# A NOVEL RE-TRACKING STRATEGY FOR MONOCULAR SLAM

*Hong Liu, Weibo Huang, Zhi Wang*

Key Laboratory of Machine Perception (Ministry of Education)
Shenzhen Graduate School, Peking University
{hongliu, huangweibo, zhiwang53}@pku.edu.cn

## ABSTRACT

Tracking failure is an inevitable event in real-time monocular simultaneous localization and mapping (SLAM) system. Relocalization procedure that focuses on exploring a visited place to relocalize a camera is usually used to handle this problem. However, this strategy abandons the poses of the frames captured after tracking failure, resulting in losing a part of trajectory with respect to the ground truth. Therefore, a re-tracking strategy (RTS) is proposed to estimate these abandoned frames. An automatic local initialization is activated to initialize a new tracking process when tracking fails. Trajectory correction and fusion are employed when a loop closure is detected. When the last frame of the sequence is detected, the trajectories that cannot loop with the original trajectory should be culled by trajectory culling procedure. Experimental results on two challenging datasets, TUM RGB-D and NewCollege, indicate that the proposed method achieves low root mean square error (RMSE) and high trajectory completeness rate (TCR), especially for rapid moving camera.

***Index Terms***— Re-tracking strategy, tracking failure, loop closure, monocular, SLAM

## 1. INTRODUCTION

Monocular SLAM is a visual motion estimation method which tracks each frame to simultaneously estimate the state of a camera and the map of environments using a monocular camera [1–3]. It has been widely applied in real-time applications, e.g., cleaning robots [4], micro aerial vehicles (MAVs) [5, 6] and augment reality (AR) [7, 8].

Various monocular SLAM methods have been developed in recent years. These methods can be broadly classified into direct methods [9, 10] and feature-based methods [11–13]. Direct methods, e.g., dense tracking and mapping (DTAM) [14] and large-scale direct monocular SLAM (LSD-SLAM) [15, 16], use the intensities of all the image pixels to estimate the trajectory of the camera and a 3D map of the scene. These methods are robust to poor texture, defocus, and motion blur. However, direct methods demand for fine initialization and high computing power (e.g., GPUs) [17, 18], which limits their applications in many fields. In the contrary, feature-based methods use sparse feature correspondences to compute the relative motion between two frames. The scene is represented as a set of sparse 3D landmarks. Klein et al. [12] introduce a dual-thread algorithm, known as parallel tracking and mapping (PTAM), which splits tracking and mapping into two separate threads. Motivated by PTAM, ORB-SLAM is proposed by Mur-Artal [13] and achieves unprecedented performance. This method separates the system into three threads: tracking, local mapping, and loop closing. Although significant progress has been achieved, monocular SLAM still remains a challenge, since the tracking process is fragile to fail (i.e., tracking failure) due to rapid moving camera, occlusions, illumination changes and motion blurs. Most of the existing monocular SLAM systems, including ORB-SLAM, utilize relocalization procedure to deal with tracking failure. The frame pose estimation will be suspended until the camera is correctly relocalized. This implies that the system will lose a part of trajectory with respect to the ground truth.

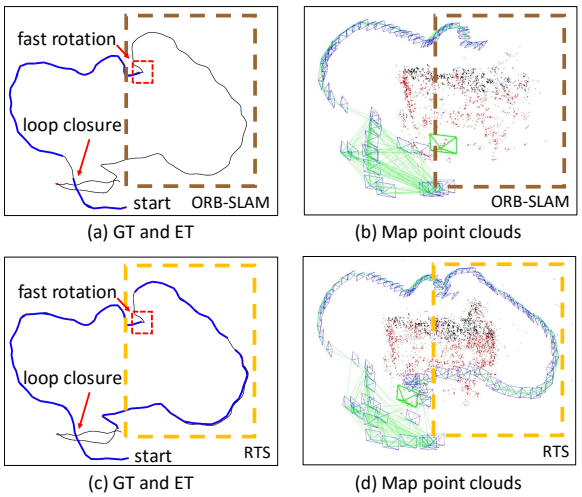



**Fig. 1**. Maps of the *fr3_long_office_household* sequence from TUM RGB-D dataset when down sampling rate is six. Black: ground truth (GT). Blue: estimated trajectory (ET). (a) and (c) Ground truth and estimated trajectory. (b) and (d) Corresponding map point clouds.

This work is supported by National High Level Talent Special Support Program, National Natural Science Foundation of China (NSFC, No.61340046,61673030,U1613209), Specialized Research Fund for the Doctoral Program of Higher Education (No.20130001110011), Natural Science Foundation of Guangdong Province (No.2015A030311034), Scientific Research Project of Guangdong Province (No.2015B010919004).

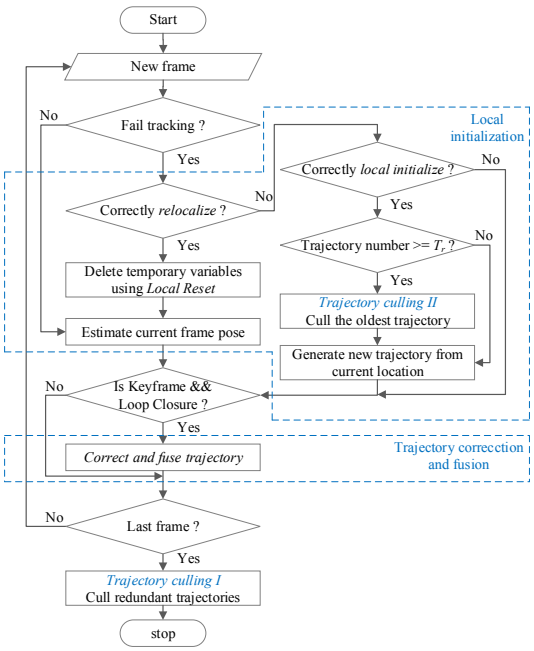



**Fig. 2**. Flowchart of proposed re-tracking strategy

As shown in Fig.1(a) and Fig.1(b), the tracking process is interrupted due to the fast rotation, thus only a part of camera trajectory can be estimated by ORB-SLAM.

To tackle this problem, a more active strategy based on ORB-SLAM, named re-tracking strategy (RTS), is proposed in this paper. Instead of waiting for relocalization to succeed and suspending frame pose estimation, the frames abandoned by forementioned monocular SLAM methods are taken full use of by RTS. The main idea of RTS is to automatically initialize a new tracking process after tracking fails, thus the poses of abandoned frames can be estimated and a new trajectory is generated. Once a loop closure is detected, the new trajectory is fused into loop trajectory. As shown in Fig.1(c) and Fig.1(d), a new trajectory is generated after tracking fails in no time. The flowchart of our re-tracking strategy is shown in Fig.2. There are three procedures in our method: local initialization, trajectory correction and fusion, and trajectory culling. When the system fails to track the pose of a new frame, a tracking failure is detected and then the local initialization is activated to relocalize this frame. If this frame fails to relocalize, our method will try to initialize a new track process and generate a new trajectory from current location. When a loop closure is detected, the trajectory correction and fusion procedure takes charge of correcting the current trajectory and fusing it into the loop trajectory. If the last frame of the sequence is detected, the trajectories that cannot loop with the original trajectory will be culled by trajectory culling.

## 2. RE-TRACKING STRATEGY

### 2.1. Local Initialization

An adaptation of automatic map initialization in ORB-SLAM is used in our algorithm. As shown in Fig.2, the relocalization procedure will be executed to relocalize a new frame when a tracking failure is detected. If the new frame is correctly relocalized, the temporary variables generated by local initialization will be deleted by local reset procedure, and then the pose of the new frame will be estimated. Otherwise, the system will try to generate a new trajectory by performing local initialization procedure. A toy example is shown in Fig.3 to illustrate the local initialization procedure. As shown in Fig.3(a), the system fails to relocalize new frames after detecting a tracking failure at keyframe $i$, which results in breakage denoted by dash line. However, the system correctly initializes a new tracking process soon after tracking failure, and then a new trajectory $l_1$ colored in green is generated. Since the tracking failure is detected twice in Fig.3(b), there are two breakages $b_1$, $b_2$ and three trajectories $l_0$, $l_1$, $l_2$, colored in blue, green, and brown respectively.

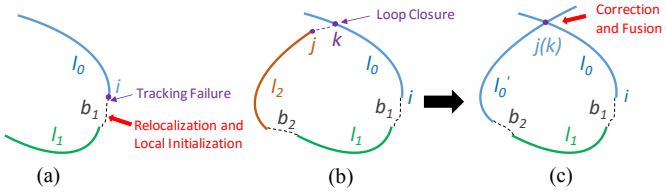



**Fig. 3**. A toy example of re-tracking strategy. $l_0$ is the original trajectory colored in blue; $l_1$ and $l_2$ are the trajectories generated by RTS after twice tracking failures; $i$, $j$, $k$ denote three keyframes; the black dash lines refers to breakages denoted by $b_1$ and $b_2$. (a) a tracking failure is detected at keyframe $i$, and trajectory $l_1$ is generated by RTS. Three trajectories are shown in (b), and a loop closure is detected between keyframe $j$ and $k$. (c) trajectory $l_2$ is corrected and fused into trajectory $l_0$, denoted by "$l_0'$".

### 2.2. Trajectory Correction and Fusion

When the scene captured by a camera is the same as visited place, (i.e., a loop closure is detected), the current trajectory needs to be corrected and fused into the loop trajectory. In Fig.3(b), a loop closure is detected between the current keyframe $j$ and the loop keyframe $k$. The accumulated error of current trajectory $l_2$ is corrected and fused into loop trajectory $l_0$, as Fig.3(c) shows.

In order to correct trajectory, the similarity transformation $\mathbf{S}_{c,l}$ from the current keyframe $c$ to the loop keyframe $l$ is firstly computed as shown in formula (1), indicating the error accumulated in the loop:

$$\mathbf{S}_{c,l} = \begin{bmatrix} s_{c,l}\mathbf{R}_{c,l} & \mathbf{t}_{c,l} \\ 0 & 1 \end{bmatrix} \in Sim(3), \tag{1}$$

where $s_{c,l} \in \mathbb{R}^+$ refers to the scale factor; $\mathbf{R}_{c,l} \in SO(3)$ and $\mathbf{t}_{c,l} \in \mathbb{R}^3$ are the rotation and translation parts of keyframe pose $\mathbf{T}_{c,w} \in SE(3)$ respectively, where $SO(3)$ and $SE(3)$ are the special orthogonal group and the special Euclidean group; $Sim(3)$ is the similarity transformation group. Then, the current keyframe pose $\mathbf{T}_{c,w}$ is corrected with $\mathbf{S}_{c,l}$:

$$\mathbf{T}_{c,w}^{cor} = \mathbf{S}_{c,l} \cdot \mathbf{T}_{l,w}, \tag{2}$$

where $w$ stands for the world reference, and $\mathbf{T}_{c,w}^{cor}$ is the corrected pose. This correction is propagated in two different

conditions. If the current keyframe $c$ and the loop keyframe $l$ are located in the same trajectory, the correction will be propagated to all the neighbor keyframes of current keyframe. The correction method of this condition can be seen from [19]. Otherwise, as shown in Fig.3(c), the correction will propagate from back to front. Therefore, all the keyframes belonging to the current trajectory will be corrected. Formulation for this condition are shown as follows.

The relative transformation $\Delta\mathbf{T}_{c-\sigma+1,c-\sigma}$ from each keyframe pose $\mathbf{T}_{c-\sigma+1,w}$ to its earlier adjacent keyframe pose $\mathbf{T}_{c-\sigma,w}$ is computed as follows:

$$\Delta\mathbf{T}_{c-\sigma+1,c-\sigma} = \mathbf{T}_{c-\sigma+1,w} \cdot \mathbf{T}^{-1}_{c-\sigma,w}, \tag{3}$$

where $\sigma = (1, 2, ..., n-1)$; $n$ refers to the keyframe number of the current trajectory; $\mathbf{T}^{-1}_{c-\sigma,w}$ is the inverse of $\mathbf{T}_{c-\sigma,w}$.

Note that the current keyframe has been corrected in formula(2), the rest of the keyframes belonging to the current trajectory need to be corrected:

$$\mathbf{T}^{cor}_{c-\sigma,w} = \Delta\mathbf{T}^{-1}_{c-\sigma+1,c-\sigma} \cdot \mathbf{T}^{cor}_{c-\sigma+1,w}. \tag{4}$$

Finally, the current trajectory label should be modified to the loop trajectory label. As shown in Fig.3(c), trajectory $l_2$ is fused into trajectory $l_0$ and colored in blue. The label "2" is modified to label "0", denoted by "$l_0^{'}$".

### 2.3. Trajectory Culling

The trajectory culling procedure works in two case. Case I: As can be noticed, several trajectories may be generated due to the frequent tracking failures. However, the monocular SLAM systems in generally need to output a single camera trajectory. Therefore, the trajectories that cannot loop with original trajectory are considered as redundant. When the last frame of the sequence is detected, the redundant trajectories should be culled. As shown in Fig.3(c), the accumulated error of the trajectory $l_1$ cannot be corrected, since it cannot loop with original trajectory $l_0$, and should be culled when the last frame is detected. Case II: In order to avoid unnecessary memory overload, a threshold $T_r$ is set to limit the quantity of trajectory. If the number of trajectory is greater than $T_r$, the trajectory with the oldest label will be culled. With this strategy, our method only needs to maintain a certain amount of trajectories, and will output a longer trajectory than the general monocular SLAM systems.

## 3. TRAJECTORY COMPLETENESS RATE

Absolute trajectory root mean square error (RMSE) is the most widely used evaluation metric in monocular SLAM, which evaluates the accuracy of the estimated trajectory. However, accuracy and completeness of the trajectory are both significant metrics in some applications, such as 3D reconstruction shown in Fig.1. The accuracy of Fig.1(a) is similar to that of Fig.1(c), however, the estimated trajectory length of ORB-SLAM is much shorter than that of RTS. Hence, a new evaluation metric named trajectory completeness rate (TCR) is defined to evaluate trajectory completeness for monocular SLAM, as follows:

$$\eta = \frac{L_{est}}{L_{gt}} \times 100\%, \tag{5}$$

$$\begin{aligned} L &= \sum_{i=1}^{N-1} ||p_{i+1} - p_i|| \cdot f_i, \\ where\ f_i &= \begin{cases} 0, & if\ \ |t_i - t_{i-1}| > T_{max}, \\ 1, & if\ \ |t_i - t_{i-1}| \leq T_{max}. \end{cases} \end{aligned} \tag{6}$$

In formula(5), $\eta$ is the TCR; $L_{est}$ and $L_{gt}$ are the lengths of estimated trajectory and ground truth respectively. In formula(6), $L$ refers to the trajectory length; $N$ is the number of keyframe; $p_i$ and $t_i$ are the camera location and the timestamp of the $i$-th keyframe respectively; $||\cdot||$ is the Euclidean distance; $|\cdot|$ is the absolute value. The adjacent pairs of which timestamp difference exceeds the threshold $T_{max}$ are discarded by multiplying an indicator $f_i$, since they are considered as mismatched pairs due to the breakages in the estimated trajectory.

## 4. EXPERIMENTS AND DISSCUSSIONS

A large number of experiments are conducted on two datasets, NewCollege [20] and TUM RGB-D [21]. To test the validity of the proposed method for fast moving camera, the TUM RGB-D sequences are down sampled with a rate of 1, 2, 4, 6 to proportionally speed up camera movement. These processed sequences are challenging since the rotation and translation are faster than the raw sequences, as well as lower field of view overlap. In our experience, the $T_r$ and $T_{max}$ are set as 10 and 2 seconds. All experiments are carried out with an Intel CPU i7-4720HQ (8 cores @2.60GHz) and 8GB RAM.

**TUM RGB-D** is an excellent dataset to evaluate the accuracy of camera localization as it provides several sequences with accurate ground truth, containing 89 sequences. Following the protocol of [13], we discard those sequences that contain strong illumination changes, no texture, or no camera motion, which are considered not suitable for pure monocular SLAM systems. Ten sequences are selected, and the average translational velocities of these sequences range from $0.06\ m/s$ to $0.41\ m/s$. For comparison, we use the novel keyframe-based ORB-SLAM in the dataset. Table I shows average performances over five runs on each of ten selected sequences. The emphasized data shows the outperformance.

Although the down sampling rate increased, the results show that both RTS and ORB-SLAM can process all the sequences. The absolute trajectory RMSE remains relatively stable, except for sequence $fr2\_360\_kidnap$. This is an industrial hall scene with few texture, and the camera rotates horizontally around 360 degrees. The camera is covered with a hand for a few seconds and then pointed to a different location. The view overlap decreases considerably when this

**Table I**. Comparisons on the TUM RGB-D dataset [21] with evaluation methodology of TCR and RMSE

| | | ORB-SLAM [13] | | | | | | | | RTS (ours) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | TCR(%)[1] | | | | RMSE(cm)[2] | | | | TCR(%)[1] | | | | RMSE(cm)[2] | | | |
| **sequence** | **Avg. T. Vel.**[1] | **1** | **2** | **4** | **6** | **1** | **2** | **4** | **6** | **1** | **2** | **4** | **6** | **1** | **2** | **4** | **6** |
| fr2_xyz | 0.06 | 20.71 | 20.31 | 31.52 | 31.18 | 0.27 | 0.27 | 0.30 | 0.27 | 25.65 | 28.62 | 28.43 | 28.70 | 0.30 | 0.28 | 0.25 | 0.28 |
| fr2_desk_person | 0.12 | 72.04 | 76.91 | 74.97 | 72.39 | 0.73 | 0.90 | 0.93 | 0.92 | 82.42 | 80.29 | 82.23 | 78.41 | 0.69 | 0.88 | 0.98 | 0.78 |
| fr3_str_tex_near | 0.14 | 99.96 | 99.98 | 95.59 | 97.04 | 1.33 | 1.19 | 1.55 | 1.74 | 99.59 | 99.75 | 99.97 | 96.08 | 1.26 | 1.33 | 1.59 | 1.81 |
| fr3_sit_halfsph | 0.18 | 85.67 | 86.02 | 86.55 | 63.93 | 1.19 | 1.41 | 1.80 | 1.39 | 85.79 | 86.34 | 86.74 | 63.09 | 1.62 | 1.51 | 1.72 | 1.68 |
| fr3_str_tex_far | 0.19 | 88.99 | 92.77 | 91.28 | 87.41 | 1.07 | 1.07 | 1.07 | 0.96 | 86.81 | 92.32 | 91.51 | 91.01 | 1.11 | 0.97 | 0.99 | 0.93 |
| fr3_walk_halfsph | 0.22 | 68.45 | 59.17 | 49.88 | 26.09 | 1.93 | 1.75 | 1.57 | 1.59 | 76.50 | 56.43 | 50.00 | 26.54 | 1.66 | 1.53 | 1.71 | 1.85 |
| fr1_xyz | 0.24 | **67.37** | **61.75** | **46.83** | **34.43** | 0.94 | 1.01 | 1.81 | 1.36 | **70.89** | **63.88** | **52.11** | **41.05** | 1.10 | 1.03 | 1.42 | 1.06 |
| fr3_long_office | 0.25 | 99.92 | 99.89 | **88.38** | **47.00** | 1.34 | 1.82 | 1.56 | 1.23 | 99.91 | 99.87 | **99.94** | **89.73** | 1.46 | 1.52 | 1.66 | 1.48 |
| fr2_360_kidnap | 0.30 | **45.02** | **41.06** | **37.19** | **18.68** | 3.33 | 4.02 | 5.58 | 6.75 | **69.79** | **69.63** | **78.45** | **36.32** | 8.03 | 8.36 | 9.47 | 12.91 |
| fr1_desk | 0.41 | 97.46 | **67.09** | **33.35** | **0.82** | 1.54 | 1.99 | 2.17 | 0.86 | 97.84 | **77.52** | **55.47** | **0.92** | 1.85 | 1.81 | 1.88 | 0.81 |

[1] The length of ground truth and average translation velocity (Avg.T.Vel) can be taken from website: http://vision.in.tum.de/data/datasets/rgbd-dataset.
[2] The estimated trajectory is aligned with ground truth to get minimal RMSE by adjusting scale factor.

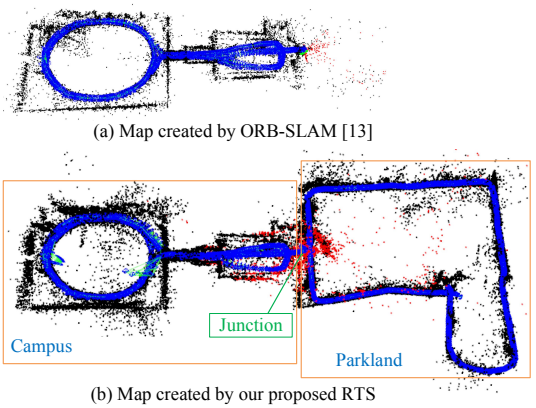
sequence is down sampled. Therefore, it's hard to accurately estimate the camera trajectory.

As for trajectory completeness, it can be observed that the TCR decreases significantly as the down sampling rate increases when performing ORB-SLAM, for the sequences with average translation velocity over $0.24m/s$. While our method maintains high TCR when tracks these sequences. In particularly, when down sampling rate is six, for $fr3\_long\_office\_household$ (fr3_long_office), the completeness of trajectory estimated by our method is up to $89.73\%$ of the ground truth, while ORB-SLAM achieves $47.00\%$. As shown in Fig.1, the large camera rotation leads to tracking failure and breaks the estimated trajectory. Fig.1(c) shows that RTS generates a new trajectory soon after breakage, while ORB-SLAM can recover tracking only after the camera is correctly relocalized.

It is noted that, the TCR of $fr1\_desk$ is rather small when the down sampling rate is six, for both ORB-SLAM and RTS. This is a cramped office scene containing several sweeps over four desks. The view changes fast and it takes a long time to initialize and then fails tracking quickly. It's quite hard to successfully perform local initialization. In the rest of the sequences, our method exhibits similar performance to ORB-SLAM, due to the low translation velocity and bare tracking failure.

**NewCollege** is a 2.2 kilometres, outdoor, dynamic long term sequence containing several loops and fast rotations, which is quite challenging for monocular SLAM. The images are on average taken by a stereo camera equipped on a robot at 20 $fps$, 1.5 $m/s$ average speed, and a resolution of $512 \times 382$. ORB-SLAM is the first monocular system which successfully processes this whole sequence at frame rate. However, when the sequence is down sampled with a rate of three, ORB-SLAM fails to estimate the trajectory of the parkland (the bigger loop on the right), shown as Fig.4(a). The tracking failure is caused by the fast rotation at the junction of parkland and campus. ORB-SLAM cannot recover tracking in the parkland until the robot returns to the campus.

Fig. 4(b) indicates that the trajectory of the parkland can be estimated completely when the robot moves clockwise by RTS. A loop closure can be detected when the robot moves back to the campus after moving around the parkland. The parkland trajectory is corrected and fused into campus trajectory. However, the final big anticlockwise loop of the parkland is unable to be corrected due to no visual loop closure being found. This anticlockwise loop is culled when the last frame is detected.




(a) Map created by ORB-SLAM [13]

(b) Map created by our proposed RTS

**Fig. 4**. Maps of *NewCollege* when down sampling rate is three. The trajectory is drawn in blue; the local map for the tracking at the junction is colored in red. The bigger loop on the right is the "Parkland", the rest is the "Campus".

## 5. CONCLUSIONS

In this work, a new method called re-tracking strategy (RTS) is proposed to deal with tracking failure problem in real-time monocular SLAM. Different from most existing methods utilizing relocalization strategy to recover tracking, RTS devotes to automatically initialize a new tracking process after tracking fails. Additionally, a new evaluation metric named trajectory completeness rate (TCR) is introduced to reflect the completeness of the estimated trajectory. Experimental results on two datasets prove that RTS can achieve longer trajectory than traditional monocular SLAM methods, while maintaining high accuracy, since the frames abandoned by traditional methods are taken full use of.

## 6. REFERENCES


[1] M. Soto-Alvarez and P. Honkamaa, "Multiple hypotheses data association propagation for robust monocular-based slam algorithms," *in IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6543–6547, 2014.

[2] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part I," *IEEE Robotics and Automation Magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[3] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (SLAM): Part II," *IEEE Robotics and Automation Magazine*, vol. 13, no. 3, pp. 108–117, 2006.

[4] L. Buonocore, R. Sérgioand A. A. Neto dos Santos, and C. L. Nascimento, "Fastslam filter implementation for indoor autonomous robot," *in IEEE Intelligent Vehicles Symposium*, pp. 484–489, 2016.

[5] S. Weiss, M. W. Achtelik, S. Lynen, M. Chli, and R. Siegwart, "Real-time onboard visual-inertial state estimation and self-calibration of mavs in unknown environments," *in IEEE International Conference on Robotics and Automation (ICRA)*, pp. 957–964, 2012.

[6] L. Heng, G. H. Lee, and M. Pollefeys, "Self-calibration and visual slam with a multi-camera system on a micro aerial vehicle," *Autonomous Robots*, vol. 39, no. 3, pp. 259–277, 2015.

[7] L. Porzi, E. Ricci, T. A. Ciarfuglia, and M. Zanin, "Visual-inertial tracking on android for augmented reality applications," *in IEEE Workshop on Environmental Energy and Structural Monitoring Systems (EESMS)*, pp. 35–41, 2012.

[8] J. Polvi, T. Taketomi, G. Yamamoto, A. Dey, C. Sandor, and H. Kato, "Slidar: A 3d positioning method for SLAM-based handheld augmented reality," *Computers & Graphics*, vol. 55, pp. 33–43, 2016.

[9] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. D. Reid, and J. J. Leonard, "Simultaneous localization and mapping: Present, future, and the robust-perception age," *http://arxiv.org/abs/1606.05830*, 2016.

[10] N. Krombach, D. Droeschel, and S. Behnke, "Combining feature-based and direct methods for semi-dense real-time stereo visual odometry," *in International Conference on Intelligent Autonomous Systems (IAS)*, in press, 2016.

[11] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.

[12] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," *in IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 225–234, 2007.

[13] R. Mur-Artal, J.M.M 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.

[14] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "DTAM: Dense tracking and mapping in real-time," *in IEEE International Conference on Computer Vision (ICCV)*, pp. 2320–2327, 2011.

[15] J. Engel, J. Sturm, and D. Cremers, "Semi-dense visual odometry for a monocular camera," *in IEEE International Conference on Computer Vision (ICCV)*, pp. 1449–1456, 2013.

[16] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," *in European Conference on Computer Vision (ECCV)*, pp. 834–849, 2014.

[17] R. A. Newcombe and A. J. Davison, "Live dense reconstruction with a single moving camera," *in IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1498–1505, 2010.

[18] M. Pizzoli, C. Forster, and D. Scaramuzza, "Remode: Probabilistic, monocular dense reconstruction in real time," *in IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2609–2616, 2014.

[19] R. Mur-Artal and J.D. Tardós, "Fast relocalisation and loop closing in keyframe-based slam," *in IEEE International Conference on Robotics and Automation (ICRA)*, pp. 846–853, 2014.

[20] M. Smith, I. Baldwin, W. Churchill, R. Paul, and P. Newman, "The new college vision and laser data set," *The International Journal of Robotics Research*, vol. 28, no. 5, pp. 595–599, 2009.

[21] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," *in IEEE International Conference on Intelligent Robot Systems (IROS)*, pp. 573–580, 2012.