16-831 Statistical Techniques in Robotics, Fall 2016, Final Report

Visual SLAM in Dynamic Environment using Semantic Segmentation

Guan-Horng Liu (guanhorl), Po-Wei Chou (poweic), Wei-Hsin Chou (weihsinc), Samuel Wang (ssw1), Shu-Kai Lin (shukail)

1. Introduction

Simultaneous localization and mapping (SLAM) in dynamic environment forms a challenging problem which involves pose estimation, scene reconstruction and dynamic objects handling at the same time. Examples of practical application include autonomous cars driving through urban area with other vehicles and pedestrians in the scene or robots navigating through a building with people walking around. With the aid of semantic segmentation and object detection, we are able to identify moving objects in the scene while executing the SLAM task. In this project, we formulate the problem over the integration of semantic segmentation and monocular-based visual SLAM systems and investigate the capability on performance enhancement in dynamic environment.

In order to separate the static background and dynamic objects with respect to the camera along image sequences, we take the advantage of image semantic segmentation and object detection bounding box. The instance-level segmentation is able to provide better understanding of the scene. The prior knowledge over moving objects is expected to improve the performance in localization and mapping tasks. In our work, we use a deep neural network to preprocess the sequence and yield the segmented images for the SLAM systems.

Two state-of-the-art visual-SLAM algorithms are studied in this project. ORB-SLAM[2] is a feature-based monocular system that can estimate the camera trajectory while reconstructing the environment with sparse feature points. On the other hand, LSD-SLAM[3] uses the direct method to achieve localization and large scale mapping by directly optimizing over pixels of the full image. With respect to the integration of semantic segmentation, the problem formulations can be divided into two kinds: (1) Ordinary SLAM problem with exclusion of feature points or image patches at the segments of identified dynamic targets, (2) SLAM and moving objects tracking at the same time[1], which requires a generalized formulation and optimization over the joint probability. In this project, we present ideas and approaches to both problems with respect to different visual SLAM algorithms, and focus on the investigation of the first problem.

In our work, we use the Cityscapes datasets[4] to train our network and generate the segmented image data with bounding box annotations. We evaluated our proposed method by testing ORB-SLAM and LSD-SLAM algorithms on the same datasets, both with and without segmentation in the loop. We compared all the results with the ground truth data and conducted quantitative evaluation on the localization performance.

2. Visual SLAM Algorithms Overview

2.1 ORB-SLAM

ORB-SLAM is an indirect sparse feature-based SLAM algorithm. The indirect method preprocesses raw sensor measurements to generate an intermediate representation and then optimizes geometric error. Standard approach involves extraction and matching a sparse set of keypoints (ORB in this case). The framework of ORB-SLAM is shown in Fig. 1(a).

2.2 LSD-SLAM (Large-Scale Direct Monocular)

In contrast to sparse method, the direct method skips the preprocessing step and directly uses the pixel intensity values to optimize photometric error. The major advantage of direct method is that it uses the whole image information instead of several sparse feature points. Thus, it is more robust in low texture environment.

LSD-SLAM uses direct method coupled with filtering-based estimation of semi-dense depth maps. It can not only track the motion of the camera but also build a consistent large-scale map of the environment. The framework of LSD-SLAM is shown in Fig. 1(b).

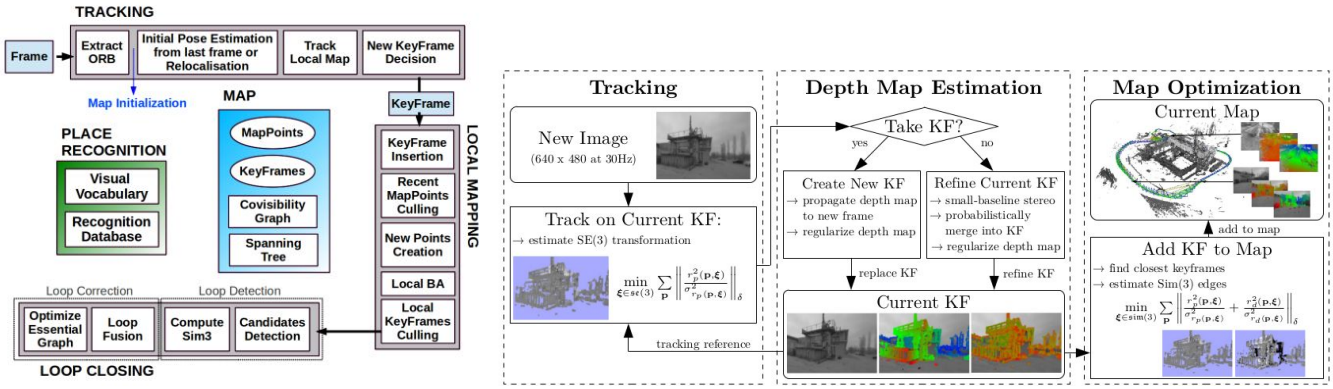

Fig. 1: (a) ORB-SLAM system overview[2] (b) LSD-SLAM system overview[3]

2.3 Comparison

Here we briefly summarize the difference between ORB-SLAM and LSD-SLAM.

Table I. Comparison between ORB-SLAM and LSD-SLAM.

	ORB-SLAM	**LSD-SLAM**
Optimization	Minimize geometric error	Minimize photometric error
Points	~1,000	~100,000
Strength	Robust in dynamic scene	Robust in low texture area
Weakness	Low texture area;Many (or big) moving objects moving slowly.	Dynamic scene

3. Proposed Method

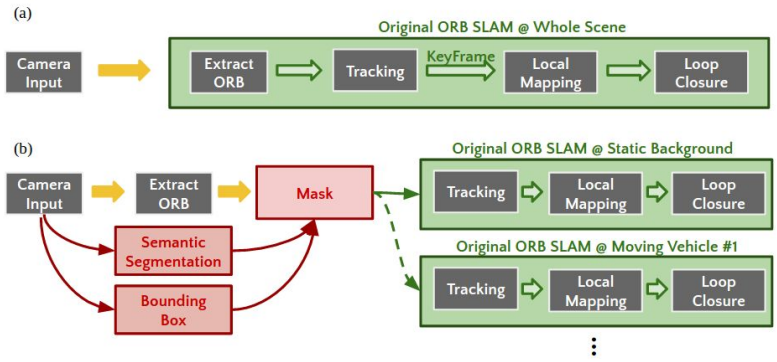

Fig. 2: Modified framework of ORB-SLAM augmented with semantic segmentation

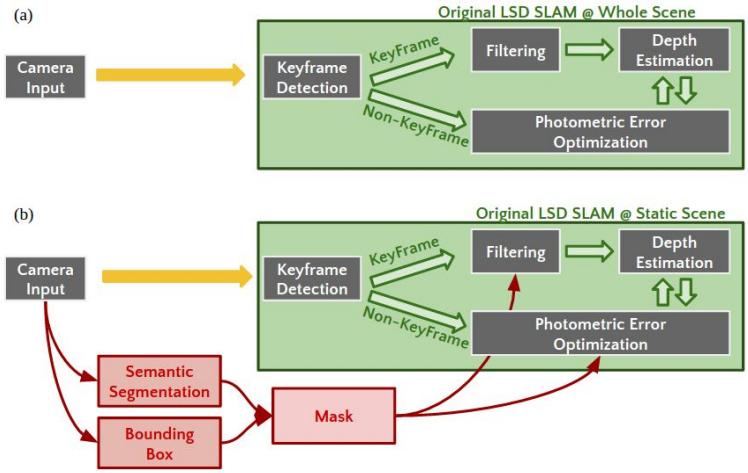

Fig. 3: Modified framework of LSD-SLAM augmented with semantic segmentation

3.1 System Workflow Overview

In Addition to the original SLAM framework, our workflow runs a separable instance-level semantic segmentation to identify slow-moving object (e.g. cars, bicycles). The semantic information is then embedded into the systems as a prior for rejecting outliers. Since the original design of two SLAM systems naturally differs with each other, the modules affected by the segmentation are also different.

For ORB-SLAM, instead of using all the extracted ORB features, the features are separated from static and dynamic objects based on semantic segmentation; then sub-SLAM systems can be initialized w.r.t. static background as well as each individual moving object in order to keep track of them. Moreover, with sufficient information of the motion model, we could potentially increase the accuracy of visual odometry by incorporating these information. The modified system for ORB-SLAM is shown in Fig. 2.

For LSD-SLAM, the main goal is to the improve the performance of visual odometry by filtering out the moving objects. Figure 3 shows the original and the augmented SLAM systems. The filtering and the photometric error optimization subsystems are modified to take the segmentation results into account. Before depth estimation in each keyframe, the moving objects are treated as potential noise and removed from the filtering subsystem. During the pose tracking, the pixel values on the moving objects are not considered into the least-squares problem either.

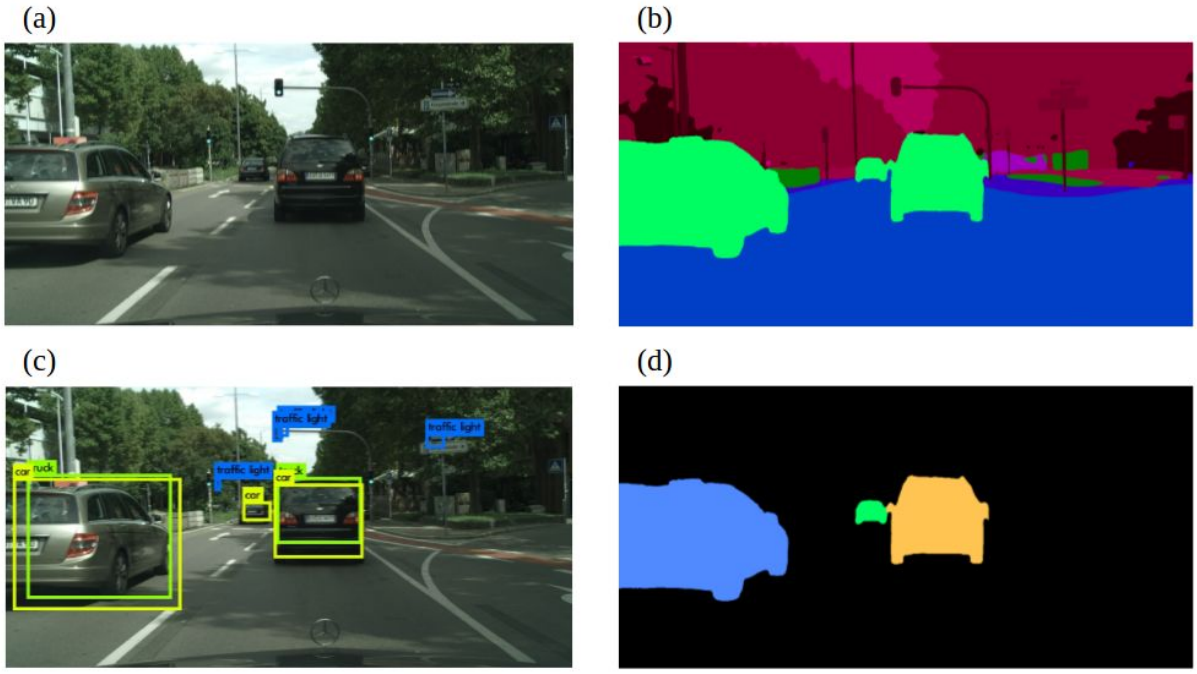

Fig.4: Instance-level semantic segmentation. (a) Raw image from camera, (b) semantic segmentation, (c) object detection with bounding box and (d) semantic segmented Instances.

3.2 Instance-Level Semantic Segmentation using Convolutional Neural Network

For instance-level semantic segmentation, we use two convolutional neural networks: one for semantic segmentation and one for bounding boxes generation. These networks were trained separately. During inference, bounding boxes with high probabilities were used to crop out the segmentation result to get an instance-level semantic segmentation.

For semantic segmentation, we use a 18-layer convolutional neural network similar to VGG-16 but with additional dilated convolutional layers to increase the width of receptive field. We pre-trained our network on ImageNet, and then fine-tuned our model on 2925 images of size 2048 x 1024 pixels from Cityscapes training set. Twenty different semantics classes such as car, pedestrians, tree, road, sidewalks, etc. were used. (see Fig. 4). Though we only consider cars and pedestrians as dynamic objects in our problem formulation, it's generally better if more classes were used. This is because (1) the benefit of structured learning and (2) for each image, more bits of label information were used, hence we should expect better performance.

For the bounding box generation, we use an off-the-shelf state-of-the-art real-time object detection toolkit called YOLO [5][6][7] to help us extracting (cropping) instance-level semantic segmentation. In [6], they treated the object detection as a regression problem to spatially separated bounding boxes with associated class probabilities in an unified architecture. At runtime, a low probabilities threshold were used to get a high recall rate. This resulting non-overlapping bounding boxes were then used to crop out the semantic segmentation result.

We trained the segmentation network with L2-norm regularization to enforce the sparsity of weight parameters and then later pruned the model heavily using L2 criterion. This gave us about 15 ms

inference time per frame (half-resolution were used) on NVIDIA Pascal Titan X GPU, which matches the speed of object detection and SLAM in the later pipeline.

4. Results

4.1 Ground Truth Data and Evaluation

To evaluate localization performance of the proposed methods, we prepared the ground truth data with the frame-based vehicle information provided by the wheel sensor and GPS measurements. The Cityscape dataset comes with the metadata of wheel speed, vehicle yaw rate and GPS coordinates at each image frame. The vehicle odometry from dead-reckoning on the wheel speed and yaw rate can provide continuous trajectory for local pose evaluation. However, vehicle odometry tends to drift away from the absolute positions over time due to accumulation of incremental errors, which makes it unsuitable for long term localization source with global consistency.

In order to provide the source for absolute pose, we fused the two sensor data with an extended Kalman filter (EKF) as shown in Fig. 5. The estimated state is the 2D global pose of the camera at each image frame, whose position and heading are with respect to the UTM (Universal Transverse Mercator) grid. The EKF is in the forced formulation which takes the odometry data (speed and yaw rate) as the input in the prediction step and propagates the state estimate with a constant velocity motion model. In the update step, the pose estimate is corrected by the GPS measurement (global position and heading). The result of sensor fusion is shown in Fig. 6 together with the vehicle odometry and GPS poses, where the arrows represent the heading direction. For the sake of clarity, the GPS poses and the filtered poses are transformed onto the local ENU (East-North-Up) frame by moving the origin of the UTM frame to the initial GPS position, which can resolve the issue of different scales when comparing these three pose sources.

Throughout the evaluation of localization performance for different SLAM systems and methods, we used the vehicle odometry as the baseline for comparison. For other datasets with loop-closures, the filtered global pose will be useful for comparison between SLAM systems.

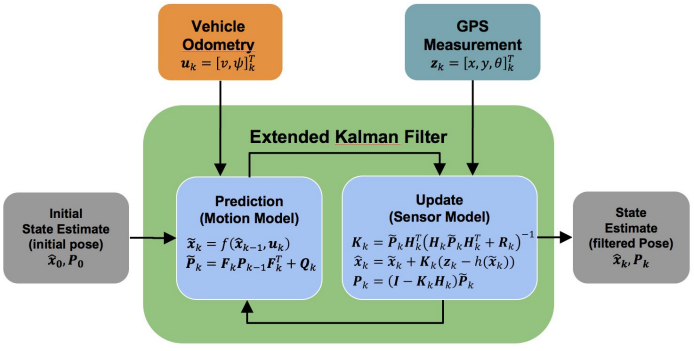

Fig.5: Extended Kalman filter for global pose estimation.

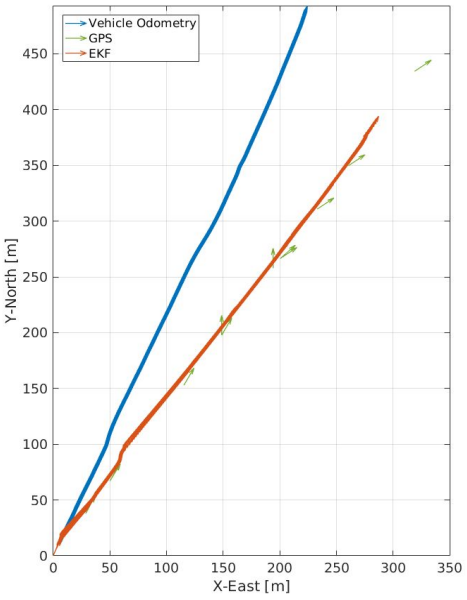

Fig. 6: Ground truth data comparison

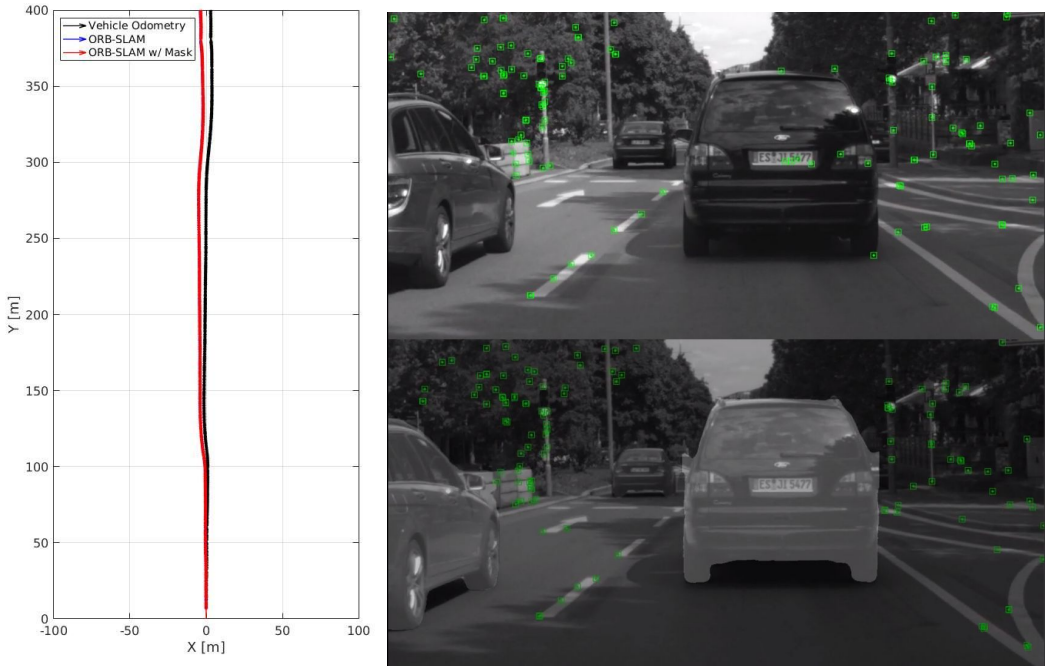

Fig. 7: (Left) Result in the form of 2D odometry. (Right) Screenshot from video stream. The upper figure is the original method, while the lower is our proposed method. Segmentations are shown in transparent. Full video is available here: https://www.youtube.com/watch?v=XZefR0smM3U

4.2 Discussion on ORB-SLAM

The visual odometry results for ORB-SLAM with and without segmentation augmented are reported in Fig. 7. Though the difference is nearly not observable, we found out there existed specific scenario where without the prior from segmentation, the original ORB-SLAM frequently failed and lost track. Such cases happen specifically when there other vehicles passing through at the same time when our vehicle was attempting to change the lane. Since most of the scene in this case were occupied by non-static objects, segmentation can be served as a good prior to robustly filter out key-points on dynamic objects, which will significantly degrade the performance of original visual odometry.

At the same time, we also tried to implement the sub-slam system in order to track the motion of dynamic object simultaneously. To fulfill multi-SLAM in the proposed method, several modifications were done to the original system, including adjusting the initial threshold for FAST detection since the moving objects are usually in low contrast texture. Due to implementational issues and the heavily parameter-dependant nature of SLAM system, we did not report experimental result here.

4.3 Discussion on LSD-SLAM

The experimental results of LSD-SLAM are shown in Fig. 8. In Fig. 8, the right side are the semi-dense maps constructed by LSD-SLAM with and without mask; the left side is comparison of visual odometry. In semi-dense maps, both LSD SLAM with and without mask generate reasonable trajectory. However, the difference between two of than is not clear. Thus , we evaluate the performance by comparing their odometry with the ground truth provided by the dataset.

In the odometry part, as you can see, both LSD-SLAM with and without mask drift when the vehicle changes lane at first time, because there is another vehicle comes through at the same time. It shows that

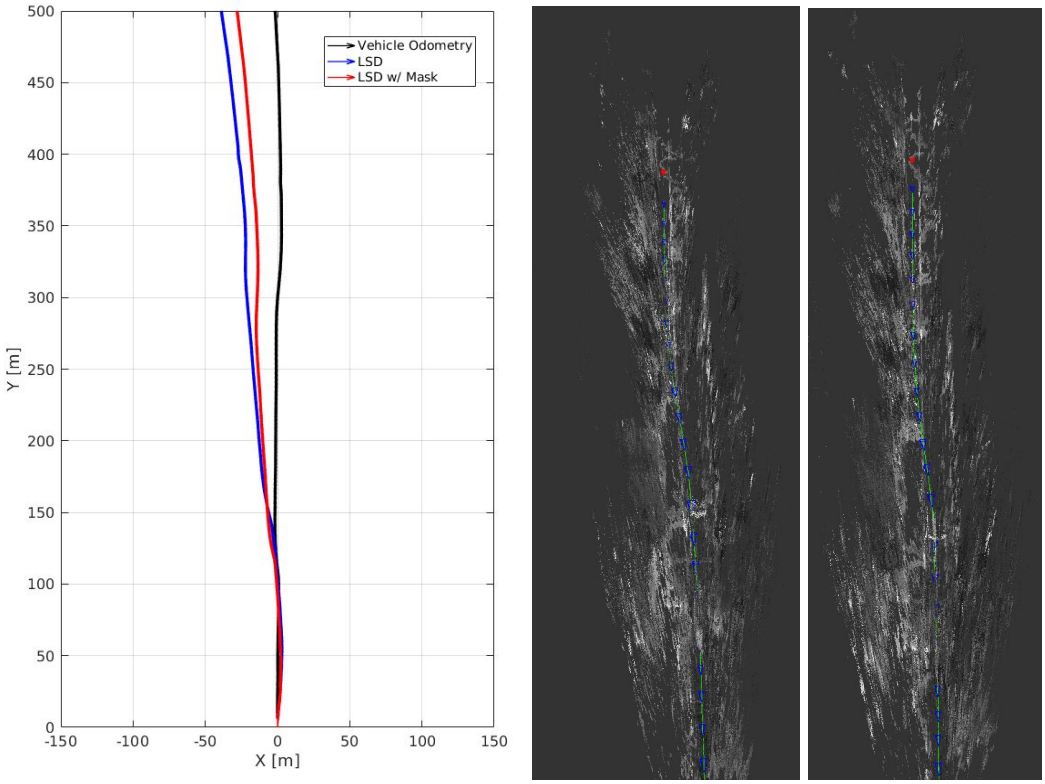

Fig. 8: Result of proposed LSD-SLAM shown in the form of 2D odometry and semi-dense maps.

the LSD-SLAM is vulnerable when the dynamic object is accounting for a large proportion of the image. The blue line in the figure is LSD-SLAM with mask. Since the dynamic object is filtered out, the odometry result is slightly better than the LSD-SLAM without mask.

4.4 Comparison

The experimental results are listed in Table II by comparing the odometry with ground truth data for different methods. There are some interesting points worthy of discussion. First, compare LSD-SLAM with and without mask, both horizontal position and orientation RMSE of LSD-SLAM with mask are less than the one without mask. It shows that the semantic segmentation does improve LSD-SLAM in dynamic scene. Second, by comparing ORB-SLAM with LSD-SLAM, RMSE of ORB-SLAM is significant less than LSD-SLAM even without the mask. This result matches the comparison we listed in Table I, ORB-SLAM is generally more robust than LSD-SLAM in dynamic scene, even without aid from segmentation.

Table II. Evaluate proposed methods with ground truth odometry

Datasets	SLAM Algorithms	Horizontal Position RMSE [m]	Orientation RMSE [deg]
Cityscape *stuttgart_01*	LSD	34.20	3.9923
	LSD w/ mask	17.76	2.6929
	ORB	3.865	0.3192
	ORB w/ mask	4.788	0.2410

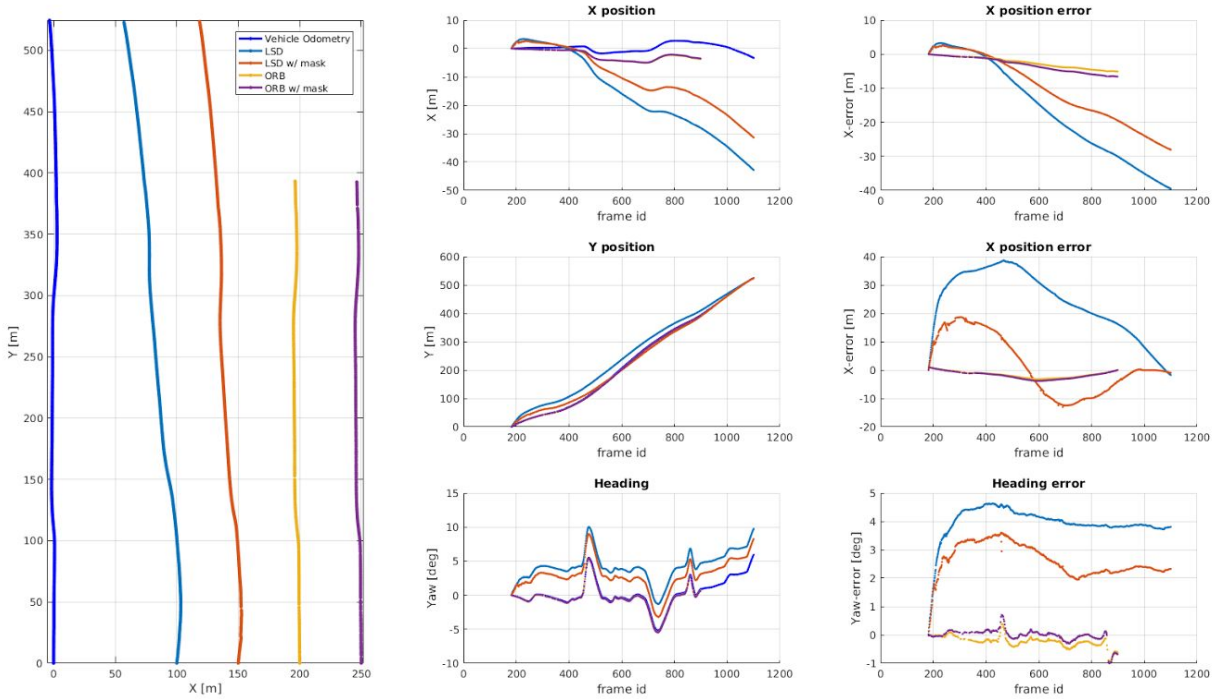

Fig. 9: Odometry comparison of SLAM systems with and without segmentation information.

References

[1] Wang, Chieh-Chih, et al. "Simultaneous localization, mapping and moving object tracking." *The International Journal of Robotics Research* 26.9 (2007): 889-916.

[2] Mur-Artal, Raul, J. M. M. Montiel, and Juan D. Tardós. "Orb-slam: a versatile and accurate monocular slam system." *IEEE Transactions on Robotics* 31.5 (2015): 1147-1163.

[3] Engel, Jakob, Thomas Schöps, and Daniel Cremers. "LSD-SLAM: Large-scale direct monocular SLAM." *European Conference on Computer Vision*. Springer International Publishing, 2014.

[4] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The Cityscapes Dataset for Semantic Urban Scene Understanding," in Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[5] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, “You Only Look Once: Unified, Real-Time Object Detection”

[6] Joseph Redmon, Anelia Angelova, “Real-Time Grasp Detection Using Convolutional Neural Networks”

[7] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, Ali Farhadi, “XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks”