Multimodal sparse LIDAR object tracking in clutter

Mircea Paul Muresan, Sergiu Nedevschi
Computer Science Department
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
{mircea.muresan, sergiu.nedevschi}@cs.utcluj.ro

Abstract— **one of the key components of the perception system in an autonomous vehicle or ADAS is the target tracking module. Using target tracking in the sea of clutter, self-driving cars are able to better understand the environment and make predictions about the surrounding objects. Cuboids obtained from a sparse LIDAR often exhibit a fluctuating behavior due to segmentation problems and errors accumulated from the motion correction module. Furthermore, targets in real life scenarios do not move in a predictable manner, so it is very difficult for a classical motion model to describe the complex behavior of any road objects in such cases. In this paper we propose a two-step data association scheme that efficiently and effectively finds correspondences between tracks and measurements. Then we aim to generate better position estimates for objects with an ambiguous dynamic behavior by associating and combining the results from two different motion models. The proposed solution runs in real time and it was validated using a high precision GPS, and also by projecting the prediction results in the corresponding intensity image and assessing whether the prediction falls on the correct item.**

Keywords— Vehicle Tracking, 3D LIDAR objects, Data Association, Clutter

I. Introduction

The autonomous driving and driver assistance systems are attracting a lot of public attention recently. One of the most important modules in understanding the traffic environments for the ADAS and self-driving cars is the detection and tracking of multiple objects. Object tracking usually seeks to encapsulate multiple unique traits of the tracked object such as object identities, velocities, positions, orientations and in the context of autonomous driving the class of objects. For the problem of vehicle perception, the task of object tracking is essential, as the environmental measurement is useful only if it is filtered (not noisy) and identifiable even in occluded situations such that the vehicle is capable of making use of the measurement and transform it into an actionable information.

Target tracking can be performed on multiple sensors. Traditional tracking methods were developed for RADAR applications, where they would operate on point observations. Such tracking solutions would vary in their methods of data association and state approximation. These algorithms are designed principally to allow for measurement inaccuracy, ambiguous relationship between observations and the physical objects which generate them and spurious observations caused by clutter or background noise [1]. In the context of intelligent autonomous vehicles or ADAS various sensors could be considered. The most common sensors used by researchers are the cameras, due to their cheap and versatile characteristics. However another research direction is to use time of flight type of sensors, with the most popular being the LIDAR (light detection and ranging) sensor. The advantage, tracking using LIDAR sensor, has to tracking in the visual color images consists in insensitiveness to illumination conditions. Most tracking solutions are implemented in 3D Cartesian coordinates under a sequential filtering framework. Some of the common steps involved in the tracking procedure are the point cloud segmentation, candidate matching and motion estimation [2]. High quality segmentation is a fundamental requirement for tracking algorithms to obtain high quality results. Several methods that exist in the literature work either on 2D grid map [3, 4, and 5] or on 3D occupancy grids with higher computational burden [5]. Candidate matching and tracking in cluttered environment are rather difficult with incorrect segmentation result. Popular methods for matching and motion estimation steps include feature based matching [16] followed by an estimation using Kalman or Extended Kalman Filter[5]. 3D objects obtained from sparse LIDAR point clouds are difficult to track mainly due to their change in appearance, unreliable dimensions and fluctuating positions in consecutive frames. Furthermore 3D objects which represent moving vehicles can have a very complex motion which cannot be described by a classical motion model. To this end in this paper we propose the following contributions:

- First we propose a two-step 3D data association scheme that associates LIDAR measurements to existing tracked objects

978-1-5386-8445-0/18/$31.00 ©2018 IEEE

- Then we propose a data association scheme for finding the correspondences between the tracks obtained from two different motion models
- Finally we combine the results of the positions of the two trackers using the vehicle orientation

The rest of the paper is organized as follows: In section II we overview the literature on data association and tracking. In Section III, the proposed solution is stated. In Section IV we evaluate the results using various experiments and compare the data to the information obtained from a high precision GPS. Section V concludes the paper.

II. Related Work

An essential part in the perception pipeline of autonomous cars and ADAS is represented by the object tracking component. A self-driving vehicle can make predictions about its surrounding objects location and behavior, and based on that it can plan next actions and make proper decisions. There is an extensive research literature on object tracking using multiple types of sensors. Generally autonomous vehicles [6, 7] can be equipped with sensors like mono and stereo cameras, thermal, night vision, LIDAR, Radar, Inertial Navigation System (INS), Global Positioning System (GPS), and Inertial Measurement Unit (IMU) in order to have a higher degree of perceiving and representing the environment.

The Kalman Filter (KF) [8] is based on the Bayesian method and it computes recursively the optimal parameter estimates from its posterior density. The Kalman Filter assumes the object dynamic function and posterior density are Gaussian distributions and the process and measurement functions are linear. As we know in the real world the object dynamics cannot be captured by linear motion model so the Kalman Filter had to be adapted to encapsulate a non-linear motion and measurement model. Two approaches were introduced, the Extended Kalman Filter (EKF) [9] and the Unscented Kalman Filter (UKF) [10]. The EKF uses a first order Taylor series expansion approximation to linearize the process function. However, linearization using Jacobians is computationally expensive so for this reason in this paper we have used the UKF, which makes an approximation based on the so called sigma point sampling.

The UKF is used widely in LIDAR MOT as presented in [2, 3 and 4] due to its low computational complexity comparable to the KF.

In this paper we will focus on tracking objects that come from a 16L 3D LIDAR, since this sensor represents the first option to acquire 3D spatial information in the intelligent vehicles context. Generally, object tracking algorithms can be divided into two categories based on representation scheme for the object [11]:

A) Tracking by detection (Discriminative) approaches

Discriminative object trackers localize the object using a pre-trained detector. Many approaches proposed for discriminative object tracking are based on monocular cameras. They main research focus for such approaches is data association. An overview of such methods can be seen in [12, 13]. In [14] Azim and Aycard proposed a method of detecting and tracking moving 3D LIDAR objects using a supervised learning approach. The main disadvantage of discriminative approaches is knowing beforehand the object categories.

B) Model Free (Generative) approaches

In order to have a reliable and robust perception system a more generic object tracker is required [15]. This type of generic tracker is able to track objects even if no prior knowledge is available. Generative tracking methods search for the next occurrence of an object by looking for the region that is most similar to a prior described model of that object. The object is usually updated online to handle its changing variations. Some of the available generative approaches track objects based on their motion [16, 17]. This is one of the most widely used methods and are related to the Detection and Tracking of Moving Objects (DATMO) [18]. One of the issues of these methods is that they are unable to detect stationary objects which could potentially move. To deal with these problems changes between two or three consecutive observations can be analyzed and identify the modifications, this is also known as frame differencing [16]. Another idea, is to build a static model of the scene, called background model and find deviations from this model. This process is known as “background modeling and subtraction” [19]. Stiller and Moosman perform a convexity based segmentation for detecting hypotheses. In their approach a KF and ICP are used for tracking moving objects and managing tracks. Their method includes the 3D reconstruction of the shape of moving objects. In [21] the authors use a GIS map to reject outliers. They track moving objects using Kalman Filter with a Constant Velocity model and ICP for pose estimation.

In an urban scenario objects do not move in a well-defined pattern. There is no defined motion model that can be used to precisely predict the trajectory of a vehicle all the time. For this reason there are some attempts in the literature that combine multiple motion models to obtain a more robust estimate. Interacting motion models are generally used for vehicle tracking having ambiguous behavior. Some approaches are presented in [29, 30]

One of the most challenging tasks when performing target tracking in heavy clutter is data association. A successful association means that a tracking hypothesis can be successfully and correctly updated. Since data association is just a small part of the tracking pipeline, this step has to be computed very fast. Given this circumstance some tracking solutions include the use of the strongest neighbor filter (SNF) [22] and Nearest Neighbor Filter (NNF) [23]. The SNF selects the measurement that has a highest score from a set of validated measurements, while the NNF selects the measurement that is closest to the predicted value. Another common approach has been the probabilistic data association filter (PDA) [24], which uses all the validated measurements to estimate the state and covariance of the object, rather than selecting a single measurement. The joint PDA is a variation of the PDA algorithm which deals with multiple targets. Another variation of the PDA is the integrated PDA, where the track existence probability and the association probabilities are estimated together [26]. Another more powerful algorithm in the family of DA algorithms is the multiple hypothesis tracker (MHT). The MHT associates all the measurements to the tracks, and gating and pruning are used to eliminate the tracks with low probability [27]. Even though the MHT has proven to be very accurate it is more expensive computationally. There are multiple variations of the MHT algorithm that try to reduce its complexity. For example the probabilistic MHT avoids assigning measurements to specific tracks and instead calculates the probability of each measurement to belong to each track [28].

III. Proposed Solution

In this section we present the proposed LIDAR object tracking in clutter solution. Detected LIDAR objects are first brought in the reference frame of the EGO vehicle and their position is compensated with respect to a reference time stamp. In the first frame, new tracks are initialized with the information coming from the LIDAR measurements. In the current work we are using two motion models. The first is the constant velocity model and the second is the constant velocity and turn rate motion model. The two models are used to cover complex vehicle motion that are present in the real world. In the following frames a two-step data association scheme is employed to make correspondences between existing tracks and new measurements. In case new measurements do not have any corresponding tracks that can be associated to them, new tracks are initialized for those measurements. In case a track was not associated with a measurement, a simple predict function is called so that the track is updated to the new time stamp using its current state and covariance. Tracks are implemented to have an internal “death clock”. This means that if a track is not associated and updated for a number of frames, it gets erased. After this step a track to track association is performed. Tracks coming from the two motion models are associated and the final position of an object is given using a weighted combination of the two with respect to the orientation of the vehicle.

A. Two step data association for LIDAR objects

Data association addresses the issue of determining corresponding relationships between targets and detections. The main issue that was observed for the LIDAR objects was their fluctuating nature. This means that the objects we are tracking did not offer reliable dimensions from frame to frame, and their position did not follow a smooth pattern. Furthermore, due to the fact that target tracking is just one part of a higher autonomous driving pipeline we could not perform any sort of cross validation using information from other sensors, in order to maintain a high running speed.

Before running our two step data association algorithm we had to change the reference point of the detected object. The current reference point of the LIDAR objects is the centroid of the object. The reference point was changed, taking into consideration the vehicle dimensions, to the corner which is closest to the ego vehicle since this is the position which has the highest reliability and its position fluctuates the least. Figure 1 below intuitively illustrates this process.

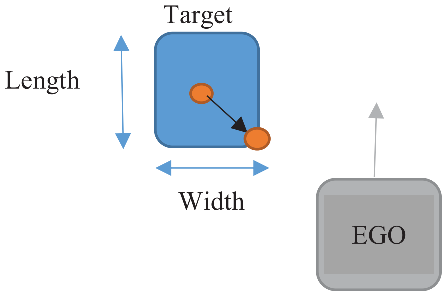

Fig 1. Changing the reference point of the target vehicle

The first step in our data association algorithm is to project the given object list onto a 2D color grid. In this grid, the cells occupied by LIDAR objects have the red channel set to 255 and the blue and green channels set to the position of the LIDAR object in the initial object list. Then we project the existing tracks onto the same virtual grid, they are depicted with white color in figure 2. The intersections with the measurements are marked with yellow for illustration purposes. It may happen that one track may fall onto multiple measurements. For this reason in every track we embed an array containing the maximum overlapping with each LIDAR object as described by (1).

$$\sum_{i=0}^{noTracks} \sum_{j=0}^{noLidarObjects} AreaIntersection_{i,j} = trackArea_i \cap LidarObjectArea_j \qquad (1)$$

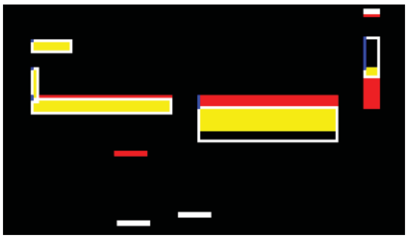

Fig 2. Overlapping tracked objects and measurements

For each LIDAR object we are selecting the tracks that are closest to it in the 2D image with respect to the Euclidian distance. This step is performed in order to filter the distant tracks from a reference LIDAR measurement. Afterwards the selected tracks are filtered further using their dimensions. Even though, the object size varies from frame to frame, tracks which are considerably smaller than the LIDAR objects and vice versa can be filtered out. So considering a target object (track or LIDAR) if the measured object area is comparable to the track area with a certain threshold we set a validation flag true (2).

$$validation_1(x,y) = \begin{cases} true, |x-y| \leq TH \\ false, otherwise \end{cases} \qquad (2)$$

In the equation above x and y are the areas for the track and LIDAR object. We further test if the largest visible face of the LIDAR object is comparable to the largest visible dimension stored in the track so far. If this condition also holds true, we compute the overlapping percentage between the track and the LIDAR object. For a LIDAR object detection, we are associating the track with the maximum overlapping percentage if it is larger than 5%. Figure 2 above illustrates the results of the first step of the association algorithm. The blue lines indicate object associations. There may be situations in which the predictions do not overlap at all on the detected LIDAR objects. This may happen due to accumulated errors in the motion correction of the point cloud or bad point cloud segmentation. The second step of our data association algorithm has been implemented to treat this case. The validation function used is presented in the equation 3 below.

$$validation_2(x,y) = \begin{cases} true,\ \ !used(x) \\ and\ !used(y) \\ and\ validation_1(x,y) \\ and\ |visibleFace_x - visibleFace_y| \leq TH \\ false, otherwise \end{cases} \quad (3)$$

In the equation above x and y represent the LIDAR and tracked object, used(a) is a function that informs us whether a specific object has been previously associated, visible face represents the maximum visible face seen with respect to the position of ego vehicle and the symbol $|a|$ denotes the absolute value of a. The track and LIDAR object pair that pass the validation step, and have minimum Euclidian distance are chosen for association.

The result of the second step can be seen in the figure 3 bellow. We marked with a blue line the correspondences between LIDAR object and track in order to better illustrate the data associations.

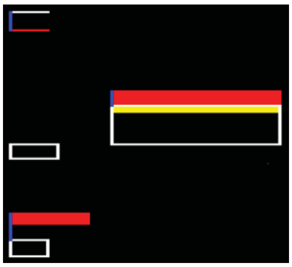

Fig 3. Result of associating objects that do not overlap

B. Target Tracking

In the real world the object dynamics cannot be captured only by linear motion model so the Kalman Filter had to be adapted to encapsulate a non-linear motion model. To reproduce a more realistic scenario two motion models are used. The two motion models used are the constant turn rate and velocity motion model (CTRV) and the constant velocity (CV) model.
The state vector is presented in equation x bellow.

$$X_k = \begin{bmatrix} px \\ py \\ v \\ \Psi \\ \dot{\Psi} \end{bmatrix} \quad (4)$$

The constant velocity model represents motion behavior of an object with almost constant velocity and the object is assumed to have no turn rate, and therefore it is heading to the same bearing on all time stamps. The system function is presented in (5).

$$X_{k+1} = \begin{bmatrix} px_k + v_k T sin(\Psi) \\ py_k + v_k T cos(\Psi) \\ 0 \\ v_k \\ \dot{\Psi} \end{bmatrix} \quad (5)$$

The CTRV motion model is described using two cases. The CTRV process model when the yaw angle is not 0 i.e. the vehicle is performing a turn, is illustrated in (6).

$$X_{k+1} = X_k + \begin{bmatrix} \frac{\mathrm{v_k}}{\dot{\Psi}_k}(\sin(\Psi_k + \dot{\Psi}_k\,\Delta t) - \sin(\Psi_k)) \\ \frac{\mathrm{v_k}}{\dot{\Psi}_k}(-\cos(\Psi_k + \dot{\Psi}_k\,\Delta t) + \sin(\Psi_k)) \\ 0 \\ \dot{\Psi}_k\,\Delta t \\ 0 \end{bmatrix} + \begin{bmatrix} \frac{1}{2}(\Delta t)^2\cos(\Psi_k)\,\gamma_{a,k} \\ \frac{1}{2}(\Delta t)^2\sin(\Psi_k)\,\gamma_{a,k} \\ \Delta t\gamma_{a,k} \\ \frac{1}{2}(\Delta t)^2\gamma_{\ddot{\Psi},k} \\ \Delta t\gamma_{\ddot{\Psi},k} \end{bmatrix} (6)$$

When the yaw angle is 0 the vehicle is moving straight and the process model becomes (7):

$$X_{k+1} = X_k + \begin{bmatrix} v_k\cos(\Psi_k)\Delta t \\ v_k\sin(\Psi_k)\Delta t \\ 0 \\ \dot{\Psi}\Delta t \\ 0 \end{bmatrix} + \begin{bmatrix} \frac{1}{2}(\Delta t)^2\cos(\Psi_k)\,\gamma_{a,k} \\ \frac{1}{2}(\Delta t)^2\sin(\Psi_k)\,\gamma_{a,k} \\ \Delta t\gamma_{a,k} \\ \frac{1}{2}(\Delta t)^2\gamma_{\ddot{\Psi},k} \\ \Delta t\gamma_{\ddot{\Psi},k} \end{bmatrix} \quad (7)$$

The UKF generates a set of sigma points and then propagates them through the non-linear process function. The Gaussian can then be recovered from the newly transformed points.
The first sigma point is the mean (8).

$$X^0_{K|K} = X^*_{K|K} \quad (8)$$

The rest of the points are generated around the mean with a spreading factor of λ

$$X^i_{K|K} = X^*_{K|K} + \sqrt{(\lambda + n_x)P_{(k|k)}} \quad (9)$$

$$X^i_{K|K} = X^*_{K|K} - \sqrt{(\lambda + n_x)P_{(k|k)}} \quad (10)$$

The sigma points are fed through the process function. Before, we had a covariance matrix and generated sigma points, and here we are doing the inverse step, we have predicted sigma points and we want to recover the covariance matrix. So we also want to invert the spreading of the sigma points. This task is performed by using the weights. Weights are calculated as described below in (11) and (12). As it can be seen the weights depend on the spreading parameter lambda

$$w_i = \frac{\lambda}{\lambda + n_a}, i = 0 \quad (11)$$

$$w_i = \frac{1}{2(\lambda + n_a)}, i = 2, \ldots, n_a \quad (12)$$

The mean and covariance are generated using (13) and (14) below.

$$X_{k+1|K} = \sum_{i=1}^{n_\sigma} w_i X_{k+1|k,i} \quad (13)$$

$$P_{k+1|K} = \sum_{i=1}^{n_\sigma} w_i (X_{k+1|k,i} - x_{k+1|k})\,(X_{k+1|k,i} - x_{k+1|k})^T \quad (14)$$

The prediction step for the constant velocity model is simpler. The state and covariance are projected forward using a transition matrix A (15) and (16).

$$X_{k+1} = AX_k \quad (15)$$

$$P_{k+1} = AP_kA + Q \quad (16)$$

The update step is similar in both trackers regardless of the motion model. Since the measurement model is linear (we are receiving x, y coordinates similar to the ones in our state model) we will not have to perform any linearization procedure. We are computing the Kalman gain based on the equation (17) bellow

$$K = P_k H^T (H P_k H^T + R)^{-1} \quad (17)$$

In the equation bellow we are updating the state and covariance based on the measurement readings (18) and (19).

$$X_k = X_k + K(z_k - HX_k) \quad (18)$$

$$P_k = (I - KH)P_k \quad (19)$$

The results of the tracking process when using the CTRV model are depicted in figure 4 bellow.

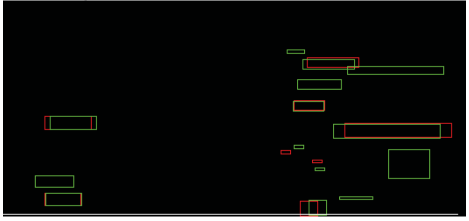

Fig 4. Result of the tracking process

The object predictions are depicted with green rectangles, the object measurements are illustrated with red rectangles.

C. Data Association and combination of tracks having different motion models

Associating and merging the results of predictions from two different motion models allows us to obtain a better overall estimate of how the objects in a scene are moving. In order to merge the results of the two filters we first have to associate them. For this task we are using an approach similar to the two-step association method presented above. Most of the time only two predictions will overlap, however there may be cases when more predictions from one filter fall within the predictions of the second filter. For this reason in this case the validation function of the association will be more restrictive with respect to the object dimensions. In the figure 5 bellow we illustrate the overlapping in a grid of the tracks obtained with the nonlinear tracker with green and the tracks having an underlying linear motion model with red.

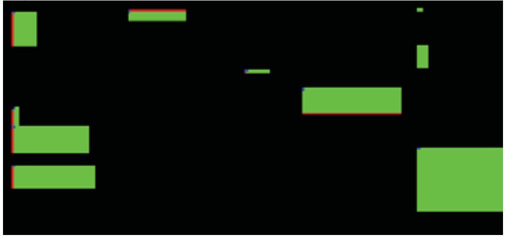

Fig 5. Result of the track to track association process

The tracker manager which handles the tracking contains an area vector similar to the one present in each track. In this case however the area vector is initialized each time we search for a correspondence. Similar to the two step association we are trying to find the maximum overlapping between two tracks. If we identify that two tracks overlap we will consider a validation function (20) that checks the maximum similarity between the two tracks. For space considerations we will denote the following expressions as described below.

$$a{:}\, area(x) - area(y) \le TH_{area}$$

$$b{:}\, x_{lenght} - y_{length} \le TH_{length}$$

$$c{:}\, x_{width} - y_{width} \le TH_{width}$$

$$d{:}\, euclidianDistance(x, y) \le TH_{dist}$$

$$valid(x, y) = \begin{cases} true, a\ AND\ b\ AND\ c\ AND\ d \\ false, otherwise \end{cases} \quad (20)$$

Each two tracks that come from the filters containing different motion models which are associated are stored in a lookup table for fast accessing.

In order to obtain the final position of an object the two tracks are combined taking into consideration the movement orientation of the vehicle. If the vehicle orientation is closer to 0°, 90°, 180°, 270° the track coming from the linear motion model will receive a higher weight and the track obtained using the CTRV model will receive a lower weight and vice versa. The final position is obtained using the weighted sum of the predicted positions (21).

$$x_{final} = w_1 x_{Linear} + w_2 x_{NonLinear}$$

$$y_{final} = w_1 y_{Linear} + w_2 y_{NonLinear} \quad (21)$$

IV. Experimental Results

In this section we will evaluate the results of the proposed solution with respect to the position given by a high precision GPS placed on a tracked target vehicle. The system on which we have tested our method contains an Intel i5-2500 CPU with 3 GHz frequency. The main characteristics of the GPS system which was implemented on a target vehicle are displayed in Table I bellow.

TABLE I. GPS Characteristics

Feature	Value
Standard	RT3003
Positioning	L1, L2
Position accuracy	0.01m
Velocity accuracy	0.05Km/h
Roll/pitch accuracy (1σ)	0. 03°
Heading accuracy (1σ)[2]	0. 1°
Track angle accuracy (1σ)[3]	0. 07°
Slip angle accuracy(1σ)[4]	0. 15°

The characteristics of the 16L Velodyne used to detect the objects are illustrated in Table II bellow.

TABLE II. Velodyne Characteristics

Features
Time of flight distance measurement with calibrated reflective
16 channels
Measurement range up to 100m
Accuracy +/- 3cm
Dual returns
Field of view (vertical): 30° (+15° to -15°)
Angular resolution (vertical): 2°
Field of view (horizontal/azimuth): 360°
Angular resolution (horizontal/azimuth): 0.1° - 0.4°
Rotation rate: 5 - 20 Hz

We will refer to the vehicle on which the GPS is mounted as the target vehicle and the car on which the velodyne is mounted as the reference (or ego) vehicle. We are selecting the nearest neighbor to the target vehicle in order to assess whether the position of the predicted cuboids is correctly identified.

In the chart below presented in figure 6 with a blue color we represent the position of the target vehicle on the x (forward) axis, with gray we represent the measurement obtained for the LIDAR cuboid and with orange we represent the cuboids position obtained from our algorithm. On the horizontal axis we represent the number of frames and the on the vertical axis the distance to the object. We plot only the x axis since the greatest oscillations exist on the axis facing forward.

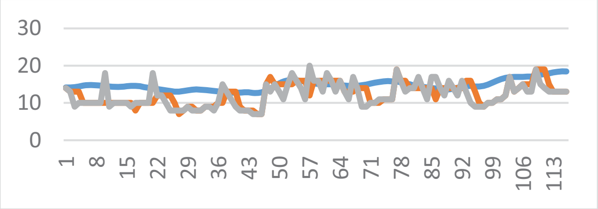

Fig 6. Position of the vehicle over a number of frames

As we can see the original data, depicted with grey is noisier in time. The result of our filtering solution represented with orange is much smoother and closer to the ground truth.
The results of our algorithm on another sequence can be visualized in figure 7. The color code remains the same as in the previous example.

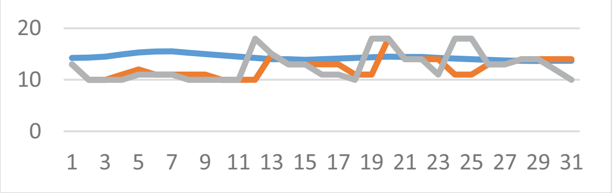

Fig 7. Another sequence depicting the vehicle position over a number of frames

Another way of evaluating our solution is to project the tracked objects in the intensity image, using the corresponding projection matrices, and visually assessing whether the tracked object falls on the target vehicle. In the figure 8 bellow we see that the prediction (with red) overlaps with the object measurement (depicted with blue) and it is very close to the target vehicle colored with green. The box dimensions on the vertical (Z) axis are changed so that in case of perfect overlapping we are still able to visualize where the projection is situated in the image.

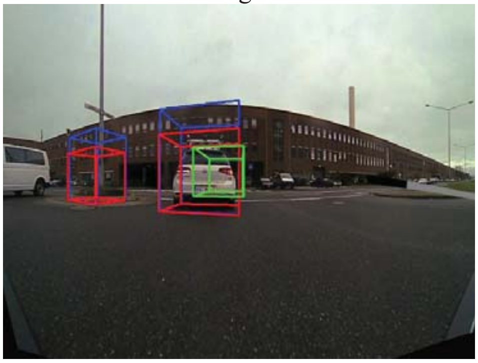

Fig 8. Measurements, predictions and GPS values projected onto the image

Another example from another scene can be observed in the image bellow. In this scene we can also see predictions from objects which are no longer present in the scene. We are keeping a track alive for a number of 10 frames, afterwards if no measurement is associated to it, the tracked object is destroyed.

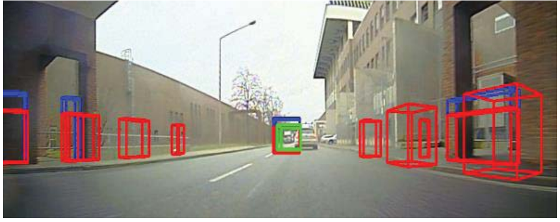

Fig 9. Tracks without measurements are kept for a number of frames

In figure 10 bellow we can see another example with multiple objects. In the left image the 3D LIDAR objects are displayed with blue, in the middle image with pink color we can see the track predictions which overlap perfectly with the measurements, and in the right image the segmentation image, overlapped over the intensity image is depicted.

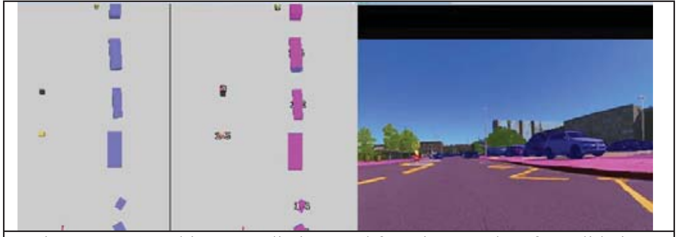

Fig 10. LIDAR objects, predictions and front image taken for validation

The proposed solution is having a running time of 20 fps on the hardware described above.

V. Conclusions And Future Works

In this paper we proposed a real time solution for tracking 3D LIDAR objects in clutter under complex real world scenarios. One of the reasons why target tracking may fail for sparse LIDAR objects is that their position and dimensions vary in each frame. This can happen due to bad point cloud motion correction or bad segmentation. Such unreliable features may lead to false associations and ultimately to poor predictions and an inferior understanding of the environment. Furthermore in an urban situation tracked objects do not move on well-defined patterns, making it difficult for a single motion model to be able to represent the object trajectory. For this reason in this paper we proposed a two-step data association scheme that finds correspondences between objects even if they change in appearance in each frame. The associated objects are tracked using two trackers one having a linear motion model and the other a non-linear model. The two models are able to better capture the dynamic behavior of traffic objects. The efficient association and combination of trackers using the orientation of the vehicle is able to provide better and more reliable estimates for the position of the vehicle.

In future work we will try to further improve the data association process using results from the segmentation image since this information is already available for other high-level processes and using it will not burden the running time of the system. We will also try to implement an efficient way of combining and encapsulating more motion models in order to have a better representation of the object movement in an urban environment.

ACKNOWLEDGMENT

This work was partially supported by the by the UP-Drive project (Automated Urban Parking and Driving), Horizon 2020 EU funded, Grant Agreement Number 688652 and partially supported by a grant of Ministry of Research and Innovation, CNCS - UEFISCDI, project number PN-III-P4-ID-PCE-2016-0727, within PNCDI III.

REFERENCES

[1] Y. Bar-Shalom. Tracking methods in a multitarget environment. Automatic Control, IEEE Transactions on , 23(4):618 – 626, August 1978

[2] Cheng J, Xiang Z, Cao T, et al. Robust vehicle detection using 3D Lidarunder complex urban environment. Robotics and Automation (ICRA), 2014 IEEE International Conference on. IEEE, 2014: 691-696

[3] Shackleton J, VanVoorst B, Hesch J. Tracking People with a 360-degree Lidar. Advanced Video and Signal Based Surveillance (AVSS), 2010 Seventh IEEE International Conference on. IEEE, 2010: 420-426.

[4] Himmelsbach M, Wuensche H. Fast segmentation of 3d point clouds for ground vehicles. Intelligent Vehicles Symposium (IV). IEEE, 2010: 560-565..

[5] Cheng J, Xiang Z, Cao T, et al. Robust vehicle detection using 3D Lidar under complex urban environment. Robotics and Automation (ICRA), 2014 IEEE International Conference on. IEEE, 2014: 691-696.

[6] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer et al. , “Autonomous driving in urban environments: Boss and the urban challenge,” Journal of Field Robotics, vol. 25, no. 8, pp. 425–466, 2008

[7] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? the kitti vision benchmark suite,” in Conference on Computer Vision and Pattern Recognition (CVPR) , 2012

[8] R. E. Kalman, “A new approach to linear filtering and Prediction Problem” J. Basic Eng., vol 82 no. 1, p 35 1960

[9] Y. Bar Shalom and T. E. Fortmann, Tracking and Data Association. Mathematics in Science and Engineering Series, Academic Press 1988

[10] S. Julier and J. Uhlmann, “Unscented Filtering and Nonlinear Estimation” Proc. IEEE vol 92, pp. 401 – 422, mar 2004.

[11] H. Yang, L. Shao, F. Zheng, L. Wang, and Z. Song, “Recent advances and trends in visual tracking: A review,” Neurocomputing, vol. 74, no. 18, pp. 3823–3831, 2011

[12] L. Leal-Taixe, A. Milan, I. Reid, S. Roth, and K. Schindler, “Motchallenge 2015: Towards a benchmark for multi-target tracking,” arXiv preprint arXiv:1504.01942, 2015.

[13] A. Milan, L. Leal-Taixe, I. Reid, S. Roth, and K. Schindler, “Mot16: A benchmark for multi-object tracking,” arXiv preprintarXiv:1603.00831, 2016

[14] A. Azim and O. Aycard, “Layer-based supervised classification ofmoving objects in outdoor dynamic environment using 3d laser scanner,” in Intelligent Vehicles Symposium Proceedings, 2014 IEEE, June 2014, pp. 1408–1414

[15] A. Osep, A. Hermans, F. Engelmann, D. Klostermann, , M. Mathias, and B. Leibe, “Multi-scale object candidates for generic object tracking in street scenes,” in ICRA, 2016

[16] A. Petrovskaya and S. Thrun, “Model based vehicle detection and tracking for autonomous urban driving,” Autonomous Robots, vol. 26, no. 2-3, pp. 123–139, 2009

[17] Asvadi, P. Peixoto, and U. Nunes, “Detection and tracking of moving objects using 2.5 d motion grids,” in Intelligent Transportation Systems (ITSC), 2015 IEEE 18th International Conference on. IEEE, 2015, pp. 788–793.

[18] A. Petrovskaya, M. Perrollaz, L. Oliveira, L. Spinello, R. Triebel, A. Makris, J.-D. Yoder, C. Laugier, U. Nunes, and P. Bessiere, “Awareness of road scene participants for autonomous driving,” in Handbook of Intelligent Vehicles. Springer, 2012, pp. 1383–1432

[19] A. Azim and O. Aycard, “Layer-based supervised classification of moving objects in outdoor dynamic environment using 3d laser scanner,” in Intelligent Vehicles Symposium Proceedings, 2014 IEEE, June 2014, pp. 1408–1414

[20] F. Moosmann and C. Stiller, “Joint self-localization and tracking of generic objects in 3d range data,” in Robotics and Automation (ICRA), 2013 IEEE International Conference on . IEEE, 2013, pp. 1146–1152.

[21] S. Hosseinyalamdary, Y. Balazadegan, and C. Toth, “Tracking 3d moving objects based on gps/imu navigation solution, laser scanner point cloud and gis data,” ISPRS International Journal of Geo Information , vol. 4, no. 3, pp. 1301–1316, 2015.

[22] "Corrections to "A Method for the Synthesis of Controllers to Handle Safety, Liveness, and Real-Time Constraints" and "Tracking in Clutter with Strongest Neighbor Measurements - Part I: Theoretical Analysis"," in *IEEE Transactions on Automatic Control*, vol. 44, no. 4, pp. 886-886, April1999.

[23] F. Sigges and M. Baum, "A nearest neighbour ensemble Kalman Filter for multi-object tracking," *2017 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, Daegu, 2017, pp. 227-232. doi: 10.1109/MFI.2017.8170433

[24] Y. Bar-Shalom, F. Daum and J. Huang, "The probabilistic data association filter," in *IEEE Control Systems*, vol. 29, no. 6, pp. 82-100, Dec. 2009. doi: 10.1109/MCS.2009.934469

[25] A. F. Tchango, V. Thomas, O. Buffet, A. Dutech and F. Flacher, "Tracking multiple interacting targets using a joint probabilistic Data Association filter," *17th International Conference on Information Fusion (FUSION)*, Salamanca, 2014, pp. 1-8.

[26] D. Musicki, R. Evans and S. Stankovic, "Integrated probabilistic data association (IPDA)," *[1992] Proceedings of the 31st IEEE Conference on Decision and Control*, Tucson, AZ, 1992, pp. 3796-3798 vol.4.

[27] S. S. Blackman, "Multiple hypothesis tracking for multiple target tracking," in *IEEE Aerospace and Electronic Systems Magazine*, vol. 19, no. 1, pp. 5-18, Jan. 2004.

[28] M. L. Krieg and D. A. Gray, "Multi-sensor, probabilistic multi-hypothesis tracking," *Proceeding of 1st Australian Data Fusion Symposium*, Adelaide, SA, 1996, pp. 153-158.

[29] Yong-shik Kim, Keum-shik Hong,Yong-shik Kim, Keum-shik Hong, An IMM algorithm for tracking maneuvering vehicles in an adaptive cruise control environment, International Journal of Control, Automation, and Systems, pp 310—318

[30] Xuezhi Wang, S. Challa and R. Evans, "Gating techniques for maneuvering target tracking in clutter," in *IEEE Transactions on Aerospace and Electronic Systems*, vol. 38, no. 3, pp. 1087-1097, Jul 2002.