

Copyright Undertaking

This thesis is protected by copyright, with all rights reserved.

By reading and using the thesis, the reader understands and agrees to the following terms:

1. The reader will abide by the rules and legal ordinances governing copyright regarding the use of the thesis.
2. The reader will use the thesis for the purpose of research or private study only and not for distribution or further reproduction or any other purpose.
3. The reader agrees to indemnify and hold the University harmless from and against any loss, damage, cost, liability or expenses arising from copyright infringement or unauthorized usage.

IMPORTANT

If you have reasons to believe that any materials in this thesis are deemed not suitable to be distributed in this form, or a copyright owner having difficulty with the material being included in our database, please contact lbsys@polyu.edu.hk providing details. The Library will look into your claim and consider taking remedial action upon receipt of the written requests.

VISUAL SMART NAVIGATION FOR UAV MISSION-ORIENTED FLIGHT

RAN DUAN

PhD

The Hong Kong Polytechnic University

2022

The Hong Kong Polytechnic University
Department of Aeronautical and Aviation Engineering

VISUAL SMART NAVIGATION FOR UAV MISSION-ORIENTED FLIGHT

RAN DUAN

A thesis submitted in partial fulfilment of the
requirements for the degree of
Doctor of Philosophy

December 2021

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

_____(Signed)

Ran Duan (Name)

To people who inspire me

ABSTRACT

This thesis addresses the accurate and robust localization problems for UAV visual navigation, including the localization of both the UAV itself and destination, during a mission-oriented flying. The self-localization process is carried out by visual odometry (VO) and we localize the destination by object tracking. Both of them are fundamental yet very challenging tasks in computer vision and robotic areas.

To achieve accurate and robust self-localization, our work investigates an inherent problem of long-term VO, i.e., why the camera pose estimation process occasionally obtains a relatively large error even when the residual of the reprojection errors are well controlled. We demonstrate that the long-term VO process suffers from the biased error distribution of estimated poses and presents a stereo orientation prior (SOP) method to perform a bias compensation in each frame. Using the stereo camera extrinsic parameters as the baseline, the SOP measures the bias of each dimension of the 6-DoF pose for every 2D-3D geometric correspondence. Unlike the commonly used error metrics that compute the total error of an inlier group, our measurement is based on the semidefinite programming of the quadratic polynomials that reformed from 2D-3D points projection system. This allows us to evaluate whether the error mainly comes from orientation or translation. Thus, the proposed system can refine the inlier group by rejecting the points with large error bias in orientation, which performs like a "soft-IMU". We show that the proposed visual odometry system achieves competitive performance in terms of accuracy and robustness even compare with the IMU-aided state-of-the-art methods.

To automatically localize the destination for the UAV, we present a deep-learning-based tracker. It rebuilds a discriminative target appearance model by selecting the representative convolutional neural network (CNN) layers and feature maps autonomously. Then a sub-network is extracted to perform the object detection for the

tracked target. To show the versatility of the proposed method, we implemented it on VGG-19 net and YOLO v3, respectively. The results demonstrate that the proposed tracker is quite competitive with the state-of-the-art CNN-based trackers in terms of accuracy, scale adaptation, robustness, and efficiency for UAV-related applications.

Finally, we integrate the visual odometry and object tracking into the UAV on-board vision system. With the stereo vision and the current UAV pose from visual odometry, the tracked targets in the 2D image can be converted to the 3D positions in the odometry local map for the UAV navigation. This allows the UAV to perform mission-oriented flying, such as object inspection or goods delivery, in full autonomy.

PUBLICATIONS ARISING FROM THE THESIS

Published:

[1] **R. Duan**, D. P. Paudel, C. Fu and P. Lu, "Stereo Orientation Prior for UAV Robust and Accurate Visual Odometry," in *IEEE/ASME Transactions on Mechatronics*, doi: 10.1109/TMECH.2022.3140923.

[2] **R. Duan**, C. Fu, K. Alexis, and E. Kayacan, "Online Recommendation-based Convolutional Features for Scale-Aware Visual Tracking," In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp.1-7, 2021.

[3] **R. Duan**, Y. Guo and P. Lu, "Object Pose Estimation for UAV Navigation Using an End-to-end Lightweight CNN," 2021 China Automation Congress (CAC), 2021, pp. 2128-2132, doi: 10.1109/CAC53003.2021.9727784.

[4] C. Fu, Z. Huang, Y. Li, **R. Duan**, and P. Lu, "Boundary Effect-Aware Visual Tracking for UAV with Online Enhanced Background Learning and Multi-Frame Consensus Verification," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4415-4422, 2019.

[5] C. Fu, Y. Zhang, Z. Huang, **R. Duan**, and Z. Xie, "Part-Based Background-Aware Tracking for UAV With Convolutional Features," in *IEEE Access*, vol. 7, pp. 79997-80010, 2019.

Under process:

[6] **R. Duan**, D. Paudel, C. Wen, P. Lu, and L. Gool, "Adjusting Camera for Every 3D Point," *IEEE Robotics and Automation Letters (RA-L)*, (under review after first revision).

Awards:

The third place in IROS 2019 Autonomous Drone Racing.

ACKNOWLEDGMENTS

My Ph.D. application has followed a long and tortuous course. In 2018, the uncertain visa policy of the US collided with my homesickness, which forced me to quit the previous position. I sincerely thank Dr. Peng Lu who understood my situation and offered me an opportunity to restart Ph.D. study at PolyU. After he changed his employment, I joined Prof. Chih-Yung Wen's group – a passionate and professional UAV team in PolyU – where I could continue my research. I would like to express my immense gratitude to my supervisors, who provided insight and expertise that guided me all the time of research and paper writing. Throughout my Ph.D. study, I have benefited enormously from their active yet rigorous attitude and critical thinking towards research problems. While the road to being a qualified scholar remains bumpy, their guidance and encouragement are the best compass and energy for me to move forward. It is my great honor to have their superior research leadership.

I would like to extend my thanks to my research collaborators, Dr. Danda Pani Paudel, Prof. Changhong Fu, Prof. Erdal Kayacan, Prof. Kostas Alexis, and Prof. Luc Van Gool, for their great help in paper writing. Particularly, I wish to thank Dr. Paudel and Prof. Fu, who were my team leaders at the University of Strasbourg in France and NTU in Singapore, respectively, for their continuous guidance in my academic life. I would also like to thank all students and staff at the Department of Aeronautical and Aviation Engineering for their kind assistance in my day-to-day work at PolyU.

Last but not the least, I want to show my deep appreciation to my family for their love and support. Due to the quarantine of the Covid-19 pandemic, I was unable to meet my grandma before she departed at the end of 2020. The same thing happened when my grandfather in law passed away at the end of 2021, three months before my

graduation. I miss them very much. We must win the battle against the Covid-19 and I wish the world could get back on track soon.

TABLE OF CONTENTS

	Page
ABSTRACT	iv
LIST OF TABLES	xii
LIST OF FIGURES	xiv
ABBREVIATIONS	xxi
1 Introduction	1
1.1 GNSS-bypass navigation for UAV	3
1.2 Onboard system design	4
1.2.1 Visual odometry	5
1.2.2 Tightly Coupled vs Loosely Coupled	6
1.2.3 Research Motivation	7
1.3 Object tracking	8
1.4 Preliminary Validation of Proposed System	10
1.5 Contributions	11
1.6 Organization of the thesis	12
2 Literature Review	13
2.1 UAV visual navigation	13
2.2 UAV object tracking	18
3 Camera Pose Estimation and Minimal Adjustment	20
3.1 Problem statement	20
3.2 Notation list	22
3.3 Preliminary	23
3.4 Problem Formulation	25
3.4.1 Adjusting Cameras for Resectioning	29
3.4.2 Rejection of Outliers for Resectioning	31

	Page
3.5 Two-view Triangulation	32
3.6 Long-Term Localization	33
3.6.1 Keyframe Processing	33
3.6.2 Non-keyframe Processing	34
3.6.3 The Algorithm	34
3.7 Experiments	35
3.7.1 CamAdj for Resectioning	36
3.7.2 CamAdj for Two-view	41
3.7.3 Long Term Localization	42
4 Stereo Orientation Prior for Visual Odometry	45
4.1 Notation list	45
4.2 Methodology	46
4.2.1 Stereo Orientation Prior	48
4.3 Experiments	55
4.3.1 Benchmark Evaluation	56
4.3.2 Contribution of Stereo Orientation Prior	58
4.3.3 UAV Indoor Flight Test	62
4.4 Different between SOPVO and CamAdj	67
5 Visual Object Tracking for Task-oriented Flying	69
5.1 Proposed Algorithm	70
5.1.1 Recommender	71
5.1.2 Target Appearance Modeling	73
5.1.3 Correlation Filters	74
5.1.4 Min-channel Scale Learning	76
5.1.5 Tracking Framework	77
5.2 Experiments	79
5.2.1 Overall Performance	80
5.2.2 Results Analysis	82

	Page
5.2.3 Algorithm Complexity	83
5.2.4 Evaluation By Attributes	83
5.3 UAV onboard test	84
6 Conclusion and Future Work	87
6.1 Conclusion	87
6.1.1 System Robustness in Harsh Environments	88
6.2 Future Works	89
6.2.1 Integration of onboard vision system	89
6.2.2 Loop-closure detection	89
6.2.3 End-to-end 3D pose estimation	90
REFERENCES	92
VITA	100

LIST OF TABLES

Table	Page
3.1 Average rejection percentage of each data group with different bounds (higher is better). The results indicate that the CamAdj filtering has different rejection performances in different types of errors. The reported experiments are conducted after the reprojection error-based filtration technique. Among the three different sets (of various noise levels), the proposed CamAdj can well identify correspondences from the highest noise level. This in turn means, correspondences from noisy pose set when identified and filtered (which cannot be done any further using reprojection error) can lead to better pose estimation.	38
3.2 Pose estimation for the third view evaluated on ETH3D. We conduct experiments illustrated in Fig. 3.4 for every 3 cameras in the listed datasets. In order to show the impact of CamAdj filtration, we tune the CamAdj bounds iteratively until the CamAdj reject 1 to 25 % points (average is given by Rejection rate). The test of each group has been repeated 50 times. Better R/t rate indicates the percentage of the tests when CamAdj filtration generates a better pose than MSAC.	39
3.3 Two-view reconstruction. The 3D reconstruction errors show a better quality of triangulation after CamAdj filtration. γ & γ' are the depth bounds (refer Corollary 3.5.2)	41
3.4 Trajectory RMSE comparison on EuRoC benchmark evaluation. The evaluation metric and results of the compared mono-VIO methods are adopted from [91], while our method runs in pure stereo-VO model. Here ‘x’ implies the failure. This comparison aims to show the accuracy level of the proposed VO. However, the computational efficiency of CamAdj is very low because all correspondences are evaluated independently. It is hard to achieve real-time processing.	43
3.5 Results corresponding to Fig.7. We sync the keyframe update with BA for our method, which yield slightly different results with free run in TABLE 4.1.44	

Table	Page
4.1 EuRoC MAV visual odometry benchmark evaluation. The results that achieved the best three levels of accuracy are marked by red , green , and blue . Except for algorithm termination (marked by \times), the $\text{RMSE} > 1$ was also considered as a fail. *The results of SVOMSF, MSCKF, ROVIO, VINS-MONO, SVOGTSAM, and OKVIS were obtained from [80] (NUC results). Other methods were evaluated using the same metric, i.e., real-time running on NUC ROS Kinetic, no loop closure detection nor bundle adjustment. The final results of each dataset were the averages of 10 runs. [†] ROS Kinetic uses OpenCV 3 by default. In OpenCV 4 the ORB detector is modified and it generates slightly different results using the same parameters.	57
4.2 Average RMSE evaluated by VICON.	65
5.1 TLE results of each dataset and average FPS: top 3 ranked results on each dataset are marked by red, green, and blue, respectively.	81
5.2 Mean CLE by attributes: top 3 ranked results on each attribute are marked by red, green, and blue, respectively.	84

LIST OF FIGURES

Figure	Page
1.1 An example of UAV mission-oriented navigation in GNSS-denied environments: survivors inspection inside the mine. The local position of the UAV is obtained by visual odometry. The object tracking provides the location of a survivor for UAV inspection maneuver. *The object tracking results are obtained from our visual tracking work, which is presented in Section 5. The local position graph is faked for illustration purposes only. .	1
1.2 UAV visual navigation platform design.	2
1.3 UAV visual navigation system for mission-oriented flight.	4
1.4 Proposed tracker rebuilds the target appearance model using only a few layers of the backbone CNN of YOLO v3.	10
1.5 Our UAV configuration for IROS 2019 drone racing competition: T265 provide the visual odometry measurement for UAV navigation, while the RGB camera detects the highlighted gate that the UAV should pass. . . .	10
2.1 Visual Simultaneous Localization And Mapping	13
3.1 Pose estimation uncertainty using reprojection error (left) and the pose bounds (right). Each point represents one independent estimation of the camera pose. The pose estimations show large variations of errors using the same reprojection error threshold. This is because we solve the pose by the regression process that assumes the errors of the inliers are Normal distributed with $N(\mu, \sigma^2)$ around the groundtruth, while it is not always true. The pose bounds control the σ in different dimensions of the error space to ensure the uncertainty of each estimation. Data source: experiments of Fig. 3.5.	22
3.2 The setup for the general SfM problem, where P_0 represents the initial pose of the reference frame and \hat{P}_k represents the estimation of an unknown P_k or an adjusted pose of a known P_k	23
3.3 The system overview of stereo visual odometry. The insufficient points will activate the reset of keyframe.	34

Figure	Page
3.4 We estimate the unknown pose of the third view using the known poses of cameras k and $k + 1$ (which can be seen as calibrated stereo). The 3D points are filtered by CamAdj, which results in a better pose estimation for the third view.	35
3.5 Reprojection error metrics and pose rotation error with CamAdj and without. In each experiment, we plot the mean errors of the ρ_a and ρ_{g2} measure of overall correspondences for each independent camera pose estimation (using ρ_{g2}) and the norm of the rotation error compared to groundtruth. The results show nontrivial fluctuations of rotation error refer to the measurements of these error metrics. The results also demonstrate that the abnormal errors can be removed by CamAdj. The distribution of the pose errors of all estimations is shown in Fig. 3.1.	36
3.6 Illustration of the singular point hypothesis.	37
3.7 Rotation estimation error AUC for the third view in the three-view configuration and two-view triangulation with unknown depths on ETH3D dataset <i>courtyard</i> . The maximum rotation error thresholds in Cayley were set to 0.0087 (1° in Euler). The results indicate that CamAdj could yield better rotation estimation overall.	40
3.8 3D points of different sequences; ground truth (colored), reconstructed by MSAC (red) and CamAdj (green).	42
3.9 Comparison of VO trajectories' absolute RMSE with and without CamAdj filtering (left). An example of the complete motion trajectory of the same settings (right).	42
3.10 Absolute rotation error in Euler angle (0 to π), in the case of long-term localization.	43

4.1	Our contributions: We visualize one dimension (yaw angle alone) of the polynomial system of the 2D-3D correspondences to illustrate the fundamental theory in geometric representation. Because of image noise and point tracking uncertainty, the zero residual of an inlier polynomial does not always appear at the groundtruth. The optimal solution to the polynomial system (usually computed using the least square method) may have a significant bias in terms of orientation. Our method employs SOS to assess data bias, with the stereo camera serving as a bias measurement baseline, much like base stations in differential GPS techniques. Unfortunately, eliminating bias in both orientation and translation may result in a scarcity of inliers. As a result, our method rejects orientation-bias outliers because they have a greater impact on odometry. In subsequent frames, the bias-compensated group of filtered 3D points can be used to regress robust and accurate poses.	47
4.2	System overview	48
4.3	The main idea of SOP is simplified in 2D for illustration. Two different 3D points \mathbf{P}^{pre} and \mathbf{P}^{cur} represent the same landmark \mathbf{P}_k from old and current frames, respectively. They are updated to the \mathbf{P}_k and evaluated by SOS feasibility. If the updated point does not result in the estimation within the given feasible bounds α and β , it is considered an outlier. Thus, for each point, the SOP ensures that it contains at least one solution in the given bounds. Statistically, the uncertainty of refined inliers is controlled by $[\alpha, \beta]$ and the bias in orientation is diminished as shown.	50
4.4	Stereo orientation prior based outlier rejection. The error power map is simplified in 2D (1D rotation and 1D translation) for illustration. Optimistic and pessimistic reprojection error thresholds preliminarily identify the points that are definitely inliers or outliers, while the remaining points are regarded as potential inliers and evaluated by stereo orientation prior. .	55
4.5	Trajectory comparison results sampled from EuRoC benchmark evaluation (Tab. 4.1).	56
4.6	CPU usage and FPS on Intel NUC5i7RYH. The results of all compared methods are obtained from [80].	58

Figure	Page
4.7 The absolute value of the yaw Euler angle in comparison with the ground truth. The orientation estimation error comparison using SOP and only reprojection error (RPE) for outlier rejection. The results were filtered by a median filter because the ground truth data given by VICON contained some abnormal peaks. The results indicate that the proposed method provided a better orientation estimation, especially when keypoint detection and tracking became difficult. We compared SOP with $RPE = 2$ because this threshold generated the best odometry estimation over other thresholds (refer to Fig. 4.8). The angle MSE (given in the legend) was computed from frame 2,000 to the end to avoid the impact of the value jump between $-\pi$ and π	60
4.8 The ATE performances on the EuRoC benchmark using SOP and only RPE. While the outlier rejection using only RPE achieved its best performance by setting the threshold as the typical image noise level (2-pixel distance), the stereo orientation prior reduced its error by 18.5%.	61
4.9 Reprojection error (RPE) vs stereo orientation prior (SOP): three different reprojection error thresholds were used to reject the outliers: pessimistic, optimistic, and (pessimistic + optimistic)/2. SOP evaluated the points between pessimistic and optimistic. For SOP, the boundary conditions were fixed as $\alpha = 0.1$ and $\beta = 0.5$. The results indicated that, regardless of how we changed the RPE, the orientation estimation was not as good as that using SOP.	62
4.10 Orientation or translation prior: we adjusted the boundary condition to further evaluate whether a small orientation boundary was the main contributor, as we analyzed. In the first row, we changed the orientation boundary α while fixing the translation boundary β , and vice versa for the second row. It is clear that when we reduced the orientation boundary, the accuracy of orientation estimation increased, resulting in improvement in the overall odometry estimation. The translation boundary did not have the same level of impact on the results.	63
4.11 Testing platforms. Left: SK520 Quadcopter with NUC; Right: QAV250 FPV Quadcopter with UP Board.	64
4.12 The fisheye view converted into undistorted images. The left figure shows the T265 fisheye view from our SK520 drone. Because the fisheye has more than 160 degrees of FOV with 848×800 resolution, some parts of the UAV, such as propellers, occupied a large area of view. The right figure shows the grid feature detection of the proposed method using the cropped and resized image.	64

Figure	Page
4.13 VICON flight test. The F250 drone flew by VICON, while the proposed method was run onboard (Intel UP Board) at an average of 12 FPS.	65
4.14 Carpark test: The SK520 drone was handheld for safety reasons, while the proposed method ran online on its onboard computer (NUC5i7RYH). The route started from a point approximately 11 m away in the front of the first corner, where we aligned the speed bump with the center of the image view. The estimated trajectory was plotted in the ROS RVIZ, where the size of each grid was 1 m. The right side shows the final results of the proposed method and the estimation from Intel T265. The camera glare effect due to the light sources might be the reason that the Intel T265 underestimated the distance. Considering the measurement error of the ground truth ($\pm 0.5m$), the drift of the proposed method was less than 0.5 m over the 116 m route ($< 0.4\%$).	66
4.15 Corridor test: The bottom-left shows the planned route. We marked the corner points of the route in real world as groundtruth because without motion capture system, we can only ensure the accurate measurements when reaching these markers during walking. The estimation errors were measured by the drifting distances of these corners. The average drift was within 20 cm ($< 5\%$ of overall distances). The right plot visualizes the estimated path and all tracked landmarks over the estimation history. . . .	67
4.16 Fundamentals of CamAdj and SOPVO.	68
5.1 Online CNN layer and feature map recommendation. The layer, in which the highest convolutional responses gather at target center, is marked by a red box. The target percept reconstructed from the recommended feature maps, on the other hand, has a better description regarding the target's central location, shape, and size.	69
5.2 Tracking framework overview	70
5.3 Diagram of DT and GT. DT aims to select the discriminative features learned by CNN. As the new figure shows, for the convolution results, the peak values with a large distribution is designed to result in a low score. DT provides a convergent non-linear weight distribution refer to the $2d_i/(1 + \beta)r$ ratio as well as labels foreground/background by the positive/negative score. GT, on the other hand, amplify the discriminative score by comparing the power of foreground and background.	72

Figure	Page	
5.4	Recommender output. The input in (a) shows the foreground (F) and background (B) regions. In (b), layer Conv4-4 has multiple dispersed peak convolutional responses, while in (c), the peak responses gather to the center. Therefore, compare to (b), the kernel learned in (c) is more likely to be the discriminative detector of the target. A sample of feature maps is shown in (d), of which the discriminative term DT is a negative value because it only represents background features. The gain term GT, on the other hand, measures the difference level between F and B.	72
5.5	Recommendation framework: The proposed recommender selects layer C^{20} weights all feature maps in layer C^{20} . Only the feature maps with the positive weights are used to rebuild the target model. To be noted that the ReLU layers are taken into account, therefore there are 37 layers in VGG-19. Hence, we extract the sub-CNN with only 20 layers for this tracking task, which optimized the convolutional feature extraction process.	73
5.6	With (our: red bounding box) or without (HCF: green bounding box) feature selection by the proposed recommender. In order to do a fair comparison, we disabled the scale adaptation of our method. In this example, the background scene contains more features than the tracked object. As the CNNs simply detect any learned features, without the recommendation of the feature maps, the background features may dominate the target appearance model, resulting in the drifting of tracking.	75
5.7	Spatiotemporal-based min-channel scale: the first block on the left describes the main idea of min-channel. The highest responses of all recommended feature maps are projecting into the min-channel map, in which we only take the pessimistic of target region into consideration. Since the highest response of feature maps are the location of target features, the variation of power distribution of the min-channel maps are used for target scale change estimation.	77
5.8	Location precision plot (CLE: pixel-based distance) of OPE using AUC and the merged plot of mean CLE and SR. This figure shows the overall performance of trackers in terms of tracking accuracy and robustness. The proposed tracker achieves the highest precision and the best average CLE (10.3).	80
5.9	Tracking results on datasets (From top to bottom): Dog1, Human5, MotorRolling, Human6.	82
5.10	Precision plot of each attribute using AUC: the proposed tracker ranked No.1 on 9 attributes (IV, OCC, DEF, MB, FM, IPR, OPR, OV, and BC) out of 11.	84

Figure	Page
5.11 During the flight, image blur and abrupt motion happens occasionally. The illuminance and target scale also variate over time.	85
5.12 UAV onboard test.	86
6.1 The comprehensive visual navigation system for task-oriented flight.	89
6.2 Recurrent image retrieval for loop-closure detection.	90
6.3 End-to-end lightweight CNN for pose estimation.	91

ABBREVIATIONS

ATE	Absolute trajectory error
BA	Bundle adjustment
CNN	Convolutional neural network
DoF	Degree of freedom
EKF	Extended Kalman filter
GNSS	Global navigation satellite system
ICP	Iterative closest point
IMU	Inertial measurement unit
KLT	Kanade-Lucas-Tomasi point tracker
MAV	Micro Aerial Vehicle
MEMS	Micro-electro-mechanical systems
ORB	Oriented FAST and rotated BRIEF
PnP	Perspective-n-point
QP	Quadratic Programming
RANSAC	Random sample Consensus
RMSE	Root mean square error
ROS	Robot operating system
SfM	Structure from motion
SLAM	Simultaneous localization and mapping
ToF	Time of flight
UAV	Unmanned Aerial Vehicle
VSLAM	Visual simultaneous localization and mapping

1. INTRODUCTION

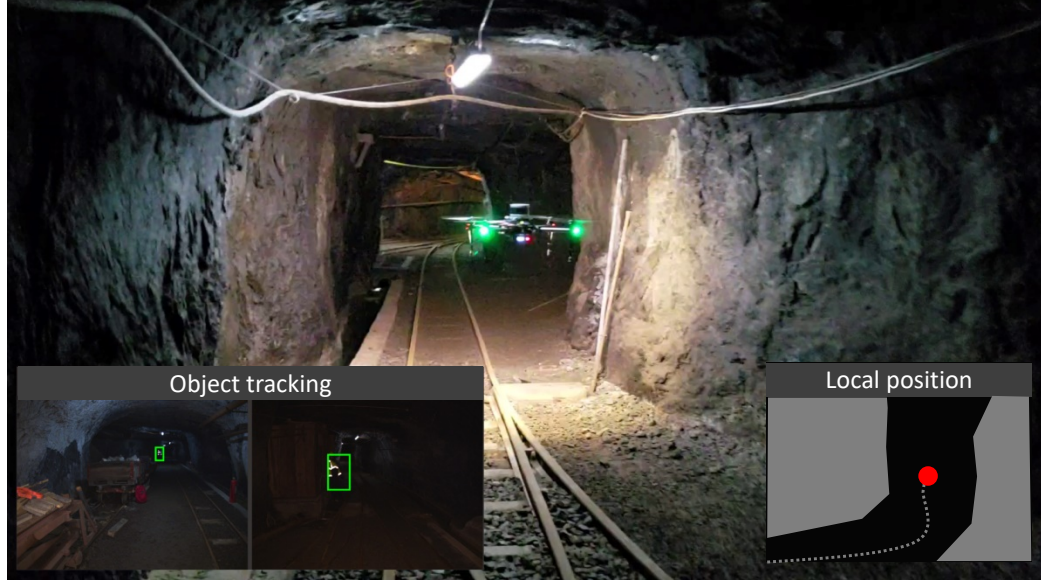


Fig. 1.1.: An example of UAV mission-oriented navigation in GNSS-denied environments: survivors inspection inside the mine. The local position of the UAV is obtained by visual odometry. The object tracking provides the location of a survivor for UAV inspection maneuver. *The object tracking results are obtained from our visual tracking work, which is presented in Section 5. The local position graph is faked for illustration purposes only.

Many UAV applications involve mission-oriented navigation, such as following a person or landing on a moving target. In these kinds of scenarios, two fundamental problems need to be solved: 1) what is the current location of the UAV and 2) where is the destination. When the visual navigation system is employed for the flight in GNSS-denied environments, the aforementioned two problems are correlated to visual odometry and object tracking, respectively. Visual odometry estimates the camera

6-DoF poses from 2D image frames and transforms it to the UAV local positions. On the other hand, object tracking locates the target in the 2D image views, which can be converted to the 3D locations in the odometry map with the known depths and camera absolute poses. An example is given in Fig. 1.1.

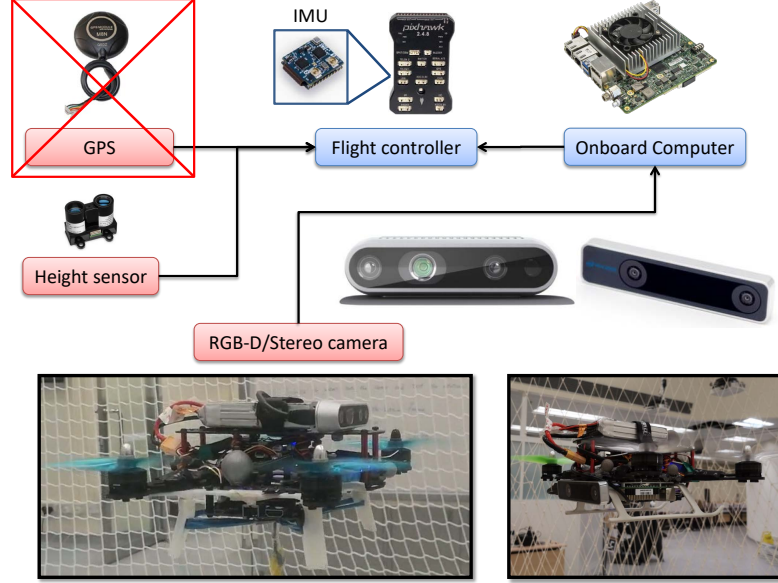


Fig. 1.2.: UAV visual navigation platform design.

This thesis addresses these challenges and presents an integrated onboard vision system with the robot self-localization and target tracking for the UAV to safely fly in GPS-denied environments. The platform design is shown in Fig. 1.2. It consists of RGB-D or stereo camera, height measurement sensor, and IMU. As the height measurement and IMU data will be directly processed by the flight controller, our research will focus on the visual input. The selection of a vision system has taken many factors into accounts, such as weight, size, cost, and application scenarios. Activate light sensors such as RGB-D or ToF cameras are widely used in indoor visual navigation tasks. However, their ranges are usually within several meters. The 3D scene measurement using LIDARs, on the other hand, is usually a heavy and high power consumption device for the UAV platform. Furthermore, it is expensive to use

an accurate LIDAR for done, as the typical price is around 50k USD dollar. Therefore, we select the stereo configuration for UAV visual perception, which measures the 3D scenes purely from vision. The visual navigation involves two major research fields: 1) UAV stereo VO, which is an on-board local positioning system that extracts the UAV motion from visual perception, this technique can also be used to reconstruct the 3D map of surroundings; 2) object tracking, which extends the navigation capability in flying task so that the UAV can locate the destination online and plan the flight route. Visual odometry, or the frontend of the VSLAM, provides real-time control feedback to the flight controller, while the backend aims to refine the past results. In other words, the frontend is the key to agile the UAV reaction during navigation.

In this chapter, we discuss the research background, challenges, motivation, and contributions in the following sections.

1.1 GNSS-bypass navigation for UAV

GNSS-bypass navigation is a critical function for a UAV (multi-rotor drone). As the GNSS suffers from the atmospheric effects, multipath effects, and many other environmental factors that contaminate the signal, the position we obtained could have more than 10 meters drift. Furthermore, there are many places that the GNSS signal is blocked. Thus, we are unable to safely fly the UAVs in cloudy weather, forest, urban area, or indoor places. To perform the GNSS-bypass flying, no matter stabilize or follow a pre-defined route, the pose (orientation and location) of the UAV is essential feedback for the flight controller. In recent years, computer vision technology has progressed dramatically and visual navigation has become a popular research topic. However, the progress has been much slower in moving those results to UAV autonomous flight. Because recovering the UAV pose state from vision only is rather difficult. Even the human pilots could get lost and result in dangerous action without reading the aircraft pose from instruments, while they indeed have a better

visual understanding of surroundings than machines. How to fly a UAV with an on-board vision system becomes the biggest challenge in many applications.

1.2 Onboard system design

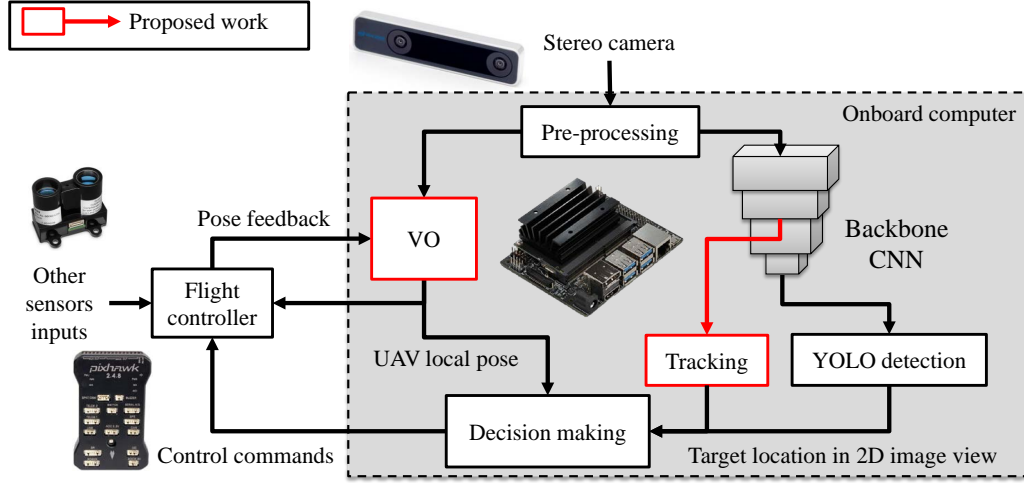


Fig. 1.3.: UAV visual navigation system for mission-oriented flight.

We show an onboard system architecture for UAV mission-oriented navigation in Fig. 1.3, where the proposed research work is marked by red color. In the beginning, the system obtains the raw data from the stereo camera and carries out some standard image processing, such as image reshaping, rectification, filtering, brightness, and contrast enhancement, etc, dependent upon the input requirement of the following processes. The preprocessed image frames are subsequently used for UAV's local pose estimation. Concurrently, the UAV finds its destination by the proposed tracker, which consists of a pre-trained backbone CNN and a tracking component. If the target is a pre-defined object and trained by YOLO, the tracker can be initialized by YOLO detection. The proposed tracker is also capable of learning an untrained target online, where the initial region is required. The motivation for designing this kind of workflow is that the UAV onboard computer cannot afford the real-time detection ($> 10\text{FPS}$) using full YOLO detection. The YOLO detection is

fundamentally a regression network that recognizes a large number of objects and the backbone CNN is acting as an image feature extractor. But object tracking does not require classification. In other words, using a detection network to perform tracking tasks produces a lot of redundant computation. Moreover, the tracked target may not be included in the training datasets. The proposed tracker automatically selects the CNN features and rebuilds the target model efficiently using fewer layers for visual tracking. This allows the UAV to save more power to process other tasks such as VO and decision making. The image features can also be used for the loop-closure process, which is included in our future work. With the UAV pose in a local map and target location in the 2D image view, the decision-making module can generate control commands based on the pre-programmed strategies. The decision-making system involves many other research topics such as visual servoing, path planning, obstacle avoidance, etc., dependent upon the flying task. Our work aims to provide accurate and robust real-time visual perception for the decision-making system.

1.2.1 Visual odometry

In visual navigation, reconstructing 3D scenes from 2D image views or determining the camera poses refer to the known scenes are the forward and backward processes of multi-view geometry. Their simultaneous process, termed SfM [1], is widely used in feature-based VO or VSLAM. Although has been studied for decades and much progress has been made, the SfM remains a very challenging problem in the robotic area because in practice we always get contaminated data. From analog to the digital world, the hardware system and algorithms inevitably introduce noise, even errors, into the raw data. For instance, in feature-based VO workflow, where the stereo pairs for triangulation are obtained by feature point matching, the accuracy of feature detection and matching algorithms significantly impact the final estimation results. However, most of the existing methods, e.g., FAST [2], ORB [3], etc., contain pixel-level location noise and mismatched pairs. Although the optimization methods

can minimize the overall estimation error in most cases, the cumulative error and occasional wrong estimation still result in poor navigation. At the top system level, IMU-aided sensor fusion in conjunction with EKF could be the solution of choice. With regard to the pure vision system, it is rather difficult for the system itself to correct the cumulative errors. The error mainly comes from the process of recovering the 3D scene structure and camera motion, where recovering the camera poses is of prime interest.

1.2.2 Tightly Coupled vs Loosely Coupled

The navigation systems can be separated into two loosely-coupled and tightly-coupled approaches from the aspect of system architecture, based on directly or indirectly fused measurements from sensors. From algorithm point of view, tightly-coupled techniques are generally more accurate and robust than loosely-coupled approaches. For instance, the tightly-coupled VIO methods are widely used in research and commercial UAVs, as well as in augmented reality applications.

However, not all UAV navigation systems employ the tightly coupled VIO. The loosely-coupled system is more popular for system modular design. In some multi-sensor-fusion systems, each sensor separately estimated its states and the fusion only happens at final stage. Which means the vision pose estimation is still implemented as an independent subsystem, whereas the sensor fusion is processed at the upper level. Thus the loosely coupling is flexible and tends to be more efficient during the customization and iterative development. For example, when new sensors or algorithms are available and the developers want to upgrade the relative parts of the system, the loosely coupled configuration allows developers to simply swap out both software and hardware components. Also, as the demand for more functions grows, the loosely coupled system is more scalable and compared to the tightly coupled one. The researcher can easily add new modules to the system without affecting the existing functionality.

In conclusion, loose coupling provides more flexibility, scalability, adaptability, efficiency, and innovation for system integration. This is simply the fact that loose coupling reduces dependencies between components, isolating the impact of changes to any given component. Therefore, we focus on developing a pure visual odometry algorithm for a loosely coupled system so that other people can build their own systems based on it.

1.2.3 Research Motivation

Without IMU correction, the cumulative error becomes significant, and the common solution is BA. However, BA reduces the effect of cumulative error averages the bias over a set of frames. This process is a delayed correction that may cause unexpected problems in UAV control.

Our research investigates an alternative method for reducing the cumulative error in each individual frame. We address the inherent problem of the pure VO method, namely, why does the camera pose estimation process occasionally obtain relatively large errors even when the residual reprojection error is well controlled. We find that the inlier group can exhibit a major bias on the orientation, which amplifies the odometry error over time. Furthermore, from the state-of-the-art VIO methods, it is clear that the IMU can significantly improve the VO performance by providing a prior estimation of orientation. It is evident that an accurate orientation estimation is a priority. We propose a novel visual odometry method that extracts prior knowledge of orientation estimation from a stereo view. In [4, 5], a sum-of-square (SOS) technique is proposed for searching the global optimal solution of camera pose estimation. The SOS feasibility could exam if a quadratic polynomial contains at least one solution within a given boundary. Then the global optimal solution is determined by branch and bound searching span the possible 6-DoF space. Inspired by them, we redefine the inliers using SOS feasibility. The orientation boundary condition extracted from the stereo setting is used to filter out the disqualified 3D points during frame-to-frame

landmark updates. The filtering is carried out by localizing the second camera with respect to the 3D points given in the first camera. Our experiments show that this filtering is critical, which gives us better control to gauge the estimated poses, unlike using the reprojection error measure for the same poses. Furthermore, our formulation also allows us to set a smaller tolerance for orientation and a larger tolerance for translation independently. This helps the system trade off the translation error with orientation error while keeping enough inliers. The goal of doing this is to remove the bias on orientation estimation, where the SOS technique with the known extrinsic parameters of the stereo camera makes the bias on each dimension of a 6-DoF pose measurable. Our experiments show that such a choice generally achieves VIO-level accuracy without IMU.

In conclusion, our work offers advantages in terms of enhancement in the orientation correction of odometry estimation using pure-vision information. From the research aspect, we demonstrate that the controlled reprojection residual does not guarantee the accurate estimation of a camera pose because of the statistical error bias. Thus, we propose a new outlier rejection method to reduce the bias of the orientation estimation. From the application perspective, the proposed method can enhance the fault tolerance of a sensor fusion system, which is critical in safety-critical applications such as UAV navigation.

1.3 Object tracking

Visual tracking, one of the fundamental problems in computer vision, has been widely used in numerous vision-based UAV applications [6–9]. Although being investigated for decades and much progress in terms of tracking accuracy and robustness has been made [10–17], object tracking still remains a challenging problem due to many uncertainty factors, such as appearance variation, occlusion, background clutter. On the other hand, deep learning methods are good options to address this kind of uncertainty problem. Recently, convolutional neural networks (CNNs) have

shown better performance in terms of accuracy and robustness for visual tracking task compared to state-of-the-art approaches [11, 12, 14, 15, 18–23].

Two-dimensional CNNs have exhibited outstanding performance in object recognition problems, for instance, YOLO [24], SSD [25], Faster RCNN [26], and Mask-RCNN [27]. In the backbone layers of those CNNs, an object can be represented by different levels of percepts, e.g., different complexity of the feature semantic combinations. Thus, a featureless object may have a better representation using low-level percepts while a complex object requires high-level percepts that contain highly discriminative information. This phenomenon is commonly referred to as the semantic gap. However, existing CNNs-based visual trackers use only one or several pre-selected layers [18–20, 28]. Furthermore, each feature map from hierarchical layers indicates the level of similarity of a specific type of feature throughout the whole image using image convolution. Therefore, taking all feature maps from a hierarchical layer is not a reasonable way to use CNN for object tracking since some of the features do not belong to the object. Another main challenge for CNN-based approaches is the scale variation. Since CNNs treat each frame as an independent image, continuity of object scale change is ignored. As the results, the CNN-based methods still have to use traditional methods for scale estimation of an untrained target. For instance, [18] solve the scale factor estimation in its updated version by extracting HoG feature in addition to CNN features.

In our research, the semantic gap issue mentioned above is addressed using a recommender for layer selection and updating appearance model using recommended feature maps. Hence the proposed method does not need the entire network in most small or featureless target tracking cases. A demonstration is shown in Fig. 1.4, the proposed system uses only the first 14 layers (YOLO v3 contains 106 layers in total) to locate the target. Those efforts aim to reduce the work load for UAV onboard computer. The scale variation problem is solved by learning object scale directly from CNN features via a spatiotemporal-based approach. In addition, the proposed scale learning framework also measures the certainty of tracking. Thus, the searching

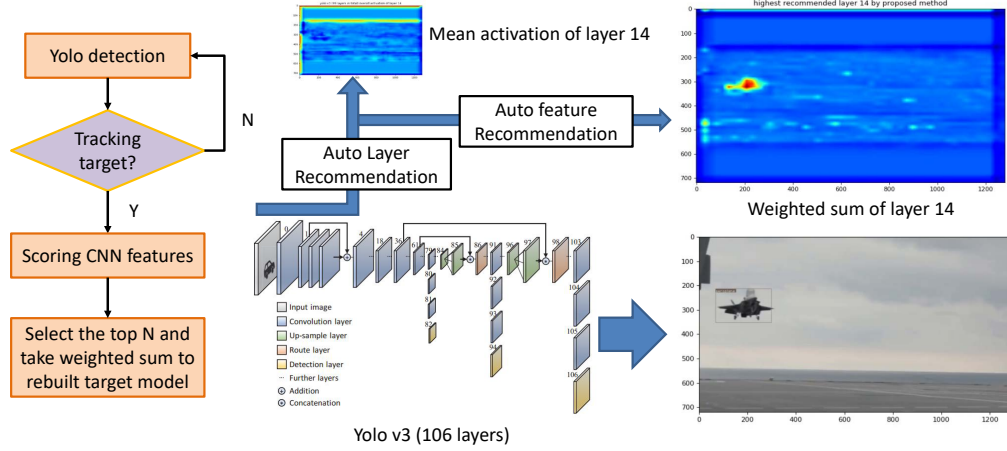


Fig. 1.4.: Proposed tracker rebuilds the target appearance model using only a few layers of the backbone CNN of YOLO v3.

region grows together with the increasing of tracking uncertainty in the presence of target lost due to fast or abrupt motion of the drone platform.

1.4 Preliminary Validation of Proposed System

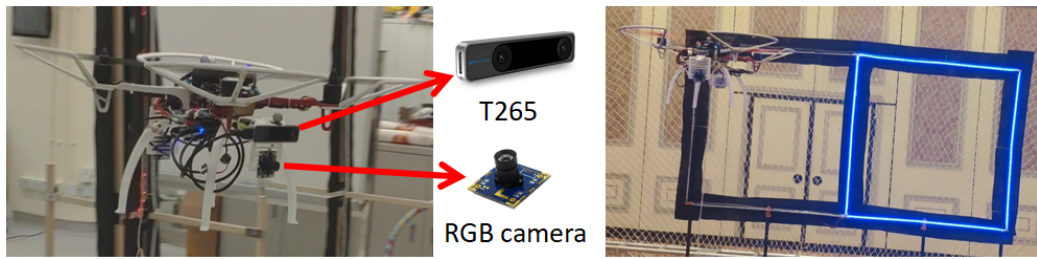


Fig. 1.5.: Our UAV configuration for IROS 2019 drone racing competition: T265 provide the visual odometry measurement for UAV navigation, while the RGB camera detects the highlighted gate that the UAV should pass.

To preliminarily validate the feasibility of the proposed system architecture, we choose a simple yet typical scenario of mission-oriented flying: passing a gate. We replace the VO algorithm module with the VIO of an Intel RealSense T265, which is

a commercial navigation product. The object tracking module is simplified as a gate tracker. We mount the T265 on the F330 UAV as Fig. 1.5 shows.

The Intel RealSense Tracking Camera T265 is a small, low-cost and light-weight navigation sensor. A computing unit is embedded in the T265 sensor to work out its motion from the stereo camera and IMU data. With fisheye lenses, a stereo camera covers more than 150° field of view. In addition, two RGB cameras (front/back-looking) are mounted for gate detection. The integrated system shows the modules are compatible and the overall system can carry out the mission efficiently. With this configuration, we won third place in the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2019) drone racing competition.

1.5 Contributions

Our main contributions can be summarized as follows:

- Demonstration of the minimal adjustment approach that results in more accurate camera pose estimation over the golden roles in structure from motion problem.

Github: <https://github.com/rduan036/CamAdj>

- A novel visual odometry framework that uses a sum-of-square-based approach to remove the orientation bias from a low-reprojection-error-inlier group. When compared with traditional approaches, the proposed process significantly reduces the overall error in pose estimation.

Github: <https://github.com/arclab-hku/SOPVO>

- A novel recommender-based tracker that is capable of selecting the representative CNN layers and feature maps autonomously, which allows the tracker to rebuild the appearance model of any untrained target and to simplify the network.

Github: <https://github.com/arclab-hku/ICRA2021tracking>

- Real-time implementation of the proposed algorithms on Both Matlab and ROS.
Code and data are released as open-source online with Gazebo simulation tools.
Demo video with Github link: <https://youtu.be/s2KD02EqbNQ>

1.6 Organization of the thesis

The rest chapters are list as follows:

- Chapter 2: literature review in terms of UAV visual navigation and object tracking.
- Chapter 3: introduce the research in camera pose estimation.
- Chapter 4: introduce the proposed visual odometry algorithm.
- Chapter 5: introduce the proposed visual tracking algorithm.
- Chapter 6: conclusion and future works.

2. LITERATURE REVIEW

2.1 UAV visual navigation

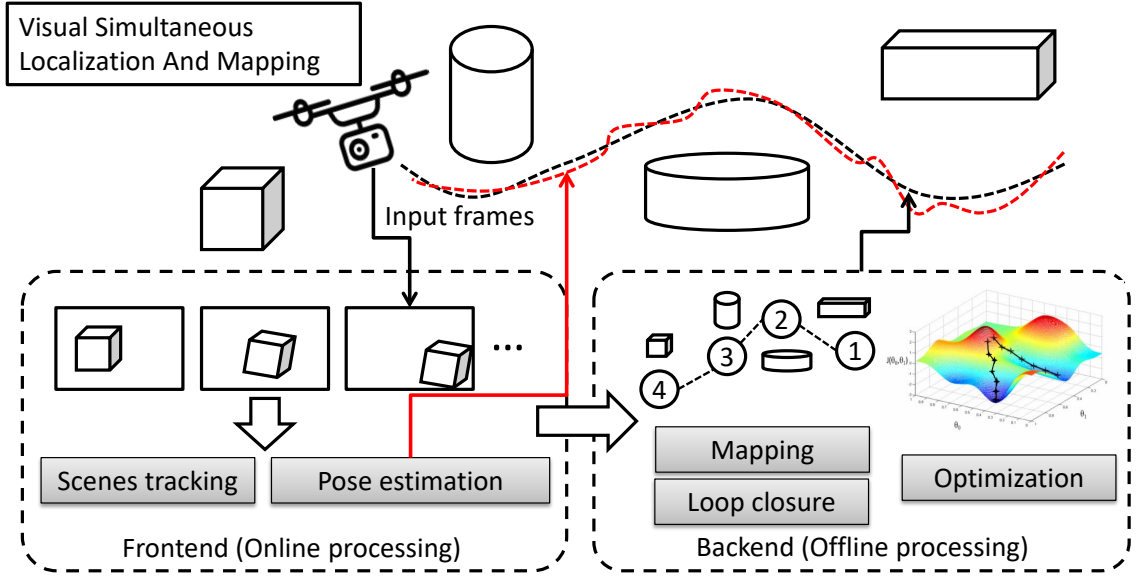


Fig. 2.1.: Visual Simultaneous Localization And Mapping

Visual navigation is an active research topic in the field of computer vision and robotics, especially for GNSS-denied environments like urban areas, indoor rooms, or underground spaces. The most common solution to the UAV on-board visual navigation system is the visual simultaneous localization and mapping (VSLAM). We illustrate general VSLAM system in Fig. 2.1. There are two major modules: frontend and backend. There are two problems to be solved in the frontend: reconstructing 3D scenes and estimating the camera pose in the current frame. However, they are dependent upon each other. The 3D reconstruction requires known poses of the camera, while the estimation of the absolute pose of the camera (with real distance) requires the known 3D scene. The basic framework to solve those two problems

concurrently is the structure from motion, as well as visual odometry when we focus on the pose estimation for navigation. With a depth sensor or stereo view, the coarse 3D scene can be generated in the beginning. Then the fundamental steps for the frontend are tracking the image scenes (feature points, landmarks, objects, etc.) between frames and estimating the camera motion with the geometry constraints. The backend records the key information from the frontend, such as the 3D scene, landmarks, and geometry constraints in keyframes over the navigation, and registers them into a global map. With that information, the backend refines the estimation results by feeding a set of keyframes into the optimizer. Meanwhile, if the frontend detected similar scenes where the backend believes the UAV has come back to a known place in the global map, the loop closure process will be activated. The loop closure detection can be both online or offline, while the optimization can only be done offline.

Structure from motion

Recovering 3D scene structure and camera motion is a fundamental process of VSLAM. Such recovery is possible by establishing algebraic relationships between knowns and unknowns. These relationships may come in the form of epipolar constraints, where known 2D correspondences are related to the unknown camera motion parameters. Similarly, algebraic relationships can also be established using known 2D-3D correspondences and unknown camera pose. The two are relative and absolute pose problems, resp., where recovering camera poses is of prime interest. If one is interested in recovering 3D from 2D image views, relationships between unknown 3D points and known relative pose and 2D correspondences are expressed in an algebraic form. Solving this algebraic system, for unknown coordinates of 3D points, is also known as triangulation.

The usage of algebraic relationships for absolute and relative pose estimations can be traced back to Grunert [29], Kruppa [30], and many others [31–38]. These for-

mulations recover the unknowns by solving systems of equations, which are generally assumed to exist on desired real manifolds. In the absence of noise, this assumption is justified. However, the desired solutions may not even exist in the presence of noise, or recovering them at the very least becomes very cumbersome. Two notable works [39, 40] that address this computational issue, have become influential in 3D vision [41].

The unknowns obtained by solving algebraic systems are often ambiguous or non-representative of the desired geometric measure. This is mainly because polynomial systems offer multiple solutions, only one of them is the desired one, while many others tend to be complex. Examples include: relative pose [40], absolute pose [39], triangulation [42], and camera calibration [43]. The algebraic geometry approach of analyzing the existence of the sought real solutions is studied in [44–47].

To circumvent the difficulty of making one choice among many, or dealing with complex solutions, the search process in practice is either randomized or formulated as the optimization over real space. The former assumes the availability of sufficiently many measurements, e.g. RANSAC, where some polynomial systems, derived for a so-called ‘minimal set’, offer desired and verifiable solutions [48–50]. The latter, often cast as non-minimal problem is non-convex optimization, for which the fast recovery of the optimal solution is difficult [51–53]. In either case, the quality of the obtained solution depends upon the algebraic formulation in case the measurement is noisy.

The current gold standard for algebraic formulation relies on some form of geometric errors [54, 55], in a normalized coordinate system [56]. For example, Bundle Adjustment (BA) [57] uses point re-projection, which is the geometric distance measure in image space. Similarly, 3D triangulation [42] minimizes the error distance in the reconstruction space. since the choice of error measure significantly influences the quality of the sought solution, different applications may require different measures. We are interest on the measure that is suitable for the long-term visual localization, and alike. Although such consideration has been made in the literature for 3D tri-

angulation [41], the same is missing for the camera pose estimation serving to the long-term localization, to the best of our knowledge.

Visual odometry

Recovering the 6-DoF ego-motion and calculating robot odometry from its camera system, also known as VO, is a fundamental problem in the field of computer vision and robotic automation [58]. The most popular solution is a high-precision lidar sensor [59–61]; however, due to budget and platform limitations, the design of indoor UAV/MAV navigation systems is mainly based on low-cost cameras and inertial navigation systems (INSs) [62–66]. Whenever an IMU is incorporated within the VO system, it is commonly referred to as VIO. To fly a UAV in full autonomous space, a sensor fusion system is essential, and VIO approaches are becoming popular [67–70]. While a flight controller, e.g., Pixhawk, takes all inputs, including vision pose, IMU data, and range measurement, and conducts the sensor fusion process, which is necessary to improve the data accuracy of each independent module. However, most state-of-the-art methods employ the VIO approach, and only a few studies focus on the output from pure vision [71–73].

Visual odometry research in computer vision has made significant progress over the past years, fueled in part by progress in deep learning [71, 74, 75]. The progress has been much slower for UAV autonomous navigation because only a few AI computing devices (e.g., NVIDIA Jetson family) are suited for UAV platforms. Moreover, those devices trade off other hardware resources for AI workloads. CPU-based devices such as NUC, UP Board, and LattePanda are still widely used for UAVs. Additionally, we need GPU to perform object detection and tracking. Therefore, this work addresses the CPU-based VO problem.

From the algorithmic point of view, there are two major types of VO: feature-based and direct methods. Feature-based methods follow the structure from a motion pipeline, which detects, tracks, and triangulates feature points across consecutive

views. The problem is nonlinear, and solving it is even more challenging when the set of 3D-2D correspondences contains bad geometry constraints (referred to as outliers hereafter). Many existing methods find the optimal solution by minimizing the reprojection error. Direct methods, in contrast, estimate the camera motion by directly minimizing the photometric errors between views. For instance, DSO [72] and its variant [71] sample image pixels that have intensity gradients, including smooth intensity variations or edges, and obtain the transformation by aligning them between two frames. However, the intensity difference between the two frames is nonconvex, which makes the direct methods susceptible to many issues, including frame inconsistencies in terms of abrupt motion, change in illumination, image blur, and rolling shutter effects introduced due to camera motion [76]. Hence, almost all of the existing direct-method-based VOs make the assumptions of sufficient scene illumination, the dominance of static scenes with abundant texture, and enough scene overlap between consecutive frames. Semidirect visual odometry (SVO) [73] takes advantage of both feature-based and direct methods by applying the direct method on small image patches that are extracted and tracked using some feature-based methods.

VO approaches can also be categorized by the type of visual input. In monovision, there is no baseline to calculate the real depth of the scene. Thus, the estimated translations will be up to scale throughout frames. Maintaining a uniform scale requires the accurate depth alignment of different scenes, which is rather difficult. Therefore, monovision-based methods usually use VIO, such as MSCKF [77], ROVIO [78], VINS-MONO [64], in conjunction with sensor fusion methods. For example, the MSCKF and ROVIO use an extended Kalman filter, while VINS-MONO employs a sliding window estimator. As VIO benefits from the independent orientation measurement using IMU, researchers have isolated the orientation estimation [79] and achieved considerable improvement. However, it is still difficult to define proper boundary conditions for orientation without using IMUs or other sensors. The drawback of mono-VIO is the initialization problem. Many existing methods fail to initialize or give poor state estimates. It is difficult for mono-vision to get quality stereo pairs for

3D scene triangulation when the view is stationary or rotates with little translation. In 2018, J. Delmerico et al. [80] conducted a comparison of monocular VIO for UAVs on the EuRoc benchmark [81] and identified the necessity of additional computations to improve VIO accuracy and robustness.

The alternatives to monocular vision are RGB-D and stereo VO, which do not suffer from depth alignment and are widely used in commercial products, such as the Intel RealSense family and Qualcomm Flight Pro. Visual odometry using stereo cameras has also matured significantly in recent years, but only limited solutions have been proposed [82, 83] compared to the vast works on monocular systems. Recent works convert monocular VO to stereo version [72, 84]. However, an accurate estimation using only vision still remains challenging, mainly due to the difficulties in tracking the camera motion, enforcing the correct set of geometry constraints, and computing the optimal solution of the triangulation.

2.2 UAV object tracking

In this section, three main tracking approaches, which are closely related to the proposed method in section 5, are presented, i.e., tracking by correlation filters, tracking by CNNs, as well as their hybrid approaches.

Tracking by correlation filters: Correlation filters have gained considerable attention because they convert the problem into the Fourier domain. The trackers, which employ correlation filters, compute the regression between the circular-shifted input features, and a Gaussian function model refers to the target. A notable work [17], popularly known as kernelized correlation filters (KCF) demonstrated excellent tracking performance by combining multi-dimensional features and kernels and finding the best filter taps that maximize the correlation response of over-sampled target.

Tracking by CNNs: The research on CNN-based visual tracking has achieved remarkable performance. For instance, the DeepTrack [85] learns effective feature representations of the target object in a purely online manner. Hong et al. [28]

take outputs from the first fully-connected layer to learn the target and background features. These approaches learn the positive and negative samples from a pre-trained CNNs. However, such models are designed to recognize numerous objects discard temporal information while the goal of visual tracking is to locate single or few objects' positions over time. Wang et al. [86] use a domain adaptation module for online adapt the pre-learned features according to the particular target object. This module has been integrated into other tracking methods and achieved significant improvement.

Tracking by KCF-CNN: Recently presented work, e.g. hierarchical convolutional features (HCF) [18] uses a KCF CNN-based hybrid approach. HCF uses 3 pre-selected layers, i.e., the max-pooling layers of *conv3*, *conv4*, and *conv5*, with fixed weights. The highest layer is able to discriminate the target while lower layers are used for precise localization. However, this method suffers from a complex background since most of the features that the CNN learned are background feature. In this paper, instead of using fixed layers, we propose a novel recommender to automatically select the best perceptive layers and the feature maps in each selected layer for the tracked object. To handle the scale variation, we present spatiotemporal-based min-channel feature maps. As a result, the target percept reconstructed from the recommended feature maps is robust to both appearance and scale changes of target objects.

3. CAMERA POSE ESTIMATION AND MINIMAL ADJUSTMENT

3.1 Problem statement

In SfM, the golden standard in 3D scene reconstruction and camera pose estimation is the minimization of the 3D point reprojection error. However, the inliers controlled by the reprojection error threshold cannot always result in low error estimations, which is demonstrated in the left plot of Fig. 3.1. The plot shows the distribution of a set of pose estimation errors under the same reprojection error threshold. The distribution’s covariance is unexpected. We summarize the characteristics of all pose estimation from Fig. 3.1 to investigate the reasons as Tab. ?? shows.

Inlier group size	Keypoint location distribution	Large pose estimation error	Sample significance level	Noise is normally distributed
Large	Evenly	Rarely	Low	Likely
Large	Unevenly	Occasionally	Unclear	Unclear
Small	Evenly	Occasionally	Unclear	Unclear
Small	Unevenly	Likely	High	Unlikely

The large pose error with a low error measure on point reprojection is usually be interpreted as a local minimum problem. While the girded keypoint detection and feature point boosting are the common solutions to reduce the effect of local minimum solutions. How to control the optimization process to avoid the unexpected local minimum lacks a systematic study of mathematical theory. To address this problem, we made an assumption: there exist low-reprojection-error correspondences that negatively affect the nonlinear refinement of pose estimation. These correspondences, alternatively called singular points, play no or little supporting role while estimating the camera pose using 2D reprojection error. The presence of noise in the singular points would hinder the regression process of accurate pose estimation. We propose a

camera adjustment (CamAdj) framework that refines the pose estimation by filtering the singular points.

In this chapter, we wonder if there exist a better error measure over the reprojection error for the problem at hand. Our approach relies on the camera pose uncertainty measure in the case of absolute and relative pose estimations. More precisely, we are interested in answering the following question for every independent measurement.

Problem 3.1.1 *Is the minimal pose adjustment necessary for a zero algebraic residual within the expected uncertainty?*

If the answer to the problem is affirmative, then we consider the corresponding measurement an inlier. Otherwise, the measurement is an outlier. In the context of this paper, we consider that some estimation of the camera pose is already known. This estimate can be obtained by either solving the algebraic system or through offline calibration (for example, calibrated stereo cameras). In fact, we are interested in how well the measurements, correspondences in our case, satisfy the estimated pose. Instead of the reprojection error (or similarity), we would like to know the influence of each measurement directly on the pose. Pose influence-based outlier filtration can play a vital role when (i) nonlinear refinement of the pose is performed in a nonminimal setup and (ii) inliers are used to localize new images. An intuitive illustration of our filtration technique is shown in Fig. 3.1 right plot.

The zero residual requirement of Problem 3.1.1 ensures the invariance of the proposed measure. In other words, for any algebraic measure (that must vanish in the absence of noise) in the presence of noise, the residual measure obtained by adjusting the camera does not change for any choice of the coordinate system. The proposed method relies on the theory of sum-of-squares (SoS), where adjusted camera poses are ensured to reside in the real space of the Euclidean group. We demonstrate the utility of the proposed filtration technique via several experiments that were conducted for accurate relative and absolute pose estimations. In this process, we filter the correspondences to obtain 3D points for later usage. The long-term effectiveness

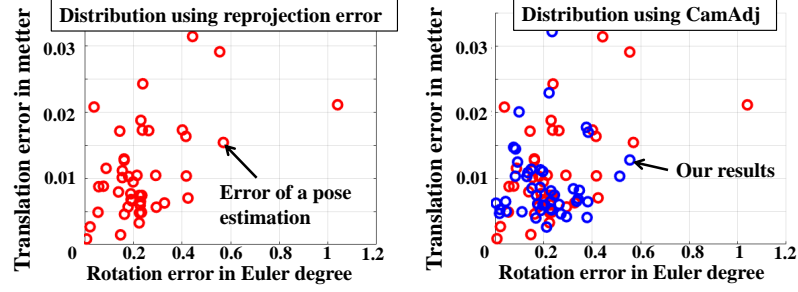


Fig. 3.1.: Pose estimation uncertainty using reprojection error (left) and the pose bounds (right). Each point represents one independent estimation of the camera pose. The pose estimations show large variations of errors using the same reprojection error threshold. This is because we solve the pose by the regression process that assumes the errors of the inliers are Normal distributed with $N(\mu, \sigma^2)$ around the groundtruth, while it is not always true. The pose bounds control the σ in different dimensions of the error space to ensure the uncertainty of each estimation. Data source: experiments of Fig. 3.5.

of our filtration is demonstrated by integrating it within the framework of VO, which is presented in chapter 4.

3.2 Notation list

The notations for this chapter are given by follow:

$u = \{u_1, u_2, \dots\}$: 2D point group in camera frame using homogeneous coordinates

$X = \{x_1, x_2, \dots\}$: 3D point group in world frame

I : 3×3 identity matrix

R : 3×3 rotation matrix

t : 3×1 translation vector

$P = [R | t]$: Transformation on the special Euclidean group

K : Camera intrinsic matrix

c : 3×1 rotation in Cayley parameters

G : Gram matrix of quadratic polynomial

\sim : Proportional to

$\llbracket \times$: Cross product

3.3 Preliminary

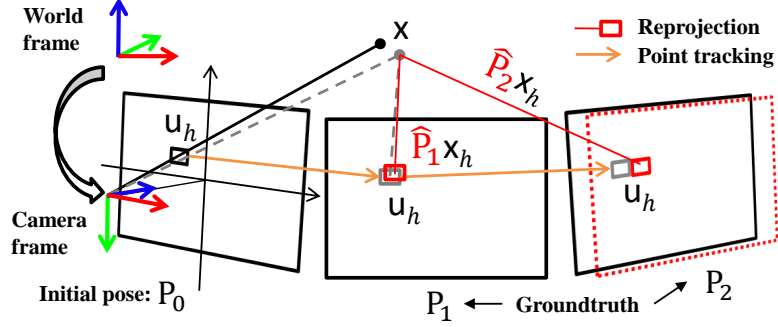


Fig. 3.2.: The setup for the general SfM problem, where P_0 represents the initial pose of the reference frame and \hat{P}_k represents the estimation of an unknown P_k or an adjusted pose of a known P_k .

We represent the Cartesian coordinates, in some world coordinate frame, of a 3D point x , by $x \in \mathbb{R}^3$. Similarly, a 2D point u is represented by $u \in \mathbb{R}^2$ in the camera coordinate frame. The setup is shown in Fig. 3.2. A pair of corresponding 3D and 2D points are related by the projection equation $u_h \sim Rx + t$ (the subscript h denotes homogeneous coordinate) for a camera with its pose given by rotation matrix $R \in SO(3)$ and translation vector $t \in \mathbb{R}^3$ such that the camera projection matrix is $P = [R \ t]$. For Cayley parameters $c \in \mathbb{R}^3$, we represent the rotation matrix as $R(c)$. The prior knowledge of the reference frame P' and its stereo pair P_1 is required for the initialization of real-scale SfM. In this setup, we are interested in knowing the following:

Problem 3.3.1 *Given a set of 3D-2D correspondences $\{\mathbf{x}_i, \mathbf{u}_i\}_{i=1}^m$ and the camera projection matrix $\hat{\mathbf{P}} = [\hat{\mathbf{R}} \ \hat{\mathbf{t}}]$, how well do the projection relationships $\mathbf{u}_{ih} \sim \hat{\mathbf{R}}\mathbf{x}_i + \hat{\mathbf{t}}$ hold?*

In practice, the location of \mathbf{x} is estimated by triangulation. Thus, the relationships $\mathbf{u}_{ih} \sim \hat{\mathbf{R}}\mathbf{x}_i + \hat{\mathbf{t}}$ are bound not to hold perfectly due to the presence of noise and outliers. Therefore, a good measure to evaluate the quality of the measurements is necessary. Let such an error measure function be $\rho(\mathbf{u}, \mathbf{x}, \hat{\mathbf{P}})$. In the literature, three commonly used error functions are $\rho_a = [\mathbf{u}_h]_{\times}(\hat{\mathbf{P}}\mathbf{x}_h)$, $\rho_{g2} = \mathbf{u} - \frac{(\hat{\mathbf{P}}\mathbf{x}_h)_{1:2}}{(\hat{\mathbf{P}}\mathbf{x}_h)_3}$, and $\rho_{g3} = (\hat{\mathbf{P}}\mathbf{x}_h)_3\mathbf{u}_h - \hat{\mathbf{P}}\mathbf{x}_h$, where $[\cdot]_{\times}$ is a skew symmetric matrix. The algebraic measure ρ_a is essentially the well-known direct linear transform (DLT) [32]. The favored geometric measure ρ_{g2} (resp. ρ_{g3}) computes the Euclidean distance between the projected 3D (resp. back-projected 2D) and measured 2D (resp. 3D) points [41]. These quality measures are of particular interest when estimating $\hat{\mathbf{P}}$ from a set of 3D-2D correspondences. However, $\hat{\mathbf{P}}$ is estimated by minimizing ρ_a using the closed-form solution of linear least-squares, followed by nonlinear refinement to minimize ρ_{g2} or ρ_{g3} . Minimizing ρ_{g2} maximizes the expectation under the assumption that the 3D points are noise free and 2D measurement noise is zero mean Gaussian. A similar assumption is made, in the opposite manner, while minimizing ρ_{g3} to maximize the expectation.

In this work, we investigate why the commonly used geometric measures occasionally generate biased errors in long-term localization ¹. This concern is raised due to the following observation: the 3D points reconstructed (or provided) could have different noise levels at different locations. Note that far away points are known to be more noisy when a stereo pair is used to reconstruct them. These far away noisy points could still result in low reprojection error. Later, when the same points are observed closely, these points must not be treated as inliers. Such treatment will result in large pose error. Knowing when to reject points exactly becomes tricky. Therefore, we wish to filter them as early as possible. A sensible solution could be modeling each point and its uncertainty with a probability distribution. But the

¹Since the issues of ρ_a have already been studied in several papers [41].

non-isotropic distributions of landmarks make it rather difficult to find the fast or closed-form solutions to the system with covariance model. Therefore, we consider an alternative way by filtering the landmark with an unexpected uncertainty. Our filtration approach assumes that the noisy 3D points are not aligned easily for perfect projection when we try to adjust the camera. Of course, this scenario could also be true for the noisy 2D points. In any case, we wish to filter 2D-3D points jointly altogether. More importantly, the noise in 3D propagates over time in long-term localization settings. Hence, our method can also be seen as the 3D noisy point filtration technique when applied to the long-term localization case. To perform the desired filtration, we are interested in determining the answer to Problem 3.3.1 in the context of accurate estimation/refinement of $\hat{\mathbf{P}}$ from noisy 3D-2D correspondences.

3.4 Problem Formulation

The representation of rotation matrix in terms of Quaternions is due to Sir William Hamilton. In this representation, a 3 dimension vector parallel to the rotation axis and the angle about it are used. It uses 4 variables in total to represent the rotation in 3D space.

Let $q = (x, y, z, w)^T$ be a quaternion vector, where (x, y, z) is the vector of rotation axis and w the rotation angle. The transformation between Euler angles rotation matrix and quaternion vector is given by:

$$R = \begin{bmatrix} w^2 + x^2 - y^2 - z^2 & 2(xy - wz) & 2(wy + xz) \\ 2(xy + wz) & w^2 - x^2 + y^2 - z^2 & 2(yz - wx) \\ 2(xz - wy) & 2(wx + yz) & w^2 - x^2 - y^2 + z^2 \end{bmatrix}. \quad (3.1)$$

Quaternions are the most widely used rotation matrix representation in non-linear optimization frameworks. However, due to an extra freedom on the number of variables, this representation also demands an extra constraint on the norm of Quaternions to be unity, so that final matrix representation is guaranteed to be a rotation

matrix. Hence, we choose to use Cayley transform based representation described as follows.

In 3D geometry, the rotation matrix representation based on Cayley transform uses only three variables. Let $\mathbf{c} = (x, y, z)^T$ be the vector of these parameters' entries, then a skew symmetric matrix $[\mathbf{c}]_{\times}$ and its final rotation matrix representation is given by:

$$[\mathbf{c}]_{\times} = \begin{bmatrix} 0 & z & -y \\ -z & 0 & x \\ y & -x & 0 \end{bmatrix} \leftrightarrow \frac{1}{K} \begin{bmatrix} x^2 - y^2 - z^2 & 2(xy - z) & 2(y + xz) \\ 2(xy + z) & -x^2 + y^2 - z^2 & 2(yz - x) \\ 2(xz - y) & 2(x + yz) & -x^2 - y^2 + z^2 \end{bmatrix}, \quad (3.2)$$

where $K = 1 + x^2 + y^2 + z^2$. In fact, the vector \mathbf{c} is the unit axis of rotation scaled by $\tan(\theta/2)$, where θ is the rotation angle. Furthermore, the transformation equivalence can be written as:

$$[\mathbf{c}]_{\times} = \begin{bmatrix} 0 & z & -y \\ -z & 0 & x \\ y & -x & 0 \end{bmatrix} \leftrightarrow (I - [\mathbf{c}]_{\times})^{-1}(I + [\mathbf{c}]_{\times}). \quad (3.3)$$

A 3D-2D correspondence $\{\mathbf{x}, \mathbf{u}\}$ satisfies the projection $\mathbf{u}_h \sim \mathbf{P}\mathbf{x}_h$ if and only if the quadratic polynomial $p(\mathbf{c}, \mathbf{r}) = [(\mathbf{I} - [\mathbf{c}]_{\times})\mathbf{u}_h]_{\times}((\mathbf{I} + [\mathbf{c}]_{\times})\mathbf{x} + \mathbf{r}) = 0$ is true.

Recall that $\mathbf{R} = (\mathbf{I} - [\mathbf{c}]_{\times})^{-1}(\mathbf{I} + [\mathbf{c}]_{\times})$ and let $\mathbf{r} = (\mathbf{I} - [\mathbf{c}]_{\times})\mathbf{t}$. Then, the following can be derived:

$$\mathbf{u}_h \sim \mathbf{P}\mathbf{x}_h = (\mathbf{I} - [\mathbf{c}]_{\times})^{-1}(\mathbf{I} + [\mathbf{c}]_{\times})\mathbf{x} + \mathbf{t}, \quad (3.4)$$

$$(\mathbf{I} - [\mathbf{c}]_{\times})\mathbf{u}_h \sim (\mathbf{I} + [\mathbf{c}]_{\times})\mathbf{x} + (\mathbf{I} - [\mathbf{c}]_{\times})\mathbf{t}, \quad (3.5)$$

$$p(\mathbf{c}, \mathbf{r}) = [(\mathbf{I} - [\mathbf{c}]_{\times})\mathbf{u}_h]_{\times}((\mathbf{I} + [\mathbf{c}]_{\times})\mathbf{x} + \mathbf{r}) = 0. \quad (3.6)$$

Because both $\mathbf{t} \in \mathbb{R}^3$ and $\mathbf{r} \in \mathbb{R}^3$, the iff condition holds true.

We denote matrices with uppercase letters and their elements by double-indexed lowercase letters: $\mathbf{A} = (a_{ij})$. Similarly, vectors are indexed: $\mathbf{a} = (a_i)$. We write $\mathbf{A} \succ 0$ (resp. $\mathbf{A} \succeq 0$) to denote that the symmetric matrix \mathbf{A} is positive definite (resp. positive

semi-definite). Any quadratic polynomial $p(\mathbf{y}) \in \mathbb{R}[\mathbf{y}]$, on the variables $\mathbf{y} \in \mathbb{R}^n$ can be represented using a homogeneous representation $\mathbf{y}_h = [\mathbf{y}^\top \ 1]^\top$ and the Gram matrix \mathbf{G} such that $p(\mathbf{y}) = \mathbf{y}_h^\top \mathbf{G} \mathbf{y}_h$. While adjusting the camera in accordance with this paper, the problem boils down to finding the root of quadratic polynomials defined on some variables \mathbf{y} around a guess $\hat{\mathbf{y}}$ within a desired radius δ . In algebraic terms, our interest in seeking the adjustment can be expressed in the following form. From Equation 3.6, we know that each correspondence generates 3 polynomials, which are denoted as f_θ .

We provide the details of f_θ and their Gram matrix \mathbf{G}_θ . Given 3D point in world system $\mathbf{X}_k = [x_w, y_w, z_w]^T$ and 2D image point in camera coordinate system $\mathbf{u}_k = [u, v, 1]^T$, Cayley's parameters $\mathbf{c} = [c_x, c_y, c_z]$, and new vector $\mathbf{r} = [r_x, r_y, r_z]^T$, the Equation (3.6) results following polynomials for every 3D-2D correspondence:

$$\begin{cases} f_1 = (c_x + v - c_z u)(z_w + r_z - x_w c_y + y_w c_x) \\ \quad - (c_y u - c_x v + 1)(y_w + r_y + x_w c_z - z_w c_x) = 0, \\ f_2 = (c_y u - c_x v + 1)(x_w + r_x - y_w c_z + z_w c_y) \\ \quad - (u - c_y + c_z v)(z_w + r_z - x_w c_y + y_w c_x) = 0, \\ f_3 = (u - c_y + c_z v)(y_w + r_y + x_w c_z - z_w c_x) \\ \quad - (c_x + v - c_z u)(x_w + r_x - y_w c_z + z_w c_y) = 0. \end{cases}$$

Thus, we can generate the \mathbf{G}_θ matrix for Equation 3.6:

$$\mathbf{G}_1 = \begin{bmatrix} y_w - z_w v & \frac{z_w u - x_w v}{2} & \frac{x_w v - y_w u}{2} & 0 & v/2 & 1/2 & z_w + y_w v \\ \frac{z_w u - x_w v}{2} & 0 & 0 & 0 & -u/2 & 0 & \frac{-x_w v - y_w u}{2} \\ \frac{x_w v - y_w u}{2} & 0 & 0 & 0 & 0 & -u/2 & \frac{-x_w - z_w u}{2} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ v/2 & -u/2 & 0 & 0 & 0 & 0 & -1/2 \\ 1/2 & 0 & -u/2 & 0 & 0 & 0 & v/2 \\ z_w + y_w v & \frac{-x_w v - y_w u}{2} & \frac{-x_w - z_w u}{2} & 0 & -1/2 & v/2 & z_w v - y_w \end{bmatrix}, \quad (3.7)$$

$$G_2 = \begin{bmatrix} 0 & \frac{y_w - z_w v}{2} & 0 & -v/2 & 0 & 0 & \frac{-x_w v + y_w u}{2} \\ \frac{y_w - z_w v}{2} & \frac{x_w v - y_w u}{2} & \frac{x_w v - y_w u}{2} & u/2 & 0 & 1/2 & z_w + x_w u \\ 0 & \frac{x_w v - y_w u}{2} & 0 & 0 & 0 & -v/2 & \frac{-y_w - z_w v}{2} \\ -v/2 & u/2 & 0 & 0 & 0 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/2 & -v/2 & 0 & 0 & 0 & -u/2 \\ \frac{-x_w v + y_w u}{2} & z_w + x_w u & \frac{-y_w - z_w v}{2} & 1/2 & 0 & -u/2 & x_w - z_w u \end{bmatrix}, \quad (3.8)$$

$$G_3 = \begin{bmatrix} 0 & 0 & \frac{y_w - z_w v}{2} & -1/2 & 0 & 0 & \frac{-x_w - z_w u}{2} \\ 0 & 0 & \frac{z_w u - x_w}{2} & 0 & -1/2 & 0 & \frac{-y_w - z_w v}{2} \\ \frac{y_w - z_w v}{2} & \frac{z_w u - x_w}{2} & x_w v - y_w u & u/2 & v/2 & 0 & v \\ -1/2 & 0 & u/2 & 0 & 0 & 0 & -v/2 \\ 0 & -1/2 & v/2 & 0 & 0 & 0 & u/2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{-x_w - z_w u}{2} & \frac{-y_w - z_w v}{2} & x_w u + y_w v & -v/2 & u/2 & 0 & y_w u - x_w v \end{bmatrix}. \quad (3.9)$$

Problem 3.4.1 Does there exists parameter \mathbf{y} such that $f_\theta(\mathbf{y}) = 0$ for $\mathbf{y} - \hat{\mathbf{y}} \leq \delta$ with $\delta \geq 0$ and $\mathbf{y} \in \mathbb{R}^n$, around $\hat{\mathbf{y}}$?

This problem can be answered using the following theorem.

Theorem 3.4.2 (Quadratic within bounds [87]) For a sufficiently small $\delta \geq 0$ and quadratic polynomial $f_\theta(\mathbf{y})$, the question of whether $f_\theta(\mathbf{y}) = 0$ for $\mathbf{y} - \hat{\mathbf{y}} \leq \delta$ and $\mathbf{y} \in \mathbb{R}^n$ is true or not, can be answered efficiently with a certificate using sum-of-squares polynomial optimization.

Theorem 3.4.2 makes use of the Gram matrices of the polynomial $f_\theta(\mathbf{y})$, say \mathbf{G}_θ , and the bound representing polynomials $f_k(\mathbf{y}) = (\underline{y}_k - y_k)(\bar{y}_k - y_k)$ for $\underline{y}_k \leq y_k \leq \bar{y}_k$, say \mathbf{G}_k . Then, the question in Problem 3.4.1 is answered by means of the following semi-definite feasibility test:

$$\begin{aligned} & \text{find} && \{\lambda_\theta \mid \theta = 1, 2, 3\}, \{\lambda_k \geq 0 \mid k = 1, \dots, 7\}, \\ & \text{subject to} && \sum_{\theta} \lambda_\theta \mathbf{G}_\theta + \sum_k \lambda_k \mathbf{G}_k \succeq 0. \end{aligned} \tag{3.10}$$

Note that $\underline{\mathbf{y}} \leq \mathbf{y} \leq \bar{\mathbf{y}}$ represents the elementwise inequality for the lower and upper bounds on variable \mathbf{y} . When we seek the existence of roots by adjusting around the initial guess $\hat{\mathbf{y}}$, we generate lower and upper bounds $\{\underline{\mathbf{y}}, \bar{\mathbf{y}}\}$ such that both extremes are within the radius δ , i.e., $\max(\mathbf{y} - \underline{\mathbf{y}}, \mathbf{y} - \bar{\mathbf{y}}) \leq \delta$. Intuitively, solving (3.10) seeks a quadratic sum-of-squares polynomial to test if $f_\theta(\mathbf{y})$ is *always* positive/negative within the bounds $\{\underline{\mathbf{y}}, \bar{\mathbf{y}}\}$. Since $f_k(\mathbf{y})$ are designed to be *negative* within bounds, an affirmative result for the feasibility test (3.10) implies that the polynomial $f_\theta(\mathbf{y})$ is always either positive or negative within bounds. Therefore, there exists no root within the investigated bounds. In other words, the answer to Problem 3.4.1 is negative, and the semidefinite problem of (3.10) is feasible. Theorem 3.4.2 also provides the answer to the "reverse" for sufficiently small δ . Note that (3.10) can be solved efficiently using semidefinite programming.

3.4.1 Adjusting Cameras for Resectioning

Camera resectioning, i.e., determine the true parameters of the camera that produced a set of given image views. In the presence of unknown noise and potential outliers, we are interested in answering Problem 3.3.1 by adjusting the camera. The qualified $\rho(\mathbf{u}, \mathbf{x}, \hat{\mathbf{P}})$, in this case, is the direct measure of the minimal necessary rigid motion adjustment for perfect resectioning. In the following, we present our proposal in mathematically formal terms.

Problem 3.4.3 For some metric for the difference between two camera poses, say $\mathbf{d}(\mathbf{P}_1, \mathbf{P}_2)$, compute the measure for the quality of the projection relationships, as asked in Problem 3.3.1, by minimally adjusting the pose $\hat{\mathbf{P}}$ to \mathbf{P} as follows:

$$\rho_d(\mathbf{u}, \mathbf{x}, \hat{\mathbf{P}}) \doteq \min_{\mathbf{P} \in SE(3)} \mathbf{d}(\mathbf{P}, \hat{\mathbf{P}}), \quad \text{s.t. } \mathbf{u}_{jh} \sim \mathbf{P}\mathbf{x}_{jh}, \forall j. \quad (3.11)$$

where j is the index of inliers. We are now interested in solving Problem 3.4.3 for each correspondence. We express $\mathbf{d}(\mathbf{P}_1, \mathbf{P}_2) = \max(|[\mathbf{c}_1^\top \ \mathbf{t}_1^\top]^\top - [\mathbf{c}_2^\top \ \mathbf{t}_2^\top]^\top|)$, for the L-infinity norm of the vector.

Recall that the rotation matrix $\mathbf{R}(\mathbf{c})$ is parameterized using Cayley parameters $\mathbf{c} \in \mathbb{R}^3$. The parameter \mathbf{r} is the reparameterized translation vector $\mathbf{r} = (\mathbf{I} - [\mathbf{c}]_\times)\mathbf{t}$. Using the new parameters, we express the projection equations as quadratic polynomial constraints with six degrees of freedom.

Proposition 3.4.1 A 3D-2D correspondence $\{\mathbf{x}_i, \mathbf{u}_i\}$ satisfies the projection $\mathbf{u}_{ih} \sim \mathbf{P}\mathbf{x}_{ih}$ if and only if the quadratic polynomial $p(\mathbf{c}, \mathbf{r}) = [(\mathbf{I} - [\mathbf{c}]_\times)\mathbf{u}_{ih}]_\times ((\mathbf{I} + [\mathbf{c}]_\times)\mathbf{x}_i + \mathbf{r}) = 0$.

To approximate the measure $\mathbf{d}(\mathbf{P}, \hat{\mathbf{P}})$ using SoS theory, we make use of a quadratic SoS polynomial such that $g(\mathbf{P}, \hat{\mathbf{P}}) = \mathbf{c} - \hat{\mathbf{c}}^2 + \mathbf{r} - \hat{\mathbf{r}}^2$. Let a new variable be $\mathbf{y} = [\mathbf{c}^\top \ \mathbf{r}^\top]^\top \in \mathbb{R}^6$ and let two Gram matrices be defined as follows:

$$\mathbf{y}^\top \mathbf{G}_\eta \mathbf{y} = g(\mathbf{P}, \hat{\mathbf{P}}) - \eta \quad \text{and} \quad \mathbf{y}^\top \mathbf{G} \mathbf{y} = p(\mathbf{y}). \quad (3.12)$$

Now, we are ready to present our preliminary result.

Proposition 3.4.2 The pose an influence measure, as defined in (3.11), any 3D-2D correspondence can be well approximated using the following iterative semidefinite program.

$$\begin{aligned} \rho(\mathbf{u}, \mathbf{x}, \hat{\mathbf{P}}) &= \min_{\lambda, \lambda_\eta, \eta \geq 0} \quad \eta, \\ \text{subject to} \quad &\lambda \mathbf{G} + \lambda_\eta \mathbf{G}_\eta \succeq 0, \\ &\nexists \lambda, \lambda_\eta \geq 0. \end{aligned} \quad (3.13)$$

If (3.13) had feasibility constraints, the problem could be solved in a single step. However, in the case of infeasibility, a bisection-like search on the variable η must be performed. Such jumps between two extremes to find the minimum positive η such that (3.13) is infeasible. Note that $\mathbf{d}(\mathbf{y})$ is SoS with zero minima and for perfect resectioning $p(\mathbf{y}) = 0$. Therefore, for perfect resectioning, $\eta \rightarrow \epsilon$ is sufficient to make (3.13) infeasible. It can be seen from (3.12) how η is related to the pose influence measure.

3.4.2 Rejection of Outliers for Resectioning

The exact measure offered in (3.13) could turn out to be computationally heavy due to its iterative nature. The iterative steps are not necessary if the threshold η is given, to decide if the 3D-2D points pair (\mathbf{u}, \mathbf{x}) with respect to $\rho_d(\mathbf{u}, \mathbf{x}, \hat{\mathbf{P}}) \leq \gamma$ for $\hat{\mathbf{P}}$. Now, we are ready to return to Problem 3.3.1 in the context of inlier/outlier filtration. Let the given set of tuples be $\mathcal{S} = \{\mathcal{S}_i\} = \{(\mathbf{u}_i, \mathbf{x}_i)\}_{i=1}^m$. For a given $\hat{\mathbf{P}}$, we are interested in determining whether the tuple \mathcal{S}_i results in $\rho_d(\mathbf{u}_i, \mathbf{x}_i, \hat{\mathbf{P}}) \leq \eta$. If the condition is satisfied, then \mathcal{S}_i is considered to be an inlier; otherwise, it is considered to be an outlier. The following corollary of our Proposition 3.4.2 concerns outlier filtering.

Corollary 3.4.4 *The set of inliers $\mathcal{I} \subseteq \mathcal{S}$ defined by $\mathcal{I} = \{\mathcal{S}_i \mid \rho_d(\mathbf{u}_i, \mathbf{x}_i, \hat{\mathbf{P}}) \leq \eta, \mathcal{S}_i \in \mathcal{S}\}$ can be obtained using*

$$\begin{aligned} & \text{find} \quad \lambda_i, \lambda_\eta, \\ & \text{subject to} \quad \lambda_i \mathbf{G}_i + \lambda_\eta \mathbf{G}_\eta \succeq 0, \end{aligned} \tag{3.14}$$

for the individual tuple $\mathcal{S}_i \in \mathcal{S}$. The feasibility of (3.14) implies that the tuple \mathcal{S}_i is an outlier; otherwise, the tuple is an inlier.

Property 3.4.5 (Invariance) *The inlier set \mathcal{I} obtained using Corollary 3.4.4 is invariant under measures $\rho_a, \rho_{g1}, \rho_{g2}$ as well as the rigid transformation of 3D points and camera.*

3.5 Two-view Triangulation

We are interested in knowing, for a calibrated stereo setup, if the given 2D correspondences result in meaningful triangulation. In other words, we want to investigate if we can fine-tune the external parameters of the pre-calibrated stereo camera for better triangulation results. As in the previous section, we seek a set of 2D correspondences that can be perfectly triangulated by minimally adjusting one of the cameras. We modify the configuration in Fig. 3.2 by fixing the reference camera $P' = P_0 = [I \ 0]$ and performing the camera adjustment on its stereo pair P_1 . Let \hat{P} be the adjusted pose of P_1 . Now, we are interested in the following for the problem of two-view triangulation.

Problem 3.5.1 *Given 2D corresponding points $\{u', u\}$ and camera projection matrices $P' = [I \ 0]$ and $\hat{P} = [\hat{R} \ \hat{t}]$, is $\rho(u', u, \hat{P}) \leq \eta$ for a known threshold η , depth γ , and*

$$\rho_{d2}(u', u, \hat{P}) = \min_{P \in SE(3)} d(P, \hat{P}), \quad s.t. \ u'_h \sim \gamma \hat{R}u_h + \hat{t} \quad (3.15)$$

To address the above problem, we make use of the following corollary of our Proposition 3.4.1 from the previous section.

Corollary 3.5.2 *Any 2D correspondence $\{u', u\}$ satisfies the two-view relationship $u'_h \sim \gamma \hat{R}u_h + \hat{t}$ if and only if $p(c, r, \gamma', \gamma) = \gamma'(I - [c]_{\times})u'_h - \gamma(I + [c]_{\times})u_h - r = 0$.*

We define parameters slightly differently in this section. For $y = [c^T \ r^T \ \gamma \ \gamma']^T \in \mathbb{R}^8$, Gram matrices G_η and G are defined similar to in (3.12). Thereafter, one can draw a parallel to the previous section to the corollary below for the Corollary 3.4.4.

Corollary 3.5.3 *For a given set of 2D correspondence tuples $\mathcal{S} = \{\mathcal{S}_i\} = \{(u'_i, u_i)\}_{i=1}^m$, the inliers $\mathcal{I} \subseteq \mathcal{S}$ defined by $\mathcal{I} = \{\mathcal{S}_i \mid \rho_{d2}(u'_i, u_i, \hat{P}) \leq \eta, \mathcal{S}_i \in \mathcal{S}\}$ can be obtained similar to (3.14), for the individual tuple $\mathcal{S}_i \in \mathcal{S}$.*

It goes without saying that the property of invariance holds for triangulation as well. In this case, however, we introduced two extra variables for depth in two views. At

this point, it is important to know that Theorem 3.4.2 is true only when the variables \mathbf{y} are bounded. Regardless, this constraint is not a problem for the case of triangulation because the chirality of the 3D points ensures that the depth must have lower bounds. Similarly, the impossibility of reconstructing the (metric) point at infinity from two views ensures the upper bound. One more issue that is still pending is the bounds on the variables \mathbf{c} and \mathbf{r} . As such, given an initial guess of the absolute or calibrated relative camera pose, for resectioning or triangulation, the bounds on \mathbf{c} and \mathbf{t} can be easily derived. From these bounds, the bounds on \mathbf{r} can also be derived using the method of interval analysis [88]. If we wish to define the allowed thresholds on each of the entries of 6DOF, unlike the general threshold on $\mathbf{d}(\mathbf{P}, \hat{\mathbf{P}})$, we can still use Theorem 3.4.2. In this case, the allowed bounds $\underline{\mathbf{c}} \leq \mathbf{c} \leq \bar{\mathbf{c}}$ and $\underline{\mathbf{r}} \leq \mathbf{r} \leq \bar{\mathbf{r}}$ are derived from the initial guess $\hat{\mathbf{P}}$ and the thresholds on the individual variables. When we know the bounds, the formulation of (3.14) takes the form of (3.10).

3.6 Long-Term Localization

In the keyframe step, as shown in Fig. 3.2, we triangulate the correspondences between the stereo pairs $[\mathbf{P}_0, \mathbf{P}_1]$. We fixed \mathbf{P}_0 as reference, then filter each 2D-3D correspondence of \mathbf{P}_1 using CamAdj (Corollary 3.4.4), where the bounds are defined by very small intervals around \mathbf{P}_1 . This step allows us to have faithful 3D reconstructed points with a very low $\mathbf{d}(\mathbf{P}_1, \hat{\mathbf{P}}_1)$. These 3D points are expected to yield low $\mathbf{d}(\mathbf{P}_k, \hat{\mathbf{P}}_k)$ for the following non-keyframes' estimation $\{\hat{\mathbf{P}}_k | k = 2, 3, \dots\}$, which is verified in next section.

3.6.1 Keyframe Processing

In the keyframe step, we extract and match features between stereo pairs. This step allows us to have a faithful 3D reconstruction with a known scale without requiring motion estimation. Such 3D is obtained by triangulating the correspondences between the stereo pairs. During the process of triangulation, 2D-2D correspondences

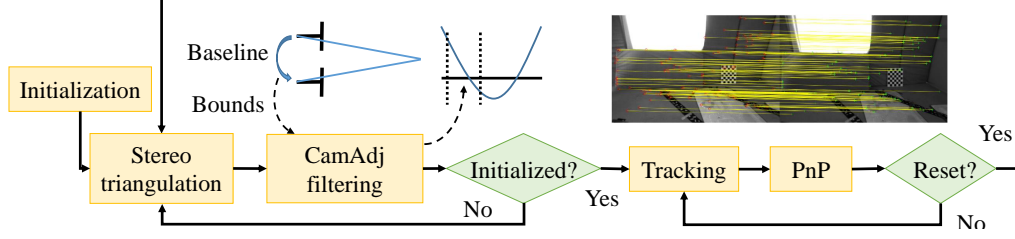


Fig. 3.3.: The system overview of stereo visual odometry. The insufficient points will activate the reset of keyframe.

are filtered using Corollary 3.5.3. More precisely, the correspondence tuples \mathcal{S} are obtained from the stereo pair, and the pose for each point is adjusted from the known pose $\hat{\mathbf{T}} = [\mathbf{R}_{12} | \mathbf{t}_{12}]$, where $[\mathbf{R}_{12} | \mathbf{t}_{12}]$ is the calibrated pose between the pair.

3.6.2 Non-keyframe Processing

The non-keyframes are localized with a typical SfM workflow. We do not detect new keypoint in non-keyframes. All correspondences are established by tracking the keypoints across frames independently for both stereo views. The obtained 2D-3D correspondences are used to localize with the perspective-n-point method [32] or variants.

3.6.3 The Algorithm

During the process of long-term localization, the 3D landmarks are reconstructed for every keyframe, which is later used to localize non-keyframes, as shown in Fig. 3.3. To validate the theoretical correctness of CamAdj, we do not add a new keypoint and maintain the 3D points in a very simplified way: only keep the tracked inliers in the non-keyframes.

3.7 Experiments

This section presents the experiments that have been conducted in relation to Sections 3.4 and 3.5. We use computer vision toolbox functions in MATLAB to perform the Perspective-n-Point (PnP) estimation for the camera pose estimation on both the simulation and the ETH3D benchmark [89]. The PnP method employed in our experiments involves the outlier rejection process using the M-estimator sample consensus (MSAC) algorithm. In contrast, we show the differences between the pose estimation results with and without CamAdj filtering. We first demonstrate that there are singular points in the reprojection error inliers that detriment the accuracy of the pose estimation. Then, we show that the proposed CamAdj filtering improves the pose estimation accuracy by rejecting these singular points. Later, we present results on the EuRoC benchmark [90] dataset for long-term stereo visual odometry using CamAdj filtering. The pose bounds $[\alpha, \beta]$ are given by $\mathbf{c} - \hat{\mathbf{c}} \leq \alpha$ and $\mathbf{r} - \hat{\mathbf{r}} \leq \beta$ for all experiments.

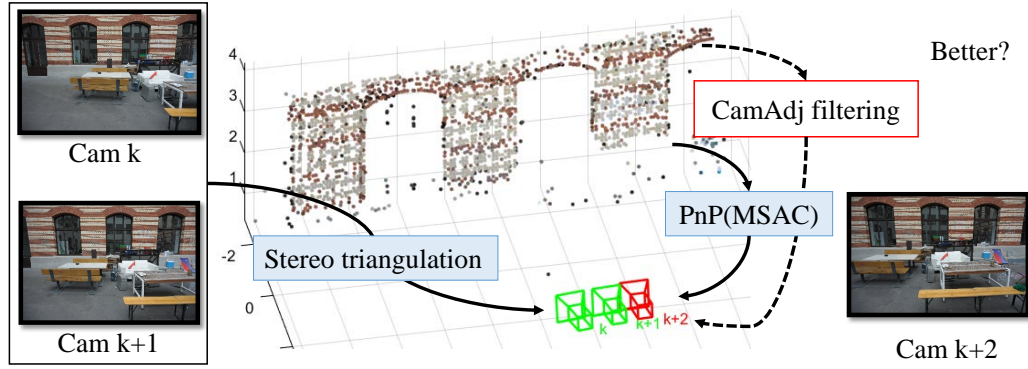


Fig. 3.4.: We estimate the unknown pose of the third view using the known poses of cameras k and $k + 1$ (which can be seen as calibrated stereo). The 3D points are filtered by CamAdj, which results in a better pose estimation for the third view.

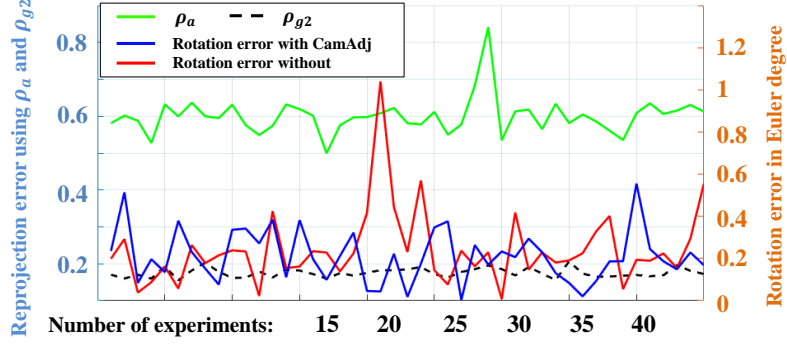


Fig. 3.5.: Reprojection error metrics and pose rotation error with CamAdj and without. In each experiment, we plot the mean errors of the ρ_a and ρ_{g2} measure of overall correspondences for each independent camera pose estimation (using ρ_{g2}) and the norm of the rotation error compared to groundtruth. The results show nontrivial fluctuations of rotation error refer to the measurements of these error metrics. The results also demonstrate that the abnormal errors can be removed by CamAdj. The distribution of the pose errors of all estimations is shown in Fig. 3.1.

3.7.1 CamAdj for Resectioning

Singular points. The singular points are 2D-3D correspondences that play no or little supporting role while estimating the camera pose using 2D reprojection error. In fact, the presence of noise in the singular points hinders the process of accurate pose estimation. To demonstrate the existence/effect of the singular points, we conduct experiments on real scene data, i.e., the ETH3D stereo dataset. The configuration of our experiments is shown in Fig. 3.4, where we simply use two known views as a stereo camera to estimate the camera pose of the third view and repeat the pose estimation for the same 3-view group in the ETH3D dataset *boulders*. In this setting, we study the relationship between the commonly used reprojection error metrics (ρ_a and ρ_{g2} in Section 3.4) and the rotation error, as shown in Fig. 3.5. Our observation shows that singular points occasionally appear in the inlier group, which misleads the pose estimation process. In other words, there exist some subsets in the 2D-3D

correspondences that generate low reprojection error but high rotation or translation error.

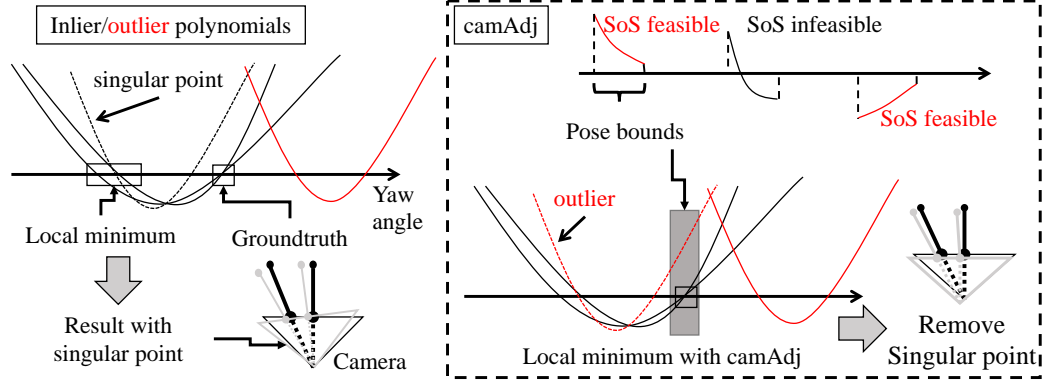


Fig. 3.6.: Illustration of the singular point hypothesis.

The singular point is our hypothesis based on an empirical observation from the experiment in VI. A and the geometric representation of the polynomial system are illustrated in Fig. 3.6 in this letter. We simplify the polynomial into one dimension (yaw angle only) for visualization purposes. Most of the state-of-the-art methods are seeking the local minimum of the overall system, where the singular points might mislead the optimizer to fall into a sub-optimal solution. Our method identifies the singular points by using the SoS feasibility of each polynomial in the given bounds (each 3D-2D correspondence). This examination is based on a known stereo pair, where the bounds are obtained from the prior knowledge of the extrinsic parameters. Thus, we can filter singular points for future estimations. To be clear that Fig. 3.6 is a hypothesis based on mathematical theory and the original polynomial system is in 6D space, which is impossible to be visualized. A rigorous demonstration is shown in Fig. 3.1, where we show the error distribution on the normalized rotation and translation of the estimated poses using real data from the experiments of Fig. 3.5.

Outlier filtration (synthetic data). We first show the filtering of the outlier (alternatively called singular) points using the proposed CamAdj technique on synthetic data. We set a virtual camera with focus $f_x = f_y = 800$ and center $cx = cy = 1024$ in

bounds $[\alpha, \beta]$	noise free	general error	R error	t error
[0.01,0.01]	0%	2%	0%	0%
[0.001,0.001]	0%	22%	18%	14%
[0.0005,0.0005]	0%	68%	62%	48%
[0.0001,0.0001]	0%	99%	98%	98%
[0.001,10 ⁻⁵]	0%	40%	36%	12%
[0.001,10 ⁻⁶]	0%	42%	42%	16%
[0.001,10 ⁻⁷]	0%	40%	43%	17%
[0.0005,0.001]	0%	58%	66%	34%
[0.0001,0.001]	0%	86%	96%	73%
$[5 \times 10^{-5}, 0.001]$	0%	88%	98%	82%
$[10^{-5}, 0.001]$	0%	90%	98%	88%

Table 3.1.: Average rejection percentage of each data group with different bounds (higher is better). The results indicate that the CamAdj filtering has different rejection performances in different types of errors. The reported experiments are conducted after the reprojection error-based filtration technique. Among the three different sets (of various noise levels), the proposed CamAdj can well identify correspondences from the highest noise level. This in turn means, correspondences from noisy pose set when identified and filtered (which cannot be done any further using reprojection error) can lead to better pose estimation.

camera matrix. During this computation, four groups of 2D-3D correspondences are generated for the same view, with various noise levels (up to 5 pixels). The 3D points are randomly generated and fixed, followed by 2D point generation using (i) inliers: ground truth (\mathbf{R}, \mathbf{t}) pose with very small Gaussian noise. (ii) General error: noisy camera pose $(\mathbf{R} + \Delta\mathbf{R}, \mathbf{t} + \Delta\mathbf{t})$. (iii) R error: $(\mathbf{R} + \Delta\mathbf{R}, \mathbf{t})$. (iv) t error: $(\mathbf{R}, \mathbf{t} + \Delta\mathbf{t})$. In our experiments, 100 points were generated for each group.

The noise on the 2D points is implicitly introduced by introducing the noise in the pose. Now, using only the 2D-3D correspondences, we wish to measure their quality for pose estimation of the next view. Using the current view’s pose, we first filter the correspondences. In this process, the camera is adjusted to measure the influence of the correspondences on the pose estimation of the next views. The adjustment is conducted within different allowed bounds for the inlier/outlier separation (please refer to Problem 3.2). In Table 3.1, we report our results for ± 0.5 (Euler degree) noise on rotation with a fixed offset and ± 0.05 (meter) on translation. We use Gaussian noise, and each correspondence is generated independently. Tab. 3.1 shows the CamAdj filtering percentage of each group with different bounds $[\alpha, \beta]$. The same bounds are used for all of the entries of \mathbf{c} and \mathbf{r} . The noise free group is reported for the reference. In fact, such a group does not exist in practice. The reported experiments are conducted after the reprojection error-based filtration technique. Our results imply that for tighter bounds on rotation and translation, CamAdj filtering has a similar outlier rejection rate on all error groups.

Dataset	Number of test	R error (Euler degree)		t error (m)		Number of pair	Rejection rate (%)	FPS	Better R rate (%)	Better t rate (%)
		MSAC	CamAdj+MSAC	MSAC	CamAdj+MSAC					
boulders	50	1.7680	1.5901	0.0106	0.0097	155	3.8710	1.3	58.0	62.0
bridge	1624	0.9074	0.8832	0.0273	0.0271	124	9.9408	1.7	49.6	46.7
courtyard	300	0.2205	0.2036	0.0068	0.0064	74	3.0452	2.6	52.0	52.0
relief_2	450	1.3275	1.2916	0.1784	0.1777	154	4.2595	1.4	51.8	52.0
terrace	39	0.8267	0.7897	0.0102	0.0096	63	4.7619	3.6	53.8	53.8
terrace_2	101	0.0420	0.0410	0.0020	0.0019	71	2.3219	2.9	52.5	54.5
Average	427	0.8822	0.8546	0.0498	0.0495	107	7.6396	2.25	52.95	53.4

Table 3.2.: Pose estimation for the third view evaluated on ETH3D. We conduct experiments illustrated in Fig. 3.4 for every 3 cameras in the listed datasets. In order to show the impact of CamAdj filtration, we tune the CamAdj bounds iteratively until the CamAdj reject 1 to 25 % points (average is given by Rejection rate). The test of each group has been repeated 50 times. Better R/t rate indicates the percentage of the tests when CamAdj filtration generates a better pose than MSAC.

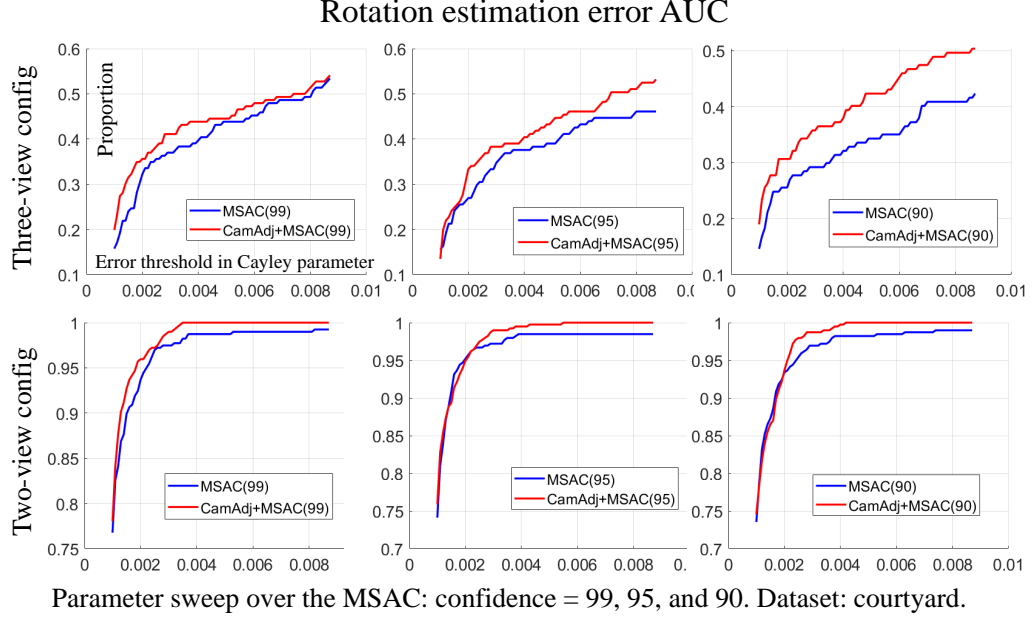


Fig. 3.7.: Rotation estimation error AUC for the third view in the three-view configuration and two-view triangulation with unknown depths on ETH3D dataset *courtyard*. The maximum rotation error thresholds in Cayley were set to 0.0087 (1° in Euler). The results indicate that CamAdj could yield better rotation estimation overall.

Pose estimation (real data). We demonstrate that the camera pose estimation can benefit from CamAdj filtering on the ETH3D benchmark evaluation. The pose estimation setup follows the illustration in Fig. 3.4. To show the impact of CamAdj filtering, we use dynamic bounds by lowering the bounds when sufficient inliers are present, thus enforcing the rejection of a few bad points. The results of the pose estimation for the third view are given in Tab. 3.2. The overall results on all datasets demonstrate the utility of CamAdj filtration by consistently offering better pose estimation of the third camera. With regard to the computational complexity, camAdj filtration takes approximately 5 ms per correspondence on a laptop with an Intel i7 8750H core. To further demonstrate the improvement of the CamAdj filtering, we report the area under the curve (AUC) on the proportion of the estimated poses that

respect the given error threshold (upper row of Fig. 3.7). In the plot, the y axis is the proportion of the results within the threshold given by the x axis. It is clear that CamAdj filtering increases the probability of having better rotation estimation.

3.7.2 CamAdj for Two-view

For the two-view triangulation problem in Section 3.5, we again report the AUC plot (of 400 time tests on the *courtyard* dataset with fixed bounds) in Fig. 3.7 on the bottom row. The results clearly show that the 2D-2D motion estimation with CamAdj filtering also has a higher probability of providing a better rotation estimation. In Tab. 3.3, we also report 3D point reconstruction obtained using the proposed method. Note that the reported results are significantly superior to those of MSAC. We visualize a few example results of both MSAC and CamAdj in Fig. 3.8. In the reported results, we use fixed bounds $[0.001, 0.5]$ for $[\alpha, \beta]$. Note that the number of final reconstructed points after CamAdj filtering is sometimes even better than that of MSAC, offering us a larger set of reconstructed points.

Dataset	Error (m) / Reconstructed points				Rejection rate (%)	γ & γ'
	MSAC		CamAdj+MSAC			
boulders	0.371	291	0.263	301	9.5	[1,30]
bridge	0.224	110	0.142	82	63.9	[1,15]
courtyard	0.253	206	0.162	189	28.5	[1,15]
relief_2	0.101	157	0.037	148	8.8	[1,15]
terrace	0.258	42	0.186	49	39.0	[1,30]
terrace_2	0.098	127	0.038	150	20.2	[1,15]
Average	0.218	156	0.138	153	28.3	-

Table 3.3.: Two-view reconstruction. The 3D reconstruction errors show a better quality of triangulation after CamAdj filtration. γ & γ' are the depth bounds (refer Corollary 3.5.2)

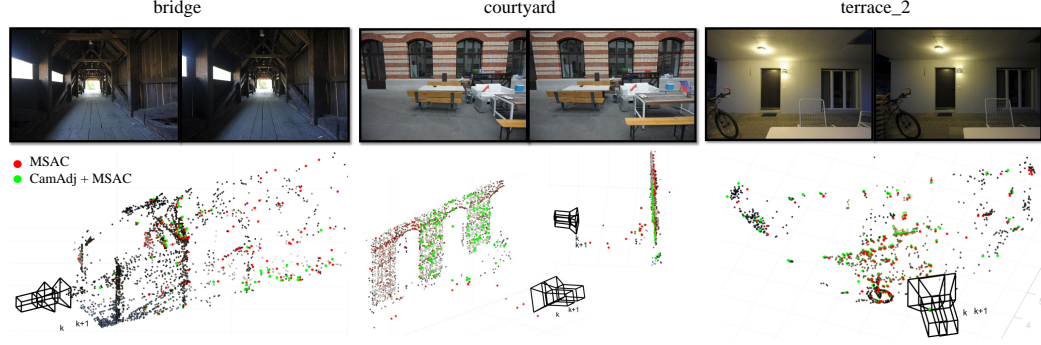


Fig. 3.8.: 3D points of different sequences; ground truth (colored), reconstructed by MSAC (red) and CamAdj (green).

3.7.3 Long Term Localization

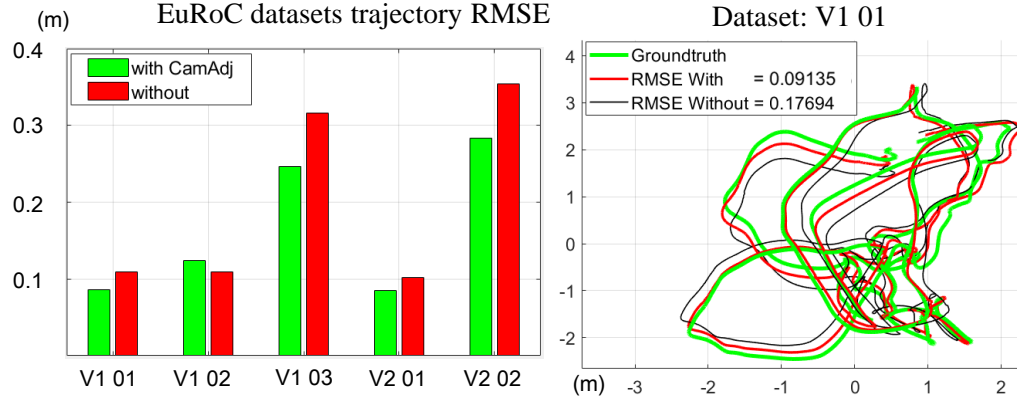


Fig. 3.9.: Comparison of VO trajectories' absolute RMSE with and without CamAdj filtering (left). An example of the complete motion trajectory of the same settings (right).

We performed the proposed CamAdj filtering for stereo visual odometry on the EuRoC benchmark. To show the impact of the CamAdj filtering, the loop closure and bundle adjustment were deactivated. The average absolute trajectory RMSE of 10 runs of visual odometry with and without CamAdj filtering is given on the left of Fig. 3.9. On the right side, we show a sampled trajectory estimation result. For the

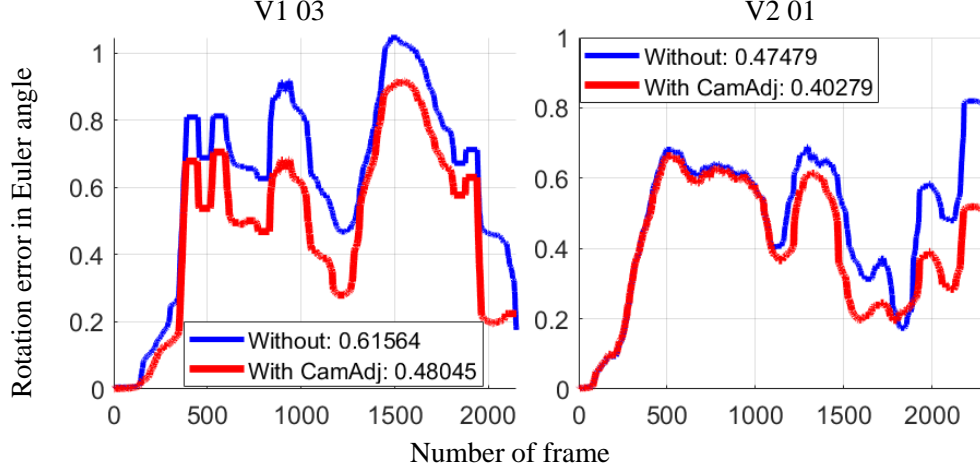


Fig. 3.10.: Absolute rotation error in Euler angle (0 to π), in the case of long-term localization.

Dataset\Method	svomsf	msckf	okvis	rovio	vins-mono	svogtsam	CamAdj (ours)
V1 01	0.40	0.34	0.09	0.10	0.07	0.07	0.05
V1 02	0.63	0.20	0.20	0.10	0.10	0.11	0.07
V1 03	x	0.67	0.24	0.14	0.13	x	0.14
V2 01	0.2	0.10	0.13	0.12	0.08	0.07	0.05
V2 02	0.37	0.16	0.16	0.14	0.08	x	0.16

Table 3.4.: Trajectory RMSE comparison on EuRoC benchmark evaluation. The evaluation metric and results of the compared mono-VIO methods are adopted from [91], while our method runs in pure stereo-VO model. Here ‘x’ implies the failure. This comparison aims to show the accuracy level of the proposed VO. However, the computational efficiency of CamAdj is very low because all correspondences are evaluated independently. It is hard to achieve real-time processing.

difficult sequences *V103* and *V202*, in which the drone was flying aggressively, the inlier group became so small that even a few singular points could mislead the final estimation. In Fig. 3.10, we compare the absolute rotation errors of sequences *V103* and *V201*. The results demonstrate that CamAdj filtering significantly improved

Dataset	CamAdj	BA	CamAdj+BA
V1 01	0.08	0.10	0.11
V1 02	0.12	0.15	0.13
V1 03	0.25	0.24	0.24
V2 01	0.08	0.17	0.14
V2 02	0.28	0.30	0.19

Table 3.5.: Results corresponding to Fig.7. We sync the keyframe update with BA for our method, which yield slightly different results with free run in TABLE 4.1.

odometry estimation for those sequences. It is important to note that the long-term localization accumulates errors over time. Therefore, long-term localization can significantly benefit from the accurate localization of individual frames. We show the results of basic keyframe-based visual odometry (implemented by following MATLAB Documentation: Structure From Motion From Multiple Views) with BA with and without CamAdj in Tab. 3.5. To fairly compare the results, we force the keyframe updating process for CamAdj to be the same as the VO using BA. As expected, BA can still benefit from camAdj. The CamAdj free run results are evaluated in Tab. 4.1.

4. STEREO ORIENTATION PRIOR FOR VISUAL ODOMETRY

In previous chapter, we studied the problem of measuring the quality of 2D-2D and 2D-3D correspondences for the tasks of absolute/relative pose estimation and 3D triangulation. Our quest was to measure the quality by means of moving the camera in such a way that the error is minimized and becomes exactly zero for every point individually. The minimal necessary adjustment then directly reflects the sought measure. In this chapter, we use the same mathematical tool to verify another assumption: there exist large-reprojection-error points that adding them into pose estimation results in an increase in translation error and a decrease in orientation error, which is a meaningful trade-off for long term visual odometry (VO) problem.

4.1 Notation list

The definition of general notations in this chapter is given as follows.

\mathbf{I} : 3×3 identity matrix

\mathbf{R} : 3×3 rotation matrix

\mathbf{t} : 3×1 translation vector

\mathbf{c} : rotation in Cayley parameters.

$\mathbf{t}_{new} = (\mathbf{I} - [\mathbf{c}]_{\times})\mathbf{t}$: 3×1 vector that represent the merged term

\mathbf{G} : Gram matrix of quadratic polynomial.

$\mathbf{P} = \{\mathbf{P}_1, \mathbf{P}_2, \dots\}$: 3D points (landmarks).

\mathbf{p}' and \mathbf{p} : Keypoints on left (left) and right (right) views.

$\mathbf{T} = [\mathbf{R}, \mathbf{t}; \mathbf{0}, 1]$: transformation matrix.

$\mathbf{x} = [\mathbf{c}^T \ \mathbf{t}^T]^T$: the vector representing the 6D pose.

α and β : the bounds of rotation and translation variation around the baseline.

4.2 Methodology

This section describes the algorithm pipeline using overview flowcharts (Fig. 4.2) and pseudocode. We illustrate the main contributions of the proposed method in Fig. ???. The proposed method consists of keyframe and nonkeyframe states. The keyframe state reconstructs the 3D scene, while nonkeyframe states estimate the pose dependent upon the 3D scene. The pseudocode of these two states is given in Algorithms 1 and 2, respectively. The Img_{left}, Img_{right} are image inputs from stereo camera. The proposed method detects keypoints in the left camera using the ORB [3] feature detector. In the keypoint detection process, the image is divided into an $M \times N$ grid to obtain a proper feature distribution. This process is denoted as *gridFeatureDetection* in Algorithm 1. These keypoints are tracked in the right camera and subsequent frames using the KLT [92] point tracker. Once the 3D points are initialized by the stereo view, the system switches to a nonkeyframe and runs until it matches the keyframe reset conditions: insufficient valid points or large motion. In a nonkeyframe, the keypoints are tracked by the KLT, and the camera absolute pose is estimated by the PnP process. Then, features are detected again to add new keypoints into the current frame and compute the location of both existing and newly added 3D points from a stereo view. The 3D points $\mathbf{P} = \{\mathbf{P}_1, \mathbf{P}_2, \dots\}$ when projected on the left and right camera are given as \mathbf{p}' and \mathbf{p} , respectively. Let P^{cur} and P^{pre} represent the same landmark triangulated in the current and previous frames, respectively. They are detected by the ORB feature detector in a keyframe and matched by the KLT point tracker in a nonkeyframe. Note that this process will always obtain new locations \mathbf{P}^{cur} of the existing 3D points \mathbf{P}^{pre} from stereoview in the current frame because no signal tracking and pose estimation is accurate enough; moreover, there is pixel-level noise. How to update them to approach the ground truth (\mathbf{P}^{new}) is crucial

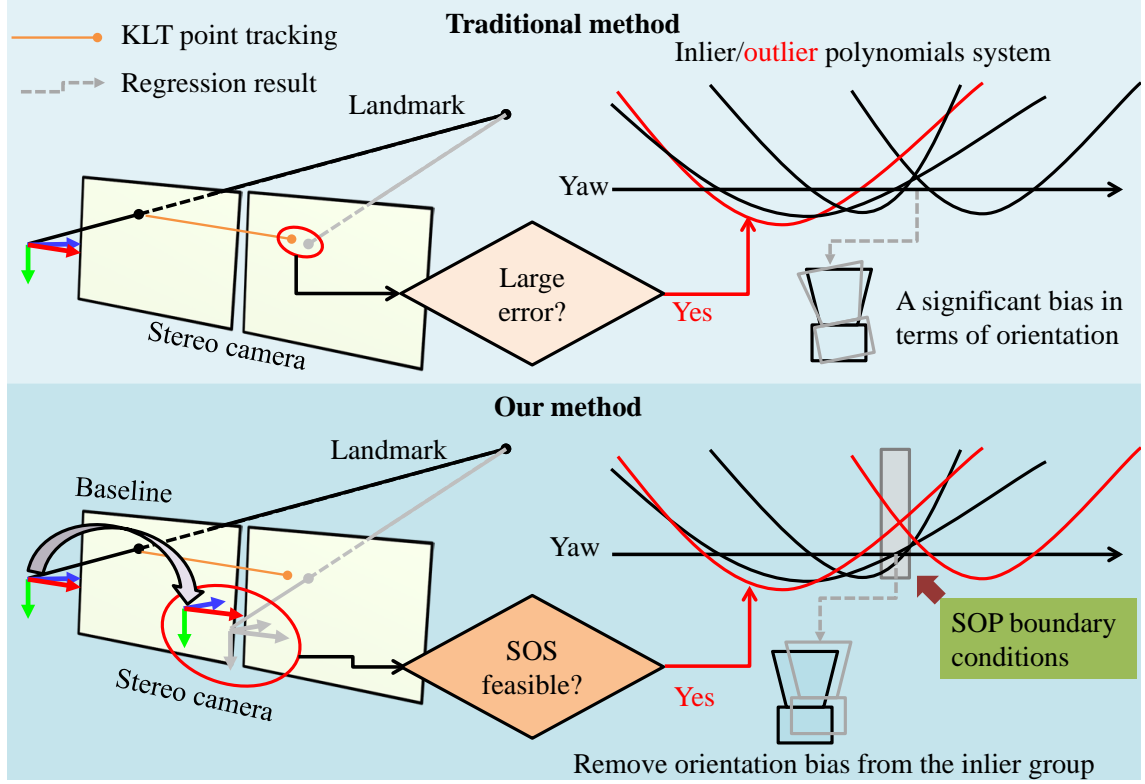


Fig. 4.1.: Our contributions: We visualize one dimension (yaw angle alone) of the polynomial system of the 2D-3D correspondences to illustrate the fundamental theory in geometric representation. Because of image noise and point tracking uncertainty, the zero residual of an inlier polynomial does not always appear at the groundtruth. The optimal solution to the polynomial system (usually computed using the least square method) may have a significant bias in terms of orientation. Our method employs SOS to assess data bias, with the stereo camera serving as a bias measurement baseline, much like base stations in differential GPS techniques. Unfortunately, eliminating bias in both orientation and translation may result in a scarcity of inliers. As a result, our method rejects orientation-bias outliers because they have a greater impact on odometry. In subsequent frames, the bias-compensated group of filtered 3D points can be used to regress robust and accurate poses.

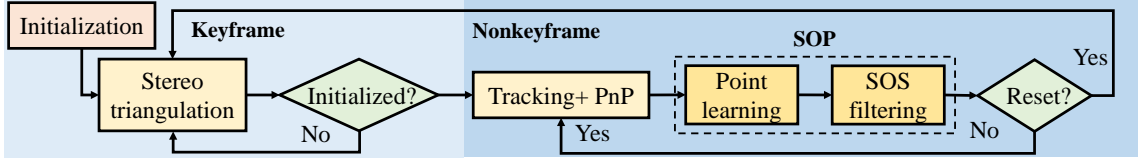


Fig. 4.2.: System overview

because the odometry estimation is completely dependent upon them. We propose a 3D point learning strategy and stereo orientation prior to merge and evaluate these points and reject outliers.

Algorithm 1: Keyframe state

Data: Img_{left}, Img_{right}

Result: \mathbf{P}

- 1 $\mathbf{p}' \leftarrow gridFeatureDetection(Img_{left})$;
 - 2 //find keypoint pairs by mutual KLT tracking ;
 - 3 $\mathbf{p} \leftarrow KLT(Img_{right}, \mathbf{p}')$, $\mathbf{P} \leftarrow triangulation(\mathbf{p}', \mathbf{p})$;
 - 4 **if** *insufficient valid points* **then**
 - 5 reset ;
 - 6 **else**
 - 7 jump to nonkeyframe state ;
-

In our experiments, the offset of the number of valid points for the switching between keyframe and nonkeyframe was set to 25. Regarding the definition of large motion, the offset of rotation was set to $\pi/2$ for all tests and the offsets of translations for EuRoC and KITTI were set to 1m and 25m, respectively.

4.2.1 Stereo Orientation Prior

We propose a sum-of-square-based stereo orientation prior (denoted as SOP) method to remove the keypoints that generate large orientation errors. The KLT

Algorithm 2: Nonkeyframe state

Data: $Img_{left}, Img_{right}, \mathbf{P}^{pre}$
Result: UAV body pose \mathbf{T}_{body} , updated 3D points \mathbf{P}

```

1 //Camera pose estimation ;
2  $\mathbf{p}' \leftarrow KLT(Img_{left}, \mathbf{p}')$  ;
3  $\mathbf{T}_{left} \leftarrow PnP(\mathbf{p}', \mathbf{P}^{pre})$  ;
4  $\mathbf{T}_{body} \leftarrow convert(\mathbf{T}_{left})$  ;
5 //Updating landmarks ;
6  $\mathbf{p}'_{add} \leftarrow gridFeatureDetection(Img_{left})$  ;
7  $\mathbf{p}' \leftarrow \{\mathbf{p}', \mathbf{p}'_{add}\}$  ;
8  $\mathbf{p} \leftarrow KLT(Img_{right}, \mathbf{p}')$  ;
9  $\mathbf{P}^{cur} \leftarrow triangulation(\mathbf{p}', \mathbf{p})$  ;
10  $\mathbf{P} \leftarrow SOP(\mathbf{P}^{cur}, \mathbf{P}^{pre}, \mathbf{p}, Img_{right}, \mathbf{T}_{right})$  ;
11 if insufficient valid points or large motion then
12   | jump to keyframe state ;
13 else
14   | data update ;

```

tracking loses some points in each new frame due to point out of view or mismatch. Therefore, the proposed method adds new keypoints in each frame. This process creates two problems: 1) the frame-to-frame VO obtains new 3D locations of old points; 2) some added keypoints are very close to the existing keypoints, as they may in fact represent the same point. We employ a simple yet effective 3D point learning process that is given by $\mathbf{P}^{new} \leftarrow (1-r)\mathbf{P}^{cur} + r\mathbf{P}^{pre}$, where r is a learning rate with a typical value of 0.1 and P_i^{new} is the new location of P_i that is computed by triangulation in the current frame. Subsequently, the algorithm evaluates the quality of the learned 3D points by stereo orientation prior. Notably, the location accuracy of the landmarks initialized in the keyframe depends upon their depth. The pose estimation may not be accurate enough if the landmarks are left unaltered in nonkeyframes.

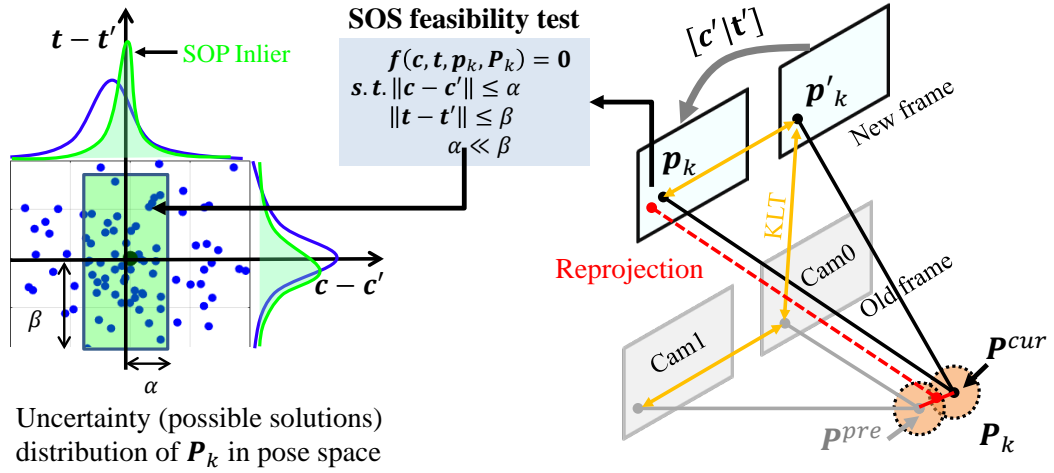


Fig. 4.3.: The main idea of SOP is simplified in 2D for illustration. Two different 3D points \mathbf{P}^{pre} and \mathbf{P}^{cur} represent the same landmark \mathbf{P}_k from old and current frames, respectively. They are updated to the \mathbf{P}_k and evaluated by SOS feasibility. If the updated point does not result in the estimation within the given feasible bounds α and β , it is considered an outlier. Thus, for each point, the SOP ensures that it contains at least one solution in the given bounds. Statistically, the uncertainty of refined inliers is controlled by $[\alpha, \beta]$ and the bias in orientation is diminished as shown.

The idea of orientation prior is defining the inliers not based on traditional reprojection error threshold but a small orientation error when reprojecting them between stereo views. The goal of using stereo orientation prior is to refine the inlier group such that the regression process can fall to a local minimum with a small error on orientation estimation. This process evaluates the quality of 3D points updated in a new frame. The main idea of the stereo orientation prior is shown in Fig. 4.3. Let $\mathbf{x} = [\mathbf{c}'^\top \mathbf{t}'^\top]^\top$ be the vector representing the 6D pose, where $\mathbf{c}' \in \mathbb{R}^3$ and $\mathbf{t}' \in \mathbb{R}^3$ are rotation and translation vectors from the left to right camera. Likewise, $\mathbf{c} \in \mathbb{R}^3$ and $\mathbf{t} \in \mathbb{R}^3$ are the estimated pose components for which an updated 2D-3D pair $\mathbf{p}_k \in \mathbf{p}$ and $\mathbf{P}_k \in \mathbf{P}$ (reprojections on right camera) generate reprojection errors (denoted as e), and $f(\mathbf{x}, \mathbf{p}_k, \mathbf{P}_k) = e$. To make use of the stereo orientation prior to avoid the error distribution bias on orientation, the proposed method gauges the quality of the 2D-3D pair $(\mathbf{p}_k, \mathbf{P}_k)$ by determining whether it can result in the PnP-based motion of the right camera being close to the calibrated stereo motion within the given tolerances, i.e., $\pm\alpha$ and $\pm\beta$ around the ground truth \mathbf{c}' and \mathbf{t}' , respectively. However, a single point can only generate 2 independent equations, which is an underdetermined problem to estimate the actual errors. On the other hand, the non-isotropic distributions of all points make modeling a single correspondence with an uncertainty distribution extremely challenging. The SOP is presented as a means to control the points' uncertainty in the reverse way. This is solved by a SOS feasibility check.

Remark 1: Formally, this work aims to find all inlier pairs $\{(\mathbf{p}_m, \mathbf{P}_m) | m \in \text{inliers}\}$ that satisfy:

$$\begin{aligned} f(\mathbf{x}, \mathbf{p}_m, \mathbf{P}_m) &= 0, \\ \|\mathbf{c} - \mathbf{c}'\| &\leq \alpha, \quad \|\mathbf{t} - \mathbf{t}'\| \leq \beta, \quad \alpha \ll \beta, \end{aligned} \tag{4.1}$$

where $f(\mathbf{x}, \mathbf{p}_m, \mathbf{P}_m)$ is the 3D to 2D projection. α and β define the variation around \mathbf{c}' and \mathbf{t}' , respectively. Note that they are the lower and upper bounds on x_j , i.e., $x_{jl} \leq x_j \leq x_{ju}$. This condition can be written as an inequality in a second-degree polynomial as follows:

$$g_j(x_j) = (-x_{jl} + x_j)(x_{ju} - x_j) \geq 0. \tag{4.2}$$

Thus, for each known 2D-3D pair $(\mathbf{p}_k, \mathbf{P}_k)$, our primal problem is to solve the feasibility problem:

$$\begin{aligned} f_i(\mathbf{x}) &= 0, i = 1, 2, 3, \\ g_j(\mathbf{x}) &\geq 0, j = 1, \dots, 7, \end{aligned} \tag{4.3}$$

where f_i is a quadratic polynomial. Each correspondence generates 3 quadratic polynomials (details are given in the Appendix). $\{g_j | j = 1, \dots, 6\}$ are obtained from the pose variable bounds, and $\{g_j | j = 7\}$ is the constraint to ensure that the depth of points is positive. Note that all of these constraints are quadratic polynomials.

The dual problem of (4.3) is defined as follows: if there exist $t_i \in \mathbb{R}$ and $s_i \geq 0$ such that $F(\mathbf{x}) - G(\mathbf{x}) \succeq 0$ with,

$$\begin{aligned} \text{Ideal} : F(\mathbf{x}) &= \sum_{i=1}^3 t_i f_i, \\ \text{Cone} : G(\mathbf{x}) &= s_0 + \sum_i^7 s_i g_i + \sum_{i \neq j}^7 s_{ij} g_i g_j \\ &\quad + \sum_{i \neq j \neq k}^7 s_{ijk} g_i g_j g_k + \dots, \end{aligned} \tag{4.4}$$

where $G(\mathbf{x}) \succeq 0$. In this process, we test the feasibility contradiction between the primal problem and dual problems. If the primal problem is feasible, we have $F(\mathbf{x}) = 0$, which leads to $F(\mathbf{x}) - G(\mathbf{x}) \succeq 0$. Therefore, the dual problem cannot be feasible.

Remark 2: For pose estimation with a set of 2D-3D correspondences $\{(\mathbf{p}_k, \mathbf{P}_k) | k = 1, 2, \dots\}$, our goal is to minimize the overall reprojection error within the bounds, which is given by:

$$\begin{aligned} \min_{\mathbf{x}, \mathbf{W}} \quad & \mathbf{W} \cdot \text{diag}\left(\left\{\sum_{i=1}^3 f_i(\mathbf{x}, \mathbf{p}_k, \mathbf{P}_k) | k = 1, 2, \dots\right\}\right) \\ \text{s.t.} \quad & g_j(\mathbf{x}) \geq 0, j = 1, \dots, 7, \\ & \mathbf{W} \in \max_{\mathbf{W}} \sum \text{diag}(\mathbf{W}), \end{aligned} \tag{4.5}$$

where \mathbf{W} is initialized as an identity matrix to indicate the inliers. Thus, if correspondences $(\mathbf{p}_k, \mathbf{P}_k)$ are outliers, the system eliminates its impact by setting the k -th diagonal element of \mathbf{W} to 0. As the system also needs to maximize the inlier group, the problem becomes bilevel optimization. However, if the VO system ignores \mathbf{W} and

directly computes the optimal solution of this problem, which can be solved by converting to its Lagrangian dual problem (LP), the outliers may have a large effect on the result. If rejecting the outliers by iteratively finding the solution of \mathbf{x} with subset data and determining whether it satisfies the boundary condition (such as RANSAC), the system may mistakenly reject many inliers due to a suboptimal initial condition of regression. Therefore, we expect to determine whether every single point has the potential to be an inlier. Thus, the system does not minimize the cost but considers the minimal solution ($f_i(\mathbf{x}) = 0$) as the constraint (equation (4.3)). It becomes an underdetermined problem with infinite solutions, and we convert it to the feasibility evaluation. The nonnegativity of the quadratic polynomial $H(\mathbf{x}) = F(\mathbf{x}) - G(\mathbf{x})$ ensures that the 2D-3D pair is an outlier for the given bounds. We test this condition using the theory of sum-of-squares, which is known as Hilbert's 17th problem. This problem has been systematically solved in the state of the art, and studies show that any positive polynomial of degree 2 can always be represented as an SOS. We illustrate how this theory helps the outlier rejection in Fig. 4.2.

Recall that we obtain 3 equations/polynomials of degree 2 from each correspondence from equation (3.6) in previous chapter, with only two of them being linearly independent. The variables of these polynomials are Cayley's parameters \mathbf{c} and new translation \mathbf{t}_{new} . From equation (4.2), we obtain 6 constraints ($g_i(\mathbf{x}) \geq 0, i = 1, \dots, 6$) on the 6 DoF poses. In addition, we introduce the scale factor in equation (3.6):

$$\lambda(\mathbf{I} - [\mathbf{c}]_{\times})\mathbf{p}_k = (\mathbf{I} + [\mathbf{c}]_{\times})\mathbf{P}_k + \mathbf{t}_{new}, \quad \lambda \geq 0. \quad (4.6)$$

By eliminating the variable λ , we obtain the constraint $g_7(\mathbf{x}) \geq 0$ to ensure the positive depth of each point.

Having built the dual problem system in equation (4.4), it can use the LMI solver to find the values of t and s for ideal and cone. All polynomials in ideal and cone are quadratic polynomials. Let matrices \mathbf{G}_{fi} and \mathbf{G}_{gj} be 7×7 symmetric matrices such that:

$$f_i = \mathbf{y}^T \mathbf{G}_{fi} \mathbf{y}, \quad g_j = \mathbf{y}^T \mathbf{G}_{gj} \mathbf{y}, \quad (4.7)$$

where $\mathbf{y} = \begin{bmatrix} \mathbf{c} \\ \mathbf{t}_{new} \\ 1 \end{bmatrix}$. The details of how the matrices \mathbf{G} are generated are provided in the appendix. If there exist $G \succeq 0$, the SOS problem is feasible. Then, we solve equation 4.4 using linear matrix inequality (LMI). The LMI system is given by:

$$\sum_{i=1}^3 t_i \mathbf{G}_{fi} - \sum_{j=1}^7 s_j \mathbf{G}_{gj} \succeq 0, \quad (4.8)$$

where $t \in \mathbb{R}$ and $s \geq 0$. The feasibility of this problem can be tested directly using semidefinite programming (SDP). For C++ implementation in ROS, we use SDPA-C as the LMI solver. The SDP problem is given by:

$$\begin{aligned} \min_h \quad & \mathbf{w}^T \mathbf{h}, \\ \text{subject to} \quad & \mathbf{H} \succeq 0. \end{aligned} \quad (4.9)$$

In the case of the LMI problem Eq. 4.8, $\mathbf{h} = [t_1, t_2, t_3, s_1, s_2, s_3, \dots]^T$ and $\mathbf{w} = [0, 0, 0, 1, 1, 1, \dots]^T$. In addition, the problem is subject to:

$$\mathbf{H} = \sum_{i=1}^3 t_i \mathbf{G}_{fi} + \sum_{j=1}^7 s_j (\mathbf{I} - \mathbf{G}_{gj}) \succeq 0. \quad (4.10)$$

The SDP is time-consuming even in C++. To achieve real-time processing, we introduce optimistic and pessimistic reprojection error thresholds to reduce the number of points in stereo orientation prior to outlier rejection. We show the algorithm flowchart in Fig. 4.4. If the reprojection error of a point is less than pessimistic, it is definitely an inlier. In the same way, an error greater than optimistic definitely implies an outlier, and the rest are potential inliers. Therefore, the prior stereo orientation only needs to check the points that are potential inliers. The typical values for pessimistic and optimistic images are between 1-3 and 3-6 pixel distances, respectively. They are selected based on the camera sensor noise and the uncertainty of KLT tracking [93].

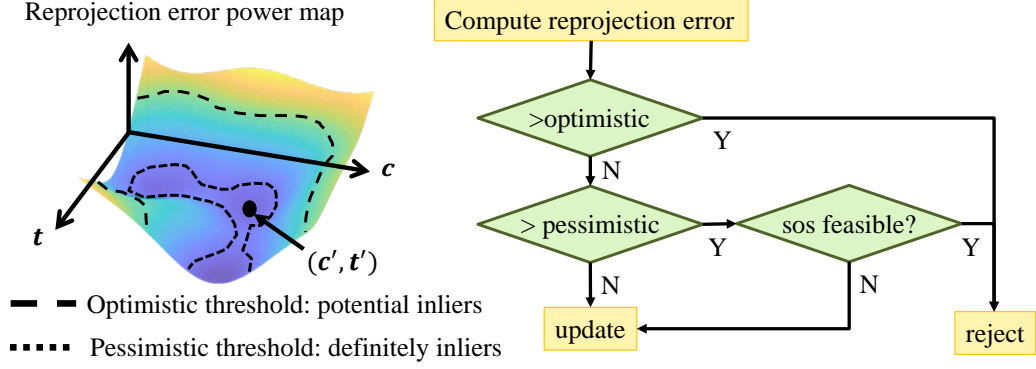


Fig. 4.4.: Stereo orientation prior based outlier rejection. The error power map is simplified in 2D (1D rotation and 1D translation) for illustration. Optimistic and pessimistic reprojection error thresholds preliminarily identify the points that are definitely inliers or outliers, while the remaining points are regarded as potential inliers and evaluated by stereo orientation prior.

4.3 Experiments

In this section, we first evaluate the overall performance of the proposed method on the EuRoC MAV benchmark. Then, we demonstrate whether and how the stereo orientation prior contributes to the odometry estimation. Then, we test the proposed method on the datasets recorded by a UAV equipped with Intel RealSense T265. Some algorithms, such as SLAM-based approaches, eliminate cumulative error by closing the loop; however, loop closure detection is not generally considered part of the VO framework [73, 77, 80, 84]. Among them, [80] compared top-ranked state-of-the-art mono VIO methods on the EuRoC benchmark using absolute trajectory error (ATE) [94] on Intel NUC (Ubuntu 16.04 and ROS Kinetic), which is a common onboard computer for UAVs. In their evaluation, the loop closure detection was deactivated for all methods, and the results were aligned with ground-truth traces and measured by root mean square error (RMSE). Both the results alignment and error measurement were strictly referred to [94] to allow a fair comparison. Therefore, all experiments were conducted with the same protocol to evaluate the proposed method.

4.3.1 Benchmark Evaluation

We compared the proposed algorithm against the state-of-the-art VO/VIO methods on the EuRoC MAV benchmark. The datasets consisted of a VICON room and a machine hall with pure static scenes, which simulated building an indoor environment. For datasets with ID 01 or 02, the camera motion was smooth, and the environment contained complex texture. In contrast, the UAV flew aggressively, and there were fewer detectable patterns in other datasets. Many methods failed to detect and track keypoints on dataset "Vicon Room 2 03" because of the insufficient feature points, aggressive motion, and brightness inconsistency, which was also reported in [84]. The results of all datasets are shown in Tab. 4.1, and the details of the two sampled results are shown in Fig. 4.5.

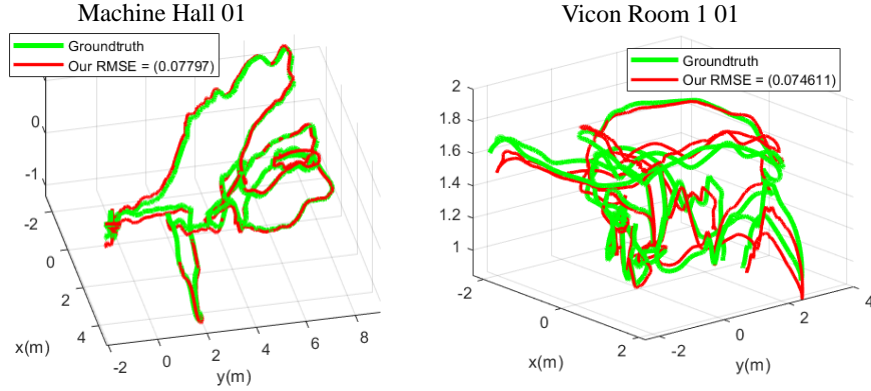


Fig. 4.5.: Trajectory comparison results sampled from EuRoC benchmark evaluation (Tab. 4.1).

Results on NUC5i7RYH: In [80], an Intel NUC with a dual-core Intel Core i7-5557U CPU @3.10 Hz, 16G RAM was used. Therefore, we test our ROS C++ version code on the same platform, i.e., NUC Kit NUC5i7RYH (Fig. 4.11), but with 8G RAM. As we found that the results of the proposed algorithm had a minor difference between using Kinetic default OpenCV and OpenCV 4 with the same parameters, we provide both of the results for references. The boundary conditions were fixed as $\alpha = 0.01$ and $\beta = 0.5$ for all EuRoC datasets. The 0.01 boundary variation in

Sensor Dataset	Mono + IMU *						Stereo + IMU	Stereo VO		
	SVOMSF	MSCKF	ROVIO	VINS-MONO	SVOTSAM	OKVIS	S-MSCKF	SVO2.0	SOPVO	SOPVO [†]
Vicon Room 1 01	0.39	0.29	0.10	0.07	0.12	0.09	0.07	0.06	0.07	0.06
Vicon Room 1 02	0.63	0.20	0.10	0.10	0.16	0.18	0.13	0.08	0.08	0.35
Vicon Room 1 03	×	0.67	0.14	0.13	×	0.24	0.20	0.24	0.17	0.39
Vicon Room 2 01	0.17	0.11	0.12	0.08	0.07	0.12	0.07	0.08	0.12	0.19
Vicon Room 2 02	0.37	0.16	0.14	0.08	×	0.17	0.18	0.20	0.21	0.19
Vicon Room 2 03	×	1.13	0.14	0.21	×	×	0.24	×	×	×
Machine Hall 01	0.22	0.43	0.21	0.27	0.08	0.21	×	0.11	0.08	0.10
Machine Hall 02	0.20	0.43	0.25	0.12	0.05	0.20	0.15	×	0.09	0.08
Machine Hall 03	0.60	0.25	0.25	0.13	0.12	0.25	0.29	0.36	0.24	0.27
Machine Hall 04	1.82	0.61	0.49	0.23	0.24	0.49	0.23	2.4	0.24	0.33
Machine Hall 05	0.93	0.48	0.52	0.34	0.16	0.56	0.29	1.20	0.33	0.50
Average / fail	0.44 / 3	0.40 / 1	0.24 / 0	0.17 / 0	0.12 / 3	0.25 / 1	0.19 / 1	0.16 / 4	0.16 / 1	0.25 / 1

Table 4.1.: EuRoC MAV visual odometry benchmark evaluation. The results that achieved the best three levels of accuracy are marked by red, green, and blue. Except for algorithm termination (marked by ×), the RMSE > 1 was also considered as a fail. *The results of SVOMSF, MSCKF, ROVIO, VINS-MONO, SVOGTSAM, and OKVIS were obtained from [80] (NUC results). Other methods were evaluated using the same metric, i.e., real-time running on NUC ROS Kinetic, no loop closure detection nor bundle adjustment. The final results of each dataset were the averages of 10 runs. † ROS Kinetic uses OpenCV 3 by default. In OpenCV 4 the ORB detector is modified and it generates slightly different results using the same parameters.

the Cayley parameters was [1.1573, 1.1342, 1.1573] degrees on XYZ at Euler angles, which means that the system allowed the estimated right camera pose to have one degree of error on each orientation axis around the baseline in the worst cases. In our test, this restrictions are narrow enough to guide the optimizer to fall into a local minimum with accurate orientation estimation. Note that the translation boundary was $\mathbf{t}_{new} = (I - [\mathbf{c}]_{\times})\mathbf{t}$; refer to equation (??). Therefore, the boundary on actual translation varied within a small interval dependent upon the actual rotation. The pessimistic and optimistic thresholds were fixed to 2 and 5, respectively. The final result of each dataset was the average of 10 runs. The comparison of algorithm efficiency, including CPU usage and FPS, is given in Fig. 4.6.

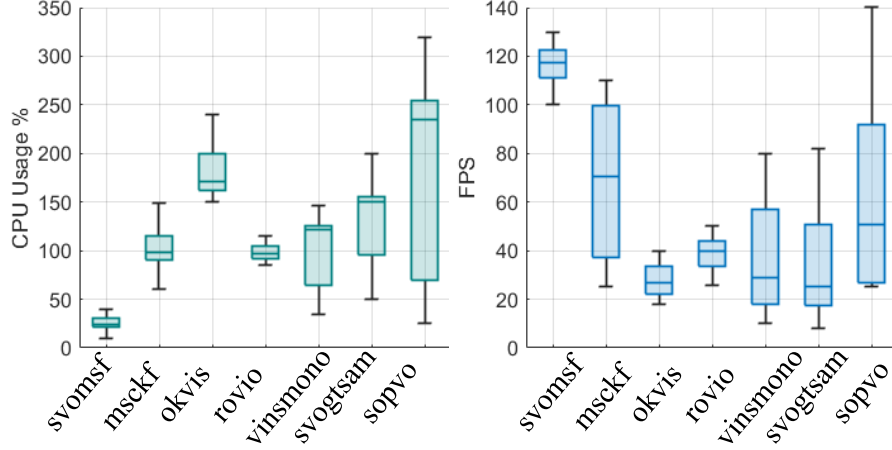


Fig. 4.6.: CPU usage and FPS on Intel NUC5i7RYH. The results of all compared methods are obtained from [80].

The algorithm was run on approximately 50 FPS on average, ranging from 20 to 140+ FPS for a single frame; additionally, the CPU usage varied from approximately 50% to 300% because it depended on how many points were evaluated by stereo orientation prior. It took approximately 0.5 ms per correspondence using NUC5i7RYH. As the algorithm only stores the landmarks of a current keyframe, only 300 M additional memory was used. The results indicate that the proposed method achieved similar performance to the state-of-the-art mono VIO methods ROVIO, VINS-MONO, and stereo VIO methods S-MSCKF, OKVIS in terms of accuracy and robustness without using IMU. Section 4.3.2 provides a detailed analysis of whether and how prior stereo orientation works.

4.3.2 Contribution of Stereo Orientation Prior

To fairly compare the results of using the proposed method with the traditional method, i.e., only reprojection error, we designed the experiment as Fig. 4.8 shows. The stereo orientation prior evaluated the points at which the reprojection errors were between pessimistic and optimistic thresholds, which were fixed as 2 and 5 for the

EuRoC benchmark test. Therefore, for outlier rejection using only reprojection error, the stereo orientation prior to checking was blocked, and pessimistic thresholds from 1 to 4 were used. From the results, we found that the best threshold for reprojection error was a 2 pixel distance, which was close to the image noise level, while a smaller threshold rejected too many points and a larger threshold could not filter outliers properly. The results indicated that the stereo orientation prior could reduce the overall odometry estimation error from 18.5% to 30% on average compared with the outlier rejection using only reprojection error. In our implementation, the SOP model was implemented on a basic pipeline of keyframe-based visual odometry. As the SOP module could improve the performance of the simplest VO pipeline, we believe it is strong evidence to show the contribution in this work.

To further support the effectiveness of the proposed method, we show the absolute value of the yaw Euler angle in comparison with the ground truth in Fig. 4.7. Because the ground truths in Machine Hall datasets were obtained by laser tracker, which only gave the 3D position, we show the results of 3 Vicon Room datasets, ranging from easy to difficult. To avoid the chaos of plots, we chose the yaw angle because it contains the major UAV motion. On dataset 01, the difference between using the stereo orientation prior and the traditional method was not noticeable because the motion of the UAV was quite smooth and the scene contained abundant high-quality features. As the UAV flew more aggressively in a featureless environment (Vicon Room 03 datasets in EuRoC), the proportion of outliers increased, which misled the optimization process to fall into a suboptimal solution. Thus, the orientation prior started showing its contribution to the estimation by guiding the optimizer to the local minimum with less orientation error. Notably, the orientation error amplified the trajectory error over time, which means that even a minor improvement may result in a large difference.

To provide intuitive results, we demonstrate one well-known KITTI sequence, i.e., *sequence00*, to show whether the prior stereo orientation can better estimate the orientation and the impact of the boundaries α and β in equation (4.1). KITTI [95] visual

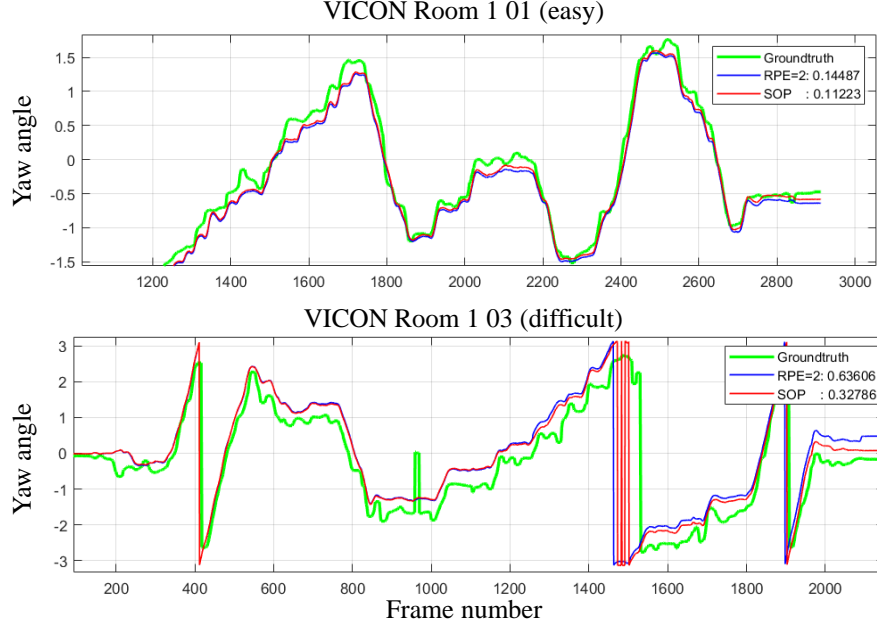


Fig. 4.7.: The absolute value of the yaw Euler angle in comparison with the ground truth. The orientation estimation error comparison using SOP and only reprojection error (RPE) for outlier rejection. The results were filtered by a median filter because the ground truth data given by VICON contained some abnormal peaks. The results indicate that the proposed method provided a better orientation estimation, especially when keypoint detection and tracking became difficult. We compared SOP with $RPE = 2$ because this threshold generated the best odometry estimation over other thresholds (refer to Fig. 4.8). The angle MSE (given in the legend) was computed from frame 2,000 to the end to avoid the impact of the value jump between $-\pi$ and π .

odometry datasets were recorded for autonomous driving. Therefore, the odometry was based on 4 DoF poses, and the datasets were outdoor environments with dynamic scenes. As the odometry results in KITTI were mainly on a 2D plane and the traces were artificial patterns (squares and grids), we believe it is a straightforward method for demonstrating the performance of stereo orientation prior. In this experiment, we set the pessimistic and optimistic thresholds to 3 and 6, respectively, as we found

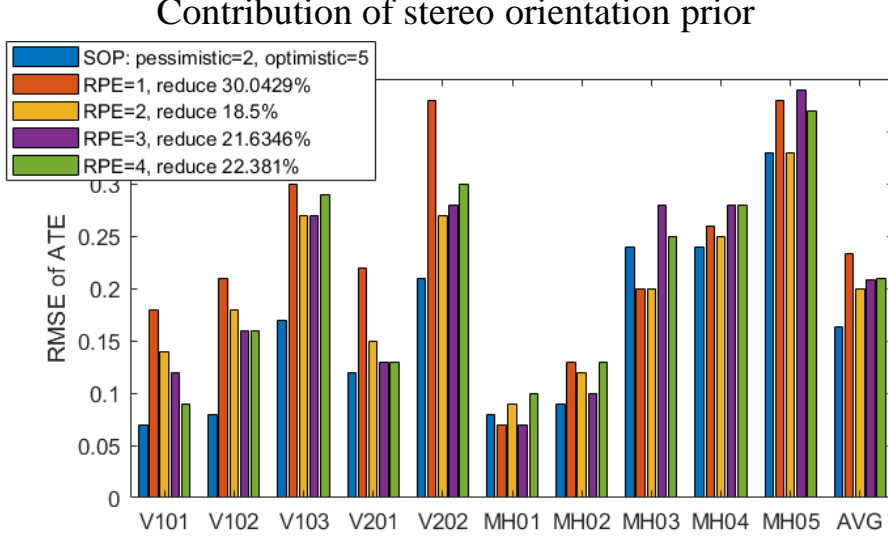


Fig. 4.8.: The ATE performances on the EuRoC benchmark using SOP and only RPE. While the outlier rejection using only RPE achieved its best performance by setting the threshold as the typical image noise level (2-pixel distance), the stereo orientation prior reduced its error by 18.5%.

that the detected points in the KITTI dataset were coarse because the scene contains highly chaotic and repeated patterns, such as trees, the surface of the road, and shadows, and the reprojection error rejection was certain to fail with a smaller threshold. For the stereo orientation prior, we fixed the boundary of orientation $\alpha = 0.1$ and translation $\beta = 0.5$. Because the image resolution was quite high in KITTI sequences, the proposed method could not achieve real-time performance with NUC. All tests in KITTI were carried out on a PC with ROS Melodic in real time, and no loop closure detection or bundle adjustment was used during the experiments so that we could guarantee that the visual odometry performance was the only factor of the results. In Fig. 4.9, the stereo orientation prior significantly enhanced the estimation, especially on orientation. Another experiment (Fig. 4.10) showed how better orientation and overall trajectory estimation are correlated to the orientation boundary.

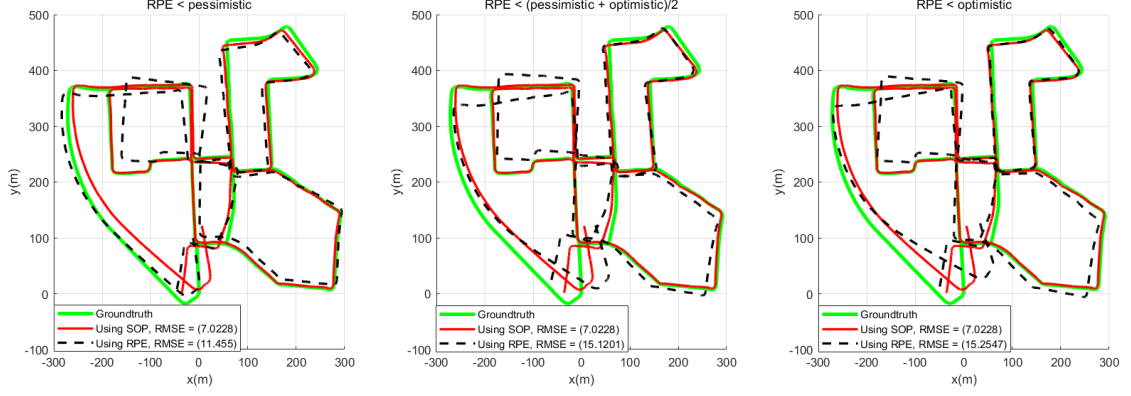


Fig. 4.9.: Reprojection error (RPE) vs stereo orientation prior (SOP): three different reprojection error thresholds were used to reject the outliers: pessimistic, optimistic, and $(\text{pessimistic} + \text{optimistic})/2$. SOP evaluated the points between pessimistic and optimistic. For SOP, the boundary conditions were fixed as $\alpha = 0.1$ and $\beta = 0.5$. The results indicated that, regardless of how we changed the RPE, the orientation estimation was not as good as that using SOP.

4.3.3 UAV Indoor Flight Test

The onboard tests were conducted on UAVs equipped with Intel NUC or UP Board for the proposed method, which are shown in Fig. 4.11. The flight tests were conducted in a VICON system with texture scenes and public indoor environments. The stereoview was obtained from an Intel RealSense T265. To achieve real-time performance (at least 10 FPS vision pose input for flight controller) on the UP Board, as well as eliminate the scene of the UAV propeller, the system cropped and resized the 848×800 resolution fisheye view from T265 raw input into the 300×300 undistorted image, as shown in Fig. 4.12. With the low-resolution stereo view, the proposed method ran on approximately 12 and 100 average FPS on UP Board and NUC, respectively.

VICON test: the F250 FPV quadcopter was flown fully autonomously under the VICON system. The route of the flight was set to a square. In addition, we

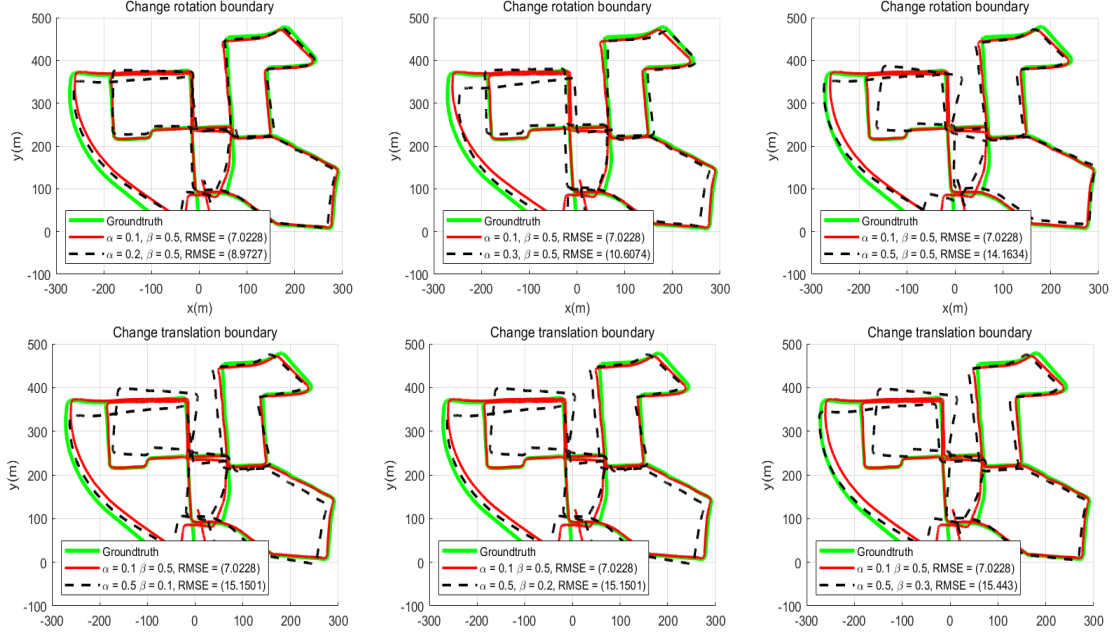


Fig. 4.10.: Orientation or translation prior: we adjusted the boundary condition to further evaluate whether a small orientation boundary was the main contributor, as we analyzed. In the first row, we changed the orientation boundary α while fixing the translation boundary β , and vice versa for the second row. It is clear that when we reduced the orientation boundary, the accuracy of orientation estimation increased, resulting in improvement in the overall odometry estimation. The translation boundary did not have the same level of impact on the results.

compared the proposed method with T265 inside-out tracking, and the results are shown in Fig. 4.13 and Tab. 4.2. The average RMSE indicates that the proposed method achieved odometry estimation accuracy similar to that of T265.

Public indoor environment test: The SK520 Quadcopter was not flown but held by hand for safety reasons. The underground carpark and a square corridor were selected for testing because there were location references for us to evaluate the estimation results. Because we do not have the floor plan of the carpark, we manually measured the length of the route refer to the centre line on the road, which was used as groundtruth(GT). The route start from a point about 11m away in the front of

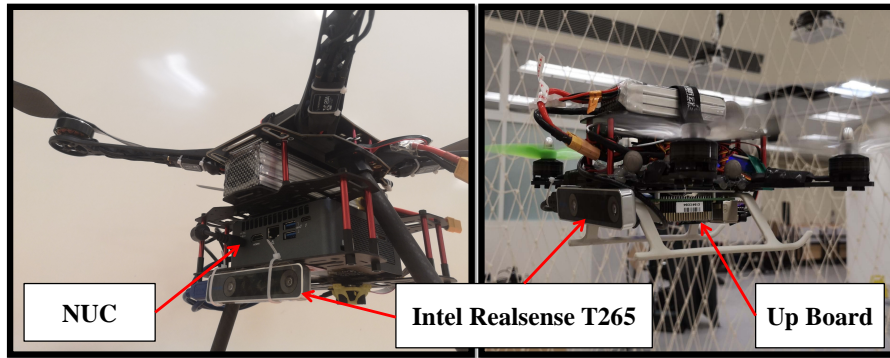


Fig. 4.11.: Testing platforms. Left: SK520 Quadcopter with NUC; Right: QAV250 FPV Quadcopter with UP Board.

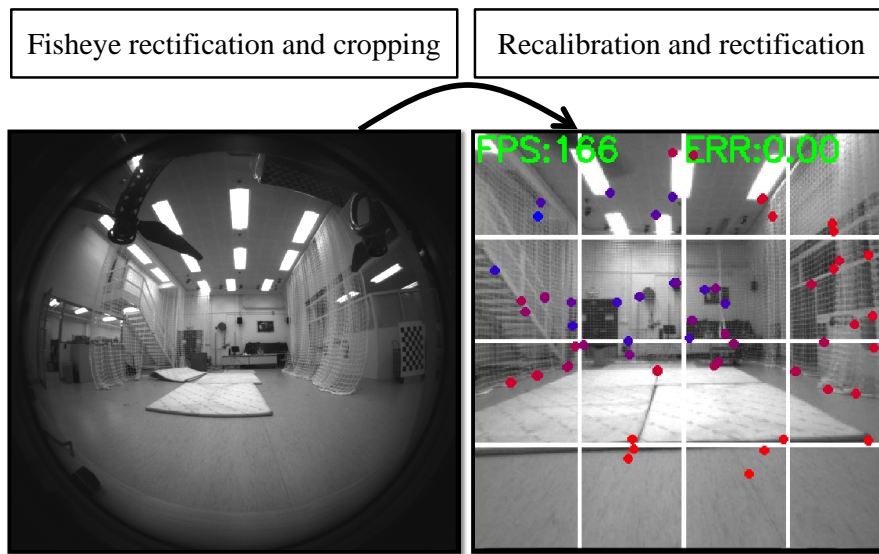


Fig. 4.12.: The fisheye view converted into undistorted images. The left figure shows the T265 fisheye view from our SK520 drone. Because the fisheye has more than 160 degrees of FOV with 848×800 resolution, some parts of the UAV, such as propellers, occupied a large area of view. The right figure shows the grid feature detection of the proposed method using the cropped and resized image.

the first corner, where we can align the speed bump with the centre of our image view. We plot the estimated trajectory in ROS RVIZ, where the size of each grid is

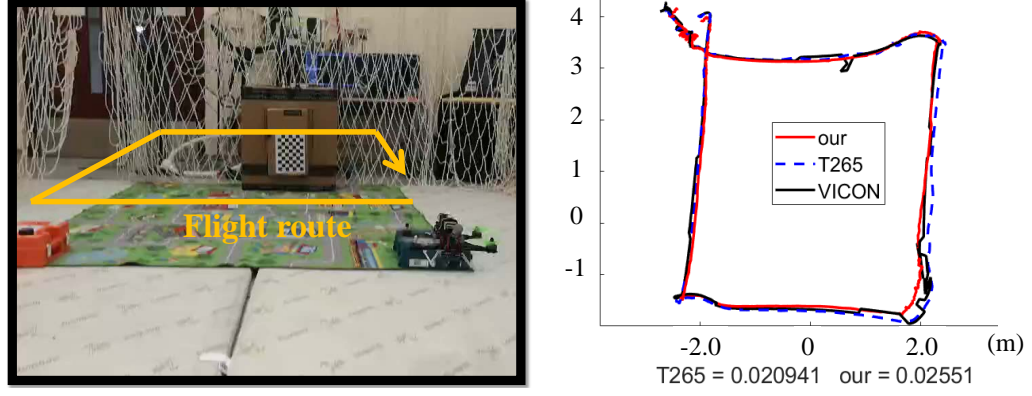


Fig. 4.13.: VICON flight test. The F250 drone flew by VICON, while the proposed method was run onboard (Intel UP Board) at an average of 12 FPS.

Method	T265 inside-out tracking	Our
Sensor	Stereo + IMU	Stereo
Resolution	848×800	300×300
Mean RMSE (m)	0.020	0.023

Table 4.2.: Average RMSE evaluated by VICON.

1m. The right side shows the final results of the proposed method and the estimation from Intel T265. The camera halo effect due to the light sources might be the reason that Intel T265 underestimated the distance. Considering the measurement error of the groundtruth ($\pm 0.5m$), the drift of proposed method is less than 0.5m over 116m route ($< 0.4\%$). Videos can be found in our Github page. In Fig. 4.15 we show another result in a square corridor environment. All the results indicate that the proposed method could provide accurate estimation (translation error $\leq 0.4\%$ for the motion on horizontal plane and $\leq 5\%$ for height estimation). Upper row shows the floor plan of the corridor and our walk route. The side length of the square corridor is 13.2m. The floor height down stairs is approximately 4m. We start from the stairs, walk along the square corridor and back to the start point, then go down

stairs. Lower row visualizes the estimated path and all tracked landmarks over the odometry estimation. From the top view we can find that the point cloud shows the proper square shape with about 13m side length, which indicate the accurate odometry estimation on the xy plane. The side view shows odometry estimation on height. It is clear that the odometry estimation on the walk along the square is almost on the same plane. When going down stairs, the proposed method can estimate the height variation within 20cm error ($< 5\%$).

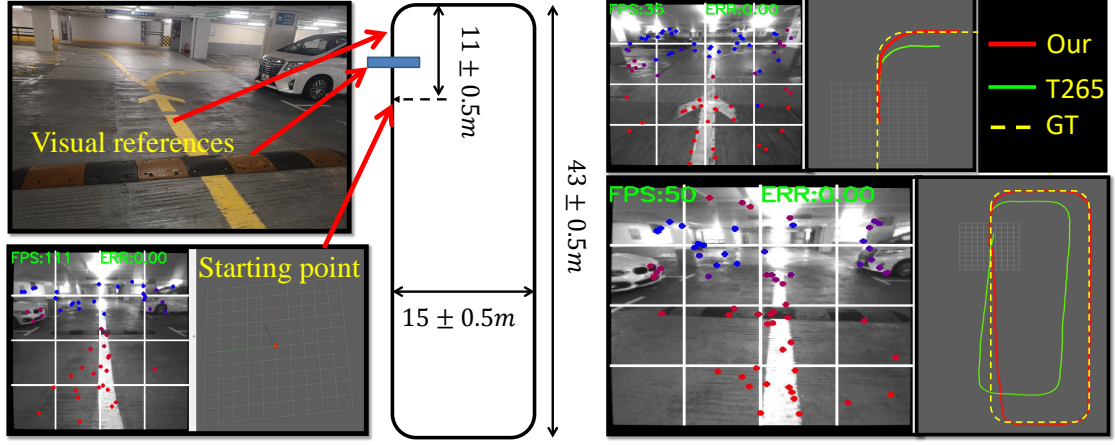


Fig. 4.14.: Carpark test: The SK520 drone was handheld for safety reasons, while the proposed method ran online on its onboard computer (NUC5i7RYH). The route started from a point approximately 11 m away in the front of the first corner, where we aligned the speed bump with the center of the image view. The estimated trajectory was plotted in the ROS RVIZ, where the size of each grid was 1 m. The right side shows the final results of the proposed method and the estimation from Intel T265. The camera glare effect due to the light sources might be the reason that the Intel T265 underestimated the distance. Considering the measurement error of the ground truth ($\pm 0.5m$), the drift of the proposed method was less than 0.5 m over the 116 m route ($< 0.4\%$).

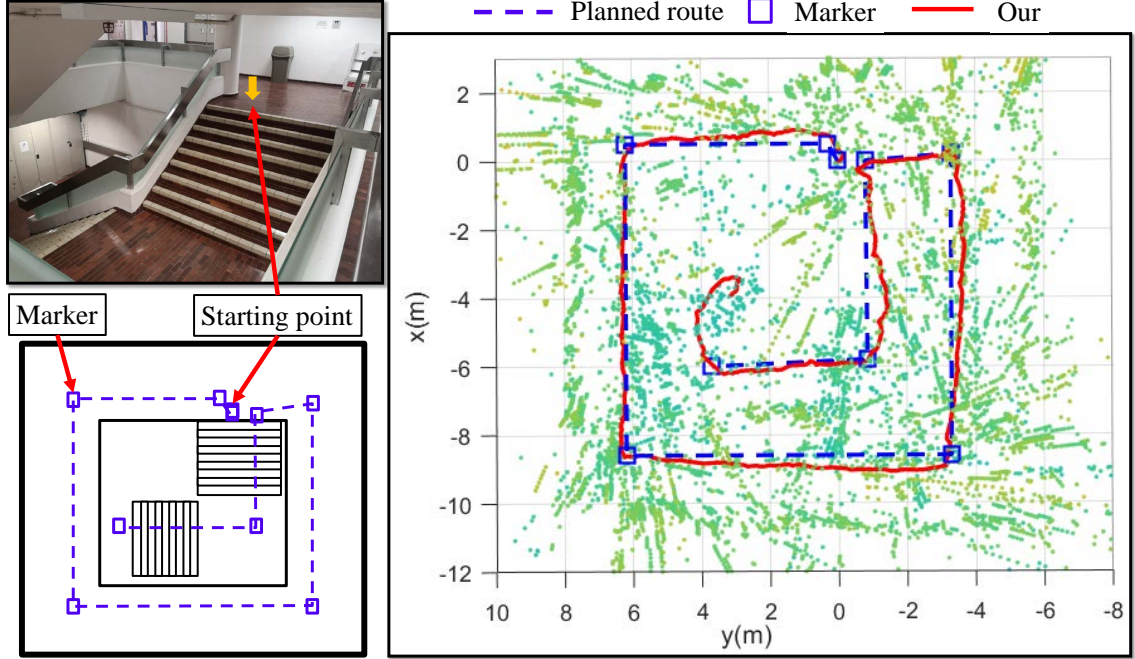


Fig. 4.15.: Corridor test: The bottom-left shows the planned route. We marked the corner points of the route in real world as groundtruth because without motion capture system, we can only ensure the accurate measurements when reaching these markers during walking. The estimation errors were measured by the drifting distances of these corners. The average drift was within 20 cm ($< 5\%$ of overall distances). The right plot visualizes the estimated path and all tracked landmarks over the estimation history.

4.4 Different between SOPVO and CamAdj

In CamAdj chapter, we made an assumption: there exist low-reprojection-error inliers, denoted as singular points, that negatively affect the nonlinear refinement of pose estimation. The comparison of the fundamentals between CamAdj and SOPVO is given in the following figure (Fig. 4.16). To obtain a deeper understanding of theoretical parts, we introduced two scenarios. Scenario 1 address the camera pose estimation for general stereo SfM or VO problem. SOPVO is an extension of the scenario 1 while based on the new assumption to explore the potential of the SoS

filtration technique. To verify the new assumption, we have modified the workflow of the VO accordingly. Scenario 2 is more related to the 3D reconstruction problem, which aims to explore the capability of using a similar approach on different applications.

In SOPVO, different assumption is made to address the real-time VO problem. We look for some large-reprojection-error points that adding them into pose estimation results in an increase in translation error and a decrease in the orientation error. The motivation for seeking these points is that we believe it is a meaningful trade-off. The accuracy of orientation has a relatively larger impact on the long-term trajectory estimation. Moreover, more inliers would increase the system's robustness. Therefore, the inlier/outlier filtration in SOPVO is done in all non-keyframes and the goal is to find more potential inliers. During this process, all tracked points will be updated based on new triangulation results and new keypoints are detected. To identify the potential inliers, we use two different reprojection thresholds (optimistic and pessimistic), only the correspondences with reprojection errors in between are evaluated by the proposed method and other points are definitely inliers or outliers. The experimental results show pieces of evidence that our assumption could be true.

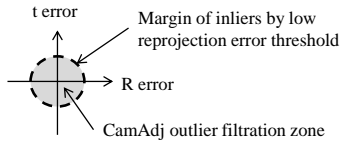
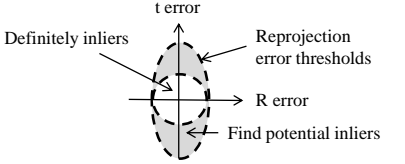
	Data group	Assumption	Approach
CamAdj	2D-3D or 2D-2D correspondences with low reprojection errors.	Exist singular points that negatively affect nonlinear refinement of pose estimation.	
SOPVO (paper [1])	2D-3D correspondences with large reprojection errors.	Some of them can still contribute to accurate orientation estimation because they generate errors mainly on translation.	

Fig. 4.16.: Fundamentals of CamAdj and SOPVO.

5. VISUAL OBJECT TRACKING FOR TASK-ORIENTED FLYING

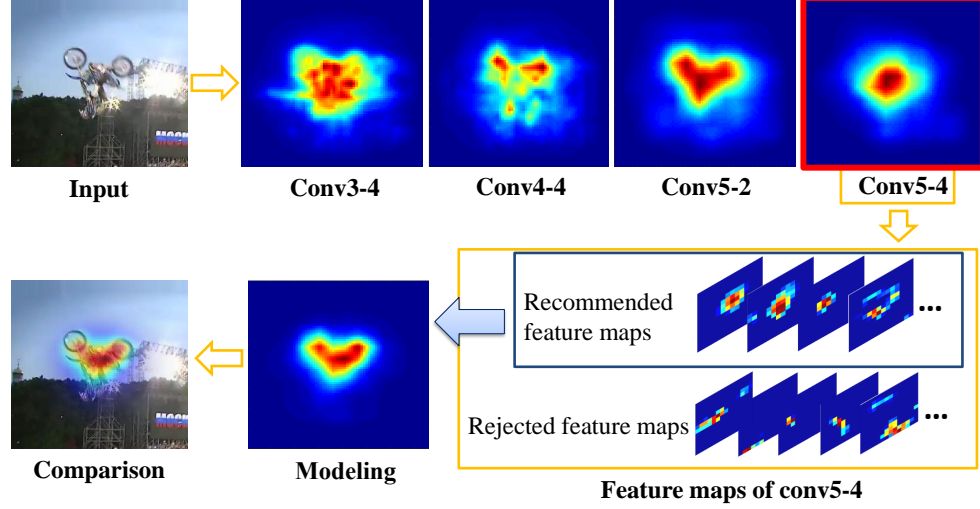


Fig. 5.1.: Online CNN layer and feature map recommendation. The layer, in which the highest convolutional responses gather at target center, is marked by a red box. The target percept reconstructed from the recommended feature maps, on the other hand, has a better description regarding the target's central location, shape, and size.

Vision intelligence has achieved significant progress fueled by the deep learning method, however, the progress becomes much slower in UAV applications due to the limitation of the hardware platform. The drones have to process many tasks concurrently, such as navigation, target detection, tracking, path planning, you name it. Unfortunately, the powerful GPU usually is not a feasible option for small UAVs or MAVs. With limited computational resources, we need to optimize the onboard system.

The main idea of the convolutional feature recommendation is illustrated in Fig. 5.1. We use VGG-19 as the backbone. The features of tracking target extract from dif-

ferent layers are given in the first row. When layer goes deeper, the semantic level of feature become higher. But we also lose the location information. Therefore, low layer features should also be used for tracking. Dr. Chao Ma's work HCF [18] uses a similar idea to perform visual tracking. In their work, however, they use fixed layers and all feature maps of those layers to rebuild the target appearance model. We argue that different target has different semantic level. For a simple target, we can use only a few layers to represent the target properly, and not all feature maps are necessary as some of them are background features.

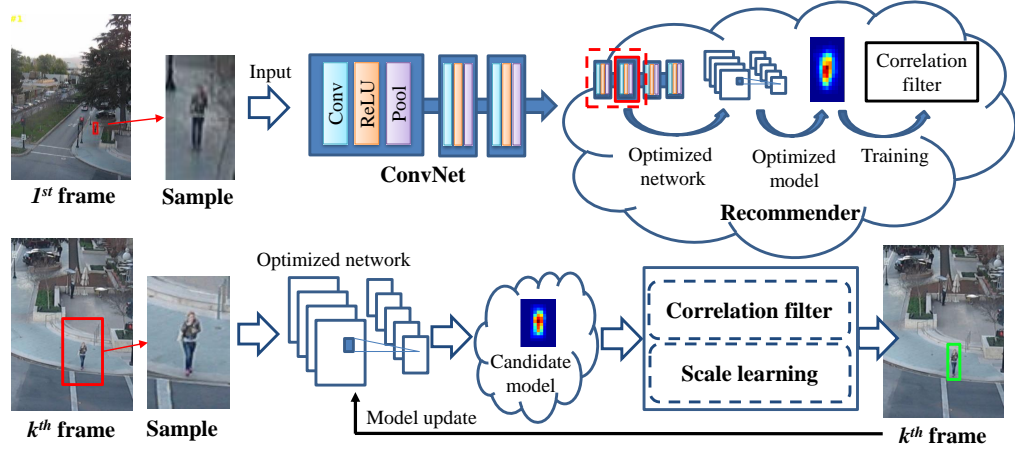


Fig. 5.2.: Tracking framework overview

5.1 Proposed Algorithm

In this section, we first give an overview of the proposed method, of which the framework is shown in Fig. 5.2. The target appearance is given in the first frame. Initially, the proposed recommender optimizes the VGGNet [96] network by finding the highest layer we actually need and build the optimized target model for the correlation filter training, which is discussed in Section 5.1.1 and Section 5.1.2, respectively. In each new frame, we take a sample patch that is an extended region of the target region in the last frame and feed it to the optimized network - to extract the convolutional features. Then we use the proposed recommender to build the candidate

model. The correlation filter works together with the proposed scale learning method to relocate the target and update the new model (Section 5.1.3 and Section 5.1.4). Finally, we summarize the tracking framework using pseudocode.

5.1.1 Recommender

Denote \bar{F} and \bar{B} as the mean response value of foreground region F (given by bounding box) and background region B (the extended searching region) of a convolutional response $X(x_1, x_2, \dots, x_N)$, respectively. The recommendation score of X is given by:

$$f(X, F, B) = DT \cdot GT, \quad (5.1)$$

where the DT and GT are distinctive term and gain term, respectively. They are defined as:

$$DT = \frac{1}{N} \sum_{i=1}^N x_i (e^{1 - (\frac{2d_i}{(1+\beta)r})^2} - 1), \quad (5.2)$$

$$GT = (\bar{F} - \bar{B})^2, \quad (5.3)$$

where d_i is the pixel distance between i -th pixel to the target center and r is the diagonal length of region F . β is the tolerance parameter. x_i is the convolution response value of i -th pixel. As Fig. 5.3 shows, DT scores the feature quality, while GT measures the statistical difference between foreground and background. Fig. 5.4 shows the recommender output.

The high response of the convolution, i.e., $i \in \{i | x_i \geq \delta\}$, means the local appearance is highly correlated to the image filter (or kernel) that learned by CNNs. In this work, we are interested in the peak responses ($\delta = \max(X)$) because our goal is single object tracking.

Remark 1: The typical value for tolerance parameter β is 0.25, which means we assume that the location shift or scale growing of the target is within 25% target size in general. It is introduced because the target may have a position shift or scale variance.

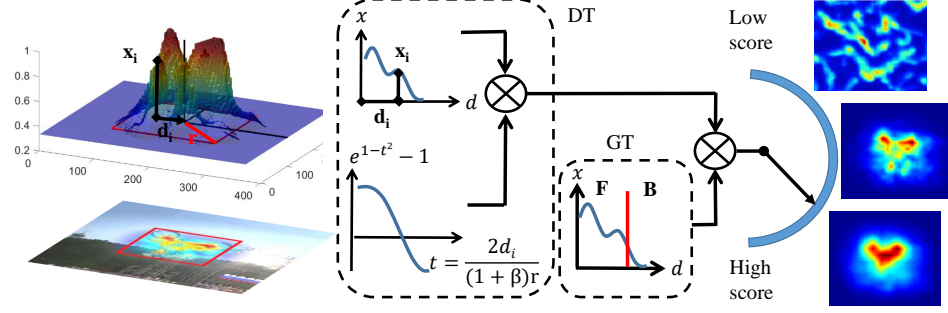


Fig. 5.3.: Diagram of DT and GT. DT aims to select the discriminative features learned by CNN. As the new figure shows, for the convolution results, the peak values with a large distribution is designed to result in a low score. DT provides a convergent non-linear weight distribution refer to the $2d_i/(1+\beta)r$ ratio as well as labels foreground/background by the positive/negative score. GT, on the other hand, amplify the discriminative score by comparing the power of foreground and background.

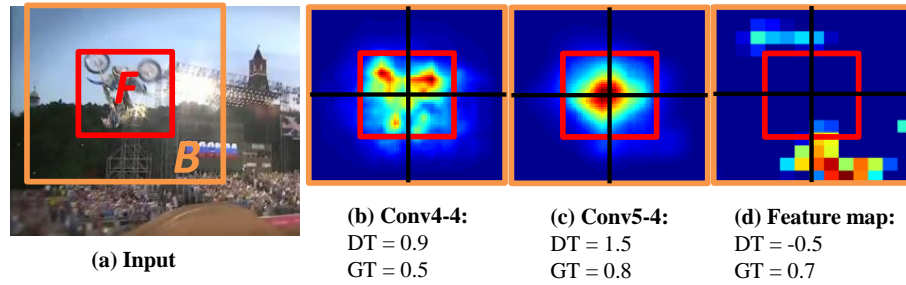


Fig. 5.4.: Recommender output. The input in (a) shows the foreground (F) and background (B) regions. In (b), layer Conv4-4 has multiple dispersed peak convolutional responses, while in (c), the peak responses gather to the center. Therefore, compare to (b), the kernel learned in (c) is more likely to be the discriminative detector of the target. A sample of feature maps is shown in (d), of which the discriminative term DT is a negative value because it only represents background features. The gain term GT, on the other hand, measures the difference level between F and B.

5.1.2 Target Appearance Modeling

Figure 5.5 shows an example of target appearance modeling. We extract target percept C^j from the j -th convolutional layer by taking the average of its Gaussian weighted feature maps h_i^j :

$$C^j = \sum_i G \circ h_i^j, \quad (5.4)$$

where the G is a cosine window that weight the feature by Hadamard product (notated as \circ). This is used to avoid the discontinuity of image bounder. The h_i^j is the i -th feature map of j -th convolutional layer. Due to max-pooling, the image size of C^j varies. Therefore, each C^j is re-sampled with a fixed size. To be noted that a normalized C (values in the range $[0,1]$) is used for further computing.

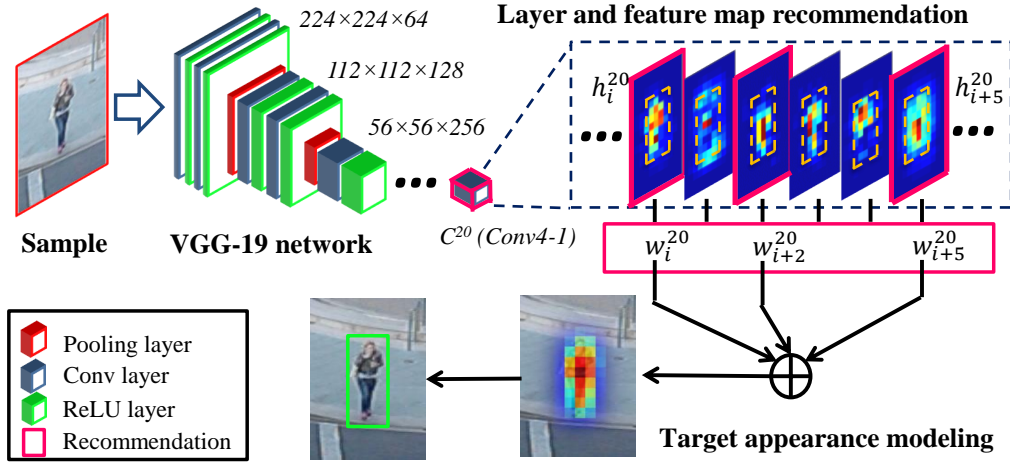


Fig. 5.5.: Recommendation framework: The proposed recommender selects layer C^{20} weights all feature maps in layer C^{20} . Only the feature maps with the positive weights are used to rebuild the target model. To be noted that the ReLU layers are taken into account, therefore there are 37 layers in VGG-19. Hence, we extract the sub-CNN with only 20 layers for this tracking task, which optimized the convolutional feature extraction process.

In the first frame, we extracted C from all layers and compute their recommendation scores:

$$\mathbf{f}^c = \{f_j^c | f_j^c = f(C^j, F, B), \forall j\}, \quad (5.5)$$

by equation (5.1). The index set ϕ of recommended layers is given by:

$$\phi = \{j | f_j^c \in TopN(\mathbf{f}^c)\}, \quad (5.6)$$

where function $TopN()$ returns the set of top N highest values of an input set.

Once ϕ is determined, we build the target model using recommended feature maps for each recommended layer. The recommendation scores of all feature maps in the j -th layer are computed as weights $\mathbf{w}^j = \{w_i^j | w_i^j = \max(0, f(h_i^j, F, B), \forall i)\}$ (a recommended feature map h satisfy $f(h, F, B) > 0$). Then our reconstructed model \mathbf{x} is define as:

$$\mathbf{x} = \{x^j | x^j = \sum_i w_i^j G \circ h_i^j, \forall j \in \phi\}. \quad (5.7)$$

Remark 2: In the CNN-based state-of-the-art trackers, such as our closest competitor HCF, the target percept is the weighted sum of the percepts obtained from pre-selected CNN layers by taking the sum of all feature maps in each layer. Since the input image patch could contain the background scene, the background features may also be updated to the target model. In our method, most of the background features are rejected by the proposed recommender and the features that represent the whole or critical parts of target dominate the result by giving higher weights. One example in the benchmark test is given in Fig. 5.6, which illustrates the importance of feature map recommendation by comparing the proposed method (without scale adaptation function) with HCF.

5.1.3 Correlation Filters

Correlation filters are trained by the linear regression method. Denote the vectorized samples of target x^n and a vectorized 2D Gaussian window $y \sim \mathcal{N}(\mu, \sigma^2)$, where μ is the center of target sample and σ is the kernel width. To be noted that

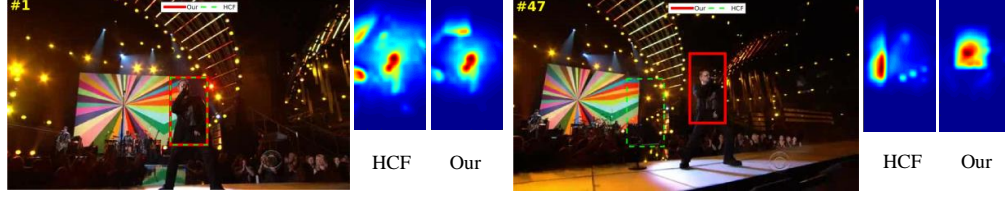


Fig. 5.6.: With (our: red bounding box) or without (HCF: green bounding box) feature selection by the proposed recommender. In order to do a fair comparison, we disabled the scale adaptation of our method. In this example, the background scene contains more features than the tracked object. As the CNNs simply detect any learned features, without the recommendation of the feature maps, the background features may dominate the target appearance model, resulting in the drifting of tracking.

y can also be the data labels because the weights in window y indicate the distances to the target center. Then the linear regression problem is formed by:

$$\arg \min_{\omega} \|\mathbf{x}\omega - y\|^2 + \lambda \|\omega\|^2, \quad (5.8)$$

where ω is the coefficients to be trained, $\mathbf{x} = \{x^1, x^2, \dots\}$, I is an identity matrix and λ is the regularization coefficient. The closed-form solution of 5.8 is given by:

$$\omega = (\mathbf{x}^T \mathbf{x} + \lambda I)^{-1} \mathbf{x}^T y. \quad (5.9)$$

To speed up the process, we train ω in Fourier domain. In the first frame, the target model \mathbf{X} is built by taking Fourier transforms of reconstructed feature $\mathbf{X} = \mathcal{F}(\mathbf{x})$ and $\mathbf{Y} = \mathcal{F}(\mathbf{y})$. The initial trained correlation filter \mathbf{W} is define as:

$$\mathbf{W} = \{W^j | W^j = \frac{Y \circ \bar{X}^j}{X^j \circ \bar{X}^j + \lambda}, \forall j \in \phi\}, \quad (5.10)$$

where the \bar{X} denote the complex conjugate of X .

From the second frame, our correlation filter estimates target location by computing the weighted correlation response R :

$$R = \mathcal{F}^{-1}(\sum_j f_j' W^j \circ X^j), \forall j \in \phi, \quad (5.11)$$

where f'_j is the normalized score over recommendation score \mathbf{f}^c from equation (5.5) with range $[0,1]$. Therefore, the local location u^* with maximum correlation response is the estimated new location of the target:

$$u^* = \arg \max_u R(u). \quad (5.12)$$

After the target is relocated, we extract the CNN feature again in order to learn the target appearance online. Let t be the index of frame sequence and α be the learning rate, the updating process of numerator \mathbf{A} and denominator \mathbf{B} of filter W can be written as:

$$\mathbf{A}_t = (1 - \alpha)\mathbf{A}_{t-1} + \alpha Y \circ \bar{\mathbf{X}}_t, \quad (5.13)$$

$$\mathbf{B}_t = (1 - \alpha)\mathbf{B}_{t-1} + \alpha \mathbf{X}_t \circ \bar{\mathbf{X}}_t, \quad (5.14)$$

$$\mathbf{W}_t = \frac{\mathbf{A}_t}{\mathbf{B}_t + \lambda}. \quad (5.15)$$

5.1.4 Min-channel Scale Learning

In visual tracking, the target size changes continuously if there is no occlusion or out of view. Therefore, for scale estimation, temporal information should be able to increase the accuracy of scale estimation. In this work, we propose a spatiotemporal-based min-channel scale learning scheme. The min-channel is a binary mask that crops the minimum target region (bounding box region) from the searching region.

The min-channel approach aims to reject the noise as much as possible. To obtain the min-channel map of the target, we project the max values at each location u throughout all recommended feature maps, i.e., find the maximum value on the third dimension of feature map set \mathbf{x} . Then crop it by the min-channel mask MC .

$$x^{min}(u) = MC(u) \cdot \max(\mathbf{x}(u)), \forall u. \quad (5.16)$$

Then the new scale s at frame t refer to the first frame is updated as:

$$s_t = \gamma \frac{\bar{s} \sum_u x_t^{min}(u)}{\sum_u x_1^{min}(u)} + (1 - \gamma) \frac{\sigma_t^w}{\sigma_1^w}, \quad (5.17)$$

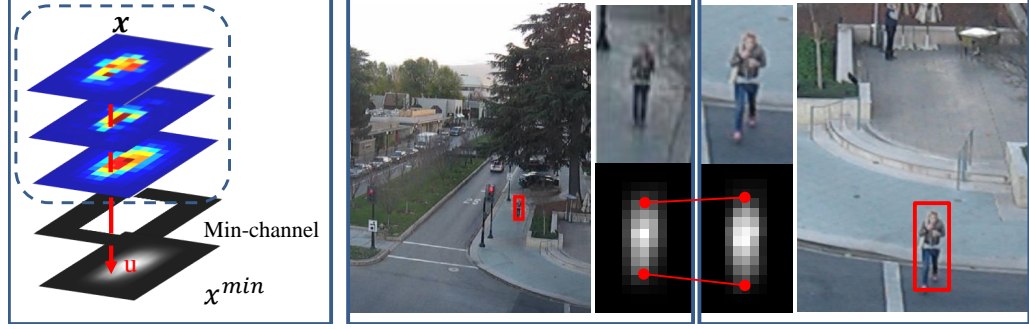


Fig. 5.7.: Spatiotemporal-based min-channel scale: the first block on the left describes the main idea of min-channel. The highest responses of all recommended feature maps are projecting into the min-channel map, in which we only take the pessimistic of target region into consideration. Since the highest response of feature maps are the location of target features, the variation of power distribution of the min-channel maps are used for target scale change estimation.

where σ^w is a weighted standard deviation of x^{min} . The average scale of a short term memory \bar{s} is computed after applying median filter to $\{s_{t-1}, s_{t-2}, \dots, s_{t-M-1}\}$, where M is the memory size. A typical value for γ is 0.9.

Remark 3: Because the feature maps extracted from CNN are re-sampled with a fixed size, \bar{s} is used to recover the true statistical characteristics of x_t^{min} . For the same reason, term $\frac{\sigma_t^w}{\sigma_1^w}$ does not affect the result when the previous scale estimation is correct. Wrong updating of target percepts tend to expand the distribution of significant response of x^{min} since they are mainly background features. Therefore, the purpose of introducing this term to extend the searching region when the uncertainty (the distribution of high convolutional response) of estimation increased.

5.1.5 Tracking Framework

We denote input t -th frame I_t , target region F_t and its extended searching region B_t . Function $CNNs()$ extracts the feature maps of input image. The tracking framework of proposed tracker is shown in algorithm 3.

Algorithm 3: Tracking framework of proposed tracker

Data: $\{I_t|t = 1, 2, \dots\}, F_1$
Result: $\{F_t|t = 2, 3, \dots\}$

- 1 initialization: $B_1, t \leftarrow 1, s_1 \leftarrow 1$;
- 2 $\mathbf{h} \leftarrow CNNs(I_t(B_t))$ // Extract CNN features
- 3 $\phi \leftarrow Eq. (5.6)(\mathbf{h})$ // Index set of recommended layers
- 4 $\mathbf{x}_t \leftarrow Eq. (5.7)(\mathbf{h}, \phi, F_t, B_t)$ // Target percept
- 5 $\mathbf{W}_t \leftarrow Eq. (5.10)(\mathbf{x}_t)$ // Correlation Filter
- 6 $x_t^{min} \leftarrow Eq. (5.16)(\mathbf{x}_t)$ // Min-channel map
- 7 **while** $t < frame\ length$ **do**
 - 8 $t \leftarrow t + 1$
 - 9 $\mathbf{h} \leftarrow CNNs(I_t(B_{t-1}))$
 - 10 $\mathbf{x} \leftarrow Eq. (5.7)(\mathbf{h}, \phi, F_{t-1}, B_{t-1})$
 - 11 $[F_t, B_t] \leftarrow Eq. (5.11) \& (5.12)(\mathbf{W}_{t-1}, \mathbf{x})$
 - 12 $x_t^{min} \leftarrow Eq. (5.16)(\mathbf{x})$
 - 13 $s_t \leftarrow Eq. (5.17)(x_t^{min}, x_1^{min}, \{s_{t-1}, \dots, s_{t-M-1}\})$
 - 14 $[F_t, B_t] \leftarrow ScaleUpdate(F_t, B_t, s_t)$
 - 15 $\mathbf{h} \leftarrow CNNs(I_t(B_t))$
 - 16 $\mathbf{x}_t \leftarrow Eq. (5.7)(\mathbf{h}, \phi, F_t, B_t)$
 - 17 $\mathbf{W}_t \leftarrow Eq. (5.15)(\mathbf{W}_{t-1}, \mathbf{x}_t)$

5.2 Experiments

In this section, we demonstrate the performance of the proposed tracker by Visual Tracker Benchmark v1.0 test. The benchmark protocol is proposed by Y. Wu, *et al* [97]. We tested our tracker on 50 datasets and evaluated with the following three evaluation methods:

Center location error (CLE). CLE measures the tracking accuracy by computing the distance between the centers of the ground truth and the estimated bounding box. On the other hand, SR measures the overlap between the ground truth and bounding box, which is related to scale adaptation.

Success rate (SR). Let the bounding box be r_t and ground truth r_a , the overlap score is defined as $score = \frac{r_t \cap r_a}{r_t \cup r_a}$. Typically, a $score > 0.5$ is considered as a successful tracking. The reported measurement SR is the percentage of successfully tracked frames in a sequence.

Target-based location error (TLE). TLE evaluation [98] evaluates the tracking accuracy by normalizing CLE with target size:

$$TLE = \min(\frac{2 * CLE}{\sqrt{x^2 + y^2}}, 2), \quad (5.18)$$

where (x, y) is the pixel length of target groundtruth. It is noted that CLE tells the pixel distance while ignoring the size of the target.

The proposed tracker is implemented in Matlab and runs on a laptop with 3.7-GHz Intel Core i7-8700K processor, 48GB RAM and NVIDIA Quadro P2000 GPU. We choose top 2 best layers and use the typical values for correlation filter parameters: $\alpha = 0.01$, $\lambda = 10^{-4}$.

Remark 4: We run our tracker as well as other 10 state-of-the-art trackers on each dataset from the first frame to the end, referred to a one-pass evaluation (OPE). Then the final tracking results for each dataset is obtained by taking the average of 10 times running. The final mean results are calculated from all frames over the 50 tested datasets.

Our test results are compared with other 10 high performance state-of-the-art trackers: HCF [18], SiamFC [19], CFNet [20], Struck [12], DCFNet [21], HDT [22], UDT [23], CSK [14], LSK [15], and MIL [11].

5.2.1 Overall Performance

The overall performance is shown in Fig. 5.8 using the area under curve (AUC) and merged plot of mean CLE and mean SR, respectively.

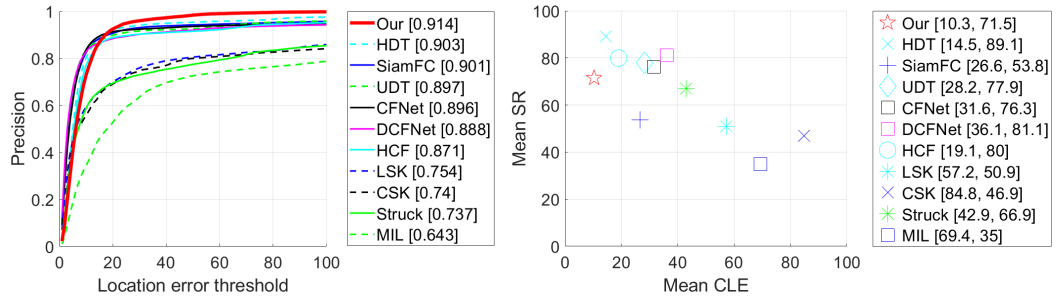


Fig. 5.8.: Location precision plot (CLE: pixel-based distance) of OPE using AUC and the merged plot of mean CLE and SR. This figure shows the overall performance of trackers in terms of tracking accuracy and robustness. The proposed tracker achieves the highest precision and the best average CLE (10.3).

The benchmark evaluation results illustrate that the proposed tracker performs competitively good tracking results in both tracking accuracy and scale adaptation compare to other 10 top-ranked state-of-the-art trackers. To provide detailed results, the tracking results of 4 challenging sequences in the image view is shown in Fig. 5.9 (only 6 selected trackers' results are displayed). To be noticed that accurate tracking may also generate large CLE when the target is big. Furthermore, the tracker may generate random CLE when it gets lost. To provide another aspect of the tracker performance, we applied target-based location error (TLE). The detailed results in TLE are also provided in table 5.1. The proposed method is again top 1 ranked by TLE(achieves 0.226, which the average tracking error is 22.6% of target size). The capability of adapting target large-scale variation and heavy occlusion of the proposed

	HCF	SiamFC	CFNet	Struck	DCFNet	HDT	UDT	CSK	LSK	MIL	Our
Car1	1.38	0.10	0.09	1.43	0.09	1.00	0.57	1.51	1.51	1.61	0.88
CarScale	0.31	0.15	0.15	0.35	0.09	0.31	0.11	0.67	0.16	0.35	0.28
Dudek	0.09	0.09	0.10	0.10	0.06	0.08	0.06	0.11	0.12	0.17	0.07
Human5	1.52	0.54	0.14	0.15	0.09	0.10	0.10	1.53	0.18	1.35	0.13
Human6	0.69	0.11	0.10	0.97	1.10	0.79	0.11	1.29	0.30	0.45	0.16
Jump	1.59	1.61	1.79	1.72	1.75	1.59	1.07	1.76	1.51	1.82	1.21
Shaking	0.23	0.10	0.11	0.59	0.14	0.26	0.13	0.32	0.50	0.47	0.17
Singer2	1.74	1.67	1.59	1.67	1.65	1.68	1.71	1.79	1.60	0.33	0.62
Skater2	0.14	0.14	0.11	0.20	0.22	0.14	0.21	0.22	0.24	0.47	0.14
Skiing	0.38	1.76	1.75	1.89	1.76	0.44	1.76	1.82	1.91	1.85	0.33
Tiger2	0.37	0.30	0.41	0.41	0.40	0.34	0.34	1.14	0.71	0.51	0.39
Trans	0.26	0.25	0.16	0.28	0.23	0.26	0.24	0.25	0.56	0.57	0.22
Vase	0.23	0.17	0.14	0.31	0.45	0.19	0.16	0.17	0.23	0.24	0.15
BlurCar2	0.07	0.08	0.04	0.11	0.04	0.06	0.04	0.07	0.14	1.24	0.13
BlurCar3	0.08	0.09	0.04	0.12	0.06	0.12	0.05	0.80	0.21	1.35	0.09
BlurFace	0.08	0.07	0.09	0.54	0.06	0.08	0.06	0.12	1.61	0.89	0.12
Bolt	0.21	1.90	1.94	1.94	0.20	0.15	0.16	1.92	0.21	1.95	0.18
Boy	0.11	0.10	0.08	0.13	0.07	0.09	0.08	0.40	0.08	0.47	0.13
Car2	0.12	0.06	0.06	0.07	0.04	0.11	0.05	0.08	1.10	1.28	0.19
Car24	0.26	0.06	0.07	1.62	0.05	0.22	0.06	0.31	0.06	1.03	0.31
Crossing	0.13	0.09	1.45	0.13	0.07	0.14	0.08	0.41	1.59	0.15	0.12
Crowds	0.10	0.12	0.12	0.26	1.76	0.11	0.12	0.13	1.95	1.83	0.15
Dancer	0.10	0.14	0.11	0.12	0.12	0.11	0.11	0.10	0.18	0.13	0.10
Dancer2	0.09	0.18	0.09	0.12	0.07	0.09	0.06	0.09	0.17	0.13	0.09
David	0.23	0.19	0.15	1.21	0.10	0.15	0.14	0.49	0.35	0.46	0.17
David3	0.05	0.59	0.07	1.09	0.07	0.05	0.07	0.51	0.90	0.40	0.14
Deer	0.10	0.11	0.09	0.10	0.21	0.10	0.10	0.10	1.24	1.59	0.16
Dog	0.28	0.28	0.19	0.51	0.25	0.36	0.24	0.33	1.65	0.41	0.45
Dog1	0.11	0.07	0.13	0.11	0.09	0.10	0.07	0.09	0.14	0.16	0.11
FaceOcc2	0.12	0.19	0.12	0.10	0.14	0.14	0.16	0.10	0.25	0.23	0.18
Fish	0.08	0.11	0.07	0.06	0.09	0.07	0.10	0.75	0.91	0.44	0.12
Football1	0.10	0.18	0.18	0.21	0.87	0.15	0.29	0.62	1.09	0.22	0.44
Freeman1	0.27	0.24	0.19	0.49	1.10	0.29	0.27	1.06	1.28	0.40	0.25
Girl	0.15	0.13	0.20	0.10	0.10	0.14	0.16	0.72	1.01	0.52	0.35
Human7	0.08	0.11	0.07	0.12	0.55	0.08	0.17	0.47	0.87	0.10	0.08
Human8	0.14	1.63	1.64	1.45	1.20	0.16	1.20	1.29	0.08	1.54	0.23
Human9	0.10	0.10	0.14	1.12	1.13	0.22	1.52	1.26	0.85	0.77	0.13
Jogging.1	0.07	0.13	0.12	1.00	0.13	0.07	1.09	1.48	1.37	1.43	0.08
Jogging.2	0.05	0.11	0.04	1.23	0.06	0.05	1.81	1.56	0.65	1.57	0.13
Man	0.10	0.06	0.07	0.06	0.07	0.09	0.07	0.08	0.09	1.55	0.14
Mhyang	0.05	0.06	0.10	0.05	0.06	0.08	0.06	0.07	0.07	0.40	0.11
MotorRolling	0.14	1.59	0.80	1.49	1.72	0.21	1.70	1.71	1.58	1.54	0.16
MountainBike	0.16	0.13	0.13	0.17	0.20	0.17	0.17	0.14	0.24	0.73	0.17
RedTeam	0.25	0.11	0.11	0.20	0.13	0.26	0.11	0.17	0.15	0.34	0.30
Singer1	0.13	0.07	0.08	0.24	0.03	0.16	0.07	0.23	0.33	0.26	0.27
Skater	0.13	0.12	0.09	0.11	0.25	0.15	0.18	0.19	0.26	0.14	0.15
Surfer	0.22	0.22	0.16	0.36	0.84	0.21	0.14	1.97	0.22	0.67	0.21
Trellis	0.14	0.42	0.11	0.16	0.09	0.12	0.10	0.42	0.10	1.19	0.11
Walking	0.15	0.12	0.09	0.18	0.07	0.23	0.06	0.27	0.74	0.11	0.10
Woman	0.13	0.20	1.55	0.08	0.15	0.16	0.13	1.51	1.52	1.52	0.22
Average TLE	0.276	0.343	0.349	0.545	0.405	0.249	0.353	0.692	0.695	0.786	0.227
Average FPS	5.68	68.26	45	17.04	42.365	5.736	51	435.81	NaN	35.42	13.25

Table 5.1.: TLE results of each dataset and average FPS: top 3 ranked results on each dataset are marked by red, green, and blue, respectively.



Fig. 5.9.: Tracking results on datasets (From top to bottom): Dog1, Human5, MotorRolling, Human6.

tracker is illustrated in Fig. 5.9. The second best CLE is given by tracker HDT. In our experiments, we simply use the top 2 recommended layers for target percept reconstruction. Although high layer carries very poor location information, our optimized target searching region and percept reconstruction significantly improved the tracking precision and tiny discontinuity of position shifting is observed in some datasets.

5.2.2 Results Analysis

From the results, it is clear that the proposed method outperforms other 10 state-of-the-art trackers by average tracking accuracy while success rate (SR) is the second echelon ranked by overall evaluation. In Fig. 5.8 we can find that the proposed method is not top ranked when the threshold is less than 20. This indicates a drawback of the proposed tracker: the poor feature location information when only high-level CNN layers are used. The low layer features have more precise location information but

less discriminate to background features. Unlike HCF which uses fixed hierarchical features to guarantee the location information, the proposed algorithm sometimes may only use high layers of CNN, which is a tradeoff between tracking accuracy and robustness. In conclusion, the proposed method achieves competitive overall performance against other 10 top-ranked state-of-the-art trackers.

5.2.3 Algorithm Complexity

The major time consumption comes from the CNN feature extraction because the complexity of the correlation filter and the proposed recommender are $\Theta(n^2)$ and $\Theta(n)$, respectively, while the CNN goes to $\Theta(\sum_{l=1}^D M_l^2 K_l^2 C_{l-1} C_l)$, where the M is the length of the feature map, K is the length of the Kernel, C is the number of channels in each layer, l is the current layer and D is the number of the layers. The proposed recommender compute the highest D that needed for a target in the first frame, which means we could have a smaller D for a simple target, resulting in a speedup of CNN feature extraction. The frame per second (FPS) of each open-source methods are given in Tab. 5.1.

5.2.4 Evaluation By Attributes

To analyze trackers' performance under different circumstances, we further evaluate the trackers on benchmark datasets regarding their attributes. The benchmark datasets are classified into 11 attributes: low resolution (LR), illumination variation (IV), scale variation (SV), occlusion (OCC), deformation (DEF), motion blur (MB), fast motion (FM), in-plane rotation (IPR), out-of-plane rotation (OPR), out-of-view (OV), and background clutters (BC). The average CLE of each attribute over all tested frames is shown in Table.5.2. The proposed method achieves the best AUC area on 9 attributes and best average CLE on 10 attributes.

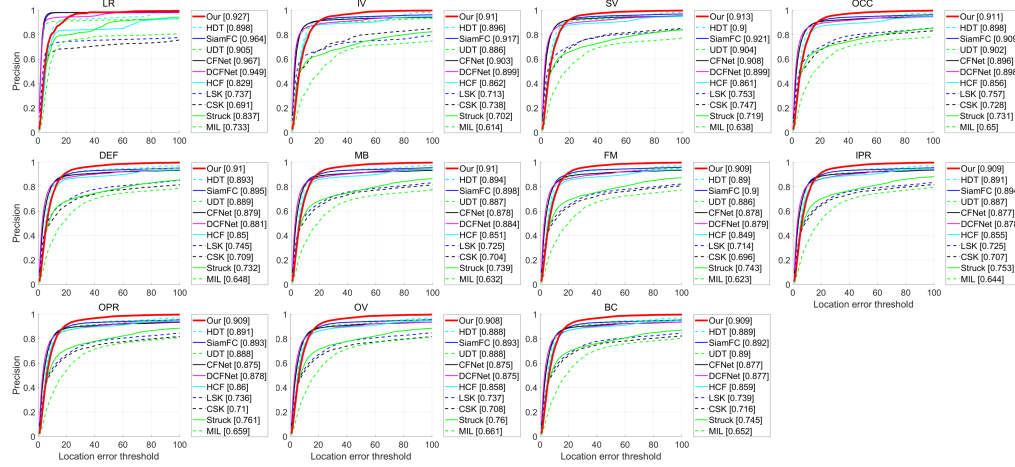


Fig. 5.10.: Precision plot of each attribute using AUC: the proposed tracker ranked No.1 on 9 attributes (IV, OCC, DEF, MB, FM, IPR, OPR, OV, and BC) out of 11.

	HCF	SiamFC	CFNet	Struck	DCFNet	HDT	UDT	CSK	LSK	MIL	Our
LR	19.425	7.938	7.794	22.768	9.463	10.694	17.909	176.382	42.53	41.128	7.825
IV	17.232	13.044	15.913	43.685	20.981	12.909	19.649	105.627	49.909	63.28	9.433
SV	19.473	12.385	14.068	40.079	19.334	12.396	15.351	90.759	39.47	58.293	9.125
OCC	21.6	14.008	17.343	39.657	19.501	12.966	14.975	89.185	37.254	58.383	9.333
DEF	23.76	16.576	21.18	40.713	23.366	14.219	16.809	88.889	39.817	61.572	9.448
MB	22.836	15.834	20.89	38.949	22.268	13.823	17.155	94.537	43.165	62.582	9.476
FM	22.406	15.123	20.333	37.469	22.946	14.379	17.219	98.938	44.809	62.203	9.569
IPR	21.139	15.765	21.042	36.75	22.573	14.203	17.119	91.652	42.858	58.519	9.603
OPR	20.117	15.853	21.743	36.168	22.576	14.423	16.984	86.262	40.621	55.467	9.596
OV	20.338	15.729	21.513	36.234	23.49	14.951	16.81	86.069	40.22	54.783	9.647
BC	20.056	15.871	21.724	39.204	23.423	14.822	16.762	86.523	40.912	57.332	9.605

Table 5.2.: Mean CLE by attributes: top 3 ranked results on each attribute are marked by red, green, and blue, respectively.

5.3 UAV onboard test

We first performed the proposed tracker on video data recorded by a rescue drone. The drone was flying inside the building or cave. The tracking targets include survivors and their belonging such as a bag. The video data contains blurred, fast motion, and discontinuous frames due to UAV platform. In Fig.5.11, we illustrate the tracking performance under blur and abrupt motion, illuminance change and large scale variation. Demo video is available on Github page.

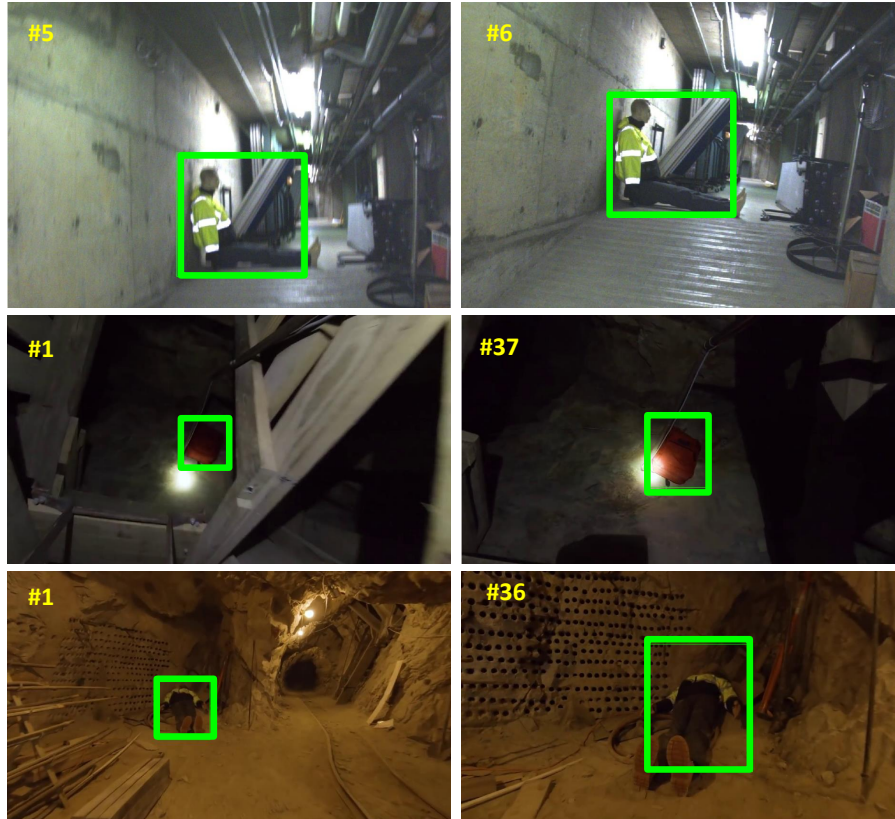


Fig. 5.11.: During the flight, image blur and abrupt motion happens occasionally. The illuminance and target scale also variate over time.

For onboard tracking, we implemented the proposed method on Nvidia Jetson TX2 using backbone CNN of a pre-trained Yolo v3 (Fig.5.12). For untrained target, we can manually select the target and the proposed tracker will learn its feature online. The UAV was flying under VICON environment. The task of the UAV is to perform the traffic light inspection in full autonomous. The Yolo detection runs on average 4 FPS. The FPS rise up to average 15 when switch to tracking model. Thus, the UAV get extra computational power to plan the path and process data analysis.



Fig. 5.12.: UAV onboard test.

6. CONCLUSION AND FUTURE WORK

6.1 Conclusion

In this work, we propose a smart vision system for UAV task-oriented flying. The system consists of visual odometry and object tracking. In visual odometry research, we demonstrated with several experiments that measuring error directly in the pose space eventually leads to better pose estimation. We have further shown that the long-term localization can be improved significantly (because the errors are accumulated over time) by detecting and avoiding points that require large adjustments. Intuitively, the requirement of larger adjustments has a larger influence during the pose estimation step. We also showed via experiments that a subset of such points cannot be detected using the reprojection-based measure, thus leading to poor long-term localization. We proposed a sum-of-square-based error measure that allows us to perform the camera minimum adjustment. With the new error measure tool, we presented a stereo orientation prior visual odometry algorithm to remove the points with large error bias in orientation, because the orientation error has a greater impact on the overall estimation accuracy. The proposed method reformulates the reprojection process and separates the orientation and translation error into different dimensions and rejects the keypoints that may mislead the estimator to fall into the local minimum with a large orientation error. The algorithm was implemented on both intel NUC and UP Board onboard computers for UAV indoor navigation. The experiments on the benchmark test and onboard test demonstrated that the performance of the proposed method is competitive with top-ranked stereo or visual-inertial odometry methods in terms of accuracy and robustness. For object tracking research, we proposed a novel CNN-based tracker that simplifies the network and learns a quality appearance model with scale estimation using a recommender for the untrained target.

Experimental results on 50 challenging benchmark datasets and UAV onboard tests demonstrated that the proposed method achieves competitive performance against other top-ranked state-of-the-art trackers in terms of tracking accuracy, scale adaptation, and tracking robustness. Thus, the entire vision system can run real-time on the UAV onboard system and perform the localization of both the UAV itself and the destination, which is critical information to a task-oriented flight.

6.1.1 System Robustness in Harsh Environments

In harsh environments, vision systems face two major challenges: complicated light sources and featureless texture. As the proposed visual odometry algorithms use traditional feature point detection and tracking methods which are sensitive to the environmental lighting and texture conditions. ORB feature detector and KLT point tracker, for example, detect the handcrafted patterns and match the similar descriptors between consecutive frames assuming the illuminance is constant. Due of complex light sources and camera auto exposure, this assumption is not always valid and it causes the point tracking failure as reported in [84]. A traditional way to solve the illuminance inconsistency problem is performing the histogram equalization over the associated frames. However, the improvement is limited because most of image enhancement tasks are typical ill-posed problem.

Recently, the research in deep learning-based image enhancement has gained popularity. Since the DNN models have been trained on massive data samples, they carry a lot of prior knowledge for solving the ill-posed problem. As a result, lightweight DNNs for image enhancement are in increasing demand in the future to assure the robotic system’s robustness in harsh environments.

6.2 Future Works

6.2.1 Integration of onboard vision system

The CNN shown its outstanding ability to extract image features throughout our visual tracking research. In the future, we will modify the system architecture as shown in the Fig. 6.1 to further improve the onboard vision system. In the revised architecture, the visual odometry, object detection and tracking are tightly coupled. The input image will first be processed by a lightweight CNN for different semantic feature extraction. The local features of image patches which can be utilized for point tracking are mainly low-level CNN features. High semantic level features that represent the entire image can be used for loop-closure detection. The future visual tracking algorithm will be based on the Siamese net.

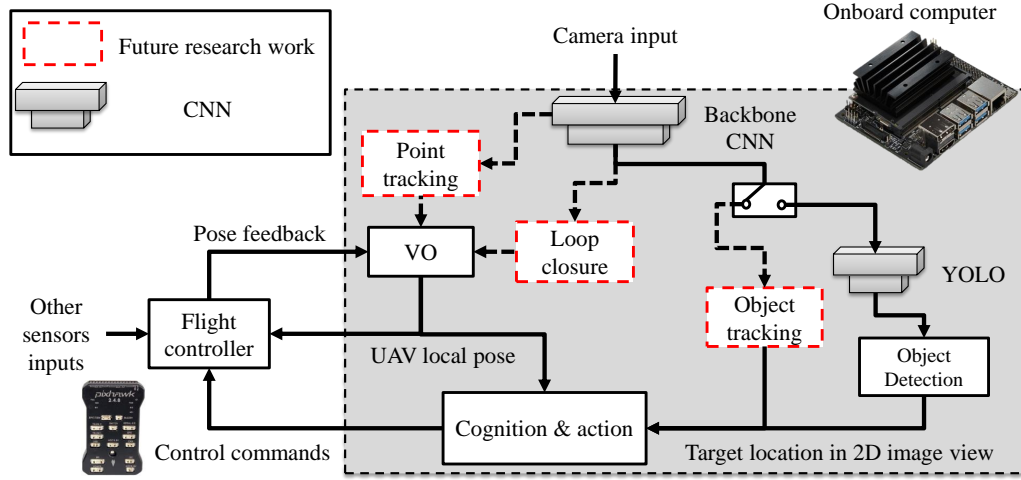


Fig. 6.1.: The comprehensive visual navigation system for task-oriented flight.

6.2.2 Loop-closure detection

To design a comprehensive and versatile visual navigation system, we need to close the loop for visual odometry. In our future work, we will implement the loop-closure detection as Fig. 6.1 shows. It takes the image perception from the same backbone

CNN with object detection/tracking module, which saves the computational power for the UAV onboard system.

The loop-closure detection helps the UAV to correct the estimation error by registering the current location with the learned one when the UAV came back to the visited places. The loop closure approaches are usually done by single image retrieval, but the challenge is, there are many similar locations, such as the corridor in the building. The errors in matches cannot be avoided in realistic. To reduce the probability of the mismatch, we will investigate the performance of the sequence matching approach. The idea is illustrated in Fig. 6.2. We close the loop by taking the order of the sequence of keyframes into consideration, which is similar to the recurrent neural network (RNN) in natural language understanding. The image will be encoded by a backbone CNN and represented by feature maps. This approach can also be used for robot co-mapping.

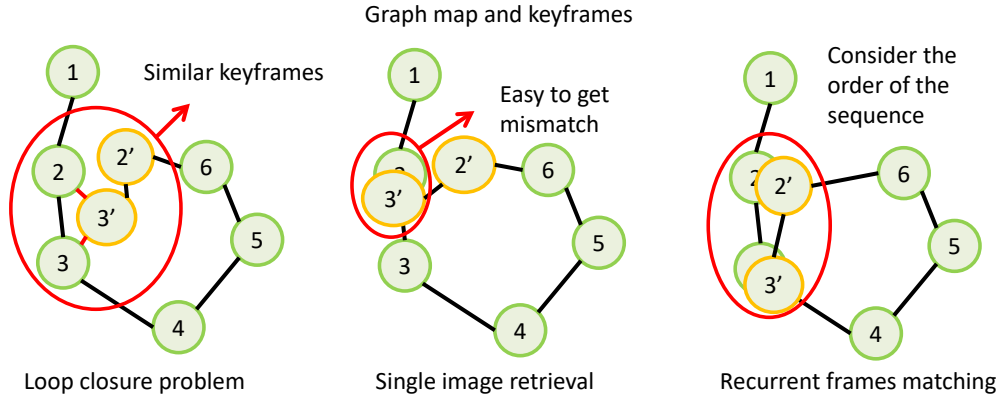


Fig. 6.2.: Recurrent image retrieval for loop-closure detection.

6.2.3 End-to-end 3D pose estimation

The awareness of surroundings, including the estimation of the object's position and orientation from the visual sensor's perception, remains a big challenge for an autonomous robot. Because such estimation is usually built upon the object detec-

tion on depth-registered vision. With the geometry correspondences, the pose of rigid objects could be computed by finding their 6-DoF transforms. But the object pose estimation is thus relatively a complex task and applicable only when an object's texture and 3D shape are well learned by the robot, the object detection, and 3D points registration should be employed as preliminary processes. In future work, we will investigate an end-to-end lightweight CNN pose estimation approach for autonomous drone navigation, based on the 3D perception from proposed visual odometry and object tracking. (Fig. 6.3).

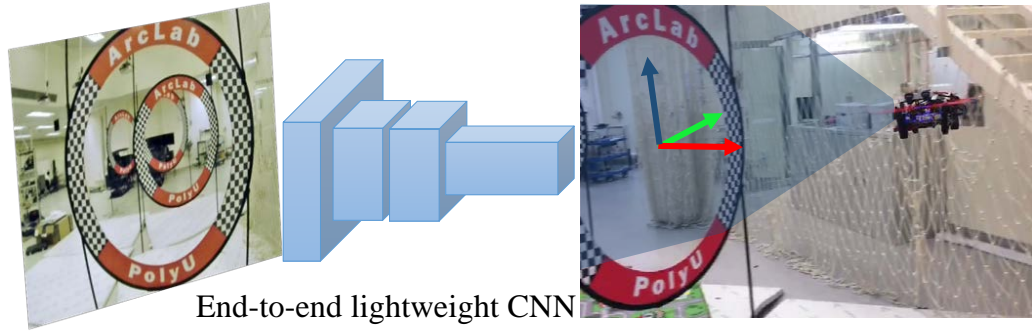


Fig. 6.3.: End-to-end lightweight CNN for pose estimation.

REFERENCES

REFERENCES

- [1] M. Westoby, J. Brasington, N. Glasser, M. Hambrey, and J. Reynolds, “‘structure-from-motion’ photogrammetry: A low-cost, effective tool for geoscience applications,” *Geomorphology*, vol. 179, pp. 300–314, 2012.
- [2] E. Rosten, R. Porter, and T. Drummond, “Faster and Better: A Machine Learning Approach to Corner Detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 1, pp. 105–119, Jan 2010.
- [3] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “ORB: An efficient alternative to SIFT or SURF,” in *IEEE/CVF International Conference on Computer Vision (ICCV)*, Nov 2011, pp. 2564–2571.
- [4] D. P. Paudel, A. Habed, C. Demonceaux, and P. Vasseur, “Robust and Optimal Sum-of-Squares-Based Point-to-Plane Registration of Image Sets and Structured Scenes,” in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015, pp. 2048–2056.
- [5] P. Speciale, D. P. Paudel, M. R. Oswald, T. Kroeger, L. V. Gool, and M. Pollefeys, “Consensus Maximization with Linear Matrix Inequality Constraints,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5048–5056.
- [6] C. Fu, A. Carrio, M. A. Olivares-Mendez, R. Suarez-Fernandez, and P. Campoy, “Robust real-time vision-based aircraft tracking from Unmanned Aerial Vehicles,” in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 5441–5446.
- [7] M. Mueller, G. Sharma, N. Smith, and B. Ghanem, “Persistent Aerial Tracking system for UAVs,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 1562–1569.
- [8] C. Fu, R. Duan, D. Kircali, and E. Kayacan, “Onboard Robust Visual Tracking for UAVs Using a Reliable Global-Local Object Model,” *Sensors*, vol. 16, no. 9, 2016.
- [9] H. Chen and P. Lu, “Computationally efficient obstacle avoidance trajectory planner for uavs based on heuristic angular search method,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2020, pp. 5693–5699.
- [10] R. Duan, C. Fu, and E. Kayacan, “Recoverable recommended keypoint-aware visual tracking using coupled-layer appearance modelling,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 4085–4091.

- [11] B. Babenko, M.-H. Yang, and S. Belongie, “Robust Object Tracking with Online Multiple Instance Learning,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, no. 8, pp. 1619–1632, 2011.
- [12] S. Hare, A. Saffari, and P. H. S. Torr, “Struck: Structured output tracking with kernels,” in *2011 International Conference on Computer Vision*, Nov 2011, pp. 263–270.
- [13] C. Fu, R. Duan, and E. Kayacan, “Visual tracking with online structural similarity-based weighted multiple instance learning,” *Information Sciences*, vol. 481, pp. 292 – 310, 2019.
- [14] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, “Exploiting the Circulant Structure of Tracking-by-detection with Kernels,” in *Proceedings of the 12th European Conference on Computer Vision - Volume Part IV*, ser. ECCV’12. Springer-Verlag, 2012, pp. 702–715.
- [15] B. Liu, J. Huang, L. Yang, and C. Kulikowsk, “Robust tracking using local sparse appearance model and K-selection,” in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, June 2011, pp. 1313–1320.
- [16] R. Duan, C. Fu, E. Kayacan, and D. P. Paudel, “Recommended keypoint-aware tracker: Adaptive real-time visual tracking using consensus feature prior ranking,” in *2016 IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 449–453.
- [17] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, “High-speed tracking with kernelized correlation filters,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, 2015.
- [18] C. Ma, J. B. Huang, X. Yang, and M. H. Yang, “Hierarchical convolutional features for visual tracking,” in *2015 IEEE International Conference on Computer Vision (ICCV)*, Dec 2015, pp. 3074–3082.
- [19] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, “Fully-convolutional siamese networks for object tracking,” *arXiv preprint arXiv:1606.09549*, 2016.
- [20] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. S. Torr, “End-to-end representation learning for correlation filter based tracking,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [21] J. X. M. Z. W. H. Qiang Wang, Jin Gao, “Dcfnet: Discriminant correlation filters network for visual tracking,” *arXiv preprint arXiv:1704.04057*, 2017.
- [22] Y. Qi, S. Zhang, L. Qin, Q. Huang, H. Yao, J. Lim, and M. Yang, “Hedging deep features for visual tracking,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 5, pp. 1116–1130, May 2019.
- [23] N. Wang, Y. Song, C. Ma, W. Zhou, W. Liu, and H. Li, “Unsupervised deep tracking,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [24] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” *CoRR*, vol. abs/1804.02767, 2018.

- [25] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Computer Vision – ECCV 2016*. Cham: Springer International Publishing, 2016, pp. 21–37.
- [26] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, June 2017.
- [27] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017, pp. 2980–2988.
- [28] S. Hong, T. You, S. Kwak, and B. Han, "Online tracking by learning discriminative saliency map with convolutional neural network," in *Proceedings of the 32nd International Conference on Machine Learning, 2015, Lille, France, 6-11 July 2015*, 2015.
- [29] J. A. Grunert, "Das pothenotische problem in erweiterter gestalt nebst bber seine anwendungen in der geodasie," *Grunerts Archiv fur Mathematik und Physik*, pp. 238–248, 1841.
- [30] E. Kruppa, *Zur Ermittlung eines Objektes aus zwei Perspektiven mit innerer Orientierung*. Hölder, 1913.
- [31] E. Merritt, "Explicit three-point resection in space," *Photogrammetric Engineering*, vol. 15, no. 4, pp. 649–655, 1949.
- [32] I. E. Sutherland, "Three-dimensional data input by tablet," *Proceedings of the IEEE*, vol. 62, no. 4, pp. 453–461, 1974.
- [33] H. C. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections," *Nature*, vol. 293, no. 5828, pp. 133–135, 1981.
- [34] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.
- [35] R. M. Haralick, H. Joo, C.-N. Lee, X. Zhuang, V. G. Vaidya, and M. B. Kim, "Pose estimation from corresponding point data," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 19, no. 6, pp. 1426–1446, 1989.
- [36] O. Faugeras, *Three-dimensional computer vision: a geometric viewpoint*. MIT press, 1993.
- [37] Z. Zhang and G. Xu, "Epipolar geometry in stereo, motion and object recognition," 1996.
- [38] L. Quan and Z. Lan, "Linear n-point camera pose determination," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 21, no. 8, pp. 774–780, 1999.
- [39] X.-S. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng, "Complete solution classification for the perspective-three-point problem," *IEEE transactions on pattern analysis and machine intelligence*, vol. 25, no. 8, pp. 930–943, 2003.

- [40] D. Nistér, “An efficient solution to the five-point relative pose problem,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 6, pp. 756–770, 2004.
- [41] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.
- [42] R. I. Hartley and P. Sturm, “Triangulation,” *Computer vision and image understanding*, vol. 68, no. 2, pp. 146–157, 1997.
- [43] M. Pollefeys, L. Van Gool, and A. Oosterlinck, “The modulus constraint: a new constraint self-calibration,” in *Proceedings of 13th International Conference on Pattern Recognition*, vol. 1. IEEE, 1996, pp. 349–353.
- [44] S. Petitjean, “Algebraic geometry and computer vision: Polynomial systems, real and complex roots,” *Journal of Mathematical Imaging and Vision*, vol. 10, no. 3, pp. 191–220, 1999.
- [45] S. Maybank, “Applications of algebraic geometry to computer vision,” in *Computational Algebraic Geometry*. Springer, 1993, pp. 185–194.
- [46] J. D. Kileel, “Algebraic geometry for computer vision,” Ph.D. dissertation, UC Berkeley, 2017.
- [47] S. Agarwal, A. Pryhuber, and R. R. Thomas, “Ideals of the multiview variety,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.
- [48] H. Stewénius, *Gröbner basis methods for minimal problems in computer vision*. Citeseer, 2005.
- [49] Z. Kukelova, M. Bujnak, and T. Pajdla, “Polynomial eigenvalue solutions to minimal problems in computer vision,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1381–1393, 2011.
- [50] T. Duff, K. Kohn, A. Leykin, and T. Pajdla, “Plmp-point-line minimal problems in complete multi-view visibility,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 1675–1684.
- [51] L. Kneip, H. Li, and Y. Seo, “Upnp: An optimal $\mathcal{O}(n)$ solution to the absolute pose problem with universal applicability,” in *European Conference on Computer Vision*. Springer, 2014, pp. 127–142.
- [52] J. Briales, L. Kneip, and J. Gonzalez-Jimenez, “A certifiably globally optimal solution to the non-minimal relative pose problem,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 145–154.
- [53] T. Probst, D. P. Paudel, A. Chhatkuli, and L. V. Gool, “Convex relaxations for consensus and non-minimal problems in 3d vision,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 10 233–10 242.
- [54] R. Hartley and F. Kahl, “Optimal algorithms in multiview geometry,” in *Asian conference on computer vision*. Springer, 2007, pp. 13–34.
- [55] F. Kahl and D. Henrion, “Globally optimal estimates for geometric reconstruction problems,” *International Journal of Computer Vision*, vol. 74, no. 1, pp. 3–15, 2007.

- [56] R. I. Hartley, "In defense of the eight-point algorithm," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 19, no. 6, pp. 580–593, 1997.
- [57] D. Brown, "The bundle adjustment-progress and prospect," in *XIII Congress of the ISPRS, Helsinki, 1976*, 1976.
- [58] H. Wang, R. Jiang, H. Zhang, and S. S. Ge, "Heading Reference-Assisted Pose Estimation for Ground Vehicles," *IEEE Transactions on Automation Science and Engineering*, vol. 16, no. 1, pp. 448–458, 2019.
- [59] Y. Lu, Z. Xue, G.-S. Xia, and L. Zhang, "A Survey on Vision-based UAV Navigation," *Geo-spatial Information Science*, vol. 21, no. 1, pp. 21–32, 2018.
- [60] J. Zhang and S. Singh, "Visual-lidar Odometry and Mapping: Low-drift, Robust, and Fast," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 2174–2181.
- [61] K. Chiang, G. Tsai, Y. Li, and N. El-Sheimy, "Development of LiDAR-Based UAV System for Environment Reconstruction," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 10, pp. 1790–1794, Oct 2017.
- [62] X. Li, Y. Li, E. P. Örnek, J. Lin, and F. Tombari, "Co-Planar Parametrization for Stereo-SLAM and Visual-Inertial Odometry," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6972–6979, 2020.
- [63] 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*, vol. 3, no. 2, pp. 965–972, 2018.
- [64] T. Qin, P. Li, and S. Shen, "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, Aug 2018.
- [65] J. Pestana, I. Mellado-Bataller, J. L. Sanchez-Lopez, C. Fu, I. F. Mondragón, and P. Campoy, "A General Purpose Configurable Controller for Indoors and Outdoors GPS-Denied Navigation for Multirotor Unmanned Aerial Vehicles," *Journal of Intelligent & Robotic Systems*, vol. 73, no. 1, pp. 387–400, Jan 2014.
- [66] C. Fu, A. Carrio, and P. Campoy, "Efficient visual odometry and mapping for Unmanned Aerial Vehicle using ARM-based stereo vision pre-processing system," in *International Conference on Unmanned Aircraft Systems (ICUAS)*, June 2015, pp. 957–962.
- [67] D. Scaramuzza and F. Fraundorfer, "Visual Odometry [Tutorial]," *IEEE Robotics Automation Magazine*, vol. 18, no. 4, pp. 80–92, 2011.
- [68] C. Fu, A. Sarabakha, E. Kayacan, C. Wagner, R. John, and J. M. Garibaldi, "Input Uncertainty Sensitivity Enhanced Nonsingleton Fuzzy Logic Controllers for Long-Term Navigation of Quadrotor UAVs," *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 2, pp. 725–734, 2018.
- [69] Y. Li, C. Fu, F. Ding, Z. Huang, and G. Lu, "AutoTrack: Towards High-Performance Visual Tracking for UAV With Automatic Spatio-Temporal Regularization," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 11 920–11 929.

- [70] T. Mouats, N. Aouf, L. Chermak, and M. A. Richardson, "Thermal Stereo Odometry for UAVs," *IEEE Sensors Journal*, vol. 15, no. 11, pp. 6335–6347, 2015.
- [71] H. Liang, N. J. Sanket, C. Fermüller, and Y. Aloimonos, "SalientDSO: Bringing Attention to Direct Sparse Odometry," *IEEE Transactions on Automation Science and Engineering*, vol. 16, no. 4, pp. 1619–1626, 2019.
- [72] R. Wang, M. Schwörer, and D. Cremers, "Stereo DSO: Large-Scale Direct Sparse Visual Odometry with Stereo Cameras," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct 2017, pp. 3923–3931.
- [73] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast Semi-direct Monocular Visual Odometry," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 15–22.
- [74] T. Feng and D. Gu, "SGANVO: Unsupervised Deep Visual Odometry and Depth Estimation With Stacked Generative Adversarial Networks," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4431–4437, 2019.
- [75] R. Wang, S. M. Pizer, and J.-M. Frahm, "Recurrent Neural Network for Unsupervised Learning of Monocular Video Visual Odometry and Depth," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5555–5564.
- [76] T. Schops, T. Sattler, and M. Pollefeys, "BAD SLAM: Bundle Adjusted Direct RGB-D SLAM," in *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 134–144.
- [77] A. I. Mourikis and S. I. Roumeliotis, "A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation," in *Proceedings of IEEE International Conference on Robotics and Automation*, April 2007, pp. 3565–3572.
- [78] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust Visual Inertial Odometry using a Direct EKF-based Approach," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2015, pp. 298–304.
- [79] P. F. Alcantarilla, L. M. Bergasa, and F. Dellaert, "Visual Odometry Priors for Robust EKF-SLAM," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2010, pp. 3501–3506.
- [80] J. Delmerico and D. Scaramuzza, "A Benchmark Comparison of Monocular Visual-Inertial Odometry Algorithms for Flying Robots," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 2502–2509.
- [81] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The EuRoC Micro Aerial Vehicle Datasets," *International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, Sep. 2016.
- [82] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based Visual-inertial Odometry Using Nonlinear Optimization," *International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, Mar. 2015.
- [83] T. Lupton and S. Sukkarieh, "Visual-Inertial-Aided Navigation for High-Dynamic Motion in Built Environments Without Initial Conditions," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 61–76, Feb 2012.

- [84] K. Sun, K. Mohta, B. Pfrommer, M. Watterson, S. Liu, Y. Mulgaonkar, C. J. Taylor, and V. S. A. Kumar, “Robust Stereo Visual Inertial Odometry for Fast Autonomous Flight,” *IEEE Robotics and Automation Letters*, vol. 3, pp. 965–972, 2017.
- [85] H. Li, Y. Li, and F. Porikli, “DeepTrack: Learning Discriminative Feature Representations Online for Robust Visual Tracking,” *IEEE Transactions on Image Processing*, vol. 25, no. 4, pp. 1834–1848, 2016.
- [86] L. Wang, T. Liu, G. Wang, K. L. Chan, and Q. Yang, “Video Tracking Using Learned Hierarchical Features,” *IEEE Transactions on Image Processing*, vol. 24, no. 4, pp. 1424–1435, April 2015.
- [87] D. Pani Paudel, A. Habed, C. Demonceaux, and P. Vasseur, “Robust and optimal sum-of-squares-based point-to-plane registration of image sets and structured scenes,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2048–2056.
- [88] R. E. Moore, *Interval analysis*. Prentice-Hall Englewood Cliffs, 1966, vol. 4.
- [89] T. Schöps, J. L. Schönberger, S. Galliani, T. Sattler, K. Schindler, M. Pollefeys, and A. Geiger, “A multi-view stereo benchmark with high-resolution images and multi-camera videos,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [90] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, “The euroc micro aerial vehicle datasets,” *The International Journal of Robotics Research*, 2016.
- [91] J. Delmerico and D. Scaramuzza, “A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 2502–2509.
- [92] C. Tomasi and T. Kanade, “Detection and Tracking of Point Features,” *Technical Report*, vol. 91, no. 21, pp. 9795–9802, 1991.
- [93] S. Sheorey, S. Keshavamurthy, H. Yu, H. Nguyen, and C. N. Taylor, “Uncertainty Estimation for KLT Tracking,” in *Asian Conference on Computer Vision (ACCV) Workshops*. Cham: Springer International Publishing, 2015, pp. 475–487.
- [94] Z. Zhang and D. Scaramuzza, “A Tutorial on Quantitative Trajectory Evaluation for Visual(-Inertial) Odometry,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2018, pp. 7244–7251.
- [95] A. Geiger, P. Lenz, and R. Urtasun, “Are We Ready for Autonomous Driving? The KITTI Vision Benchmark Suite,” in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3354–3361.
- [96] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, “ImageNet Large Scale Visual Recognition Challenge,” *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

- [97] Y. Wu, J. Lim, and M. Yang, “Object Tracking Benchmark,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 37, no. 9, pp. 1834–1848, 2015.
- [98] R. Duan, C. Fu, and E. Kayacan, “Tracking-Recommendation-Detection: A Novel Online Target Modeling for Visual Tracking,” *Engineering Applications of Artificial Intelligence*, vol. 64, pp. 128 – 139, 2017.

VITA

VITA

Born Mar, 1990 in Suining, Sichuan, China

Education

- Ph. D. in Aeronautical and Aviation Engineering *2019-Present*
The Hong Kong Polytechnic University, Hong Kong SAR, China
- M. Sc. in Computer Vision (European VIBOT program) *2013-2015*
University of Burgundy, France
- B. Eng. in Communication Engineering *2009-2013*
Southwest University of Science and Technology, Mianyang, China

Research Experience

- Visiting Scholar *2017-2018*
The City College of New York, USA
- Research Associate *2015-2017*
Nanyang Technological University, Singapore
- Research Intern *2014-2015*
University of Strasbourg, France