Mini-project: A visual odometry pipelines!

GROUP MEMBERS

Yue Li Jiacheng Qiu Qi Ma

I. DESCRIPTION OF THE PROJECT

This project aims to build a monocular visual odometry pipeline, consisting of bootstrapping and continuous operations. We implement and then test our pineline on the three provided datasets: KITTI, Malaga and parking. Additionally, we adopt this pinepine on our AMZ racing dataset. Despite some challenges, we compare the results of two tracking method: direct method and feature-based method. Further discussion and idea about the results are provided in the end. Our pineline results are available at this link: Mini Project Pipeline.

II. PIPELINE IMPLEMENTATION

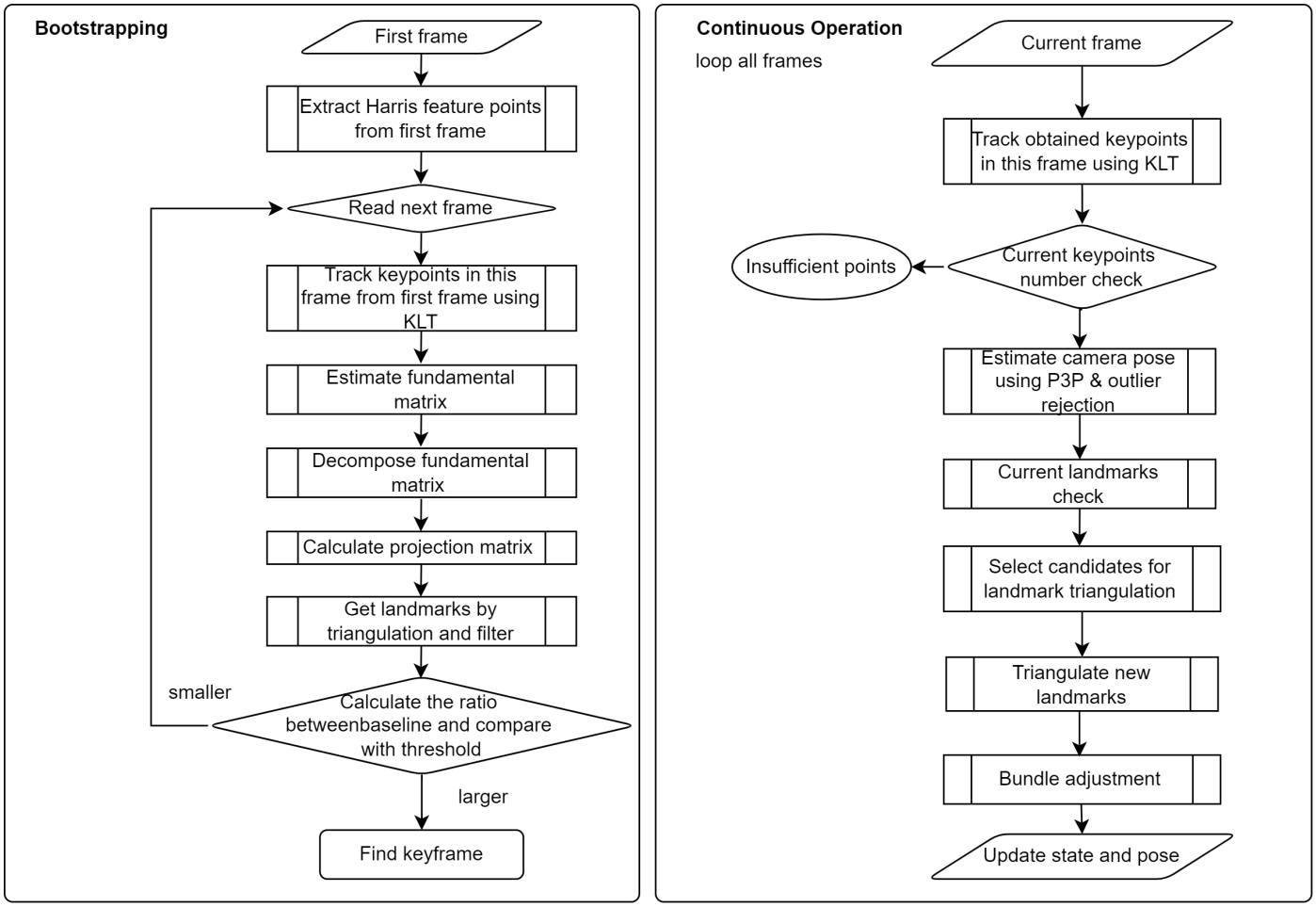


Fig. 1: Flow chart of bootstrapping and continuous operation

A. *Bootstrapping*

The purpose of the bootstrapping part is to find a appropriate keyframe so that we can utilise the initial frame and the keyframe found to generate 3D landmarks which are applied in the continuous operation section after bootstrapping. The whole procedure of the bootstrapping is illustrated in the left part of Figure 1.

1) Keypoints extraction: we first use Harris corner detector to extract the keypoints from the initial frame. In order to get high score and uniformly in frame distributed keypoints, we adopt an adaptive non-maximal suppression algorithm [1] to filter extracted keypoints. The details of this algorithm are demonstrated in the chapter Experiment and result.

2) KLT tracking: KLT is used to track the keypoints among the frames by comparing a pixel with other pixels in a certain large window to find the most similar pixel, which allows tracking trajectories in real time. The prerequisites for the good performance of KLT algorithm are brightness constancy, temporal consistency and spatial coherency. By using this algorithm, we can track the keypoints from the first frame in its following frames quickly.

3) Fundamental matrix estimation and decomposition: With the keypoints from two different frames, we can estimate the fundamental matrix. In the pipeline, we adopt least median of squares method, which at least 50% of the points should be inliers in both frames. After acquiring the fundamental matrix, we can decompose it to find the relative motion between two frames. Since two rotation matrices and two transition vectors are obtained during the decomposition, we have to find the right rotation and translation based on the inliers.

4) Linear triangulation: Since intrinsic matrix is known, the projection matrix for both two frames can be calculated with the estimated rotation matrix and transition vector. Therefore, the position of 3D landmarks can be calculated by linear square approximation and optimized by minimizing the reprojection errors.

5) Keyframe selection: After obtaining the landmarks and the baseline, we calculate the ratio between the average depth of the landmarks and the length of the baseline. We set a threshold and once the ratio surpasses the threshold, this frame is chosen as the keyframe and meanwhile the bootstrapping part is terminated. The value of the threshold depends on the scenario and needs to be tuned.

B. Continuous Operation

After bootstrapping, the pipeline processes new frames sequentially. This part in general take previous state, previous image and current image as input, and output the current state and camera pose. To achieve this, newly tracked keypoints is associated to existing landmarks, followed by pose estimation with 3D-2D correspondences, and meanwhile triangulating new landmarks. The process is shown in Figure 1.

1) KLT tracking: The tracking part is the same as in bootstrapping. The three provided datasets all meet the prerequisite of KLT well. With the help of coarse-to-fine estimation, sufficient keypoints are tracked along the trajectory.

2) Pose and triangulation refinement: P3P, requiring only 3 points and another one for disambiguation, allows outlier rejection with less iterations. Though it is suggested in the project statement that the afterwards DLT solution may be worse than the best P3P guess, we find it not true for our pipeline. The comparison between P3P with refinement and P3P only is shown in Figure 3, without refinement it often leads to losing track. Thus, we use P3P with refinement in all of our experiments later. Furthermore, we add refinement to triangulation to minimize the sum of reprojection errors.

3) Current landmarks check: In every step of continuous operation, existing keypoints are matched against new frame. Previous keypoints that are not tracked in the new frame are removed in the state vector. Besides, we have to check if the landmarks corresponding to the keypoints are still in front of the image plane. Another reprojection factor is introduced to remove landmarks that are too far away compared to the current average landmarks depth, since they include large depth uncertainty.

4) Candidates selection for triangulation: It is necessary to continuously add new landmarks since invalid ones are being removed every step. We implement this by taking notes of the track length of each keypoint candidate. As soon as the angle between corresponding keypoint vectors exceed the angle threshold, we triangulate this keypoint candidate and get a new landmark candidate. We calculate reprojection errors of obtained candidates, those with reprojection errors larger than the threshold is also removed.

5) Bundle adjustment: Bundle adjustment is crucial in this task since it solves the underlying non-linear least-squares problem [2]. In our pipeline we run global bundle adjustment at first when total frames is limited and switch to sliding window bundle adjustment in later tracking for computation saving purpose.

III. Experiment and result

A. Keypoints extraction

The performance of our visual odometry pipeline depends largely how we choose the feature points on the frames. If the feature points are poorly extracted, the matching between the frames can fail and the whole procedure collapses. A good feature points extraction should be: 1. The extracted points have high confidence; 2. The extracted points should well-distributed. Extracting high confidence points can guarantee a large possibility that the same feature points can also be found in other frames and scattered keypoints can avoid the redundant information. In order to realize aforementioned two purposes, we adopted an adaptive non-maximal suppression algorithm via square covering [1], which uses a grid with a adaptive size to eliminate the points around the extracted keypoints. The code of the algorithm comes from SSC.

In order to visualize the this ANMS algorithm and compare with other extraction method, we use Harris corner detector to extract 600 keypoints from the first frame in the KITTI dataset. Figure 2 shows three different methods for keypoints extraction. In Figure 2(a), we extract 600 keypoints with highest scores. Almost all points are clustered along the edges or around the corners. Even though we have 600 points, we can only make use of much less information from them. Figure 2(b) shows that 600 keypoints are well-scattered in the frame and only few clusters can be observed. However, this also sacrifices the quality of the selected keypoints. We can see some points locating in the sky which are not expected to be the right keypoints. The keypoints extracted by our adopted algorithm is shown in Figure 2(c). The keypoints are well-distributed in the frame and no cluster can be observed. Meanwhile, there are no points extracted from the sky which is also reasonable.

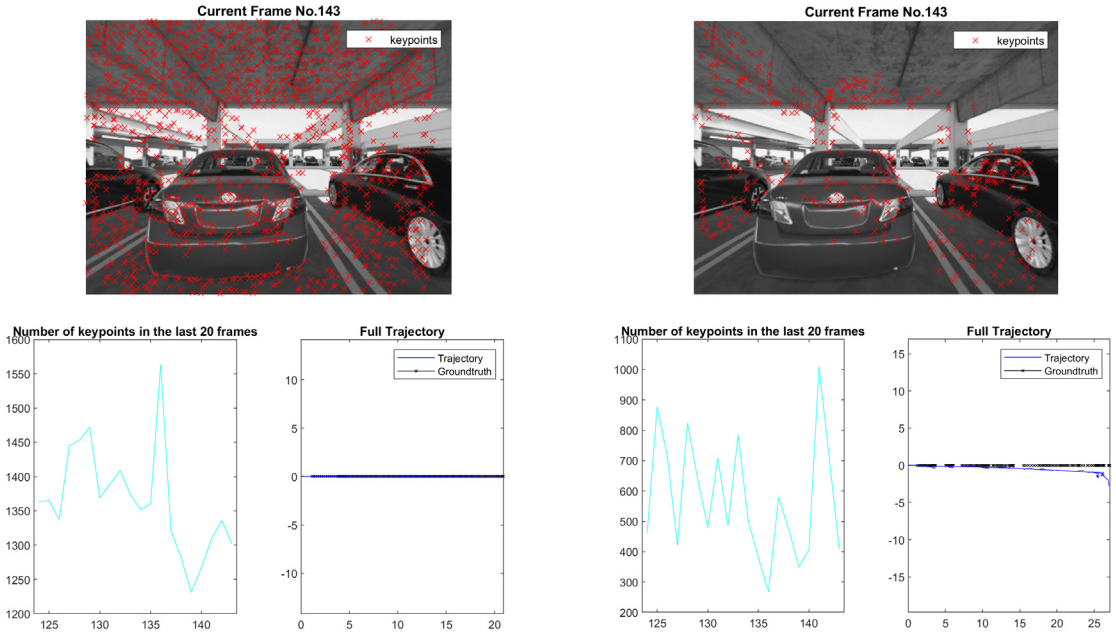
(a) Originally extracted keypoints

(b) Uniformly extracted keypoints

(c) ANMS extracted keypoints

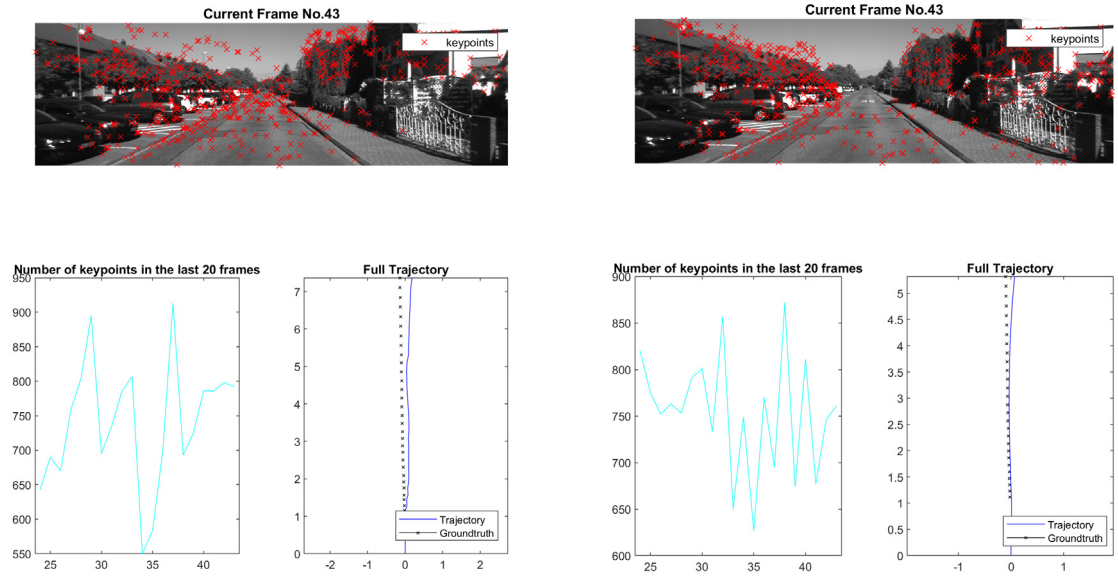
Fig. 2: Distribution of keypoints by different methods

B. Pose and triangulation refinement



(a) parking dataset with DLT refinement (b) parking dataset without DLT refinement

Fig. 3: Effect of afterwards DLT in pose estimation



(a) KITTI without BA, without triangulation refinement (b) KITTI without BA, with triangulation refinement

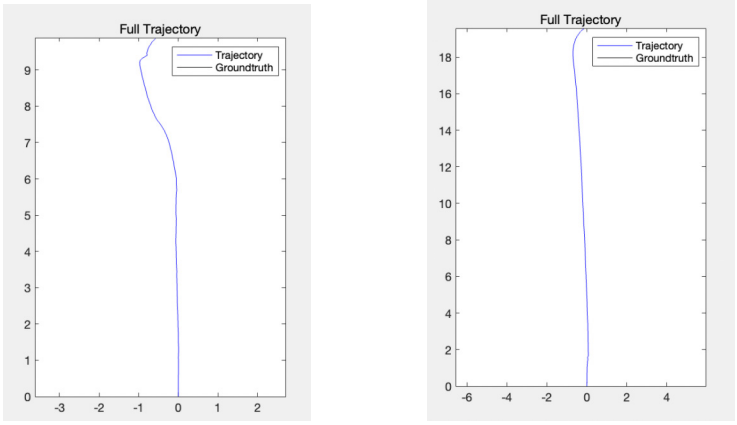
Fig. 4: Effect of triangulation refinement

We test pose refinement with DLT once the maximum set of inliers has been determined and find it important to our pipeline. Without refinement, it leads to deviation from course on parking dataset, and collapse at the beginning on KITTI and Malaga dataset. Triangulation refinement is another step contributes to trajectory consistency, as shown in Figure 4. It slows down the full visual odometry a bit and is disabled in parameter tuning. However

we find that adding this refinement is beneficial to our distance threshold in landmarks selection, as shown in the figure, more far away landmarks are removed.

C. Bundle adjustment

We use bundle adjustment for joint optimization of structure and motion. For coding we implement the exercise 9 which use jacobian matrix pattern for faster optimization. Different from exercise we hold the first two poses fixed to keep the same scale. We compare the non-bundle adjustment and bundle adjustment trajectory as Fig 5 .The full trajectory is much smooth and straight after optimization.



(a) Without Bundle adjustment the road is curvy

(b) Much straight and better result with Bundle adjustment

Fig. 5: Effect of bundle adjustment in kitti first 140 frame

IV. Additional feature: custom dataset

A. Introduction

Since one member of us: **Qi Ma is currently working in AMZ racing team** which also interested at adding Visual Odometry in the FSG Competition, we recorded AMZ racing dataset including calibration data and autocross autonomous driving data. This dataset is challenging because of textureless road and repetitive feature.

(a) Dataset example

(b) Calibration example

Fig. 6: Custom dataset

B. Data collection

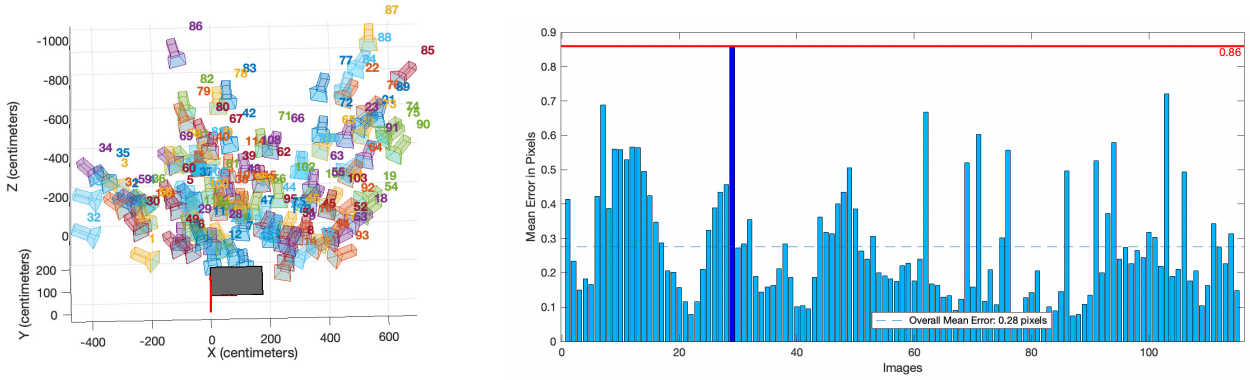
We collect the data with the Pilatus driverless in 7 vehicle which have 3 camera in total. We use the forward camera because it installed horizontally and captures most features. In data collection the vehicle was switched to autonomous driving mode and all data is collected by its own. The road consists of different cones where blue indicate left track and yellow indicates right. The original image size is 2592 x 1600 but the interested area is basically on road plane so we crop it to 2592 x 640 images.

(a) Pilatus driverless vehicle (b) Sensor setup

Fig. 7: Collecting data with AMZ Driverless

C. Camera calibration

In this part we implement the Zhang's calibration method. We collect the data with a big Checkerboard which has 25cm square length. In order to match the actual competition scene we take videos from multiple angles and also cover long distances like Figure 9. We use the Camera Calibrator application from Matlab and as the Figure 9 shows, all frames reprojection error are smaller than 1 pixel therefore the calibration is good.

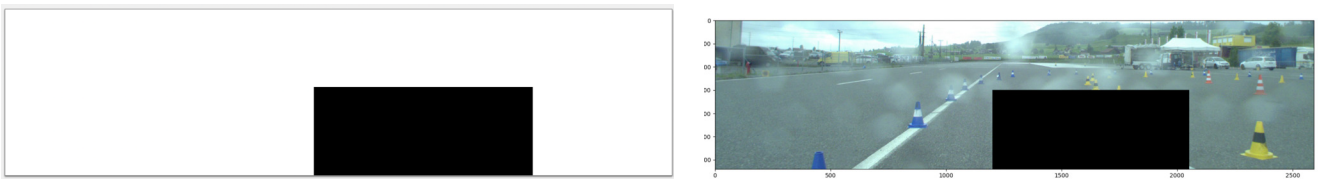


(a) Calibration camera view (b) Reprojection error

Fig. 8: Calibration result

D. Lost tracking with original solution

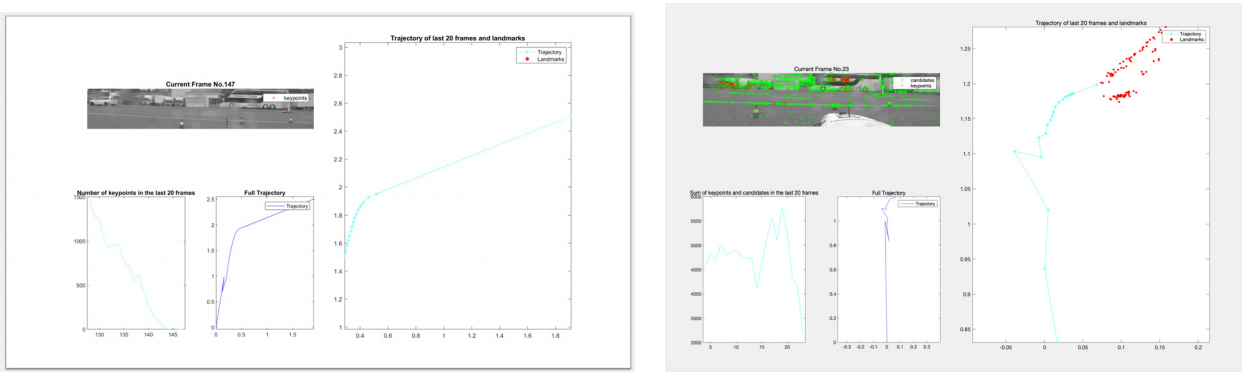
1) Mask racing car nose: Different from other provided dataset, in our customized dataset the nose of racing car always appearing in the frames, which may caused bad matching and outliers. So we add a binary mask to get rid of the nose part.



(a) Binary mask for car nose (b) No keypoints in mask

Fig. 9: Illustration of masking

2) Failure analysis: For using KLT point tracker, there are some crucial assumption we need to pay attention.1)Brightness constancy, 2)Temporal consistency and 3)Spatial coherency. In the racing dataset the first and second assumption are violated since the brightness changes fast because of strong sun reflections and drops of water shown in 6. Moreover, the racing car moves fast especially in turning, the movement is big so our original solution lose tracking soon like Figure 10(a).



(a) Lose tracking in turning with KLT

(b) Successfully tracking with feature based method

Fig. 10: KLT tracking and matching-based method

E. From direct method to feature based method

Unlike the direct methods, the feature based method use invariant feature descriptor and matches them in successive frames. The robust feature detectors and descriptors allow matching images under large illumination and view-point changes [2]. Inspired by this we use the matching-based VO as majority did. As shown in Figure 12

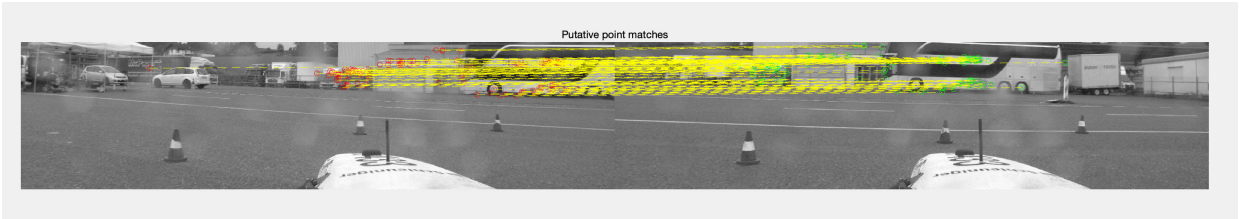


Fig. 11: Sift matching in large view point change

1) SIFT feature based matching VO: In this feature based method, we replace the module Track obtained keypoints in the frame using KLT with SIFT feature matching. from Matlab. For all tracked keypoint we will try match their correspondent feature with new detected features in next frame. The successfully detected keypoints which passes the triangulation check will be used to triangulate new landmarks. By adopting this method, we can do better in turning case as shown in Figure 10(b).

2) Discussion: Although matching based method is robust to large viewpoint changes, we still sometimes suffer from tracking lost and scale shift. we discussed here about possible reason and solution for the next stage of development.

- **Sequential matching shortcomings**: Using Markov way of data flow is intuitive and reasonable, however, for the customized dataset the global SFM could have done better. For example in a sharp turn we may not track the keypoints because of large rotation and too close scene. But we may have seen the scene in previous frame which can be used as matching.
- **Large scene changes**: In our customized dataset we suffer from large scene changes which means sometimes all reliable feature are very far but suddenly we are facing a close building. The large-scale scene changes confuse the tracker and the scale shift very soon.
- **Textureless and repetitive features**: In the customized dataset we also suffer from poor feature like road and repetitive features like cone and road track.

For possible solution we will try global SFM and also inertial integration in future. Moreover, adding loop closure will alleviate the problem of scale shift.

3) Feasibility verification of solutions: In order to verify the effect of global SFM, we use the COLMAP for custom dataset. The trajectory is locally right and obviously the loop is not closed due to shift of SFM. Therefore loop closure is essential.

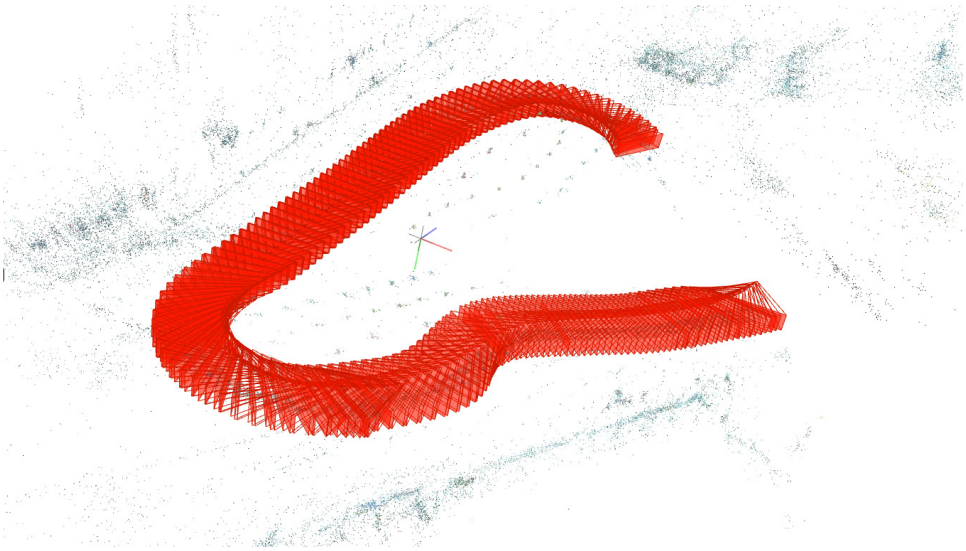


Fig. 12: COLMAP global SFM result

REFERENCES

[1] Oleksandr Bailo, Francois Rameau, Kyungdon Joo, Jinsun Park, Oleksandr Bogdan, and In So Kweon. Efficient adaptive non-maximal suppression algorithms for homogeneous spatial keypoint distribution. *Pattern Recognition Letters*, 106:53–60, 2018.

[2] Christian Forster, Zichao Zhang, Michael Gassner, Manuel Werlberger, and Davide Scaramuzza. Svo: Semidirect visual odometry for monocular and multicamera systems. *IEEE Transactions on Robotics*, 33(2):249–265, 2016.

APPENDIX

TABLE I: Parameters Tuned in Pipeline

Process	Parameter Name	KITTI Value	Malaga Value	Parking Value
Harris	points	800	300	500
	quality	1e-7	1e-7	1e-7
	filter	3	3	3
KLT	max bidirectional error	3	3	3
	block size	33	35	33
	max iterations	50	50	50
	reliability of track	0.99	0.99	0.99
	confidence score	1	1	1
Triangulation	distance factor	5	5	5
	angle threshold	1.5	1.5	1.5
	reprojection error threshold	30	30	20
BA	period	10	10	6
	windows size	20	20	3
	max iterations	15	20	50