Online Ego-Motion Estimation Using LiDAR Point Cloud Clustering

Oded Fischler
Innoviz Technologies
odedf@innoviz-tech.com

Raja Giryes
Innoviz Technologies
Tel Aviv University
raja.giryes@innoviz-tech.com

Abstract

Reliable ego-vehicle motion measurement is a key component to guarantee stable and safe autonomous driving algorithms. Compared with visual-based localization, LiDAR has higher accuracy and better stability. In this work, we propose a novel technique for ego-motion estimation based on approximating the motion between clusters in pairs of consecutive point clouds. This, in turn, allows a good estimation of the car's ego-motion and to an automatic detection of moving objects, where both tasks are very important in autonomous driving. Our strategy possess the following two important properties: (i) it is computationally efficient and thus applicable for real-time system; (ii) it does not require expensive annotated data as it is unsupervised.

1. Introduction

Estimating 3D position and orientation of a mobile platform is a fundamental problem in 3D computer vision, as it provides important navigation information for autonomous driving. Mobile platforms usually collect real-time information from the environment using on-board sensors such as LiDAR, Inertial Measurement Units (IMU), or cameras, to estimate their motions. LiDAR can obtain robust features of different environments as it is not sensitive to lighting conditions, and it also acquires more accurate distance information than 2D cameras. Thus, developing an accurate and robust real-time ego-motion estimation is desirable and often a crucial part in the perception layer.

Classic point cloud based global registration methods use pose estimation strategies such as Normal Distributions Transform (NDT) [2], Iterative Closest Point (ICP) [1, 8] and LOAM [10], which often fail in real driving scenes due to significant portion of moving objects within the field-of-view of the sensor and large variation of points densities at different distances in the scene. Other approaches, which are based on supervised learning and rely on 3D point cloud data, e.g., [6, 7, 9], suffer from high computational complexity (or high latency) and from the need for expensive accurate ground-truth data.

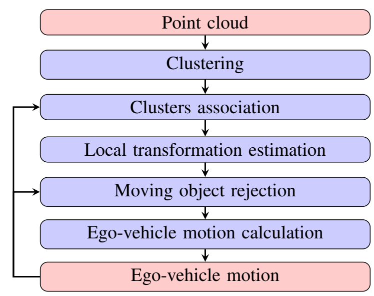

Figure 1. Ego-vehicle motion estimation pipeline.

The main contributions of this work are: i) A novel scan-to-scan ego-motion estimation, which simultaneously provide the vehicle ego-motion and the movement model of the moving entities within the view of the LiDAR (our method can be considered in some sense as calculating a 3D rigid-objects scene flow); and ii) Our approach is unsupervised and does not rely on expensive ground-truth data collection.

2. Our Approach

The main idea behind our approach is calculating the motions in a given scene by analyzing the motion between matched clusters in a pair of consecutive LiDAR scans. Fig. 1 summarizes our scheme. We first apply a method for ground removal and clustering, which significantly reduces the point-cloud data into a more compact and efficient representation. Then, clusters between two scans are associated to each other based on their descriptors. The matching between clusters is used both for detecting and removing moving objects in the scene, and for the ego-motion estimation. Once these are calculated, the moving object rejection may be refined and the whole process may be iterated. Next, we give more details on each stage.

Clustering. To segment the point-cloud into rigid entities, the method in [3] is used. Based on the angle between the LiDAR and scanned points, the ground is removed and a (iterative) clustering is performed to the remaining points.

Clusters Association. This stage matches corresponding clusters between the two consecutive frames. For each cluster, a geometrical descriptor consisting of the first and second order moments of the points in the cluster is calculated. The matching is done using a Nearest Neighbors (NN) algorithm with ℓ_2-norm between the clusters descriptors. Moreover, assuming that the ego-motion cannot change drastically from the previous frame, we are using the previous ego-motion result to update the point cloud before calculating the descriptors.

Local Transformation per Cluster. This part estimates the transformation $T_{BA} \in \mathrm{SE}(3)$ between p_A and p_B (the point clouds of frames A and B), such that $p_B = T_{BA} p_A$. The registration of each pair is done using (per point) local features correspondence combined with gradient descent. Clearly, a representation that exploits the surface spatial smoothness could be more robust and compact. For example, straight surfaces such as traffic signs or sign gantries could be describe with only three parameters. Also, even more sophisticated representations, e.g., [4], can be used. We defer these directions to a future work.

Moving Objects Rejection. As mentioned before, for any realistic driving scenario, some of the clusters may belong to moving entities. Thus, these clusters should be ignored when estimating the ego-vehicle movement. To this end, we use the previous ego-motion estimation as a prior for the possible ranges in the transformations of non-moving objects.Fig. 3 gives an example of identified moving objects.

Ego-Motion Calculation. Following the moving-objects removal, we calculate the ego-vehicle transformation using the remaining static objects. Currently, this calculation is done using the same registration approach used for local clusters. Moreover, the point cloud is uniformly sampled in order to avoid a bias toward denser regions.

Computational complexity. It was shown in [3] that ground removal and clustering runs at 74 Hz fps on a mobile platform. As the rest of the pipeline processes at most an order of tens of clusters, it adds a negligible run-time cost.

3. Experiments

We evaluated our approach on the odometry data set of the KITTI Vision Benchmark [5], where we used the provided point clouds from the Velodyne HDL-64E. Fig.2 compares the algorithm results to the ground-truth on an entire sequence for 2D and 3D trajectories. In addition to the KITTI challenge metrics, we have analyzed our algorithm based on the relative pose error between Δ consecutive frames. We define $\mathrm{trans.err}_\Delta$ and $\mathrm{rot.err}_\Delta$ as the translation and rotation errors with respect to the ground-truth relative transformation between these Δ frames.

In Table 1, the translation and rotation errors are calculated for several values of Δ. For operational requirements, the results indicate that a separation between static and moving object can be done for any object at speed above approximately 1 km/h. In addition, Fig. 4 describes the translation error vs. the vehicle ground-truth speed. It can be seen that the mean and std of the translation and rotation angles errors remain relatively stable for the measured vehicle speeds and they are no more than 3 cm and 0.2 degrees correspondingly.

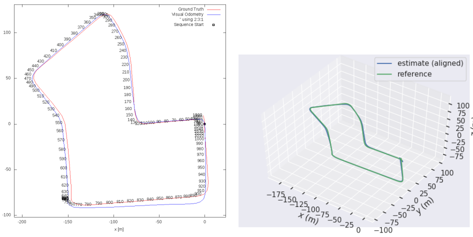

Figure 2. Trajectory plots of KITTI Seq. 07 with ground truth 2D (left) and 3D (right).

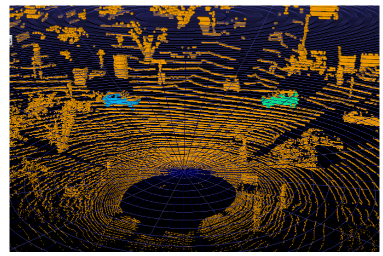

Figure 3. Example of the moving object rejection output.

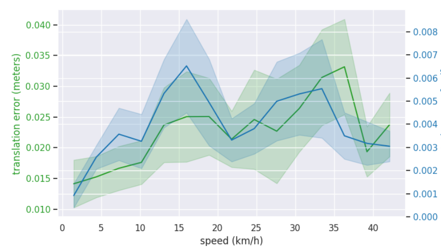

Figure 4. Average translation errors vs. ground-truth speed.

Frame-interval Δ	5	10	20	50	100
$\mathrm{trans.err}_\Delta$ [**m**]	0.102	0.186	0.334	0.781	1.618
$\mathrm{rot.err}_\Delta$ [**rad**]	0.014	0.019	0.026	0.035	0.049

Table 1. Average translation and rotation errors as a function of Δ.

4. Conclusion

In this paper, a cluster based approach for ego-motion estimation using a LiDAR sensor is proposed. The strength of our approach is that it distinguishes between static and moving entities. It can be also used to calculate the more challenging problem of 3D scene flow of rigid-objects. An important property of our method is that it is highly efficient and suitable for real time platforms.

References

[1] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, Feb 1992.

[2] P. Biber and W. Strasser. The normal distributions transform: a new approach to laser scan matching. In *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, volume 3, pages 2743–2748 vol.3, Oct 2003.

[3] Igor Bogoslavskyi and Cyrill Stachniss. Efficient online segmentation for sparse 3d laser scans. volume 85, pages 41–52, 2017.

[4] Benjamin Eckart, Kihwan Kim, and Jan Kautz. Hgmr: Hierarchical gaussian mixtures for adaptive 3d registration. In *ECCV*, 2018.

[5] A Geiger, P Lenz, C Stiller, and R Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[6] Qing Li, Shaoyang Chen, Cheng Wang, Xin Li, Chenglu Wen, Ming Cheng, and Jonathan Li. Lo-net: Deep real-time lidar odometry. *ArXiv*, abs/1904.08242, 2019.

[7] Wei-Chiu Ma, Shenlong Wang, Rui Hu, Yuwen Xiong, and Raquel Urtasun. Deep rigid instance scene flow. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[8] Franois Pomerleau, Francis Colas, Roland Siegwart, and Stéphane Magnenat. Comparing icp variants on real-world data sets. *Autonomous Robots*, 34:133–148, 2013.

[9] Martin Velas, Michal Spanel, Michal Hradis, and Adam Herout. Cnn for imu assisted odometry estimation using velodyne lidar. *2018 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pages 71–77, 2017.

[10] Ji Zhang and Sanjiv Singh. Loam: Lidar odometry and mapping in real-time.