Application of Hybrid Monocular SLAM Method in Augmented Reality

Zuoming Zhang¹, Hanwen Wang², Man Shu³ and Xin Wang¹*

¹,²,³ College of Software Engineering, Jilin University, Changchun, Jilin, 130000, China
¹*Key Laboratory of Symbolic Computation and Knowledge Engineer of Ministry of Education, Jilin University, Changchun, Jilin, 130000, China
⁷Corresponding author’s e-mail: w_x@jlu.edu.cn

Abstract. In this paper, we design a hybrid (semi-direct) approach to simultaneous localization and mapping (SLAM) for monocular cameras and apply it to augmented reality (AR) for monocular cameras. We combine the advantages of the direct method and the feature point method. We use both photometric bundle adjustment which is robust to camera exposure time and motion bundle adjustment which is geometrically robust based on feature points to do tracking process. This approach can maintain an intuitive direct local map as well as a reusable global sparse feature point map. Through the processing of point clouds, such as PCA plane detection and grid reconstruction, we greatly improve the effect of the augmented reality system.

1. Introduction

Real-time visual odometry (VO) and simultaneous localization and mapping (SLAM) are technologies for real-time sensing of surroundings and can be applied to a wide range of applications including unmanned aerial vehicles (uavs), robotic navigation, and virtual reality (VR) or augmented reality (AR). In augmented reality, virtual objects need to be accurately positioned in the real world. In addition, virtual objects should also have interactive functions with the real environment, such as shielding or collision. The monocular camera-based VO can calculate the camera pose in real-time and maintain a small local point clouds map in real-time. However, compared with the SLAM scheme, VO cannot realize loop-closing detection, so as to maintain a large and reusable global map.

In order to achieve the requirements of real-time and robustness, most Monocular SLAM schemes adopt the feature point-based indirect method due to the stable operation of feature points and insensitivity to illumination and dynamic objects [4]. However, the direct method [2] based on the assumption of constant gray level can obtain semi-dense or even dense maps with high recognition, but it is very sensitive to light. In order to better apply SLAM to augmented reality, as shown in Figure 1, our work has the following contributions:

- We design a tracking process that combines photometric bundle adjustment and geometric bundle adjustment to achieve higher illumination and geometric robustness.
- We maintain a short-range semi-dense map and a global reusable sparse feature point map to meet the requirements of virtual and real interaction and positioning respectively.
- We propose a plane detection scheme with higher efficiency and stability to improve the interactive experience.
The results show that the augmented reality system can well deal with the rapid movement of various scenes and cameras.

Figure 1. The two images above are the mapping based on feature points of ORB-SLAM [1], which is sparse but not intuitive and difficult to deal with the lack of texture. The lower left is the semidense map of LSD-SLAM [2], which cannot cope with the rapid camera movement. The lower right [3] is the SLAM of RGBD camera, which is complete but requires high equipment.

2. Related Work

2.1. Indirect method

The indirect method is also called feature-based method. This method guarantees geometric robustness by extracting feature points. Two adjacent frames are matched by feature points, such as FLANN [5] or RANSAC [6], [7], to obtain camera motion by solving PnP or ICP. Early monocular SLAM systems had PTAM [8] based on filter optimization, but later ORB-SLAM [1] was more widely used, and the proposed geometric(Motion) bundle adjustment adopted a nonlinear optimization method to optimize the camera pose. The approach used to extract ORB feature points is robust and computation time superior. But extracting feature points is still a very expensive method, although stable, but the continuous calculation of feature points is not suitable for long-term applications in augmented reality systems. In addition, the maps obtained by this method are sparse, and more point clouds information is usually obtained using RGBD cameras. But this method is still time-consuming and requires a lot of equipment.

2.2. Direct method

Direct method does not take extra time to calculate the feature points, but extracts the high gradient pixel points by setting parameters and tracks them directly. The advantage of this is that we don’t need very expensive equipment to extract semi-dense, dense maps. Since the most primitive direct method is based on the assumption of invariable gray level, and modern smartphone devices tend to adjust the exposure parameters automatically, it is extremely sensitive to light in practical applications. LSD-SLAM [2] is the first complete SLAM system based on the direct method, and the concept of photometric bundle adjustment is proposed. After adding loop closing detection, an intuitive semi-dense map can be built. In [9], the map obtained by LSD SLAM is used for rough mesh fitting to
realize the effect of a collision between virtual objects and real objects. In [10], a direct method of searching points and lines based on objects is proposed. After that, another VO system called DSO [11] appeared. Compared with LSD-SLAM, this system deleted some set prior information to speed up marginalize, and further improved the BA by combining photometric camera calibration with the SSD method, and its illumination robustness was significantly improved. However, it is a VO system after all. We need a SLAM like a method in [12]. With the advance of time, the drift and error of the system will increase continuously, making it difficult to maintain large-scale reusable global maps [13], [14].

2.3. Hybrid method
It’s also known as a semi-direct method, this system can combine the advantages of direct and indirect, such as SVO [15], only need to extract sparse feature points, and use image alignment between each frame (similar to direct). The calculation speed of this method is very fast, and it can maintain high accuracy in fast motion after IMU [16] is added. However, this method is only a VO system, which also has no functions loop-closing detection. In [17], there is a loosely coupled SLAM system. Unlike SVO, the tracking process allocates the images between frames, extracts the feature points, and then builds both semi-dense map and sparse map. Although this method is highly efficient, it is less robust than ORB-SLAM and DSO, but consider the limitations of the actual device, it has greater potential in practical applications than purely direct or indirect in AR.

3. Our method
See Figure 3. Through the main method proposed in DSO, namely photometric bundle adjustment, combined with photometric camera calibration of the camera, and via the sliding window method, we formed a direct module to take charge of most of the tracking work. Through keyframes and inverse depth [18], [19] that are already in the sliding window, the information for each frame is directly estimated. The method used by DSO can circumvent the geometric prior information, so keyframes obtained can be quickly marginalized. These keyframes are then passed to the indirect module. At the same time, the position and scale of these keyframes are passed into the indirect module in the form of Lie algebra. Extracting feature points only for this part of the key frame can greatly reduce the time of computing feature points. Now that we’ve got the pose and scale. So we further restore the depth of the feature points from this information and then use the motion only bundle adjustment to solve, similar to ICP. Next, implement the complete system through loop-closing a full BA in [3]. In the above process, we can get two kinds of maps, the semi-dense local map obtained by the direct module and the reusable sparse feature point map obtained by the indirect module, as shown in Figure 2. For the global map, we use a PCA method, which is more effective than others, to carry out plane detection on the map to realize the positioning function of augmented reality. Combine with ROS or Bullet, or even the Unity engine, we achieve the augmented reality system.

Figure 2. System Overview
3.1. Photometric BA

The photometric model proposed in DSO combined with camera calibration can avoid errors caused by changes in environmental light source, automatic adjustment of equipment exposure time and parameters. In combination with the TUM monoVO data set, we can define the following model [20]:

\[
I_i(x) = t_i B_i(x) = \frac{G^{-1}(I'_i(x))}{V(x)}
\]  

where \( B_i \) and \( I'_i \) are irradiance and the observed pixel intensity in frame, \( t_i \) is the exposure time, \( V \) is the fading of the lens and the range is \([0,1]\), \( G \) is the photometric response function, and \( G^{-1} \) is used to restore the pixel value at that point to the energy value. The function of this formula is to convert the gray value into the product of the energy of the observation point and the exposure time.

The direct module maintains a sliding window, which persists throughout the tracking process. When a new frame arrives, the direct module combines the map points that already exist in the keyframe in the window and projects them into the new frame. We use the [11] method to extract keyframes. For the detected new keyframes, we further extract the immature points and turn them into active points. In order to adapt to the light and shade changes caused by camera exposure, the photometric correction model should be used to optimize the BA process

\[
E_{p_i} := \sum_{p \in \mathcal{N}_i} \omega_p \left\| (I_j[p'] - b_j) - \frac{t_e}{t_e} (I_j[p] - b_i) \right\|^2
\]

With

\[
\omega_p = \frac{c^2}{c^2 + \left\| \nabla I_i(p) \right\|^2}
\]

\[
p' = \| I_i(R[p] - (p, d_p) + t)
\]

Where, we use \( \| \) to describe the mapping of spatial coordinates to pixel coordinates, which is obviously related to the internal parameters of the camera, \( a \) and \( b \) are the frame’s brighter transfer function parameters. We use inverse depth \( d_p \) to describe the spatial position of map points. For pixel \( p \), we optimize not just one pixel and its projection, but eight points (SSD) in \( \mathcal{N}_p \). For the representation of weight \( \omega_p \), it is obvious that the larger the pixel gradient, the less influence this point has on the result. To keep the results from growing too fast, we use the Huber kernel. For an active point in \( i \) in a certain keyframe, the corresponding projection may not be found in all other keyframes, so we can only optimize the solution for frame \( j \) of all visible pixel projection points in the window.

For all the frames in the window, we can merge all the frames to minimize photometric errors

\[
E_{\text{photo}} := \sum_{i \in \mathcal{F}} \sum_{p \in \mathcal{P}} \sum_{j \in \text{obs}(p)} E_{p_j}
\]

If we know the exposure time of the camera, we need to minimize a prior error first

\[
E_{\text{prior}} := \sum_{i \in \mathcal{F}} (\lambda_a a_i^2 + \lambda_b b_i^2)
\]
Figure 3. Blue points are high gradient pixel points extracted based on direct method, which are used to maintain semi-dense local maps; red is extracted feature points, which are used to maintain global sparse maps. I is just introduced into indirect marginalized KF module. IV and V is respectively is receives the corresponding KF (II and III) map.

However, for most augmented reality devices, we don't know the exposure time exactly, so we can set the exposure time to 1 in (2) and the parameter $\lambda = 0$ in (6) at the very beginning. In the process of solving BA, Gauss-Newton or other nonlinear optimization algorithms[21] can be used to continuously optimize iteratively. In the process of using Lie algebra perturbation, we will continuously optimize all the active variables in the window, including camera pose, inverse depth, photometric camera calibration. In this process, Schur complement can be used to accelerate the rate of marginalization, thus keeping the size of the sliding window stable.

3.2. Geometric BA

When a keyframe in the direct module is marginalized, the indirect module can be passed in. The purpose of the indirect module is to improve robustness, extract ORB feature points and optimize posture. The incoming keyframe carries the initial camera position of the frame, the relative scale and the inverse depth of the active points. The former can be directly used by motion BA, but the depth of the feature points extracted again needs to be further solved. In polar search, in order to increase accuracy, we extract small blocks of a certain size around feature points and then use Normalized Cross Correlation (NCC)[22] to express the Correlation of two small blocks. Then, the reasonable depth information is searched and expressed as the inverse depth.

$$S(A, B)_{NCC} = \frac{\sum_{i,j} A(i, j)B(i, j)}{\sqrt{\sum_{i,j} A(i, j)^2 B(i, j)^2}}$$

(7)

Here, we use a depth filter like the Kalman method to preliminarily restore the 3D depth information after the feature points are extracted from 2D images are searched and matched with blocks by polar lines. For one feature point $p$, its inverse depth is computed

$$d_p = \frac{\sum_{i:e_p} d_i/\sigma_k^2}{\sum_{i:e_p} 1/\sigma_k^2}$$

(8)

$p_p$ denote the set of all map points whose projection in the current frame is equal to $p$.

Relative scale ensures that indirect modules can be initialized when the keyframes are marginalized. Once the feature point of the keyframe and its inverse depth are calculated, it is equivalent to the preliminary calculation of the VO part completed by the indirect module. But this part is just
preliminary linear operation, so we need to further consider bundle adjustment to optimize the configuration. Here, we can obtain the 3D information of the feature points to solve the ICP. Here, we can also use Gauss-Newton or other nonlinear optimization algorithms to solve the ICP iteratively. The solution objective is

\[ E_{\text{motion}} := \sum_{p \in F} \left\| p' - \Pi_j(\exp(\zeta^e)\Pi_j^{-1}(p, d^e_s)) \right\| \]

(9)

Where, \( j \) represents the previous frame, \( p' \) represents the projection of feature points in the current frame, and \( \zeta \) represents the Lie algebra of the position of the keyframe. Here \( \exp(\zeta^e) \in \text{Sim}(3) \). In the calculation process, since different feature points are located in different levels of the image pyramid, the weight of the level should be considered in the calculation.

3.3. Global Mapping
In 3.1 and 3.2, we completed tracking and optimization, but we need to further add loop-closing to eliminate accumulated errors. Here, DBoW2\([23,24]\) is used in combination with the method proposed in ORB-SLAM2 to create a separate thread and implement loop-closing through word bag. In addition, we performed global BA\([3]\) at last to ensure the scale and reusability of mapping. In fact, we still use the feature point map to achieve the positioning effect of augmented reality, to cope with the possible rapid movement of the camera and eliminate the drift error.

3.4. Plane Detection
Principal component analysis (PCA) is a statistical process that uses orthogonal transformations to convert observations of a set of potentially correlated variables into values of linearly independent variables of the Principal components. The number of different principal components is equal to the original number or the observed number minus one. After obtaining the map, we carried out planar detection on the point cloud map based on the idea of PCA. Firstly, several points (more than 3) are randomly selected from the point cloud, and PCA is used to calculate the initial plane parameters and the normal vector, which are taken as the initial plane. Further, calculate the orthogonal distance of each point and order it from large to small. The first \( k \) points are extracted from this sequence and then PCA is used for progressive plane fitting. Then, the threshold value (more than 2) was set and the z-score method\([25]\) was used to eliminate the abnormal points before we detect the plane.

\[ Z_i = \frac{|x_i - \text{median}(x_j)|}{k \cdot \text{median}|x_i - \text{median}(x_j)|} \]

(10)

Here \( k \) is a constant, in a normal distribution \( k = 1.4826 \). When the score is greater than 2.5, we eliminate that point. Finally, the normal vector was calculated by the plane quasi-merger again to get the final plane result.

Through our comparative experiments, for maps with sparse feature points, the plane fitting method based on PCA is more efficient than the pure solution of the homography matrix. See Table 2. We designed the following experiment to compare the plane detection effect under the sparse point cloud map. According to the gaussian distribution, point clouds with the size of 1500-10000 were generated randomly, and the rate of abnormal points was assumed to be 10% and 20%, and the average result was calculated by running the program 1000 times. We used CIR(Correct Identification Rate) and SR(Swamping Rate)\([25]\). By comparing with the RANSAC algorithm and setting the RANSAC to iterate 50 times, we obtained the results in Table 1.
It can be seen that the calculation time of the algorithm in this paper is significantly higher than that of the RANSAC algorithm, and the SR is lower than that of the RANSAC due to the use of z-score to eliminate abnormal points. But the overall performance is better than RANSAC.

Table 1. Comparison between PCA(in this paper) and RANSAC

| Algorithm | Rate of Abnormal Points(%) | Times(s) |
|-----------|---------------------------|----------|
|           | CIR | SR  | CIR | SR             |
| PCA       | 100 | 7.1 | 100 | 0.7 | 0.012          |
| RANSAC    | 100 | 23.9| 100 | 6.2 | 0.279          |

4. Experiments

The verification in this paper is based on the environment of Ubuntu 14.04, and the hardware configuration is AMD a12-9700p RADEON R7, 10 COMPUTE CORES 4C+6G 2.50ghz, and 8 gigabytes of memory. We tested Hybrid's SLAM on fifty TUM mono datasets. The TUM mono dataset contains information needed by various direct modules, such as camera exposure parameters, to maximize the performance of photometric photography. And some of the datasets are noticeably missing textures in some scenes (e.g. sequence_40), which tests the system's ability to successfully track and build reusable maps. We used pure direct DSO, pure indirect ORB-SLAM and Hybrid SLAM to test 15 times on each dataset, and finally counted the proportion of tracking failure and tracking time of each frame image, as shown in Table 2.

Table 2. System Tracking Comparison in TUM Mono(Sequence_01 ~ 50)

| SLAM     | Tracking Lost(%) | Tracking Times(ms) |
|----------|------------------|--------------------|
|          | Median | Mean  | Median | Mean   |
| ORB-SLAM | 2.83   | 2.26  | 30.71  | 31.23  |
| DSO      | 0.59   | 0.62  | 3.08   | 3.93   |
| Hybrid   | 0.45   | 0.61  | 8.47   | 7.83   |

The results show that the hybrid method is more robust and has a lower rate of tracking failure. But in terms of tracking time, pure direct-based DSO is better, because the Hybrid method still needs to compute a portion of the feature points and needs to compute additional depth information, which can take up a significant amount of resources. But the trade-off is greater robustness to camera motion, which is clearly not a bad deal.

Similarly, in order to examine the accuracy of reusable global maps, we also compared the pose trajectories of the three systems. As shown in Figure 4.

As we can see from the results, ORB-SLAM is significantly different in scale calculations than the other two systems and tends to converge at the later stages of the tracking, such as sequence_02 and sequence_08 and sequence_18. This is because these data sets have many less texturing scenarios later in the movement, which is obviously not a good thing for ORB-SLAM, which relies on feature point computing. In addition, in these three data sets, DSO and Hybrid methods have a
Figure 4. We randomly selected several data sets, and it can be seen that the hybrid method combines the advantages of the other two methods, which are both better than the other two systems in terms of relative scale calculation and robustness. It’s also more sensitive to camera movements.

more detailed perception of camera motion, so that virtual objects will not deviate too much due to camera motion in practical applications. In sequence_40, ORB-SLAM is completely missing the trace, proving that relying on tracking high gradient pixels makes more sense in augmented reality. In sequence_20, the pure direct DSO drifts significantly at the late tracking stage, while Hybrid is significantly improved by combining the advantages of the feature point method with the loop-closing of ORB-SLAM.

Finally, we read the camera information and realized the effect of augmented reality with the help of OpenGL. During this process, it was found that the virtual object would not deviate significantly under the fast motion of the camera. See Figure 5. However, due to the combination of too much content, in the practical application, it still needs to spend a considerable amount of time to continue to initialize, until the direct module detects the available keyframes and marginalizes them before they can be used.
5. Conclusion
As it turns out, a hybrid slam method has a greater prospect in augmented reality than both direct and indirect methods, because it can adapt to all kinds of situations, from changes in lighting to rapid camera movement. And the practical applications are broader. After the addition of IMU, scene information can be perceived faster and better, and denser local maps can be obtained to realize virtual and real interaction. In the future, it can even be combined with deep learning to develop more immersive augmented reality functions.

Figure 5. Augmented reality effects of moving and shooting from different angles

References
[1] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, “Orbslam: a versatile and accurate monocular slam system,” IEEE transactions on robotics, vol. 31, no. 5, pp. 1147–1163, 2015. Podani, J. (1994) Multivariate Data Analysis in Ecology and Systematics. SPB Publishing, The Hague.
[2] J. Engel, T. Schöps, and D. Cremers, “Lsd-slam: Large-scale direct monocular slam,” in European conference on computer vision. Springer, 2014, pp. 834–849.
[3] R. Mur-Artal and J. D. Tardós, “Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras,” IEEE Transactions on Robotics, vol. 33, no. 5, pp. 1255–1262, 2017.
[4] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “Orb: An efficient alternative to sift or surf,” in 2011 International conference on computer vision. IEEE, 2011, pp. 2564–2571.
[5] M. Muja and D. G. Lowe, “Fast approximate nearest neighbors with automatic algorithm configuration.” VISAPP (1), vol. 2, no. 2, pp. 331–340, 2009.
[6] E. Rosten and T. Drummond, “Machine learning for high-speed corner detection,” in European conference on computer vision. Springer, 2006, pp. 430–443.
[7] J. Bacardit and X. Llorà, “Large-scale data mining using genetics-based machine learning,” Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol. 3, no. 1, pp. 37–61, 2013.
[8] G. Klein and D. Murray, “Parallel tracking and mapping for small ar workspaces,” in 2007 6th IEEE and ACM international symposium on mixed and augmented reality. IEEE, 2007, pp. 225–234.
[9] T. Schöps, J. Engel, and D. Cremers, “Semi-dense visual odometry for ar on a smartphone,” in 2014 IEEE international symposium on mixed and augmented reality (ISMAR). IEEE, 2014, pp. 145–150.
[10] S. Yang and S. Scherer, “Direct monocular odometry using points and lines,” in 2017 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2017, pp. 3871–3877.
[11] J. Engel, V. Koltun, and D. Cremers, “Direct sparse odometry,” IEEE transactions on pattern analysis and machine intelligence, vol. 40, no. 3, pp. 611–625, 2017.
[12] X. Gao, R. Wang, N. Demmel, and D. Cremers, “Ldso: Direct sparse odometry with loop closure,” in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018, pp. 2198–2204.
[13] J. Tang, J. Folkesson, and P. Jensfelt, “Sparse2dense: From direct sparse odometry to dense 3d reconstruction,” IEEE Robotics and Automation Letters, vol. 4, no. 2, pp. 530–537, 2019.
[14] D. V. Ouellette, “Schur complements and statistics,” Linear Algebra and its Applications, vol. 36, pp. 187–295, 1981.
[15] C. Forster, M. Pizzoli, and D. Scaramuzza, “Svo: Fast semidirect monocular visual odometry,” in 2014 IEEE international conference on robotics and automation (ICRA). IEEE, 2014, pp. 15–22.
[16] S. Triputen, K. Schreve, V. Tkachev, and M. Ratsch, “Closedform solution for imu based lsd-slam point cloud conversion into the scaled 3d world environment,” arXiv preprint arXiv:1707.05982, 2017.
[17] S. H. Lee and J. Civera, “Loosely-coupled semi-direct monocular slam,” IEEE Robotics and Automation Letters, vol. 4, no. 2, pp. 399–406, 2018.
[18] J. Civera, A. J. Davison, and J. M. Montiel, “Inverse depth parametrization for monocular slam,” IEEE transactions on robotics, vol. 24, no. 5, pp. 932–945, 2008.
[19] R. Ranftl, V. Vineet, Q. Chen, and V. Koltun, “Dense monocular depth estimation in complex dynamic scenes,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 4058–4066.
[20] J. Engel, V. Usenko, and D. Cremers, “A photometrically calibrated benchmark for monocular visual odometry,” arXiv preprint arXiv:1607.02555, 2016.
[21] G. Grisetti, R. Kümmerle, H. Strasdat, and K. Konolige, “g2o: a general framework for (hyper) graph optimization,” in Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China, 2011, pp. 9–13.
[22] H. Hirschmuller and D. Scharstein, “Evaluation of cost functions for stereo matching,” in 2007 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2007, pp. 1–8.
[23] D. Gálvez-López and J. D. Tardos, “Bags of binary words for fast place recognition in image sequences,” IEEE Transactions on Robotics, vol. 28, no. 5, pp. 1188–1197, 2012.
[24] R. Mur-Artal and J. D. Tardós, “Fast relocalisation and loop closing in keyframe-based slam,” in 2014 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2014, pp. 846–853.
[25] P. J. Rousseeuw and M. Hubert, “Robust statistics for outlier detection,” Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol. 1, no. 1, pp. 73–79, 2011.