CubemapSLAM: A Piecewise-Pinhole Monocular Fisheye SLAM System*

Yahui Wang[1], Shaojun Cai[2], Shi-Jie Li[1], Yun Liu[1], Yangyan Guo[3], Tao Li[1], and Ming-Ming Cheng[1]

[1] The College of Computer and Control Engineering, Nankai University, China
[2] UISEE Technology (Beijing) Co., Ltd.
[3] University of Chinese Academy of Sciences, China

Abstract. We present a real-time feature-based SLAM (Simultaneous Localization and Mapping) system for fisheye cameras featured by a large field-of-view (FoV). Large FoV cameras are beneficial for large-scale outdoor SLAM applications, because they increase visual overlap between consecutive frames and capture more pixels belonging to the static parts of the environment. However, current feature-based SLAM systems such as PTAM and ORB-SLAM limit their camera model to pinhole only. To compensate for the vacancy, we propose a novel SLAM system with the cubemap model that utilizes the full FoV without introducing distortion from the fisheye lens, which greatly benefits the feature matching pipeline. In the initialization and point triangulation stages, we adopt a unified vector-based representation to efficiently handle matches across multiple faces, and based on this representation we propose and analyze a novel inlier checking metric. In the optimization stage, we design and test a novel multi-pinhole reprojection error metric that outperforms other metrics by a large margin. We evaluate our system comprehensively on a public dataset as well as a self-collected dataset that contains real-world challenging sequences. The results suggest that our system is more robust and accurate than other feature-based fisheye SLAM approaches. The CubemapSLAM system has been released into the public domain.

Keywords: Omnidirectional Vision · Fisheye SLAM · Cubemap

1 Introduction

SLAM techniques have been widely applied in the robotics and automation industry. Specifically, Visual SLAM (**VSLAM**) is gaining increasing popularity, because cameras are much cheaper than other alternatives such as differential GPS (D-GPS) and LIDAR. However, traditional VSLAM systems suffer from problems such as occlusions, moving objects and drastic turns due to the limited FoV of perspective cameras. In contrast, large FoV cameras significantly increase

* This work is partially supported by the National Natural Science Foundation (61872200), the Natural Science Foundation of Tianjin (17JCQNJC00300) and the National Key Research and Development Program of China (2016YFC0400709).

the visual overlap between consecutive frames. In addition, large FoV cameras capture more information from the environment, therefore making the SLAM system less likely to fail.

However, there are still many challenges in SLAM with large FoV cameras. The first challenge is that most of the widely-used feature descriptors are designed for low-distortion images. Some systems [21, 24, 26] choose more robust features such as SIFT [16] or design new features suited for highly distorted images [1,31], but they are too time-consuming to satisfy the real-time demands of many applications. Others [6,10,13] try to remove distortion effect by directly rectifying fisheye images into pinhole images, but the remaining FoV is much smaller after rectification. Multicol-SLAM [27] adapts ORB-SLAM [17] to operate on the raw distorted images, but the open-source version fails to achieve satisfying results.

In this paper, we redesign the pipeline of ORB-SLAM to fit the piecewise linear camera model that utilizes full FoV without introducing distortion. We thus propose an efficient and compact feature-based SLAM system dedicated to large FoV cameras. Our system achieves better performance than directly rectifying the fisheye image into a pinhole image and the other existing feature-based fisheye SLAM system [27]. Despite the limited angular resolution of a fisheye camera, we achieve comparable accuracy to ORB-SLAM with a pinhole camera while performing much more robustly. Specifically, our work has the following contributions:

1. We propose the first cubemap solution for feature-based fisheye SLAM. The piecewise-pinhole nature of the cubemap model is especially desirable for feature descriptors, and there is no need to retrain Bag-of-Words (**BoW**) [7] vocabulary for fisheye images.
2. In the initialization and point triangulation stages, we adopt a unified vector-based representation which efficiently handles the matches across multiple faces. Based on this representation, we propose a novel and systematic inlier checking metric for RANSAC with essential matrix constraint, and we provide a rigorous analysis of the correctness of this metric.
3. In the optimization stage, we carry out thorough comparisons of different error metrics, and we propose a novel reprojection error metric for the cubemap model that outperforms other metrics.
4. We present an extensive evaluation on public datasets, and a self-collected one containing typical outdoor driving scenarios. We also discover that by carefully choosing the camera mounting position, the problem of a low angular resolution in outdoor scenes mentioned in [30] can be greatly reduced.

1.1 Related Work

The VSLAM techniques have been widely used in various applications such as self-driving cars [14, 22, 32]. However, limited FoV of pinhole camera may cause the localization system to fail when there is little overlap between consecutive frames. Consequently, large FoV cameras are gaining attention. For instance, the

V-Charge project [6, 13] builds a car surrounded with 4 synchronously triggered fisheye cameras modeled as a generalized camera [20]. In recent years, many works have discussed the methods to exploit large FoV cameras. [11] transforms the panoramas from the PointGrey LadyBug camera into cubic panoramas, but they aim to estimate poses of input cubic panoramas rather than build real-time SLAM system. A piecewise-pinhole model is presented in [29], but in the work the map needs to be built offline, and the local and global bundle adjustment as well as the loop closing based on the proposed model are not performed. A number of semi-direct or direct SLAM systems based on fisheye models have been proposed recently [9, 15]. In their works an adapted GPU-based plane-sweep stereo algorithm is used to find matching patches between stereo image from the raw fisheye images. Omni-LSD [3] also proposes a similar pinhole array model as part of the extension to origin LSD-SLAM [4]. While direct method is shown to be robust in scale-diverse environment, its performance in large outdoor environments is still unknown. To our knowledge, Multicol-SLAM [27] is the only existing feature-based fisheye SLAM, but it tries to extract features directly on highly distorted images, which may lead to false matches. Further comparison with MultiCol-SLAM will be presented in the experiment section.

In this work, we propose an efficient and practical cubemap SLAM solution aimed at large-scale outdoor applications. In the following sections, we will first introduce the theoretical adaptions we have made in order to maximally utilize the power of a cubemap model, and then we will demonstrate the advantage of our system in extensive large real-world experiments.

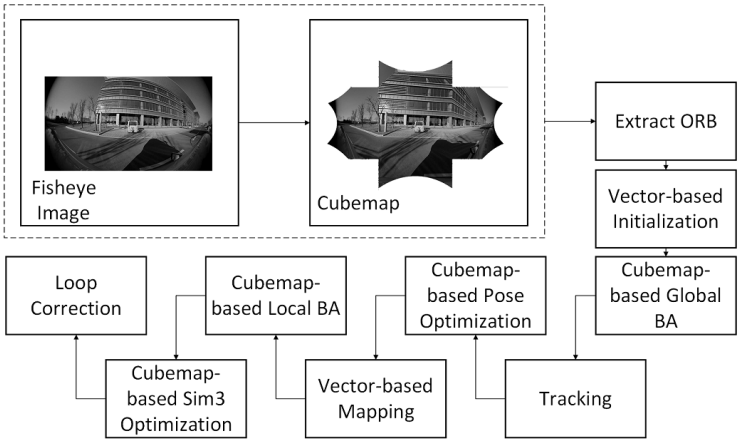


Fig. 1. System overview

2 Algorithm

In this section, we describe the pipeline of the proposed method. As shown in Fig. 1, we acquire cubemap image by calibrating the fisheye camera and mapping fisheye images onto a cube. A vector-based RANSAC is used to solve the essential matrix to recover camera motion and build the initial map. A cubemap-based global bundle adjustment is used to refine camera poses and the initial map. After initialization, the tracking thread estimates camera poses by tracking the local map and refines the poses with a cubemap-based pose optimization algorithm. When the tracking thread decides to insert current frame into the map as a keyframe, a vector-based triangulation algorithm is used to create new map points, and the frame is converted into BoW vectors for loop detection. When a loop is detected, a Sim_3 transformation for loop closing is computed by an adapted Sim_3 optimization algorithm and a loop correction is performed.

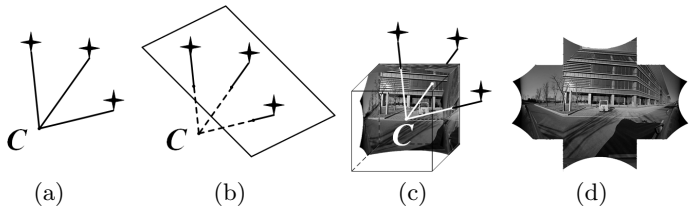


Fig. 2. A demonstration of projecting a fisheye image onto a cube. C are camera centers in the figures. (a) bearing vectors from fisheye image points. (b) project rays onto a single image plane. (c) project rays onto cube. (d) unfolded cubemap image

2.1 Fisheye Camera Model and the Cubemap Model

We choose the omnidirectional camera model from [23] to calibrate the fisheye camera in our work. By calibrating fisheye camera, we acquire a polynomial which transforms image points into bearing vectors as shown in Fig. 2(a). Since the bearing vectors are actually viewing rays, a pinhole image can be acquired by projecting the rays to an image plane with specified camera projection matrix, as in Fig. 2(b). However, projecting on a single pinhole plane would result in a much smaller FoV. To make full use of the large FoV of fisheye camera, we project bearing vectors to multiple image planes. For simplicity, we choose to project onto a cube where each cube face can be seen as an image plane of a virtual pinhole camera with 90° FoV, and the virtual camera shares the same camera parameters as in Fig. 2(c). After projection, we can get a cubemap image as in Fig. 2(d).

2.2 Initialization

Initialization is an important component in SLAM. In perspective camera situation, feature matching followed by RANSAC is used for solving the fundamental matrix F or the homography matrix H between two frames, and camera motion can be recovered afterwards.

Although the F and H models are widely used in SLAM for pinhole cameras, it is not possible to calculate them directly on the distorted fisheye images. Moreover, the F matrix does not exist because the pinhole camera projection matrix K, which is used to derive the F matrix, is undefined for fisheye cameras. To make use of the models, we can either rectify the fisheye images into multiple pinhole images that cover the full FoV, and model each pinhole image separately with F or H models, or transform the image points into bearing vectors through the calibrated fisheye model. For the former approach, we equivalently operate on a multiple pinhole SLAM system as in [5]. However, the inter-pinhole correspondence points have to be transformed to the same coordinate first before they can be handled correctly, which increase the complexity. For the latter approach, the essential matrix model E and H for vectors can be adopted in a vector form, and the intra-pinhole and inter-pinhole correspondences can be handled in a unified framework. In the experiment we find essential matrix model E works for most of the scenarios. Therefore, we represent each measurement as a bearing vector as in [12], and apply essential matrix model E for initialization.

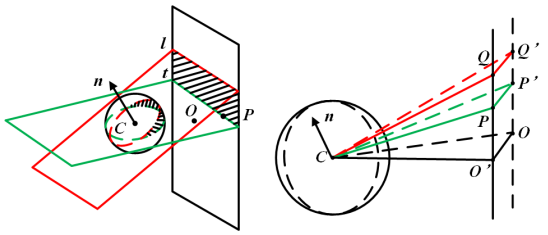


Fig. 3. Threshold of inlier checking in vector-based initialization. The left figure shows the corresponding inlier regions (the shadow areas) between the image plane and the unit sphere. The right figure shows the side view of the left figure.

2.3 Epipolar Constraints on the Unit Sphere

For SLAM systems, epipolar geometry is used to check whether two points are in correspondence when the F matrix is known, or whether the F assumption is correct when the point correspondences are assumed to be right. To achieve an inlier probability of 95%, the following criteria are used for inlier checking,

$$\dot{\mathbf{p}}_2^T F \dot{\mathbf{p}}_1 < 3.84\sigma^2 \tag{1}$$

where $\dot{\mathbf{p}}_1$ and $\dot{\mathbf{p}}_2$ are homogeneous representation of the points p_1 and p_2 on the images, and σ is the variance of the measurement noise (cf. [8]). In Eq. 1, $F\dot{\mathbf{p}}_1$ can be geometrically explained as the epipolar line in the second image where p_2 belongs. Thus a product with $\mathbf{p}_2^T$ yields the distance of point p_2 to the epipolar line. Similarly, the essential matrix constraint can be written as

$$\mathbf{r}_2^T E \mathbf{r}_1 = 0 \tag{2}$$

but $\mathbf{r}_1$ and $\mathbf{r}_2$ are bearing vectors rather than image points on the plane. We noticed that since E can be decomposed as $E = [t]_\times R$, where $E\mathbf{r}_1$ indicates the normal of the epipolar plane, the formula $\mathbf{r}_2^T E \mathbf{r}_1$ can be explained as the signed distance of $\mathbf{r}_2$ to the epipolar plane.

As shown in section 2.1, for each vector, there is a corresponding image point on the cubemap. To find inlier threshold for measurements on the unit sphere, we propose to map the well-defined inlier region on image plane to the unit sphere as in Fig. 3. For simplicity, we only show the process on the front cubemap face. In Fig. 3, the sphere is the unit sphere with point C as camera center, and the plane in black is the front face of cubemap with point O as the center. Line l is the epipolar line from the intersection of the epipolar plane (red) and the image plane (black). $\mathbf{n}$ is the normal of epipolar plane. And we have

$$\mathbf{n} = E\mathbf{r}_1 \tag{3}$$

as we mentioned above. A point P is considered as inlier if the distance to l is within a threshold as Eq. 1 indicates. The area within the threshold is represented by the shadow area with line t as boundary. We assume P is on the boundary line t to reveal the boundary conditions. For convenience we only draw area under l. The area above l can be handled in the same way. The mapped area on unit sphere is also shown in shadow, from which we can see that the corresponding threshold on unit sphere is not uniformly distributed. For area closer to the image plane, the threshold is larger, and for area further from image plane the threshold is smaller. Thus a constant threshold is not reasonable.

To illustrate the geometry relations of the threshold on the image plane and unit sphere, we show the side view of the model in the right figure of Fig. 3. In the figure, OQ' is perpendicular to the epipolar line QQ', and parallel to QO' which passes through P. The plane $OO'QQ'$ corresponds to the image plane in left figure. The line segment CO, which denotes the focal line, is perpendicular to the image plane and thus perpendicular to OQ'. Both $P'Q'$ and PQ indicate the threshold on image plane, thus the arc on unit sphere between QC and CP is the inlier region we demand. For simplicity, we notate $\angle PCO'$ as ϕ, $\angle QCP$ as θ, and length of QP as th, which is usually set to 1 pixel. We observe that:

$$\tan(\phi+\theta) = \frac{\|QO'\|}{\|CO'\|} = \frac{\|Q'O\|}{\|CO'\|} = \frac{\|th\| + \|PO'\|}{\sqrt{\|CO\|^2 + \|OO'\|^2}} \tag{4}$$

$$\tan\phi = \frac{\|PO'\|}{\|CO'\|} = \frac{\|PO'\|}{\sqrt{\|CO\|^2 + \|OO'\|^2}} \tag{5}$$

We notice in Eq. 4 and 5 the length of OO' and PO' are the only unknowns. And $\boldsymbol{OP}$ can be derived immediately since the coordinates of the camera center O and image point P are already known. To calculate the length of OO' and PO', we can first solve the direction vector $\mathbf{e}$ of the epipolar line QQ'. Since QQ' is the intersection of the image plane and epipolar plane, $\mathbf{e}$ can be derived by the cross product of normals of the planes. By making

$$\boldsymbol{z} = \frac{\boldsymbol{CO}}{\|CO\|} = (0, 0, 1)^T \tag{6}$$

as the normal of image plane, the direction vector $\mathbf{e}$ can be derived by:

$$\boldsymbol{e} = \boldsymbol{n} \times \boldsymbol{z} \tag{7}$$

As a result we have:

$$\|OO'\| = \frac{|\boldsymbol{e} \cdot \boldsymbol{OP}|}{\|\boldsymbol{e}\|} \tag{8}$$

$$\|PO'\| = \sqrt{\|OP\|^2 - \|OO'\|^2} \tag{9}$$

We can derive $\tan(\phi + \theta)$ and $\tan\phi$ by substituting Eq. 8 and Eq. 9 into Eq. 4 and Eq. 5, we have:

$$\tan\theta = \frac{\tan(\phi+\theta) - \tan\phi}{1 + \tan(\phi+\theta)\tan\phi}, \sin\theta = \frac{\tan\theta}{\sqrt{\tan^2\theta + 1}} \tag{10}$$

Then from Eq.3 and Eq.2, we get our inlier metric for the unit shpere as:

$$\left| \frac{\mathbf{r}_2^T E \mathbf{r}_1}{\|\mathbf{r}_2\| \, \|E\mathbf{r}_1\|} \right| = \left| \frac{\mathbf{r}_2^T \mathbf{n}}{\|\mathbf{r}_2\| \, \|\mathbf{n}\|} \right| \leq \left| \cos(\frac{\pi}{2} \pm \theta) \right| = |\sin\theta| \tag{11}$$

2.4 Optimization

To perform optimizations in vector-based vision systems, several metrics have been proposed. [12] proposes to minimize the angular error between bearing vectors, and [19] studies different metrics and shows that the tangential error has the best performance. Zhang et al. [30] evaluate the above metrics as well as a vector difference with a semi-direct VO [5]. Inspired by the multi-pinhole nature of cubemap, we propose to minimize reprojection errors of all cube faces as a multi-camera system.

The multi-camera model is used extensively in previous multiple-camera SLAM systems [5, 6, 13, 27]. In the multi-camera model, a body frame B rigidly attached to camera frames is set as the reference frame. Transformations T_{C_iB} from body frame to camera local frames C_i can be obtained by extrinsic calibration, where i represent the camera index. In cubemap model, different faces are equivalent to pinhole cameras as in section 2.1. We set the front-facing virtual

camera as the body frame, and all the pinhole cameras are transformed to the body frame by a rotation R_{C_iB}. The projection model of cubemap is:

$$u = KR_{C_iB}T_{BW}P \tag{12}$$

where $P = (x, y, z)^T$ is the 3D point in world frame, T_{BW} is transformation from world frame to body frame, and u is the local point coordinate in the image coordinate of each cubemap face.

We can represent $T \in SE_3$ with $\xi = (\phi^T, \rho^T)^T$ [2] and expand Eq. 12 into:

$$u = KP_1, P_1 = R_{C_iB}P_2, P_2 = T_{BW}P \tag{13}$$

The Jacobian of the measurement u to camera pose T therefore can be derived according to the chain rule:

$$J_\xi = -\frac{\partial u}{\partial P_1} \cdot R_{C_iB} \cdot [-P_2^\wedge, I_{3\times3}] \tag{14}$$

where $P_2^\wedge$ is the skew-symmetric matrix of P_2. The Jacobian for map point position is given by:

$$J_p = -\frac{\partial u}{\partial P_1} \cdot \frac{\partial P_1}{\partial P} = -\frac{\partial u}{\partial P_1} R_{C_iB} R_{BW} \tag{15}$$

where R_{BW} is the rotation part of T_{BW}.

To find the best metric for CubemapSLAM, we thoroughly evaluated the metrics. For convenience we keep the notation used in [30], where the angular metrics are denoted as r_{a1} and r_{a2}, and the tangential metric and vector difference metric are denoted as r_t and r_f respectively. The multi-camera model based metric is denoted as r_u. We compute the ATE RMSE(Absolute trajectory error) [25] of the system with different metrics in pose optimization. The evaluation is performed on a long straight track with local bundle adjustment disabled. In the result, r_t and r_f achieve errors as $11.27m$ and $20.53m$ and fail to keep the scale of the map, and r_{a1} and r_{a2} fail quickly after initialization. In contrast r_u achieves the most accurate result as $1.03m$, which indicates that the multi-pinhole model is more suitable for our system.

3 Experiments

We evaluate the performance of our system in the multi-fisheye dataset Lafida [28] as well as a dataset collected in large outdoor environments with our autonomous vehicle. In the Lafida dataset, we evaluate CubemapSLAM and Multicol-SLAM [27] on both accuracy and robustness. Then on the dataset collected by ourselves, we first evaluate the systems under two types of camera settings as in section 3.2. We further investigate the effect of different mounting positions of the camera and loop closure. The result shows that our CubemapSLAM system performs consistently more robustly in all the experiments than the other ones, and it provides competitive accuracy.

3.1 Dataset

The Lafida dataset [28] is a multi-fisheye camera dataset collected for evaluating multi-fisheye SLAM systems. There are 6 sequences in total, which are *in_dynamic*, *in_static*, *out_static*, *out_static*2, *out_rotation* and *out_large_loop* captured from three rigidly mounted fisheye cameras. All the cameras share the same resolution of 754×480 pixels. As for our dataset, we equip the vehicle with two types of cameras: a pinhole camera with 80° FoV, and a fisheye camera with 190° FoV. Note that the pinhole camera and fisheye camera share the same model of sensor chip but use different lenses, so the pinhole and raw fisheye image have the same resolution of 1280×720 pixels.

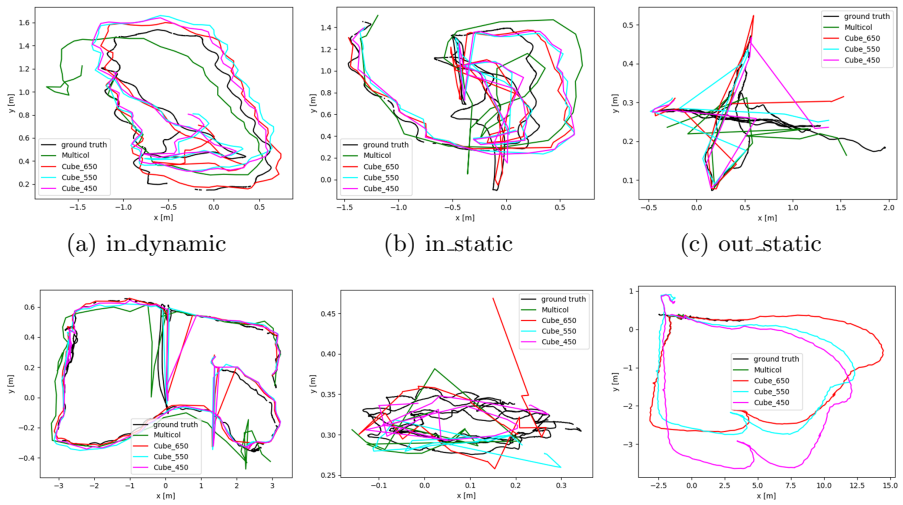
The experiments on our dataset contain two camera settings. In the first setting, a pinhole camera and a fisheye are mounted at the frontal part of the vehicle (both facing front), and the other fisheye camera is mounted at the left side of the vehicle (facing left). In this setting, we choose various routes, including a large loop around an industrial park (*loop*1 sequence), a smaller loop inside the park with sharp turns (*loop*2 sequence), a large u-turn on the *loop*1 sequence (*uturn* sequence), a large sequence in a town with no loop but with traffic lights and traffic jams (*town* sequence), and a loopy route in an outdoor parking lot (*parkinglot* sequence). To further investigate the performance of the lateral mounting cameras, we create a second setting where a fisheye camera and a pinhole camera are both mounted laterally. We travel along the routes of *loop*1, *town*1 and *parkinglot* and recollect the data under the new camera setting. We name the collected data *loop*1_*c_clockwise*, *loop*1_*clockwise*, *town*_1 and *parkinglot*_1. *loop*1_*c_clockwise* and *loop*1_*clockwise* share the same route but drive in opposite directions. For all the sequences, D-GPS is used as groundtruth.

3.2 Baseline Comparison

We first compare our system with Multicol-SLAM [27] on the Lafida dataset [28], where Multicol-SLAM is sufficiently tested and well performed. For fairness, Multicol-SLAM is configured with one fisheye camera which is the same one as CubemapSLAM. For a comprehensive comparison, we set the resolution of the faces as 450×450, 550×550 and 650×650 pixels, respectively. In the experiment, both of the systems are configured to extract 2000 features, which we consider it is enough and representative for the dataset considering the image resolution.

On our dataset, the CubemapSLAM operates on the front (**Cube-F**) and left(**Cube-L**) fisheye cameras. We simply set the face resolution as 650×650 pixels. The first baseline comparision is to perform ORB-SLAM [18] on rectified fisheye images from front (**ORB-Rect-F**) and left (**ORB-Rect-L**) cameras. The rectified images are set to 100° FoV with a resolution of 775×775 pixels which share the same focal length with cubemap virtual cameras. Another baseline approach is to perform ORB-SLAM [18] on pinhole images from front(**ORB-Pin-F**) and left cameras(**ORB-Pin-L**). We also tested Multicol-SLAM [27] on the collected dataset. However, we find Multicol-SLAM fails soon after initialization stage in most of the sequences as it does in Lafida *out_large_scale* sequence, so we do not include trajectories of Multicol-SLAM in result comparison.

We first test the systems with loop closing thread disabled to evaluate system performance with only VO. In the experiments, all the systems are configured to extract 3000 features per image. For each output trajectory, we align it with the ground truth by a 7-DoF transformation since the scale is unknown. After that, we compute the ATE RMSE [25] of each trajectory for comparison. We also evaluate the tracking and mapping quality of all the systems by measuring the average number of tracked keypoints in each sequence. Note that for all the entries in the tables, we add a mark of *lost* in the entry if the system gets lost after finishing more than half of the sequence, and we add a mark of X if the system gets lost soon after initialization.



(a) in_dynamic (b) in_static (c) out_static

(d) out_static2 (e) out_rotation (f) out_large_loop

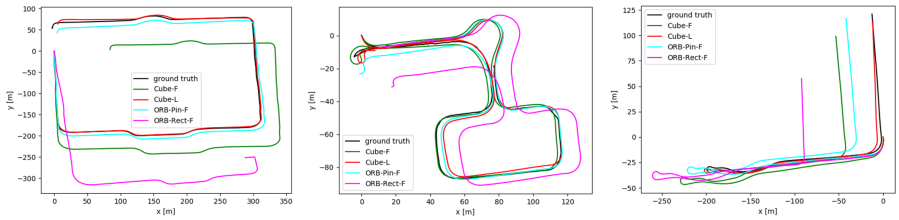
Fig. 4. Trajectories on Lafida dataset [28] aligned to groundtruth with a 7-DoF transformation of Multicol-SLAM and CubemapSLAM with resolution of face as 450×450, 550×550 and 650×650 pixels, respectively.

3.3 Results on Lafida Dataset

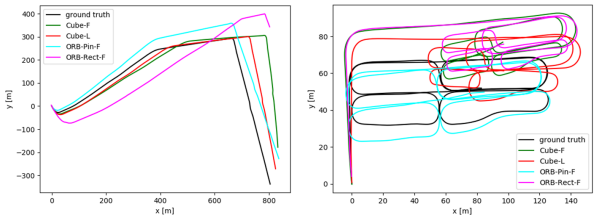
We carefully evaluate both systems on all the six sequences, and the qualitative and quantitative results are shown in Table 1 and Fig.4. The results show that the CubemapSLAM performs better than Multicol-SLAM in most sequences and the performance is stable with various face size. Also it should be noted that in *out_large_scale* although the error of Multicol-SLAM is slightly lower, the number of tracked frames are significantly less than ours. We've tested Multicol with several different start points for fairness, but the results do not show much difference. We notice Multicol-SLAM usually fails when the camera motion becomes large. However, large motions are very common in large-scale outdoor dataset.

Table 1. ATE RMSE and tracked frames(over all frames) on Lafida dataset [28] (m)

	in_dynamic	in_static	out_static	out_static2	out_rotation	out_large_loop
Multicol	0.78 (880/899)	0.32 (1001/1015)	0.07 (**726**/755)	0.31 (1314/1642)	0.05 (397/779)	0.06 (202/3175)
Cubemap 650 × 650	**0.17** (**893**/899)	**0.15** (997/1015)	**0.02** (722/755)	0.14 (1604/1642)	0.06 (253/779)	**0.16** (3111/3175)
Cubemap 550 × 550	0.28 (**893**/899)	0.16 (**1006**/1015)	0.03 (717/755)	0.15 (1604/1642)	0.10 (399/779)	0.44 (**3132**/3175)
Cubemap 450 × 450	0.24 (**893**/899)	0.17 (**1006**/1015)	0.03 (716/755)	**0.13** (**1605**/1642)	**0.04** (**743**/779)	0.39 (3129/3175)



(a) loop1 (b) loop2 (c) uturn



(d) town (e) parkinglot

Fig. 5. Trajectories aligned to groundtruth with a 7-DoF transformation of the **Cube-F**, **Cube-L**, **ORB-Pin-F** and **ORB-Rect-F**.

Table 2. ATE RMSE (m) and Average Number of Tracked Points (pt)

	ATE RMSE				Average Number of Tracked Points			
	Cube-F	Cube-L	ORB-Pin-F	ORB-Rect-F	Cube-F	Cube-L	ORB-Pin-F	ORB-Rect-F
loop1	22.73	**3.12**	3.92	121.14	194.21	198.73	**206.84**	190.73
loop2	2.68	**2.26**	2.54(lost)	6.66(lost)	210.64	229.35	**232.28**	201.45
uturn	12.14	**2.22**	7.99	27.85	196.78	**236.65**	176.74	150.93
town	57.32	23.58	**13.41**	207.15	256.06	**338.06**	222.50	207.29
parkinglot	16.47	9.54	**2.29**	16.92	170.34	**182.45**	148.36	164.34

3.4 Results of Setting1

The ATE RSME results for each method can be found in Table 2, and the comparison of the trajectories from different systems are shown in Fig. 5. In all the sequences, the ATE error of **ORB-Rect-F** is significantly larger than the other methods, which is the consequence of both a reduced FoV from the full fisheye image and a lower angular resolution than that of the pinhole camera. **ORB-Pin-F** has a low ATE RMSE in most of the sequences due to its higher angular resolution than fisheye image. However, in the *loop*2 sequence, **ORB-Pin-F** fails to complete the entire trajectory due to a drastic turn at the end of the sequence. In contrast, both **Cube-F** and **Cube-L** successfully complete all the sequences including the difficult *loop*2 with the help of a larger FoV. In addition, we find that in *loop*1, *loop*2 and *uturn* sequences, **Cube-L** achieves the best result. In the *town* and the *parkinglot* sequences, as the feature points are relatively far from the camera, **ORB-Pin-F** outperforms **Cube-L** by a small margin, but we will show that the gap is significantly reduced after loop closure.

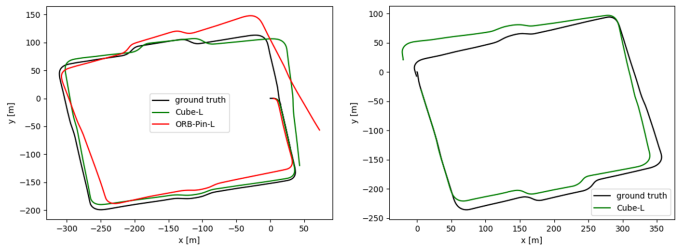
For the number of tracked keypoints, as in Table 2, **Cube-L** tracks the most points in *uturn*, *town* and *parkinglot*, and performs close to **ORB-Pin-F** in *loop*1 and *loop*2. Note that besides the advantage of better tracking quality, more tracked keypoints also contributes to a denser and more structural map.

Table 3. ATE RMSE (m) and Average Number of Tracked Points (pt)

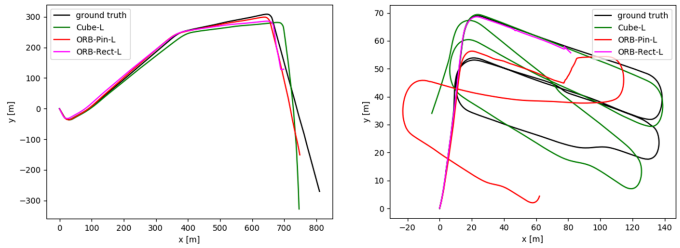
	ATE RMSE			Average Number of Tracked Points		
	Cube-L	ORB-Pin-L	ORB-Rect-L	Cube-L	ORB-Pin-L	ORB-Rect-L
loop1_c_clockwise	**9.84**	15.60	X	**280.01**	215.06	241.97
loop1_clockwise	**8.94**	X	X	**136.99**	X	X
town_1	**14.75**	16.20(lost)	6.94(lost)	366.14	**379.21**	375.09
parkinglot_1	**5.44**	21.74(lost)	X	**193.80**	191.01	152.76

3.5 Results of Setting2

To make the comparison fair for the lateral mounting cameras, we mount both types of cameras towards left and evaluate the performance respectively. Quantitative and qualitative results are shown in Table 3 and Fig. 6. In all the sequences, **Cube-L** outperforms the other two methods by a large margin. In the clockwise sequence where the cameras look outwards the park, both **ORB-Pin-L** and **ORB-Rect-L** get lost soon after initialization due to lack of texture and occlusions by objects close-by. We therefore do not compute the error and replace each field with a X. Also in *town*_1, both **ORB-Rect-L** and **ORB-Pin-L** get lost before finishing the sequence due to occlusion from cars passing by. In addition, we list the number of average tracked points in Table 3, in which **Cube-L** achieves better overall performance than the other systems.

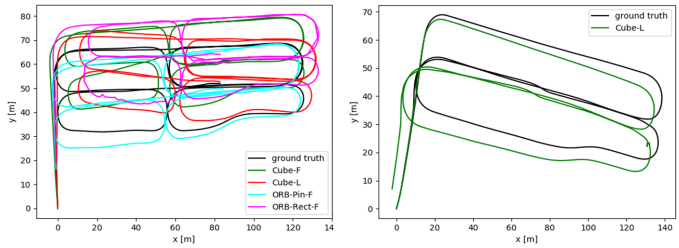


(a) loop1_1_c_clockwise (b) loop1_1_clockwise



(c) town_1 (d) parkinglot_1

Fig. 6. Trajectories aligned to groundtruth with a 7-DoF transformation of the **Cube-L**, **ORB-Pin-L** and **ORB-Rect-L**.



(a) parkinglot (b) parkinglot_1

Fig. 7. Trajectories aligned to groundtruth with a 7-DoF transformation of the Cubemap and ORB-SLAM with loop closing.

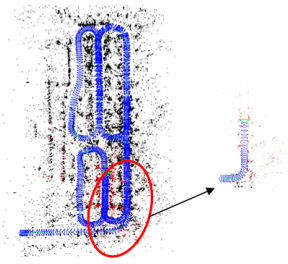


Fig. 8. Qualitative results of **Cube-F** (left) and **ORB-Pin-F** (right) on the *parkinglot* sequence with a lower frame rate by choosing one image from every three images.

Table 4. ATE RMSE with Loop Closure (m) on parkinglot and parkinglot_1 Sequences

	parkinglot				parkinglot_1		
	Cube-F	Cube-L	ORB-Pin-F	ORB-Rect-F	Cube-L	ORB-Pin-L	ORB-Rect-L
w/o loop closing	16.47	9.54	**2.29**	16.92	**5.44**	21.74(lost)	X
w/ loop closing	4.47	3.02	**1.69**	4.73	**2.41**	X	X

3.6 Results of Loop Closing

To evaluate system performance with loop closing thread enabled, we test the performances of the systems in both *parkinglot* and *parkinglot_1* sequences with and without loop closing. Results show that errors of **Cube-F** and **Cube-L** are greatly reduced and getting comparable to **ORB-Pin-F** with loop closing (see Table 4), and the robustness consistently outperform the rest. A qualitative result is shown in Fig. 7. To further reveal the advantage of a large FoV camera, we test **ORB-Pin-F** and **Cube-F** in the *parkinglot* sequence with a reduced frame rate by choosing one image out of three in the sequences. We find that **ORB-Pin-F** is hard to initialize and fails soon after initialization, while **Cube-F** is able to initialize fast, track stably, and successfully perform loop closing. A qualitative result of the trajectories is shown in Fig. 8.

4 Conclusions

This work presents a novel CubemapSLAM system that incorporates the cubemap model into the state-of-the-art feature based SLAM system. The cubemap model utilizes the large FoV of fisheye camera without affecting the performance of feature descriptors. In addition, CubemapSLAM is efficiently implemented and can run in real time. In the experiments, we extensively evaluate our systems in various challenging real-world cases and prove that our CubemapSLAM solution is consistently more robust than other approaches without losing accuracy. We also discover that by optimizing the mounting position of the fisheye camera and enabling the loop closing thread, CubemapSLAM can achieve even better accuracy than pinhole cameras, despite the limited angular resolution of the sensor. Overall, we provide an efficient and practical fisheye SLAM solution. Future work includes extending the cubemap model to stereo or multiple camera setting to further improve robustness as well as recover the absolute scale. The source code is available at `https://github.com/nkwangyh/CubemapSLAM`.

References

1. Arican, Z., Frossard, P.: Omnisift: Scale invariant features in omnidirectional images. In: Image Processing (ICIP), 2010 17th IEEE International Conference on. pp. 3505–3508. IEEE (2010)
2. Barfoot, T.D.: State Estimation for Robotics. Cambridge University Press (2017)

3. Caruso, D., Engel, J., Cremers, D.: Large-scale direct slam for omnidirectional cameras. In: Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on. pp. 141–148. IEEE (2015)
4. Engel, J., Schöps, T., Cremers, D.: Lsd-slam: Large-scale direct monocular slam. In: European Conference on Computer Vision. pp. 834–849. Springer (2014)
5. Forster, C., Zhang, Z., Gassner, M., Werlberger, M., Scaramuzza, D.: Svo: Semidirect visual odometry for monocular and multicamera systems. IEEE Transactions on Robotics **33**(2), 249–265 (2017)
6. Furgale, P., Schwesinger, U., Rufli, M., Derendarz, W., Grimmett, H., Mühlfellner, P., Wonneberger, S., Timpner, J., Rottmann, S., Li, B., et al.: Toward automated driving in cities using close-to-market sensors: An overview of the v-charge project. In: Intelligent Vehicles Symposium (IV), 2013 IEEE. pp. 809–816. IEEE (2013)
7. Gálvez-López, D., Tardos, J.D.: Bags of binary words for fast place recognition in image sequences. IEEE Transactions on Robotics **28**(5), 1188–1197 (2012)
8. Hartley, R., Zisserman, A.: Multiple view geometry in computer vision. Cambridge university press (2003)
9. Heng, L., Choi, B.: Semi-direct visual odometry for a fisheye-stereo camera. In: Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on. pp. 4077–4084. IEEE (2016)
10. Heng, L., Li, B., Pollefeys, M.: Camodocal: Automatic intrinsic and extrinsic calibration of a rig with multiple generic cameras and odometry. In: Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on. pp. 1793–1800. IEEE (2013)
11. Kangni, F., Laganiere, R.: Orientation and pose recovery from spherical panoramas. In: Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on. pp. 1–8. IEEE (2007)
12. Kneip, L., Furgale, P.: Opengv: A unified and generalized approach to real-time calibrated geometric vision. In: Robotics and Automation (ICRA), 2014 IEEE International Conference on. pp. 1–8. IEEE (2014)
13. Lee, G.H., Faundorfer, F., Pollefeys, M.: Motion estimation for self-driving cars with a generalized camera. In: Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on. pp. 2746–2753. IEEE (2013)
14. Linegar, C., Churchill, W., Newman, P.: Work smart, not hard: Recalling relevant experiences for vast-scale but time-constrained localisation. In: Robotics and Automation (ICRA), 2015 IEEE International Conference on. pp. 90–97. IEEE (2015)
15. Liu, P., Heng, L., Sattler, T., Geiger, A., Pollefeys, M.: Direct visual odometry for a fisheye-stereo camera. In: International Conference on Intelligent Robots and Systems (IROS), Vancouver, Canada (2017)
16. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International journal of computer vision **60**(2), 91–110 (2004)
17. Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: Orb-slam: a versatile and accurate monocular slam system. IEEE Transactions on Robotics **31**(5), 1147–1163 (2015)
18. Mur-Artal, R., Tardós, J.D.: Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. IEEE Transactions on Robotics **33**(5), 1255–1262 (2017)
19. Pagani, A., Stricker, D.: Structure from motion using full spherical panoramic cameras. In: Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on. pp. 375–382. IEEE (2011)

20. Pless, R.: Using many cameras as one. In: Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on. vol. 2, pp. II–587. IEEE (2003)
21. Rituerto, A., Puig, L., Guerrero, J.J.: Visual slam with an omnidirectional camera. In: Pattern Recognition (ICPR), 2010 20th International Conference on. pp. 348–351. IEEE (2010)
22. Ros, G., Sappa, A., Ponsa, D., Lopez, A.M.: Visual slam for driverless cars: A brief survey. In: Intelligent Vehicles Symposium (IV) Workshops. vol. 2 (2012)
23. Scaramuzza, D., Martinelli, A., Siegwart, R.: A toolbox for easily calibrating omnidirectional cameras. In: Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on. pp. 5695–5701. IEEE (2006)
24. Scaramuzza, D., Siegwart, R.: Appearance-guided monocular omnidirectional visual odometry for outdoor ground vehicles. IEEE transactions on robotics **24**(5), 1015–1026 (2008)
25. Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.: A benchmark for the evaluation of rgb-d slam systems. In: Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on. pp. 573–580. IEEE (2012)
26. Tardif, J.P., Pavlidis, Y., Daniilidis, K.: Monocular visual odometry in urban environments using an omnidirectional camera. In: Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on. pp. 2531–2538. IEEE (2008)
27. Urban, S., Hinz, S.: Multicol-slam-a modular real-time multi-camera slam system. arXiv preprint arXiv:1610.07336 (2016)
28. Urban, S., Jutzi, B.: Lafidaa laserscanner multi-fisheye camera dataset. Journal of Imaging **3**(1), 5 (2017)
29. Ventura, J., Höllerer, T.: Wide-area scene mapping for mobile visual tracking. In: Mixed and Augmented Reality (ISMAR), 2012 IEEE International Symposium on. pp. 3–12. IEEE (2012)
30. Zhang, Z., Rebecq, H., Forster, C., Scaramuzza, D.: Benefit of large field-of-view cameras for visual odometry. In: Robotics and Automation (ICRA), 2016 IEEE International Conference on. pp. 801–808. IEEE (2016)
31. Zhao, Q., Feng, W., Wan, L., Zhang, J.: Sphorb: A fast and robust binary feature on the sphere. International Journal of Computer Vision **113**(2), 143–159 (2015)
32. Ziegler, J., Bender, P., Schreiber, M., Lategahn, H., Strauss, T., Stiller, C., Dang, T., Franke, U., Appenrodt, N., Keller, C.G., et al.: Making bertha drivean autonomous journey on a historic route. IEEE Intelligent Transportation Systems Magazine **6**(2), 8–20 (2014)