness than the keyframe-based method (*e.g.* 100% vs. 68% when outlier ratio is 40%) because the random forest effectively uses multiple frames in prediction.

4.2 Real Data Experiments

Quantitative results: Table 3 shows the quantitative results of our method and four baselines. On both datasets, our method achieves significantly lower estimation errors than the online baselines except for baseline 4. The result of baseline 1 shows that the player detection part improves the accuracy of our system. Baseline 2 has higher errors on the basketball dataset and is lost on the soccer dataset because many keypoints are not at the playing ground. In baseline 3, more errors are accumulated compared with our method. Baseline 4 is slightly better than our method in terms of mean errors. However, it has much higher maximum errors on the basketball sequence. Its maximum pan error is 15.89° (ours is 0.96°) because baseline 4 does not consider temporal information in the process, making it unsuitable for

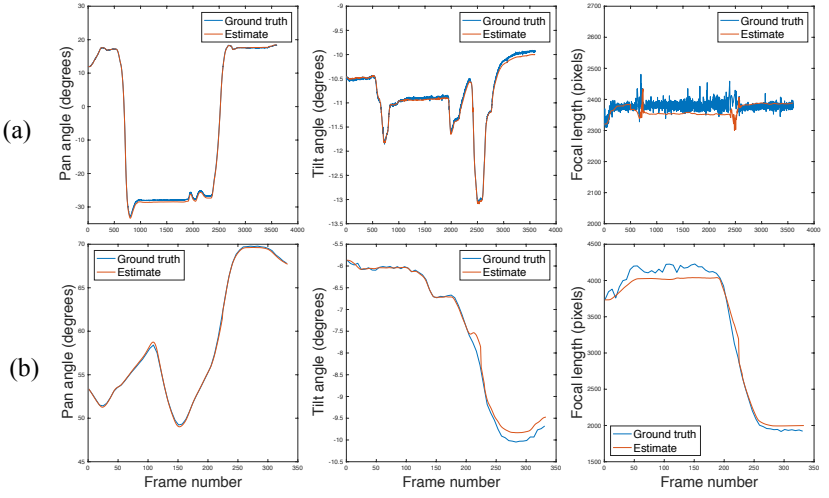

Figure 5: Estimated camera trajectories of our method. (a) basketball; (b) soccer.

applications such as augmented reality which requires smooth camera trajectories.

Qualitative results: Figure 5 shows the estimated pan, tilt and focal length on the basketball and soccer sequences. Our result is very close to the ground truth [1]. Our result is smoother because of the EKF method. In the soccer sequence, camera rotation and the focal length change dramatically in a short time. Our method successfully tracks the camera with small errors. The results demonstrate the robustness of our system for challenging sequences in different sports.

Figure 6 provides a detailed comparison of our method with EKF-H on the basketball pan angle estimation. The camera tracking of EKF-H is lost when the camera quickly moves from one side to the other side of the court (the first semi-transparent green area). After that, the estimated camera has a significant drift from the ground truth. Although EKF-H recovers the camera around frame 2,500 (the second semi-transparent green area), it still has larger errors than our method. On the other hand, our method successfully tracks the camera along the whole sequence. More results are in the supplementary material.

Implementation: Our system is implemented with Python on a 4.20 GHz Intel CPU, 32 GB memory Ubuntu system. The player detection model is running on a TITAN Xp GPU with 12 GB memory. The running speed for our system is not optimized in the current implementation. The speed can be improved by implementing with C++ and using a GPU to compute image features. The code is available online[2].

For keypoint detection and description, we tried DoG + SIFT, FAST + ORB and LATCH [22] + ORB. We found ORB is much faster than SIFT in descriptor computation and matching. However, ORB matching is not stable when cameras rotate rapidly. On the other hand, SIFT is more reliable for rapid camera rotation. It also requires a much fewer number of keypoints to achieve similar accuracy. We finally choose DoG + SIFT in our implementation. In EKF-based tracking, the parameters are as follows: the variance of keypoints location, pan/tilt angles and focal lenth is 0.1 pixels, 0.001° and 1 pixel, respectively.

[1] In the basketball sequence, the range of focal length is small (from about 2,300 pixels to 2,480 pixels). As a result, the plot of the ground truth has some noise because the ground truth camera pose is manually annotated frame by frame.

[2] https://github.com/lulufa390/Pan-tilt-zoom-SLAM

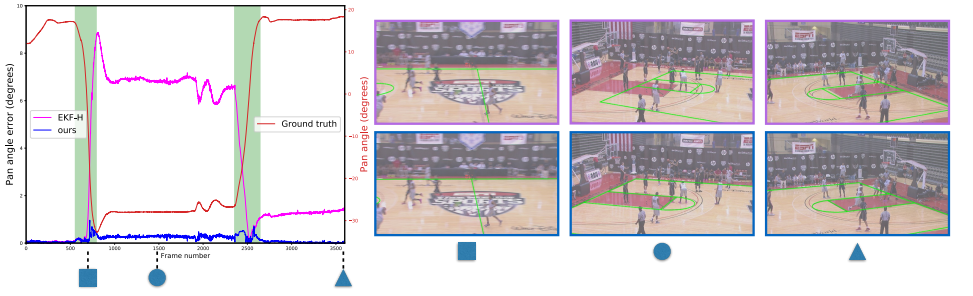

Figure 6: Qualitative comparison with EKF-H. The left figure shows pan angle errors of EKF-H (magenta) and ours (blue). The red line is the ground truth of pan angles that has two fast-moving periods (semi-transparent green areas). EKF-H loses the tracking at the first fast-moving period. Our method successfully tracks the camera. On the right, we visualize camera poses at three frames for both methods (EKF-H is on the top and ours is on the bottom). Best viewed in color.

In relocalization, we tested the nearest neighbor search (NNS) as suggested by reviewers. In training, NNS builds a feature-label database using KD-trees. In testing, the method finds pixel-ray correspondences using the nearest neighbor search in the database, followed by a RANSAC-based camera pose optimization. Although we tried ways to improve the performance, NNS is still not stable on the basketball dataset. One explanation is that the background has repeating patterns that are difficult to distinguish in the feature space. Please note, random forest methods optimize tree parameters using both features and rays. Thus, the learned model is more discriminative than KD-trees. This result agrees with the exceptional performance of random forests in camera relocalization [4, 26, 36]

Discussion: We develop a PTZ SLAM system and successfully apply it to challenging sports videos. However, there are several limitations at the current stage. For example, our system requires the camera pose of the first frame and camera base parameters to initialize the system. Both of them can be estimated by fully automatic methods such as DSM [20] and bundle adjustment [42], respectively. Also, the experiments on real data are limited because there are no large public datasets available. We mitigate this limitation by careful and dense testing on a synthetic dataset.

5 Conclusion

In this work, we developed a robust pan-tilt-zoom SLAM system that can track fast-moving cameras in sports videos. We use a novel camera model for tracking, an online random forest for mapping and player detection to deal with moving foreground objects. The experimental results show that our method is more robust than homography-based methods. In the future, we would like to speed up our system and test it on more image sequences. Adding semantic segmentation [8, 18, 21] and exploring other features (*e.g.* lines and circles) are two promising directions.

Acknowledgements This work was partially supported by grants from NSERC. We thank the anonymous reviewers for insightful suggestions.

References

[1] Michael Bloesch, Jan Czarnowski, Ronald Clark, Stefan Leutenegger, and Andrew J Davison. CodeSLAM - Learning a compact, optimisable representation for dense visual SLAM. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[2] Guillaume Bresson, Zayed Alsayed, Li Yu, and Sébastien Glaser. Simultaneous localization and mapping: A survey of current trends in autonomous driving. *IEEE Transactions on Intelligent Vehicles*, 20:1–1, 2017.

[3] Matthew Brown and David G Lowe. Automatic panoramic image stitching using invariant features. *International Journal of Computer Vision (IJCV)*, 74(1):59–73, 2007.

[4] Tommaso Cavallari, Luca Bertinetto, Jishnu Mukhoti, Philip Torr, and Stuart Golodetz. Let's take this online: Adapting scene coordinate regression network predictions for online RGB-D camera relocalisation. *arXiv preprint arXiv:1906.08744*, 2019.

[5] Jianhui Chen and Peter Carr. Mimicking human camera operators. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2015.

[6] Jianhui Chen and James J Little. Sports camera calibration via synthetic data. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019.

[7] Jianhui Chen, Fangrui Zhu, and James J Little. A two-point method for PTZ camera calibration in sports. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018.

[8] Anthony Cioppa, Adrien Deliege, Maxime Istasse, Christophe De Vleeschouwer, and Marc Van Droogenbroeck. ARTHuS: Adaptive real-time human segmentation in sports through online distillation. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019.

[9] Javier Civera, Andrew J Davison, Juan A Magallón, and JMM Montiel. Drift-free real-time sequential mosaicing. *International Journal of Computer Vision (IJCV)*, 81(2): 128–137, 2009.

[10] Alejo Concha, Giuseppe Loianno, Vijay Kumar, and Javier Civera. Visual-inertial direct SLAM. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[11] Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, (6):1052–1067, 2007.

[12] Alberto Del Bimbo, Fabrizio Dini, Giuseppe Lisanti, and Federico Pernici. Exploiting distinctive visual landmark maps in pan–tilt–zoom camera networks. *Computer Vision and Image Understanding (CVIU)*, 114(6):611–623, 2010.

[13] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 40(3):611–625, 2018.

[14] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[15] Steffen Gauglitz, Chris Sweeney, Jonathan Ventura, Matthew Turk, and Tobias Höllerer. Live tracking and mapping from both general and rotation-only camera motion. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2012.

[16] Ankur Gupta, James J Little, and Robert J Woodham. Using line and ellipse features for rectification of broadcast hockey video. In *Canadian Conference on Computer and Robot Vision (CRV)*, 2011.

[17] Jean-Bernard Hayet, Justus Piater, and Jacques Verly. Robust incremental rectification of sports video sequences. In *British Machine Vision Conference (BMVC)*, 2004.

[18] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[19] Adrian Hilton, Jean-Yves Guillemaut, Joe Kilner, Oliver Grau, and Graham Thomas. 3D-TV production from conventional cameras for sports broadcast. *IEEE Transactions on Broadcasting*, 57(2):462–476, 2011.

[20] Namdar Homayounfar, Sanja Fidler, and Raquel Urtasun. Sports field localization via deep structured models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[21] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[22] Gil Levi and Tal Hassner. LATCH: learned arrangements of three patch codes. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016.

[23] Giuseppe Lisanti, Iacopo Masi, Federico Pernici, and Alberto Del Bimbo. Continuous localization and mapping of a pan-tilt-zoom camera for wide area tracking. *Machine Vision and Applications (MVA)*, 27(7):1071–1085, 2016.

[24] Steven Lovegrove and Andrew J Davison. Real-time spherical mosaicing using whole image alignment. In *European Conference on Computer Vision (ECCV)*, 2010.

[25] Keyu Lu, Jianhui Chen, James J Little, and He Hangen. Light cascaded convolutional neural networks for accurate player detection. In *British Machine Vision Conference (BMVC)*, 2017.

[26] Lili Meng, Jianhui Chen, Frederick Tung, James Little J., Julien Valentin, and Clarence Silva. Backtracking regression forests for accurate camera relocalization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[27] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31 (5):1147–1163, 2015.

[28] Hemanth Pidaparthy and James Elder. Keep your eye on the puck: Automatic hockey videography. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2019.

[29] Christian Pirchheim and Gerhard Reitmayr. Homography-based planar mapping and tracking for mobile phones. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2011.

[30] Christian Pirchheim, Dieter Schmalstieg, and Gerhard Reitmayr. Handling pure camera rotation in keyframe-based slam. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2013.

[31] Horst Possegger, Matthias Rüther, Sabine Sternig, Thomas Mauthner, Manfred Klopschitz, Peter M Roth, and Horst Bischof. Unsupervised calibration of camera networks and virtual PTZ cameras. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2012.

[32] Jens Puwein, Remo Ziegler, Julia Vogel, and Marc Pollefeys. Robust multi-view camera calibration for wide-baseline camera networks. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2011.

[33] Konstantinos Rematas, Ira Kemelmacher-Shlizerman, Brian Curless, and Steve Seitz. Soccer on your tabletop. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[34] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[35] Amir Saffari, Christian Leistner, Jakob Santner, Martin Godec, and Horst Bischof. Online random forests. In *IEEE International Conference on Computer Vision (ICCV) Workshops*, 2009.

[36] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[37] Sudipta N Sinha and Marc Pollefeys. Pan–tilt–zoom camera calibration and high-resolution mosaic generation. *Computer Vision and Image Understanding (CVIU)*, 103(3):170–183, 2006.

[38] Yuxiang Sun, Ming Liu, and Max Q-H Meng. Improving RGB-D SLAM in dynamic environments: A motion removal approach. *Robotics and Autonomous Systems (RAS)*, 89:110–122, 2017.

[39] Keisuke Tateno, Federico Tombari, Iro Laina, and Nassir Navab. CNN-SLAM: Real-time dense monocular slam with learned depth prediction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[40] Graham Thomas. Real-time camera tracking using sports pitch markings. *Journal of Real-Time Image Processing*, 2(2-3):117–132, 2007.

[41] Graham Thomas, Rikke Gade, Thomas B Moeslund, Peter Carr, and Adrian Hilton. Computer vision for sports: Current applications and research topics. *Computer Vision and Image Understanding (CVIU)*, 159:3–18, 2017.

[42] Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. Bundle adjustment – a modern synthesis. In *International workshop on vision algorithms*, 1999.

[43] Chieh-Chih Wang and Chuck Thorpe. Simultaneous localization and mapping with detection and tracking of moving objects. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2002.

[44] Shichao Yang, Yu Song, Michael Kaess, and Sebastian Scherer. Pop-up SLAM: Semantic monocular plane SLAM for low-texture environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[45] Georges Younes, Daniel Asmar, Elie Shammas, and John Zelek. Keyframe-based monocular SLAM: design, survey, and future directions. *Robotics and Autonomous Systems*, 98:67–88, 2017.

[46] Shuaifeng Zhi, Michael Bloesch, Stefan Leutenegger, and Andrew J Davison. SceneCode: Monocular dense semantic reconstruction using learned encoded scene representations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[47] Fangwei Zhong, Sheng Wang, Ziqi Zhang, and Yizhou Wang. Detect-SLAM: Making object detection and SLAM mutually beneficial. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018.