Article

SD-VIS: A Fast and Accurate Semi-Direct Monocular Visual-Inertial Simultaneous Localization and Mapping (SLAM)

Quanpan Liu, Zhengjie Wang * and Huan Wang
School of Mechatronical Engineering, Beijing Institute of Technology, Beijing 100081, China; liuquanpan@126.com (Q.L.); WH978992767@163.com (H.W.)
* Correspondence: wangzhengjie@bit.edu.cn; Tel.: +86-186-1164-6723

Received: 17 February 2020; Accepted: 7 March 2020; Published: 9 March 2020

Abstract: In practical applications, how to achieve a perfect balance between high accuracy and computational efficiency can be the main challenge for simultaneous localization and mapping (SLAM). To solve this challenge, we propose SD-VIS, a novel fast and accurate semi-direct visual-inertial SLAM framework, which can estimate camera motion and structure of surrounding sparse scenes. In the initialization procedure, we align the pre-integrated IMU measurements and visual images and calibrate out the metric scale, initial velocity, gravity vector, and gyroscope bias by using multiple view geometry (MVG) theory based on the feature-based method. At the front-end, keyframes are tracked by feature-based method and used for back-end optimization and loop closure detection, while non-keyframes are utilized for fast-tracking by direct method. This strategy makes the system not only have the better real-time performance of direct method, but also have high accuracy and loop closing detection ability based on feature-based method. At the back-end, we propose a sliding window-based tightly-coupled optimization framework, which can get more accurate state estimation by minimizing the visual and IMU measurement errors. In order to limit the computational complexity, we adopt the marginalization strategy to fix the number of keyframes in the sliding window. Experimental evaluation on EuRoC dataset demonstrates the feasibility and superior real-time performance of SD-VIS. Compared with state-of-the-art SLAM systems, we can achieve a better balance between accuracy and speed.

Keywords: visual-inertial; semi-direct SLAM; multi-sensor fusion

1. Introduction

Simultaneous localization and mapping (SLAM) plays an important role in self-driving cars, virtual reality, unmanned aerial vehicles (UAV), augmented reality and artificial intelligence [1,2]. This technology can provide reliable state estimation for UAV and self-driving cars in GPS-denied environments by relying on its sensors. Various types of sensors can be utilized in SLAM, such as stereo camera, lidar, inertial measurement units (IMU), and monocular camera. However, they have significant disadvantages when used individually: the metric scale of stereo camera can be obtained directly by using fixed baseline length, but it can only be estimated accurately in a limited depth range [3]; lidar has high precision in indoor, but it will encounter the reflection problem of glass surface in outdoor [4]; cheap IMUs are extremely susceptible to bias and noise [5]; monocular camera cannot estimate the absolute metric scale [6]. This paper mainly studies the monocular vision-inertial navigation system (VINS) based on multi-sensor fusion [7], which has the advantages of small size, lightweight, observable scale, roll, and pitch angle, etc.

According to the different methods of image information processing, there are two categories of SLAM: feature-based method and direct method. The standard process of feature-based method
proceeds in three steps [8–10]. Firstly, a group of sparse point or line features is extracted from each image, and feature matching between adjacent frames is performed by using invariant feature descriptors. Secondly, estimate the camera motion and the 3D position of sparse feature points by using multiple view geometry theory. Finally, optimize the camera motion and the 3D position of sparse feature points by minimizing visual re-projection errors. However, the feature-based method has the following disadvantages: feature extraction and matching are very time-consuming; fewer features extractable in low-texture environments; repeated texture environments will cause incorrect feature matching. Therefore, the accuracy and robustness of the feature-based method depend on the correct feature extraction and matching.

The direct method considers the entire image or some pixels with a large gradient and directly estimates the camera motion and scene structure by minimizing the photometric error [11–14]. Therefore, in the low-texture environments and repeated texture environments, the performance of the direct method is better than the feature-based method. In addition, without feature extraction and matching, the calculation speed of the direct method is faster than the feature-based method. However, the photometric error function is highly non-convex, it is difficult for the direct method to converge when large baseline motion and image blur occurs. Due to the inability to perform effective loop closure detection, the cumulative drift generated during long-term operation is still an unresolved problem [15].

Academic research on SLAM is very extensive. Well known methods include PTAM, SVO, VINS-Mono, LSD-SLAM, ORB-SLAM, DSO. PTAM was one of the most representative systems in the early stage of the feature-based SLAM algorithm. This algorithm first proposed to divide tracking and mapping into two parallel threads [16]. After that, most feature-based SLAM algorithms have adopted this idea, including ORB-SLAM. ORB-SLAM is the most successful feature-based SLAM, which uses ORB features in tracking, mapping, re-location, and loop closure detection [17]. VINS-Mono is a compelling multi-sensor monocular vision-inertial SLAM based on tight coupling and non-linear optimization [18]. PL-VIO is an extension of the line feature of VINS-Mono, which can optimize the re-projection error of point and line features in a sliding window [19]. VINS-Fusion is a multi-sensor fusion platform based on non-linear optimization, which is the continuation of VINS-Mono in the direction of multi-sensor fusion [20]. LSD-SLAM is a novel method of real-time monocular SLAM based on direct method and can create large-scale semi-dense maps in real-time on a laptop without GPU acceleration [21]. Based on LSD-SLAM, DSO adds complete photometric calibration and uniformly samples pixels with large gradients throughout the image, thereby improving tracking accuracy and robustness [11]. On the basis of DSO, LDSO adds loop closure detection, which makes up for the shortcomings of DSO incapable of loop closure detection and eliminates the cumulative error generated during long navigation [22]. Stereo DSO is an improved version of DSO, which can realize the real-time estimation of motion and 3D structure with high accuracy and robustness on the moving stereo camera [3]. Lee authors in [7] presents a new implementation method for efficient simultaneous localization and mapping using a forward-viewing monocular vision sensor. The method is developed to be applicable in real time on a low-cost embedded system for indoor service robots.

In recent years, SLAM systems that combine the feature-based method and direct method have become popular. Semi-direct visual odometry (SVO) is an effective hybrid method that combines the advantages of feature-based method and direct method [23]. However, since the algorithm was originally designed for the drone’s look-down camera, it is easy to lose camera tracking in other settings. Lee [24] proposed a loose-coupled method by combining ORB-SLAM and DSO to improve positioning accuracy. However, its frontend and backend are almost independent, which cannot share estimation information to further improve the pose precision. In [25,26], different semi-direct approaches were proposed for stereo odometry. Both methods use feature-based tracking to obtain a motion prior, and then perform direct semi-dense or sparse alignment to refine the camera pose. SVL [27] can be considered a combination of ORB-SLAM and SVO. The method for ORB-SLAM is adopted in keyframes, and SVO is adopted in non-keyframes. Therefore SVL can achieve good balance
between speed and accuracy according to camera motion and environment. Our method is motivated by SVL, but we go one step further in real-time performance. Specifically, thanks to the keyframe selection strategy and sliding window-based back-end, we only need to extract new feature points on the keyframes and track them with KLT sparse optical flow algorithm, which can further reduce the calculation complexity while ensuring accuracy.

In this paper, we present SD-VIS, a novel fast and accurate semi-direct visual-inertial SLAM framework, which combines the exactness of feature-based method and quickness of direct method. The keyframes in SD-VIS are tracked by feature-based method, which is used for sliding window-based non-linear optimization and loop closure detection. This strategy solves the problem of drift in the long-term operation and ensures the robustness of the algorithm in case of large baseline motion and image blur. Non-keyframes are tracked by the direct method, and the distance between adjacent non-keyframes is minimal, which ensures the convergence of error function. Compared with the direct method, SD-VIS exhibits the function of loop closure detection and solves the problem of drift in long-term operation. Compared with the feature-based method, SD-VIS can achieve the same accuracy while maintaining a faster speed.

2. System Framework Overview

Figure 1 demonstrates the framework of the semi-direct vision-inertial SLAM system. Sensor data comes from a monocular camera and IMU. IMU measurements between two consecutive images are pre-integrated, and the pre-integration is used as the constraint of IMU between two images (Section 3.2). In the initialization procedure, we detect the feature points on each image and track them with KLT sparse optical flow algorithm [15].

![Figure 1. The semi-direct visual-inertial SLAM system framework](image-url)
In the following visual-inertial alignment, we align the pre-integrated IMU measurements and visual images and calibrate out the metric scale, initial velocity, gravity vector, and gyroscope bias by using multiple view geometry (MVG) theory based on the feature-based method. (Section 3.3). After initialization, keyframe selection will be performed based on the IMU pre-integration and previous feature matching results. The previous feature matching refers to the feature matching between the last frame and the penultimate frame in the sliding window, and the matching is completed before the current frame arrives. Non-keyframes are used for fast-tracking and localization by direct method [11], and keyframes are tracked by feature-based method [18] and used for non-linear optimization and loop closure detection (Section 4). In the following tight coupling optimization framework, we can get more accurate state estimation by minimizing visual re-projection error, IMU residual, prior information from marginalization, and re-location information from loop closure detection (Section 5).

3. IMU Measurements and Visual-Inertial Alignment

3.1. Definition of Symbols

Figure 2 shows the definition of symbols in the semi-direct visual-inertial SLAM framework. C, B, and W are the camera coordinate system, the IMU body coordinate system, and the world coordinate system, respectively. We define \( T_{WB} = ( R_W^B, P_W^B ) \) as the motion of B relative to W. \( T_{BC} = ( R_C^B, P_C^B ) \) represents the extrinsic parameters between C and B, which can be calibrated in advance. \( T_{t:t+1} \in SE(3) \) represents the motion from time \( t \) to time \( t + 1 \) in the coordinate system C, and \( Z_{t:t+1} \) represents a pre-integrated IMU measurement between the camera coordinate system \( C_t \) and \( C_{t+1} \).

![Figure 2. Symbol definition of algorithm](image)

A 3D point \( F_1, F_2 \in \mathbb{R}^3 \) represents the spatial feature points observed simultaneously by the camera coordinate system \( C_t \) and \( C_{t+1} \). \( P_1, P_2, P_3, P_4 \in \mathbb{R}^2 \) are the projections of feature points on the image coordinate system. We adopt the traditional pinhole camera model to map the \( F_1 \) in the camera coordinate system to the image coordinate system by the projection function \( \pi: \mathbb{R}^3 \to \mathbb{R}^2 \):

\[
P_1 = \pi F_1 = \begin{bmatrix} f_u \frac{x_c}{z_c} + c_u \\ f_v \frac{y_c}{z_c} + c_v \end{bmatrix} F_1 = \begin{bmatrix} x_c & y_c & z_c \end{bmatrix}^T
\]

(1)

where \( \begin{bmatrix} f_u & f_v \end{bmatrix}^T \) and \( \begin{bmatrix} c_u & c_v \end{bmatrix}^T \) is camera internal parameters.
3.2. IMU Pre-Integration

In the back-end optimization and visual-inertial alignment, the constraints of vision and IMU need to be optimized in the same frame, so the IMU measurements between two adjacent frames need to be integrated into one constraint.

IMU can output 3-axis angular velocity \( \tilde{\omega} \) and 3-axis acceleration \( \tilde{\alpha} \) including bias and Gaussian white noise:

\[
\tilde{\omega} = \omega^{B} + b^{B}_{\omega} + n^{B}_{\omega} \tag{2}
\]
\[
\tilde{\alpha} = R^{B}_{W}(\alpha^{W} + g^{W}) + b^{B}_{\alpha} + n^{B}_{\alpha} \tag{3}
\]

where \( n^{B}_{\omega} \sim N(0, \sigma^{2}_{\omega}) \), \( n^{B}_{\alpha} \sim N(0, \sigma^{2}_{\alpha}) \) represents the Gaussian white noise. \( g^{W} = [0, 0, g]^{T} \) is the gravity vector. \( R^{B}_{W} \) represents the rotation matrix from W to B. \( b^{B}_{\omega}, b^{B}_{\alpha} \) represents the biases of gyroscopes and accelerometer.

We define \( p^{W}_{B_{j}}, v^{W}_{B_{j}}, q^{W}_{B_{j}} \) as the translation, velocity, and rotation quaternions in the IMU body coordinate system \( B_{j} \). If we know the measurements value of IMU during \( t \in [i, j] \), we can calculate \( p^{W}_{B_{j}}, v^{W}_{B_{j}}, q^{W}_{B_{j}} \) in the coordinate system \( B_{j} \):

\[
p^{W}_{B_{j}} = p^{W}_{B_{i}} + v^{W}_{B_{i}} \Delta t + \int_{t \in [i, j]} [R^{W}_{B_{i}}(\tilde{\alpha}_{t} - b^{B}_{\alpha}) - g^{W} \Delta t^{2}] \delta t \tag{4}
\]
\[
v^{W}_{B_{j}} = v^{W}_{B_{i}} + \int_{t \in [i, j]} [R^{W}_{B_{i}}(\tilde{\alpha}_{t} - b^{B}_{\alpha}) - g^{W} \Delta t] \delta t \tag{5}
\]
\[
q^{W}_{B_{j}} = q^{W}_{B_{i}} \otimes \int_{t \in [i, j]} \frac{1}{2} \Omega(\tilde{\omega}_{t} - b^{B}_{\omega}) q^{W}_{B_{i}} \delta t \tag{6}
\]

where:

\[
\Omega(\omega) = \begin{bmatrix}
-\omega_{x} & \omega_{y} & \omega_{z} \\
-\omega_{y} & \omega_{z} & -\omega_{x} \\
-\omega_{z} & -\omega_{x} & \omega_{y}
\end{bmatrix},
\omega_{x} = \begin{bmatrix}
0 & -\omega_{z} & \omega_{y} \\
\omega_{z} & 0 & -\omega_{x} \\
-\omega_{y} & \omega_{x} & 0
\end{bmatrix}
\]

From formulas (4)–(6), we can know that the IMU body state \( p^{W}_{B_{j}}, v^{W}_{B_{j}}, q^{W}_{B_{j}} \) is related to the IMU body coordinate system \( B_{j} \). In the back-end tightly coupling optimization, we will continuously iteratively update the IMU state variables in the sliding window. When the IMU body state at time \( t = i \) is iteratively updated, we need to recalculate the state at time \( t = j \), which is very time-consuming. We adopt IMU pre-integration technology to avoid unnecessary time consumption. Formulas (4)–(6) can be written as:

\[
R^{B}_{W} p^{W}_{B_{j}} = R^{B}_{W}(p^{W}_{B_{i}} + v^{W}_{B_{i}} \Delta t - \frac{1}{2} g^{W} \Delta t^{2}) + q^{B}_{B_{j}} \tag{7}
\]
\[
R^{W}_{B_{j}} v^{W}_{B_{j}} = R^{W}_{B_{i}}(v^{W}_{B_{i}} \Delta t - g^{W} \Delta t) + \beta^{B}_{B_{j}} \tag{8}
\]
\[
q^{B}_{W} \otimes q^{W}_{B_{j}} = \gamma^{B}_{B_{j}} \tag{9}
\]

where:

\[
\delta^{B}_{B_{j}} = \int_{t \in [i, j]} [R^{B}_{B_{i}}(\tilde{\alpha}_{t} - b^{B}_{\alpha})] \delta t^{2} \tag{10}
\]
\[
\beta^{B}_{B_{j}} = \int_{t \in [i, j]} [R^{B}_{B_{i}}(\tilde{\alpha}_{t} - b^{B}_{\alpha})] \delta t \tag{11}
\]
\[
\gamma^{B}_{B_{j}} = \int_{t \in [i, j]} \frac{1}{2} \Omega(\tilde{\omega}_{t} - b^{B}_{\omega}) q^{B}_{B_{i}} \delta t \tag{12}
\]
where:

\[ q_{B_i}^{C_0} = q_{C_i}^C \otimes q_{B_i}^{C_0} \]

From formulas (10)–(12), the pre-integration measurements \( \theta_{B_i}^{B_i}, \beta_{B_i}^{B_i}, \gamma_{B_i}^{B_i} \) and the IMU body coordinate system \( B_i \) are independent of each other. This means that when the states in the \( B_i \) coordinate system are iteratively updated, there is no need to recalculate the states in the \( B_j \) coordinate system. Since the IMU pre-integrated measurements \( \theta_{B_i}^{B_i}, \beta_{B_i}^{B_i}, \gamma_{B_i}^{B_i} \) is affected by the bias, when the bias is updated iteratively, we will update the pre-integrated measurement by the first-order approximation method:

\[
\begin{align*}
\theta_{B_i}^{B_i} & \approx \tilde{\theta}_{B_i}^{B_i} + J_{\theta}^{B_i} b_{\theta}^{B_i} + J_{\theta}^{B_i} b_{\omega}^{B_i} \\
\beta_{B_i}^{B_i} & \approx \tilde{\beta}_{B_i}^{B_i} + J_{\beta}^{B_i} b_{\theta}^{B_i} + J_{\beta}^{B_i} b_{\omega}^{B_i} \\
\gamma_{B_i}^{B_i} & \approx \tilde{\gamma}_{B_i}^{B_i} \otimes \begin{bmatrix} 0 \\
\frac{1}{2} J_{\gamma}^{B_i} b_{\omega}^{B_i} \end{bmatrix}
\end{align*}
\]

where \( J_{\theta}^{B_i}, J_{\beta}^{B_i}, J_{\gamma}^{B_i} \) are the Jacobian matrices of pre-integrated measurements with respect to bias.

### 3.3. Visual-Inertial Alignment

The convergence speed and effect of nonlinear visual-inertial SLAM systems depend heavily on reliable initial values. Therefore, in the initialization procedure of SD-VIS, we align the pre-integrated IMU measurements with the visual image to complete the system initialization.

#### 3.3.1. Gyroscope Bias Correction

We regard the camera coordinate system \( C_0 \) as the world coordinate system. We detect the feature points on each image and track them with KLT sparse optical flow algorithm, and then the rotation \( R_{C_i}^{C_1} \) and \( R_{C_{i+1}}^{C_i} \) of the two adjacent frames \( C_i \) and \( C_{i+1} \) can be estimated by using visual structure from motion (SfM). Rotation increment \( \gamma_{B_i}^{B_{i+1}} \) between the IMU body coordinate system \( B_i \) and \( B_{i+1} \) can be estimated by IMU pre-integration. We can get the following formula:

\[
\min_{b_{\theta}^{B_i}} \sum ||R_{C_{i+1}}^{C_i} R_{C_i}^{C_0} \gamma_{B_i}^{B_{i+1}}||^2
\]

where:

\[
\gamma_{B_i}^{B_{i+1}} \approx \tilde{\gamma}_{B_i}^{B_{i+1}} \otimes \begin{bmatrix} 0 \\
\frac{1}{2} J_{\gamma}^{B_i} b_{\omega}^{B_i} \end{bmatrix}
\]

We solve the above least squares problem to get the initial calibration of the gyroscope bias and use it to update \( \theta_{B_i}^{B_i}, \beta_{B_i}^{B_i}, \gamma_{B_i}^{B_i} \).

#### 3.3.2. Gravity Vector, Initial Velocity, and Metric Scale Correction

We define the variables that need to be calibrated as:

\[
X_t = \left[ \begin{array}{cccc}
V_{b_0}^{b_t} & V_{b_1}^{b_t} & \cdots & V_{b_m}^{b_t} & g_{C_0}^{C_t} & s
\end{array} \right]
\]

where \( V_{b_k}^{b_t} \) is the initial velocity in the IMU body coordinate system, \( g_{C_0}^{C_t} \) is the gravity vector in the camera coordinate system, \( s \) represents the metric scale of semi-direct visual-inertial SLAM framework.

Suppose we have obtained pre-calibrated external parameter \( T_{BC} = (R_B^C, P_B^C) \), we can transform the pose \( T_{C_i}^{C_0} = (R_{C_i}^{C_0}, P_{C_i}^{C_0}) \) from the camera coordinate system to the IMU body coordinate system:

\[
q_{B_i}^{C_0} = q_{C_i}^{C_0} \otimes q_{B_i}^{C_0}
\]
\[ s \mathbf{p}_{B_t}^O = s \mathbf{p}_{C_t}^O - R_{B_t}^C \mathbf{p}_{C_t}^B \]  
(20)

Considering two adjacent keyframes \(B_t\) and \(B_{t+1}\), then formulas (7) and (8) can be rewritten as:

\[
\theta_{B_t}^{B_{t+1}} = R_{C_t}^C \left( \mathbf{p}_{B_{t+1}}^C - \mathbf{p}_{B_t}^C \right) - R_{B_t}^C \mathbf{v}_{B_t}^C \Delta t + \frac{1}{2} R_{B_t}^C \mathbf{a}_t \Delta t^2
\]
(21)

\[
\mathbf{p}_{B_{t+1}}^B = R_{B_t}^C \left( R_{B_{t+1}}^C \mathbf{v}_{B_{t+1}}^C - R_{B_t}^C \mathbf{v}_{B_t}^C + \mathbf{g}_t \Delta t \right)
\]
(22)

We combine formulas (19)–(22) to get the following formula:

\[
\mathbf{Z}_{B_{t+1}} \cong \begin{bmatrix}
\theta_{B_t}^{B_{t+1}} - \mathbf{p}_{B_t}^C + R_{B_t}^C \mathbf{p}_{B_{t+1}}^C & \mathbf{p}_{B_{t+1}}^B
\end{bmatrix} = H_{B_{t+1}}^B \mathbf{x}_t + \mathbf{n}_{B_t}^B
\]
(23)

where:

\[
H_{B_{t+1}}^B = \begin{bmatrix}
-I \Delta t & 0 & \frac{1}{2} R_{C_t}^C \mathbf{a}_t \Delta t^2 & R_{B_t}^C \left( \mathbf{p}_{C_t}^C - \mathbf{p}_{C_t}^C \right) \\
-I & R_{B_t}^C \mathbf{p}_{B_t}^C & R_{B_t}^C \Delta t & 0
\end{bmatrix}
\]
(24)

In the above formula \(R_{B_t}^C\), \(R_{B_{t+1}}^C\), \(\mathbf{p}_{B_t}^C\), \(\mathbf{p}_{B_{t+1}}^C\) can be obtained through visual SfM:

\[
\min_{\mathbf{x}_t} \sum \left\| \mathbf{Z}_{B_{t+1}}^B - H_{B_{t+1}}^B \mathbf{x}_t \right\|^2
\]
(25)

Solving the above formula, we can calibrate the initial velocity for each keyframe, gravity vector, and absolute metric scale. After estimating the scale, we will adjust the translation vector of the vision SfM to make the system have an observable scale.

3.3.3. Gravity Vector Refinement

We can know the magnitude of the gravity vector in advance, so we refer to the VINS-Mono [18] method to re-parameterize the gravity vector obtained in Section 3.3.2 with two variables in tangent space, and perform further optimization.

After obtaining the accurate gravity vector, we can rotate the coordinate system \(C_0\), which is temporarily the world coordinate system, to the real world coordinate system \(W\). However, since the yaw angle in the visual-inertial SLAM system is unobservable, the yaw angle of the \(C_0\) coordinate system remains unchanged during the rotation process. At this time, the initialization procedure of the semi-direct visual-inertial SLAM system is completed.

4. Visual Measurements

4.1. Keyframe Selection

We have three different keyframe selection strategies. Satisfying one of these three strategies makes the current frame a keyframe. The first and third strategies of keyframe selection are based on the feature matching results of the last frame and the penultimate frame in the sliding window, which has been matched before the current frame arrives. The first selection strategy is the tracking number of feature points. No new feature points will be extracted when tracking non-keyframes. The translational motion of the camera will lead to the decrease of tracking feature points. If the number of tracking points in the last frame in the sliding window is less than 70% of the minimum tracking point threshold, the current frame will be treated as a keyframe. The second selection strategy is related to IMU pre-integration. If the translation distance between the last two adjacent frames in the sliding window calculated by the IMU pre-integration exceeds a preset threshold, the current frame is also considered as a new keyframe. The third selection strategy is the average parallax of the feature points tracked on the the last frame and the penultimate frame in the sliding window. The translation
or rotation of camera will cause parallax. When the average parallax exceeds the threshold, the current frame will also be regarded as a keyframe.

4.2. Keyframes Tracking

If the current frame is treated as a keyframe, we first use the fast feature detector [28] to add new feature points in the last frame in the sliding window and then use the KLT sparse optical flow algorithm to track them in the current frame (Figure 3). At least 200 feature points will be maintained in each frame. Since there is no need to calculate the feature point descriptor, the optical flow method can save more time. In addition, we also use RANSAC [29] with the fundamental matrix model to eliminate outliers generated during tracking.

![Figure 3. The top chart (a) shows how to track a keyframe. The black triangle represents the newly extracted feature points on the 10th frame (last frame) of the sliding window, while the green dotted line represents tracking them in the keyframe by KLT sparse optical flow algorithm. The bottom chart (b) shows how to track a non-keyframe. The red dotted line indicates that the feature points on the 10th frame of the sliding window are tracked on the non-keyframe by the direct method.](image)

4.3. Non-Keyframes Tracking

If the current frame is considered a non-keyframe, we use direct image alignment to estimate the relative pose $T_{t,t+1}$ between the current frame and the last frame in the sliding window. The initial value of the relative pose can be obtained directly by IMU pre-integration. The feature points observed in the last frame are projected into the current frame according to the estimated pose $T_{t,t+1}$. Due to the hypothesis of photometric invariance, if the same feature point is observed by two adjacent frames, the photometric values of the projection points on the two adjacent frames are equal (Figure 4). Therefore, we can optimize the relative pose $T_{t,t+1}$ by minimizing the photometric error between image blocks ($4 \times 4$ pixels):

$$T_{t,t+1} = \underset{T}{\arg\min} \iint_{R} \rho[\delta l(T, u)]du$$

(26)
where $\rho[.] = \frac{1}{2}\|\cdot\|^2$, $u$ is the position of the feature point of the last frame in the sliding window. $R$ is the image region where the depth is known in the last frame, and the back-projected points are visible in the current frame domain.

Figure 4. Adjusting the relative pose $T_{t,t+1}$ between the current frame and last frame in the sliding window means moving the re-projection position $u'$ of feature points on the current frame.

The photometric error $\delta I$ is:

$$
\delta I(T_{t,t+1},u) = I_{t+1}(\pi T_{t,t+1}^{-1}(u,d_p)) - I_t(u) \quad \forall u \in R
$$

where $d_p$ is the depth of the feature point in the last frame in the sliding window. $I_t$ represents the intensity image in the $k$-th frame.

We use the inverse compositional formulation [30] of the photometric error, which can avoid unnecessary Jacobian derivation. The update step $T(\xi)$ for the last frame in the sliding window is:

$$
\delta I(\xi,u) = I_{t+1}(\pi T_{t,t+1}^{-1}(u,d_p)) - I_t(T(\xi)\pi^{-1}(u,d_p)) \forall u \in R
$$

We solve it in an iterative Gauss Newton method and update $T_{t,t+1}$ in the following way:

$$
T_{t,t+1} \leftarrow T_{t,t+1}T(\xi)^{-1}
$$

After image alignment, we can get the optimized relative pose $T_{t,t+1}$ between the current frame and the last frame in the sliding window. We define all 3D points observed in all frames in the sliding window as the local map, and project the local map to the current frame to find the visible 3D points of the current frame. Due to the inaccuracy of the visible 3D point position and the camera pose, there will be errors in the projection position of the current frame. To make the projection position more accurate, the current frame needs to be aligned with the local map. The feature matching step optimizes the positions of all the projection points in the current frame by minimizing the photometric errors of the projection blocks (5 × 5 pixels) in the current frame and the reference frame (Figure 5):

$$
u'_i = \arg\min_{u'_i} \frac{1}{2}\|I_{t+1}(u'_i) - A_i: I_t(u_i)\|^2 \quad \forall i
$$

Solving the above formula in an iterative Gauss Newton method, we can get the update of the projection block position. The reference frame is usually far away from the current frame, so we apply an affine warping $A_i$ to the reference patch.

Through image alignment and feature matching, we get the implicit results of direct motion estimation — feature correspondence with sub-pixel accuracy. Note that when tracking non-keyframes with the direct method, no new feature points are extracted. In the back-end optimization, we will
combine IMU residual, visual re-projection error, prior information, and re-localization information to optimize the camera pose and 3D point position again.

![Figure 5. Adjust the position of the projection block $u'_i$ on the current frame to minimize the photometric error of the projection block in the current frame and the reference frame in the sliding window.](image)

5. Sliding Window-based Tightly-coupled Optimization Framework

After tracking non-keyframes and keyframes, we proceed with a sliding window-based tightly-coupled optimization framework for high accuracy and robust state estimation. In the optimization framework, we combined IMU residual, visual re-projection error, prior information, and re-localization information to optimize the camera pose and 3D point position again.

5.1. Formulation

The state variables to be estimated by SD-VIS are defined as:

$$X = \left[ \begin{array}{cccc} x_0 & x_1 & \ldots & x_n \\ \alpha & \beta & \ldots & \alpha_{n} \\ \omega & & & \omega_{n} \\ \end{array} \right]$$

$$X_k = \left[ \begin{array}{cccc} p_{B_k}^W & v_{B_k}^W & q_{B_k}^\alpha & b_{B_k}^\omega \\ x_{C_k}^B & y_{C_k}^B & z_{C_k}^B \\ \end{array} \right]_{k \in [0,n]}$$

where $X_k$ includes the translation, velocity, and rotation quaternions of the $k^{th}$ IMU body coordinate system concerning the world coordinate system, as well as the bias of gyroscope and accelerometer. $n$ represents the size of the sliding window. By minimizing the sum of IMU residuals, visual re-projection errors, prior information, and relocation information in the sliding window, we can obtain a robust and accurate semi-direct visual-inertial SLAM system:

$$\min_X \left\{ \sum k \left( \right) \right\}$$

where $r_B(Z_{B_{k+1}}^B, X), r_C(Z_{C_{k+1}}^C, X), r_P, H, P$, and $r_C(Z_{C_{k+1}}^C, X)$ are IMU residuals, visual re-projection errors, prior information and re-localization information respectively.
5.2. IMU Residuals

According to the formulas (4)–(6) in Section 3.2, we can get the IMU measurement residual:

\[
    r_B(z_B^i, X) = \begin{bmatrix}
        \delta \theta_{B1} \\
        \delta \beta_{B1} \\
        \delta \gamma_{B1} \\
        \delta b_{1\alpha} \\
        \delta b_{1\omega}
    \end{bmatrix} = \begin{bmatrix}
        R_W^B(p_{W1}^B - p_{W1}^B - V_{W1}^B \Delta t + \frac{1}{2} g_{W1}^B \Delta t^2) - \theta_{B1} \\
        R_W^B(V_{W1}^B - V_{W1}^B + g_{W1}^B \Delta t) - \beta_{B1} \\
        2(d_{B1}^W - d_{Bj}^W \otimes (\gamma_{Bj}^W)^{-1})_{xyz} \\
        b_{1\alpha} - b_{j\alpha} \\
        b_{1\omega} - b_{j\omega}
    \end{bmatrix}_{15\times1}
\]

where \([\cdot]_{\text{xyz}}\) represents the real part of the quaternion. \([\vec{b}_{B1}, \vec{b}_{Bj}, \vec{b}_{Bj}]^T\) are the IMU pre-integration between two adjacent keyframes \(B_1\) and \(B_j\).

5.3. Visual Re-Projection Errors

When the feature point \(F_1\) is first observed in the \(i\)th image, the visual re-projection error in the \(j\)th image can be defined by the pinhole camera model as:

\[
    r_C(z_F^j, X) = \pi^{-1} \left( \begin{bmatrix} \vec{u}_{F1}^j \\ \vec{v}_{F1}^j \end{bmatrix} - R_C \left( R_W^B(p_{W1}^B + R_C^W(R_C^{Bj} \pi^{-1} \begin{bmatrix} \vec{u}_{F1}^j \\ \vec{v}_{F1}^j \end{bmatrix} + P_C^B + p_{W1}^B - p_{Wj}^B) - P_C^B \right) \right)
\]

where \([\vec{u}_{F1}^j, \vec{v}_{F1}^j]\) and \([\vec{u}_{F1}^j, \vec{v}_{F1}^j]\) represent the coordinates of the pixels projected from the feature point \(F_1\) to the \(i\)th and \(j\)th frame image, respectively. \(\pi^{-1}\) is the back-projection function of the pinhole camera model.

5.4. Marginalization Strategy

In order to limit the computational complexity of SD-VIS, the back-end adopts a sliding window-based tightly-coupled optimization framework, so we use the marginalization strategy [31] to make the correct operation of sliding windows. As shown in Figure 6, if the current frame is determined as a keyframe, the frame will remain in the sliding window, and the oldest frame is marginalized out. When the oldest frame is marginalized, the feature points that can only be observed by the oldest frame will be discarded directly, and other visual and inertial measurements associated with the frame will be removed from the sliding window by Schur complement. The new prior information constructed by Schur complement will be added to the existing prior information. If the current frame is not a keyframe, the last frame in the sliding window will be marginalized, and all visual measurements related to that frame will be removed directly from the sliding window.

5.5. Re-Localization

Due to the global 3D position and yaw angle are unobservable, there will be inevitable accumulative errors in the vision-inertial SLAM system. To eliminate the accumulated error, we introduce the re-localization module. After the keyframe is traced successfully, it can be judged whether the SLAM system has been here before by loop closure detection. We utilize DBoW2, a state-of-the-art bag-of-word place recognition approach, for loop closure detection. When a loop is detected, the re-localization module can effectively align the current sliding window, thus eliminating the accumulated error. For a detailed description of re-location, readers may refer to [18].
observable, there will be inevitable accumulative errors in the vision-inertial SLAM system. To eliminate the accumulated error, we marginalize strategy of SD-VIS.

Figure 6. Marginalization strategy of SD-VIS.

6. Experiment

We evaluate the accuracy, robustness, and real-time performance of SD-VIS on the EuRoC dataset [32]. The SD-VIS method is compared with the state-of-the-art vision SLAM methods, such as VINS-mono [18] and VINS-Fusion [20]. In Section 6.1, the accuracy and robustness of SD-VIS are evaluated, and the experimental results show that the accuracy and robustness of the proposed method reach the same level as the state-of-the-art method. Section 6.2 evaluates real-time performance. The experimental results show that the proposed method achieves a good balance between accuracy and real-time performance. Section 6.3 evaluates the loop closure detection capability and verifies the overall feasibility of the SLAM system.

6.1. Accuracy and Robustness Evaluate

In the experiments on the EuRoC dataset, we adopt the open source tool EVO [33] to evaluate the performance of SD-VIS. By comparing the estimated value with the actual value, we calculate the absolute pose error (APE) as an index of the evaluation algorithm [34]. Table 1 shows the root mean square error (RMSE) of the translation on the EuRoC dataset. For fairness, the following algorithms do not use the loop closure detection module.

Table 1. The root mean square error (RMSE) results of the translation (m).

| Dataset      | VINS-Mono | VINS-Fusion | SD-VIS  |
|--------------|-----------|------------|---------|
| MH_01_easy   | 0.254651  | 0.364247   | 0.260793|
| MH_02_easy   | 0.263258  | 0.339122   | 0.289663|
| MH_03_medium | 0.547901  | 0.483257   | 0.577422|
| MH_04_medium | 0.590191  | 0.614950   | 0.497288|
| MH_05_difficult | 0.512011 | 0.524107   | 0.512458|
| V1_01_easy   | 0.217083  | 0.247467   | 0.245990|
| V1_02_medium | 0.492645  | 0.434756   | 0.502134|
| V1_03_difficult | 0.361321 | 0.345895   | 0.368939|
| V2_01_easy   | 0.170790  | 0.177467   | 0.202474|
| V2_02_medium | 0.424259  | 0.370081   | 0.454785|
| V2_03_difficult | 0.475561 | 0.521573   | 0.444759|
As can be seen from Table 1, in terms of accuracy, SD-VIS and VINS-Mono and VINS-Fusion are at the same level. The accuracy of SD-VIS is slightly lower than that of VINS-Mono when moving at low and medium speed in the environment with abundant feature points (such as MH_01_easy and MH_03_medium). This is due to the susceptibility to various illumination changes when tracking non-keyframes using the direct method. We have observed that some datasets exhibit strong exposure changes between images and, therefore, the tracking effect of the direct method is reduced in these cases. In addition, in order to further improve the real-time performance of the algorithm, we only extract new feature points in keyframes, which results in that the number of feature points in non-keyframes will be less than keyframes, which will also bring some negative effects. Although the accuracy performance is not significantly better than the traditional method, it has achieved the same level of accuracy as VINS-Mono while greatly improving real-time performance. Therefore, our algorithm is very suitable for small-sized unmanned platform with limited computing resources. The accuracy of SD-VIS is higher than that of VINS-Mono and VINS-Fusion when moving fast in the low-texture environment (such as V2_03_difficult). This is due to the excellent performance of the direct method in the low-texture environment. In addition, the keyframe selection strategy will tend to generate more keyframes during fast motion, which will also improve the accuracy performance of the algorithm. Figure 7 shows more intuitively the trajectory heat map estimated by SD-VIS, VINS-Mono, and VINS-Fusion in MH_01_easy. Figure 8 shows the change of translation absolute pose error with time in MH_01_easy, MH_04_medium, and V2_03_difficult. Through Figures 7 and 8, we came to the conclusion that the accuracy and robustness of our algorithm have reached the level of the state-of-the-art algorithm. Especially in the initialization procedure and low-texture environment, our algorithm performs better.

**Figure 7.** The trajectory heat map estimated by SD-VIS, VINS-Mono, and VINS-Fusion in MH_01_easy.
6.2. Real-Time Performance Evaluate

In this section, we evaluate the real-time performance of SD-VIS. We compared the average time required to track an image (Table 2).

Figure 8. The change of translation absolute pose error with time in MH_01_easy, MH_04_medium and V2_03_difficult. Blue, green, and red represent SD-VIS, VINS-Mono, and VINS-Fusion respectively.
Table 2. Average time (ms) spent tracking an image.

| Dataset        | ORB-SLAM2 | VINS-Mono | VINS-Fusion | SD-VIS |
|----------------|-----------|-----------|-------------|--------|
| MH_01_easy     | 37.82     | 17.67     | 38.91       | 6.72   |
| MH_02_easy     | 35.31     | 13.60     | 45.86       | 6.58   |
| MH_03_medium   | 34.20     | 12.28     | 44.15       | 7.87   |
| MH_04_medium   | 30.43     | 12.77     | 42.54       | 6.91   |
| MH_05_difficult| 28.46     | 13.50     | 45.40       | 10.96  |
| V1_01_easy     | 33.01     | 13.04     | 45.56       | 10.22  |
| V1_03_difficult| 30.41     | 13.04     | 45.56       | 10.22  |
| V2_01_easy     | 33.01     | 13.04     | 45.56       | 10.22  |
| V2_02_medium   | 30.41     | 13.04     | 45.56       | 10.22  |

As can be seen from Table 2, the image tracking of ORB-SLAM2 [11] uses the feature-based method to extract and match the ORB features of each frame, which takes a long time. However, VINS-Mono uses the optical flow method to track FAST features, which saves the calculation of feature descriptors, so the time consumption is less than ORB-SLAM2. VINS-Fusion is a stereo visual-inertial fusion SLAM algorithm, and image tracking also takes a long time. In SD-VIS, non-keyframes are used for fast-tracking and localization by direct method, and keyframes are tracked by feature-based method and used for back-end optimization and loop closure detection. This algorithm saves a lot of time and minimizes the average time of SD-VIS tracking images. Due to the keyframe selection strategy, in some low-speed motion scenes (such as MH_01_easy and V1_01_easy), the number of keyframes will be less, and the time to track a frame of the image will be reduced. In some fast-moving scenes (such as V1_02_medium and V2_03_difficult), as the number of keyframes increases, the time required to track an image will be increased.

In summary, the reason why we can obtain good real-time performance is due to the use of KLT sparse optical flow algorithm when tracking keyframes, which eliminates the calculation of descriptors and feature matching. In addition, for non-keyframes, only the direct method is used to track existing feature points, and new feature points are not extracted. Due to the close distance between two adjacent non-keyframes, the direct method of image alignment and feature matching can quickly converge.

Compared with the feature-based method, we use the direct method to track non-keyframes and accelerate the algorithm without reducing the accuracy and robustness. As shown in Figure 9, in MH_02_easy, 26% of the frames are determined to be keyframes, while 74% of the frames are determined to be non-keyframes. The time consumption of tracking keyframes is 65%, while that of non-keyframes are only 35%. Combined with Section 6.1, we can conclude that compared with the state-of-the-art SLAM systems, we can achieve a better balance between quickness and exactness.

Figure 9. The left picture (a) shows a comparison of the number of keyframes and non-keyframes. The right picture (b) shows the comparison of time consumption between tracking keyframes and non-keyframes.
6.3. Loop Closure Detection Evaluate

Finally, in order to verify the integrity and feasibility of the proposed algorithm, we evaluate the loop closure detection capability of SD-VIS. As can be seen from Figures 10 and 11, the accuracy of SD-VIS with loop detection is improved obviously. Compared with the direct method, SD-VIS exhibits the function of loop closure detection and solves the problem of drift in long-term operation.

![Figure 10. The trajectory heat map estimated by SD-VIS and SD-VIS-LOOP in V1_01_easy.](image1)

![Figure 11. The change of translation absolute pose error with time in V1_01_easy. Blue and green represent SD-VIS and SD-VIS-LOOP respectively.](image2)

7. Conclusions

We present SD-VIS, a novel fast and accurate semi-direct visual-inertial SLAM framework, which combines the exactness of feature-based method and quickness of direct method. Compared with the state-of-the-art feature-based method, we use the direct method to track non-keyframes and accelerate the algorithm without reducing the accuracy and robustness. Compared with the direct method, SD-VIS exhibits the function of loop closure detection and solves the problem of drift in long-term operation. We get a better balance between accuracy and speed, so the algorithm is more suitable for the platform with limited computing resources. In the future, we will extend the algorithm to support more types of multi-sensor fusion to increase its robustness in complex environments.
**Author Contributions:** Conceptualization, Q.L. and H.W.; methodology, Q.L.; software, Q.L. and H.W.; validation, Q.L., Z.W. and H.W.; formal analysis, Q.L.; investigation, Q.L.; resources, Q.L. and H.W.; data curation, Q.L.; writing—original draft preparation, Q.L.; writing—review and editing, Q.L. and Z.W.; visualization, Q.L.; supervision, Z.W.; funding acquisition, Z.W. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the National defense Innovation Fund.

**Acknowledgments:** We express our high respect and gratitude to the editors and reviewers, and their help and suggestions are very helpful to our manuscript.

**Conflicts of Interest:** The authors declare no conflict of interest.

**References**

1. Fuentes-Pacheco, J.; Ruiz-Ascencio, J.; Manuel Rendon-Mancha, J. Visual simultaneous localization and mapping: A survey. *Artif. Intell. Rev.* **2012**, *43*, 55–81. [CrossRef]

2. Bailey, T.; Durrant-Whyte, H. Simultaneous localization and mapping (SLAM): Part II. *IEEE Robot. Autom. Mag.* **2006**, *13*, 108–117. [CrossRef]

3. Wang, R.; Schworer, M.; Cremers, D. Stereo DSO: Large-scale direct sparse visual odometry with stereo cameras. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 3903–3911.

4. Wang, X.; Wang, J. Detecting glass in simultaneous localization and mapping. *Robot. Auton. Syst.* **2017**, *88*, 97–103. [CrossRef]

5. Titterton, D.; Weston, J.L. *Strapdown Inertial Navigation Technology*; The Institution of Engineering and Technology: Stevenage, UK, 2004; Volume 17.

6. Davison, A.J.; Reid, I.D.; Molton, N.D.; Stasse, O. MonoSLAM: Real-Time Single Camera SLAM. *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, *29*, 1052–1067. [CrossRef]

7. Lee, T.; Kim, C.; Cho, D.D. A Monocular Vision Sensor-Based Efficient SLAM Method for Indoor Service Robots. *IEEE Trans. Ind. Electron.* **2019**, *66*, 318–328. [CrossRef]

8. Shen, S.J.; Michael, N.; Kumar, V. Tightly-Coupled Monocular Visual-Inertial Fusion for Autonomous Flight of Rotorcraft MAVs. In Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA, 26–30 May 2015; pp. 5303–5310.

9. Leutenegger, S.; Lynen, S.; Bosse, M.; Siegwart, R.; Furgale, P. Keyframe-based visual-inertial odometry using nonlinear optimization. *Int. J. Robot. Res.* **2015**, *34*, 314–334. [CrossRef]

10. Mur-Artal, R.; Montiel, J.M.M.; Tardos, J.D. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Trans. Robot.* **2015**, *31*, 1147–1163. [CrossRef]

11. Engel, J.; Koltun, V.; Cremers, D. Direct sparse odometry. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 611–625. [CrossRef] [PubMed]

12. Usenko, V.; Engel, J.; Stuckler, J.; Cremers, D. Direct Visual-Inertial Odometry with Stereo Cameras. In Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016; pp. 1885–1892.

13. Falquez, J.M.; Kasper, M.; Sibley, G. Inertial Aided Dense & Semi-Dense Methods for Robust Direct Visual Odometry. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots & Systems, Daejeon, Korea, 9–14 October 2016; pp. 3601–3607.

14. Newcombe, R.A.; Lovegrove, S.J.; Davison, A.J. DTAM: Dense tracking and mapping in real-time. In Proceedings of the 2011 IEEE International Conference on Computer Vision (ICCV), Barcelona, Spain, 6–13 November 2011; pp. 2320–2327.

15. Lucas, B.D.; Kanade, T. An iterative image registration technique with an application to stereo vision. In Proceedings of the International Joint Conference on Artificial Intelligence, Vancouver, BC, Canada, August 1981; pp. 24–28.

16. Klein, G.; Murray, D. Parallel tracking and mapping for small AR workspaces (PTAM). In Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, Washington, DC, USA, 13–16 November 2007; pp. 1–10.

17. Mur-Artal, R.; Tardos, J.D. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Trans. Robot.* **2017**, *33*, 1255–1262. [CrossRef]
18. Qin, T.; Li, P.; Shen, S. VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator. *IEEE Trans. Robot.* 2018, 34, 1004–1020. [CrossRef]

19. He, Y.; Zhao, J.; Guo, Y.; He, W.; Yuan, K. Pi-VIO: Tightly coupled monocular visual inertial odometry using point and line features. *Sensors* 2018, 18, 1159. [CrossRef] [PubMed]

20. Qin, T.; Pan, J.; Cao, S.; Shen, S.J. A General Optimization-based Framework for Local Odometry Estimation with Multiple Sensors. *arXiv Preprint* 2019, arXiv:1901.03638.

21. Engel, J.; Schöps, T.; Cremers, D. LSD-SLAM: Large-scale direct monocular SLAM. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; pp. 834–849.

22. Gao, X.; Wang, R.; Demmel, N.; Cremers, D. LDSO: Direct Sparse Odometry with Loop Closure. In Proceedings of the International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018.

23. Forster, C.; Pizzoli, M.; Scaramuzza, D. SVO: Fast semi-direct monocular visual odometry. In Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–5 June 2014; pp. 15–22.

24. Lee, S.H.; Civera, J. Loosely-coupled semi-direct monocular slam. *arXiv Preprint* 2018, arXiv:1807.10073. [CrossRef]

25. Krombach, N.; Droeschel, D.; Houben, S.; Behnke, S. Feature based visual odometry prior for real-time semi-dense stereo slam. *Robot. Auton. Syst.* 2018, 109, 38–58. [CrossRef]

26. Kim, O.; Lee, H.; Kim, H.J. Autonomous flight with robust visual odometry under dynamic lighting conditions. *Auton. Robot.* 2019, 43, 1605–1622. [CrossRef]

27. Li, S.P.; Zhang, T.; Gao, X.; Wang, D.; Xian, Y. Semi-direct monocular visual and visual-inertial SLAM with loop closure detection. *Robot. Auton. Syst.* 2019, 112, 201–202. [CrossRef]

28. Shi, J. Good features to track. In Proceedings of the 1994 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’94), Seattle, WA, USA, 21–23 June 1994; pp. 593–600.

29. Hartley, R.; Zisserman, A. *Multiple View Geometry in Computer Vision*; Cambridge University Press: Cambridge, UK, 2003.

30. Baker, S.; Matthews, I. Lucas-Kanade 20 Years On: A Unifying Framework: Part 1. *Int. J. Comput. Vis.* 2002, 56, 221–255. [CrossRef]

31. Sibley, G.; Matthies, L.; Sukhatme, G. Sliding window filter with application to planetary landing. *J. Field Robot.* 2010, 27, 587–608. [CrossRef]

32. Burri, M.; Nikolic, J.; Gohl, P.; Schneider, T.; Rehder, J.; Omari, S.; Achtelik, M.W.; Siegwart, R. The EuRoC micro aerial vehicle datasets. *Int. J. Robot. Res.* 2016, 35, 1157–1163. [CrossRef]

33. Michael Grupp. EVO. Available online: https://github.com/MichaelGrupp/evo (accessed on 8 August 2019).

34. Sturm, J.; Engelhard, N.; Endres, F.; Burgard, W.; Cremers, D. A benchmark for the evaluation of RGB-D SLAM systems. In Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vilamoura, Portugal, 7–12 October 2012; pp. 573–580.

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).