Wuhan University
Journal of Natural Sciences

2023, Vol.28 No.1, 020-028

Article ID 1007-1202(2023)01-0020-09
DOI https://doi.org/10.1051/wujns/2023281020

RRVPE: A Robust and Real-Time Visual-Inertial-GNSS Pose Estimator for Aerial Robot Navigation

□ ZHANG Chi[1], YANG Zhong[1†], XU Hao[1,2], LIAO Luwei[1], ZHU Tang[1], LI Guotao[1], YANG Xin[1], ZHANG Qiuyan[3]

1. College of Automation Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, Jiangsu, China;
2. School of Mathematics & Physics, Anhui University of Technology, Maanshan 243000, Anhui, China;
3. Electric Power Research Institute of Guizhou Power Grid Co., Ltd., Guiyang 550002, Guizhou, China

Abstract: Self-localization and orientation estimation are the essential capabilities for mobile robot navigation. In this article, a robust and real-time visual-inertial-GNSS(Global Navigation Satellite System) tightly coupled pose estimation (RRVPE) method for aerial robot navigation is presented. The aerial robot carries a front-facing stereo camera for self-localization and an RGB-D camera to generate 3D voxel map. Ulteriorly, a GNSS receiver is used to continuously provide pseudorange, Doppler frequency shift and universal time coordinated (UTC) pulse signals to the pose estimator. The proposed system leverages the Kanade Lucas algorithm to track Shi-Tomasi features in each video frame, and the local factor graph solution process is bounded in a circumscribed container, which can immensely abandon the computational complexity in nonlinear optimization procedure. The proposed robot pose estimator can achieve camera-rate (30 Hz) performance on the aerial robot companion computer. We thoroughly experimented the RRVPE system in both simulated and practical circumstances, and the results demonstrate dramatic advantages over the state-of-the-art robot pose estimators.

Key words: computer vision; visual-inertial-GNSS(Global Navigation Satellite System) pose estimation; real-time autonomous navigation; sensor fusion; robotics

CLC number: TP 391

0 Introduction

Aerial robots will soon play a significant role in industrial inspection, accident warning, and national defense[1-3]. For such operations, flight mode dependent on the human remote control can no longer meet the mission requirements under complex conditions. At present, the autonomous navigation ability has become an important indicator to measure robot intelligent level. It is difficult to obtain the aerial robot pose in real time and solve the problem of autonomous robot navigation. Fully autonomous navigation requires aerial robots to have accurate pose estimation and robust environmental awareness. Due to the aerial robot's swing during flying, the state estimation results are difficult to converge during movement, which leads to the instability of the exist-

Received date: 2022-07-25
Foundation item: Supported by the Guizhou Provincial Science and Technology Projects ([2020]2Y044), the Science and Technology Projects of China Southern Power Grid Co. Ltd. (066600KK52170074), and the National Natural Science Foundation of China (61473144)
Biography: ZHANG Chi, male, Ph.D. candidate, research direction: robot navigation. E-mail: laozhang@nuaa.edu.cn
† To whom correspondence should be addressed. E-mail: YangZhong@nuaa.edu.cn

ing pose estimation algorithms.

Compared with the aerial robot pose estimator based on a single sensor, the multi-sensor fusion pose estimation technologies[4-8] can make full use of different kinds of sensors to obtain more accurate and robust robot pose estimation results. Stereo camera and inertial measurement unit (IMU) can output the robot position with centimeter-level precision in the local coordinate system[9, 10], but the pose in the local frame will drift with the aerial robot motion. Global navigation satellite system (GNSS) has been widely used in various mobile robot navigation tasks to provide drift-free position information for agents[11,12]. However, the GNSS-based agent navigation method is not suitable to use indoors, and is vulnerable to noise and multipath effects, resulting in only meter-level positioning accuracy. In order to leverage the complementary characteristics of different sensors, the pose estimation algorithm based on vision-IMU-GNSS information fusion can make full use of the respective advantages of stereo cameras, accelerometers, gyroscopes and navigation satellites to obtain accurate and drift free aerial robot pose estimation. Unfortunately, the pose estimation strategy based on vision-IMU-GNSS sensor fusion will face the following problems: Firstly, the output frequencies from each sensor are different (the frequencies of camera, IMU and GNSS receiver are 30, 200, and 10 Hz, respectively). How to fuse these raw measurements from different sensors will be an intractable problem[13,14]; Secondly, how can the pose estimator quickly restore to normal state when one of the sensors suddenly fails?

To deal with the aforementioned problems, we propose a robust and real-time visual-inertial-GNSS tightly coupled pose estimation (RRVPE) method, which is a probabilistic factor graph optimization-based pose estimation strategy, for aerial robot navigation. The RRVPE system can achieve real-time robot state estimation after compute unified device architecture (CUDA) acceleration on an airborne computer. The main novelties of RRVPE are exhibited as below:

1) The RRVPE system leverages the Kanade-Lucas[15] algorithm to track Shi-Tomasi[16] features in each video frame, and the image corners describing and matching between adjacent frames are not required in corner tracking procedures. After NVIDIA CUDA acceleration, the robot pose estimator can achieve camera-rate (30 Hz) performance on a small embedded platform.

2) The local bundle adjustment (BA) and factor graph solution process are bounded in a circumscribed container, which can dramatically abandon the number of variables in nonlinear optimization procedure. Furthermore, the computation complexity of the RRVPE system is discharged by the additional marginalization strategy.

3) By making full use of the GNSS raw measurements, the intrinsic drift from the vision-IMU odometry will be abandoned, and the yaw angle residual between the odometry frame and the world frame will be updated without any offline calibration. The aerial robot pose estimator is able to rapidly execute in unpredictable environments and achieves local smoothness and global consistency without visual closed loop detection.

1 System Overview

The aerial robot equipped with the RRVPE navigation system is shown in Fig. 1, which tightly fuses sparse optical flow tracking and inertial measurements with GNSS raw data for precise and driftless aerial robot pose estimation.

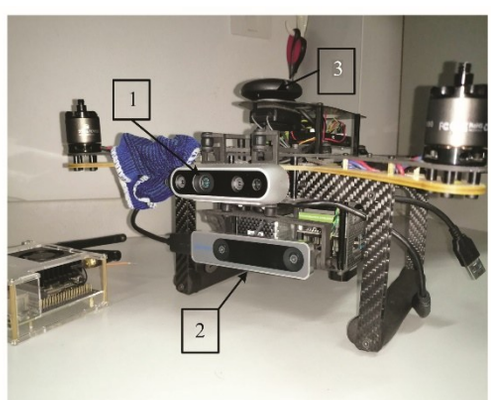

Fig. 1 The aerial robot equipped with the RRVPE navigation system

1. Intel RealSense D435i camera: responsible for building real-world 3D voxel map; 2. Intel RealSense T265 camera: responsible for providing binocular video stream and inertial measurement information; 3. U-Blox ZED-F9P receiver: responsible for receiving GNSS pseudorange, Doppler, ephemeris and time pulse information

The structure of the RRVPE system overview is illustrated in Fig. 2. First, the raw sensor data from the aerial robot are preprocessed, including visual feature extraction and tracking, IMU pre-integration, and GNSS signal filtering. Then, vision, IMU and GNSS cost functions are formulated respectively, and vision-IMU-

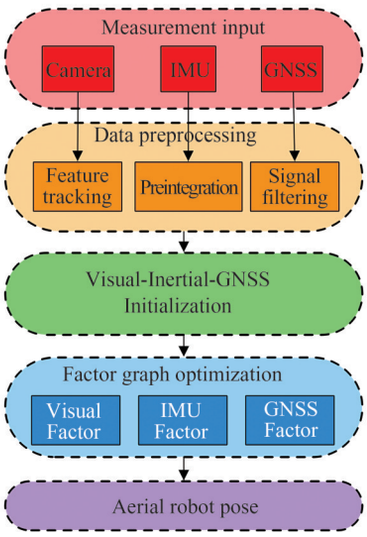

Fig. 2 Main parallel threads of RRVPE system

GNSS information is jointly initialized to obtain all the initial values of the aerial robot pose estimator. Finally, the aerial robot pose solving process is constructed as a state estimation problem. In the meantime, the corresponding probability factor graph model and marginalization strategy are designed. The aerial robot pose is solved by non-linear optimization, and subsequently the accurate, robust and drift free robot pose can be achieved.

In the system initialization stage, the camera trajectory is solved by the structure from motion (SfM) algorithm[17-19], and the IMU raw measurement data is pre-integrated[20, 21] to initialize the vision-IMU tightly coupled robot odometry. Then, the rough robot position in the world coordinate frame is solved by the single point positioning (SPP) algorithm. Under the condition that visual-IMU odometry is used as prior information, the transformation matrix between the odometry coordinate frame and the world coordinate frame is solved in nonlinear optimization. Finally, the precise pose of the aerial robot in the global coordinate system is modified by probability factor graph optimization.

After the estimator initialization, constraints from all sensor measurements are tightly coupled to calculate aerial robot states within a circumscribed container. If the GNSS broadcast is not available or cannot be entirely initialized, the RRVPE system will naturally degrade to visual-IMU odometry. In order to maintain the real-time performance of the estimation system, the additional marginalization scheme[6] is also applied after each optimization.

We define $(\cdot)^{\mathrm{r}}$ as the robot coordinate system, $(\cdot)^{\mathrm{c}}$ as the camera coordinate system and $(\cdot)^{\mathrm{o}}$ as the odometry frame, where the direction of the gravity is aligned with the Z axis. World coordinate system $(\cdot)^{\mathrm{w}}$ is a semiglobal frame, where the X and Y axes direct to the east and north direction respectively, and the Z axis is also gravity aligned. The earth-centered, earth-fixed (ECEF) frame $(\cdot)^{\mathrm{e}}$ and the earth-centered inertial (ECI) frame $(\cdot)^{\mathrm{E}}$ are global coordinate system that is fixed with respect to the center of the earth. The difference between the ECEF and the ECI frame is that the latter's coordinate axis does not change with the rotation of the earth.

2 Aerial Robot Pose Estimator

2.1 Formulation

In this section, the aerial robot pose estimation is formulated as a probabilistic factor graph optimization procedure, and sensor measurement information constitutes a composite of multifarious factors in the graph, which constrains the aerial robot states. The factors in the probabilistic graph are composed of visual factor, inertia factor and GNSS factor. All of the factors in the factor graph will be formulated in detail through this section.

We can take advantage of a sliding window-based tightly coupled visual-inertial-GNSS pose estimator for exceedingly robust and real-time aerial robot state estimation. The whole states $\boldsymbol{\chi}$ inside the sliding window can be summarized as:

$$\begin{cases} \boldsymbol{\chi} = \left[x_0, x_1, \cdots, x_n, \lambda_1, \lambda_2, \cdots, \lambda_m, \psi \right]^{\mathrm{T}} \\ \boldsymbol{x}_k = \left[o^{\mathrm{w}}_{\mathrm{r}_{t_k}}, v^{\mathrm{w}}_{\mathrm{r}_{t_k}}, p^{\mathrm{w}}_{\mathrm{r}_{t_k}}, b_{\omega_{t_k}}, b_{a_{t_k}}, \delta t, \delta t' \right]^{\mathrm{T}}, \ k \in [0, n] \end{cases} \tag{1}$$

where $\boldsymbol{x}_k$ is the aerial robot state at the time t_k that the k-th video frame is captured. It contains orientation $o^{\mathrm{w}}_{\mathrm{r}_{t_k}}$, velocity $v^{\mathrm{w}}_{\mathrm{r}_{t_k}}$, position $p^{\mathrm{w}}_{\mathrm{r}_{t_k}}$, gyroscope bias $b_{\omega_{t_k}}$ and acceleration bias $b_{a_{t_k}}$ of the aerial robot in the odometry frame. δt and $\delta t'$ correspond to the clock biases and bias drifting rate of the GNSS receiver, respectively. n is the window size and m is the total number of visual features in the sliding window. λ_l is the inverse depth of the l-th visual feature. ψ is the yaw bias between the odometry and the world frame.

2.2 Visual Constraint

The visual factor constraint in the probabilistic graph is constructed from a sequence of sparse corner

points. Considering the unstable vibration of the aerial robot, we separate the Shi-Tomasi[16] sparse feature points for the Kanade-Lucas-Tomasi (KLT) optical flow tracking[15].

For each input video frame, when the number of feature points is less than 120, new corner points are extracted to maintain a sufficient number of tracking features. Meanwhile, a uniform feature point distribution is carried out by setting a minimum pixel space between neighboring corners. It is worth noting that the corner extraction and KLT optical flow tracking procedures can achieve camera-rate performance on the NVIDIA Jetson Xavier NX board after being accelerated by CUDA. Assume the homogeneous coordinates of the image feature point l in the world coordinate frame are:

$$\tilde{\boldsymbol{p}}_l^{\mathrm{w}} = \left[X_l \diagup Z_l ,\ Y_l \diagup Z_l ,\ 1 \right]^{\mathrm{T}} \tag{2}$$

Then the homogeneous coordinates of feature point l in the pixel plane of video frame i can be expressed as:

$$\tilde{\boldsymbol{P}}_l^{\mathrm{c}_i} = \left[u_l^i ,\ v_l^i ,\ 1 \right]^{\mathrm{T}} \tag{3}$$

where u and v are coordinate values on the pixel plane. The projection model of the airborne camera can be expressed as:

$$\tilde{\boldsymbol{P}}_l^{\mathrm{c}_i} = \boldsymbol{K}\boldsymbol{T}_{\mathrm{r}}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{w}}^{\mathrm{r}_i}\tilde{\boldsymbol{p}}_l^{\mathrm{w}} + n_{\mathrm{c}} \tag{4}$$

where $\boldsymbol{T}$ is the transformation matrix, n_{c} is the camera imaging noise, and $\boldsymbol{K}$ is the camera internal parameter matrix:

$$\boldsymbol{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

where f and c represent scaling and translation during camera projection, respectively. The elements of the internal parameter matrix can be obtained by the camera calibration process, and the reprojection model of the feature point l from the video frame i to the video frame j can be formulated as:

$$\hat{\tilde{\boldsymbol{P}}}_l^{\mathrm{c}_j} = \boldsymbol{K}\boldsymbol{T}_{\mathrm{r}}^{\mathrm{c}}\boldsymbol{T}_{\mathrm{w}}^{\mathrm{r}_j}\left[\boldsymbol{T}_{\mathrm{r}_i}^{\mathrm{w}}\boldsymbol{T}_{\mathrm{c}}^{\mathrm{r}}\boldsymbol{K}^{-1}\left(Z_l^{\mathrm{c}_i}\tilde{\boldsymbol{P}}_l^{\mathrm{c}_i} \right) \right] \tag{6}$$

with

$$Z_l^{\mathrm{c}_i} = \lambda_l^{\mathrm{c}_i}\frac{f_x f_y}{\sqrt{{f_x}^2{f_y}^2 + \left(u_l^i - c_x\right)^2{f_y}^2 + \left(v_l^i - c_y\right)^2{f_x}^2}} \tag{7}$$

where $\lambda_l^{\mathrm{c}_i}$ represents the inverse depth of feature point l relative to the airborne camera c_i.

Then the visual factor constraint can be expressed as the deviation between the actual position $\tilde{\boldsymbol{P}}_l^{\mathrm{c}_j}$ of the image feature point l in the video frame j and the measurement position $\hat{\tilde{\boldsymbol{P}}}_l^{\mathrm{c}_j}$:

$$\boldsymbol{E}_{\mathrm{V}}\left(\hat{Z}_l^{\mathrm{c}_j}, \boldsymbol{\chi}_{\mathrm{V}} \right) = \tilde{\boldsymbol{P}}_1^{\mathrm{c}_j} - \hat{\tilde{\boldsymbol{P}}}_l^{\mathrm{c}_j} \tag{8}$$

where $\boldsymbol{\chi}_{\mathrm{V}}$ represents the sub-vector related to visual information in the aerial robot state vector.

2.3 Inertial Measurements Constraint

In the world coordinate frame, the aerial robot's pose and velocity can be obtained by the raw data of the inertial measurement unit that are measured in the aerial robot body frame. The IMU raw data includes two parts: gyroscope measurement $\hat{\omega}^{\mathrm{r}_t}$ and accelerometer measurement $\hat{a}^{\mathrm{r}_t}$, both of which are affected by the gyroscope bias b_ω and the acceleration bias b_a, respectively. The raw measurement values of the gyroscope and accelerometer can be constructed by the following formulas:

$$\begin{cases} \hat{\omega}^{\mathrm{r}_t} = \omega^{\mathrm{r}_t} + b_{\omega^{\mathrm{r}_t}} + n_{\omega^{\mathrm{r}_t}} \\ \hat{a}^{\mathrm{r}_t} = a^{\mathrm{r}_t} + b_{a^{\mathrm{r}_t}} + n_{a^{\mathrm{r}_t}} + R_{\mathrm{w}}^{\mathrm{r}_t} g^{\mathrm{w}} \end{cases} \tag{9}$$

where, symbols $\hat{\omega}^{\mathrm{r}_t}$ and $\hat{a}^{\mathrm{r}_t}$ represent the measured values of the gyroscope and accelerometer at time t with the current body coordinate system as the reference system respectively; b_ω and b_a are the gyroscope bias and accelerometer bias; n_ω and n_a are gyroscope noise and accelerometer noise; g^{w} is the gravitational acceleration. The gyroscope and accelerometer noises are Gaussian white noise; the gyroscope biases and the accelerometer biases obey Brownian motion, and their derivatives obey Gaussian distribution.

Assuming that the motion time of the aerial robot in two consecutive video frames is t_k and t_{k+1}, then the orientation (o), velocity (v) and position (p) of the aerial robot at time t+1 in the local world coordinate system can be expressed by the following formula:

$$\begin{cases} o^{\mathrm{w}_{t_{k+1}}} = o^{\mathrm{w}_{t_k}} \otimes \displaystyle\int_{t\in[t_k,t_{k+1}]} \frac{1}{2}\boldsymbol{\Phi}(\hat{\omega}^{\mathrm{r}_t} - b_{\omega^{\mathrm{r}_t}} - n_{\omega^{\mathrm{r}_t}}) o_{\mathrm{r}_t}^{\mathrm{r}_{t_k}} \mathrm{d}t \\ v^{\mathrm{w}_{t_{k+1}}} = v^{\mathrm{w}_{t_k}} + \displaystyle\int_{t\in[t_k,t_{k+1}]} \left[R_{\mathrm{r}_t}^{\mathrm{w}_t}(\hat{a}^{\mathrm{r}_t} - b_{a^{\mathrm{r}_t}} - n_{a^{\mathrm{r}_t}}) - g^{\mathrm{w}_t} \right] \mathrm{d}t \\ p^{\mathrm{w}_{t_{k+1}}} = p^{\mathrm{w}_{t_k}} + (t_{k+1} - t_k)v^{\mathrm{w}_{t_k}} + \displaystyle\iint_{t\in[t_k,t_{k+1}]} \left[R_{\mathrm{r}_t}^{\mathrm{w}_t}(\hat{a}^{\mathrm{r}_t} - b_{a^{\mathrm{r}_t}} - n_{a^{\mathrm{r}_t}}) - g^{\mathrm{w}_t} \right] \mathrm{d}t^2 \end{cases} \tag{10}$$

where

$$\boldsymbol{\Phi}(\boldsymbol{\omega})=\begin{bmatrix}-\lfloor\boldsymbol{\omega}\rfloor_\times & \boldsymbol{\omega}\\ -\boldsymbol{\omega}^{\mathrm{T}} & 0\end{bmatrix},\lfloor\boldsymbol{\omega}\rfloor_\times=\begin{bmatrix}0 & -\omega_z & \omega_y\\ \omega_z & 0 & -\omega_x\\ -\omega_y & \omega_x & 0\end{bmatrix} \tag{11}$$

In the above formula, symbols $\hat{\omega}$ and $\hat{a}$ are the measured values from gyroscope and accelerometer, and the symbols $\otimes$ represent quaternion multiplications.

If the reference coordinate system is converted from the local world coordinate system (w) to the robot coordinate system (r), the above formula can be rewritten as:

$$\begin{cases} o_{\mathrm{w}}^{\mathrm{r}_{t_k}}\otimes o^{\mathrm{w}_{t_{k+1}}}=\alpha_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}}\\ R_{\mathrm{w}}^{\mathrm{r}_{t_k}}v^{\mathrm{w}_{t_{k+1}}}=R_{\mathrm{w}}^{\mathrm{r}_{t_k}}[v^{\mathrm{w}_{t_k}}-(t_{k+1}-t_k)g^{\mathrm{w}}]+\beta_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}}\\ R_{\mathrm{w}}^{\mathrm{r}_{t_k}}p^{\mathrm{w}_{t_{k+1}}}=R_{\mathrm{w}}^{\mathrm{r}_{t_k}}[p^{\mathrm{w}_{t_k}}+(t_{k+1}-t_k)v^{\mathrm{w}_{t_k}}-\frac{1}{2}g^{\mathrm{w}}(t_{k+1}-t_k)^2]+\gamma_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}}\end{cases} \tag{12}$$

where the IMU pre-integration term can be expressed as:

$$\begin{cases}\alpha_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}}=\alpha_{\mathrm{r}_t}^{\mathrm{r}_{t_k}}\int_{t\in[t_k,t_{k+1}]}\frac{1}{2}\boldsymbol{\Phi}(\hat{\omega}^{\mathrm{r}_t}-b_{\omega^{\mathrm{r}_t}}-n_{\omega^{\mathrm{r}_t}})\mathrm{d}t\\ \beta_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}}=\int_{t\in[t_k,t_{k+1}]}R_{\mathrm{r}_t}^{\mathrm{r}_{t_k}}(\hat{a}^{\mathrm{r}_t}-b_{a^{\mathrm{r}_t}}-n_{a^{\mathrm{r}_t}})\mathrm{d}t\\ \gamma_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}}=\iint_{t\in[t_k,t_{k+1}]}R_{\mathrm{r}_t}^{\mathrm{r}_{t_k}}(\hat{a}^{\mathrm{r}_t}-b_{a^{\mathrm{r}_t}}-n_{a^{\mathrm{r}_t}})\mathrm{d}t^2\end{cases} \tag{13}$$

Then the first-order Jacobian approximation of the IMU pre-integration term can be expressed by the following formula:

$$\begin{cases}\alpha_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}}\approx\hat{\alpha}_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}}\otimes\begin{bmatrix}1\\ \frac{1}{2}J_{b_\omega}^{a}\Delta b\omega_{t_k}\end{bmatrix}\\ \beta_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}}\approx\hat{\beta}_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}}+J_{b_a}^{\beta}\Delta ba_{t_k}+J_{b_\omega}^{\beta}\Delta b\omega_{t_k}\\ \gamma_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}}\approx\hat{\gamma}_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}}+J_{b_a}^{\gamma}\Delta ba_{t_k}+J_{b_\omega}^{\gamma}\Delta b\omega_{t_k}\end{cases} \tag{14}$$

This formula represents a sub-matrix of the Jacobian matrix. When the gyroscope or accelerometer bias changes, the above first-order Jacobian approximation can be used to replace the IMU pre-integration without reintegration.

The gyroscope factor constraint term is constructed as a rotation residual based on quaternion outer product. Simultaneously, the accelerometer factor constraint term is constructed as velocity pre-integration residual and translation pre-integration residual respectively. The gyroscope bias and accelerometer bias factor terms are obtained from the bias difference between two consecutive video frames. Then the IMU factor constraint can be constructed as follows:

$$\boldsymbol{E}_{\mathrm{I}}\left(\hat{Z}_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}},\boldsymbol{\chi}_{\mathrm{I}}\right)$$

$$=\left[\alpha_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}}\otimes\left(\hat{\alpha}_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}}\right)^{-1},\beta_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}}-\hat{\beta}_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}},\gamma_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}}-\hat{\gamma}_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}},\delta b_\omega,\delta b_a\right]^{\mathrm{T}}$$

$$=\begin{bmatrix}2\left[o_{\mathrm{w}}^{\mathrm{r}_{t_k}}\otimes o^{\mathrm{w}_{t_{k+1}}}\otimes\left(\hat{\alpha}_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}}\right)^{-1}\right]_{\mathrm{imag}}\\ R_{\mathrm{w}}^{\mathrm{r}_{t_k}}\left[v^{\mathrm{w}_{t_{k+1}}}-v^{\mathrm{w}_{t_k}}+(t_{k+1}-t_k)g^{\mathrm{w}}\right]-\hat{\beta}_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}}\\ R_{\mathrm{w}}^{\mathrm{r}_{t_k}}\left[p^{\mathrm{w}_{t_{k+1}}}-p^{\mathrm{w}_{t_k}}-(t_{k+1}-t_k)v^{\mathrm{w}_{t_k}}+\frac{1}{2}g^{\mathrm{w}}(t_{k+1}-t_k)^2\right]-\hat{\gamma}_{\mathrm{r}_{t_{k+1}}}^{\mathrm{r}_{t_k}}\\ b_{\omega^{\mathrm{r}_{t+1}}}-b_{\omega^{\mathrm{r}_t}}\\ b_{a^{\mathrm{r}_{t+1}}}-b_{a^{\mathrm{r}_t}}\end{bmatrix} \tag{15}$$

where $\boldsymbol{\chi}_{\mathrm{I}}$ represents the sub-vector related to IMU in the aerial robot state vector.

2.4 GNSS Constraint

Currently, there are 4 complete and independently operated GNSS constellations, namely, BeiDou, GPS, Galileo, GLONASS. The navigation satellites in each GNSS constellation ceaselessly broadcast modulated carrier signals, and consequently the ground receiver can distinguish the navigation satellites and demodulate the original messages. The GNSS factor constraint in the probability factor graph model is composed of pseudorange factor, Doppler frequency shift factor and receiver clock offset factor. The pseudorange measurement model between the receiver and the navigation satellite can be expressed as:

$$\hat{P}_{\mathrm{r}}^{\mathrm{s}}=\left\|p_{\mathrm{r}}^{\mathrm{E}}-p_{\mathrm{s}}^{\mathrm{E}}\right\|+c\left(\delta t_{\mathrm{r}}+\delta t_{\mathrm{s}}+\Delta t_{\mathrm{tro}}+\Delta t_{\mathrm{ion}}+\Delta t_{\mathrm{mul}}\right)+n_{\mathrm{pr}} \tag{16}$$

with

$$\begin{cases}p_{\mathrm{r}_{t_k}}^{\mathrm{E}}=R(\omega_{\mathrm{earth}}t_{\mathrm{r}}^{\mathrm{s}})p_{\mathrm{r}_{t_k}}^{\mathrm{e}}\\ p_{\mathrm{s}_{t_i}}^{\mathrm{E}}=R(\omega_{\mathrm{earth}}t_{\mathrm{r}}^{\mathrm{s}})p_{\mathrm{s}_{t_i}}^{\mathrm{e}}\end{cases} \tag{17}$$

Here, $p_{\mathrm{r}}^{\mathrm{E}}$ and $p_{\mathrm{s}}^{\mathrm{E}}$ are the positions of the ground receiver and navigation satellite in the Earth-centered inertial (ECI) coordinate system respectively. $\hat{P}_{\mathrm{r}}^{\mathrm{s}}$ is the measured value of GNSS pseudorange, c is the propagation speed of light in vacuum, δt_{r} and δt_{s} are the clock offset of the receiver and satellite, respectively, Δt_{tro} and Δt_{ion} are the delay of troposphere and ionosphere in the atmosphere, respectively, Δt_{mul} is the delay caused by multipath effect, n_{pr} is the noise of pseudo range signal, ω_{earth} is the earth's rotation speed, $t_{\mathrm{r}}^{\mathrm{s}}$ represents the signal propagation time from the satellite to the receiver.

Then the GNSS pseudorange factor constraint can be constructed as the residual between the true pseudorange and the receiver measured pseudorange:

$$\boldsymbol{E}_{\mathrm{pr}}\left(\hat{Z}_{\mathrm{r}_{t_k}}^{\mathrm{s}_i},\boldsymbol{\chi}_{\mathrm{pr}}\right) = \left\| p_{\mathrm{r}_{t_k}}^{\mathrm{E}} - p_{\mathrm{s}_i}^{\mathrm{E}} \right\| + c\left(\delta t_{\mathrm{r}_{t_k}} + \delta t_{\mathrm{s}_i} + \Delta t_{\mathrm{tro}} + \Delta t_{\mathrm{ion}} + \Delta t_{\mathrm{mul}}\right) - \hat{P}_{\mathrm{r}_{t_k}}^{\mathrm{s}_i} \quad (18)$$

where $\boldsymbol{\chi}_{\mathrm{pr}}$ represents the sub-vector related to the GNSS pseudorange in the aerial robot state vector.

Besides pseudorange, Doppler frequency shift is also an important navigation information in GNSS modulated signal. The Doppler frequency shift measurement of GNSS receiver and navigation satellite can be modeled as:

$$\delta\hat{f}_{\mathrm{r}}^{\mathrm{s}} = -\frac{1}{\lambda}\left[\Im_{\mathrm{r}}^{s}\left(v_{\mathrm{r}}^{\mathrm{E}} - v_{\mathrm{s}}^{\mathrm{E}}\right) + c\left(\delta t_{\mathrm{r}}' + \delta t_{\mathrm{s}}'\right)\right] + n_{\mathrm{dp}} \quad (19)$$

with

$$\begin{cases} v_{\mathrm{r}_{t_k}}^{\mathrm{E}} = R(\omega_{\mathrm{earth}} t_{\mathrm{r}}^{s}) v_{\mathrm{r}_{t_k}}^{\mathrm{e}} \\ v_{\mathrm{s}_i}^{\mathrm{E}} = R(\omega_{\mathrm{earth}} t_{\mathrm{r}}^{s}) v_{\mathrm{s}_i}^{\mathrm{e}} \end{cases} \quad (20)$$

where λ is the carrier wavelength, $\Im_r^s$ is the direction vector between the satellite and the receiver, $v_{\mathrm{r}_{t_k}}^{\mathrm{E}}$ and $v_{\mathrm{s}_i}^{\mathrm{E}}$ are the speed of the receiver and the satellite respectively, and $\delta t_{\mathrm{r}_{t_k}}'$ and $\delta t_{\mathrm{s}_i}'$ are the clock offset drifting rate of the receiver and the satellite respectively.

Then the GNSS Doppler shift factor constraint can be constructed as the residual between the true carrier Doppler shift and the Doppler shift measurement:

$$E_{\mathrm{dp}}\left(\hat{z}_{\mathrm{r}_{t_k}}^{\mathrm{s}_i},\boldsymbol{\chi}_{\mathrm{dp}}\right) = -\frac{1}{\lambda}\left[\Im_{\mathrm{r}_{t_k}}^{\mathrm{s}_i}\left(v_{\mathrm{r}_{t_k}}^{\mathrm{E}} - v_{\mathrm{s}_i}^{\mathrm{E}}\right) + c\left(\delta t_{\mathrm{r}_{t_k}}' + \delta t_{\mathrm{s}_i}'\right)\right] - \delta\hat{f}_{\mathrm{r}_{t_k}}^{\mathrm{s}_i} \quad (21)$$

where, $\boldsymbol{\chi}_{\mathrm{dp}}$ represents the sub-vector related to GNSS Doppler frequency shift in the agent state vector $\boldsymbol{\chi}$, and $\delta\hat{f}_{\mathrm{r}_{t_k}}^{\mathrm{s}_i}$ is the Doppler frequency shift measurement value.

Now, the GNSS receiver clock offset error from t_k to t_{k+1} is constructed as follows:

$$E_{\tau}\left(\hat{Z}_{k-1}^{k},\boldsymbol{\chi}_{\tau}\right) = \delta t_{\mathrm{r}_{t_k}} - \delta t_{\mathrm{r}_{t_{k-1}}} - \left(t_k - t_{k-1}\right)\delta t_{\mathrm{r}_{t_{k-1}}}' \quad (22)$$

By combining the pseudorange factor E_{pr}, the Doppler frequency shift factor E_{dp} and the receiver clock offset factor E_{τ}, the GNSS factor constraint item in the aerial robot probability factor graph model can be formed.

2.5 Tightly Coupled Pose Estimation

Considering the aerial robot pose solving process as a state estimation problem, the optimal state of the aerial robot is the maximum a posteriori estimation of the robot state vector. Assuming that the measurement signals of the aerial robot's airborne camera, IMU, and GNSS receiver are independent of each other, and the measurement noise conforms to a Gaussian distribution with zero mean, the maximum a posteriori estimation problem can be equivalent to minimizing the sum of errors, then the solution process of the aerial robot's state vector $\boldsymbol{\chi}$ can be expressed as:

$$\begin{aligned} \boldsymbol{\chi} &= \arg\max_{\chi} P\left(\boldsymbol{\chi}|z\right) \\ &= \arg\min_{\chi}\left(\left\| e_{\mathrm{p}} - \boldsymbol{H}_{\mathrm{p}}\boldsymbol{\chi} \right\|^2 + \sum_{k=1}^{n}\left\| E\left(z_k,\boldsymbol{\chi}\right)\right\|^2\right) \end{aligned} \quad (23)$$

where, z is the aerial robot linear observation model, e_{p} represents the prior error, $\boldsymbol{H}_{\mathrm{p}}$ matrix is the prior pose information obtained by the airborne camera, n is the number of robot state vectors in the sliding window, and $E(\cdot)$ represents the sum of all sensor measurement error factors.

Finally, by solving the aerial robot state vector χ by means of probability factor graph optimization, the complete robot pose information can be obtained.

3 Experiments

3.1 Implementation for Aerial Robot Navigation

We chose the NVIDIA jetson Xavier NX board as the companion computer for aerial robot autonomous navigation. The Intel RealSense T265 binocular camera is employed to provide visual and inertial raw measurement information for the aerial robot pose estimation. Simultaneously, the Intel Realsense D435i RGB-D camera can provide 3D point cloud map. The U-Blox ZED-F9P is used for GNSS receiver module, which can continuously provide GNSS pseudorange, Doppler frequency shift and universal time coordinated (UTC) pulse signals to aerial robot pose estimator. A carbon fiber quadrotor unmanned aerial vehicle (UAV) with 410 mm wheelbase can be used as the carrier of companion computer and sensors. Pixhawk4 is chosen as the flight control automatic pilot, and the PX4 is employed as the flight control firmware. Both onboard cameras and GNSS receiver are connected to the companion computer via USB. The ground station is connected with the Pixhawk4 automatic pilot and companion computer through WiFi 2.4G and Ethernet, respectively. The detailed description is shown in Fig. 3.

3.2 Simulation in Public Dataset

The EuRoC datasets[22] are collected from a binocular fisheye camera (Aptina MT9V034) and synchronized inertial measurement unit (Analog Devices ADIS 16448) carried by a micro aerial robot. The resolution of the binocular camera is 752×480, and the exposure

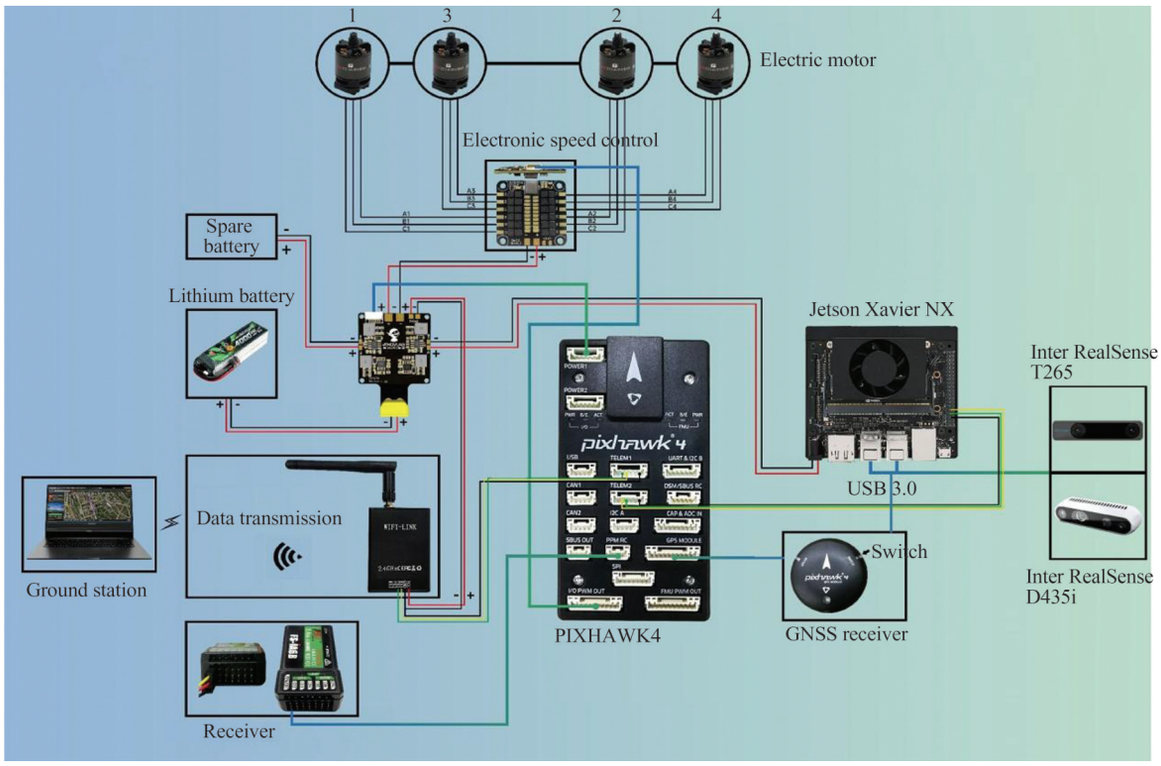

Fig. 3 The aerial robot implementation scheme

mode of this camera is a global shutter with 20 Hz output frequency. The EuRoC datasets[22] contain 11 sequences, which includes different lighting conditions and different environments. We compare the proposed RRVPE with OKVIS[4] and VINS-Fusion[16] in EuRoC datasets. OKVIS is another nonlinear optimization-based visual-inertial odometry, and VINS-Fusion is the state-of-the-art KLT sparse optical flow tracking-based tightly coupled agent state estimator.

All methods are compared in a NVIDIA Jetson Xavier NX embedded device, as shown in Fig. 4. The NVIDIA Jetson series devices are slightly different from other onboard computers on the score of its GPU module with 384 CUDA cores, which allows the RRVPE system to execute in real time with CUDA parallel acceleration. The comparison of experimental results on root-mean-square error (RMSE) are shown in Table 1, which is verified by an absolute trajectory error (ATE). Figure 5 shows the system consistency on absolute pose error (APE) as time goes on in the sequence MH01. RRVPE will inevitably generate some accumulation errors over time, which is an inherent characteristic of all visual-

Fig. 4 NVIDIA Jetson Xavier NX

Table 1 Performance comparison in the EuRoC datasets on RMSE m

Sequence	OKVIS	VINS-Fusion	RRVPE
MH01	0.16	0.18	0.17
MH02	0.22	0.12	0.14
MH03	0.24	0.23	0.13
MH04	0.34	0.29	0.23
MH05	0.47	0.25	0.35
V101	0.09	0.12	0.14
V102	0.20	0.13	0.05
V103	0.24	0.07	0.13
V201	0.13	0.09	0.07
V202	0.16	0.14	0.09
V203	0.29	0.23	0.18
Average	0.23	0.17	0.15

based robot state estimators. Fortunately, due to the local bundle adjustment, the accumulation error of the RRVPE system is always within a reasonable range. The experimental results show that, on the NVIDIA Jetson Xavier NX embedded companion computer, RRVPE system shows a favorable accuracy comparing with other state-of-the-art agent sate estimators, and achieves real-time performance.

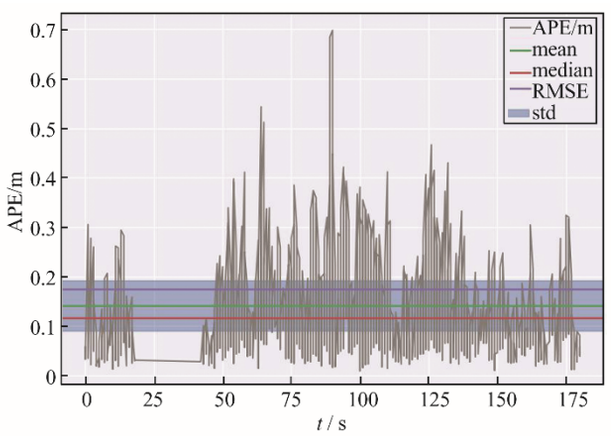

Fig. 5 The change of APE as time goes on in sequence MH01

3.3 Simulation for Aerial Robot Navigation

Due to the instability of the open source flight control algorithm, a simulation test is needed before real-world flight, which can effectively avoid the aerial robot crash caused by a program error. We carried out the virtual experiment for aerial robot autonomous navigation in the Gazebo simulator, as shown in Fig. 6. After taking off, the aerial robot leverages a virtual plug-in stereo camera and GNSS raw signals to obtain the spatial position. Meanwhile, a 3D voxel map calculated by a virtual plug-in RGB-D camera is structured to further capture the transformation matrix between the aerial robot and neighbouring obstruction. When the flight destination is entered manually, the trajectory planner generates a path

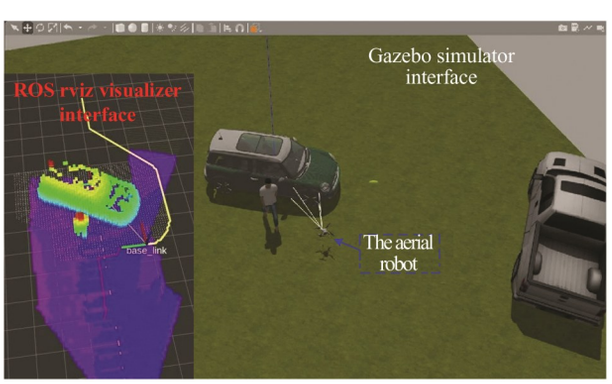

Fig. 6 Aerial robot navigation test carried out in the Gazebo simulator

for the aerial robot motion and sends the desired speed to the flight controller, then gradually approaches the destination and keeps a fixed distance from the neighbouring obstruction.

3.4 Real-World Aerial Robot Navigation

In order to verify the robustness and practicability of the proposed aerial robot navigation system, we conduct both simulation and real-world physical verification similar to the Gazebo test. The visual-inertial sensor used in our real-world test is an Intel RealSense T265 binocular camera. In the meantime, an Intel RealSense D435i RGB-D camera is used to capture the 3D environmental map. In addition, the U-Blox ZED-F9P is employed as GNSS receiver that is a high-precision multi-band receiver with multi-constellation support. The real-world experiment was conducted on a campus tennis court, where the sky is open and most of the navigation satellites are well tracked. The terrain crossed by the aerial robot is an artificial arbitrary obstruction, as shown in Fig. 7.

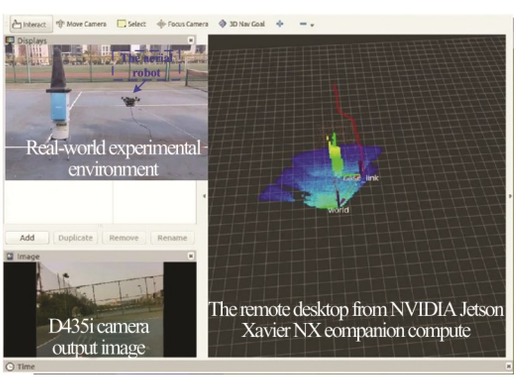

Fig. 7 The real-world navigation environment conducted on a campus tennis court

During flight, the aerial robot can change its route when approaching an obstacle and always keep a reasonable distance from the neighbouring obstruction. It is worth noting that all flight control commands are generated by the NVIDIA Jetson Xavier NX board, and the flight autopilot does not receive any external control signal generated by external environment.

4 Conclusion

In this paper, we proposed RRVPE: a robust and real-time visual-inertial-GNSS tightly coupled pose estimator for aerial robot navigation, which combines KLT sparse optical flow, inertial measurements and GNSS

raw signal to estimate aerial robot state between consecutive images. In the nonlinear optimization phase, visual-inertial-GNSS raw measurements were formulated by the probabilistic factor graph in a small sliding container. The RRVPE system can achieve real-time robot state estimation with CUDA acceleration on an airborne computer. The proposed system is evaluated using both simulated and real-world physical experiments, demonstrating clear advantages over state-of-the-art approaches.

References

[1] Lei L, Li Z H, Yang H, *et al*. Extraction of the leaf area density of maize using UAV-LiDAR data [J]. *Geomatics and Information Science of Wuhan University*, 2021, **46**(11): 1737-1745(Ch).

[2] Chen J J, Li S, Liu D H, *et al*. AiRobSim: Simulating a multisensor aerial robot for urban search and rescue operation and training [J]. *Sensors* (*Basel*, *Switzerland*), 2020, **20**(18): 5223.

[3] Tabib W, Goel K, Yao J, *et al*. Autonomous cave surveying with an aerial robot [EB/OL].[2022-06-25]. https://arxiv.org/abs/2003.13883.

[4] Geneva P, Eckenhoff K, Lee W, *et al*. OpenVINS: A research platform for visual-inertial estimation [C]// 2020 *IEEE International Conference on Robotics and Automation* (*ICRA*). New York: IEEE, 2020: 4666-4672.

[5] Paul M K, Roumeliotis S I. Alternating-stereo VINS: Observability analysis and performance evaluation [C]//2018 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. New York: IEEE, 2018: 4729-4737.

[6] Qin T, Li P L, Shen S J. VINS-mono: A robust and versatile monocular visual-inertial state estimator [J]. *IEEE Transactions on Robotics*, 2018, **34**(4): 1004-1020.

[7] Rosinol A, Abate M, Chang Y, *et al*. Kimera: An open-source library for real-time metric-semantic localization and mapping [C]// 2020 *IEEE International Conference on Robotics and Automation* (*ICRA*). New York: IEEE, 2020: 1689-1696.

[8] Campos C, Elvira R, Rodríguez J J G, *et al*. ORB-SLAM3: An accurate open-source library for visual, visual-inertial, and multimap SLAM [J]. *IEEE Transactions on Robotics*, 2021, **37**(6): 1874-1890.

[9] Mur-Artal R, Montiel J M M, Tardós J D. ORB-SLAM: A versatile and accurate monocular SLAM system [J]. *IEEE Transactions on Robotics*, 2015, **31**(5): 1147-1163.

[10] Mur-Artal R, Tardós J D. ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras [J]. *IEEE Transactions on Robotics*, 2017, **33**(5): 1255-1262.

[11] Li T, Zhang H P, Gao Z Z, *et al*. Tight fusion of a monocular camera, MEMS-IMU, and single-frequency multi-GNSS RTK for precise navigation in GNSS-challenged environments [J]. *Remote Sensing*, 2019, **11**(6): 610.

[12] Cao S Z, Lu X Y, Shen S J. GVINS: Tightly coupled GNSS-visual-inertial fusion for smooth and consistent state estimation [J]. *IEEE Transactions on Robotics*, 2022, **38**(4): 2004-2021.

[13] Zhang C, Yang Z, Fang Q H, *et al*. FRL-SLAM: A fast, robust and lightweight SLAM system for quadruped robot navigation [C]//2021 *IEEE International Conference on Robotics and Biomimetics* (*ROBIO*). New York: IEEE, 2022: 1165-1170.

[14] Zhang C, Yang Z, Liao L W, *et al*. RPEOD: A real-time pose estimation and object detection system for aerial robot target tracking [J]. *Machines*, 2022, **10**(3): 181.

[15] Lucas B D, Kanade T. An iterative image registration technique with an application to stereo vision [C]// *Proceedings of the 7th International Joint Conference on Artificial Intelligence*. New York: ACM, 1981: 674-679.

[16] Shi J B, Tomasi. Good features to track [C]//1994 *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. New York: IEEE, 2002: 593-600.

[17] Leutenegger S, Lynen S, Bosse M, *et al*. Keyframe-based visual-inertial odometry using nonlinear optimization [J]. *The International Journal of Robotics Research*, 2015, **34** (3): 314-334.

[18] Forster C, Pizzoli M, Scaramuzza D. SVO: Fast semi-direct monocular visual odometry [C]// 2014 *IEEE International Conference on Robotics and Automation* (*ICRA*). New York: IEEE, 2014: 15-22.

[19] Engel J, Schöps T, Cremers D. LSD-SLAM: Large-scale direct monocular SLAM [C]// *Proceedings of the European Conference on Computer Vision* (*ECCV*). Berlin: Springer, 2014: 834-849.

[20] Qin T, Li P L, Shen S J. Relocalization, global optimization and map merging for monocular visual-inertial SLAM [C]// 2018 *IEEE International Conference on Robotics and Automation* (*ICRA*). New York: IEEE, 2018: 1197-1204.

[21] Qin T, Shen S J. Robust initialization of monocular visual-inertial estimation on aerial robots [C]//2017 *IEEE/RSJ International Conference on Intelligent Robots and Systems* (*IROS*). New York: IEEE, 2017: 4225-4232.

[22] Burri M, Nikolic J, Gohl P, *et al*. The EuRoC micro aerial vehicle datasets [J]. *The International Journal of Robotics-Research*, 2016, **35**(10): 1157-1163. □