Stereo Visual Odometry with Light and Adaptive Feature Tracking

Xin Huang[1, 2], Shuming Tang[1, *], Lifu Zhang[1, 2], Haibing Zhu[1], Qingxiu Du[1]
[1] Institute of Automation, Chinese Academy of Sciences, Beijing, China
[2] University of Chinese Academy of Sciences, Beijing, China
Email: {huangxin2017, shuming.tang, zhanglifu2018, haibing.zhu, qingxiu.du}@ia.ac.cn

***Abstract*—Localization technology plays a key role in autonomous driving. Stereo visual odometry is a meaningful visual localization method to estimate the pose of autonomous vehicles. VINS-Fusion provides a state-of-the-art stereo visual odometry with Kanade-Lucas-Tomasi (KLT) tracker to achieve fast feature tracking. However, KLT tracker is prone to fall into local minima in urban environments due to illumination changes and large displacements, leading to catastrophic cumulative drift over time. Aiming to solve this problem, we present a light and adaptive feature tracking technique for VINS-Fusion to get a reliable set of measurements for pose estimation. First, a disparity constraint is incorporated into left-right check to refine the measurements. Next, we propose a light bi-circular check to further remove outliers, which has high efficiency with the ingenious design. Additionally, an adaptive strategy for feature selection is proposed to dynamically balance the quantity and quality of the measurements. Experiments demonstrate that our method outperforms VINS-Fusion by producing more accurate pose estimation with 20% speedup on the KITTI odometry benchmark.**

Keywords- stereo visual odomerty; feature tracking; VINS-Fusion; bi-circular check; adaptive feature selection

I. INTRODUCTION

Localization is one of the key technologies in vehicle speed control, path planning, and obstacle avoidance, etc. Stereo visual odometry [1][2] is a meaningful vision-based localization method to estimate the pose of autonomous vehicles. With the advancement of image processing algorithms and computing power, stereo visual odometry has gained more popularity in embedded systems and becomes a vital component of autonomous driving. It has great superiority in utilizing captured images for 3D pose estimation with lower cost, especially in GPS-denied environments.

The concept of visual odometry was first proposed by Nister *et al.* [3] in 2004. In 2012, Fraundorfer *et al.* [4] indicated that stereo visual odometry could achieve good estimation of relative translation and rotation by minimizing the reprojection error of 3D space features on the 2D images. The core steps of stereo visual odometry consist of feature association and pose estimation. Feature association is based on the measurements of feature detection and tracking, and pose estimation is based on feature association. Therefore, how to obtain a reliable set of measurements is of great significance.

In order to obtain a reliable set of measurements, Deigmoeller *et al.* [5] adopted forward-backward check and disparity measure to limit the feature association obtained by Lucas & Kanade optical flows [6]. Besides, circular check was used for outlier removal. Feng *et al.* [7] incorporated KLT tracker [8] to track features in stereo images, and used time-consuming 2-point Random Sample Consensus (RANSAC) [9] and circular matching to remove outliers. Cvisic *et al.* [10] utilized SAD check, circular matching and NCC check to remove outliers and obtain feature association for pose estimation. Fanfani *et al.* [11] used sGLOH descriptor with sCOr nearest neighbor matching to get initial feature association, and refined them using loop chain matching integrated with four RANSAC. Wu *et al.* [12] adopted ego-motion prior and automatic tracking failure detection scheme in order to improve the robustness of KLT tracker.

VINS-Fusion is a general optimization-based framework for pose estimation [13]. It provides four algorithms for different sensor suites: stereo cameras, monocular camera with an IMU, stereo cameras with an IMU and stereo cameras with GPS. In this paper, we focus on the algorithm for stereo cameras. Thence the VINS-Fusion in subsequent content refers to the stereo visual odometry of VINS-Fusion. VINS-Fusion incorporates KLT tracker for fast feature tracking, then adopts bundle adjustment [14] and marginalization techniques to get accurate pose estimation on the KITTI Benchmark [15]. However, KLT tracker, as proposed in [8], is prone to fail in urban environments in the cases of illumination changes and large displacements. VINS-Fusion does not take it into full consideration and adopts forward-backward check as the only criterion to judge whether a feature is tracked successfully or needs to be deleted. This simple check makes insufficient tracking failure detection with remaining outliers, and also increases the burden of optimization. Even worse, it will cause a bias in pose estimation and lead to cumulative drift over time.

In this paper, we propose a stereo visual odometry with light and adaptive feature tracking based on VINS-Fusion. The main contributions include:

- A disparity constraint is incorporated with left-right check in VINS-Fusion to refine the measurements and improve feature tracking efficiency.
- Inspired by circular matching, a light bi-circular check with high efficiency is proposed to further remove outliers.
- In order to dynamically balance the quantity and quality of measurements, an adaptive strategy for the feature selection is proposed by automatically

* Corresponding author

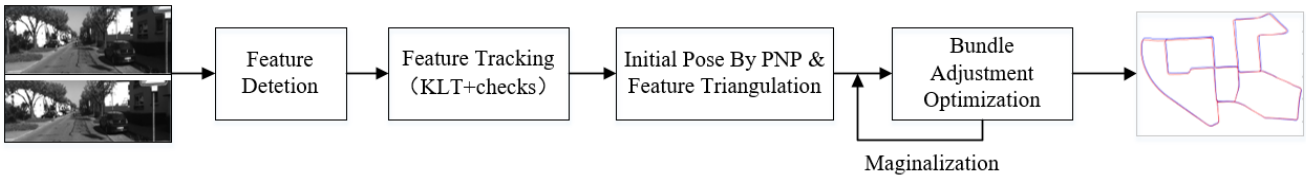


Figure 1. Flow chart of VINS-Fusion. Feature tracking is implemented by KLT tracker with checks.

adjusting the threshold of bi-circular check.

- Experimental results on the KITTI odometry benchmark have shown that our method achieves more accurate pose estimation with 20% speedup than VINS-Fusion.

II. METHODOLOGY

In this section, we first review VINS-Fusion. Then our innovations are introduced in the feature tracking module of VINS-Fusion, including the disparity constraint, light bi-circular check and adaptive feature selection.

A. Vins-Fusion

VINS-Fusion is a state-of-the-art stereo visual odometry. The flow chart of VINS-Fusion is shown in Fig. 1.

In VINS-Fusion, the states in a sliding window to be estimated are defined as:

$$\chi = [\mathrm{p}_0, \mathrm{R}_0, \mathrm{p}_1, \mathrm{R}_1, \dots, \mathrm{p}_n, \mathrm{R}_n, \lambda_0, \lambda_1, \dots, \lambda_n], \quad (1)$$

where p_i and R_i are relative translation and rotation in the world coordinate. λ_i is the depth of per feature.

In each stereo image, corner features [16] are detected and tracked by KLT tracker to establish the feature association. Features are first tracked in the temporal frame with forward-backward check, and then tracked in the spatial frame with left-right check. If one feature is tracked successfully in the temporal frame, its age is increased by 1.

Next, the 2D features in the current frame and their corresponding 3D map points are used to get the initial pose by Perspective-N-Point (PnP) method. Some new features in the frame are triangulated for the following tracking.

Based on the feature association, VINS-Fusion constructs the reprojection residual with per feature in each frame by projecting the feature from its first observation into the following frames. Suppose that the first observation of feature *l* is in the *i*-th frame, its reprojection residual in the *t*-th frame is:

$$\begin{aligned} \mathrm{z}_t^l - h_t^l(\chi) &= \mathrm{z}_t^l - h_t^l(\mathrm{R}_i, \mathrm{p}_i, \mathrm{R}_t, \mathrm{p}_t, \lambda_l) \\ &= \begin{bmatrix} u_t^l \\ v_t^l \end{bmatrix} - \pi_c\left(\mathrm{T}_t^{-1}\mathrm{T}_i\pi_c^{-1}\left(\lambda_l, \begin{bmatrix} u_i^l \\ v_i^l \end{bmatrix}\right)\right), \end{aligned} \quad (2)$$

where $\begin{bmatrix} u_t^l \\ v_t^l \end{bmatrix}$ is the observation of feature *l* on the *t*-th frame. π_c is the pinhole model which projects the feature from camera coordinate to image coordinate. T represents the camera pose, which is a 4x4 homogeneous matrix $\begin{bmatrix} \mathrm{R} & \mathrm{p} \\ 0 & 1 \end{bmatrix}$. For stereo visual odometry, VINS-Fusion also needs to construct spatial residual by the same way as it does for temporal residual.

VINS-Fusion uses a bundle adjustment formulation, and obtains the maximum likelihood estimation of states by minimizing the reprojection residual:

$$\chi^* = \arg\min_{\chi} \sum_{t=0}^{n} \sum_{k\in \mathrm{S}} \left\| \mathrm{z}_t^k - h_t^k(\chi) \right\|_{\Omega_t^k}^2, \quad (3)$$

where S is the set of measurements and Ω_t^k is covariance matrix of the reprojection residual. The norm is defined as $\|\mathrm{x}\|_{\Omega}^2 = \mathrm{x}^T\Omega^{-1}\mathrm{x}$. This formulation can be solved by Gauss-Newton or Levenberg-Marquardt approaches.

As system states increase over time, marginalization is adopted to reduce computational complexity. It converts the previous measurements into a prior for the remaining states in the sliding window, and removes the past partial states from system states in order to balance accuracy and efficiency.

B. Disparity Constraint

VINS-Fusion uses KLT tracker for fast feature tracking. As an optical flow method, KLT tracker is prone to fall into local minimum in the cases of illumination changes and large displacements, leading to performance deterioration in urban environments. VINS-Fusion takes forward-backward check as the only criterion to judge whether a feature is tracked successfully or needs to be deleted. However, the established feature association may not be reliable, since this check is not thorough and still leaves some outliers. To deal with this problem, we incorporate disparity constraint with left-right check after forward-backward check.

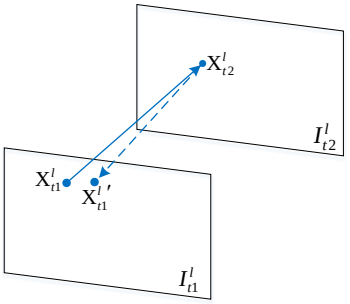


Figure 2. Forward-backward check.

We first describe forward-backward check in detail. Suppose that $\{I_{t1}^l, I_{t1}^r, I_{t2}^l, I_{t2}^r\}$ are the input images, where

t_1 and t_2 denote the consecutive frames, l and r denote the left and right images. As shown in Fig. 2, forward-backward check in VINS-Fusion is only performed between I^l_{t1} and I^l_{t2}. The feature $\mathbf{x}^l_{t1}$ in I^l_{t1} is tracked by KLT tracker to obtain the temporal matching $\mathbf{x}^l_{t2}$ in I^l_{t2}, and then $\mathbf{x}^l_{t2}$ is tracked back to I^l_{t1} with the reverse point $\mathbf{x}^l_{t1}{}'$. If the distance between $\mathbf{x}^l_{t1}$ and $\mathbf{x}^l_{t1}{}'$ is less than the threshold δ_1, the feature passes forward-backward check. Otherwise, it is rejected and deleted. The constraint for forward-backward check can be defined as:

$$\left\|\mathbf{x}^l_{t1} - \mathbf{x}^l_{t1}{}'\right\| < \delta_1. \tag{4}$$

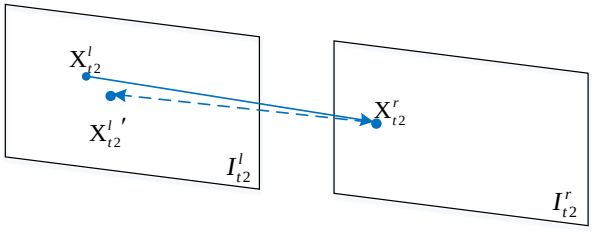


Figure 3. Left-right check.

Similarly, left-right check is performed between I^l_{t2} and I^r_{t2} in the same manner. As shown in Fig. 3, feature $\mathbf{x}^l_{t2}$ is tracked to the right image with the corresponding feature $\mathbf{x}^r_{t2}$, and tracked back with the reverse point $\mathbf{x}^l_{t2}{}'$. The constraint for left-right check is:

$$\left\|\mathbf{x}^l_{t2} - \mathbf{x}^l_{t2}{}'\right\| < \delta_2. \tag{5}$$

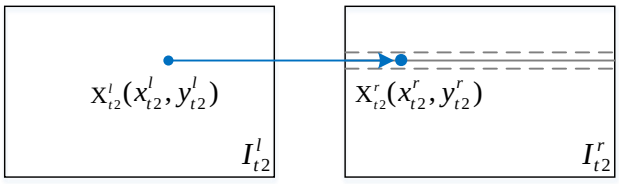


Figure 4. Disparity constraint.

Based on the left-right check, we incorporate a disparity constraint to refine the measurements as shown in Fig. 4. In the calibrated stereo images, the epipolar lines are horizontal, which means that the y-component of the spatial matching $(\mathbf{x}^l_{t2}, \mathbf{x}^r_{t2})$ should be the same. However, considering that the calibration may not be such precise, we retain the feature when the y-component difference is within a certain range. After features are tracked to the right image, we get many spatial matchings. A matching is passed if it satisfies disparity constraint. The disparity constraint is defined as:

$$\left|y^l_{t2} - y^r_{t2}\right| < \delta_3, \tag{6}$$

where the threshold δ_3 mainly depends on the calibration level. Otherwise, we directly delete the feature and do not track it back to avoid unnecessary computation on reverse KLT tracking and left-right check.

C. Light Bi-circular Check

According to [10], circular matching is an effective method to remove outliers and check the consistency of feature association. As shown in Fig. 5(a), circular matching means that per feature needs to be matched between two consecutive frames of left and right images following the order $I^l_{t2} \rightarrow I^r_{t2} \rightarrow I^r_{t1} \rightarrow I^l_{t1} \rightarrow I^l_{t2}$. We hope to combine circular matching to further remove outliers and get a reliable set of measurements for pose estimation. However, it requires double computation if circular matching is performed after forward-backward check and left-right check. Therefore, we propose a light bi-circular check, which is similar to circular matching in structure but has high efficiency. The ingenious design of bi-circular check is described as follow.

The sketch map of the proposed method is shown in Fig. 5(b). In the method, forward-backward check and left-right check are regarded as part of the bi-circular check. As another left-right check has already been performed between I^l_{t1} and I^r_{t1} in the last round, we only need to perform an additional forward-backward check between I^r_{t2} and I^r_{t1}. It is defined as:

$$\left\|\mathbf{x}^r_{t2} - \mathbf{x}^r_{t2}{}'\right\| < \delta_d, \tag{7}$$

where $\mathbf{x}^r_{t2}{}'$ is the reverse point of $\mathbf{x}^r_{t2}$ tracked to I^r_{t1} and then tracked back to I^r_{t2}. If one feature fails this check, we regard it as a tracking failure and delete it. With this light modification, a two-way loop is constructed and we name it bi-circular check. Moreover, in order to reduce computational burden, we directly delete one feature if it fails any check, rather than tracking it till the whole loop is completed.

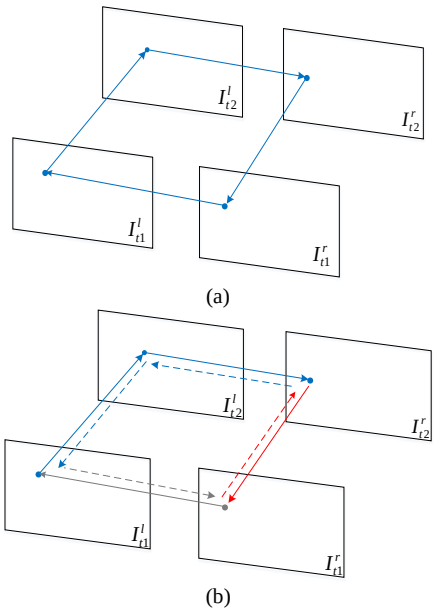


Figure 5. Sketch maps of (a) circular matching and (b) light bi-circular check. Forward-backward check and left-right check(blue) are regarded as part of the bi-circular check, and we have performed another left-right check (gray) in the last round. With additional forward-backward check between $\boldsymbol{I}^r_{t1}$ and $\boldsymbol{I}^r_{t2}$ (red), light bi-circular check is constructed.

D. Adaptive Feature Selection

In above subsections, disparity constraint is utilized to refine the measurements and light bi-circular is proposed for further outlier removal. However, the simple combination of two methods may remove valuable features since the restriction is too strict. In order to balance the quantity and quality of measurements, we propose an adaptive strategy for feature selection by automatically adjusting the threshold of bi-circular check δ_d.

Two factors are considered in determining the threshold, the number of lost features as $loss$ and the maximum of feature age as $maxtrack$. The adaptive threshold is defined as:

$$\delta_d = \rho * \frac{loss}{maxtrack} + \varepsilon, \tag{8}$$

where ρ and ε are predefined parameters. On one hand, the threshold δ_d is positively correlated with *loss*. When δ_d increases with the growth of *loss*, our model retains more features with a loose constraint. On the other hand, the threshold δ_d is negatively correlated with *maxtrack*. As stated in [10], the feature with an older age is more stable and has lower probability to be an outlier. Therefore, we consider a *maxtrack* as an index of the quality of features. When the *maxtrack* is large, features are stable and our model retains features with a tight constraint. In this way, adaptive feature selection is carried out to balance the quantity and quality of features.

To further illustrate our model, we analyze two typical examples in detail. When an autonomous vehicle is in a large movement, such as a sharp turn, the vision changes greatly and many old features are lost frequently. It will result in an increasing $loss$ and a decreasing $maxtrack$. Under this circumstance, the increasing threshold δ_d can make our model retain more useful features and alleviate performance deterioration. When the autonomous vehicle is in a mild movement, such as on the straight with a slow speed, *loss* is generally stable and *maxtrack* continues to increase. In this case, the threshold δ_d approaches ε, and features satisfied with a tight constraint are retained.

III. EVALUATION

To evaluate the proposed methods, we conduct experiments on the KITTI odometry benchmark. This dataset is usually used to evaluate the performance of visual odometry in an autonomous vehicle. KITTI contains images from real-world scenes such as urban, rural, and highways with illumination changes and displacement variations. We first introduce the experiment setup and evaluation criteria, then compare the proposed methods with VINS-Fusion in accuracy and computation time evaluation.

A. Experiment Setup

We use C++ to implement the proposed methods. All experiments are performed on an Intel® CoreTM i7-4770 CPU (3.40 GHz) computer with 16 GB memory. For the proposed methods and VINS-Fusion, the max number of features is set to 200, and features are detected and tracked using OpenCV library tools. The thresholds of forward-backward check δ_1 and left-right check δ_2 are both set to 0.5, and the threshold of disparity constraint δ_3 is set to 2. The predefined parameters ρ is set to 0.23 and ε is set to 5. We set δ_d to 5 in the experiment when the adaptive feature selection is not used.

B. Evaluation Criteria

The evaluation metric computes translational and rotational errors for all the subsequences of length (100, 200, …, 800) meters from all test sequences. We use the evaluation toolkit provided by the KITTI benchmark to compute the root mean squared error (RMSE) of translation t_{rel} and rotation r_{rel}. They are computed as:

$$t_{rel} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(t_{test} - t_{gt})^2}, \tag{9}$$

$$r_{rel} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(\theta_{test} - \theta_{gt})^2}, \tag{10}$$

where t and θ are translation and rotation of each frame, *test* and *gt* are test results and the ground truth.

C. Experiment 1:Accuracy Evaluation

We evaluate VINS-Fusion and the proposed methods as follow on 00-10 sequences of the KITTI benchmark:

- VINS-Fusion + disparity constraint (VINS-Fusion+dc)
- VINS-Fusion + bi-circular check (VINS-Fusion+bc)
- VINS-Fusion + disparity constraint + bi-circular check (VINS-Fusion+db)
- VINS-Fusion + disparity constraint + bi-circular check + adaptive feature selection (Ours).

Table I shows the average pose estimation RMSE results of these methods on the KITTI benchmark. Compared to VINS-Fusion, VINS-Fusion+dc and VINS-Fusion+bc reduce translation errors by 0.07% and 0.1%, rotation errors by 0.07 deg/100m and 0.09 deg/100m respectively. These results demonstrate that both disparity constraint and bi-circular check techniques can improve accuracy of pose estimation as they impose more restrictions on feature tracking and retain features with higher quality for pose estimation.

TABLE I. POSE ESTIMATION RMSE COMPARISON

Method	t_{rel} (%)	r_{rel} (°/100m)
VINS-Fusion	1.53	0.63
VINS-Fusion+dc	1.46	0.56
VINS-Fusion+bc	1.43	0.54
VINS-Fusion+db	1.50	0.56
Ours	1.38	0.51

t_{rel}(%): average translational RMSE drift (%) on a length of 100-800m. r_{rel}: average rotational RMSE drift (°/100m) on a length of 100-800m.

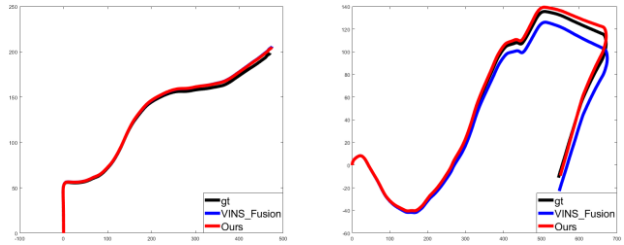
However, with the combination of disparity constraint and bi-circular check with fixed threshold (δ_d=5) to restrict feature tracking, the accuracy improvement of VINS-Fusion +db decreases. One reason is that the restrictions on features is so strict that some useful features are refused. Fig. 6 illustrates that only 130 useful features are tracked by

VINS-Fusion+db and 146 useful features are tracked by our method. Obviously, the latter is better than the former. In other words, there are more wrong rejections without adaptive feature selection in a sharp turn. When an autonomous vehicle is on the straight with a low speed, 152 features are tracked by VINS-Fusion+db and 153 features are tracked by our method, as shown in Fig.7. These two methods have similar performance on feature selection. We can see that the adaptive feature selection is able to improve tracking quality by automatically adjusting the threshold of bi-circular check. With this adaptive strategy to balance the quantity and quality of features, our method further reduces translation and rotation errors to 1.38% and 0.51 deg/100m.

Figure 6. Feature tracking in a sharp turn. Only 130 useful features are tracked by VINS-Fusion+db and 146 useful features are tracked by our method.

Figure 7. Feature tracking on the straight with a low speed. 152 features are tracked by VINS-Fusion+db and 153 features are tracked by our method.

Fig. 8 shows that the reconstructed paths from our method are closer to ground truth. In other words, our method produces more accurate pose estimation than VINS-Fusion.



(a)Sequence03 (b)Sequence 10

Figure 8. Reconstructed paths from VINS-Fusion, our method and ground truth. (a) Sequence 03. (b) Sequence 10.

D. Experiment 2:Computation Time Evaluation

The comparison of computation time is performed among VINS-Fusion and the proposed methods in this paper. The baseline is VINS-Fusion. Table II shows that our method reduces average tracking time from 38.01 ms to 21.22 ms and total time from 101.16 ms to 81.13 ms. It achieves 44% and 20% speedup over the baseline, which proves that the light and adaptive feature tracking technique has good performance. The improvement comes from two aspects. One is the effect of disparity constraint. Features unsatisfied with disparity constraint are deleted and not tracked back, which avoids unnecessary computation on reverse KLT tracking. It decreases tracking time by 15.04 ms and total time by 21.04 ms. The other is the influence of light bi-circular check. If a feature fails any check, we delete it at once instead of tracking it till the end of bi-circular check. It decreases tracking time by 16.91 ms and total time by 21.60 ms. Therefore, features with good quality and quantity are tracked for pose estimation, and algorithm efficiency is improved.

According to Table II, we can see that our method consumes a little more total time than VINS-Fusion+dc and VINS-Fusion+bc, since it requires computation on more checks. VINS-Fusion+db acquires the best efficiency because least features are used for pose estimation. With adaptive feature selection, some useful features are retained for tracking and pose estimation, resulting in the increase of total computation time in our method.

TABLE II. COMPUTATION TIME COMPARISON

Method	Tracking (ms)	Total (ms)
VINS-Fusion	38.01	101.16
VINS-Fusion+dc	22.97	80.12
VINS-Fusion+bc	21.10	79.56
VINS-Fusion+db	21.54	69.08
Ours	21.22	81.13

IV. CONCLUSIONS

This paper presents a stereo visual odometry with light and adaptive feature tracking based on VINS-Fusion, aiming to overcome the limitations of KLT tracker. First, a disparity constraint is incorporated with left-right check to refine the measurements and improve feature tracking efficiency. Next, we propose a light bi-circular check to further remove outliers which has high efficiency with the ingenious design. Additionally, in order to balance the quantity and quality of features, an adaptive strategy for feature selection is proposed by automatically adjusting the threshold of bi-circular check. With all these tools, our method obtains a relatively reliable set of measurements for pose estimation. The experiments on the KITTI odometry benchmark have shown that our method not only performs well in accuracy but also reduces 20% calculation time compared with VINS-Fusion.

Our future work will focus on trying to utilize IMU to provide an ego-motion prior for feature tracking and further improve accuracy of pose estimation.

ACKNOWLEDGMENT

This work is supported by the National Key R&D Program of China (No. 2017YFB1002800).

REFERENCES

[1] A. Howard, “Real-time stereo visual odometry for autonomous ground vehicles,” in IEEE/RSJ International Conference on Intelligent Robots & Systems, 2008.

[2] P. Hu, X. Hao, J. Li, C. Cheng, and A. Wang, “Design and implementation of binocular vision system with an adjustable baseline and high synchronization,” 06 2018, pp. 566–570.

[3] D. Nister, O. Naroditsky, and J. Bergen, “Visual odometry,” in IEEE Computer Society Conference on Computer Vision & Pattern Recognition, 2004.

[4] F. Fraundorfer and D. Scaramuzza, “Visual odometry : Part ii: Matching, robustness, optimization, and applications,” IEEE Robotics & Automation Magazine, vol. 19, no. 2, pp. 0–0, 2012.

[5] J. Deigmoeller and J. Eggert, Stereo Visual Odometry Without Temporal Filtering, 2016.

[6] S. Baker and I. Matthews, “Lucas-kanade 20 years on: A unifying framework,” International Journal of Computer Vision, vol. 56, no. 3, pp. 221–255, 2004.

[7] Z. Feng, G. Tsai, Z. Zhe, S. Liu, C. C. Chu, and H. Hu, “Trifo-VIO: Robust and efficient stereo visual inertial odometry using points and lines,” 2018.

[8] C. Tomasi and T. Kanade, “Detection and Tracking of Point Features,” Technical Report CMU-CS-91-132, Carnegie Mellon Univ., 1991.

[9] C. Troiani, A. Martinelli, C. Laugier, and D. Scaramuzza, “2-point-based outlier rejection for camera-imu systems with applications to micro aerial vehicles,” in 2014 IEEE international conference on robotics and automation (ICRA). IEEE, 2014, pp. 5530–5536.

[10] I. Cvisic and I. Petrovic, “Stereo odometry based on careful feature selection and tracking,” in European Conference on Mobile Robots, 2015.

[11] M. Fanfani, F. Bellavia, and C. Colombo, “Accurate keyframe selection and keypoint tracking for robust visual odometry,” Machine Vision & Applications, vol. 27, no. 6, pp. 1–12, 2016.

[12] M. Wu, S. K. Lam, and T. Srikanthan, “A framework for fast and robust visual odometry,” IEEE Transactions on Intelligent Transportation Systems, vol. PP, no. 99, pp. 1–16, 2017.

[13] T. Qin, J. Pan, S. Cao, S. Shen, “A General Optimization-based Framework for Local Odometry Estimation with Multiple Sensors,” arXiv:1901.03638 [cs.CV] unpublished.

[14] B. Triggs, P. F. Mclauchlan, R. I. Hartley, and A. W. Fitzgibbon, “Bundle adjustment — a modern synthesis,” in International Workshop on Vision Algorithms: Theory & Practice, 1999.

[15] A. Geiger, “Are we ready for autonomous driving? the kitti vision benchmark suite,” in IEEE Conference on Computer Vision & Pattern Recognition, 2012.

[16] J. Shi and C. Tomasi, “Good features to track,” in IEEE Conference on Computer Vision & Pattern Recognition, 2002.