Landmark filtering techniques for semantic mapping in urban environments

dr. J.B.P. Vuurens, V.G Ramlochan-Tewarie, N. Evangelou, J. van Hoven, B. van Elburg, C.N.A. Ros, S. de Lanooi, I. Isaku, D.G. Doran, K.I. van Veen

Abstract **- In this paper we present various optimizations within the Visual Simultaneous Localization And Mapping (SLAM) algorithm for use in autonomous driving cars. We observed potential weaknesses in existing SLAM algorithms and propose simple mechanisms for landmark selection, filtering and matching to improve visual SLAM. The results of comparing our approach with other SLAM algorithms shows that, by using filtering techniques the accuracy of our trajectory can be improved.**

Introduction

In today's world, autonomous vehicles are being developed by a variety of companies and educational institutions. One of the main reasons for the existence of autonomous vehicles, is that it wants to reduce the amount of fatal car accidents. Companies like Google, Tesla and Toyota have made an effort into improving their own autonomous vehicles in these past few years. These vehicles require information of the environment to safely navigate to their point of destination. Generated sensor data is mandatory to achieving autonomous driving.

The Urbinn Learning Lab aims to develop vehicles that can drive autonomously in urban areas. It's goal is to build a solution for last-mile transportation and for driving around tourists in the city of Delft, The Netherlands. For this project, we strive to accurately map the city, determine the position of the car, nearby obstacles and traffic in real-time. This paper reports on our progress to accurately estimate the camera's odometry and object localization. We reveal potential weaknesses in existing solutions for Visual Simultaneous Localization And Mapping (SLAM), and propose simple mechanisms for landmark selection, filtering and matching to improve visual SLAM.

Furthermore, by identifying objects through the use of an object detection system we are able to create a semantic map (Nüchter & Hertzberg, 2008) which can be used to navigate urban environments. Based on the motion of dynamic objects within the semantic map we create the ability for the autonomous vehicle to make decisions based on the velocity and direction.

In the following sections we discuss related work, our own approach, the executed experiments, the results of the experiments and the conclusion.

I. Related work

In this section we discuss related work on sparse visual methods for SLAM. Our discussion, as well as the evaluation is focused only on SLAM approaches.

SLAM

SLAM stands for Simultaneous Localization and Mapping and is a family of algorithms to simultaneously estimate the camera position and the position of observed landmarks with an autonomous mobile robot. Different kind of sensors can utilize the SLAM algorithm to be used as input data for this process, e.g. LIDAR (Zhang & Singh, 2015), stereo camera (Mur-Artal, Tardós, 2017), RGB-D (Kerl, Sturm & Cremers 2013). This paper focuses on Sparse Visual Stereo SLAM, as described in (Strasdat, Montiel & Davison, 2010).

There are different types of SLAM algorithms that have their own approach in using sensor data. LSD-SLAM and SVO-SLAM make use of a direct SLAM method called monocular SLAM. ORB-SLAM2 the stereo implementation of SLAM. One of the shortcomings of SLAM algorithms is that they generate many landmarks, close loops on trajectory only, which spreads errors across the trajectory instead of fixing the true invalid pose estimations. Therefore, SLAM algorithms make use of a technique called Bundle Adjustment (BA).

LSD-SLAM

The LSD method is a SLAM algorithm is based on a keyframe localization and mapping approach. LSD-SLAM uses monocular and stereo images to track the motion of the camera and allows to build consistent, large-scale maps of the direct environment. LSD-SLAM uses direct image alignment coupled with filtering-based estimation of semi-dense depth maps. The global map is represented as a pose graph consisting of keyframes as vertices with 3D similarity transforms as edges, that incorporate changing scale of the environment and allowing to detect and correct accumulated drift. LSD-SLAM is not suitable for our research, since the open source project is not longer supported by the developers. Furthermore, LSD-SLAM only supports one specific input format for images which limits the use of different types of sensor data.

SVO-SLAM

SVO-SLAM (Semi-direct Visual Odometry) is a semi-direct monocular visual odometry algorithm. This operates directly on pixel intensities, which results in subpixel precision at high frame-rates.

ORB-SLAM2

The ORB-SLAM2 algorithm can be used for monocular, stereo and RGB-D cameras and results in a sparse 3D reconstruction. This algorithm utilizes loop detection and relocalization to establish the position of the camera in real time.

Object Detection

The real-time object detection system You Only Look Once (YOLO) (Redmon, Divvala, Girshick & Farhadi 2016) learns by using labeled examples (images) to predict, in a single pass, and draws the bounding box around an object. A single neural network is used which predicts the bounding boxes and class probabilities directly from full images in one evaluation.

Semantic Mapping

According to Nüchter & Hertzberg (Nüchter & Hertzberg, 2008), a semantic map contains assignments of mapped features to entities of known classes, as well as spatial information about the environment. Ideally, a semantic map should allow the autonomous driving system to reason about the environment and make appropriate decisions. When the semantic map is being built, the entities are bound to pose, velocity and behavior which are described by attributes.

II. Design

We present approaches for landmark selection, filtering and matching used within SLAM algorithms. We also present a technique to use object detection and classification to optimize landmark selection. The source code of our algorithm is publicly available for further research purposes.

Landmark selection

Sparse visual SLAM methods rely on selecting landmarks that can easily be identified across images in order to estimate the distance and camera poses. For efficiency, we prefer to compute only a select number of pixels, and thus rely on a selection mechanism that identifies landmarks that confidently match the same landmark in another image. Because horizontal displacement is used for depth estimation, we choose to select landmarks that lie on vertical edges by using the Prewitt operator (Chaple, Daruwala & Gofane, 2015). In order to confidently match landmarks in other images we additionally use two filters that identify the top and bottom of these vertical edges. Corner detectors have been used in previous work (e.g. ORB2) and we will show that this variant is effective and efficient.

Landmark filtering

We use image patches around landmarks to identify the same landmark in other images. In our experiments we have identified that a patch with low contrast matching a patch in another image has an increasing chance of being a false positive. Therefore we propose to filter out low contrast patches to increase the likelihood that an extracted patch is correctly matched to a patch in another image. The contrast of a patch is estimated by its standard deviation across pixels.

Landmark matching

To account differences in contrast between images we propose to use the L2-norm between two patches divided by the L2-norm with the image median and find that this does allow straightforward filtering of false positives with a simple threshold.

Stereo matching

To estimate the distance of a specific landmark in a left-hand-camera frame, we match the landmark in the corresponding right-hand-camera frame. Given that the images are rectified, we search for the corresponding position in the right frame by trying all positions to the left on the same horizontal line. Similarity between two landmarks is estimated using the L1-norm between 17x17 pixel image patches around each position. Image patch matching can result in false positives, especially when there are recurring patterns like windows in a facade. We find that these cases are effectively reduced by filtering out landmarks for which the similarity to the best matching position does not exceed that of the second-best position by some threshold. If a landmark is kept, we use subpixel estimation in the right-hand frame to further refine the disparity and use this to estimate the distance.

Object detection and classification

To detect objects and estimate their location, we use an existing object detector. The object detector used in our experiments returns bounding boxes and object classes of all detected objects in an image. Initially, we assume that a landmark inside a bounding box belongs to an object of the identified class, and after a filtering and clustering step we can use these landmarks' coordinates to estimate the precise location of static objects within the map.

Pose estimation

Our system uses motion-only bundle adjustment (Mur-Artal, Tardós, 2017) to determine the camera position of a subsequent frame based on the covisible landmarks between frames.

III. Experiment

To analyse our proposed design we used the KITTI-dataset. The KITTI-dataset consists of sensor recordings taken when driving a car in Karlsruhe, a dense village area in Germany, including stereo images from a front-facing camera that is mounted on top of the car. This dataset consists of recordings, with exception to sequence 1, which takes place on the highway. The KITTI dataset is arguably the most often used dataset to evaluate the odometry for SLAM methods, making this dataset suitable to analyze problems in localization.

Evaluation setup

We have evaluated our setup by comparing our results with the ground truth of the KITTI-dataset. This is accomplished by comparing the trajectory Fig. 1 (right) from a top-down perspective with the ground truth Fig. 1 (left). With this approach we can evaluate the accuracy of the results created by our setup.

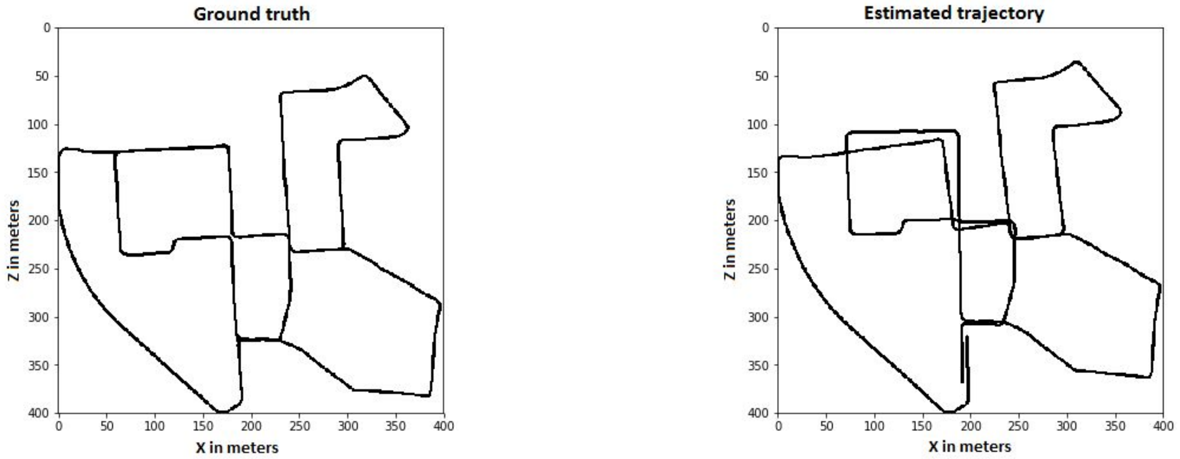

Fig 1. KITTI sequence 0, ground-truth trajectory (Left). Estimated trajectory (Right).

The results show that there is a slight error in our trajectory. To correct this error we propose the use of full bundle adjustment which will apply loop closure.

Loop closure

As described in (Briskin, Geva, Rivlin & Rotstein, 2017), loop closure in existing visual SLAM methods depend on recognizing the position it has passed earlier in combination with the landmarks it has observed. A similar pose earlier in the trajectory cannot recognize the locations of opposite directions which were previously passed. This information is used to compute a 3D structure of the environment together with the relative motion of the recording.

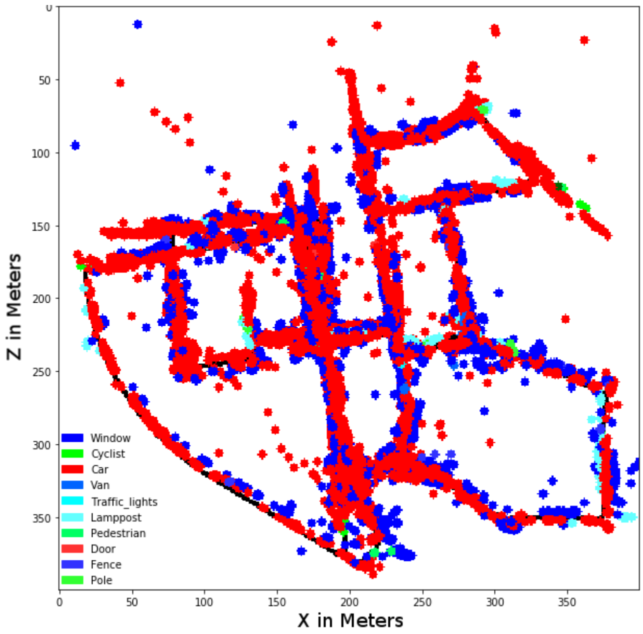

Fig. 2. Early version of semantic map with detected objects.

Semantic map

In combination with an object detection system we are able to locate objects within a 2D space. In Fig. 2 we have plotted the trajectory and all the detected objects within the frames. In future work we can use this information to classify objects as dynamic or static. By filtering out dynamic objects we establish a semantic map with landmarks that can be used for optimization in the relocalization algorithm.

Reprojection

In our experiments, it appeared that bundle adjustment is quite susceptible to mismatched landmarks or the use of landmarks that are moving, even when used with a Huber kernel. In this paper, we propose to improve pose estimation not by full bundle adjustment, but aim to filter out landmarks that do not correspond to the correct pose estimation. By reprojecting every landmark using the estimated pose we can identify the landmarks that were responsible for that pose. We iteratively rerun motion-only bundle adjustment with all landmarks but those responsible for the previous pose estimations. Finally, the lowest L2-norm to the pose of the previous frame is chosen.

IV. Results

To evaluate the results of our algorithm, we make use of the KITTI-devkit[1]. This tool allows us to compare the ground truth of sequence 00 through 10 of the KITTI-dataset with the results of our own trajectory. For example, Fig. 3 shows a comparison of our trajectory (blue) with the ground truth (red).

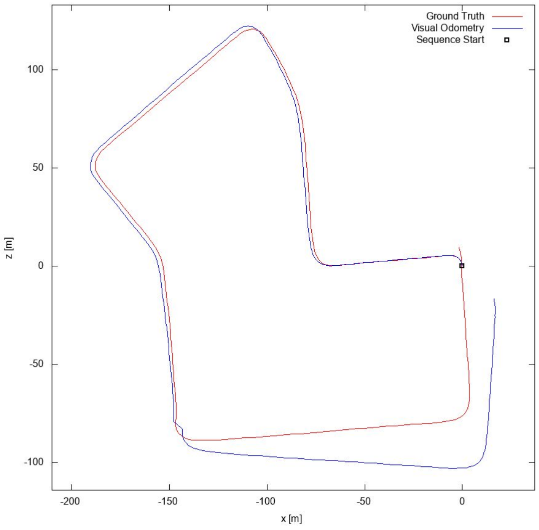

Fig. 3. Evaluation of KITTI sequence 07

To evaluate the trajectory the KITTI-devkit provides us with a translation error and a rotation error. The translation error is the deviation (indicated as percentages) compared to the ground truth. Whereas the rotation error is the deviation in degrees per meter.

SLAM algorithm	**Translation error**	**Rotation error**
Our algorithm	3.99%	0.00319 [deg/m]
ORB-SLAM2	1.15%	0.0027 [deg/m]
S-LSD-SLAM	1.20%	0.0033 [deg/m]

Fig. 4. Translation error and Rotation error table

V. Discussion

This discussion will argue the improvements we have made during the development of our algorithm. Furthermore, we will highlight findings which require further research.

Improvements

During our research we found that some improvements made noticeable differences, e.g. eliminating invalid frames (Fig. 3). This optimizes the trajectory without the need of applying Bundle Adjustment. In Fig. 5 (left) we show the trajectory of sequence 00 of the

[1] *KITTI odometry development kit (http://kitti.is.tue.mpg.de/kitti/devkit_odometry.zip)*

KITTI-dataset without the elimination of the invalid frames. On Fig. 5 (right) these invalid frames have been removed and shows a resemblance to the ground truth (Fig. 1 left).

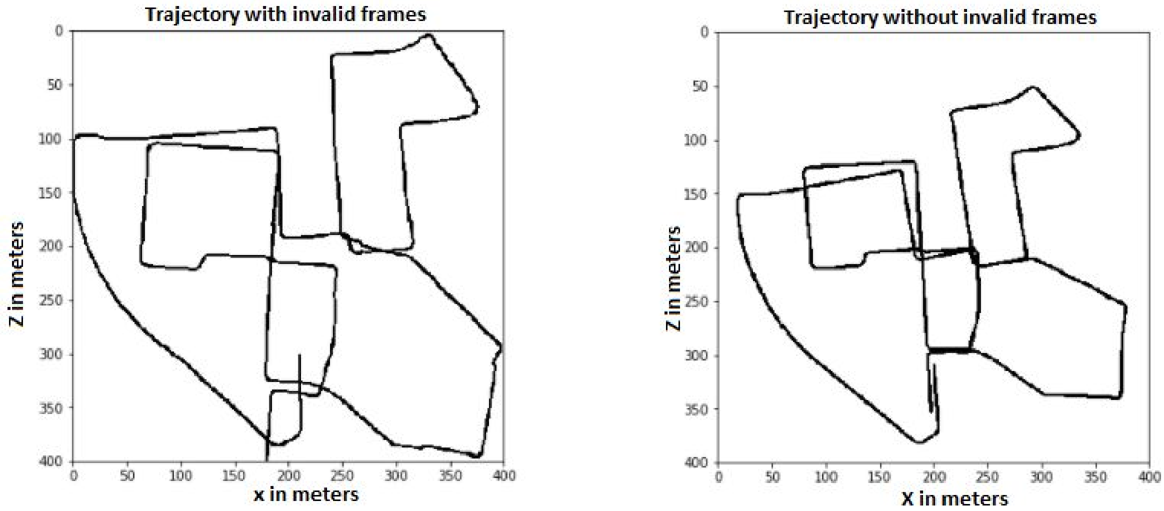

Fig. 5. Trajectory with invalid frames (Left). Trajectory without invalid frames (Right).

As the results from Fig. 5 and the KITTI-devkit[1] shows, we are under the impression that by applying bundle adjustment the translation error of 3.99% will decrease. This will bring our algorithm in range of other state-of-the-art algorithms.

Findings

In our experiments, we found several situations in which errors in estimated odometry occur: This requires further research.

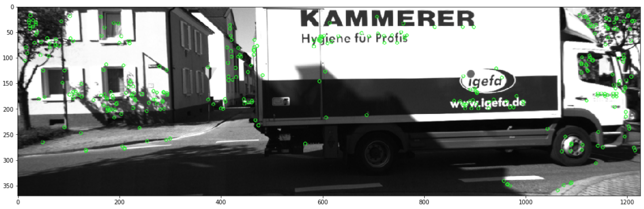

Fig. 6. Invalid frame

In Fig. 6, a truck crosses the intersection which in some frames causes the algorithm to use the truck as a static landmark and thus falsely predicts lateral movement of the camera. This image is considered an invalid frame.

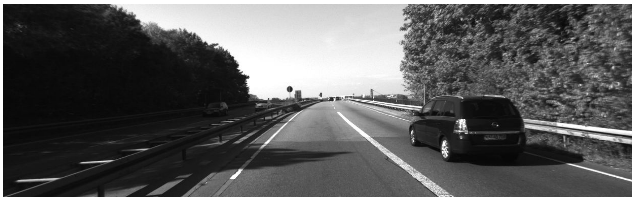

Fig. 7. Highway

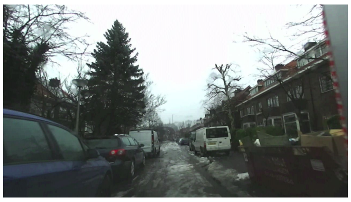

Fig. 8. Bad weather conditions

In Fig. 7, we are passing a car on the highway. The landmarks that are on the car give a false indication of our movement. Similarly, many other landmarks are on perspective lines (e.g. guardrails, lane markers) making it more difficult to correctly identify that landmarks between images are truly the same.

We have created our own dataset with recordings of the city Delft in The Netherlands. After processing this dataset in our algorithm, the result was that there were not sufficient landmarks to create a frame. We suspect the cause of this was due to weather conditions (Fig. 8) at the time of recording. This creates the lack of contrast in the frames. However, further experiments are needed to confirm this.

VI. Conclusion

In conclusion we have created an algorithm that utilizes vertical edge detection and stereo matching to create a semantic map. Based on the results of the KITTI-devkit, our algorithm scored a translation error percentage of 3.99%. This is below average of other state-of-the-art SLAM algorithms.

During our experiments we encountered errors which influenced the output of the trajectory. By analysing the landmarks, we detected that moving objects, e.g. trucks, see Fig. 6, which take up a significant portion of the frame, causes the algorithm to inaccurately estimate the trajectory. By identifying similar situations in a frame, e.g. Fig. 7 and using the described filtering techniques, we are able to improve the quality of the trajectory.

References

J. Zhang and S. Singh, "Visual-lidar odometry and mapping: low-drift, robust, and fast", 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, 2015, pp. 2174-2181. http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=7139486

R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," in IEEE Transactions on Robotics, vol. 33, no. 5, pp. 1255-1262, Oct. 2017. http://ieeexplore.ieee.org/document/7946260/

C. Kerl, J. Sturm and D. Cremers, "Dense visual SLAM for RGB-D cameras," 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, 2013, pp. 2100-2106. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6696650

H. Strasdat, J. M. M. Montiel and A. J. Davison, "Real-time monocular SLAM: Why filter?," 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, 2010, pp. 2657-2664. http://ieeexplore.ieee.org/document/5509636/

A. Nüchter, J. Hertzberg "Towards semantic maps for mobile robots", Robotics and Autonomous Systems 56 915–926, Elsevier, 2018. http://kos.informatik.uni-osnabrueck.de/download/ras2008.pdf

J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 779-788. http://ieeexplore.ieee.org/document/7780460/

G. Briskin, A. Geva, E. Rivlin and H. Rotstein, "Estimating Pose and Motion Using Bundle Adjustment and Digital Elevation Model Constraints," in IEEE Transactions on Aerospace and Electronic Systems, vol. 53, no. 4, pp. 1614-1624, Aug. 2017. http://ieeexplore.ieee.org/document/7851083/

G. N. Chaple, R. D. Daruwala and M. S. Gofane, "Comparisions of Robert, Prewitt, Sobel operator based edge detection methods for real time uses on FPGA," *2015 International Conference on Technologies for Sustainable Development (ICTSD)*, Mumbai, 2015, pp. 1-4. http://ieeexplore.ieee.org/document/7095920/