Research Article

INTERNATIONAL JOURNAL OF
ADVANCED ROBOTIC SYSTEMS

International Journal of Advanced Robotic Systems
September-October 2017: 1–11
© The Author(s) 2017
DOI: 10.1177/1729881417735667
journals.sagepub.com/home/arx
SAGE

Semantic segmentation–aided visual odometry for urban autonomous driving

Lifeng An[1], Xinyu Zhang[2], Hongbo Gao[3] and Yuchao Liu[1]

Abstract

Visual odometry plays an important role in urban autonomous driving cars. Feature-based visual odometry methods sample the candidates randomly from all available feature points, while alignment-based visual odometry methods take all pixels into account. These methods hold an assumption that quantitative majority of candidate visual cues could represent the truth of motions. But in real urban traffic scenes, this assumption could be broken by lots of dynamic traffic participants. Big trucks or buses may occupy the main image parts of a front-view monocular camera and result in wrong visual odometry estimation. Finding available visual cues that could represent real motion is the most important and hardest step for visual odometry in the dynamic environment. Semantic attributes of pixels could be considered as a more reasonable factor for candidate selection in that case. This article analyzed the availability of all visual cues with the help of pixel-level semantic information and proposed a new visual odometry method that combines feature-based and alignment-based visual odometry methods with one optimization pipeline. The proposed method was compared with three open-source visual odometry algorithms on Kitti benchmark data sets and our own data set. Experimental results confirmed that the new approach provided effective improvement both on accurate and robustness in the complex dynamic scenes.

Keywords

Visual odometry, dynamic scene, semantic segmentation, deep learning

Date received: 31 January 2017; accepted: 19 July 2017

Topic: Special Issue – Multimodal Fusion for Robotics
Topic Editor: Huaping Liu
Associate Editor: Huaping Liu

Introduction

Visual odometry (VO) is the most important part of visual simultaneous location and mapping (V-SLAM) algorithm and has already been widely used in the optical mouses, small mobile robot, and unmanned aerial vehicles (UAVs). As a kind of relative affordable and lightweight solution, VO plays a more and more important role in visual-based navigation system for autonomous driving cars.

The VO term had been first used by Nistér in 2004,[1] but the relevant researches in this area had been focused over 30 years.[2,3] VO uses camera as main sensor and takes the current image frames as input and compares with previous frame, then estimates camera's pose transformation and the trajectory like wheel odometry does.[2] Moravec proposed first visual motion estimation pipeline and used it for

[1] Department of Computer Science and Technology, Tsinghua University, Beijing, China
[2] Information Technology Center, Tsinghua University, Beijing, China
[3] Key Laboratory of Automotive Safety and Energy, Tsinghua University, Beijing, China

Corresponding authors:
Xinyu Zhang, Information Technology Center, Tsinghua University, Beijing 100084 China.
Email: xyzhang@tsinghua.edu.cn

Hongbo Gao, Key Laboratory of Automotive Safety and Energy, Tsinghua University, Beijing 100084, China.
Email: ghb48@mail.tsinghua.edu.cn

NASA Mars rover in 1980.[4] Nistér found efficient five-point algorithm and built the first real-time VO pipeline.[1] A common VO pipeline included four steps: *capture*, *matching*, *estimation*, and *optimization* or *filtering*. The *capture* step grabbed and rectified the images taken from cameras. The *matching* step computed point-wise or patches-wise correspondences. It could be achieved by a feature point-level matching[5] or directly used raw pixel/subregion values for patches alignment.[6,7] Optical flow (OF)[8] and tracking method[9] could also be integrated in this step and reduce the computation cost of feature extraction and matching. The *estimation* step often solved a perspective from *n* points (PnP) problem to recover camera pose transformation and 3-D points' structure,[10] and finally *optimization* or *filtering* step recovered the part or whole trajectory a local/global optimization process or a filter method.[11–13]

Most of the VO methods worked fine in static indoor scene. However, in the outdoor environment, especially in dynamic urban scenes, too many factors impacted accuracy and availability of VO. Lots of moving objects were the biggest challenges. High-level object recognition or segmentation methods could provide semantic information for better environment understanding. This could be done by cameras only or by fusion multiple sensors.[14–19] But some distance sensors, like sonar and lidar, needed accumulated multiple data frames for effective recognition,[20] which limited their robustness in dynamic environments. Buczko and Willert present a feature-adaptive scaling method for outliers removal.[21] Engel et al. proposed a direct sparse odometry (DSO) approach that jointly optimizes the full likelihood for all involved model parameters.[22] These methods still tried to overcome the problem in measurement level. Recently, deep learning had been used successful in object detection and image semantic segmentation[23] and spatial semantics learning.[24] These semantic information could provide more causal factors for visual motion estimation and helped to improve robustness in complex environment.[25] Mohanty et al. proposed a deep VO method that estimated the odometry vectors between any arbitrary image pair by a trained convolutional neural network (CNN).[26] These efforts could provide a better way to look inside how VO using features or pixels and make evaluation method of VO toward the way that humans could easily understand.

This article focused on dynamic scene VO problem for the urban autonomous driving cars. A robustness VO system could reflect a kind of cognitive process how to understand dynamic scenes correctly and discover the real motions from not only low pixel-level matching but also high-level semantic understanding. This work analyzed the accurateness and robustness impacts of different semantic segmentations from a statistical point of view. Then a deep learning neural network was used for preprocessing pixel-level semantics. These semantic information were used to select reasonable visual cues and remove outliers in *matching* step. After that, a new VO pipeline was provided to combine feature-based method and alignment-based VO method. The contributions of this article were a novel semantic-aided probabilistic model for outliers removal and alignment patches selection in dynamic scenes and a new feature-based and alignment-based combined VO pipeline.

This article was organized as follows: In section "Related work," we reviewed the relative VO works. In section "System overview," we introduced the semantic segmentation by a deep learning network and described our algorithm model. In section "Experimental results," we evaluate the accuracy and robustness for the three different models on a Kitti benchmark data set and our own real-world data sets. Finally, in section "Conclusions," we concluded the method and lined out future work.

Related work

A lot of efforts and researches had been focused on developing usable VO systems which show successful applications. Parallel tracking and mapping (PTAM)[27] was a first feature-based VO method running in real time. It had two parallelized threads computing motion estimation and mapping, respectively. PTAM ran an efficient bundle adjustment (BA) on all keyframes, which limits it could only be used in small environment. Civera et al. proposed EKF-based method using one-point random sample consensus (RANSAC) and reduced the size of the data subset to instantiate a hypothesis to one point.[28] Dense tracking and mapping (DTAM)[13] was a typical direct method and computed pose transformation by whole image alignment on a depth map. Semi-direct visual odometry (SVO) algorithm proposed by Forster et al.[6] used a sparse model-based image alignment algorithm for motion estimation, which tracks some corner points and uses the 4 × 4 patches around them for direct alignment. Geiger et al. presented VISO[29] to compute six-DOF motion of a moving stereo/monocular camera and tested it in urban scene data set Kitti.[30] DSO puts intensity resident, exposure time, attenuation, and irradiance in one energy function and optimized motion estimation, geometrical, and photometric calibration in a joint framework simultaneously. A lot of V-SLAM system contained VO part. ORB_SLAM2 had a feature-based VO thread and a loop closure thread[31] and could compute the camera trajectory real time in a wide variety of environments.

Most of the successful VO or V-SLAM systems focused on static environment. In dynamic scene, removing outliers became a more necessary step for the accurate motion estimation. Choosing correct pixels or keypoints, which represents the real camera motion, would help to improve VO robustness in the complex scene. As a common tool, RANSAC-based method is often used to reject outliers.[32] Given an expected rate of success *P*, the necessary iteration

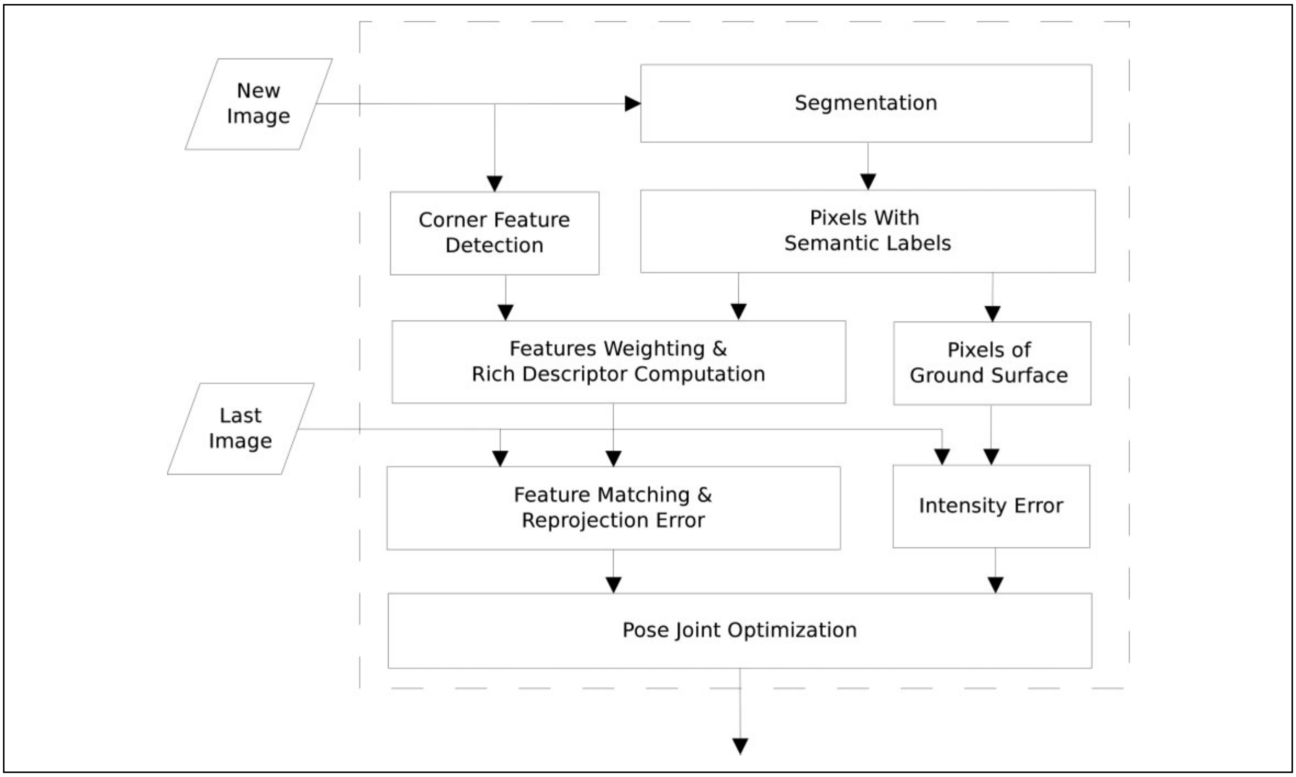

Figure 1. Semantic segmentation–aided visual odometry pipeline.

times of RANSAC N could be computed by the number of data points s and outliers rate ε

$$N = \frac{\log(1-p)}{\log(1-(1-\varepsilon)^s)} \tag{1}$$

With higher ε, N could reach thousand times in many cases. Preemptive RANSAC[33] tried to fix N using motion hypotheses. The progressive sample consensus[34] computed similarities of the correspondences for ranking and sampling to increase the convergence process. Using RANSAC-based method to reject outliers has an assumption that the noise samples are far less than the correct samples. They only tried to find a probabilistic stable set of inliers by growing iterations. In dynamic urban traffic, this assumption would not be always hold. A front-view monocular camera with limited field-of-view (FOV) lens could suffer from the occlusions and disturbances from the moving vehicle nearby. RANSAC-based methods could not guarantee to choose the pixels or keypoints that belong to static object in real world and could lead to hard data association for motion estimation in dynamic complex scene. Recently, Buczko and Willert proposed a normalized reprojection error method[21] which shows an increased error for outliers and a constant offset for inlier. But this method focused on high-speed scene with an assumption of small rotation, longitudinal motion only and didn't consider semantic attributions of all points.

A robust VO in dynamic scenes should have the ability that distinguishes static object and moving traffic participants. In this semantic level, scene understanding could help VO by a higher level visual cues selection process. Civera et al. proposed a semantic SLAM using a monocular extended kalman filter (EKF) SLAM and inserted 3-D objects into geometric map.[35] Anand et al. trained a graphical model for contextually guided semantic labeling.[36] Yang et al. proposed a method to solve navigation and vehicle distance estimation simultaneously and used dynamic object tracking to divide view field of camera into static and dynamic parts.[37] This method would be hard to distinguish a moving object which has the same speed to observer. Geiger et al. provided a probabilistic model combining semantic scene labels, occupancy grid, vanishing points, and moving object tracklets to discover the intersection model.[38] In his work, the semantic labels provide a probability of label class given a road layout. In that work, the labels are three simple classes, foreground, background, and sky, and contribute little for motion estimation. Pop-up SLAM proposed by Yang used pop-up model and large-scale direct monocular SLAM (LSD)[39] to predict depth and demonstrated that scene understanding improves state estimation and dense mapping.[40]

System overview

This section introduced semantic segmentation–aided VO (SAVO) method. The whole pipeline is shown in Figure 1. The VO system took monocular RGB image sequence $I_0, \ldots, I_{k-1}, I_k, \ldots, I_n$ as input, followed by a feature detection pipeline and deep learning segmentation network. The feature points in current frame were computed by pointwise matching to previous image and weighted by the segmentation category labels, which depended on their contribution to reduce reprojection errors. Then the inlier points were sampled by a RANSAC process with the semantic weights and used to estimate camera pose translation. The selected segmentation patch, which had semantic meaning of static, was used to direct alignment between previous frame and current one. The two motion assumptions from two paralleled methods were fused for output as final pose estimation.

Throughout this work, the image at time step k was I_k, and the pose of camera was represented by $T_k \in \mathrm{SE}(3)$. The transformation between two consecutive frames I_{k-1} and I_k could be $T_k = T_k{}' T_{k-1}$[41]

$$T_k = \begin{pmatrix} R_k & t_k \\ 0 & 1 \end{pmatrix} \tag{2}$$

with rotation $R_k \in \mathrm{SO}(3)$ and translation $t \in R_k^3$. A 2-D pixel was represented by $\mathbf{u} = [u, v]^T, \mathbf{u} \in \mathbb{R}^2$. A 3-D world point was $\mathbf{X} = [x, y, z]^T, \mathbf{X} \in \mathbb{R}^3$. The project function $\pi : \mathbb{R}^2 \mapsto \mathbb{R}^3$ mapped 2-D pixel $\mathbf{u}$ to 3-D point $\mathbf{X}$

$$\mathbf{X} = \pi(\mathbf{u}, d) \tag{3}$$

d was an inverse depth of pixel $\mathbf{u}$. If $d = 1$, the pixel would be projected into a 3-D unit sphere surface.

Semantic segmentation

Semantic segmentation was widely used in autonomous driving for scene parsing and understanding. This work used a modified SegNet with a pretrained driving model proposed by Badrinarayanan et al.[23] SegNet was a deep learning encoder network with 13 convolutional layers of VGG16 model[42] and had 12 segmentation categories, including *Sky*, *Building*, *Pole*, *Road Marking*, *Road*, *Pavement*, *Tree*, *Sign Symbol*, *Fence*, *Vehicle*, *Pedestrian*, and *Bike*. These category information provided a kind of semantic understanding for an urban road scene, which could help us to distinguish the object is movable or not. To find static pixels was a very important factor for VO in dynamic urban traffic scene. A moving object would bring too much uncertainty in motion estimation process. This work assumed that each semantic category would have different contributions to VO. The contribution was relative to the errors of motion estimation brought by the pixel's category. For example, in a dynamic urban traffic scene, a pixel from *Building* category could be more reliable than a pixel from a *Car* for motion estimation and should be sampled as a candidate with higher probability in VO process. The contribution of category c_i was represented by a probability variable P_{c_i} in this article

$$P_{c_i} = \frac{1}{Z} \frac{n_{c_i}}{\sum_{i=0}^{n_{c_i}} r_{c_i}} \tag{4}$$

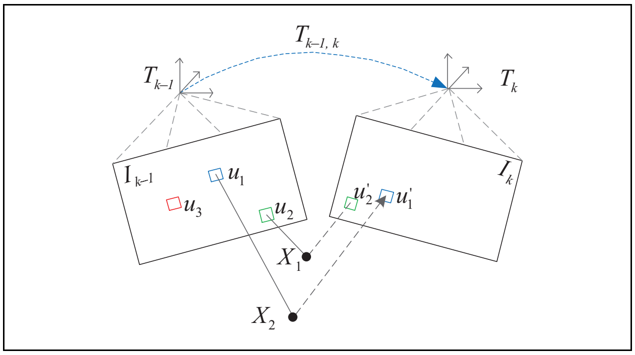

Figure 2. 2-D points $\mathbf{u_1}$, $\mathbf{u_2}$, $\mathbf{u_3}$ in the frame I_{k-1} could be projected to 3-D world points $\mathbf{X_1}$, $\mathbf{X_2}$ by π and reprojected to $\mathbf{u}'_1$, $\mathbf{u}'_2$ in frame I_k with optimization variable $T_{k-1,k}$. The reprojection error could be computed by $\mathbf{u}'$ and its ground truth point V. Here not all of the matching pairs should be taken into account. For example, $\mathbf{u_3}$ belongs to a movable object category and should have lower contribution to real motion estimation or just dropped out.

Z was a normalized factor. r_{c_i} was the reprojection error of category c_i

$$r_{c_i} = \sum (\hat{\mathbf{u}}_{j,I_k} - \mathbf{u}'_{j,I_k}) = \sum (\hat{\mathbf{u}}_{j,I_k} - E * \mathbf{u}_{j,I_{k-1}}) \tag{5}$$

E was the essential matrix

$$E_{k-1,k} = K^{-1} * [t_{k-1,k}]_x * R_{k-1,k} * K \tag{6}$$

In some open benchmark data set, the ground truth of transformation between frames $T_{k-1,k}$ was provided. So the reprojection errors of every semantic categories could be computed by cumulation of its pixels' reprojection errors. In some ways, these errors implicitly provided the level of contributions to correct motion estimation.

Visual odometry

Feature-based method. This part was similar to a traditional feature-based approach. In the current image frame I_k, the feature points and their rich descriptors were extracted. A k-nearest neighbor (KNN)-based method matched them to the keypoints from the previous frame I_{k-1}. The correspondence of these 2-D points was refined by direction symmetry check and ratio check. Given the 2-D correspondences, the essential matrix E could be computed by epipolar constrain equations and PnP method. The main difference of the proposed method was the sampling step. Rather than choosing the correspondent 2-D point pairs randomly in the RANSAC iteration, this work sampled them depending on their contribution probabilities which came from pixels' semantic segmentation described as before.

The transformation $T_{\text{feature-based},k} = [R|t]$ with rotation R and translation t was computed by $\mathrm{SVD}(E)$ and optimized by a window BA. The 2-D keypoint $\mathbf{u}_j$ in previous image I_{k-1} had a correspondent keypoint $\hat{\mathbf{u}}_j$ in current image I_k (Figure 2)

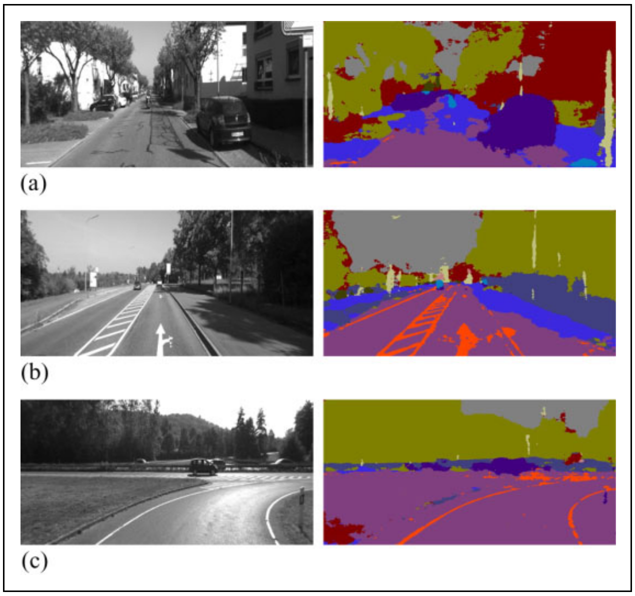

Figure 3. Examples of Kitti data set sequence (a) 00, (b) 01, and (c) 02 and their semantic segmentation results.

$$\mathbf{u}^{'} = \pi^{-1}(X^{'}) = \pi^{-1}(T_k X) = \pi^{-1}(T_k \pi(\mathbf{u}, d)) \tag{7}$$

$\mathbf{u}^{'}$ was the reprojected 2-D point in I_k from 2-D point $\mathbf{u}$ in I_{k-1}. The correspondent point was $\hat{\mathbf{u}}$ in I_k from point-wise matching of $\mathbf{u}$. So the reprojection error $r_{\text{reprojection}}$ was

$$r_{\text{reprojection}}(T_{k-1,k}, \mathbf{u}) = \| \mathbf{u}^{'} - \hat{\mathbf{u}} \| \tag{8}$$

Then the cost function $J_{\text{reprojection}}$ was built by

$$J_{\text{reprojection}} = \sum_i \| r(T_{k-1,k}, p_i) \|^2 \tag{9}$$

And every transformation $T_{k-1,k}$ between two continuous frames was solved by least-squares (LS) minimization method

$$T_{k-1,k} = \arg\min_T J_{\text{reprojection}} \tag{10}$$

Alignment-based method. This process used a semi-dense image alignment framework. Comparing with traditional direct method, there were three main differences: Firstly, instead of using whole image pixels for alignment, the proposed method only used partial image that had specific segmentation labels. These patches had the semantic priori knowledge that they were motionless objects. Secondly, these patches indicated a set of objects that belonged to one planar surface, which was a basic assumption for image alignment. Thirdly, the depth from these candidate patches or pixels from mono-camera should be estimated easily and could be used for weighting or sorting residual blocks in LS minimization process. In this work, the pixels, which labeled with *Road Marking*, *Road*, and *Pavement*, were selected for indirect VO estimation. The 3-D points belonged to these patches were regarded as being static to global coordinate system and assumed to lay on a rough road plane.

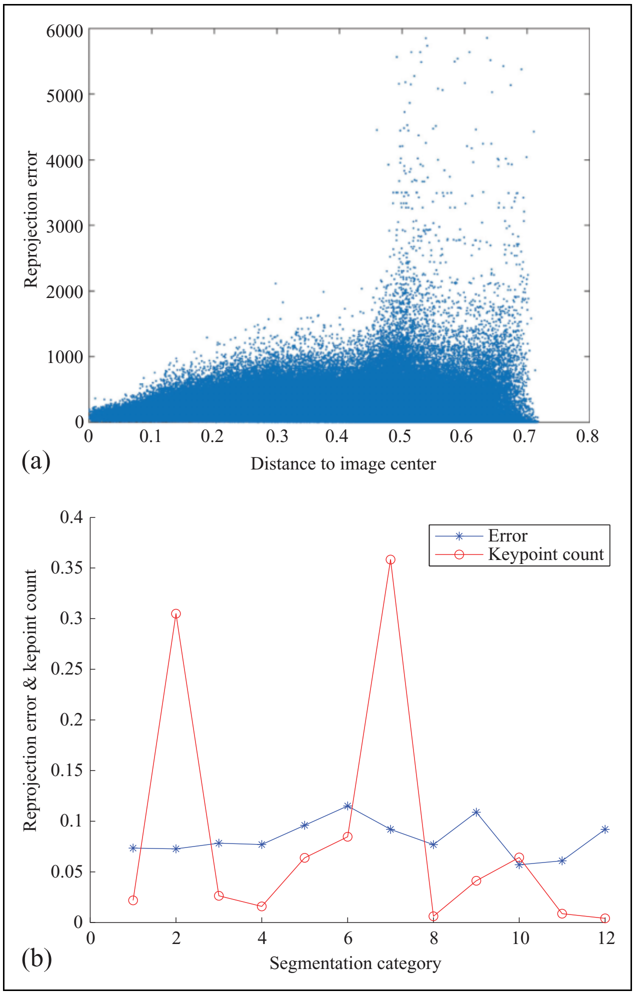

Figure 4. Reprojection errors. (a) Considering coordinate distance to image center. (b) Errors distribution on 12 semantic segmentation categories. All statistic results were computed on Kitti odometry data set.

The candidate patch set Ω that belongs to category c was represented by I_{Ω_c}. And every pixel in the set was filtered by an intensity gradient threshold, which provided a semi-dense 2-D point and dropout the trivial points with less texture gradients. The pose transformation $T_{k-1,k}$ between images $I_{\Omega_c,k-1}$ and $I_{\Omega_c,k}$ was computed by minimizing the intensity residuals $r_{\text{intensity}}$

$$r_{\text{intensity}}(T_{k-1,k}, \mathbf{u}) = I_{\Omega_c,k-1}(u) - I_{\Omega_c,k}(u^{'}) \tag{11}$$

Then the cost function $J_{\text{intensity}}$ was built by

$$J_{\text{intensity}} = \sum_{i \in \Omega_c} \| r(T_{k-1,k}, \mathbf{u}_i) \|^2 \tag{12}$$

And the transformation $T_{k-1,k}$ was formed as a LS minimization optimization problem

$$T_{k-1,k} = \arg\min_T J_{\text{intensity}} \tag{13}$$

The whole problem cost function J combined the costs of two parts

Table 1. Rotation error.

Sequence	SAVO	VISO	DSO	ORB_SLAM2
0	0.000156	0.000702	0.005729	0.000056
1	0.000621	0.001238	0.002207	0.001579
2	0.000174	0.000519	0.004907	0.000049
3	0.000166	0.000495	0.005719	0.000033
4	0.000126	0.000285	0.000204	0.000046
5	0.000163	0.000803	0.005072	0.000041
6	0.000109	0.000498	0.001381	0.000043
7	0.000372	0.001375	0.009974	0.000057
8	0.000172	0.000657	0.005437	0.000053
9	0.000231	0.000548	0.005485	0.000061
10	0.000343	0.000867	0.004496	0.000214

SAVO: semantic segmentation–aided visual odometry; DSO: direct sparse odometry.

Table 2. Translation error.

Sequence	SAVO	VISO	DSO	ORB_SLAM2
0	0.017481	0.168392	0.638761	0.525636
1	0.159909	0.424336	0.962693	0.955000
2	0.019727	0.237248	0.704573	0.672709
3	0.013100	0.293990	0.981139	0.913303
4	0.008170	0.166139	0.980135	0.980062
5	0.018440	0.140762	0.605002	0.504923
6	0.011571	0.116012	0.579200	0.546418
7	0.033184	0.163079	0.605035	0.513819
8	0.022087	0.121467	0.622131	0.563724
9	0.033228	0.162112	0.747357	0.704958
10	0.039921	0.182698	0.927605	0.804431

SAVO: semantic segmentation–aided visual odometry; DSO: direct sparse odometry.

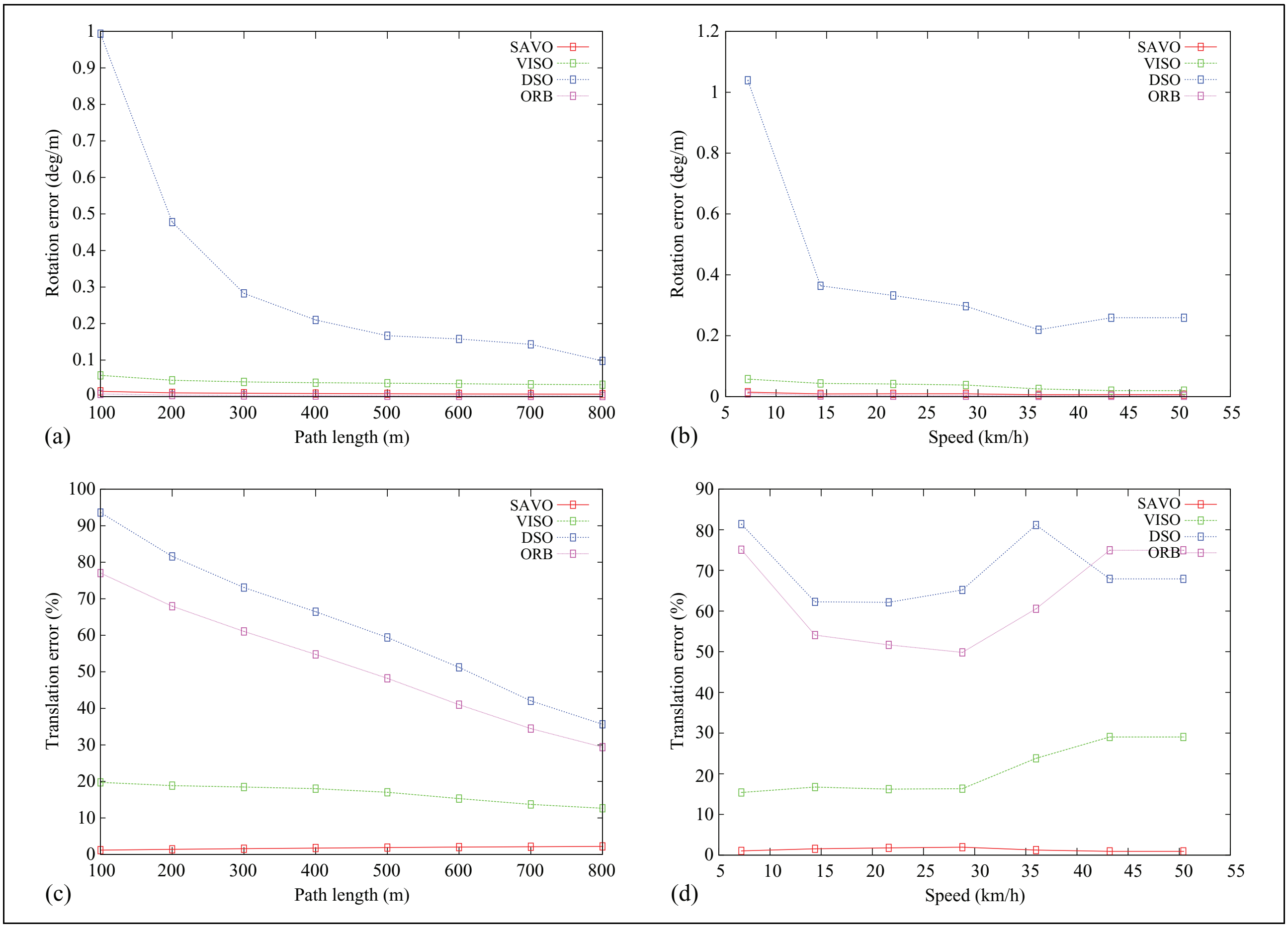

Figure 5. The transformation errors of proposed method SAVO on Kitti sequence00. SAVO showed lowest values on both rotational and translational errors. (a) Rotational error versus sequence length. (b) Rotational error versus driving speed. (c) Translational error versus sequence length. (d) Translational error versus driving speed. SAVO: semantic segmentation–aided visual odometry.

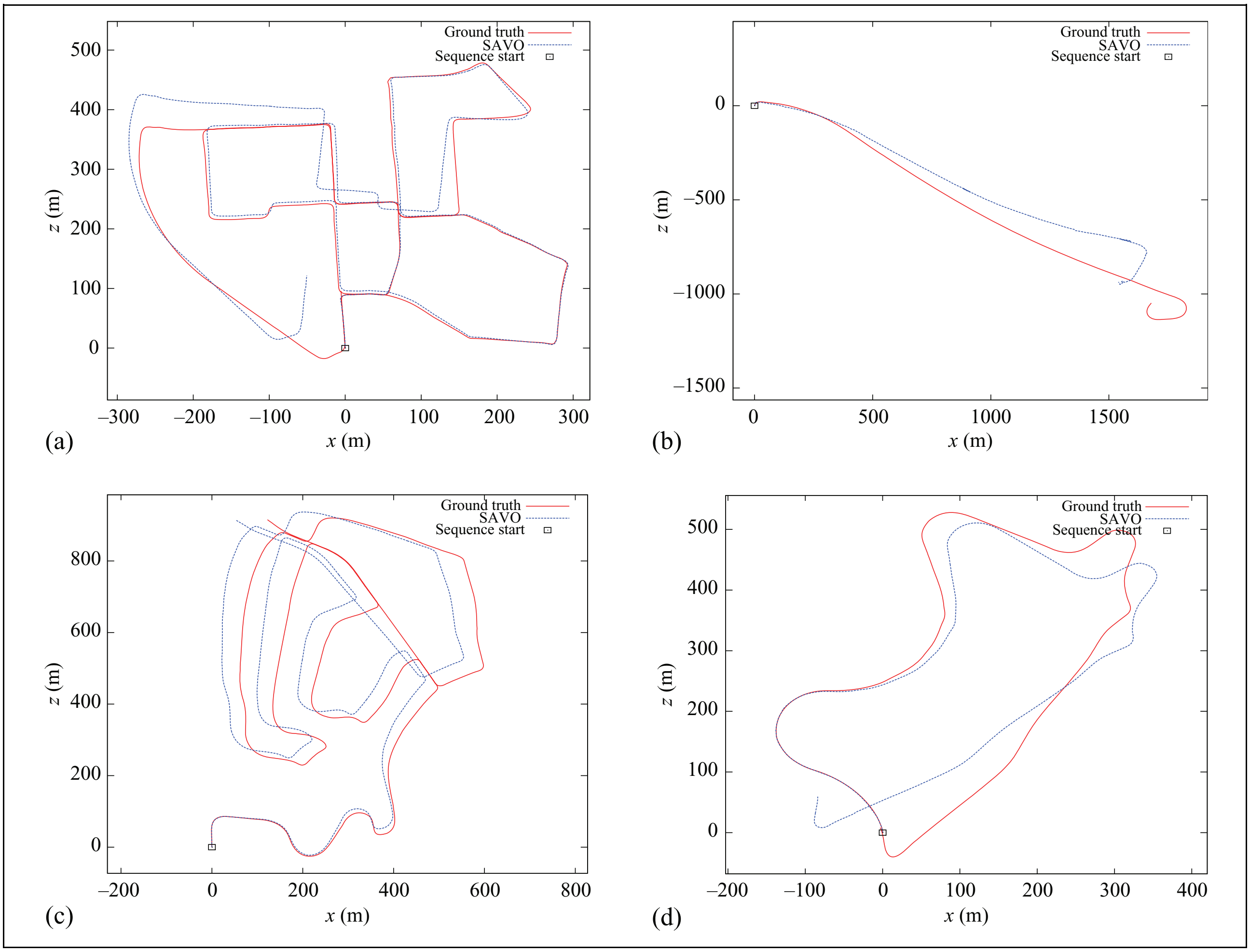

Figure 6. Trajectories of SAVO on Kitti odometry data set sequences (a) 00, (b) 01, (c) 02, (d) 09. SAVO: semantic segmentation–aided visual odometry.

$$J = \alpha J_{\text{reprojection}} + (1-\alpha) J_{\text{intensity}} \tag{14}$$

α was a hand-tuned parameter by experience. Lie group representation and dual number method were used to compute Jacobian. The pose was represented by $\xi = [\rho\, \phi]^T \in R^6$, $\rho \in R^3$ was 3-D translation parameter, and $\phi \in R^3$ was the rotation parameter including yaw, pitch, and roll. The Jacobians of transformation in $J_{\text{reprojection}}$ and $J_{\text{intensity}}$ part had the same formulation

$$\frac{\partial r}{\partial \delta\xi} = \left[\frac{\partial \rho}{\partial \delta\xi} \Big| \frac{\partial \phi}{\partial \delta\xi}\right]$$

$$= \begin{bmatrix} \frac{f_x}{z'} & 0 & -\frac{f_x x'}{z'^2} & -\frac{f_x x' y'}{z'^2} & f_x + \frac{f_x x^2}{z'^2} & -\frac{f_x y'}{z'} \\ 0 & \frac{fy}{z'} & -\frac{fyy'}{z'^2} & -f_y - \frac{f_y y'^2}{z'^2} & \frac{f_y x' y'}{z'^2} & \frac{f_y x'}{z'} \end{bmatrix} \tag{15}$$

$X = [x, y, z]^T$ and $X' = [x', y', z']^T$ were the same 3-D point in different coordinate systems of I_{k-1} and I_k. f_x and f_y were camera focal lengths. The $J_{\text{intensity}}$ also had a partial derivative factor of $A\frac{\partial I}{\partial \mathbf{u}}$, A was a constant projection matrix for bird view of road surface, $\frac{\partial I}{\partial \mathbf{u}}$ was the gradient of intensity in the image. Then a classical LS method could be used to solve the problem.

Experimental results

Data set

Proposed model was tested on open benchmark Kitti odometry data set.[20] It contained 20 rectified stereo image sequences with calibration file and was recorded from a car traveling in urban blocks. We selected first 11 sequences which had pose ground truth and only used the monocular data of left camera. The proposed method was also tested on our autonomous driving data set. The intelligent vehicle platform was retrofitted from a Changan Raeton car, equipped with two AVT® 1394 Pike F-200c cameras capturing front view stereo images, one OxTS inertial IMU and Novatel RTK-GPS, and Velodyne VLP-16 LIDAR on

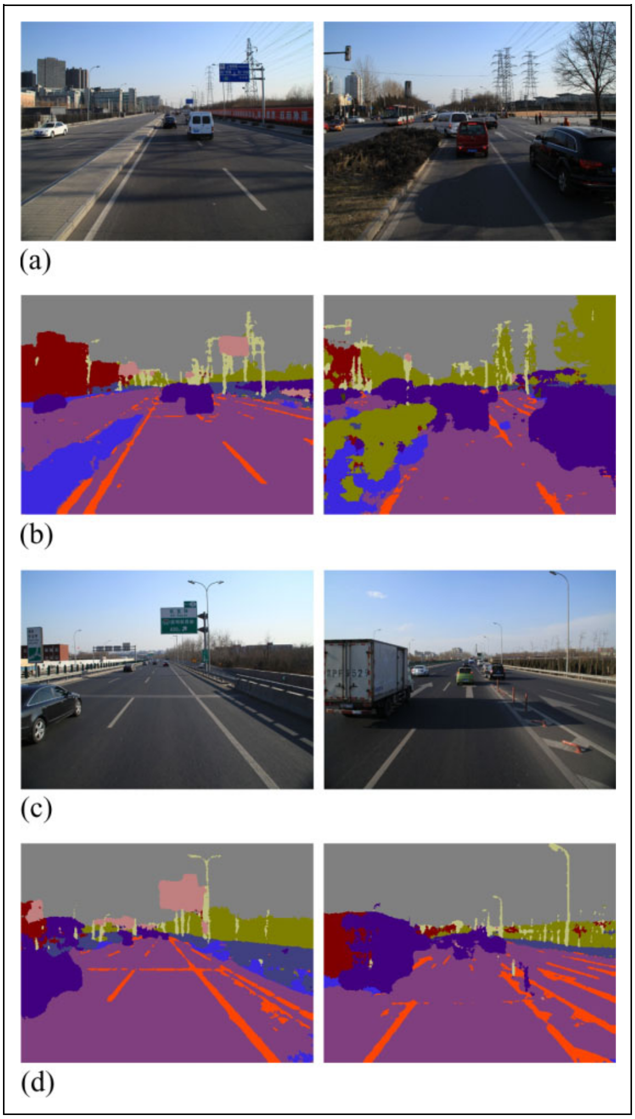

Figure 7. Trajectories of SAVO on Beijing Wuhuan data set sequences 00, 01. ORB_SLAM2, and DSO failed directly, and VISO could not provide any reasonable trajectories. SAVO's recovered trajectories with very big noise. The noises always came with those contained many dynamic vehicles. (a) seq00 raw image; (b) seq00 segmentation; (c) seq01 raw image; (d) seq01 segmentation. SAVO: semantic segmentation–aided visual odometry.

the top of vehicle. We collected about 20 km real road data on Wuhuan Road in Beijing with different traffic conditions.

Semantic segmentation contribution analysis

Eleven sequences in Kitti odometry data set were selected to evaluate semantic segmentation's reprojection error. First, segmentation process calculated the categories of every pixel on all Kitti odometry data set. Then, in each sequence, ground truth pose file was used to compute the transformation matrix $T_{k-1,k}$ between every two consecutive image frames. SIFT features were extracted and used for neighbor frame matching. All matching keypoints were filtered by consistence check and ratio check and were projected from previous frame to next one. The reprojection errors were computed with the ground truth and used to evaluate segmentation's contribution to odometry estimation. The feature point count in each semantic category and the impact of planar distance on image center of each feature point were also considered.

The reprojection errors were accumulated at each feature point and were sorted by planar distance to image center. As shown in Figure 3(a), horizontal axis represented normalized distance to image center and the vertical coordinates showed average value of reprojection error. For rectified images, this average error was increased with the growth of distance. In general, the border points in an image always had bigger errors. And nearer to image center, smaller were the errors. These errors were relative to inherent physical characteristics of camera lens and imaging sensors and hardly eliminated even after image rectification.

Figure 4(b) shows the normalized reprojection errors and keypoint counts about 12 semantic segmentation categories, such as *Sky*, *Building*, *Pole*, *Road Marking*, *Road*, *Pavement*, *Tree*, *Sign Symbol*, *Fence*, *Vehicle*, *Pedestrian*, and *Bike*. The higher count of keypoints means that image patches of this category had more pixels and stronger texture than other categories. The reprojection error represented the uncertainty for each patch used in motion estimation. In the figure, the categories of *Building* and *Tree* have more keypoints and lower error. In Kitti data set, the buildings and trees usually appeared in the middle part of images. And they also occupied more area with various textures. These pixels were motionless objects and they did not stay in one plane. So they could be suitable for feature-based motion estimation. The categories of movable objects, *Vehicle*, *Pedestrian*, and *Bike*, showed low reprojection error too. The reason was most of the vehicles in Kitti data set were parking cars, and the moving pedestrian and bike were not presented in most of the sequences. The *Bike* had higher speed than *Pedestrian* and brought higher errors. In a dynamic urban traffic scene, dropping out the pixels of cars, pedestrian, and bikes would reduce dynamic disturbance and avoid the difficulties of moving object tracking and motion judgment. Though the *Road* pixels were static, they were hard to extract feature points and led to higher error. On the opposite side, *Road Marker* pixels had lower error because they had more corner points than *Road* and could be easily tracked and matched.

VO performance analysis

This work was tested on both data sets and evaluated the performances compared with VISO, DSO, and

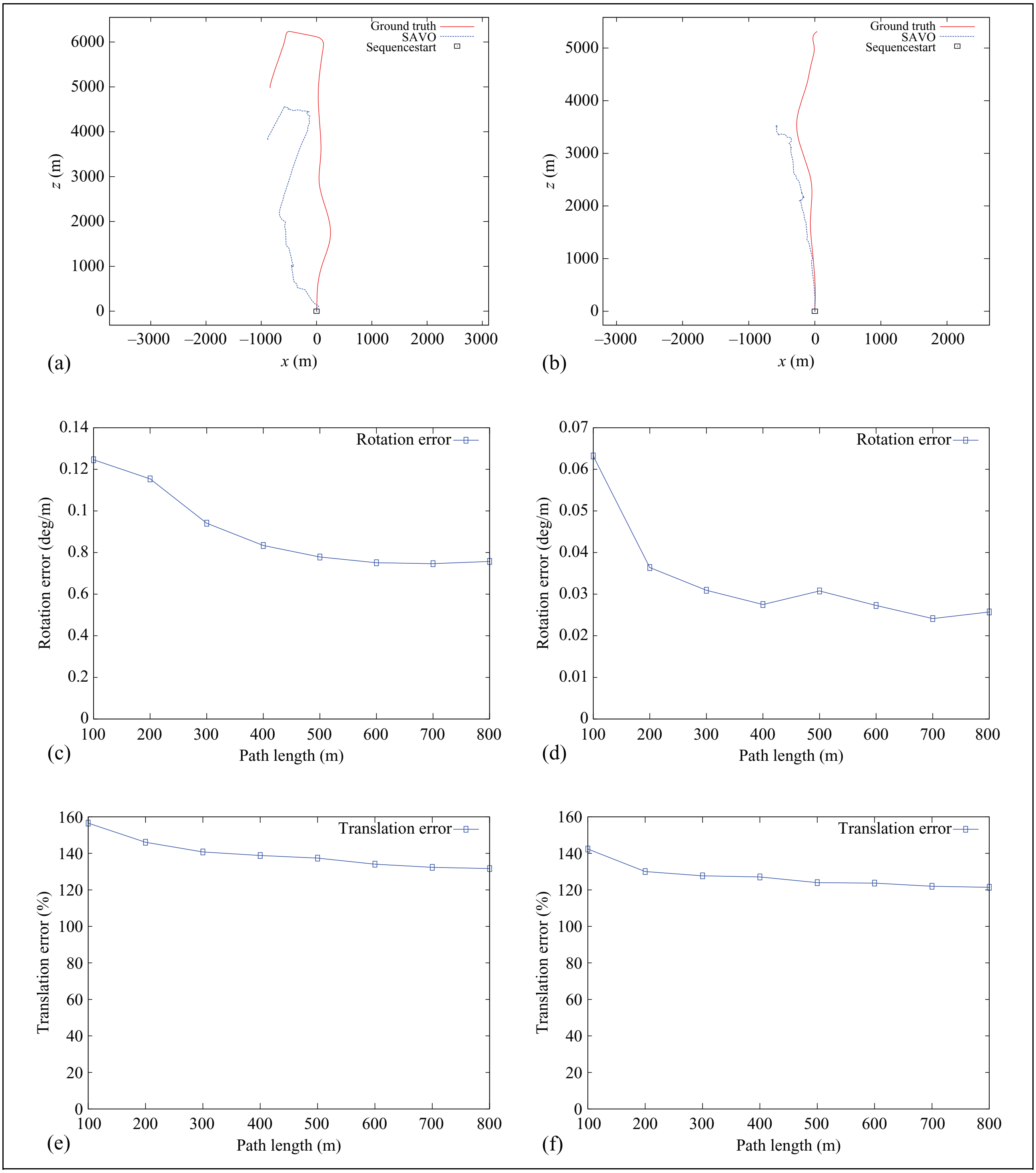

Figure 8. Trajectories, rotational, and translational errors of SAVO on Beijing Wuhuan data set sequences 00, 01. Both errors in this data set had more higher value levels than Kitti sequences. (a) Trajectory of sequence 00; (b) trajectory of sequence 01; (c) average rotational error on sequence 00; (d) average rotational error on sequence 01; (e) average translational error on sequence 00; (f) average translational error on sequence 01. SAVO: semantic segmentation–aided visual odometry.

ORB_SLAM2. FAST corner points and SURF rich descriptors were computed. The threshold of ratio check was 0.7. The contribution probabilities of segmentation categories were computed as given in section "System Overview," and the probabilities of *Sky*, *Vehicle*, *Pedestrian*, and *Bike* were tuned to zero. For monocular VO performance test only, ORB_SLAM2's loop closure thread was turned off and its feature number was set to 3000. ORB_SLAM2 used default vocabulary. DSO was set to pinhole model with regular FOV lens, and its gamma and vignette configure were ignored.

Kitti odometry data set. The results of proposed method on average translational errors and rotational errors were shown in Tables 1 and 2. ORB_SLAM2 showed best rotation precision on most of the sequences which had little moving cars or other traffic participants. In sequence01, there were several moving cars running on neighboring lanes. Proposed method showed better robustness and precision than other algorithm. The proposed method showed distinct improvements in translation estimation on all Kitti sequences. Most of the monocular VO couldn't recover real scale. This work used a prior knowledge of camera position with fixed height to the ground and assumed the road surface was a plane. These two factors made proposed method benefit a lot from Kitti data set. Figure 5 shows rotational and translational errors relative to travel distance and speed in Kitti sequence00. The proposed method had the lowest errors on both rotation and translation. More details about the evaluation method could be found in VISO.[29,43] Some of the final trajectories are shown in Figure 6.

Beijing Wuhuan data set. As shown in Figure 7, the data set was collected in a sunny day afternoon with normal traffic flow condition. Two sequences in the data set were used for performance experiment. The first one contained 313 pictures and took 936 s. It started at a ring road and stopped at an open expressway and had 7.8 km distance. The second one contained 212 pictures and took 455 s. This one covered 5.9 km distance of the ring road. Tens of moving vehicles could be found in these sequences. This made VISO, DSO, and ORB_SLAM2 fail to estimate a reasonable trajectory. As shown in Figure 8, proposed method could recover the traces with noises. On one hand, semantic segmentation provided a prior probability to sample correct candidate points and image patches. This helped VO try to avoid the disturbance of moving object in the limit FOV. It made possible to use traditional VO in dynamic urban traffic environment. Making full use of fixed camera position information and assumption of road plane not only could provide a scale estimation for monocular camera but also covered the shortage of reduction of available points. On the other hand, limited by the precision of segmentation, the sampling process could not guarantee the sampled pixels were all static. The experimental result showed that dynamic objects are still one of the biggest factors to robustness of VO.

Conclusions

This article proposed a new semantic segmentation–aided VO pipeline. The new method used a deep learning network to segment input image with 12 semantic categories. Then a probabilistic model about categories and reprojection errors was computed for each pixel and used to weighing and sampling pixel candidates for feature-based VO pipeline. And semantic segmentation results also helped to select road plane for alignment-based VO pipeline. These two pipelines brought cost functions of reprojection and intensity errors, respectively, and were combined into a joint optimization in motion estimation process. These helped VO to reduce impacts of moving objects and make full use of motionless pixels by their geometry and physical characters. The experimental results on dynamic urban traffic scene data sets showed that new method provided higher precision and robustness than three state-of-the-art VO solutions. To improve the pipeline real-time performance and study how the VO impact segmenting procedure would be useful works in the future.

Acknowledgements

The authors would like to thank Dr. Chong Xue and Dr. Siyi Zheng for their helpful discussions.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by the National Key Research and Development Program (2016YFB0100903), the Beijing Municipal Science and Technology Commission special major (D171100005017002), and the National Natural Science Foundation of China under grant nos U1664263 and 9142020.

References

1. Nistér D, Naroditsky O, and Bergen J. Visual odometry. In: *Proceedings of the 2004 IEEE computer society conference on computer vision and pattern recognition*, USA, 27 June–2 July 2004, pp. 652–659.
2. Scaramuzza D and Fraundorfer F. Visual odometry: Part I: the first 30 years and fundamentals. *IEEE Robot Autom Mag* 2011; 18(4): 80–92.
3. Fraundorfer F and Scaramuzza D. Visual odometry: Part II: matching, robustness, optimization, and applications. *IEEE Robot Autom Mag* 2012; 19(2): 78–90.
4. Moravec HP. *Obstacle avoidance and navigation in the real world by a seeing robot rover*. California: Stanford University, 1980.
5. Kneip L, Chli M, Siegwart R, et al. Robust real-time visual odometry with a single camera and an IMU. In: *BMVC*, 2011, pp. 1–11.
6. Forster C, Pizzoli M and Scaramuzza D. SVO: fast semi-direct monocular visual odometry. In: *2014 IEEE international conference on robotics and automation (ICRA)*, Hong Kong, 31 May–7 June 2004, pp. 15–22.
7. Lovegrove S, Davison AJ and Ibanez-Guzmán J. Accurate visual odometry from a rear parking camera. In: *2011 IEEE intelligent vehicles symposium (IV)*, Germany, 5–9 June 2011, pp. 788–793. IEEE.
8. Peter Corke SS and Strelow D. Omnidirectional visual odometry for a planetary rover. In: *Proceedings of the intelligent robots and systems*, Japan, 28 September–2 October 2004, pp. 4007–4012.

9. Nistér D, Naroditsky O and Bergen J. Visual odometry for ground vehicle applications. *Field Robot* 2006; 23(1): 3–20.
10. Maimone M, Cheng Y and Matthies L. Two years of visual odometry on the mars exploration rovers. *J Field Robot* 2007; 24(3): 169–186.
11. Li M and Mourikis AI. High-precision, consistent EKF-based visual-inertial odometry. *Int J Robot Res* 2013; 32(6): 690–711.
12. Leutenegger S, Lynen S, Bosse M, et al. Keyframe-based visual-inertial odometry using nonlinear optimization. *Int J Robot Res* 2015; 34(3): 314–334.
13. Strasdat H, Montiel JMM and Davison AJ. Real-time monocular slam: Why filter? In: *2010 IEEE International conference on robotics and automation*, USA, 3–7 May 2010, pp. 2657–2664.
14. Liu H, Yu Y, Sun F, et al. Visual-tactile fusion for object recognition. *IEEE Trans Autom Sci Eng* 2017; 14(2): 996–1008.
15. Liu H, Wu Y, Sun F, et al. Weakly paired multimodal fusion for object recognition. *IEEE Trans Autom Sci Eng* 2017; PP(99): 1–12.
16. Liu H, Sun F, Guo D, et al. Structured output-associated dictionary learning for haptic understanding. *IEEE Trans Syst Man Cybern Syst* 2017; 47(7): 1564–1574.
17. Liu H, Guo D and Sun F. Object recognition using tactile measurements: kernel sparse coding methods. *IEEE Trans Instrum Meas* 2016; 65(3): 656–665.
18. Liu H, Liu Y and Sun F. Robust exemplar extraction using structured sparse coding. *IEEE Trans Neural Netw Learn Syst* 2015; 26(8): 1816–1821.
19. Liu H, Qin J, Sun F, et al. Extreme kernel sparse learning for tactile object recognition. *IEEE Trans Cybern* 2017; PP(99): 1–12.
20. Liu H, Sun F, Fang B, et al. Robotic room-level localization using multiple sets of sonar measurements. *IEEE Trans Instrum Meas* 2017; 66(1): 2–13.
21. Buczko M and Willert V. How to distinguish inliers from outliers in visual odometry for high-speed automotive applications. In: *Proceedings of the 2016 IEEE intelligent vehicles symposium (IV)*, Sweden, 19–22 June 2016, pp. 47–59.
22. Engel J, Koltun V and Cremers D. Direct sparse odometry. *EEE Trans on Pattern Anal and Mac Intell*, July 2016.
23. Badrinarayanan V, Kendall A and Cipolla R. SegNet: a deep convolutional encoder–decoder architecture for image segmentation. *arXiv preprint arXiv:1511.00561*, 2015.
24. Zampogiannis K, Yang Y, Fermüller C, et al. Learning the spatial semantics of manipulation actions through preposition grounding. In: *2015 IEEE international conference on robotics and automation (ICRA)*, USA, 26–30 May 2015, pp. 1389–1396. IEEE.
25. Cadena C, Carlone L, Carrillo H, et al. Past, present, and future of simultaneous localization and mapping: towards the robust-perception age. *IEEE Trans Robot* 2016; 32(6): 1309–1332.
26. Mohanty V, Agrawal S, Datta S, et al. DeepVO: a deep learning approach for monocular visual odometry. *arXiv preprint arXiv: 1611.06069*, 2016.
27. Klein G and Murray D. Parallel tracking and mapping for small AR workspaces. In: *IEEE and ACM international symposium on mixed and augmented reality*, Japan, 13–16 November 2007, pp. 13–16.
28. Civera J, Grasa OG, Davison AJ, et al. 1-Point RANSAC for EKF-based structure from motion. In: *IEEE/RSJ international conference on intelligent robots and systems*, USA, 10–15 October 2009, pp. 3498–3504.
29. Geiger A, Ziegler J and Stiller C. StereoScan: dense 3d reconstruction in real-time. In: *IEEE intelligent vehicles symposium*, Germany, 5–9 June 2011, pp. 963–968.
30. Geiger A, Lenz P and Urtasun R. Are we ready for autonomous driving? The Kitti vision benchmark suite. In: *Conference on computer vision and pattern recognition (CVPR)*, USA, 16–21 June 2012, pp. 3354–3361.
31. Mur-Artal R and Tardós JD. ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras. *arXiv preprint arXiv: 1610.06475*, 2016.
32. Fischler MA and Bolles RC. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun ACM* 1981; 24(6): 381–395.
33. Nistér D. Preemptive RANSAC for live structure and motion estimation. *Mach Vis – Appl* 2003; 16(5): 321–329.
34. Chum O and Matas J. Matching with PROSAC progressive sample consensus. In: *Proceedings of the 2005 Computer Vision and Pattern Recognition (CVPR 2005)*, USA, 20–25 June 2005, pp. 220–226.
35. Civera J, Gálvez-López D, Riazuelo L, et al. Towards semantic SLAM using a monocular camera. In: *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems*, 25–30 September 2011, pp. 1277–1284.
36. Anand A, Koppula HS, Joachims T, et al. Contextually guided semantic labeling and search for 3D point clouds. *Int J of Robot Res* 2011; 32(1): 19–34.
37. Yang DF, Sun FC, Wang SC, et al. Simultaneous estimation of ego-motion and vehicle distance by using a monocular camera. *Sci China Inf Sci* 2014; 57(5): 1–10.
38. Geiger A, Lauer M, Wojek C, et al. 3D traffic scene understanding from movable platforms. *Pattern Anal Mach Intell* 2014; 36(5): 1012–1025.
39. Engel J, Stueckler J and Cremers D. Large-scale direct SLAM with stereo cameras. In: *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems (IROS)*, Germany, 28 September–2 October 2015, pp. 21–29.
40. Yang S, Song Y, Kaess M, et al. Pop-up SLAM: semantic monocular plane SLAM for low-texture environments. In *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems (IROS)*, Korea, 9–14 October 2016, pp. 1222–1229.
41. Kümmerle R, Strasdat H, Grisetti G, et al. g2o: a general framework for graph optimization. In: *IEEE international conference on robotics and automation*, China, 9–13 May 2011, pp. 3607–3613.
42. Long J, Shelhamer E and Darrell T. Fully convolutional networks for semantic segmentation. *arXiv preprint arXiv: 1411.4038*, 2014.
43. Kitt B, Geiger A and Lategahn H. Visual odometry based on stereo image sequences with RANSAC-based outlier rejection scheme. In: *Intelligent vehicles symposium (IV)*, USA, 21–24 June 2010, pp. 486–492.