Proceedings of the 16th International IEEE Annual Conference on
Intelligent Transportation Systems (ITSC 2013), The Hague, The
Netherlands, October 6-9, 2013

Multi-Target Tracking using a 3D-Lidar Sensor for Autonomous Vehicles

Jaebum Choi, Simon Ulbrich, Bernd Lichte, and Markus Maurer

Abstract — **Environmental perception is a prerequisite for autonomous driving and also a challenging task particularly in cluttered dynamic environments such as complex urban situations. In this paper, we present a robust algorithm for Multi-Target Tracking (MTT) using a Velodyne 3D HDL-64 Lidar sensor. The main contribution of this paper is a practical framework for selecting and representing useful information from the sensor raw data. Since the sensor produces a huge amount of data, a perception algorithm cannot be carried out in real-time without simplifying the sensor information. Unlike prior works, we introduce hybrid ground classification and the Region of Interest (ROI) identification method in order to filter out the amount of unwanted raw data for the actual tracking. And the environment is also abstracted based on an occupancy grid map. Moreover, we introduce feature based object geometry for precise estimation of the system state. In contrast to prior approaches, which use object geometry for the classification, we use it in order to compensate the unintended dynamics caused by shape change or occlusion. Our proposed MTT algorithm is able to run in real-time with an average processing time of 20ms. We evaluate it using our experimental vehicle "Leonie" in complex urban scenarios.**

I. Introduction

In fact, partially automated systems such as ACC (Adaptive Cruise Control), PCS (Pre-Crash Safety System), or LKS (Lane Keeping System) are already mass produced all over the world. This kind of system can improve not only the convenience but also the safety of vehicle passengers and other road users. However, there are still a number of problems to realize highly automated driving. In the project *Stadtpilot*, the Technische Universität Braunschweig focuses on automated driving in urban environments. The goal of the project is to drive autonomously through the entire ring road around the inner city of Braunschweig [13]. In this project, reliable perception of the environment is the first and most important step for autonomous driving. Recently, 3D-Lidar sensors have been widely used in this area because they provide rich and accurate data of spatial information around the vehicle [1, 3, 4, 7, 10]. In this paper, we propose an effective MTT algorithm using a Velodyne 3D HDL-64 Lidar sensor. Especially, the practical framework for selecting and representing useful information from the sensor raw data and the feature based object geometry for precise estimation of the system state are introduced.

The sensor has 64 laser beams which have different pitch angles and rotate as a single unit at a frame rate of 10Hz. The sensor output has 360 degree horizontal Field of View (FoV) and 26.8 degree vertical FoV with 120m detection range. It produces approximately 130,000 points in a single scan which makes a signal processing challenging tasks. Therefore, an efficient signal pre-processing algorithm such as ground classification and ROI identification is a prerequisite to handle dense data from the sensor in real-time.

The sensor can depict the whole surrounding of the vehicle without the aid of additional sensors. However, not all measurements correspond to objects but some of them belong to the surface. Therefore, ground classification and ground removal has to be executed before any following processing steps. The typical algorithms for that are summarized in [4] and limitations of each algorithm are discussed as well. Another approach for ground classification is presented in [10] which uses a 2.5D occupancy grid. However, it also has a problem with regard to overhanging objects. Since none of the mentioned approaches is robust enough separately, we have tried to implement a hybrid method. Especially, two main methods such as adjacent-beams-comparison and 2.5D occupancy grid methods are used in our application.

When the ground data is removed, a large number of data points still remain and it is not easy to consider all of them for tracking with limited computational resources. Moreover, some of them are not really needed for dynamic object tracking. Hence, the ROI is identified and then only objects in such regions become candidates for the tracking. In [3], a digital road map is used to restrict the search to the road regions, anyhow a problem is that precise road map data is not always available. Therefore, the road feature classification method is implemented in our application. The ROI is established with classified road features such as buildings, fences, etc. Even though the classification algorithm is similar to [1] which uses the properties of an object hypothesis such as the ratio between the length, width and height, the purpose of our classification is the ROI identification instead of the moving objects detection.

The resulting data is clustered into segments to represent objects in the environment. Detected objects have to be tracked to estimate the states of the object. The main difficulty of MTT comes from association strategies but various approaches to solve this problem have been developed as presented in [9]. Another issue of the object tracking is the dependency between geometric and dynamic properties of an object [3]. Since the Lidar sensor delivers only the visible section of objects, not only the position but also the shape and size of objects are changed over time, and this leads to inaccurate estimation of system states consequently. The shape change according to the observation position or occlusion is one of the typical examples for that. To solve this problem, a model based

J. Choi, S. Ulbrich, B. Lichte and M. Maurer are with the Institute of Control Engineering, Technische Universität Braunschweig, Hans-Sommer-Str. 66, 38106 Braunschweig, Germany (e-mail: jaebum, ulbrich, lichte, maurer@ifr.ing.tu-bs.de)

978-1-4799-2914-6/13/$31.00 ©2013 European Union

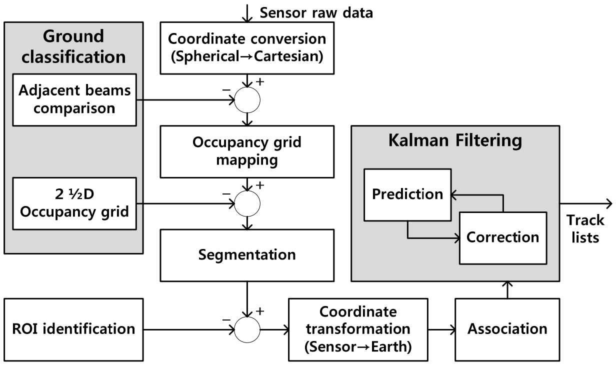

Figure 1. The architecture of tracking process

approach was proposed, which encompasses both geometric and dynamic properties of the tracked vehicle. In contrast to prior works, which use object geometry for the classification [1], the object geometry is modeled to eliminate the ambiguity between the shape and motion of the object. Our system is based on the approach in [3], in which the sensor data are represented as a 2D virtual scan and a Rao-Blackwellized particle filter (RBPF) is used to estimate both geometric and dynamic properties of the tracked vehicle. This approach eliminates the need for data segmentation and association steps. Since only the closest object in each angular grid cell is considered for the tracking, the computational complexity of the filter is acceptable. However, that is unacceptable in our application in which all objects are candidates for the tracking. Therefore, we implement a Kalman filter instead of a RBPF and the object geometry is estimated by segmentation and association functions. Consequently, sensor measurements are revised every time step by geometric model and filter correction is performed with compensated measurements.

II. System configuration

Our goal is to produce a list with movable objects using the raw data acquired from the 3D-Lidar sensor and provide it to each application. The architecture of the whole tracking process is shown in Figure 1. In our approach, dense data from the sensor is simplified by hybrid ground classification and ROI identification. The data which is not essential for tracking dynamic objects is removed in these steps. Then, measurement object hypotheses are constructed using the segmentation function. Afterwards, each measurement object hypothesis is assigned to a relevant track through the association function. Finally, the system states of each track are estimated with the system model and measurements recursively. Additionally, coordinate conversion/transformation and occupancy grid mapping is essential to handle the data properly at each step. The details of these functions are described in the following section.

A. Coordinate conversion and transformation

Although the output data from the sensor are based on Spherical coordinates, we perform all calculations in Cartesian coordinates to set up the state equation linearly. In our system, the Cartesian coordinates are defined according to the DIN 70000 standard (described in [6]). The X axis points to the forward or driving direction. The Y axis which represents the lateral direction is positive to the left from the driver position and the Z axis satisfies the right hand rule and therefore points upward. The conversion from Spherical to Cartesian coordinate is governed by the following equations.

$$\begin{cases} x = r\cos\zeta\cos\varphi \\ y = r\cos\zeta\sin(-\varphi) \\ z = r\sin\zeta \end{cases} \quad where\,, \begin{cases} r: \text{radial distance} \\ \zeta: \text{pitch angle of each beam} \\ \varphi: \text{rotation angle} \end{cases}$$

In addition, multiple coordinate systems such as sensor, vehicle and earth fixed coordinate systems are used in our tracking process. Therefore, it is needed to use coordinate transformation to convert back and forth between them. The formula for the transformation can be stated as below.

$$\vec{X}' = R\vec{X} + \vec{P} \qquad where\,, \vec{X} = [\,x\;y\;z\,]^T$$

The vector $\vec{P}$ is interpreted as a shift in the origin of the coordinate system. In our application, the origin of the vehicle fixed coordinate system is located at the center of the rear axle. The matrix R is the rotation matrix which is represented as a multiplication of Euler rotation matrices for each of the three axes.

B. Occupancy grid mapping

With the assumption that the Ego-vehicle position is known, we can build an occupancy grid map to represent the position of the objects in the environment as described in [2]. In fact, the position representation of a grid cell is not accurate because of the approximation of points. However it can provide all necessary data by choosing a small enough cell size. Since the occupancy grid abstracts the environment, the amount of data to be handled can be decreased especially when the sensor produces extremely dense data points surrounding the vehicle. In fact, a 3D mapping provides more detailed information about the environment compared with 2D mapping, whereas it has a problem with system resources such as CPU power and memory during real-time computation. Therefore, we implement a revised 2D occupancy grid in our application. It is organized as a radial grid and each cell holds information, whether it is occupied or not, how many points are inside of it and the min/max value of the height. This additional information beside the x-y position of each cell contributes to a better performance of segmentation and classification. Besides, the occupancy grid map is constructed locally relative to the Ego-vehicle. As an Ego-vehicle moves to another position the occupancy grid map is created newly and the information of each cell is not accumulated but updated in every time scan. As shown in Figure 1, most of tracking tasks are performed directly on the occupancy grid mapping representation.

III. Ground classification and ROI identification

In order to track dynamic objects from dense Lidar data, we attempt to minimize the amount of points as much as possible. This is achieved by a two step procedure: In the first step, we do a ground classification and remove those points belonging to the ground. In a second step, we remove objects, which are not essential for objects tracking by ROI identification.

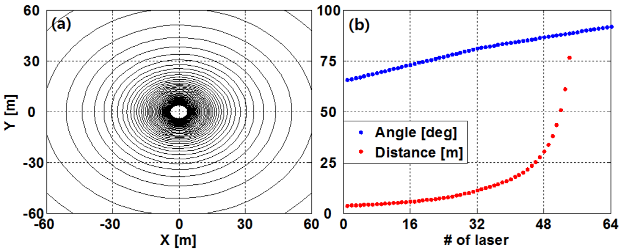

Figure 2. Sensor characteristics: (a) Top view of each beam, (b) Pitch angle and radius of each beam (∵ sensor height is fixed, 1.78 m and ground is flat)

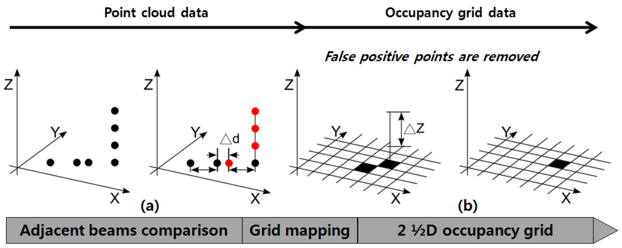

Figure 3. Hybrid ground classification and removal: (a) adjacent-beams-comparison (black: ground, red: objects), (b) 2.5D occupancy grid (false positive points are removed based on height difference of grid cell)

A. *Ground classification*

Before discussing the algorithm, the reader should note some characteristics of the sensor. Since each beam of the sensor has a fixed pitch angle and rotates 360 degree, it forms a circle and the radius of the circle depends on the height and pitch angle as shown in Figure 2. For ground classification, a hybrid method using adjacent-beams-comparison and 2.5D occupancy grid methods is applied in our application, and it is illustrated in Figure 3.

Initially, the ground is removed with an adjacent-beams-comparison method based on point cloud data. This method uses a radius difference between adjacent beams. This radius difference between adjacent beams is almost zero at the object, while it is not zero, but ideally a constant value at the ground because each beam has a different radius according to the pitch angle as shown in Figure 2. Therefore, the points which have a bigger difference in the radius than a given threshold are treated as ground and can be removed. The important thing is that the threshold is not a fixed value but function of radius in order to overcome the problem generated by road slope, pitch and roll of Ego-vehicle. Since these factors affect the radius of each beam, the radius difference of adjacent beams is also influenced even if adjacent beams have the same pitch angle. Therefore, it is reasonable to set the threshold according to the radius of the beams as described in [3].

The resulting points which do not belong to the ground are mapped into the grid as explained in section II-B. After occupancy grid mapping, some false positive points caused by noise and inaccuracy of the sensor are removed with 2.5D occupancy grid method. This method uses the height difference of each cell instead of radial difference between adjacent beams. This is achieved by removing grid cells which have a height difference below a threshold. Figure 4 shows an example of ground classification and removal generated from

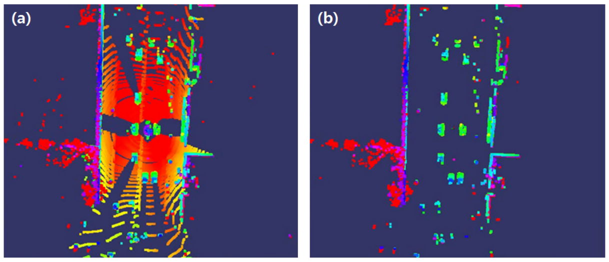

Figure 4. Ground classification and removal: (a) before classification, (b) after classification (difference colors are used to represent height of points)

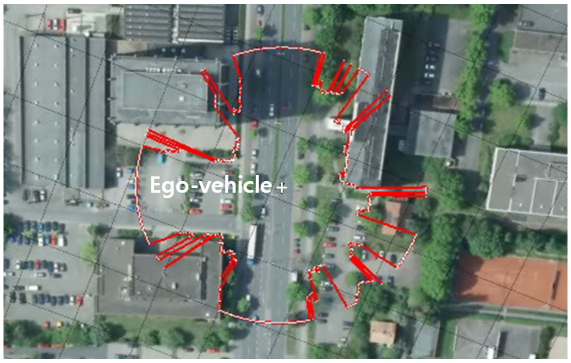

Figure 5. ROI identification using road features classification. Aerial image: ©Stadt Braunschweig, Abteilung Geoinformation (NR. 011/2010)

the hybrid method. Additionally, overhanging objects whose minimum height is bigger than a given threshold are ignored because they are not obstacles which interrupt the driving of the Ego-vehicle.

B. ROI identification

The ROI is identified by a classification of road features. Then, the objects which are outside of ROI can be ignored in the filtering process. To classify road features, segmentation of the occupied grid cell is initially performed. It is based not only on the Euclidean distance in x-y plane but also on a maximum height difference between neighboring cells. In other words, all grid cells of a segment are quite close to each other and have a similar value of maximum height. In here, each segment is depicted as a hypothesis of a 3D bounding box which is organized with width, length and height. In this representation, big sized segments or the segments whose ratio between width, length and height belong to a special category are recognized as road features. A true negative classification (e.g. not classifying a tree as a road feature) could only degrade the performance of the ROI identification algorithm. On the contrary, a false positive classification (e.g. classifying vehicle as a road feature) may cause an important dynamic object to be ignored, which can be very dangerous from an application point of view. Therefore, it is important to set the threshold not to avoid classifying any dynamic objects as road features. The line which connects all the segments of the road features is a boundary of ROI. While the segments inside of ROI are considered as tracking objects, the segments outside of ROI are simply removed. The amount of the reduced computational burden depends on the driving environment but it's true that the algorithm is more effective in complex urban situation

978-1-4799-2914-6/13/$31.00 ©2013 European Union

comparing with the open space. Figure 5 shows one of ROI identification results on Braunschweig's city ring.

IV. SYSTEM MODELING AND FILTERING

In this section, the system modeling and filtering method are described. After ground classification and ROI identification, only selected segments are considered as dynamic objects for tracking. Afterwards, the system states of each dynamic object are estimated using a Kalman filter. In our application, both the dynamic and geometric models are implemented for more accurate estimation of the system states. Since the object hypothesis is represented by a 3D bounding box and the shape of hypothesis is not fixed but changes over time, its geometric properties cause a movement by themselves. Therefore, the geometric model has to be considered during the filtering process for the precise motion estimation.

A. Dynamic model

The dynamic model of the object is comprised of the process and measurement model. Initially, the dynamic equation of the process model is derived by assuming that the tracked object moves with constant velocity during cycle time. This assumption is possible under the condition that the cycle time is small enough so that the effect of the acceleration or deceleration can be ignored. In our application, the sensor frame rate is high enough and therefore the dynamic equation is given as follows.

$$x_k = F_k x_{k-1} + w_{k-1} \quad (\because x_k = [x\, y\, \dot{x}\, \dot{y}]^T)$$

$$where, \quad F_k = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \ w_k \sim N(0, Q_k)$$

F_k is the state transition matrix and w_k is assumed to be zero-mean, Gaussian white process noise with covariance Q_k. In the above equation, it is clear that the process model is based on the constant velocity motion model since the velocity of the current time step is the same as the previous one.

Since only the position of objects is directly measured by the sensor, the equation of the measurement model can be written as follows.

$$z_k = H_k x_k + v_k$$

$$where, \quad H_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \ v_k \sim N(0, R_k)$$

H_k is the observation matrix and v_k is the observation noise which is assumed to be zero-mean, Gaussian white noise with covariance R_k.

B. Geometric model

In our application, the system states are position and velocity and these are defined based on a reference point which is normally the center point of the object. However, the problem is that the reference point is polluted by shape changes according to the observation position or occlusion as shown in Figure 6. That means geometric characteristics can make a dynamics themselves. For example, although the target object is static, the shape of hypothesis can be changed according to the position of the Ego-vehicle and it generates velocity as the reference point moves within the hypothesis (see Figure 6 a). Consequently, the system states are not estimated correctly as long as the geometric characteristics are not considered.

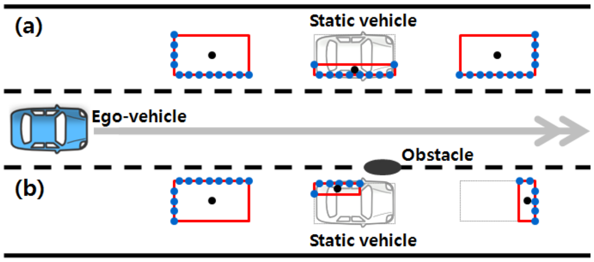

Figure 6. System state error caused by shape change according to the observation position (a) and occlusion (b). (blue dot: detection point, black dot: reference point of object, gray dotted rectangle: actual shape of object, red rectangle: object hypothesis)

The idea of our approach to overcome this problem is to do measurement updates of a Kalman filter considering geometric factors. Initially, the track split or velocity error caused by occlusion is solved by basic geometric factors such as width W, length L and height H. As long as the new measurement is within the predicted hypothesis, the width, length and height of the track are not updated. Therefore, the object hypothesis is maintained in spite of the occlusion and therefore its reference point gets to be also stable. In addition, the velocity error caused by shape change according to the observation position has to be solved as well. To compensate this velocity error, the width and length difference between the previous state and new measurement are additionally defined in the geometric model. These factors compensate the measurement offset generated by shape changes. In fact, only the velocity is updated with compensated measurement because the position is updated with object hypothesis simultaneously. So, the complete geometric model is designed as follows.

$$G = \{W, L, H, \Delta W, \Delta L\}$$

This geometric model is independent from the type of objects and the error of the geometric model is not accumulated because all geometric factors are updated at every time scan. The details of measurement updates using the geometric model are explained in next section, IV-C.

C. Kalman filtering

Since we model the object as a linear system perturbed by Gaussian noise, the Kalman filter can be implemented for the tracking. An overall filtering process is depicted in Figure 7. While the system states of the object are estimated with system model and previous states in the prediction phase, an appropriate measurement is needed to correct the predicted states in the correction phase. The final measurement for the correction is selected and summarized by the gating and association functions.

The gating function draws a boundary around the prediction position of the track. While measurements lying inside of the gate are considered valid, those outside of it are rejected. In

978-1-4799-2914-6/13/$31.00 ©2013 European Union

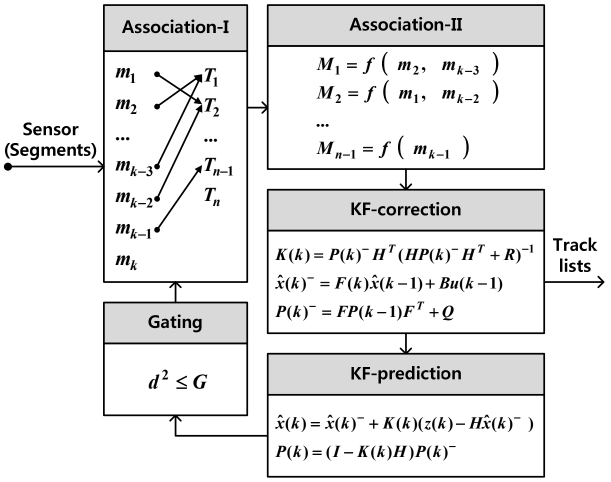

Figure 7. Filtering process (m: measurements, T: tracks, M: final measurements which are used in filtering update)

other words, this function eliminates unlikely pairing between measurement and track before data association is done. Therefore, the amount of computation can be reduced because only relevant measurement data is processed. In our application, adaptive gate according to the velocity vector is used. Although the gate has the shape of a rectangle box and its size is fixed, the direction of the gate is changed according to the velocity vector.

The association function is one of the most important parts in the filtering process since assigning the wrong measurement to the track often results in track loss and split. Normally, the association function can be divided into two steps. One is to decide which measurement belongs to which track (see Association-I in Figure 7). Although some measurements are already filtered out by the previous gating function, there are still possibilities that two or more tracks can occupy the same measurement. Moreover, some measurements cannot be associated with any tracks. Therefore, all measurements need to be assigned to each track properly. The other step is to summarize the related measurements into a single one (see Association-II in Figure 7). A number of techniques such as Global Nearest Neighbor (GNN) [11], Joint Probabilistic Data Association (JPDA) [12] and Multiple Hypothesis Tracking (MHT) [5] have been developed in this area. Although more complex JPDA and MHT provide improved performance, they are difficult to implement and require extensive computational resources. Therefore, the GNN method is used in our application and we have made an index function for that as follows.

$$A_i = (kd + (1-k)s)/k \quad (\because k \le 1)$$

Association index (A_i) is function of distance (d) and size (s) of a hypothesis. So, the final measurement which is associated with the track is decided with weighted sum of distance and size. Besides, a measurement which is not matched to any track builds a new track (see m_k in Figure 7) and a track which is not associated with any measurement is terminated (see T_n in Figure 7).

In measurement updates of the Kalman filter, the geometric model as described in the previous section is considered. While the position is updated with an original measurement, the velocity is updated with a compensated measurement by the geometric model. However, this is possible under the assumption that the shape of hypothesis is changed based on the closest sides from the Ego-vehicle as shown in Figure 8. This assumption is possible because the sensor detects the close parts of the object first. When the object hypothesis of time scan t-1 and t are H_{t-1} and H_t, the geometric change besides dynamic motion has to be considered. For that reason, both hypotheses are arranged based on the closest side from Ego-vehicle and then final measurements in the correction phase are determined as follows.

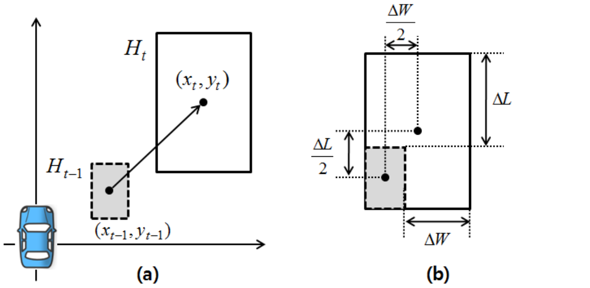

Figure 8. Filtering update considering geometric properties : (a) dynamic motion and shape change of object hypothesis, (b) Hypotheses arrangement based on the closest sides from Ego-vehicle.

$$\hat{x}(k) = \hat{x}(k)^- + K(k)[z(k) - H\hat{x}(k)^-]$$

$$where, z_p(k) = \begin{bmatrix} x_t \\ y_t \end{bmatrix}, \ z_v(k) = \begin{bmatrix} x_t - \dfrac{\Delta W}{2} \\ y_t - \dfrac{\Delta L}{2} \end{bmatrix}$$

The above formula is an equation of the measurement correction using measurement z(k) in Kalman filter. The measurement for velocity update is adjusted to compensate the unintended error caused by shape change.

V. Experimental Results

To verify our algorithm, the experimental vehicle “Leonie” is used [8, 13]. The Velodyne 3D HDL-64 Lidar sensor is mounted on top of the vehicle. The algorithm is verified in real-time on Braunschweig’s city ring in which various road users increase the difficulty of the MTT, especially motorbikes, trucks and pedestrians are also part of the test scenario. The ground classification and ROI identification results are presented already in Figure 4 and 5. Figure 9 and 10 represent a benefit of the geometric model visually. In Figure 6 (a), the Ego-vehicle goes straight nearby a static object on the left side and the results according to the application of geometric model are compared in Figure 9. As you see, the result with geometric model is more stable compared with the other one because the velocity is updated with compensated measurement as explained in the previous section. Figure 10 shows the example scene of the occlusion. Since there is an obstacle between the Ego-vehicle and a tracked object, the measurement data is divided into two parts and therefore a track split and velocity error occurs. However, the hypothesis

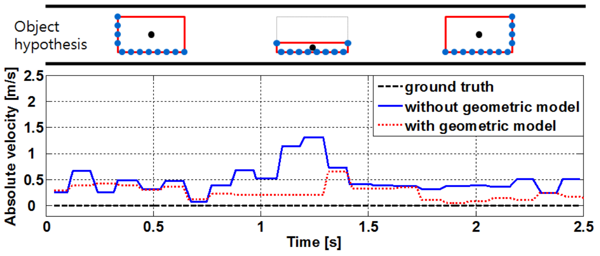

Figure 9. Test result in the situation illustrated in Figure 6 (a).

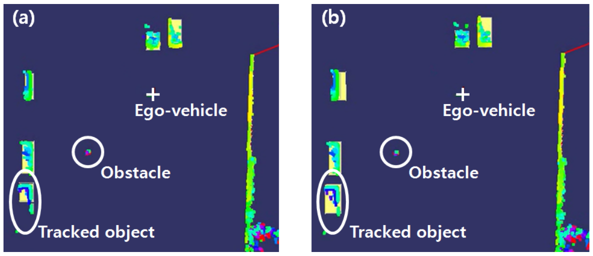

Figure 10. Example of occlusion (yellow boxes are tracks): while the track split and velocity error is occurred without geometric model (a), the hypothesis is maintained with geometric model (b) .

of the track can be maintained with geometric model and therefore the velocity is stable as well. Figure 11 shows a quantitative test result of the tracking. This is performed in a relatively complex urban scenario. There are a lot of objects, road features and noise around the vehicle as shown in Figure 11 (a). Initially, a considerable portion of points is removed with ground classification and the ROI identification. Then, the remaining points are segmented and tracked as shown in Figure 11 (b). Figure 11 (c) and (d) are position and velocity comparison results of a target object.

VI. Conclusion and Future Work

In this paper, an MTT algorithm using a Velodyne 3D HDL-64 Lidar sensor was implemented and evaluated. Experimental results have shown that our algorithm achieves the MTT successfully. Initially, valid objects which are used for tracking are selected through hybrid ground classification and ROI identification. The hybrid ground classification function using both adjacent-beams-comparison and 2.5D occupancy grid methods presents more robust results than a standalone method. And ROI identification shows that it is capable of significantly reducing the number of objects outside of ROI. Afterwards, the objects are modeled with geometric and dynamic properties together and then the system states are estimated using Kalman filter. More reasonable and improved estimation results are obtained by considering the geometric model.

Future works will be focused on more utilization of the sensor such as lane marking detection using the intensity of the beam. In this research, it is assumed that the Ego-vehicle position is known exactly. However, the local offset exists between the real position and the position oriented by GPS actually. In this situation, the localization problem can be improved with lane marking detection and it leads to a better performance of MTT algorithm as well.

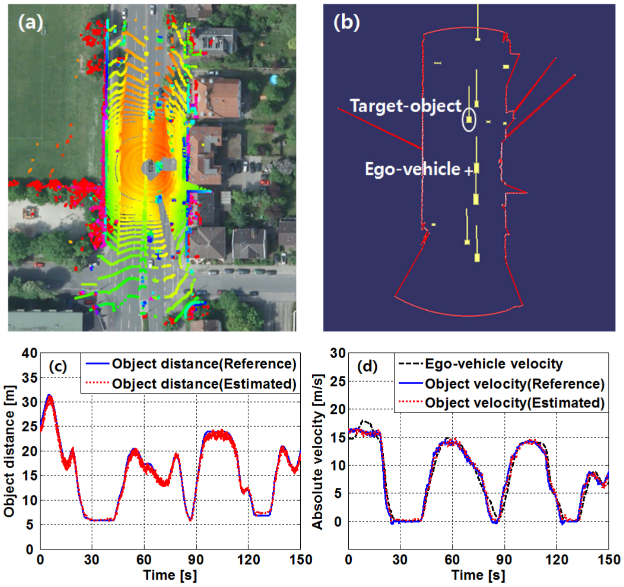

Figure 11. Test results: (a) Point clouds data around vehicle, (b) Tracking results (red line: ROI, yellow box: track or object, yellow line: velocity vector), (c), (d) Position and velocity comparison data. Aerial image: ©Stadt Braunschweig, Abteilung Geoinformation (NR. 011/2010)

References

[1] A. Azim and O. Aycard, “Detection, Classification and Tracking of Moving Objects in a 3D Environment,” in *Proc. 2012 IEEE Intelligent Vehicles Symposium*, Madrid, Spain, 2012.

[2] A. Elfes, “Occupancy grids: A probabilistic framework for robot perception and navigation,” *Journal of Robotics and Automation*, 3:249–265, 1987.

[3] A. Petrovskaya and S. Thrun, “Model based vehicle detection and tracking for autonomous urban driving,” *Journal of Autonomous Robots*, vol. 26, pp. 123-139, 2009.

[4] C. Guo, W. Sato, L. Han, S. Mita, and D. McAllester, “Graph-based 2D Road Representation of 3D Point Clouds for Intelligent Vehicles,” in *Proc. 2011 IEEE Intelligent Vehicles Symposium*, Baden-Baden, Germany, 2011.

[5] D. B. Reid, “An Algorithm for Tracking Multiple Targets,” *IEEE Trans. Automatic control*, Vol. Ac-24, No. 6, 1979.

[6] DIN 70000, Straßenfahrzeuge; Fahrzeugdynamik und Fahrverhalten; Begriffe, (ISO 8855:1991, modified), 1994.

[7] D. Steinhauser, O. Ruepp, and D. Burschka, “Motion Segmentation and Scene Classification from 3D LIDAR Data,” in *Proc. 2008 IEEE Intelligent Vehicle Symposium*, Eindhoven, Netherland, 2008.

[8] J. M. Wille, F. Saust, and M. Maurer, “ Stadtpilot: Driving Autonomously on Braunschweig's Inner Ring Road,” in *Proc. 2010 IEEE Intelligent Vehicle Symposium*, San Diego, USA, 2010.

[9] H.W. de Waard, “A New Approach to Distributed Data Fusion”, PhD Thesis, The University of Amsterdam, 2008.

[10] M. Himmelsbach, A. Müller, T. Lüttel, and H. J. Wünsche, “LIDAR based 3D Object Perception,” in *Proc. 1st International Workshop on Cognition for Technical Systems*, München, Germany, 2008.

[11] S. S. Blackman, *Multiple-Target Tracking with Radar Applications*, Artech House, 1986.

[12] T. E. Fortmann, Y. Bar-shalom, and M. Scheffe, “Sonar Tracking of Multiple Targets Using Joint Probabilistic Data Association,” *IEEE J. Oceanic Engineering*, Vol. OE-8 No.3, 1983.

[13] F. Saust, J. M. Wille, B. Lichte, and M. Maurer, “Autonomous Vehicle Guidance on Braunschweig’s Inner Ring Road within the Stadtpilot Project,” In Proc. 2011 IEEE Intelligent Vehicle Symposium, Baden-Baden, Germany, 2011.

978-1-4799-2914-6/13/$31.00 ©2013 European Union