Proceedings of the 2017 IEEE
International Conference on Robotics and Biomimetics
December 5-8, 2017, Macau SAR, China

Camera-Odometer Calibration and Fusion using Graph Based Optimization

Yijia He[1,2], Yue Guo[1,2], Aixue Ye[1,2], and Kui Yuan[1,2]

Abstract— **Monocular visual odometry (vo) estimates the camera motion only up to a scale which is prone to localization failure when the light is changing. The wheel encoders can provide metric information and accurate local localization. Fusing camera information with wheel odometer data is a good way to estimate robot motion. In such methods, calibrating camera-odometer extrinsic parameters and fusing sensor information to perform localization are key problems. We solve these problems by transforming the wheel odometry measurement to the camera frame that can construct a factor-graph edge between every two keyframes. By building factor graph, we can use graph-based optimization technology to estimate camera-odometer extrinsic parameters and fuse sensor information to estimate robot motion. We also derive the covariance matrix of the wheel odometry edges which is important when using graph-based optimization. Simulation experiments are used to validate the extrinsic calibration. For real-world experiments, we use our method to fuse the semi-direct visual odometry (SVO) with wheel encoder data, and the results show the fusion approach is effective.**

I. INTRODUCTION

The indoor mobile robot becomes more and more popular, people like to use the mobile robot to clean the room or deliver food. However, most of the robot can only move in a random walk way or follow pre-set trajectory. Fusing variety of sensors to do accurate localization is the premise that the robot can complete more complex tasks.

Since most of the mobile robot is equipped with wheel encoders, the basic way to locate the robot is using wheel odometry [1]. However, it can only provide short term local localization even though combined with Inertial Measurement Units (IMUs) as the accumulated error can not be eliminated. A mobile robot equipped with laser scanner can provide accurate localization with sub-decimeter error [2]. However, the laser scanner is expensive for a service mobile robot. Recently, many significant Visual Odometry (VO) method or Visual Simultaneous Localization and Mapping (VSLAM) method have been proposed [3], [4], [5]. Especially, the semi-direct visual odometry (SVO) can runs at 55 frames per second on embedded platform [3]. But, monocular VO or VSLAM method estimate the camera motion only up to a scale factor [6], and VO is easy to locate failure when the light changing. Considering the wheel encoder provide metric information and have a high local localization accuracy. Combine the monocular camera with the wheel encoder to estimate the robot motion is a good way.

In order to fuse the information from camera and wheel encoder, the rotation and translation between the camera frame and the robot frame need to be estimated firstly. Antonelli et al. [7] use the planar marker to solve the extrinsic calibration problem. As the marker required, this method needs to prepare marker that not convenient for the user. The works of [8] and [9] uses the motion measurements from visual odometry and wheel odometry to calibrate extrinsic directly. Guo et al. [8] provide an analytical least-squares solution to the calibration problem. the calibration method of [9] is similar to [8], they refine the estimated extrinsic parameters by using the visual landmarks. However, in the work [9], they do not treat the calibration problem as a factor graph and do not consider the covariance matrix of the wheel odometry measurements.

For sensor fusion problem, there are two different methods: filter based methods [10], and graph based methods [11], [12]. Moore et al. [10] use the Extended Kalman Filter (EKF) to fuse three type of information from the camera, IMU, wheel encoder. Since the EKF methods process a measurement only once, the graph based method do batch optimization allows linearizing multiple times [13]. Li et al.[11] construct a factor graph to calibrate extrinsic parameters online and locate the robot. However, their work based on stereo camera, the scale factor does not need to be considered.

The main contribution of our work is that we transform the wheel odometry measurement to a virtual measurement for the camera. we use the virtual measurement to construct a factor-graph edge between every two keyframes so that can use graph-based optimization to estimate camera-odometer extrinsic parameters and fuse sensor information. We also derive the covariance matrix of the wheel odometry edges which is important when using graph-based optimization. Simulation experiments are used to validate the extrinsic calibration. For the real-world experiments, we combine the wheel odometry with semi-direct visual odometry (SVO) by using our method, the results from real-world experiments show the fusion approach is effectiveness.

II. PRELIMINARIES

In this paper, we solve the camera-odometer extrinsic calibration and sensor fusion problem by using graph based optimization. In this section, before go to our method, firstly we introduce the relevant geometry relationship between each frame, then we introduce the camera pin-hole model

*This work was supported by National Natural Science Foundation of China No.61421004

Yijia He, Yue Guo, Aixue Ye, and Kui Yuan are with Institute of Automation, Chinese Academy of Sciences and University of Chinese Academy of Sciences, Beijing 100190, China(email: { heyijia2013, guoyue2013, aixue.ye, kui.yuan }@ia.ac.cn)

978-1-5386-3742-5/17/$31.00 © 2017 IEEE

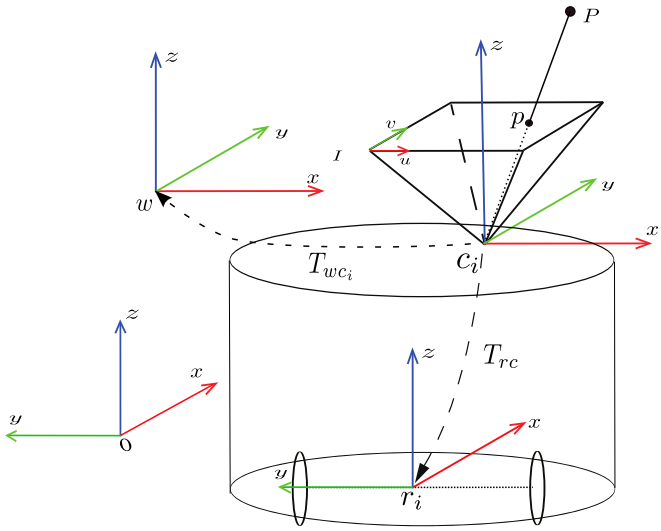

Fig. 1. The coordinates system.

and wheel odometry model, lastly we give a brief view of graph based optimization on Lie-manifolds.

A. Notations of Geometry

As shown in Fig. 1, the camera frame and robot frame are represented by lower case letters $\{c\}$ and $\{r\}$. At time step $t = 0$, we set the camera frame as the world frame $\{w\}$, the robot frame as the wheel odometry frame $\{o\}$. The 3D landmark $\mathbf{P}$ in the world frame project to camera image plane with $\mathbf{p}$.

A 3D rigid body motion $\mathbf{T} \in SE(3)$ is defined by

$$T = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \tag{1}$$

with $\mathbf{R} \in SO(3)$ is a 3×3 rotation matrix and $\mathbf{t} = (t_x, t_y, t_z)^T$ is a 3×1 vector represent a translation in 3D. Monocular camera is mounted on robot platform, the transformation from camera frame to the robot frame is represented by a 3D rigid body transform $\mathbf{T}_{rc}$. Considering the monocular camera estimate the motion only up to a scale s, we use a 3D similarity transformation $\mathbf{S} \in Sim(3)$, defined by (2), to build the relationship between camera motion segment $T_{c_i c_j}$ and odometer motion segment $T_{r_i r_j}$.

$$\mathbf{S} = \begin{bmatrix} s\mathbf{R} & \mathbf{t} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \tag{2}$$

B. Model of Sensor

1) Camera pinhole model: Given a 3D point in the camera frame $\mathbf{P}_c = (X_c, Y_c, Z_c)^T$ and the corresponding 2d point $\mathbf{p} = (u, v)^T$ in image plan I, we can use the pin-hole camera model to project the 3D point to 2d image plan. The projection function is defined by

$$\begin{aligned} \mathbf{p} &= \pi(\mathbf{P}_c) \\ &= (\frac{X_c f_x}{Z_c} + c_x, \frac{Y_c f_y}{Z_c} + c_y)^T \end{aligned} \tag{3}$$

where f_x, f_y are the focal lengths of camera on x axis and y axis, and $(c_x, c_y)^T$ is the coordinate of image plane center. If a 3D point is in the world frame $\mathbf{P}_w = (X_w, Y_w, Z_w)^T$, we can transform the point from the world frame to the camera frame with the inverse of camera pose $\mathbf{T}_{cw}$, i.e.

$$\mathbf{P}_c = T(\mathbf{T}_{cw}, \mathbf{P}_w) = \mathbf{R}_{wc}\mathbf{P}_w + \mathbf{t}_{wc} \tag{4}$$

combine the transformation process and projection process, we can map a 3D point in world frame to image plane.

2) Odometry model: Odometer use the wheel encoder data to estimate the robot motion. The measurement model of wheel encoder is

$$\begin{aligned} v_m &= v + n_v \\ \omega_m &= \omega + n_\omega \end{aligned} \tag{5}$$

where v_m and ω_m are measurements of robot's linear and angular velocity, $\eta^n = [n_v, n_\omega]^T$ is Gaussian white noise with known covariance $\mathbf{Q}$ [1]. Integrate the velocity measurements over time to give the robot postition and heading in wheel odometry frame, i.e.

$$\begin{aligned} x_{i+1} &= x_i + v_m \Delta t \cos\theta_i \\ y_{i+1} &= y_i + v_m \Delta t \sin\theta_i \\ \theta_{i+1} &= \theta_i + \omega_m \Delta t \end{aligned} \tag{6}$$

Since the mobile robot move on a plane, there are only three degrees of freedom, $\eta = (x, y, \theta)^T$, one degree for rotaion and two degrees for tranlation. The covariance of wheel odometry measurement $\mathbf{\Gamma}_{ii+1}$ is a 3×3 matrix. Considering most of the open source visual odometry method estimate the camera motion with 3D rigid body transformation (6 dof), we extend the motion estimated by wheel odometry to a 3D transformation with zero translation measurement in the z-coordinate and zero rotation measurement about the x-axis and y-axis.

C. Graph Based Optimization on Lie-manifolds

Many problems in robotics can be solved by formulate a graph, where nodes in the graph is a vector of parameters need be estimated and edges between nodes are the sensor measurements. i.e.

$$\mathbf{z}_{ij} = \mathbf{h}_{ij}(\mathbf{x}_i, \mathbf{x}_j) + \mathbf{n}_{ij} \tag{7}$$

where $\mathbf{x}_i$ are parameters at time step i need to be estimated. $\mathbf{z}_{ij}$ is measurement between nodes $\mathbf{x}_i$ and $\mathbf{x}_j$ that corrupted by zero-mean Gaussian white noise, $\mathbf{n}_{ij} \in \mathcal{N}(0, \mathbf{\Sigma}_{ij})$. $\mathbf{h}_{ij}$ is a measurement function can be used to map the nodes $\mathbf{x}$ to the predicated measurement. Therefore, the error fucntion of a measurement can be defined by

$$\mathbf{e}_{ij} = \mathbf{z}_{ij} - \mathbf{h}_{ij}(\mathbf{x}_i, \mathbf{x}_j) \tag{8}$$

we construct an edge (constraint) from the two nodes, this type of edge called binary edge. An edge connected with a node which called unitary edge. Since the noise is subject to normal distribution, the parameters can be solved by minimizing the cost function:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \sum_{<i,j>\in\mathcal{C}} \mathbf{e}_{ij}^T \mathbf{\Sigma}_{ij}^{-1} \mathbf{e}_{ij} \tag{9}$$

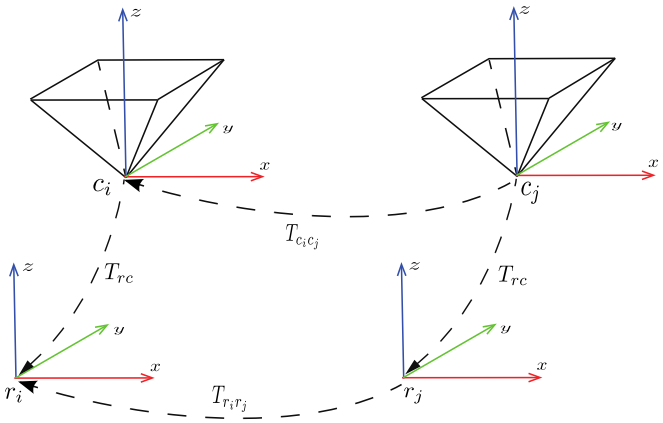

Fig. 2. The rigid body transformation between camera $\{c\}$ and robot frame $\{r\}$, from time-step $\{j\}$ to $\{i\}$.

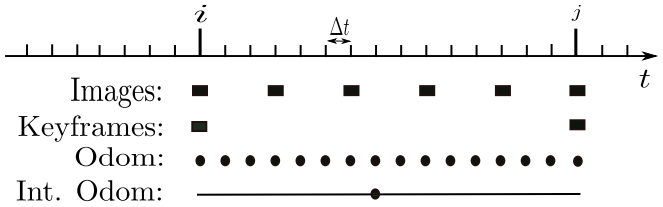

Fig. 3. Timestamp from odometer and camera.

where $\mathbf{x} = (\mathbf{x}_1^T, ..., \mathbf{x}_n^T)^T$ is all the parameters, $\mathcal{C}$ is a set of measuments. In robot application, it is a nonlinear least squares problem. The popular Gauss-Newton or Levenberg-Marquardt algorithm are used to solve the problem. In each iteration, the increment $\Delta\mathbf{x}$ is computed by

$$\sum_{<i,j>\in\mathcal{C}} \mathbf{J}_{ij}^T \mathbf{\Sigma}_{ij}^{-1} \mathbf{J}_{ij} \Delta\mathbf{x}^{(k)} = - \sum_{<i,j>\in\mathcal{C}} \mathbf{J}_{ij}^T \mathbf{\Sigma}_{ij}^{-1} \mathbf{e}_{ij} \quad (10)$$

where $\mathbf{J}_{ij} = \frac{\partial \mathbf{e}_{ij}}{\partial \mathbf{x}}|_{\hat{\mathbf{x}}^{(k)}}$ is the derivation of the error vector with state. The parameters $\hat{\mathbf{x}}$ can be updated with $\hat{\mathbf{x}}^{(k+1)} = \hat{\mathbf{x}}^{(k)} + \Delta\mathbf{x}^{(k)}$. It's important to notice that we need optimize the parameters on a manifold when non-Euclidean parameters are included [14]. During the optimization, the transformation matrix $\mathbf{T}_{ij}$ represented by twist coordinates $\xi_{ij} \in \mathbb{R}^6$ in Lie algebra $se(3)$.

$$\mathbf{T}_{ij}(\xi_{ij}) = \exp(\hat{\xi}_{ij}) \quad (11)$$

$$\xi_{ij} = \log(\mathbf{T}_{ij}) \quad (12)$$

III. GRAPH-BASED EXTRINSIC CALIBRATION AND SENSOR FUSION

The base of our method is to transform the wheel odometry measurement to a virtual measurement for camera and construct factor graph to solve the camera-odometer extrinsic parameters and fuse the sensor information to localization the robot.

A. The Constraint between Camera Motion and Wheel Motion

Camera and wheel odometer work with different frequence (see Fig. 3) and estimate the robot motion independently. At timestamp $t = i$, monocular VO estimate the camera pose $\mathbf{T}_{wc_i}$ and select it as a keyframe. At the same time, Wheel odometer start to integrate the encoder data to estimate the robot motion. When monocular VO select another keyframe at timestamp $t = j$, The transformation $\mathbf{T}_{c_ic_j}$ from the camera frame $\mathbf{T}_{wc_j}$ to $\mathbf{T}_{wc_i}$ can be estimated and the wheel odometer also provide it measurement $T_{r_ir_j}$. As shown in Fig. 2, the camera pose c_j can transform to the robot frame r_i through two path. One way is firstly transform the camera's pose c_j to the robot frame r_j by using the camera-odometer extrinsic transform matrix, then transform it to frame r_i. The another way is firstly transform the camera's pose c_j to c_i, then transform it to r_i. Since the camera pose from monocular visual odometry is up to a scale factor, the constraint is model as [8]:

$$\mathbf{R}_{r_ir_j}\mathbf{R}_{rc} = \mathbf{R}_{rc}\mathbf{R}_{c_ic_j} \quad (13)$$

$$\mathbf{R}_{r_ir_j}\mathbf{t}_{rc} + \mathbf{t}_{r_ir_j} = s\mathbf{R}_{rc}\mathbf{t}_{c_ic_j} + \mathbf{t}_{rc} \quad (14)$$

combine (13) and (14), we can get the compact formula for the camera-odometer constraint.

$$\mathbf{T}_{r_ir_j}\mathbf{S}_{rc} = \mathbf{S}_{rc}\mathbf{T}_{c_ic_j} \quad (15)$$

B. Factor Graph to Extrnsic Calibration

From (15), we can predict the measurment of visual odometry with the wheel odometry.

$$\mathbf{T}'_{c_ic_j} = \mathbf{S}_{rc}^{-1}\mathbf{T}_{r_ir_j}\mathbf{S}_{rc} \quad (16)$$

We define the covariance of virtual measurement $\mathbf{T}'_{c_ic_j}$ with $\mathbf{\Sigma}_{ij}$. When the motion between two keyframes are estimated by monocular VO, the motion error between measurements from the two sensor can be defined by

$$\mathbf{e}_{ij}(\mathbf{S}_{rc}) = \log(\mathbf{T}'_{c_ic_j}\mathbf{T}_{c_ic_j}^{-1}) \quad (17)$$

In the (17), only $\mathbf{S}_{rc}$ need to be estimated. Thus, we can construct a factor graph as shown in Fig. 4. After collect a series of measurement, $\mathbf{S}_{rc}$ can be estimated by minimizing the cost function.

$$\hat{\mathbf{S}}_{rc} = \arg\min_{\mathbf{S}_{rc}} \sum_{<i,j>\in\mathcal{K}} \mathbf{e}_{ij}^T(\mathbf{S}_{rc})\mathbf{\Sigma}_{ij}^{-1}\mathbf{e}_{ij}(\mathbf{S}_{rc}) \quad (18)$$

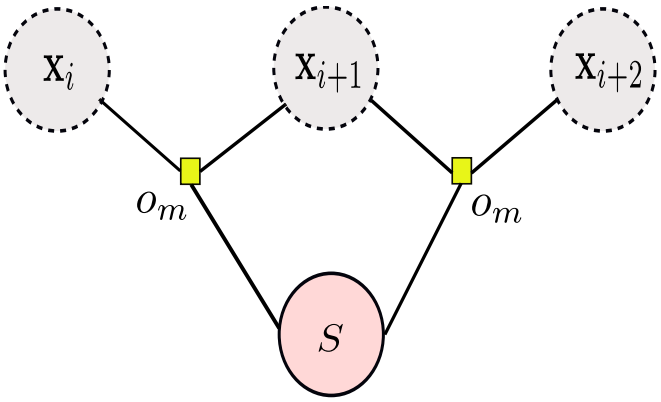

Fig. 4. Factor graph to calibrate extrinsic parameters and scale factor. $\mathbf{S}$ is the node in the factor graph need to be solved. $\mathbf{x}_i$ is camera pose from monocular VO and as a fixed node is the graph. o_m is the virtual camera measurement from wheel odometer.

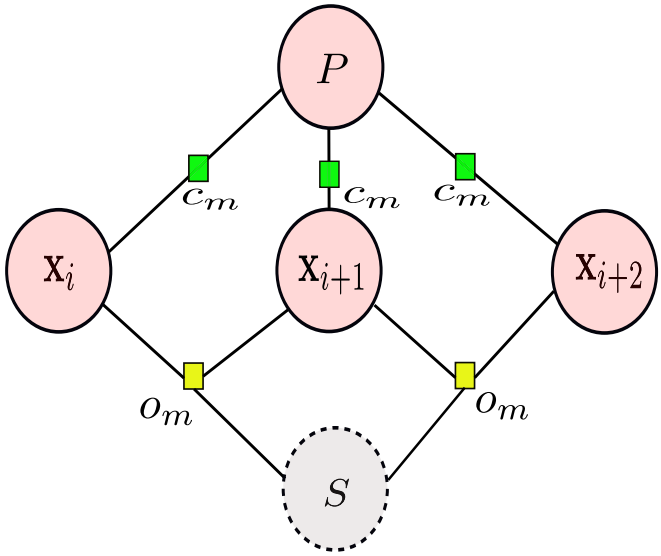

Fig. 5. Factor graph to fusion camera-odometer information to estimate the robot motion. As extrinsic parameters and scale factor calibated before, $\mathbf{S}$ is fixed node in the factor graph. camera pose $\mathbf{x}_i$ and landmark coordinates P is node need to be estimated. o_m is the virtual camera measurement from wheel odometer. c_m is the observation of landmark at each key frame.

Where $\mathcal{K}$ is set of keyframes.

C. Factor Graph to Sensor Fusion

After calibrated the extrinsic parameters, we can fuse the wheel odometry with monocular camera to estimate the robot motion. $\mathbf{T}_{rc}$ is a constant parameter since the camera is fixed on the robot platform. However, the scale factor will be different when the visual odometry initialize in the new environment. Thus, when the visual odometry is initializing, we get the wheel odometry at same time, and use (14) to calculate the scale factor s. Once get the scale factor, we need scale the map point and camera pose estimated by visual odometry to metric.

As the motion estimated by visual odometry are metric, the scale factor in (15) can be removed.

$$\mathbf{T}^{'}_{c_i c_j} = \mathbf{T}^{-1}_{rc}\mathbf{T}_{r_i r_j}\mathbf{T}_{rc} \tag{19}$$

The motion error between measurments from the two sensor can be defined by

$$\mathbf{e}_{ij}(\mathbf{T}_{c_j w}, \mathbf{T}_{c_i w}) = \log(\mathbf{T}^{-1}_{c_j w}\mathbf{T}^{'}_{c_j c_i}\mathbf{T}_{c_i w}) \tag{20}$$

Since 3D landmarks are also estimated by visual odometry, these information need be included in the factor graph. We can construct a re-project error by re-projected the landmark to keyframe. The re-project error is calculated with

$$\mathbf{e}_{il}(\mathbf{T}_{c_i w}, \mathbf{P}_{wl}) = \mathbf{p}_{il} - \pi(T(\mathbf{T}_{c_i w}, \mathbf{P}_{wl})) \tag{21}$$

where $\mathbf{P}_{wl}$ is the l-th landmark in world frame, $\mathbf{p}_{il}$ is the observation of $\mathbf{P}_{wl}$ at the i-th keyframe image plane. To simplify the formula, we use $\mathbf{e}_{ij}$ and $\mathbf{e}_{il}$ represent motion error and re-project error respectively. We stack all the parameters to be estimated in to a vector $\mathbf{X} = (\xi_{1w}, ..., \xi_{nw}, \mathbf{P}_{w_1}, ..., \mathbf{P}_{w_m})$

$$\hat{\mathbf{X}} = \arg\min_{\mathbf{X}} \sum_{<i,j>\in\mathcal{K}} \mathbf{e}^T_{ij}\mathbf{\Sigma}^{-1}_{ij}\mathbf{e}_{ij} + \sum_{i\in\mathcal{C}}\sum_{l\in\mathcal{L}} \mathbf{e}^T_{il}\mathbf{\Sigma}^{-1}_{il}\mathbf{e}_{il} \tag{22}$$

where $\mathbf{\Sigma}_{il}$ is the covariance matrix of the re-project error.

D. Covariance Matrix

The covariance matrix played an important role in (17) and (20) to weight each error term. For $\mathbf{\Sigma}_{il}$, we can set it as a constant since the measurement noise come from the pixel noise. However, the virtual measurement $\mathbf{T}^{'}_{c_i c_j}$ is calculated based on wheel odometry, $\mathbf{\Sigma}_{ij}$ need compute according to wheel odometry noise.

The wheel odometry measurement $\mathbf{T}_{r_i r_j}$ is estimated by integrated the linear and angular velocity from time step i to time step j according to the wheel doometry model (6). According to the covariance propagation theory, we can compute the wheel odometry measurement covariance $\mathbf{\Gamma}_{ij}$ iteratively from the initial conditions $\mathbf{\Gamma}_{ii} = \mathbf{0}_{3\times3}$.

$$\mathbf{\Gamma}_{ij} = \mathbf{A}_{j-1}\mathbf{\Gamma}_{ij-1}\mathbf{A}^T_{j-1} + \mathbf{B}_{j-1}\mathbf{Q}\mathbf{B}^T_{j-1} \tag{23}$$

where

$$\mathbf{A}_{j-1} = \frac{\partial \eta_j}{\partial \eta_{j-1}} = \begin{bmatrix} 1 & 0 & -v\Delta t\sin(\theta_{j-1}) \\ 0 & 1 & v\Delta t\cos(\theta_{j-1}) \\ 0 & 0 & 1 \end{bmatrix} \tag{24}$$

$$\mathbf{B}_{j-1} = \frac{\partial \eta_j}{\partial \eta^n_{j-1}} = \begin{bmatrix} \Delta t\cos(\theta_{j-1}) & \Delta t\sin(\theta_{j-1}) & 0 \\ 0 & 0 & \Delta t \end{bmatrix} \tag{25}$$

Since 3D transformation matrix $\mathbf{T}_{r_i r_j}$ is 6 dof, the odometry covariance matrix $\mathbf{\Gamma}_{ij}$ need to be extended to 6×6 matrix $\mathbf{\Gamma}^{'}_{ij}$ before compute covariance $\mathbf{\Sigma}_{ij}$.

$$\mathbf{\Gamma}^{'}_{ij} = \begin{bmatrix} \delta & 0 & \mathbf{0}_{3\times3} & 0 \\ 0 & \delta & \mathbf{0}_{3\times3} & 0 \\ 0 & 0 & \mathbf{\Gamma}_{ij} & 0 \\ 0 & 0 & \mathbf{0}_{3\times3} & \delta \end{bmatrix}_{6\times6} \tag{26}$$

where δ is a const number $\delta = 10^{-6}$, it means we belive these componet in error vector. Similarity, we can estimated $\mathbf{\Sigma}_{ij}$ by propagating $\mathbf{\Gamma}^{'}_{ij}$

$$\mathbf{\Sigma}_{ij} = \mathbf{J}_{cr}\mathbf{\Gamma}^{'}_{ij}\mathbf{J}^T_{cr} \tag{27}$$

where

$$\begin{aligned} \mathbf{J}_{cr} = \frac{\partial \xi^{'}_{c_{ij}}}{\partial \xi_{r_{ij}}} &= \frac{\partial log(\mathbf{T}^{-1}_{rc}\mathbf{T}_{r_i r_j}\mathbf{T}_{rc})}{\partial \xi_{r_{ij}}} \\ &= \frac{\partial log(\mathbf{T}^{-1}_{rc}\mathbf{T}_{rc}\exp([adj(\mathbf{T}^{-1}_{rc})\xi_{r_{ii}}]_\times))}{\partial \xi_{r_{ij}}} \\ &= \frac{\partial adj(\mathbf{T}^{-1}_{rc})\xi_{r_{ij}}}{\partial \xi_{r_{ij}}} \\ &= adj(\mathbf{T}^{-1}_{rc}) \end{aligned} \tag{28}$$

where $adj()$ is adjoint representation of the algebra. When fusing sensor information to locate the robot, the extrinsic parameters are calculated before, $\mathbf{\Sigma}_{ij}$ can be estimated by

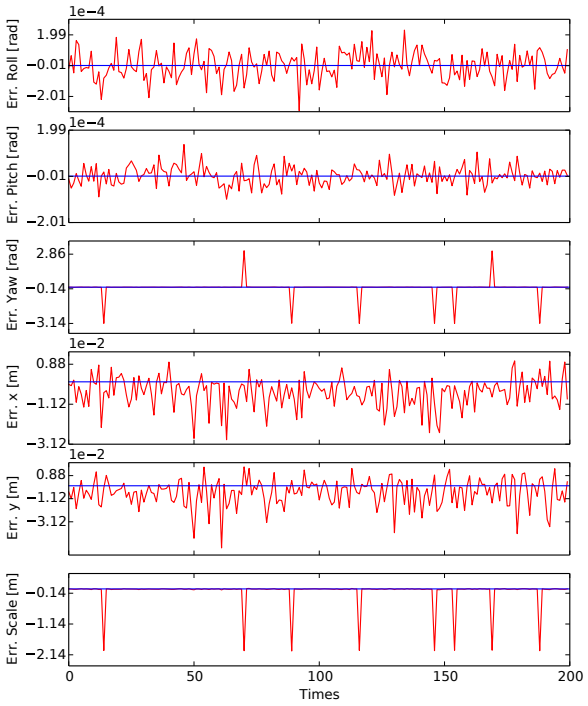

Fig. 6. Extrinsic calibration error.

using (27). However, when calibrating extrinsic parameters according (18), (28) could not estimated, we set $\mathbf{\Sigma}_{ij}$ as a constant matrix.

IV. EXPERIMENTAL EVALUATION

In this section, we use simulation data to test the extrinsic calibration, and we fuse wheel encoder data with SVO. The results show the fusion approach is effectiveness.

A. Simulation Experiments for Extrinsic calibration

In order to validate the factor graph for extrinsic calibration, we have performed 200 times of simulations. In each simulation, The Euler angle of the extrinsic are sampled from a uniform random distribution $\mathcal{U}[-\frac{\pi}{2}, \frac{\pi}{2}]$, the translation of the extrinsic we generated from a uniform random distribution $\mathcal{U}[0, 1]$, the monocular visual odometry scale factor is set as a constant number $s = 1.0$. The robot's trajectory is a square. it can be generated from two type of motion, linear motion and rotary motion. For linear motion, at each time step, it moves forward $0.2m$ with the zero mean Gauss white noise $\mathcal{N}(0, 0.0001)$. For rotary motion, at each time step, it turns left $\frac{\pi}{20} rad$ with the zero mean Gauss white noise $\mathcal{N}(0, 0.0001)$. The camera trajectory is generated by transforming the wheel odometry measurements to the camera frame with the true extrinsic parameters. The initial guess of all parameters are zero when using Gauss-Newton method to solve the optimization problem.

As the robot move on a planar $z = 0$, there are no observation for the translation along z-axis, thus the parameter t_z can not be estimated. However, t_z has no effect on the system when robot move on a planar $z = 0$, we can set t_z as 0. Fig. 6 shows the calibration error of the other parameters $(\theta_x, \theta_y, \theta_z, t_x, t_y, s)$. We can find that roll error, pitch error and translation error are near zero. But at some sometimes, the absolute value of yaw error near π, scal error near -2. This will be happened when the absolute value of yaw is near $\frac{\pi}{2}$, $\hat{\theta}_z \approx -\frac{\pi}{2}, \frac{\pi}{2}$, while the estimated value $\theta_z \approx \frac{\pi}{2}, -\frac{\pi}{2}$. It's can be solved by directly assign a opposite value to the estimate parameters $(\theta_z = -\theta_z, s = -s)$ if we have prior knowledge of yaw.

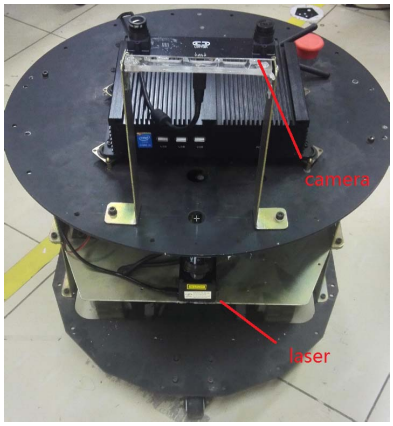

Fig. 7. The mobile robot platform.

B. Real Experiments for Sensor Fusion

We use the factor graph described in Sec. III-C to fuse SVO with wheel encoder data. In the following, we first introduce the details of our implemention, and then present experiments results.

1) Implementation: We use g2o [14] as the factor graph solver. To combine the wheel data, the SVO need calculating the scale factor. When SVO get the first two keyframe and the corresponding wheel odometry measurement, we use (14) to estimate the scale factor s. Then, we zoom the 3D landmark coordinates and the translation between the first two keyframe with the scale factor. After the system initialization is completed, when every new keyframe is set, we construct the fusion factor graph with 7 keyframes near the new keyframe. To save the computation time, we randomly select only 30 landmarks in each key frame to compute the re-project error. When SVO locate failure, we will use virtual measurement for camera from wheel odometry as the localization output and restart the SVO system at same time.

2) Experiments: We use the CASIA-I mobile robot as the platform which equips with Hokuyo URG-04LX scanning laser and a global shutter camera (MT9v034). The laser-odometer extrinsic parameters are calibrated with our method. We control the robot moving around in the room and record the trajectories from SVO and our fusion method. The trajectory estimated by Hector SLAM [15] based on the laser measurements are recorded as the ground truth. All of the trajectories are transformed to the wheel odometry frame. As shown in Fig. 8, the wheel odometry is not accurate since the accumulated error is large. SVO has high accurate localization that its trajectory close to the result of Hector

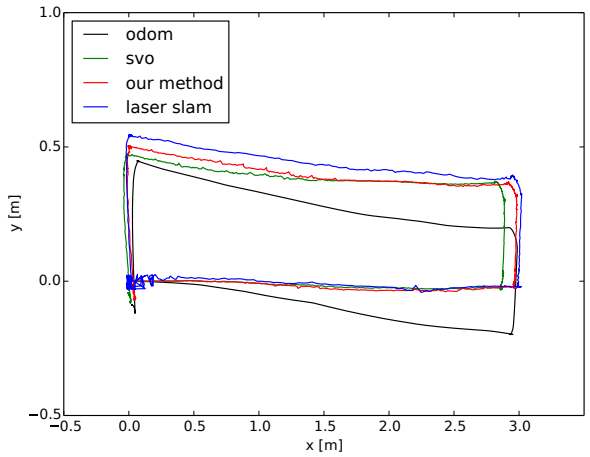

Fig. 8. 2D trajectory.

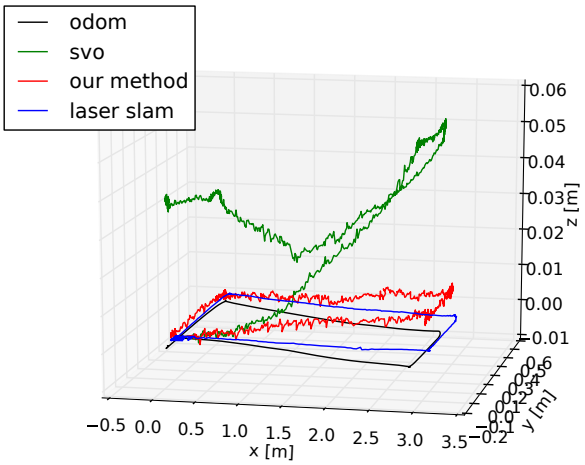

Fig. 9. 3D trajectory.

SLAM. While the scale is drift that its trajectory is short than the ground truth. As shown in Fig. 9, the SVO drift along z-axis that $z_{max} \approx 0.05m$. However, the trajectory estimated by our method have a smaller drift along z-axis. In Fig. 10, we show that fuse SVO with wheel odometry can make the system more robust. The results show that fuse SVO with wheel encoder data with our method can improve the localization accuracy and make the system more robust.

V. CONCLUSION

In this paper, we construct a virtual measurement for the camera by transforming the wheel odometry measurement to camera frame. We use the virtual measurement to construct a factor-graph edge between every two keyframes so that can use graph-based optimization to estimate camera-odometer extrinsic parameters and fuse sensor information. We also derive the covariance matrix of the wheel odometry edges which is important when using graph-based optimization. Simulation and real-world experiments show the fusion approach is effective. However, the camera-odometer extrinsic parameters are still calibrated at offline. In the future work, we want to extend our method to perform online calibration and localization.

Fig. 10. VO locate failure, using wheel oodmetry as location output and restart VO. The red line is the robot trajectory, the blue dot are 3D landmarks of the room ceil in map, the red dot are the active landmarks used to estimate the camera motion.

REFERENCES

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005.

[2] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2d lidar slam," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 1271–1278.

[3] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[4] R. Mur-Artal, J. Montiel, and J. D. Tardós, "Orb-slam: a versatile and accurate monocular slam system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[5] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[6] F. Fraundorfer and D. Scaramuzza, "Visual odometry: Part i: The first 30 years and fundamentals," *IEEE Robotics and Automation Magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[7] G. Antonelli, F. Caccavale, F. Grossi, and A. Marino, "Simultaneous calibration of odometry and camera for a differential drive mobile robot," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 5417–5422.

[8] C. X. Guo, F. M. Mirzaei, and S. I. Roumeliotis, "An analytical least-squares solution to the odometer-camera extrinsic calibration problem," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 3962–3968.

[9] L. Heng, B. Li, and M. Pollefeys, "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*. IEEE, 2013, pp. 1793–1800.

[10] T. Moore and D. Stouch, "A generalized extended kalman filter implementation for the robot operating system," in *Intelligent Autonomous Systems 13*. Springer, 2016, pp. 335–348.

[11] D. Li, K. Eckenhoff, K. Wu, Y. Wang, R. Xiong, and G. Huang, "Gyro-aided camera-odometer online calibration and localization," in *American Control Conference (ACC), 2017*. IEEE, 2017, pp. 3579–3586.

[12] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "isam2: Incremental smoothing and mapping using the bayes tree," *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, 2012.

[13] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual–inertial odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2017.

[14] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g 2 o: A general framework for graph optimization," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 3607–3613.

[15] S. Kohlbrecher, J. Meyer, T. Graber, K. Petersen, U. Klingauf, and O. von Stryk, "Hector open source modules for autonomous mapping and navigation with rescue robots," in *Robot Soccer World Cup*. Springer, 2013, pp. 624–631.