3D LiDAR-IMU Integration for State Estimation and Verification Using a GNSS/INS/LiDAR Simulation Chain

Daniela Sánchez-Morales, Mohamed Bochkati, Shan Zhao, Thomas Pany
Institute of Space Technology and Space Applications (ISTA), Space Systems Research Center (FZ-Space)
Universität der Bundeswehr München

Research Track: Algorithms and Methods, MULTISENSOR INTEGRATED SYSTEM TECHNOLOGIES

Keywords— Multisensor integrated navigation system, GNSS denied environments, 3D LiDAR-IMU integration, Multisensor simulation tool chain

ABSTRACT

Multi-sensor fusion is an inevitable part when designing an autonomous vehicle. The integration of measurements coming from different sensors has the advantage that the navigation system is more likely to provide a solution in a more varied range of scenarios. Due to the nature of the measurement principle of each sensor, the error dynamics are completely different and the strengths of a certain sensor compensate for the weaknesses of another. Nevertheless, in order to design and tune our navigation system according to the application we are targeting, it is needed to have a flexible framework where we can test different type of sensors, different sensor configurations, environments (urban, semi-urban, rural) and vehicle dynamics. Acquiring data from measurement campaigns is not only costly and time consuming, but usually one cannot have complete control of the environment and it is also challenging to acquire ground truth data.

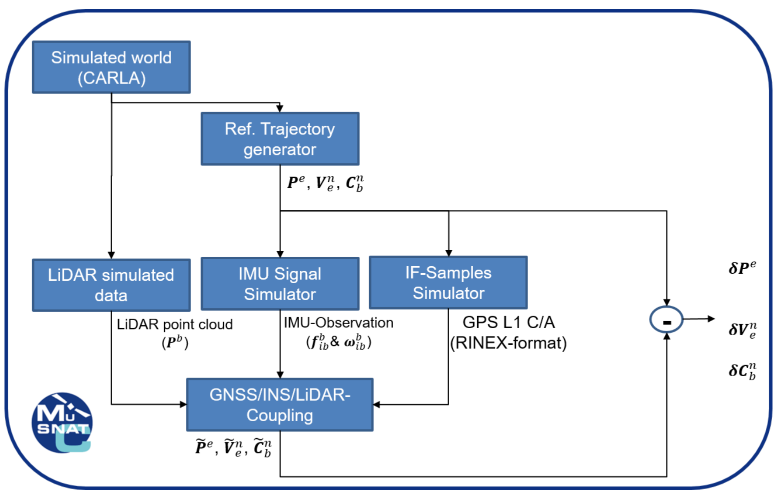

Figure 1: Extended SIMSS tool.

On a previous work [1], we presented the Satellite and Inertial Measurement Simulation System (SIMSS), where we successfully showed the generated synthetic data for both GNSS and IMU sensors. In this paper, we introduce an extension to the SIMSS-tool that is now connected to an Autonomous Driving simulator called CARLA [2] [3], from which we can get simulated LiDAR data. The advantage that it brings compared with other approaches is that CARLA already supports a 3D LiDAR sensor that can be configured and from which we can obtain point clouds with different properties, like data acquisition rates, different vertical angle resolutions and the possibility to exchange and import maps and test diverse environments. Moreover, one can have the possibility to test more realistic vehicle dynamics as the vehicle follows realistic traffic rules and traffic behavior. Contrary to the capabilities of CARLA, the SIMSS-tool is able to generate synthetic data of IMUs, i.e. accelerations and rotation rates in the body-frame, where one can explicitly define more advanced error models for different classes of IMU sensors, as well as synthetic RTK-processable phase data at intermediate frequency (IF) and RINEX level. Whereas CARLA only provides a very basic sensor model definition for that pair of sensors (gyro biases and assumes white noise for all axes).

We also present the updated framework (see Figure 1) of the interconnected simulation tools within SIMSS and the MuSNAT (Multi-Sensor Navigation Analysis Tool) [4], which is a versatile platform that uses various advanced GNSS positioning techniques and sensors such as IMU

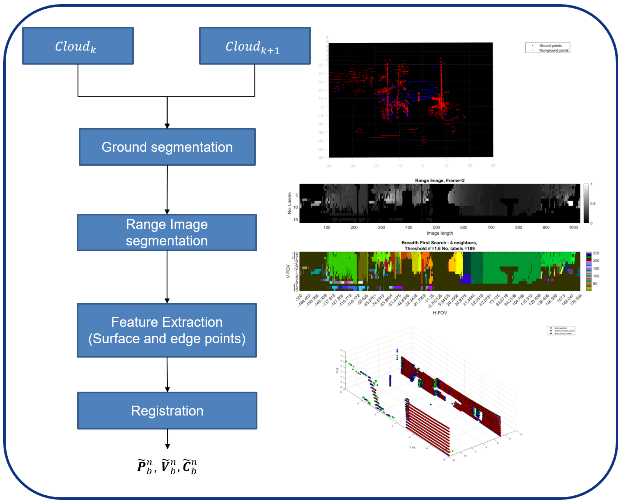

Figure 2: LiDAR processing

and LiDAR. We will be using this framework for future research. One of the main differences from the previous version of the SIMSS tool is that now the trajectory comes from CARLA as it provides the map, where the car navigates. Therefore, an interface between CARLA and SIMSS has been created. A simple python module has been developed in order to perform a way-point navigation and be able to design our own routes in the given environment model. From the simulation we acquire, besides the LiDAR point clouds, the state of the vehicle and the values recorded from an IMU sensor, the latest only for the sake of completeness. It is worth to clarify that CARLA is built over a client-server architecture. The server, which runs the simulation, and the clients, which are Python scripts running in the same or different computers, and retrieve information or send changes to the server. As we are retrieving information from more than one source, synchronization becomes relevant. Therefore, the complete control of the simulation is taken by the client and the server waits for the next instruction. Lastly, the simulation has been configured so that the step between two simulation moments have a fixed value to avoid unreliable data acquisition from the sensors.

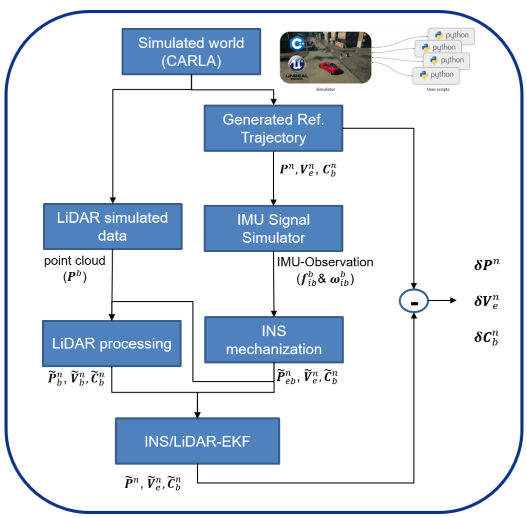

Figure 3: Proposed closed loop for LiDAR-IMU integration.

The rest of our work will be focused on the integration of the LiDAR and IMU data using the extended SIMSS tool. Figure 2 shows the LiDAR processing module. This module is mainly composed of the following steps: Pre-processing, segmentation, feature extraction and registration. During pre-processing, the points laying in the ground are extracted from the data set, this is done by using a common method in 2D LiDAR point clouds, 2D line tracking. We followed the approach suggested by [5], as it claims to be very cost effective computationally, which is very important for the type of applications we are aiming. For the segmentation, we chose to generate range images from the LiDAR measurements and take again advantage of a 2D data set (the images) as done in [6] in order to perform the segmentation. The method is based on the information that the depth of neighbors in the image can provide, through a *depth angle* threshold, one can differentiate which data points belong to a particular object in the scene. We believe that it is a research topic that has to be further exploited as one can take advantages of common image processing methods to extract information from this type of data sets, especially because the range in most LiDAR sensors are very accurate. We have found that principally when the vertical resolution of a data set is poor, this method may not be best approach. Once the point cloud is segmented, we still add a module for feature extraction, where we classify the points and based on the characteristics of the cluster we label them. The features that are supported are mostly vertical structures like poles and trees and planar structures. Based on the final clusters the registration is performed.

The proposed strategy for the 3D LiDAR-IMU integration is shown in Figure 3, where we aim to help the LiDAR registration process by providing an initial translation and rotation guess computed by the INS mechanization module. This not only helps the registration to converge faster, but it reduces the chances of having a registration error, particularly when not many features are detected. Likewise, it is also expected to reduce the IMU drift by feeding back the best estimation of the IMU errors every time a new successful pair of point clouds have been registered. Previous works have tackled this problem [7] [8] [9], nevertheless they usually assume a very simple error model for the IMU. In contrast, our implementation considers not only accelerometer and gyro bias but also a scale factor and other stochastic models like Gauss-Markov (GM) model. This is particularly important when working with low-cost MEMS-IMU sensors, as a simpler error model would not characterize correctly the sensor behaviour. Another advantage is that for long distance drives, a gravity model (e.g. Somigliana) is also available, so a more realistic g values in the mechanization process can be use instead of a constant value.

Finally, an integrity assessment will be performed based on different scenarios. We will conclude how our approach performs under different situations, namely: feature-rich and featureless environments, vehicle dynamics and different quality graded IMUs, i.e. from consumer grade to navigation grade devices. As a result, we will obtain a module that will strengthen our navigation system in cases like GNSS outages, certain vehicle dynamics and urban environments.

The future that we foresee from this development is to further aid the tracking loops of GNSS with reliable velocity information coming from the 3D LiDAR and the IMU, the so called doppler-Aiding.

REFERENCES

[1] A. Schütz, M. Bochkati, D. Maier, and T. Pany, "Closed-Loop GNSS/INS Simulation Chain with RTK-Accuracy for Sensor Fusion Algorithm Verification," in Proceedings of the 33rd International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2020), 2020, pp. 2867–2877.

[2] (2017) CARLA open-source simulator for autonomous driving research. [Online]. Available: https://carla.org/

[3] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An Open Urban Driving Simulator," in Conference on robot learning. PMLR, 2017, pp. 1–16.

[4] T. Pany, D. Dötterböck, H. Gomez-Martinez, M. S. Hammed, F. Hörkner, T. Kraus, D. Maier, D. Sánchez-Morales, A. Schütz, P. Klima et al., "The Multi-Sensor Navigation Analysis Tool (MuSNAT)–Architecture, LiDAR, GPU/CPU GNSS Signal Processing," in Proceedings of the 32nd International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2019), 2019, pp. 4087–4115.

[5] M. Himmelsbach, F. V. Hundelshausen, and H.-J. Wuensche, "Fast segmentation of 3d point clouds for ground vehicles," in 2010 IEEE Intelligent Vehicles Symposium. IEEE, 2010, pp. 560–565.

[6] I. Bogoslavskyi and C. Stachniss, "Efficient online segmentation for sparse 3d laser scans," PFG–Journal of Photogrammetry, Remote Sensing and Geoinformation Science, vol. 85, no. 1, pp. 41–52, 2017.

[7] H. Ye and M. Liu, "LiDAR and Inertial Fusion for Pose Estimation by Non-linear Optimization," arXiv preprint arXiv:1710.07104, 2017.

[8] H. Ye, Y. Chen, and M. Liu, "Tightly Coupled 3D Lidar Inertial Odometry and Mapping," in 2019 International Conference on Robotics and Automation (ICRA). IEEE, 2019, pp. 3144–3150.

[9] K.-W. Chiang, G.-J. Tsai, Y.-H. Li, Y. Li, and N. El-Sheimy, "Navigation Engine Design for Automated Driving Using INS/GNSS/3D LiDAR-SLAM and Integrity Assessment," Remote Sensing, vol. 12, no. 10, p. 1564, 2020.