Visual SLAM using Multiple RGB-D Cameras

Shaowu Yang, Xiaodong Yi, Zhiyuan Wang, Yanzhen Wang and Xuejun Yang

***Abstract*— In this paper, we present a solution to visual simultaneous localization and mapping (SLAM) using multiple RGB-D cameras. In the SLAM system, we integrate visual and depth measurements from those RGB-D cameras to achieve more robust pose tracking and more detailed environmental mapping in unknown environments. We present the mathematical analysis of the iterative optimizations for pose tracking and map refinement of a RGB-D SLAM system in multi-camera cases. The resulted SLAM system allows configurations of multiple RGB-D cameras with non-overlapping fields of view (FOVs). Furthermore, we provide a SLAM-based semi-automatic method for extrinsic calibration among such cameras. Finally, the experiments in complex indoor scenarios demonstrate the efficiency of the proposed visual SLAM algorithm.**

I. Introduction

In order to achieve more robust pose tracking of visual SLAM, previous work has made much effort in fusing information from multi-modal sensors, e.g. fusing inertial measurements [20], [26], [31], [11]. Recently, the robotics community has shown a growing interest in improving the performance of visual SLAM by utilising multiple cameras [16], [32], [9]. The reason for the above two trends is that pose tracking of a monocular visual SLAM system may easily fail in complex environments due to poor visual features which can be observed by one camera. This also applies to stereo visual SLAM and RGB-D SLAM, which employ cameras looking in one specific direction with limited FOVs. Actually, a larger effective FOV can be obtained by integrating multiple cameras looking into different directions, so that more reliable visual features can be observed to fulfil the pose tracking and map updating tasks. This implies that better pose tracking robustness can be achieved by extending monocular visual SLAM to utilise measurements from multiple cameras.

In this work, we propose using multiple RGB-D cameras for visual SLAM. We integrate depth measurements in multi-camera visual SLAM, and thus, we can benefit from both a multi-camera configuration and depth measurements provided by RGB-D cameras. Comparing with conventional cameras, RGB-D cameras, e.g. the Kinect from Microsoft, can provide direct depth measurements correspond to visual features. These measurements can be used for avoiding scale drift of a visual SLAM system [25] and for 3D occupancy grid mapping [4]. Meanwhile, we do not require each visual feature to have a depth measurement from a RGB-D camera.

Dr. S. Yang, Asso.Prof. Dr. X. Yi, Dr. Z. Wang, Dr. Y. Wang and Prof. Dr. X. Yang are with the State Key Laboratory of High Performance Computing (HPCL), National University of Defense Technology, Changsha, China {shaowu.yang, yixiaodong, wzy, yzwang, xjyang}@nudt.edu.cn

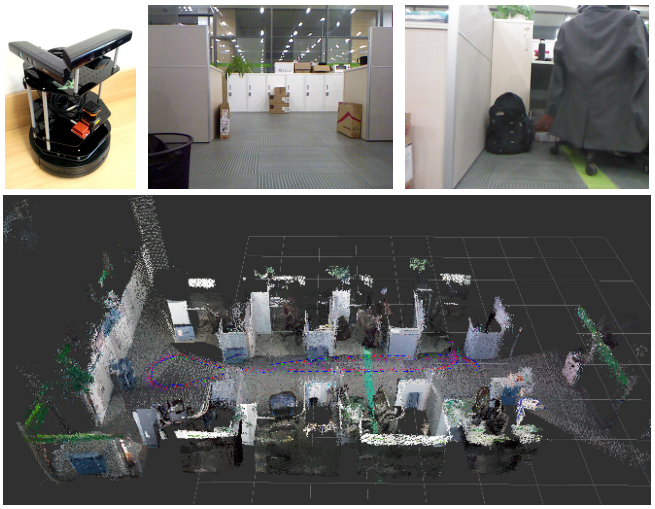


Fig. 1: Visual SLAM using two Kinects mounted on a mobile robot in a complex indoor environment with poor-textured glass walls and floor. Top left: The two Kinects mounted on a Turtlebot, one facing forward, and the other one facing the right side. Top middle and top right: A front camera view and a right-side camera view at the same time-stamp during an indoor navigation. Bottom: The built dense point cloud using the two Kinects in a SLAM process. The trajectory and keyframe poses are also plotted in this map.

Depth measurements are treated as additional constraints in the SLAM system, which is of great advantages when using those low-cost RGB-D cameras, since depth information is very likely to be unavailable for some regions of an image. Moreover, in this way, time costs introduced by integrating depth measurements can be nearly negligible.

To the best of our knowledge, no published work has utilised multiple RGB-D cameras with non-overlapping FOVs in a single SLAM system. Our method allows flexible configuration of RGB-D cameras on a robot, in order to obtain various useful perspectives without requiring overlaps in their FOVs, or requiring the cameras to be mounted in a specific way in order to keep a single-viewpoint model as some omnidirectional cameras. We integrate both visual and depth measurements from multiple RGB-D cameras, and present the mathematical analysis on how those measurements should be fused in the optimisations of visual SLAM, which is not a trivial issue since multiple cameras no longer preserve a single-viewpoint model.

II. Related Work

Early work related to pose estimation using multi-camera systems can be found in the context of structure from motion (SFM). The work in [21] presents a theoretical treatment of multi-camera systems in SFM deriving the generalized epipolar constraint. In [6], a virtual camera is proposed as a representation of a multi-camera system for pose estimation. A structure-from-motion scheme is achieved using multiple cameras in this work. A number of more recent work on multi-camera systems for pose estimation of mobile robots can be found in the literature. In [23], pose estimation of a mobile robot is solved by using two pairs of stereo cameras combined with the Extended Kalman Filter. The work in [16] adopts a generalized camera model for a multi-camera system, to estimate the ego-motion of a self-driving car using a 2-Point RANSAC scheme. In a further work in [18], this problem is solved with a minimal set of three-point correspondences.

Recently, multi-camera systems have been found in visual SLAM. The work in [13] presents a solution to visual SLAM with a multi-camera rig using Harris corner detector [7]. Further in [14], a Bayesian approach to data association is presented taking into account moving features which can be observed by cameras under robot motion. The work in [27] provides solutions to two different problems in multi-camera visual SLAM: automatic self-calibration of a stereo rig while performing SLAM and cooperative monocular SLAM.

Multi-camera visual SLAM is further implemented for autonomous navigation of micro aerial vehicles (MAVs). Our previous work in [32] utilises two cameras with non-overlapping FOV in visual SLAM for autonomous navigation of a MAV in complex environments based on the Parallel Tracking and Mapping (PTAM) system [15]. It proved that more robust pose tracking can be achieved by a multi-camera configuration in SLAM. In [29], multi-camera visual SLAM is achieved based on another modified version of PTAM. This SLAM system allows convergence in pose tracking and mapping with the absence of accurate metric scale, taking advantages of the Taylor omnidirectional camera model and a spherical coordinate update method. For autonomous navigation of MAVs in large scale environments, the work in [33] modifies the work in [32] to operate as a robust visual odometry, and implements an efficient back-end for loop closing. In [9], two stereo pairs are used to estimate the pose of an MAV based on a generalized camera model. Inertial information is used to recover the relative motion of the MAV with metric scale. Real-time loop closing runs on-board of MAVs is achieved in both [33] and [9]. In the work in [17], multiple cameras are also introduced to loop closing. In this work, the relative pose between two loop-closing pose-graph vertices is obtained from the epipolar geometry of a multi-camera system, without requiring the reconstruction of 3D scene points.

Although to the author's knowledge no published work has demonstrated SLAM using multiple RGB-D cameras, much work on SLAM using a single RGB-D camera can be found in the literature. In [10], the depth measurements of matched features between consecutive RGB images are used to determine their 3D positions, which serve as an initialisation to the more accurate Iterative Closest Point (ICP) algorithm. The SLAM system in [28] uses a two-level approach for efficient optimisation of the map. It also works with RGB-D cameras. The work in [25] utilises depth information in the PTAM system to solve the scale ambiguity and scale drift problems in monocular visual SLAM. The SLAM system in [5] utilises two planar mirrors to split the FOV of a RGB-D camera to cover both front and rear view of a mobile robot. The work in [30] perform RGB-D visual odometry on an MAV to enable its autonomous flight. 3D occupancy grid map is then built for path planning.

In order to use multiple cameras for pose estimation, extrinsic calibration among those cameras needs to be performed. Our interest on this aspect is mainly on automatic calibration of cameras with non-overlapping FOVs. The work in [3] proposes a SLAM-based automatic calibration scheme for multiple cameras. It uses global bundle adjustment to optimize the alignment of maps built by different visual SLAM instances, each of which corresponds to one camera. The proposed solution computes the relative 3D poses among cameras up to scale. In a more recent work in [8], automatic extrinsic calibration is achieved based on a previously built accurate map of the environment. The synchronised cameras to be calibrated moves in the mapped environment, so that a set of pose estimates of them can be obtained by 2D-3D correspondences, which are then utilised in a non-linear optimisation process to estimate the relative camera poses after an initial guess is found.

III. SLAM with Multiple RGB-D Cameras

A. The camera projection model and pose update

Using the same calibrated camera model as in [15], the image projection of the map point p_j to the camera C_i is

$$\mathbf{u}_{ji} = \mathcal{P}_{C_i}\left(E_{c_iw}\mathbf{p}_j\right), \tag{1}$$

where $\mathcal{P}_{C_i}$ is the projection function of the camera C_i considering lens distortion, $\mathbf{p}_j$ are the world coordinates of the map point p_j, and E_{c_iw} is a member of the Lie group SE(3), which represents the camera pose in the world coordinate system, containing a rotation and a translation component.

In a multi-camera system, we compute the pose update of one specific camera C_1, based on measurements from all cameras, as illustrated in Fig. 2a. The pose of other cameras can be updated by assuming constant transformations relative to C_1, i.e. pose updates of all cameras can be expressed with one single six-element vector μ using the exponential map:

$$E'_{c_iw} = E_{i1} \cdot e^{\mu} \cdot E_{c_1w}, \tag{2}$$

where μ is an element of the Lie algebra $se(3)$, and E_{i1} is the pose of C_1 in the camera coordinate system of C_i. The pose tracking (and a part of mapping) problem of the SLAM system now mainly consists of how to obtain an optimized

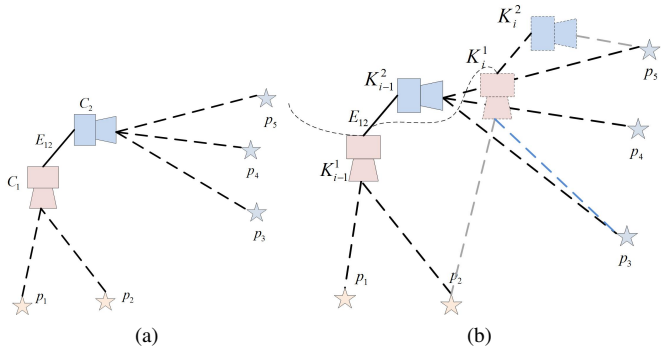


Fig. 2: (a) A 2D illustration of the camera pose updates in a dual-camera case and (b) four keyframes obtained by the dual cameras at two points in time. For camera-pose update, image measurements from all cameras are used to find the pose update of camera C_1. For bundle adjustment, image measurements in all keyframes are used to find the pose updates of keyframes $\mathbf{K}_1$ (K_{11}, K_{12}) and position updates of all measured measured map points.

μ as a pose update for camera C_1 by minimizing a certain objective function. The advantage of the parametrization of the camera pose updates using the six-element vector μ is that it allows a closed-form differentiation of Eq. 2.

B. The map

The map of the SLAM system consists of a set of keyframes $\mathbf{K}$ obtained by all the cameras in the vision system at various time-stamps and a set of 3D map points $\mathbf{P}$ measured by these cameras. A map point p_j may be located and added to the map by stereo triangulation of matched FAST corners [24] in successive keyframes from a certain camera, or by using direct depth measurement of the corresponding image feature. Each keyframe saves its absolute pose and all the observations to the map points. We allow possible observation of each map point by different cameras. Fig. 2b illustrates a map segment with four keyframes in a dual camera case.

C. Overview of the SLAM system

The implementation of our SLAM system is based on the PTAM framework. It mainly consists of two separate threads, in which images from multiple cameras are utilised: In the first thread responsible for camera pose tracking, fixed-range searches are applied to FAST features correspond to potentially observable map points in images from multiple cameras. The resulted 2D-3D correspondences are then used for camera-pose tracking. The second thread integrates new keyframes from multiple cameras to the map and creates new map points. Furthermore, local bundle adjustment is performed to refine the map.

D. Optimizations in SLAM

In our visual SLAM system, the camera pose tracking and map refinement (using bundle adjustment) are done by iteratively minimizing a robust objective function of the reprojection errors of sets of image measurements S_i, which are observed map points in each camera (or keyframe) i.

In an n-camera (or n-keyframe) system, we need to minimize the function

$$\sum_{i=1}^{n}\sum_{j\in S_i}\mathrm{Obj}\left(\frac{|\,\mathbf{e}_{ji}\,|}{\sigma_{ji}},\sigma_T\right), \tag{3}$$

where Obj is the Tukey biweight objective function [12], $|\,\mathbf{e}_{ji}\,|$ is the reprojection error of point j measured in camera (or keyframe) i, σ_{ji} is the estimated measurement noise of point j, and σ_T is a median-based robust standard-deviation estimate of the distribution of all reprojection errors. $\mathbf{e}_{ji}$ is defined as the difference between the image reprojection of map point j and its actual image measurement:

$$\mathbf{e}_{ji} = \mathbf{u}_{ji} - \hat{\mathbf{u}}_{ji}. \tag{4}$$

When RGB-D cameras are used, we utilise depth measurements D_i to the corresponding image observations in S_i for bundle adjustment similarly as in [25]. However, since measurement errors in depth and image pixels cannot be modeled in the same unit, we model the function to be minimised in a unified form as

$$F(\mathbf{C},\mathbf{S},\mathbf{D}) = \sum_{i=1}^{n}\sum_{j\in S_i}\mathrm{Obj}\left(\frac{|\,\mathbf{e}_{ji}\,|}{\sigma_{ji}},\sigma_T\right) + \sum_{i=1}^{n}\sum_{j\in D_i}\mathrm{Obj}\left(\frac{|\,e^d_{ji}\,|}{\sigma^d_{ji}},\sigma_D\right), \tag{5}$$

where $\mathbf{C}$ includes the n cameras (in pose tracking) or the n keyframes (in bundle adjustment), $\mathbf{S}$ and $\mathbf{D}$ are the sets of S_i and D_i, respectively. e^d_{ji} is the depth measurement error of d_{ji}, which will be analysed in Sect. III-E. This function does not require each observation in S_i to have a depth measurement.

For the pose tracking of an n-camera system, the optimization problem is to find the optimal pose update μ for camera C_1 using image measurements by all the cameras at the current time-stamp t:

$$\mu' = \underset{\mu}{\mathrm{argmin}} F(\mathbf{C}_t,\mathbf{S}_t). \tag{6}$$

Then the pose of all cameras can be derived by Eq. 2.

Local bundle adjustment is used to refine the recent keyframe poses and related map point positions in [15]. In our work, image measurements from all cameras are utilised to compute the optimal pose updates of the keyframe set $\mathbf{K}_1$ which are obtained by the first camera C_1. The poses of other rigidly connected keyframes $\mathbf{K}_c$ are computed based on the updated poses of their associated keyframes in $\mathbf{K}_1$, as illustrated in Fig. 2b.The positions of map points $\mathbf{P}_a$ which are measured in $\mathbf{K}_a$ or $\mathbf{K}_c$ are also updated in the bundle adjustment. Thus, the local bundle adjustment in our SLAM system means solving the following minimization problem:

$$\{\{\mu_{i\in\mathbf{K}_1}\},\{\mathbf{p}_{j\in\mathbf{P}_a}\}\} = \underset{\{\{\mu\},\{\mathbf{p}\}\}}{\mathrm{argmin}} F(\mathbf{C}_l,\mathbf{S}_l,\mathbf{D}_l). \tag{7}$$

Here, $\mathbf{C}_l = \mathbf{K}_a \cup \mathbf{K}_c \cup \mathbf{K}_f$. $\mathbf{K}_f$ is the further fixed keyframe set containing keyframes in which a measurement of any point in $\mathbf{p}_a$ has been made. $\mathbf{S}_l$ is the image measurements for $\mathbf{P}_a$.

$\mathbf{D}_l$ includes all the depth measurements that can be obtained for $\mathbf{P}_a$.

E. The Jacobians

The above minimization problems can be solved by iterations of reweighted nonlinear least squares. One fundamental requirement to do this efficiently is to differentiate measurement errors, i.e. to obtain the Jacobians of them, with respect to those parameters that need to be estimated. In pose tracking, the derivatives of $\mathbf{e}_{ji}$ with respect to the estimated camera pose update μ need to be computed. In bundle adjustment, the derivatives of $\mathbf{e}_{ji}$ and e^d_{ji} with respect to μ and the map point position $\mathbf{p}_j$ are also required. In [32], we provided the derivatives of $\mathbf{e}_{ji}$ for multiple conventional cameras. In this section, we extend the work to analysis depth measurement errors from RGB-D cameras.

For a map point j measured by the camera C_1, we can compute the Jacobian matrix of $\mathbf{e}_{ji}$ with respect to the estimated C_1 pose update μ at $\mu = 0$ using the chain rule as

$$\mathbf{J}_{1\mu} = \frac{\partial \mathcal{P}_{C_1}\left(e^{\mu} E_{c_1 w}\mathbf{p}_j\right)}{\partial \mu} = \left.\frac{\partial \mathcal{P}_{C_1}(\mathbf{c})}{\partial \mathbf{c}}\right|_{\mathbf{c}=E_{c_1 w}\mathbf{p}_j} \cdot \frac{\partial\left(e^{\mu} E_{c_1 w}\mathbf{p}_j\right)}{\partial \mu}. \tag{8}$$

The first term of the above matrix product is the Jacobian of the camera projection function in Eq. 1, and the last term is:

$$\frac{\partial\left(e^{\mu} E_{c_1 w}\mathbf{p}_j\right)}{\partial \mu} = \begin{pmatrix}\mathbf{I}_3 & -[E_{c_1 w}\mathbf{p}_j]_{\times}\end{pmatrix}. \tag{9}$$

However, for map points measured by other cameras, with Eq. 2, the differentiation becomes:

$$\mathbf{J}_{i\mu} = \frac{\partial \mathcal{P}_{C_i}\left(E_{i1} e^{\mu} E_{c_1 w}\mathbf{p}_j\right)}{\partial \mu} = \left.\frac{\partial \mathcal{P}_{C_i}(\mathbf{c})}{\partial \mathbf{c}}\right|_{\mathbf{c}=E_{c_i w}\mathbf{p}_j} \cdot \frac{\partial\left(E_{i1} e^{\mu} E_{c_1 w}\mathbf{p}_j\right)}{\partial \mu}. \tag{10}$$

Its difference to Eq. 8 lies in the last term of this equation:

$$\frac{\partial\left(E_{i1} e^{\mu} E_{c_1 w}\mathbf{p}_j\right)}{\partial \mu} = Rot(E_{i1}) \cdot \begin{pmatrix}\mathbf{I}_3 & -[E_{c_1 w}\mathbf{p}_j]_{\times}\end{pmatrix}, \tag{11}$$

where $Rot(E_{i1})$ represents the rotation component of E_{i1}. The Jacobians are taken at the vicinity of the present estimates of the camera poses or point positions.

The Jacobian of $\mathbf{e}_{ji}$ with respect to the estimated point j pose can be expressed in a consistent way, regardless of the camera used to measure this point:

$$\mathbf{J}_{\mathbf{p}_j} = \frac{\partial \mathcal{P}_{C_i}\left(E_{c_i w}\mathbf{p}_j\right)}{\partial \mathbf{p}_j} = \left.\frac{\partial \mathcal{P}_{C_i}(\mathbf{c})}{\partial \mathbf{c}}\right|_{\mathbf{c}=E_{c_i w}\mathbf{p}_j} \cdot \frac{\partial\left(E_{c_i w}\mathbf{p}_j\right)}{\partial \mathbf{p}_j}. \tag{12}$$

The last term is:

$$\frac{\partial\left(E_{c_i w}\mathbf{p}_j\right)}{\partial \mathbf{p}_j} = Rot\left(E_{c_i w}\right). \tag{13}$$

When the depth measurement to a map point is available, we can derive its error by beginning with the 3D-measurement error in the camera coordinate system:

$$\mathbf{e}'_{ji} = \mathbf{p}_{ji} - \hat{\mathbf{p}}_{ji}, \tag{14}$$

so that the camera projection model in Eq. 1 can be ignored. Now, the resulted Jacobians of $\mathbf{e}'_{ji}$ in Eq. 8, 10, and 12 only remain their last terms. Since e^d_{ji} in Eq. 5 is simply the z-axis value in $\mathbf{e}'_{ji}$, the Jacobians of e^d_{ji} are equivalent to the last row of Eq. 9, 11, and 13, respectively.

IV. Relocalization

A relocalization module is commonly required in SLAM systems due to possible tracking failures. In our multi-camera system, current images from all those cameras can be used for this purpose. If pose tracking fails, we use EPnP [19] in a standard RANSAC scheme to locate the current image frame to its most recent keyframe K_r obtained by the same camera. This process iterates through all cameras until a proper hypothesis is achieved. BRIEF descriptors [2] of all extracted FAST features after maximal suppression are used to match the two frames to obtain a set of 2D-3D correspondences.

V. Experiments and Results

A. Experiment setup

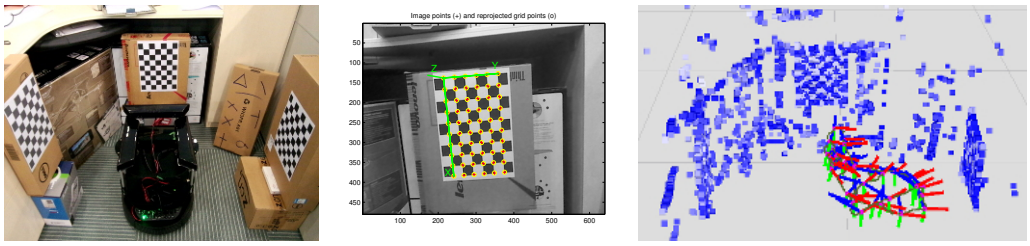
In the experiment, a Turtlebot platform with two Kinect sensors as shown in Fig. 1 is used for logging RGB-D images. The main reason for the choice of the count of Kinects is that the computer on our Turtlebot has only two USB 2.0 buses, which allows at most two pairs of RGB-D images to be transported in parallel. Measurement error distributions of the Kinect sensor as calibrated in [25] is applied to the SLAM system. Our SLAM system runs on a laptop equipped with an Intel i3 core 2.4 GHz CPU and a 6 GB RAM. The SLAM system is implemented in the Robot Operating System (ROS) [22] in Ubuntu.

B. Semi-automatic camera extrinsic calibration

Since the different RGB-D cameras in our vision system can be mounted without overlapping in their FOVs, standard extrinsic calibration methods for stereo cameras cannot be applied. In [32], an external pose tracking system is used to facilitate the calibration. In this paper, we propose a semi-automatic calibration method based on visual SLAM without requiring expensive external facilities.

As shown in Fig. 3a, we fix a planar pattern P_i, which is used for monocular-camera calibrations using the Matlab-toolbox proposed in [1], in front of each camera C_i to be calibrated. P_i defines its own coordinate frames W_i. After C_i takes an image of P_i, the accurate pose T^p_i of C_i in W_i can be obtained by performing extrinsic calibration using the Matlab toolbox. Then we perform the SLAM system proposed in this paper with a single Kinect C_s around those planar patterns, its pose will be accurately tracked in SLAM. Thus, when we do the same calibration process for C_s as to C_i in the above, the absolute pose T^w_{pi} of each pattern P_i in the SLAM coordinate frames W can be obtained by a coordinate transformation. Therefore, the absolute pose of camera C_i in W is

$$T^w_i = T^w_{pi} \cdot T^p_i. \tag{15}$$



(a) (b) (c)

Fig. 3: Semi-automatic extrinsic calibration among multiple RGB-D cameras. (a) Experimental setup of Kinects and planar patterns. (b) A planar pattern viewed by one Kinect. The extrinsic calibration results using the Matlab toolbox are also attached in. (c) The map built by the single-Kinect SLAM system, and the keyframe poses marked as RGB-axes.

Finally, the relative pose of camera C_1 to another camera C_j can be derived as $T_{1j} = T_1^{wT} \cdot T_j^w$.

An exemplary image of a planar pattern taken by a Kinect is shown in Fig. 3b, in which the extrinsic calibration results using the Matlab toolbox are also attached. The map built by the SLAM system during the calibration process is shown in Fig. 3c, in which the keyframe poses are illustrated as RGB-axes. A drawback of our method compared to a fully automatic method, e.g. the method in [8], is that our method relies on pose estimates obtained in a single time point as mentioned above. As a result, the calibration performance may be affected by measurement noises, especially the pose tracking errors in SLAM. In order to reduce such errors, environments with rich visual features are highly recommended for the SLAM process, e.g. the environment shown in Fig. 3b. An alternative way to obtaining T_i^p manually is to use augmented reality tags to estimate camera poses automatically. However, we chose the current way for its accuracy and convenience.

C. Localisation and mapping results

We manually drive our Turtlebot in our office in HPCL, and take a video logfile of RGB-D images from the two Kinects as a ROS bag. As can be found in Fig. 1, very rare reliable visual feature can be found on the ground and glass walls. Reflections of glass walls around the office even introduce more noises to visual SLAM systems. To evaluate the localisation and mapping results, we replay the logfile offline to serve as image input to our SLAM system. The performance of the SLAM system using dual Kinects is compared to that using a single Kinect. Since the Kinect mounted towards the side of the robot can often obtain too few reliable features for pose tracking by its own in this scenario, we do not consider it in the single-Kinect SLAM system.

The trajectory of the Turtlebot during this navigation is shown in Fig. 4a, in which pose estimates by SLAM with the dual Kinects (DK) are plotted in red, and results with a single forward-looking Kinect (SK) in green. A segment of the trajectory marked in black rectangle in Fig. 4a is enlarged and shown in Fig. 4b.

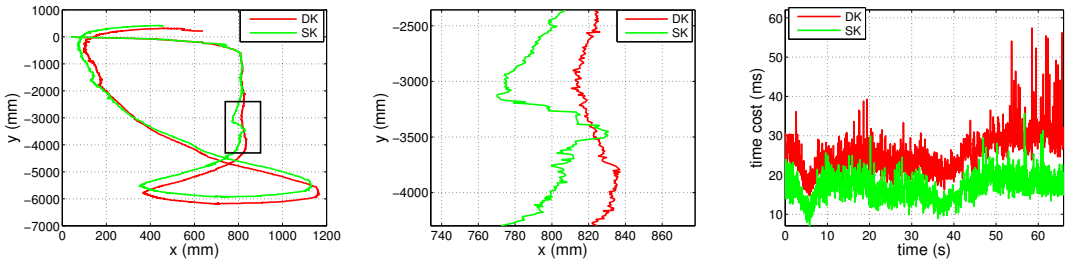
Before comparing to ground truth data, we can evaluate the performance of the SLAM system in the following qualitative way: Since we control our robot in a smooth way when we take logfiles, we can find that the trajectory estimated with the dual Kinect is more close to the actual movement of the robot, while the pose estimates plotted in green show obvious vibrations. Such vibrations normally happen when no enough good visual features can be tracked for pose estimation. When using multiple cameras, less chances of such situations may happen to the SLAM system.

Time costs of the pose tracking thread in the two camera configurations are shown in Fig. 4c. One major factor affecting the time cost of the pose tracking thread is the count of FAST features detected in the images. Moreover, since we are not using a real-time Operating System, vibrations in time cost will happen even for processes with very similar computation complexities. Furthermore, as the size of the map grows, time cost of the pose tracking thread will slightly grow, since all map points will be tried to re-project to the current image frames, to check whether they are potentially visible.

Fig. 5 shows the map and the point cloud built by our SLAM system using dual Kinects from two viewing angles. The trajectory and keyframes during this navigation are also plotted in those figures. The SLAM system with two Kinects produces an accurate environment dense point-cloud as shown in Fig. 5a. This dense point cloud provides more detailed information of the environment than a single Kinect could achieve. The sparse map points produced by the color cameras of the two Kinects are shown in different colors in Fig 5b: The points in blue are obtained by the camera facing forward of the robot, and the points in red by the camera facing right side. Our SLAM method allows those points to be measured by both cameras in both pose tracking and bundle adjustment processes.

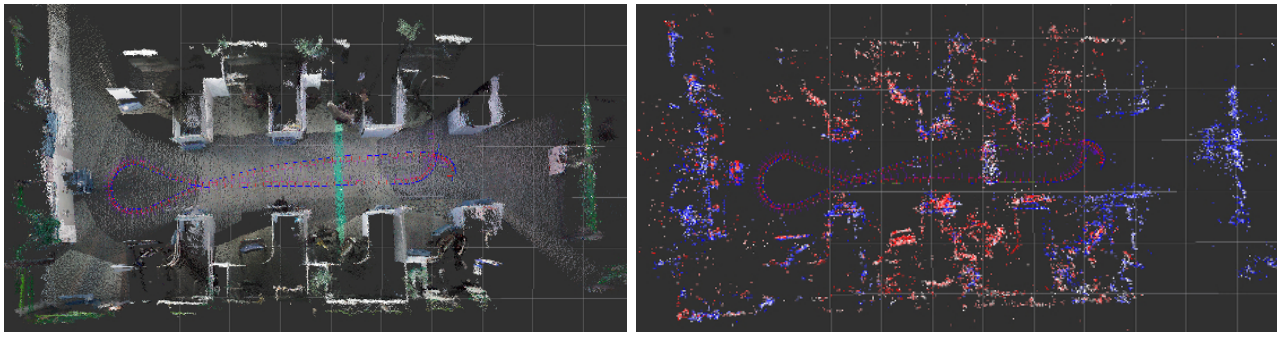
D. Comparisons with ground truth data

In this experiment, accurate pose estimates from a 2D laser-based SLAM system are used as ground truth data for further evaluations. Our Turtlebot is driven in a similar way as in the last experiment, in the same room covering slightly larger area.



(a) (b) (c)

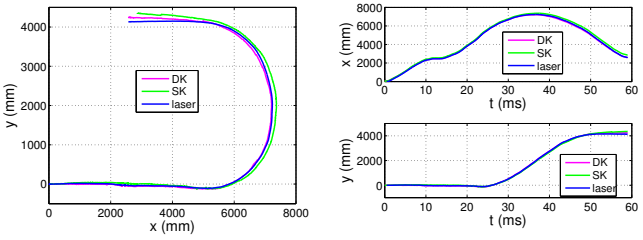
Fig. 4: The trajectory of the Turtlebot and time costs of our SLAM system in the experiment. (a) The full trajectory during the navigation. (b) The trajectory segment corresponds to the part of trajectory marked in black rectangle in (a). (c)Time costs of the pose tracking thread using the two different camera configurations.



(a) (b)

Fig. 5: The point cloud (a) and sparse map (b) built by our visual SLAM system using two Kinects, viewing from a top perspective. Another perspective of the point cloud is shown in Fig. 1.

Fig. 6 shows the trajectories of the Turtlebot estimated with different sensor setups during a same run of the robot. Position estimates from our SLAM system using both dual Kinects and a single forward-looking Kinect fit well to the ground truth data provided by the laser-based SLAM system (laser). The root-mean-square errors (RMSEs) of estimates from the dual-Kinect RGB-D SLAM system during the whole run are (4.5,4.2) *centimeters* in the $x-y$ directions, while RMSEs of that from the single-Kinect RGB-D SLAM system are (16.8,9.7) *centimeters*. Using the dual-Kinect SLAM method provides better pose tracking performance in this run. Again, this can be explained by that more reliable visual features can be obtained using the dual Kinects. Lacking of reliable visual features in the single-Kinect system could introduce larger pose drifts, which will be accumulated during the whole operation.



(a) (b)

Fig. 6: The trajectories of the Turtlebot measured by our SLAM system using dual Kinects (DK), single forward-looking Kinect, and by a laser-based SLAM system (laser). (a) Trajectories on the $x-y$ plane, (b) pose estimates on the x–axis and (c) on the y–axis. The SLAM process using the dual Kinects during this run is illustrated in the accompanying video.

VI. Conclusions and Discussions

In this paper, we propose using multiple RGB-D cameras in visual SLAM for better pose tracking performance and more detailed environment mapping. The mathematical analysis in this paper explains how visual and depth measurements from those cameras could be fused into one single SLAM system by solving certain optimisation problems. In the experiments, we finally use two Kinects in SLAM, due to limitations of the available hardware. The experiments with image logfiles taken from a ground vehicle has demonstrated the theory presented in this work. Another contribution of this paper is that we make the source code related to this work publicly available as a ROS package

online at https://github.com/ShaowuYang/idSLAM. A video demonstration of this work can be found in the accompanying video, or at https://youtu.be/yDwH5TNlsRQ and http://v.youku.com/v_show/id_XMTI5NzA2MjYyOA.

Recent work would be implementing an efficient backend for loop closing, in which images from multiple cameras should be considered. Furthermore, since synchronising multiple Kinect sensors is not practical, to utilise them for more accurate SLAM, future work could be compensating feature measurements among different cameras in the image space based on a tracked camera motion model. Further work related to camera extrinsic calibration could be done in a fully automatic way similarly to the work in [8] and provide an open-source toolbox which can be easily handled.

VII. Acknowledgment

The authors would like to thank Prof. Andreas Zell and Sebastian A. Scherer, from the Department of Computer Science, University of Tuebingen, for helpful discussions and providing us the source code related to the work in [25].

This work is supported by Research on Foundations of Major Applications, Research Programs of NUDT, Project ZDYYJCYJ20140601, and NSFC Project 61303185, 61403409.

References

[1] J.Y Bouguet. Camera calibration toolbox for matlab. *http://www.vision.caltech.edu/bouguetj/calib_doc*, 2001.

[2] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. *Computer Vision–ECCV 2010*, pages 778–792, 2010.

[3] G. Carrera, A. Angeli, and A.J. Davison. SLAM-based automatic extrinsic calibration of a multi-camera rig. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2652–2659, 2011.

[4] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard. An evaluation of the RGB-D SLAM system. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1691–1696, May 2012.

[5] Felix Endres, Christoph Sprunk, Rainer Kummerle, and Wolfram Burgard. A catadioptric extension for rgb-d cameras. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 466–471. IEEE, 2014.

[6] Jan-Michael Frahm, Kevin Köser, and Reinhard Koch. Pose estimation for multi-camera systems. In *Pattern Recognition*, volume 3175 of *LNCS*, pages 286–293. Springer Berlin Heidelberg, 2004.

[7] Chris Harris and Mike Stephens. A combined corner and edge detector. In *the 4th Alvey Vision Conference*, volume 15, pages 147–151. Manchester, UK, 1988.

[8] Lionel Heng, Mathias Burki, Gim Hee Lee, Paul Furgale, Roland Siegwart, and Marc Pollefeys. Infrastructure-based calibration of a multi-camera rig. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 4912–4919. IEEE, 2014.

[9] Lionel Heng, Gim Hee Lee, and Marc Pollefeys. Self-calibration and visual slam with a multi-camera system on a micro aerial vehicle. In *Proceedings of Robotics: Science and Systems*, Berkeley, USA, July 2014.

[10] Peter Henry, Michael Krainin, Evan Herbst, Xiaofeng Ren, and Dieter Fox. Rgb-d mapping: Using depth cameras for dense 3d modeling of indoor environments. In *In the 12th International Symposium on Experimental Robotics (ISER*. Citeseer, 2010.

[11] Joel A Hesch, Dimitrios G Kottas, Sean L Bowman, and Stergios I Roumeliotis. Camera-imu-based localization: Observability analysis and consistency improvement. *The International Journal of Robotics Research*, 33(1):182–201, 2014.

[12] PeterJ. Huber. Robust statistics. In Miodrag Lovric, editor, *International Encyclopedia of Statistical Science*, pages 1248–1251. Springer Berlin Heidelberg, 2011.

[13] Michael Kaess and Frank Dellaert. Visual SLAM with a multi-camera rig. Technical report, 2006.

[14] Michael Kaess and Frank Dellaert. Probabilistic structure matching for visual SLAM with a multi-camera rig. *Computer Vision and Image Understanding*, 114(2):286 – 296, 2010.

[15] Georg Klein and David Murray. Parallel tracking and mapping for small AR workspaces. In *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, Nara, Japan, November 2007.

[16] Gim Hee Lee, Friedrich Faundorfer, and Marc Pollefeys. Motion estimation for self-driving cars with a generalized camera. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 2746–2753. IEEE, 2013.

[17] Gim Hee Lee, F. Fraundorfer, and M. Pollefeys. Structureless pose-graph loop-closure with a multi-camera system on a self-driving car. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 564–571, Nov 2013.

[18] Gim Hee Lee, Bo Li, Marc Pollefeys, and Friedrich Fraundorfer. Minimal solutions for the multi-camera pose estimation problem. *The International Journal of Robotics Research*, 2015.

[19] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. *International journal of computer vision*, 81(2):155–166, 2009.

[20] Mingyang Li and Anastasios I. Mourikis. High-precision, consistent ekf-based visual–inertial odometry. *The International Journal of Robotics Research*, 32(6):690–711, 2013.

[21] R. Pless. Using many cameras as one. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–587–93 vol.2, June 2003.

[22] Morgan Quigley, Brian Gerkey, Ken Conley, Josh Faust, Tully Foote, Jeremy Leibs, Eric Berger, Rob Wheeler, and Andrew Ng. ROS: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, 2009.

[23] Mohammad Ehab Mohammad Ragab. *Multiple Camera Pose Estimation*. PhD thesis, The Chinese University of Hong Kong (People's Republic of China), 2008. AAI3348872.

[24] E. Rosten and T. Drummond. Machine Learning for High-Speed Corner Detection. In *European Conference on Computer Vision (ECCV)*, pages 430–443. Springer, 2006.

[25] S.A. Scherer, D. Dube, and A. Zell. Using depth in visual simultaneous localisation and mapping. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 5216–5221, May 2012.

[26] S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar. Vision-based state estimation for autonomous rotorcraft MAVs in complex environments. In *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, Karlsruhe, Germany, may 2013.

[27] J. Solà, A. Monin, M. Devy, and T. Vidal-Calleja. Fusing monocular information in multicamera SLAM. *Robotics, IEEE Transactions on*, 24(5):958–968, Oct 2008.

[28] H. Strasdat, A.J. Davison, J. M M Montiel, and K. Konolige. Double window optimisation for constant time visual SLAM. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2352–2359, 2011.

[29] Michael J. Tribou, Adam Harmat, David W.L. Wang, Inna Sharf, and Steven L. Waslander. Multi-camera parallel tracking and mapping with non-overlapping fields of view. *The International Journal of Robotics Research*, 2015. Online.

[30] Roberto G Valenti, Ivan Dryanovski, Carlos Jaramillo, Daniel Perea Strom, and Jizhong Xiao. Autonomous quadrotor flight using onboard rgb-d visual odometry. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 5233–5238. IEEE, 2014.

[31] Stephan Weiss, Markus W. Achtelik, Simon Lynen, Michael C. Achtelik, Laurent Kneip, Margarita Chli, and Roland Siegwart. Monocular vision for long-term micro aerial vehicle state estimation: A compendium. *Journal of Field Robotics*, 30(5):803–831, 2013.

[32] Shaowu Yang, Sebastian A. Scherer, and Andreas Zell. Visual SLAM for autonomous MAVs with dual cameras. In *2014 International Conference on Robotics and Automation (ICRA'14)*, pages 5227–5232, Hongkong, China, June 2014.

[33] Shaowu Yang, SebastianA. Scherer, and Andreas Zell. Robust onboard visual slam for autonomous mavs. In *Intelligent Autonomous Systems 13*, volume 302 of *Advances in Intelligent Systems and Computing*, pages 361–373. Springer International Publishing, 2016.