Proceedings of the 42nd Chinese Control Conference
July 24-26, 2023, Tianjin, China

RL-VIO: A Robust and Lightweight Visual-inertial Odometry in Dynamic Environments

Feixiang Zheng[1], Wanbiao Lin[2], Lei Sun[1]

1. College of Artificial Intelligence, Nankai University, Tianjin 300350, P. R. China
E-mail: 2120210382@mail.nankai.edu.cn, sunl@nankai.edu.cn

2. Shenzhen Research Institute, Nankai University, Shenzhen 518081, P. R. China
E-mail: 15222071259@139.com

Abstract: The invariable location of the features in the world is the precondition of calculating poses and building maps in SLAM algorithms whose performance will descend in dynamic environments. In this paper, we present a novel visual-inertial odometry which is lightweight and robust enough to produce an accurate trajectory while working with moving objects, called RL-VIO. First of all, a sliding window filter scheme is used as the main estimator to fuse visual and inertial information. The efficiency of feature correspondence is maintained by a robust feature tracking method. Then, the proposed dynamic points removal strategy prevents the system from being influenced by non-static features. A comprehensive stationary detection function which employs both visual and inertial measurements is also presented to further improve the accuracy of the filter. Finally, we use the public VIODE dataset to evalute our algorithm and the results reveal that the RMSE ATE of RL-VIO is much lower than our predecessor HybVIO with time consumption being still at a low level.

Key Words: Visual-inertial Odometry, dynamic environment, stationary detection

1 Introduction

The foundation of the autonomous mobile platforms moving in GPS-denied environments is to precisely estimate their states using sensors equipped on them. Leveraging camera and IMU to finish this job has aroused great interest among the research community and industry because of their low costs and property of complementing with each other. Compared to the whole SLAM, pure odometry concentrates on pose estimation rather than generating a consistent map which is more suitable for commercial devices and edge computers. Thus the visual-inertial odometry (VIO) algorithms have achieved vigorous development in the past few decades and been the core technique of autonomous driving, unmanned flying drones and AR/VR.

However, state-of-the-art VIO algorithms [1–4] work on the assumption that the objects in the world are definitely static so the trajectory estimated by these algorithms might diverge if there are dynamic objects moving into view. Although combining inertial measurements with visual inputs makes VIO methods more robust than pure visual algorithms in short-term dynamic scenarios, explicit elimination of the severe effects is still crucial if dynamic objects are main features in the surrounding area.

Some researches like [5, 6] detect dynamic points by introducing depth information measured by a RGB-D sensor while the accuracy of the depth measurements of the RGB-D sensors become unreliable when working outdoor. RANSAC-based methods like [7] could identify a certain number of incorrect feature pairs only when there are more static correspondences than dynamic ones. Plenty of works such as [8–10] choose to detect dynamic regions in the image based on deep learning. Neglect the high computation demanding, these algorithms merely recognize predefined objects. Other approaches like the optical-flow-based method [11] and the probability-based method [12] are designed to improve the robustness and accuracy of pure visual algorithms and few works aim at addressing positioning issues in dynamic environments using a VIO without deep neural networks or prior depth measurements.

For this reason, in this paper we present RL-VIO, a robust and lightweight visual-inertial odometry in dynamic environments. The core of our algorithm is a filter-based sliding window visual-inertial estimator including robust feature tracking, dynamic points removal and comprehensive stationary detection. This framework can tolerate varying amounts of dynamic objects in different situations without introducing DNNs or a RGB-D camera hence the high efficiency is guaranteed while providing competitive accuracy against other SOTA algorithms.

2 Related Works

2.1 Visual-inertial Odometry

To futher address the limitations of the pure visual odometry or SLAM methods, abundant researches adopted the IMU to correct the scale and improve the robustness. OKVIS [13] introduced a framework based on recent keyframes which implements local optimization of trajectory using visual reprojection residuals and IMU preintegration residuals. VINS-Mono [1] whose feature tracking is finished by the KLT sparse optical flow [14] applies a tightly-coupled, nonlinear optimization-based method which is one of the most famous SOTA algorithms. VI-DSO [2] is the representative of the direct motheds fusing photometric errors and IMU errors to jointly estimate camera poses and sparse scene geometry.

Filter-based methods are more efficient than those based on optimization while produce estimated results of the same quality. ROVIO [4] is a robocentric approach that performs minimization directly on image intensity patches using an iterate EKF. MSCKF [15] is the first tightly-coupled visual inertial algorithm that maintains a sliding window of past poses to balance the computational complexity and the re-

This work is supported by National Natural Science Foundation (NNSF) of China under Grant 62173192 and Shenzhen Natural Science Foundation under Grant JCYJ20220530162202005.

sulting accuracy, which is the predecessor of plentiful filter-based frameworks such as S-MSCKF [16] and OpenVINS [17]. The visual updates of these MSCKF-based approaches need explicit marginalization triggered by the null space projection which brings errors influencing the accuracy. Hyb-VIO [3] tackled this issue by integrating the estimated feature coordinate into the update model and the linearization error is the only error during the update process.

These algorithms mentioned above have shown an excellent performance in static environments while the ability to deal with the dynamic objects is still limited.

2.2 Dynamic Objects Processing Without DNNs or Prior Depth Measurements in SLAM

Diverse solutions have been proposed to reduce the affects caused by the dynamic objects. Cheng et al. [11] proposed an approach whose input is only RGB images choosing optical flow to distinguish dynamic features. The transformantion between two visual frames is firstly obtained through an ego-motion estimation and a feature pair is rejected if the optical flow value calculated by the ego-motion and sparse KLT algorithm is greater than the threshold. Based on a monocular camera, not only the FVB contraint was derived to detect dynamic objects when the epipolar constraint fails but a bayesian framework was deduced to further model the static probability of a pixel in [18]. Fan et al. [19] constructed two constraints to locate the dynamic regions by decomposing the camera motions. RANSAC is frequently used to acquire the correct mathematic model from a group of stochastic data in many SLAM frameworks. Based on this, a GC-RANSAC algorithm which utilizes the static epipolar geometry knowledge to filter out the dynamic feature pairs in an iterative way was proposed in [7]. The detection ability of these multi-view geometry-based approaches require accurate camera poses while the consistency will decline if the moving objects are the major content without an inertial sensor.

Fu et al. [20] proposed a method using IMU measurements between two camera frames. Erroneous feature matchings is eliminated if the geometry constraint caculated by the integration is abnormal. However, the bias of the sensor makes the data from the gyroscope and accelerometer unstable, thus the results of identifying dynamic points mainly counting on the IMU are not reliable enough. The system in [21] is similar to ours. The feature tracking is implemented in a forward-reverse way and the rejection of dynamic points is finished by a multi-stage outlier removal including a triangulation depth check, a flow vector distance judgement and a chi-square test. Nevertheless, a stereo configuration is needed and in the paper the filter is MSCKF-based while the visual updates are carried out without the null space projection in our algorithm which provide higher accuracy as mentioned earlier and a sensor set that includes only a monocular camera and an IMU is required in our framework.

3 Visual-inertial Information Fusion

3.1 State Definition

The state vector of our system consists of the current states and a fixed-length window of past poses as shown below:

$$x_k = (p_k, q_k, v_k, \lambda_k, b_k^a, b_k^g, T_k^a, \pi^{(1)}, \pi^{(2)}, \ldots, \pi^{(n_a)}) \quad (1)$$

where p_k, q_k is the pose of the IMU sensor at the latest timestamp k and v_k is the current velocity. An IMU-camera time shift is represented by λ_k. (b_k^a, b_k^g, T_k^a) is a vector of IMU biases and note that only the diagonal elements of T_k^a is used to compensate the scale error of the accelerometer. The remain variables constitute a sliding window of past poses of the IMU sensor with a fixed length n_a.

3.2 IMU Propagation

Every IMU measurement is used to propagate the states as an EKF prediction process by means of a discrete-time step:

$$\begin{pmatrix} p_k \\ v_k \\ q_k \end{pmatrix} = \begin{pmatrix} p_{k-1} + v_{k-1}\Delta t_k \\ v_{k-1} + \left[q_k\left(a_k^* + \varepsilon_k^a\right)\overline{q}_k - g\right]\Delta t_k \\ \Omega\left[\left(\omega_k^* + \varepsilon_k^\omega\right)\Delta t_k\right] q_{k-1} \end{pmatrix} \quad (2)$$

where Δt_k is the newest time interval and the inputs of the gyroscope and accelerometer are biased as $a_k^* = T_k^a a_k - b_k^a$ and $\omega_k^* = \omega_k - b_k^g$. The rotation using the quaternion is denoted by $q_k\left(\bullet\right)\overline{q}_k$ and the increment of the quaternion is calculated by $\Omega : R^3 -> R^{4\times4}$. g is the constant gravity.

3.3 Track Selection

An extracted feature j is tracked frame by frame using the method proposed in Section 4 and a track path t_j is left in the sliding window as the pixel coordinate of the feature changes continuously. Useful tracks are selected to implement the visual updates. One novelty of the selection mechanism is that the points of the same track used in the last visual update is passed to insure the fresher information has the priority:

$$t_{j,k}^* = \{1, \ldots, f_{j,k-1} - 1\} \cup \{f_{j,max}\} \quad (3)$$

where $t_{j,k}^*$ is the selected frame indices of track j and $f_{j,k-1}$ is the minimal frame index used by the last visual update. The maximum index of the track kept in the window $f_{j,max} = max(f_{j,start}, f_{j,kept})$ is also chosen to get the sufficient accuracy of the triangulation (see the following subsection). $f_{j,start}$ is the index which belongs to the first frame detecting the feature j and $f_{j,kept}$ denotes the oldest frame kept in the window observing it.

On the other hand, the longer a point is tracked, the more constraints on the trajectory are provided. Hence the tracking length is the criterion to pick the features that possess more information:

$$A_k = \{i \in C_i | L(t_{i,k}) > m(t_{\bullet,k})\} \quad (4)$$

where A_k is the set of the final selected tracks. C_i is the indices of tracks to select. And

$$L(t_{i,k}) = \Sigma_{n \in t_{i,k}^* / f_{i,max}} \parallel x_n - x_{n-1} \parallel_1 \quad (5)$$

where x_n is the pixel coordinate of the track of the frame n in the window. $m(t_{\bullet,k})$ is the average tracking length of all tracks to select and a random way is chosen to implement the selection.

3.4 Visual Updates

An available feature track triggers a visual update to prevent the estimation from diverging. The complete measurement function of the selected feature j in frame i is:

$$y_{k,i}^j = h(T_{imu->cam} T_w^i \varrho_w^j) + \varepsilon^y \quad (6)$$

where $h(\bullet)$ is the camera projection function and ε^y is the measurement noise.The extrinsic transformation is denoted by $T_{imu->cam}$ and T_w^i is the pose of the frame i. The world coordinate of the feature j represented by ϱ_w^j is obtained through a multi-frame triangulation process:

$$\varrho_w^j = Tri(x_k, y_k^j) \tag{7}$$

using the GN method. The full correlation between all the state variables is explicitly calculated in this step and the jacobian of ϱ_w^j with respect to the state vector x_k is acquired. Thus the null space projection in [15] is avoided.

Finally, by stacking all the measurements of the feature j, a linearization is implemented to get the whole measurement jacobian to be used in the visual update:

$$y_k^j = J_k^j(x_0)(x_k - x_0) + h_0 + \varepsilon^y \tag{8}$$

and before this a chi-square test is also performed to reject illegal residuals.

3.5 Pose Augmentation

The poses in the sliding window is updated as the newer frames come. The current pose of the body is inserted at the head of the window and therefore an old pose d is discarded using matrix multiplication:

$$x_k^* = Aug_d x_k \tag{9}$$

with

$$Aug_d = \begin{pmatrix} I_{20} & & & \\ I_7 & & & \\ & I_{7(d-1)} & & \\ & & & I_{7(n_a-d)} \end{pmatrix}. \tag{10}$$

The number d can be flexibly set to formulate a smart pose augmentation mechanism and a Tower-of-Hanoi scheme is applied in our algorithm to make the old poses more spatially dispersed.

4 Robust Feature Tracking

The first step of the front end is to detect strong point features in the image which is performed by the GFTT algorithm [22] in our framework. Before features are tracked by the sparse KLT optical flow [14], we introduce the inertial information to get the initial values of them in the next image to improve the stability of the tracking in the case of fast movement.

For two consecutive images M_1 and M_2, the rotation R_{21} is integrated using the gyroscope measurements between them. We denote ϱ_1 and ϱ_2 as the coordinates of the same 3D point ϱ_w on the normalized planes of M_1 and M_2 with ν_1 and ν_2 being the corresponding pixel coordinates. ζ_1 and ζ_2 is their depths. The transformation between ϱ_1 and ϱ_2 is described by the following equation:

$$\zeta_2 \varrho_2 = \zeta_1 R_{21} \varrho_1 + t_{21} \tag{11}$$

where t_{21} is the translation between the two frames. With the camera intrinsic parameter K, the equation (11) can be rewritten as:

$$\zeta_2 K^{-1} \nu_2 = \zeta_1 R_{21} K^{-1} \nu_1 + t_{21}. \tag{12}$$

The t_{21} between the two consecutive images is small enough to be ignored and hence the values of ζ_1 and ζ_2 can be seen as equal. Based on this we finally get the IMU-prediction equation of the feature tracking:

$$\nu_2 = K R_{21} K^{-1} \nu_1. \tag{13}$$

After the prediction step, a forward-reverse tracking method is utilized to further reject the sliding features caused by the edges or dynamic objects in the image. We use the initial values estimated by the process above to perform the forward optical flow to determine the new positions of the points in M_2 and then a reverse tracking from M_2 to M_1 is triggered using the original positions in M_1 as the initial values. A feature pair is reserved when the following equation is satisfied:

$$\gamma = \| \nu_1^* - \nu_1 \|_1 < \sigma_\gamma \tag{14}$$

where ν_1^* is the reverse location of ν_1 and σ_γ is the rejection threshold.

5 Dynamic Points Removal

Although the forward-reverse feature tracking could filter out a small quantity of incorrect feature matches, a dominating dynamic points removal procedure is equipped in our front end to better eliminate the non-static tracking results caused by the moving objects. We use the truth that the epipolar constraint is violated when a point observed by a camera is dynamic, which is shown in Fig. 1.

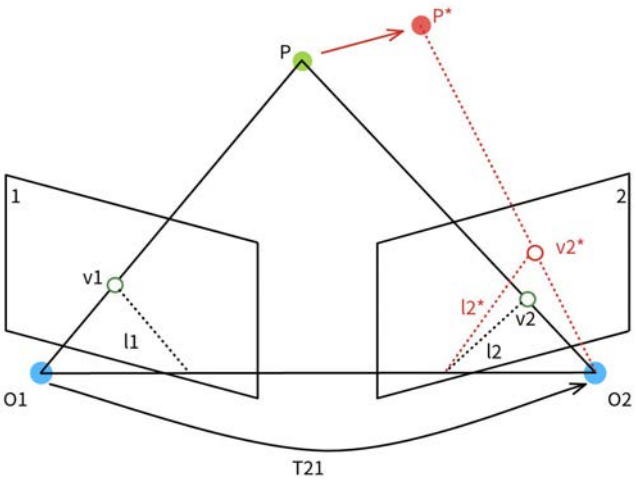

Fig. 1: The illegal consequence of the dynamic pair

For two adjacent frames image 1 and image 2, a point P in the world is viewed by them and $v1$ and $v2$ are the corresponding pixels if P is static. The epipolar lines are marked as $l1$ and $l2$. Once the coordinate of P changes before image 2 is received, the normal position $v2$ will be replaced by the dynamic pixel $v2^*$, which brings a feature match having harmful effects on pose estimation. In other words, the distance between the tracked feature and the epipolar line whose parameters can be calculated with $T21$ and $v1$ is supposed to be zero. Hence we accept a tracked feature if

$$d = \frac{|A_l x_2 + B_l y_2 + C_l|}{\sqrt{A_l^2 + B_l^2}} < \sigma^d \tag{15}$$

where $l = [A_l, B_l, C_l]^T$ is the parameters of the epipolar line in the image 2 and $v2 = [x_2, y_2]$ is the coordinate of the tracked feature. A threshold σ^d is used because of the measurement noise.

The key of the calculation of the epipolar line is the precision of $T21$, that is to say the fundamental matrix F. As long as we have the matrix, l is obtained using equation (16).

$$l = FP1 \tag{16}$$

where $P1 = [x_1, y_1, 1]^T$ with $v1 = [x_1, y_1]^T$. To get the transformation, inspired by [23] , we divide the image into nine regions which is shown in Fig. 2 and then the average

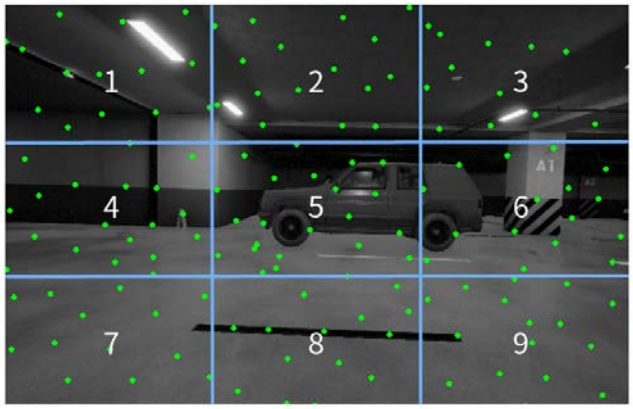

Fig. 2: The nine regions divided in the image

tracked distance of all the features in every region is calculated. The features in the region with the maximum and minimum distance are overlooked which means only seven regions are used to compute F. The reason why the region with the minimum distance is also refused is to improve the robustness of the algorithm in the case that dynamic objects moves in parallel with the camera. The experiments in Section 7 corroborate the effectiveness of this removal method. And finally a hybrid 2-point and 5-point RANSAC is implemented to further discard inaccurate feature matches.

6 Comprehensive Stationary Detection

On the basis of the management in Section 3.5, the stationary of the estimated platform will cause a sharp contraction of the sliding window thus a special care is required. A single prediction step is launched in our algorithm to perform a pose unaugmentation if we check a stationary case:

$$x_k^* = Aug_{n_a}^T x_k + \begin{pmatrix} 0_{dim(x_k)-7} & \\ & \varepsilon^u \end{pmatrix}, \tag{17}$$

which makes the previous augmented frame discarded and other frames remain in the state vector x_k^*.

The first way employed in the algorithm to determine the stationary of the system counts on the image features, that is, if

$$T_k^v = \max_i \parallel v_{i,curr} - v_{i,prev} \parallel < \sigma^v \tag{18}$$

where i is the track number in the selected track set and $\delta v = v_{\bullet,curr} - v_{\bullet,prev}$ denotes the pixel difference between the current frame and the previous frame, the visual stationary signal s_v is considered as true.

Nonetheless, carrying out stationary detection using only image features is not always dependable in dynamic environments because of the moving points. The inertial measurements are related to the motion of the mobile platform merely and we introduce two zero velocity detectors summarized in [24]. The first one is called acceleration moving variance detector:

$$T_k^a = \frac{1}{N_a} \sum_{j \in \Omega^a} \parallel y_a - \overline{y_a} \parallel < \sigma^a \tag{19}$$

where N^a is the number of the acceleration measurements between the consecutive frames. The acceleration stationary signal s_a is set to true if equation (19) is satisfied. The other one is angular rate energy detector, and if

$$T_k^w = \frac{1}{N_w} \sum_{j \in \Omega^w} \parallel y_w \parallel < \sigma^w, \tag{20}$$

the angular stationary signal s_w is set to true. Only when

$$s = s_v \&\& s_a \&\& s_w \tag{21}$$

is true, a zero velocity update is conducted and equation (17) is performed.

The biases may make the two inertial judgements above get mistaken results so the visual criterion is persisted in our framework. Besides, we perform a velocity magnitude test in advance to prevent the generation of the inertial stationary signals when the platform is moving in a constant-velocity way.

7 Experiment

7.1 Dataset and Preparation

We choose a public dynamic dataset called VIODE [25] to evaluate the proposed VIO and one of the SOTA filter-based methods HybVIO. VIODE dataset is released to push the development of the VIO algorithms in dynamic environments. The simulated dynamic scenes in which an UAV with a stereo camera and an IMU navigates are built in the dataset. There are three prepared categories named *city_day*, *city_night* and *parking_lot* and every one of them has four sequences ranging in difficulty from none to high in which the drone runs the same path and the only difference between them is the number of the moving objects. Moreover, the fake dynamic units such as static cars are also placed, which makes the dataset a challenging benchmark for the visual-based methods.

We use *evo* toolbox to compute the root mean square error (RMSE) of the absolute trajectory error (ATE) for comparison. Every sequence is tested for five times and we use the average RMSE for evalution because of randomness. A desktop computer with an Intel Core i7-9750H processor @2.60GHz is utilized as the main platform running Ubuntu 18.04 with ROS Melodic.

7.2 Results

As Table 1 illustrates, our algorithm outperforms HybVIO in most of the sequences especially in mids and $high$s. The robust tracking method proposed in Section 4 guarantees the accuracy of the feature correspondences thus the results of our algorithm are better even in static scenes in the $none$s.

The fact is that the performance of the both VIO methods become worse as the number of dynamic objects increases. But the RMSE ATE of ours still remains at a much lower level, which proves the better robustness of the proposed algorithm with the dynamic points removal strategy in Section 5.

Table 1: RMSE ATE of VIODE Dataset

Sequences		HybVIO	ours
city_day	none	0.204	**0.186**
	low	0.216	**0.207**
	mid	0.218	**0.152**
	high	0.619	**0.223**
city_night	none	0.344	**0.300**
	low	0.927	**0.376**
	mid	1.259	**0.365**
	high	0.971	**0.580**
parking_lot	none	0.167	**0.157**
	low	**0.180**	0.195
	mid	1.256	**0.287**
	high	0.608	**0.328**

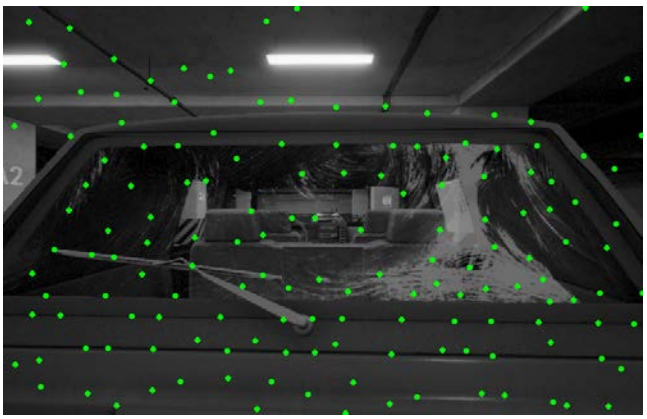

Fig. 3: A captured image from the camera with a driving car having the parallel forward motion

Furthermore, in mid and $high$ of $parking\_lot$, there is a special case that a driving car comes into the view of the camera and then keeps moving forward in parallel with the camera, which is shown in Fig 3. Thus a stationary signal is produced by HybVIO because of the static detecting based only on image features so that the estimated pose remains unchanged while the simulated drone is still having a forward flight. Fig 4 illustrates the comparison of the trajectories estimated by HybVIO and ours in mid sequence of $parking\_lot$. On the contrary, the comprehensive stationary detection in Section 6 is able to deal with the situation correctly, as a result, an accurate trajectory can be produced by our algorithm.

7.3 Time Analysis

Table 2: Time Consumption of RL-VIO

Module	Front End	Visual Update	Others	Total
Time(ms)	16.0	5.2	2.1	23.3

We further evaluate the time consumption of our RL-VIO using the mid sequence of $parking\_lot$, which is shown in

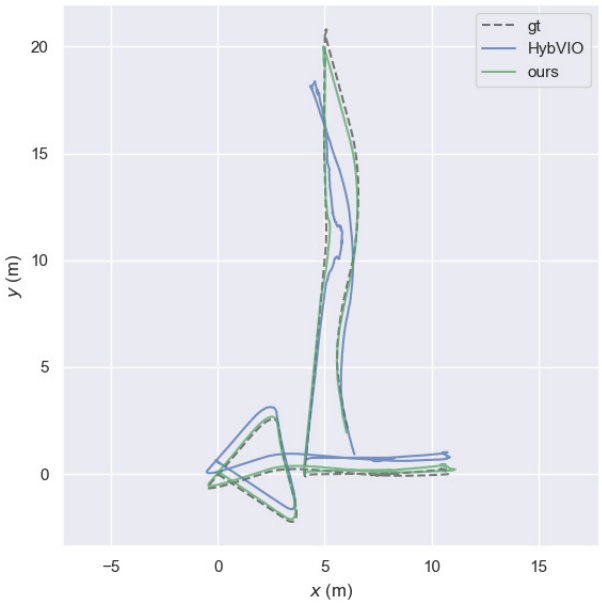

Fig. 4: Results of HybVIO and ours in mid sequence of $parking\_lot$

Table 2. With no DNNs, high computation load is avoided thus the costed time of every module is small and performing estimation at a high frequency is not a tough task for our algorithm even using some edge computers.

8 Conclusions

In this paper, RL-VIO is proposed as a robust and lightweight visual-inertial odometry in dynamic environments. An exquisite sliding window filter with discrete propagation, special track selection, accurate visual update and flexible pose management is the main estimator. A strong front end consists of robust feature tracking, dynamic points removal and comprehensive stationary detection offers our algorithm improved robustness and accuracy when working with moving objects without demanding heavy computation cost, which is demonstrated on the public VIODE dataset. In the future, we will try to introduce some extra residuals related to dynamic objects in visual updates to form a robust back end eliminating incorrect information more precisely.

References

[1] T. Qin, P. Li, and S. Shen, VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator, *IEEE Transactions on Robotics*, 34(4), 1004-1020, 2018.

[2] L. V. Stumberg, V. Usenko, and D. Cremers, Direct Sparse Visual-Inertial Odometry using Dynamic Marginalization, in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2018: 2510-2517.

[3] O. Seiskari, P. Rantalankila, J. Kannala, J. Ylilammi, E, Rahtu, and A. Solin, HybVIO: Pushing the Limits of Real-time Visual-inertial Odometry, in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022: 701-710.

[4] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, Robust Visual Inertial Odometry Using a Direct EKF-Based Approach, in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015: 298-304.

[5] Y. Liu, Y. Wu, and W. Pan, Dynamic RGB-D SLAM Based on Static Probability and Observation Number, *IEEE Transactions on Instrumentation and Measurement*, 70, 1-11, 2021.

[6] W. Dai, Y. Zhang, P. Li, Z. Fang, and S. Scherer, RGB-D SLAM in Dynamic Environments Using Point Correlations, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(1), 373-389, 2020.

[7] Z. Du, S. Huang, T. Mu, Q. Zhao, R. R. Martin, and K. Xu, Accurate Dynamic SLAM Using CRF-Based Long-Term Consistency, *IEEE Transactions on Visualization and Computer Graphics*, 28(4), 1745-1757, 2020.

[8] B. Bescos, J. M. Fácil, J. Civera, and J. Neira, DynaSLAM: Tracking, Mapping and Inpainting in Dynamic Scenes, *IEEE Robotics and Automation Letters*, 3(4), 4076-4083, 2018.

[9] C. Yu, Z. Liu, X. Liu, F. Xie, Y. Yang, Q. Wei, and Q Fei, DS-SLAM: A Semantic Visual SLAM towards Dynamic Environments, in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018: 1168-1174.

[10] Z. Xing, X. Zhong, and D. Dong, DE-SLAM: SLAM for highly dynamic environment, *Journal of Field Robotics*, 39(5), 528-542, 2022.

[11] J. Cheng, Y. Sun, and M. Q.-H. Meng, Improving monocular visual SLAM in dynamic environments: an optical-flow-based approach, *Advanced Robotics*, 33(12), 576-589, 2019.

[12] X. Hu, Y. Zhang, Z. Cao, R. Ma, Y. Wu, Z. Deng, and W. Sun, CFP-SLAM: A Real-time Visual SLAM Based on Coarse-to-Fine Probability in Dynamic Environments, in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2022: 4399-4406.

[13] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, Keyframe-Based Visual-Inertial Odometry Using Nonlinear Optimization, *The International Journal of Robotics Research*, 34(3), 314-334, 2015.

[14] B. D. Lucas, and T. Kanade, An iterative image registration technique with an application to stereo vision, in *Proceedings of the International Conference on Artificial Intelligence*, 1981: 674–679.

[15] A. I. Mourikis, and S. I. Roumeliotis, A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation, in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2007: 3565–3572.

[16] K. Sun, K. Mohta, B. Pfrommer, M. Watterson, S. Liu, Y. Mulgaonkar, C. J. Taylor, and V. Kumar, Robust Stereo Visual Inertial Odometry for Fast Autonomous Flight, *IEEE Robotics and Automation Letters*, 3(2), 965-972, 2018.

[17] P. Geneva, K. Eckenhoff, W. Lee, Y. Yang, and G. Huang, OpenVINS: A Research Platform for Visual-Inertial Estimation, in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2020: 4666-4672.

[18] A. Kundu, K. M. Krishna, and J. Sivaswamy, Moving Object Detection by Multi-View Geometric Techniques from a Single Camera Mounted Robot, in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009: 4306-4312.

[19] Y. Fan, H. Han, Y. Tang, and T. Zhi, Dynamic objects elimination in SLAM based on image fusion, *Pattern Recognition Letters*, 127, 191-201, 2018.

[20] D. Fu, H. Xia, and Y. Qiao, Monocular Visual-Inertial Navigation for Dynamic Environment, *Remote Sensing*, 13(9), 1610, 2021.

[21] D. V. Nam, and K. Gon-Woo, Robust Stereo Visual Inertial Navigation System Based on Multi-Stage Outlier Removal in Dynamic Environments, *Sensors*, 20(10), 2922, 2020.

[22] J. Shi, and C. Tomasi, Good features to track, in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1994: 593-600.

[23] X. Qiu, H. Zhang, and W. Fu, Lightweight hybrid visual-inertial odometry with closed-form zero velocity update, *Chinese Journal of Aeronautics*, 33(12), 3344-3359, 2020.

[24] I. Skog, P. Händel, J.-O. Nilsson and J. Rantakokko, Zero-Velocity Detection—An Algorithm Evaluation, *IEEE transactions on biomedical engineering*, 57(11), 2657-2666, 2010.

[25] K. Minoda, F. Schilling, V. Wüest, D. Floreano, and T. Yairi, VIODE: A Simulated Dataset to Address the Challenges of Visual-Inertial Odometry in Dynamic Environments, *IEEE Robotics and Automation Letters*, 6(2), 1343-1350, 2021.