Semi-Direct Visual Odometry and Mapping with RGB-D Camera

Xinliang Zhong, …Authors put here

***Abstract*— We propose a …**

# I. INTRODUCTION

The problem of simultaneous localization and mapping (SLAM) is one of the hotspots in the field of robotics and computer vision community over the past decade. Precise positioning is the basis for robot control and navigation in GPS-denied environments. Especially for micro aerial vehicle (MAV) working in complex and cluttered unknown indoor environments. They need to constantly update their position at high rates and low latency for position and orientation control. At the same time, they can only carry limited weight and power consumption of the sensor and the processor. While previously many SLAM systems relied on expensive and heavy laser scanners [1], [2] . RGB-D cameras based on structured light provide a powerful alternative and well suited for such application. For instance, the Asus Xtion sensor provides both color and depth images directly in real-time. As the sensor weighs only 77 grams and consumes less than 2.5 watt, it can be easily used for localization, mapping and navigation of MAVs.

To our knowledge, most Visual Odometry (VO) or SLAM systems are feature based [3], [4], [5], [6] which typically extract sparse keypoints from the camera image and then estimate the motion from the consecutive frame. In contrast to feature based methods, direct methods [10], [11] which use all the pixels based on photometric error minimization are becoming increasingly popular.

In this paper we propose, to our best knowledge, a semi-direct VO and mapping system which inherits feature based and direct methods tightly to improve the precision and robustness. We track the pixels with gradient and choose keyframes which are used for mapping and loop closure. A robust sensor model based on the t-distribution [9], [10] and an error function which mixes the depth error and photometric error are used to our SLAM system. In contract to some VO-only based system [7], we propose a hybrid approach that combines the state-of-art loop closure method **—** Bag of Words (BoW) [8] and method based on spatial location constraints. Our method achieves higher robustness and precision just using CPU, which can be easily migrated to embedded devices and applied to MAVs.

The main contributions of this paper are:

• The first one put here…

• The second put here…

• The third put here…

Section II provides… and Section III, thereafter, introduces… Section IV and Section V explain …

# II. RELATED WORK

Visual SLAM approaches, also referred to as “structure and motion estimation” (SFM) [\*] compute the robot’s motion and the map using cameras as sensors. A series of works for camera pose estimation and optimization using RGB-D data have been published over the past few years. According to the implementation, we classify them as following categories.

a) *Feature-Based Methods:* The standard approach is to extract feature from consecutive images by keypoint detectors and descriptors such as SIFT [\*], SURF [\*] and ORB [\*]. Camera pose can be estimated by matching the keypoints to last frame [\*], [\*], [\*]. Local Bundle Adjustment (BA) [\*] and Random sample consensus (RANSAC) [\*] are applied to refining the matches and restricting the outliers to ensure the precise transformation between frames. The first RGB-D SLAM system was proposed by Henry et al. [\*] who extract SIFT features by SIFTGPU in combination with ICP algorithm [\*].

They created and optimized the pose graph by a sparse BA method. Similarly, in order to make the system more general, Endres et al. [\*] used SURF and ORB features to realize the VO. They accomplished the pose-graph optimization by g2o framework instead of BA. Yet, all of these approaches ignore most of the image information, because a few part of (typically 200-500) interest points are extracted to use.

b) *Direct Methods:* Direct methods aim at using the whole image to estimate the structure and motion. Koch [\*] showed that the camera pose can be estimated efficiently by minimizing the photometric error which can be seen as an extension of the Lukas-Kanade tracker [\*] between the observed images. KinectFusion [\*] is an impressive approach that using a variant ICP for image model alignment. Drawbacks of ICP algorithm include the dependency on a good initial guess to avoid getting stuck in a local minimum. Similarly, DTAM is another SLAM approach which based on photometric error. However, too much GPU memory and extensive computational power are required to make both KinectFusion and DATM run in real-time. A robust odometry estimation were proposed by Kerl et al. [\*]. They use a robust photometric error function based on t-distribution that reduces the influence of large residuals between consecutive images. In their further study, an entropy-based criterion is proposed for keyframes selection and pose graph optimization. Such method is also applied in monocular SLAM algorithm like LSD-SLAM [\*]. Need to add something…

c) Semi-*Direct Methods:* Recently, Semi-Direct method approach is popular in monocular SLAM system. Engel et al. [\*] introduced an efficient epipolar search method, enabling real-time VO and semi-dense map reconstruction. An efficient Semi-Direct monocular VO (SVO) [\*] approach was proposed by Forster et al. Their algorithm is applied to MAV state estimation in GPS-denied environments and run at 55 frames per second on the onboard embedded computer. However, the success of SVO depend on a good depth initialization result from the probabilistic depth filter. Furthermore, it is only a monocular VO system without loop closure, which means drift is more likely to occur. Bu et al. [\*] improve it with loop closure detection and extend the approach with RGB-D cameras.

In contrast to all previous work on RGB-D SLAM system, our method inherits the advantages of feature-based and direct method tightly. Put the main contribution here!!!

# III. APPROACH

1. *System Overview*

In general, a visual SLAM system can be divided into three modules [\*]: Frontend, backend and final map representation. The Frontend is mainly focus on computing the camera’s motion from a sequence of images. As the existence of inherent and estimation uncertainty, backend is used to optimize the camera’s pose by a maximum likelihood solution [\*] or some other methods. With the trajectory between frames, we can generate a map (usually point clouds) for what we have observed. Figure 1 shows an overview of our system. The algorithm uses three parallel threads, one for estimating the camera motion, and the second one for optimizing the camera’s poses and the third one for mapping.

The motion estimation thread implements the proposed semi-direct visual odometry. In contrast to estimating the camera poses by minimizing photometric error between pixels corresponding to the projected location of the same 3D points (see Figure 3), our approach creates an error function which mixes the photometric error and depth error as the depth error function is stated in Figure 2. Pixels with gradient are tracked and a weighting function based on t-distribution is used to the residual between pixels. Some corresponding points may not contain valid depth measurements (the depth value is assigned to be zero when it is not valid), to enhance the robustness of the system these points may get a small weight and be ignored because of the huge residual.

In the graph optimization thread, USE G2O FRAME WORK, MIX THE DOW AND spatial location constraints.

After the …

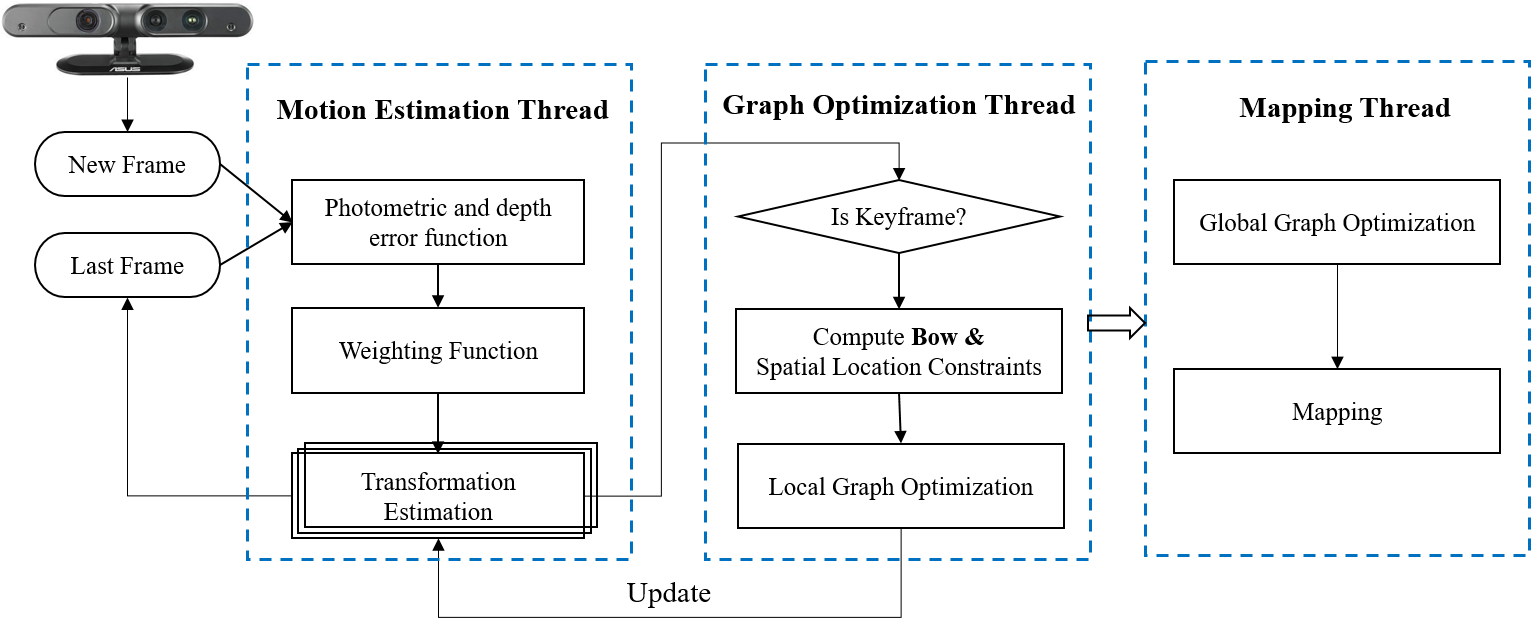


Fig. 1: Semi-Direct Visual Odometry and mapping pipeline

1. *Motion Estimation*

Since our algorithm mixes the photometric error and depth error as the error function to optimize the camera’s pose, we need to combine them reasonably to make Motion Estimation Thread work. The Asus Xtion sensor obtains depth information with an infrared (IR) projector and an IR camera. Figure 2 shows the Simplified model of the RGB-D camera. Point ** is the projector origin and point  is the origin of IR camera. P is the point with ground-truth depth Z.  is the depth estimation corresponding to error *e* in the image plane. The following equations can be obtained from the triangle similarity relationship:

 (1)

where *f* is the focal length and *B* is the distance between projector center and camera center. With further derivation of equation (1) we can obtain:

 (2)

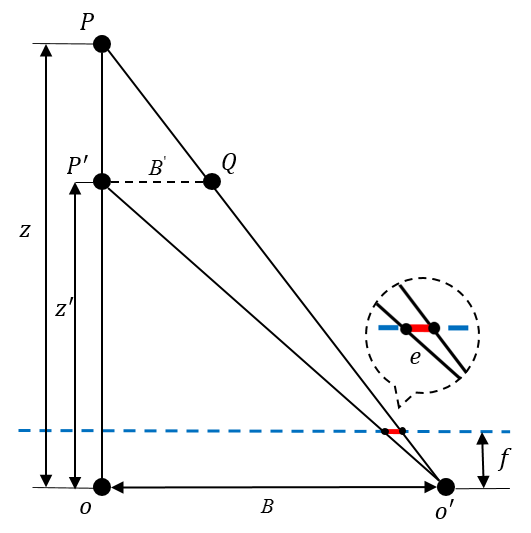
It shows that error *e* is linear with inverse depth. That is the reason inverse depth rather than depth is used for optimization.

Before the Motion Estimation Thread is detailed, we briefly define the notion that is used throughout the paper.

Any 3D world point ***P*** = (X, Y, Z)T maps to the image coordinates ***p* =** (u, v, 1/*d*)T through the pinhole camera projection model ::

 (3)

where *f*x,*f*y are the focal lengths and *c*x, *c*y are the coordinates of the camera center in the standard pinhole camera model.

Fig. 2: The relationship between depth errorand error *e.*

On the contrary, the world 3D point corresponding to an image coordinate p can be recovered, given the inverse projection and the depth *d* :

 (4)

For the moment, we assume that a world point *Pi* is observed by consecutive frame *Ik-*1 and *Ik* (see Figure 3). In the coordinate frame of *Ik*, the point *Pi* is rotated and translated according to the rigid body motion . It comprises a rotation represented as a orthogonal matrix  and a translation represented as a vector . Consequently, the point *Pi* in the frame *Ik* is given as:

 (5)

During the optimization, we need a minimal representation of the transformation and, thus, use Lie algebra  corresponding to the tangent space of  at the identity. The rotation matrix and translation vector of *T* can be calculated from **** with the exponential map [\*]:

 (6)

here **** corresponding to the twist coordinates, and 

,where and is called the angular velocity and linear velocity respectively.

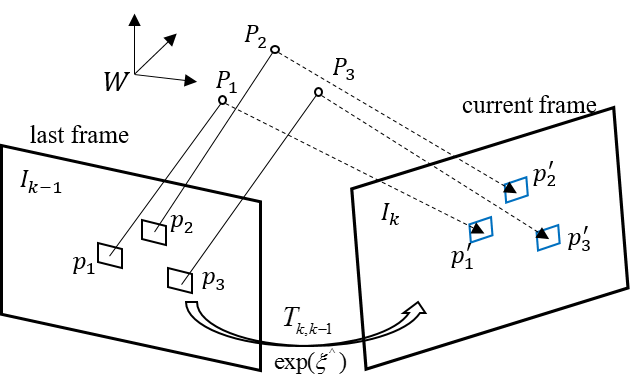


Fig. 3:

The proposed method computes an initial guess of the relative camera motion and the feature correspondences using direct methods. As we can see in Figure 3, we assume that the photo-consistency assumption holds equally for all n pixels (pixels with high gradient) *p*i with  in the image.The world point *P*i is corresponding to the *p*i in the image. The intensity residualis defined by the photometric difference between pixels observing the same 3D point. It can be computed by back-projecting a 2D point *p*i from the last frame  and subsequently projecting it into the current camera view :

 (7)

According to equation (3) and (4), we can get:

 (8)

where the depth  is known at frame  and for which the back-projected points  should be visible in the current frame.

In contrast to other RGB-D slam works [ ], [ ], [ ], we propose a robust error function which inherits both pixels and depth error to estimate the camera motion:



 (9)

Theoretically, the error  should be zero. However, due to the sensor and measurement noises, the residuals would be distributed according to the probabilistic sensor model  or . Our goal is to find the camera motion or  that maximizes . By assuming that the noise is independent and identically distributed, using Bayes’s rule we can get:

 (10)

Note that  denotes the prior distribution over camera motion. We can estimate the camera motion  by minimizing the squared error function:

 (11)

The equation (11) then corresponds to the least squares problem: .

We use an robust *weighting function*  based on t-distribution [ ], as it describes the intensity of a particular error  is considered during minimization. Furthermore, we obtain which minimizes the weighted least squares problem:

 (12)

Note that we consider that  is normally distributed, then  is constant, which means that the equation (12) is the same as normal least squares minimization.

Since equation (12) is nonlinear in , we use the Gauss-Newton method to iteratively solve it. For this, we linearize around the current state by computing the first order Taylor approximation of :

 (13)

Where is computed with the chain-rule:

 (14)

In conjunction with equation (3) and the basic Lie algebra theory, we can get the full Jacobian matrix:

(15)

With the approach described so far, the camera motion can be estimated accurately by the RGB-D frames which contain both color and depth image.

1. *Local Graph Optimization*

The above procedure is good enough to estimate the camera motion from the consecutive frames. However, the estimated error which may cause drift to the final camera’s pose is unavoidable. To reduce the drift and improve the accuracy of camera motion, the camera pose should be aligned with respect to the map points, rather than to the last frame. In addition, keyframes are also used to solve this problem. On one hand, it can simplify the map because the information between two consecutive frames is basically the same if the camera’s movement is not too fast. On the other hand, it can be used to loop closure which is significant to the robustness of the whole map (cf. Section D).

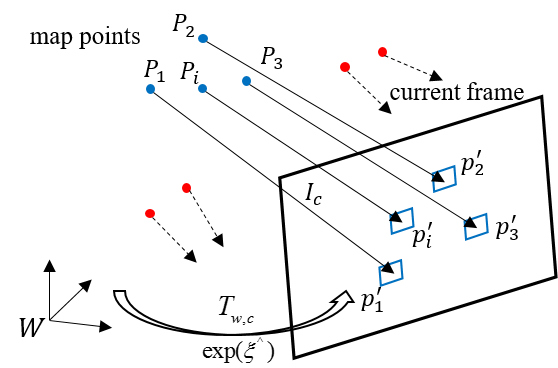


Fig. 4:

Assuming that we have got parts of map around the environments from the RGB-D frames (see Fig. 4), our goal is estimated the camera motion . Map points which are visible in the current frame (blue points in Fig. 4) are used to estimate camera pose. Similarly, we can optimize the camera pose  to minimize the reprojection residuals:

 (16)

Normally, the map points are the set of those keypoints in the keyframes. We insert it to the map if the estimation is good enough. Keyframes are selected by determine the difference between the current camera's motion and the last keyframe. We choose a frame as a keyframe if the rotation or translation of the camera beyond the threshold we set. To make the map points more robustness, several procedures are need to be executed:

1. For new keypoints that don’t have valid depth, we find their correspondences through the epipolar search in the adjacent key frames and get their positions from the triangulation.
2. A map point may be observed by several keyframes, we need to combine the same map point to reduce redundancy.
3. Bad map points that are observed by less than 3 keyframes should be removed because they are harmful to motion estimation and local graph optimization.

Note that the first frame is chosen as keyframe to initialize the map.

For the moment, we can estimate the camera motion and create the map around the unknown environments from the observed images. In order to further improve the accuracy and reduce the drifts of the motion estimation. We can create a pose graph, which is a graph where the camera poses are the nodes and the rigid-body transformations between the camera poses are the edges between nodes. Each additional transformation that is known can be added as an edge into the pose graph. We again optimize the camera pose  to minimize the reprojection residuals:

 (17)

Note that we not only optimize the camera’s pose but also optimize the 3D map points . This is the well- known problem of Bundle Adjustment (BA) and can be solved using a nonlinear optimization algorithm such as Levenberg-Marquardt. In this paper, G2O framework is used to create the pose graph and solve the problem of BA.

1. *Loop Closure*

# REFERENCES

1. Shen, Shaojie, N. Michael, and V. Kumar. "Autonomous multi-floor indoor navigation with a computationally constrained MAV." IEEE International Conference on Robotics and Automation IEEE, 2011:20-25.
2. Beul, M, et al. "A high-performance MAV for autonomous navigation in complex 3D environments." International Conference on Unmanned Aircraft Systems 2015:1241-1250.
3. Blösch, M, et al. "Vision based MAV navigation in unknown and unstructured environments." IEEE International Conference on Robotics and Automation IEEE, 2010:21-28.
4. Weiss, Stephan, et al. "Monocular Vision for Long‐term Micro Aerial Vehicle State Estimation: A Compendium." Journal of Field Robotics 30.5(2013):803-831.
5. Kneip, Laurent, M. Chli, and R. Siegwart. "Robust Real-Time Visual Odometry with a Single Camera and an IMU." British Machine Vision Conference 2011.
6. Engel, Jakob, J. Sturm, and D. Cremers. "Accurate Figure Flying with a Quadrocopter Using Onboard Visual and Inertial Sensing." IMU (2012).
7. Forster, Christian, M. Pizzoli, and D. Scaramuzza. "SVO: Fast semi-direct monocular visual odometry." IEEE International Conference on Robotics and Automation IEEE, 2014:15-22.
8. Mur-Artal, Raul, and J. D. Tardós. "Fast relocalisation and loop closing in keyframe-based SLAM." IEEE International Conference on Robotics and Automation IEEE, 2014:846-853.
9. Kerl, Christian, J. Sturm, and D. Cremers. "Dense visual SLAM for RGB-D cameras." Ieee/rsj International Conference on Intelligent Robots and Systems IEEE, 2014:2100-2106.
10. Kerl, C, J. Sturm, and D. Cremers. "Robust odometry estimation for RGB-D cameras." IEEE International Conference on Robotics and Automation IEEE, 2013:3748-3754.
11. Tykkälä, Tommi, C. Audras, and A. I. Comport. "Direct Iterative Closest Point for real-time visual odometry." IEEE International Conference on Computer Vision Workshops, ICCV 2011 Workshops, Barcelona, Spain, November DBLP, 2011:2050-2056.