2018 IEEE Intelligent Vehicles Symposium (IV)
Changshu, Suzhou, China, June 26-30, 2018

Real-time Omnidirectional Visual SLAM with Semi-Dense Mapping

Senbo Wang[1], Jiguang Yue[1], Yanchao Dong[1*], Runjie Shen[1] and Xinyu Zhang[2]

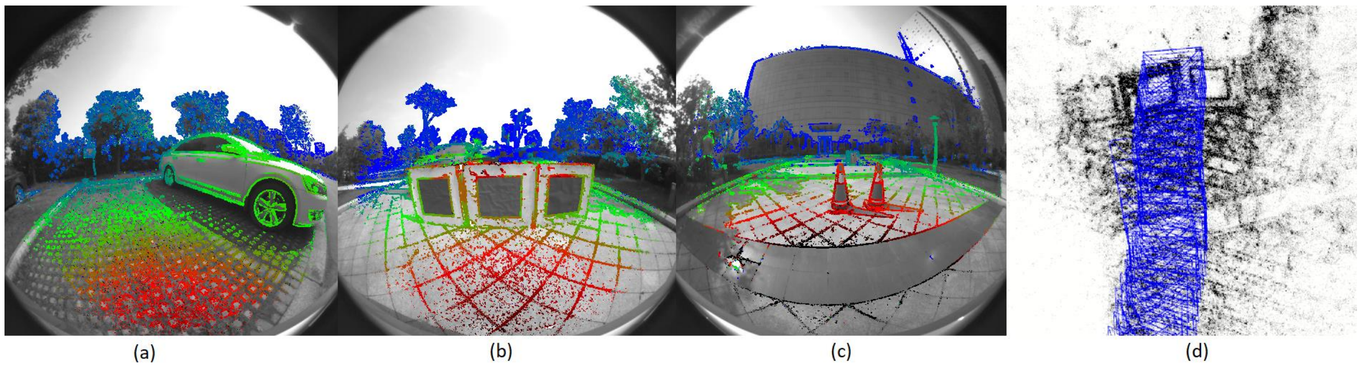


Fig. 1: Results of our omnidirectional Visual SLAM system on three different scenories. (a)(b)(c) are captured images overlaied with color coded semi-dense depth maps. (d) is the reconstructed 3D depth map of (b).

***Abstract*— The state of art Visual SLAM is going from sparse feature to semi-dense feature to provide more information for environment perception, whereas the semi-dense methods often suffer from inaccurate depth map estimation and are easy to become instable for some real-world scenarios. The paper proposes to extend the ORB-SLAM2 framework, which is a robust sparse feature SLAM system tracking camera motion with map maintenance and loop closure, by introducing the unified spherical camera model and the semi-dense depth map. The unified spherical camera model fits the omnidirectional camera well, therefore the proposed Visual SLAM system could handle fisheye cameras which are commonly installed on modern vehicles to provide larger perceiving region. In addition to the sparse corners features the proposed system also utilizes high gradient regions as semi-dense features, thereby providing rich environment information. The paper presents in detail how the unified spherical camera model and the semi-dense feature matching are fused with the original SLAM system. Both accuracies of camera tracking and estimated depth map of the proposed SLAM system are evaluated using real-world data and CG rendered data where the ground truth of the depth map is available.**

I. Introduction

Simultaneous localization and mapping (SLAM) and visual odometry (VO) are fundamental elements for many emerging technologies – from robots to autonomous vehicles to augmented reality. Recent years we've seen rapid development of real-time Visual SLAM systems, especially methods that don't rely on local image features of points to gain visual correspondence between two frames [1-3], namely "direct methods" or "dense methods". Comparing with traditional sparse methods which can only reconstruct a few points on the camera scene, dense methods reconstruct most part of the image in real-time, thus give extra information on camera's surroundings. On the other hand, "indirect methods" or "sparse methods" [4, 5] utilize robust local image features to match a group of sparse points on the scene. State-of-the-art sparse methods provide reliable camera localization results and robustness, especially for outdoor environments. However, only know a few sparse feature points is inefficacious for tasks requiring accurate depth map such as static obstacle detection.

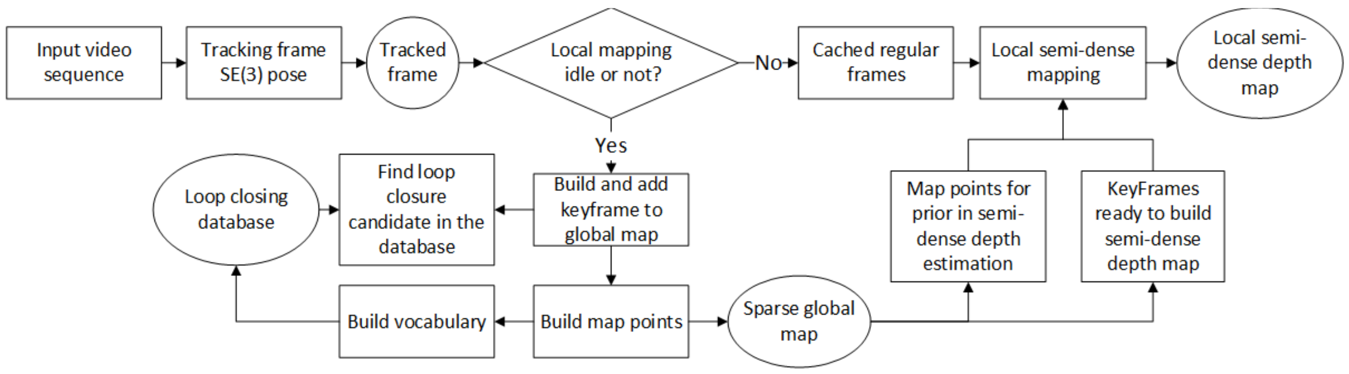
Omnidirectional cameras have been widely applied on various fields, both for research and for industry, thanks to the property that the camera can provide larger view of its immediate surroundings than common cameras. The character that omnidirectional cameras have large Field-of-View (FoV) is useful when the system tries to gain accurate correspondences even between frames even with relatively large search baseline, thus contributes to both the accuracy of camera localization and reconstruction of surroundings. However, lower angular resolution, strong non-linear distortion across the image and complex camera model adds extra challenge to the above reconstruction process, leave us with space for studying and developing. Most existing SLAM methods about omnidirectional camera use catastrophic cameras, but recent years we've also seen methods using commonly-used fisheye cameras.

Recent years we've seen various kinds of omnidirectional visual SLAM methods, both for sparse methods and dense methods, as we will discuss in the next chapter.

This work is partially supported by the Natural Science Foundation of Shanghai (No. 18ZR1442500), the Fundamental Research Funds for the Central Universities and the Science and Technology Project (No. 10.11) from China Datang Corporation Renewable Power Co., Limited.

Senbo Wang (1410472@tongji.edu.cn), Jiguang Yue (yuejiguang@tongji.edu.cn), Yanchao Dong (*Corresponding Author, dongyanchao@tongji.edu.cn) and Runjie Shen (shenrunjie@tongji.edu.cn) are with Tongji University, No.4800 Cao'an Road, Shanghai, P. R. China.

Xinyu Zhang is with the Datang Guoxin Binhai Offshore Wind Power Co., Ltd.



Input video sequence
Tracking frame SE(3) pose
Tracked frame
Local mapping idle or not?
No
Cached regular frames
Local semi-dense mapping
Local semi-dense depth map
Yes
Loop closing database
Find loop closure candidate in the database
Build and add keyframe to global map
Map points for prior in semi-dense depth estimation
KeyFrames ready to build semi-dense depth map
Build vocabulary
Build map points
Sparse global map

Fig. 2: Overview of the proposed algorithm. Tracking frame pose, build keyframes and map points, detect loop closure and semi-dense mapping are done on three separate threads. The local semi-dense mapping part is detailed in section III.

For sparse visual SLAM methods, several works [6-8] have proposed a similar method that utilize robust features (SIFT) and traditional EKF-SLAM to accurately estimate camera location; however, these methods lack efficient loop closing and relocalization technique; some of these methods cannot work in real-time, as they can only handle less than 2 images per second. Lourenco proposed localization method [9] use modified version of SIFT image features to achieve accurate visual place recognition. The method needs a group of pre-defined omnidirectional image as environment map to estimate camera position, limits its applicable scope. A more complete method using variants of SIFT feature descriptor is described here [10], yet the method is only a simple visual odometry; comparing with the State-of-the-art Visual SLAM method, rotation error of the proposed method is relatively high.

Works like [5, 11, 12] use multiple cameras looking at different positions to achieve accurate and robust sparse Visual SLAM. These methods require calibrated multi-camera rig like Around View Monitoring system on vehicles, which introduce extra constraints like dense stereo correspondence to aid the navigation algorithm.

Earlier works on dense omnidirectional Visual SLAM are carried out by appearance based methods [13, 14]. Both the above two works use high-dimensional image features, but they cannot perform reconstruction of the surroundings, and the time-consuming feature computation process prevents them to accomplish real-time large-scale Visual SLAM. Method from Caruso [15] is based on the previous work [2] and can achieve real-time reconstruction and camera pose estimation. However, this work is relatively vulnerable for some challenging scenes, and its random-initialization technique is unreliable under certain test scenes. Some works [12] use sparse feature points as pose estimation front-end, and calculate 3D reconstruction by cost-volume optimization, which allows robots to estimate high-dimensional scene information such as room layout on-the-fly. These works can effectively reconstruct camera's nearby but cannot be carried out in real-time with only the support of Desktop CPU.

In practice, most Visual SLAM methods focus on estimating camera's location, mapping the surroundings is one of the steps in helping the system handle complex and unconstraint environments; the resulting map seems to be an "extra gift". However, moving robots and autonomous vehicles have the urgent need to get a brief idea of 3D shape and key features of remarkable objects in the environment by exploration. The recent fast developing real-time direct Visual SLAM methods can describe 3D shape of the environments by semi-dense depth mapping, yet these systems only use the map for instant camera localization, thus lack appropriate map management mechanism for adding, fusing and deleting an area in the map; also, when comparing with sparse Visual SLAM methods, these Visual SLAM methods' robustness is relatively lower. On the other hand, dense 3D reconstruction algorithm like [12] could deliver solid performance both in evaluating camera's nearby and estimating camera's position, but these algorithms require GPU to operate, which blocks moving the algorithms to embedded systems. In this paper we present a system that fits in the space between the above two categories of existing methods: to achieve accurate localization and reconstruction at the same time with only desktop CPU. To achieve this, we combine and modify the real-time SLAM method from [4, 5], semi-dense mapping method from [2, 15, 16] to propose a method that use sparse Visual SLAM on omnidirectional camera as the front end to estimate camera's location and semi-dense mapping system as the back end to get reconstruction of camera's surroundings.

In our approach, we sample sparse image features for camera pose estimation done by bundle adjustment on unit sphere on unified omnidirectional camera models. We formulate semi-dense depth map on each individual keyframe in the map by small-baseline omnidirectional epipolar line correspondence search on frames near the keyframe in use. The semi-dense map is controlled and managed by map management engine of the sparse SLAM front end, which allows the system to build up a brief shape of the surroundings based on the optimized and adjusted sparse feature map. The structure of the algorithm is shown in Fig.2. The whole algorithm is tested by real-world datasets and CG datasets rendered. Out contribution is as follows: (1) Accurate and complete Visual SLAM method on spherical image of omnidirectional camera; (2) Robust and accurate semi-dense depth-map estimation algorithm on omnidirectional camera without the drawback on state-of-the-art semi-dense Visual SLAM system.

II. OMNIDIRECTIONAL VISUAL SLAM

In this section, we present our framework on omnidirectional Visual SLAM algorithm. We follow the implementation in ORB-SLAM2 [4] that use separate threads in tracking, map building and loop closing, and build covisibilty map to help local bundle adjustment, manage keyframes and store loop closing candidates, yet our method features special orientation for fisheye camera. An overview of details of the algorithm can be found in Fig.2.

A. Bundle adjustment

We use here keypoint-based bundle adjustment implementation to estimate camera poses and correct map point positions in real-time. Assuming known 3D position of a group of feature points, the pose T_k of camera in a frame in world coordinate system can be calculated by minimizing cost function:

$$T_k = argmin \sum_{P_i \in Q} (\| p_i - \pi(T_k^{-1} * P_i) \|_h) \tag{1}$$

The Q is a group of 3D points which matching 2D points can be found in the corresponding frame, namely map points that can be matched by feature matching on this frame; $\pi(\cdot)$ is the projection function of omnidirectional camera. We use camera model from [17] to project image points onto unit sphere. P_i and p_i are 3D coordinates and 2D coordinates on the image frame, respectively.

This minimization problem can be solved by the Levenberg-Marquardt algorithm. Thus, for each iteration, we need to solve a left-multiplied increment for each point:

$$\Delta T_k = \left(J_i^T W J_i + \lambda diag(J_i^T W J_i)\right)^{-1} J_i^T r_i \tag{2}$$

Every Jacobian matrix J_i is associated with the point, and can be estimated by chain rule as:

$$J_i = \boldsymbol{J_{i|\xi} J_{\xi|s} J_{s|c} J_{c|T}} \tag{3}$$

Where $\boldsymbol{J_{i|\xi}}$ is a 2×2 Jacobian of the function that maps the image point onto the normalized plane in [17]; the $\boldsymbol{J_{\xi|s}}$ is the 2×3 Jacobian of the function that map points on the normalized plane to the united sphere; the $\boldsymbol{J_{s|c}}$ is the derivative matrix that describes the relationship between the point on the unit sphere and 3D point in the camera coordinate system; the $\boldsymbol{J_{c|T}}$ is a 3×6 Jacobian matrix denoting the left compositional derivative of the transformation that map the world coordinates into camera coordinate system [15].

For refining cameras' location and map points' 3D position in the map, we carry out local bundle adjustment and global bundle adjustment, with cost function like (1). As we are using g2o [18] to do the optimization, the Jacobian can be written in the form:

$$J = (0, \cdots, 0, A, 0, \cdots, 0, J_i, 0, \cdots) \tag{4}$$

$$A = \boldsymbol{J_{i|\xi} J_{\xi|s} J_{s|c} R} \tag{5}$$

B. Map maintenance

We keep track of the 3D feature points in the map using the distinctive descriptor of the "map point", which has median descriptor distance to the descriptors of all the know observations of the map point [4]. The covisibility map which can give a brief description of a keyframe's surroundings can thus be built by fusing map points in one specified keyframes and other nearby keyframes [19]. We use only the status of mapping thread as the criteria for adding new keyframe to the map. Comparing with the original strategy in [4], this let us add more keyframe within a short distance, which distribute the computational cost of the semi-dense mapping session evenly into the SLAM process.

III. Semi-Dense Mapping

In this section, we present our method on semi-dense depth mapping.

A. Overview of the algorithm

For real-time semi-dense (or dense) feature correspondence search, there are mainly two different techniques: stereo matching and cost-volume minimization.

The current works on large-baseline semi-dense mapping [16] and classical small-baseline approach [2] share the same problem that they cannot recognize overlapped part in two individually mapped frames; they also lack the capability to actively manage and re-use the already mapped area.

To solve the above problems, we map pixels on keyframes of the proposed sparse SLAM methods utilizing map points on the keyframe being mapped. The semi-dense map on each keyframe is formulated by patches owned by map points observed on that keyframe. We give each map point a fixed size patch and calculate depth of the "significate" area (pixels with high gradient modulo) by stereo depth search. The references for stereo search are common frames near the keyframe. For each new keyframe, we propagate depth from the old map points onto the keyframe and calculate new depth for newly added map points. The overlapping area between two patches belonging to different map points will thus receive multiple depth calculation results; we check the capability between two results d_a and d_b using χ^2 test at 95% [16]:

$$\frac{(d_a - d_b)^2}{\sigma_a^2} + \frac{(d_a - d_b)^2}{\sigma_b^2} < 95\% \tag{6}$$

σ_a and σ_a are uncertainties of the depth estimation respectively. For the group of best capable results, we fuse their results by EKF fusion. An overview of the process is shown in Fig.3.

B. Semi-dense correspondence Matching

Each point with high gradient modulo is searched on the epipolar "curve" segment on the image to find a match. The known depth of map points on the keyframe provide us with a perfect prior estimation of depth interval to search. This work follows the idea from [15] that build a straight line segment between points related to d_{max} and d_{min} on the epipolar "curve", and search along that line to get a group of possible match candidates. A point on that line segment in the 3D space of the united sphere coordinate system can be expressed as:

$$q = \alpha p_{min} + (1 - \alpha) p_{max} \tag{7}$$

We start the searching process from the farthest point p_{max} to minimize the search interval. Pixels are traversed on the line segment by reevaluating the value of α in each step by adding an increment to the α:

$$\alpha_n = \alpha_{n-1} + \delta_{n-1} \tag{8}$$

We try to keep that, for each step, the distance between two points on image is roughly 1 pixel. Apply the first order Tylor expansion of $\pi(\cdot)$ gives us the relationship between image coordinates $\pi(q_n)$ and $\pi(q_{n-1})$:

$$\pi\big(q(\alpha_n)\big) = \pi\big(q(\alpha_{n-1})\big) + \pi'\big(q(\alpha_{n-1})\big)(\alpha_n - \alpha_{n-1}) \quad (9)$$

Thus, the gain δ_{n-1} should be:

$$\delta_{n-1} = \frac{1}{\sqrt{\Big(\pi'\big(q(\alpha_{n-1})\big)_x\Big)^2 + \Big(\pi'\big(q(\alpha_{n-1})\big)_y\Big)^2}} \quad (10)$$

The equation works fine when p_{min} and p_{max} lies near the center of the image; for points near the image border, the calculated image coordinates for points near the p_{min} will be too coarse. Thus, we first calculate all the image points ready to be searched, then add interpolation points on the search line segment between steps larger than 1.5 pixel. By adding extra searching points, the distances between match candidates in succession are controlled to no larger than 1 pixel.

We use the SSD error to measure similarity between the two pixels. Note that the full-range of the search interval is searched; and only when the error of the best match is much lower than the error of the second-best match (judged by a threshold), we consider it as a reliable match. A more vivid illustration of the searching process is shown in Fig.4.

After we have computed the pixel correspondence between the keyframe and the references, we perform a similar uncertainty calculation, point triangulation and inverse depth map smoothing as proposed in [2].

IV. Experiments

We evaluate our algorithm regarding depth map formulation accuracy and camera localization accuracy on real-world data and computer-rendered CG data. The pose and depth rendered is aligned to the coordinate system in SLAM system using a 7DoF transformation. With the depth and pose of every frame, we can calculate pixel correspondences between frames, namely the "optical flow truth". The camera in CG data is set to be a fisheye camera with viewing angle of approximately 200°, and calibrated by placing virtual checkerboards in the scene using toolbox from [20]. The resolution for CG rendered picture is 900×900. The real data is recorded by fisheye cameras mounted on the back of the car, the viewing angle of the camera is roughly 185°. The resolution for real-world picture is 1280×1024. For a sequence of 1200 images of CG data, we spend approximately 139 seconds to track and map the entire sequence. Examples of real-world data and CG data can be found in Fig.1 and Fig.5 respectively.

We designed 2 experiments to evaluate accuracy of our system: the pose estimation experiment and depth map estimation experiment.

The pose estimation experiment is carried out by recording final positions of all keyframes' center in the map and calculate root mean square error (RMSE) as the average error of the pose estimation algorithm.

The depth map estimation experiment will estimate the error for semi-dense mapping. To achieve this, we define the error of depth in one pixel as:

$$e = \| d_{truth} - d_{measure} \| \quad (11)$$

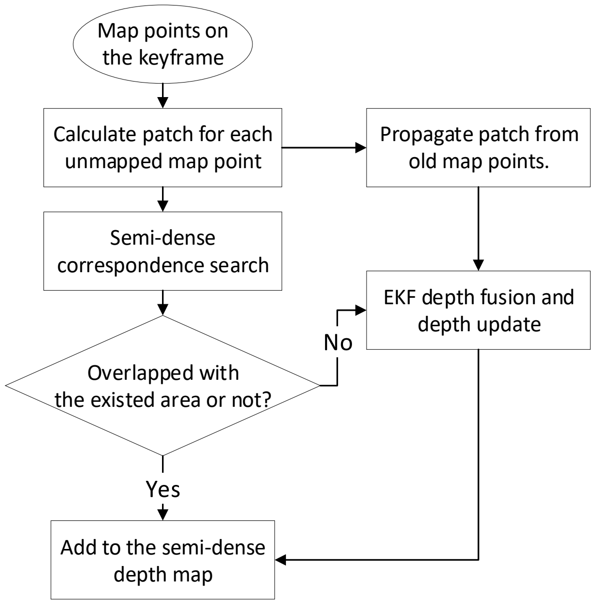

Map points on the keyframe
Calculate patch for each unmapped map point
Propagate patch from old map points.
Semi-dense correspondence search
EKF depth fusion and depth update
No
Overlapped with the existed area or not?
Yes
Add to the semi-dense depth map

Fig. 3: Overview of the semi-dense mapping algorithm. We build semi-dense depth map ultilizing map points on that frame. Details of the stereo seach is detailed in section III.

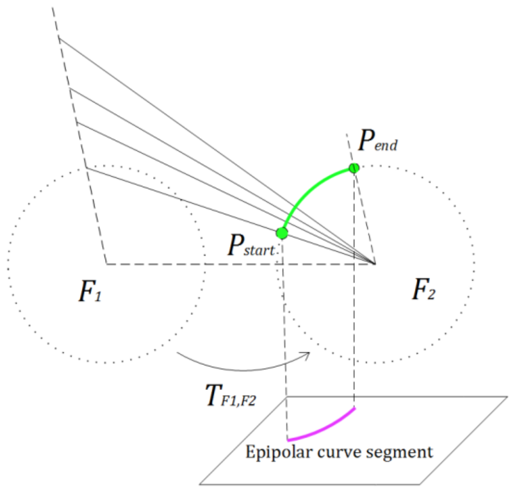


Fig. 4: Correspondence searching process. Ray1 is the point on the unit sphere related to the pixel on the keyframe ready to be searched. We define P_{start} if the pixel has inifinite depth, and P_{end} if the pixel lies on the sphere directly. The two points formulate a curve on the image for searching; the actually searched curve segment is a group of short line segments formulated by a group of points on the curve.

The error increase as the pixel distance d_{truth} increase, hence we treat the ratio between the error and its true distance as the accuracy result. We set the maximum distance of evaluation to 12m, the minimum distance to 0.8m, to simulate obstacle distribution on city roads.

We calculate root mean square error ratio in percentage among every pixel with depth calculated and formulate curves describing relationship between error ratio and distance. Both the accuracy of camera pose estimation and semi-dense correspondence can influence the result of semi-dense depth mapping, thus we conduct experiments based on CG data to evaluate the impact from the two factors.

B. *Pose estimation experiments results*

We test camera localization in our method on two different sets of CG data with different camera routes, the comparison of position between keyframes and the ground truth is shown in Fig.6 (a) and Fig.6 (b) respectively. Fig.6 (c) and Fig.6 (d) provide a detailed view of every keyframe's positioning error. The RMSE error of all the keyframes in the test sequence is about 1.0m for route in Fig.6 (a) and 2.1m for route in Fig.6 (b).

C. *Depth map estimation experiments results*

We demonstrate the semi-dense reconstruction result on real-world data in Fig.1 and on CG data in Fig.7. In Fig.1 we use color coding to demonstrate the depth of the pixel; in Fig.7 we use color to describe error in the frame: the red pixels represent positive errors, and the blue pixel represents negative errors. The deeper the color, the greater the error. The depth error and its relationship with the pixel distance to the camera is also shown in Fig.7.

Fig.7 (a) and Fig.7 (b) demonstrate the error of the semi-dense mapping algorithm. Most of the pixels with error is distributed on part of the image where there exist repeated patterns that disturb matching process or large amount of non-linear distortion. The error related to distance to the camera is also shown. Within 6m to the camera, the RMSE error of the pixel is about 5%~8%, and the maximum error is about 13%. As the pixel goes further, the error increase, over 10m the maximum error is increased to 15% or above.

Using the camera pose provided by the ground truth, we substitute the estimated pose to the truth pose to evaluate the impact on the total errors of depth map by the estimated pose. Fig.7 (c) and Fig.7 (d) demonstrates the result. We can see that the distribution of the error and the average and maximum error line remains almost the same, which implies that the error of pose estimation is ignorable when calculating semi-dense depth map.

From the above result we can learn that the errors of feature correspondence have a major effect on the semi-dense pose estimation. We prove this by substituting semi-dense correspondence by the "optical flow truth" and observe that the average error and maximum error fall below 6% even when pixel distance is more than 10m. The results are shown in Fig.7 (e) and Fig.7 (f) respectively.

V. CONCLUSION

The paper presents a real-time omnidirectional visual SLAM system with semi-dense mapping. The proposed system based on the ORB-SLAM2 framework introduces the unified spherical camera model and semi-dense feature mapping. The paper gives in detail the system's mathematical model and the implementations. The proposed system runs in real-time on desktop PC with only the calculation of CPU and is evaluated using real world data and CG rendered data with motion and map ground truth. It shows the proposed system achieves solid performance, while errors for semi-dense mapping is still relatively large especially on regions with repeated image pattern and with large image distortion.

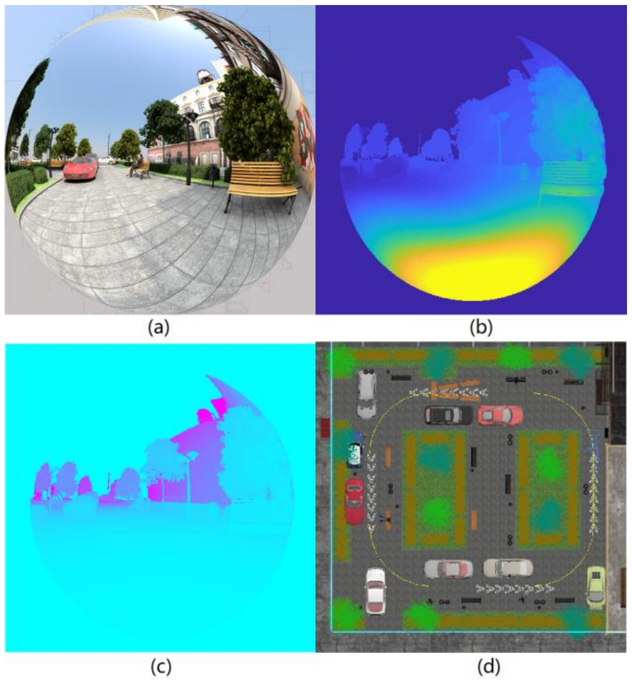


Fig. 5: Computer-rendered CG data. We build up city models in Autodesk 3dsmax and use V-ray sofware to render the scene into a sequence of successive images accroding to pre-defined camera route, one frame of the result is shown in (a). The depth image is rendered simutaously, and is shown in (b). The optical flow image shown in (c) is calculated by pixel depth and frame pose. One of the routes used that the camera moves in the artifical scene is shown in (d).

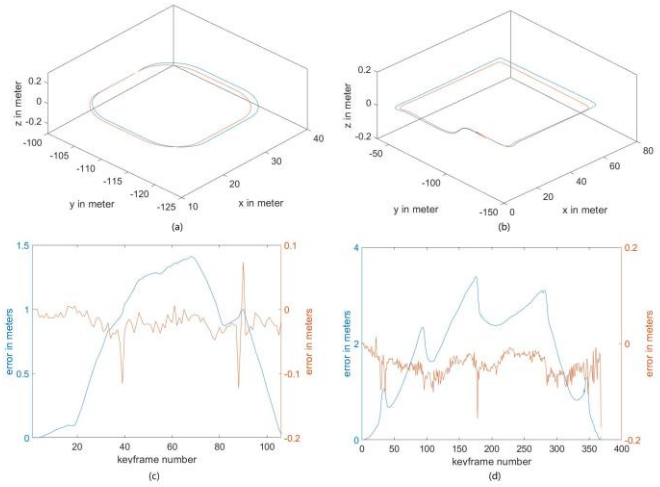


Fig. 6: The result of camera motion tracking algorithm for two routes. There are two routes tested and localization results compared with ground truth are shown in (a) and (b) respectively, and their positioning error of every keyframe in the sequence are shown in (c) and (d): the blue line is the position error, the red line is the error between two successive keyframe.

However, the error for semi-dense mapping system is still relatively high, as the average error exceeds 10% when the distance of the object to the camera is larger than 12 meters. Methods like cost-volume filtering and image alignment which are often used in accurate image correspondence estimation can be adopted in our future work to achieve better accuracy and faster speed.

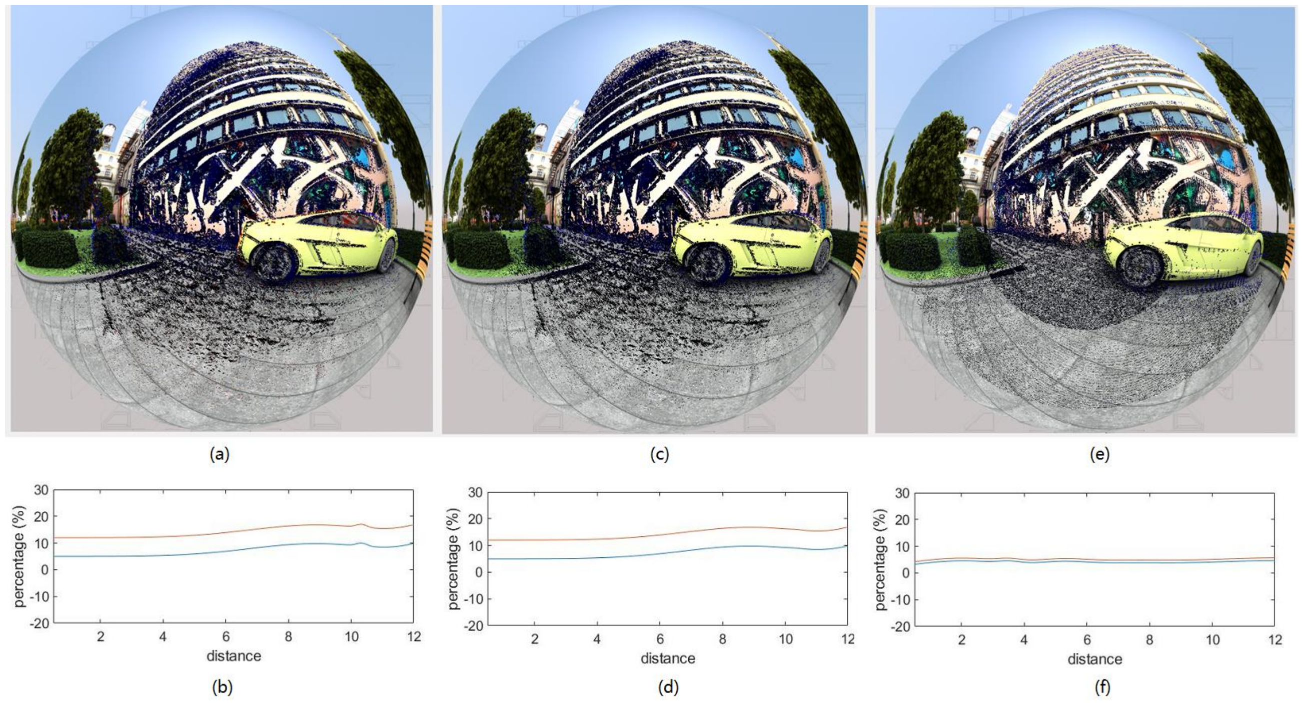


Fig. 7: The result of semi-dense mapping. Images on the first row are demonstration of the mapping algorithm, we use colored pixels to represent pixels with depth errors; pictures on the second row are relations of depth erros and pixel distance to the camera across the test sequence. The blue lines on chart (b)(d)(f) are the average error of all the system, the red lines are the "maximum" error, which means that more than 95% of the pixels' error will be lower than values on this line. Picture (a) and (b) are results of the semi-dense mapping without any aid from ground truth; (c) and (d) are results with ground truth pose, (e) and (f) are results with optical flow truth.

References

[1] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Trans. Pattern Anal. and Mach. Intell.,* vol. 40, no. 3, pp. 611-625, 2018.

[2] J. Engel, J. Sturm, and D. Cremers, "Semi-dense visual odometry for a monocular camera," in *Proc. IEEE Int. Conf. Comput. Vis.*, Sydney, 2013, pp. 1449-1456.

[3] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "Svo: Semidirect visual odometry for monocular and multicamera systems," *IEEE Trans. Robot.,* vol. 33, no. 2, pp. 249-265, 2017.

[4] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Trans. Robot.,* vol. 33, no. 5, pp. 1255-1262, 2017.

[5] S. Urban, S. Wursthorn, J. Leitloff, and S. Hinz, "MultiCol Bundle Adjustment: A Generic Method for Pose Estimation, Simultaneous Self-Calibration and Reconstruction for Arbitrary Multi-Camera Systems," *Int. J. Computer Vision,* vol. 121, no. 2, pp. 234-252, 2017.

[6] D. Valiente, A. Gil, L. Fernández, and Ó. Reinoso, "A comparison of EKF and SGD applied to a view-based SLAM approach with omnidirectional images," *Robot. Auton. Syst.,* vol. 62, no. 2, pp. 108-119, 2014.

[7] D. Valiente, M. G. Jadidi, J. V. Miró, A. Gil, and O. Reinoso, "Information-based view initialization in visual SLAM with a single omnidirectional camera," *Robot. Auton. Syst.,* vol. 72, pp. 93-104, 2015.

[8] D. Valiente, A. Gil, L. Payá, J. M. Sebastián, and Ó. Reinoso, "Robust Visual Localization with Dynamic Uncertainty Management in Omnidirectional SLAM," *Applied Sciences,* vol. 7, no. 12, p. 1294, 2017.

[9] M. Lourenço, V. Pedro, and J. P. Barreto, "Localization in indoor environments by querying omnidirectional visual maps using perspective images," in *Proc. IEEE Int. Conf. Robot. Autom.*, Saint Paul, 2012, pp. 2189-2195.

[10] H. Hadj-Abdelkader, E. Malis, and P. Rives, "Spherical image processing for accurate visual odometry with omnidirectional cameras," in *The 8th Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras-OMNIVIS*, Marseille, 2008.

[11] L. Heng, P. Furgale, and M. Pollefeys, "Leveraging Image‐based Localization for Infrastructure‐based Calibration of a Multi‐camera Rig," *J. Field Robotics,* vol. 32, no. 5, pp. 775-802, 2015.

[12] R. Lukierski, S. Leutenegger, and A. J. Davison, "Rapid free-space mapping from a single omnidirectional camera," in *Proc. Euro. Conf. Mobile Robots,* Lincoln, 2015, pp. 1-8: IEEE.

[13] Y. Berenguer, L. Payá, M. Ballesta, and O. Reinoso, "Position estimation and local mapping using omnidirectional images and global appearance descriptors," *Sensors,* vol. 15, no. 10, pp. 26368-26395, 2015.

[14] L. Payá, F. Amorós, L. Fernández, and O. Reinoso, "Performance of global-appearance descriptors in map building and localization using omnidirectional vision," *Sensors,* vol. 14, no. 2, pp. 3033-3064, 2014.

[15] D. Caruso, J. Engel, and D. Cremers, "Large-scale direct slam for omnidirectional cameras," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Systems (IROS)*, Hamburg, 2015, pp. 141-148.

[16] R. Mur-Artal and J. D. Tardós, "Probabilistic Semi-Dense Mapping from Highly Accurate Feature-Based Monocular SLAM," in *Robotics: Science and Systems*, Rome, 2015.

[17] C. Mei and P. Rives, "Single view point omnidirectional camera calibration from planar grids," in *Proc. IEEE Int. Conf. Robot. Autom.,* Roma, 2007, pp. 3945-3950.

[18] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g 2 o: A general framework for graph optimization," in *Proc. IEEE Int. Conf. Robot. Autom.*, Shanghai, 2011, pp. 3607-3613.

[19] H. Strasdat, A. J. Davison, J. M. Montiel, and K. Konolige, "Double window optimisation for constant time visual SLAM," in *Proc. IEEE Int. Conf. Comput. Vis.*, Barcelona, 2011, pp. 2352-2359.

[20] J. Maye, P. Furgale, and R. Siegwart, "Self-supervised calibration for robotic systems," in *Proc. IEEE Intell. Vehicles Symposium (IV),* Gold Coast, 2013, pp. 473-480.