Detection, Classification and Tracking of Moving Objects in a 3D Environment

Asma Azim and Olivier Aycard

Abstract— **In this paper, we present a framework based on 3D range data to solve the problem of simultaneous localization and mapping (SLAM) with detection and tracking of moving objects (DATMO) in dynamic environments. The basic idea is to use an octree based Occupancy Grid representation to model the dynamic environment surrounding the vehicle and to detect moving objects based on inconsistencies between scans. The proposed method for the discrimination between moving and stationary objects without a priori knowledge of the targets is the main contribution of this paper. Moreover, the detected moving objects are classified and tracked using Global Nearest Neighbor (GNN) technique. The proposed method can be used in conjunction with any type of range sensors however we have demonstrated it using the data acquired from a Velodyne HDL-64E LIDAR sensor. The merit of our approach is that it allows for an efficient three dimensional representation of a dynamic environment with a minimum memory consumption, keeping in view the enormous amount of information provided by 3D range sensors.**

I. INTRODUCTION

The reliable perception of the surrounding environment is a very important step for an intelligent vehicle. It is usually divided into two subtasks: simultaneous localization and mapping (SLAM) and detection and tracking of moving objects (DATMO). The purpose of SLAM is to provide the vehicle with a map consisting of static entities of the environment while DATMO uses that map to detect and track dynamic entities.

During the last few years, the advent of affordable rangefinders has attracted a great amount of research in the area of environment perception. Both SLAM and DATMO are being nearly exclusively performed nowadays based on the data acquired by 2D range sensors. These sensors scan the environment along a plane within a limited viewing angle thus the objects above or below this plane cannot be detected. Recently, three-dimensional range scanners have been commercially introduced which provide 3D point cloud instead of 2D slice of the environment. Although there has been an overwhelming amount of work on perception in 2D and 2.5D but the problem of perception in 3D has been addressed by comparatively fewer researchers yet. One of its main reasons is the enormous amount of data provided by 3D sensors. The amount of data in a single scan of 3D sensor is usually several times larger than that of a 2D scan. Fig. 1 is a typical illustration of this. It is showing the point cloud generated by a single scan of the Velodyne[1] HDL 64E lidar which consists of approximately 100,000 points. Therefore, the processing of this data requires efficient algorithms and data structures. In addition, the extraction and interpretation of geometry of 3D range data is very complicated in comparison to 2D. Another problem related to 3D laser data is that the lower layers of the scanner usually perceive the ground or floor (Fig. 1) which makes the interpretation of data a lot more complex. This is not usually an issue with 2D laser data other than the rare case of being on a steep slope.

Asma Azim and Olivier Aycard are with University of Grenoble1 & Laboratoire d'Informatique de Grenoble, France, (e-mails: Asma.Azim@imag.fr, Olivier.Aycard@imag.fr)

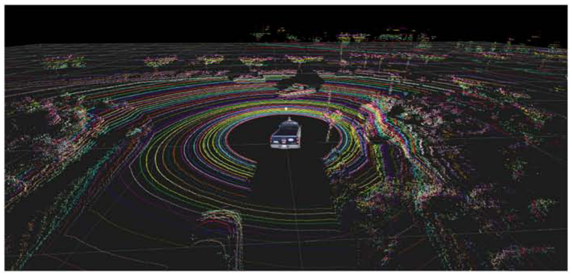

Fig. 1. An illustration of the raw data provided by the high definition lidar sensor of Velodyne.

A number of approaches are available in literature for solving SLAM and DATMO problem using 2D laser scanners[4][13][12]. Most of these use the grid maps, specially a well-established mapping technique of occupancy grid algorithm [2], to represent the environment. The noise in the perception of the sensor together with the size of the environment to be mapped makes mapping a hard problem. Mapping of 3D outdoor environments makes this problem even harder by increasing the size of the grids tremendously. It was demonstrated in some early works such as [6] and [10] which proposed to model the environment using rigid grid of cubic volumes of equal size (voxels). These implementations are not tractable in realtime and cannot be used for DATMO. An optimized solution using multi-level surface maps was proposed in [11] for outdoor environment mapping and detection of moving objects. It implements an efficient data structure using a 2D projection of the 3D space for outdoor mapping. This means that the useful detail of 3D input space is compromised by reducing it to a 2D map used for representing the environment and detecting motion.

Another classical approach is to perform mapping using point clouds to avoid the discretization of the environment. It is successfully used in [1] and [7] for 3D mapping of

[1]www.velodyne.com/lidar

outdoor environment. Despite being a good representation, it only provides information about the occupied areas and not the free or unknown areas. Therefore it cannot be used for detection and tracking of moving objects. Previously, the moving object detection problem has been solved by addition of vision sensors [14] but visual classification does not help to distinguish moving vehicles from stationary.

An optimization of the 3D grid is possible by using a tree-based approach such as octrees. An octree is a tree data structure in which each internal node has exactly eight children. Octrees are used to partition a three dimensional space by recursively subdividing it into eight octants. It was first used in [5] for modeling. Recently, it has been used in several works [3][8][15] for adapting occupancy grid mapping from 2D to 3D but, to our knowledge, it has never been used for detection and tracking of moving objects before this work. In this work, we have used the octree-based 3D mapping approach described in [15] for mapping of the environment.

This paper presents a framework based on 3D range data to solve the problem of SLAM and DATMO in dynamic environments. The basic idea is to use an octree based Occupancy Grid representation to model the dynamic environment surrounding the vehicle and to detect moving objects based on inconsistencies between scans. The major contribution of this work is the discrimination between moving and stationary objects without a priori knowledge of the targets. Second, detected moving objects are classified and tracked using Global Nearest Neighbor (GNN) [9] technique. The proposed method is not restricted to a particular sensor and it can be used in conjunction with any type of range sensors however we have demonstrated it using the data acquired from a Velodyne HDL-64E LIDAR. The merit of our approach is that it allows for an efficient three dimensional representation of a dynamic environment with a minimum memory consumption, keeping in view the enormous amount of information provided by 3D range sensors.

This paper is organized as follows. In the next section, we summarize the technique that we have used for three dimensional mapping of the environment. In section III, we detail our approach for detection, classification and tracking of moving objects. Experimental results are given in section IV. Section V concludes this paper and provides a perspective for future research in this area.

II. SIMULTANEOUS LOCALIZATION AND MAPPING

The main contribution of this work lies in the detection, classification and tracking of moving objects (DATMO). For a reliable and efficient DATMO, a good and efficient representation of the environment is extremely necessary. For 3D mapping of the environment, we have used the octree based 3D occupancy grid approach presented in [15]. The main attraction of this approach is to represent full 3D maps in which free and unknown areas are explicitly modeled. The tree-based implementation provides maximum flexibility regarding the area and resolution of the map. The occupancy is estimated probabilistically which copes with the sensor noise. The localization of the vehicle is performed by an integrated navigation system (INS) consisting of GPS, wheel speed sensors and inertial measurement unit (IMU). This unit provides the 6D pose, denoting the rear axis of the vehicle, which is then used for calculating the tranformations between the scans.

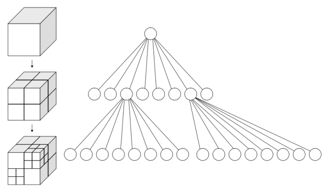

Fig. 2. Recursive subdivision of a cube into octants and the corresponding octree.

An octree is a tree data structure in which each internal node has exactly eight children. Octrees are used to partition a three dimensional space by recursively subdividing it into eight octants or cubic volumes (called voxels), as shown in Fig. 2. This subdivision continues until a desired minimum voxel size is reached which determines the resolution of the octree. Being a hierarchical data structure, the tree can be cut at any level to obtain a coarser subdivision. Here, the octree is used to model occupancy of a volume based on the sensor measurements. If a certain volume is measured as occupied, the corresponding node in the octree is initialized. Any uninitialized node could possibly be free or unknown in this boolean setting. To resolve this ambiguity, free voxels are explicitly represented as free nodes in the tree.

The probability $P(n|z_{1:t})$ of a leaf node n being occupied given the sensor measurements $z_{1:t}$ is estimated according to the following equation:

$$P(n|z_{1:t}) = \left[1+\frac{1-P(n|z_t)}{P(n|z_t)}\frac{1-P(n|z_{1:t-1})}{P(n|z_{1:t-1})}\frac{P(n)}{1-P(n)}\right]^{-1} \tag{1}$$

Assuming a uniform prior distribution ($P(n) = 0.5$) and by using the $logOdds(L)$ notation, Eq. 1 can be simplified to the following equation:

$$L(n|z_{1:t}) = L(n|z_{1:t-1}) + L(n|z_t) \tag{2}$$

This formulation allows for faster updates in case of pre-computed sensor models. Here $L(n|z_t)$ is the inverse sensor model which is specific to the sensor used for mapping. In case of the laser sensor, a beam-based inverse sensor model is employed. A ray-tracing operation is performed to efficiently calculate the volumes to be updated using the following model:

$$L(n|z_t) = \begin{cases} l_{occ} & \text{, if ray is reflected within volume} \\ l_{free} & \text{, if ray traversed the volume} \end{cases} \tag{3}$$

Furthermore, the *clamping update policy* proposed in [16] is used to ensure that the confidence in the map remains bounded. Thus, using the upper and lower bound of l_{max} and l_{min}, the $logOdds(L)$ is calculated by:

$$L(n|z_{1:t}) = \max(\min(L(n|z_{1:t-1}) + L(n|z_t), l_{max}), l_{min}) \quad (4)$$

Whenever the $logOdds$ value of a node reaches either the threshold $lmin$ or $lmax$, it is considered to be stable, and thus measured free or occupied with high confidence. If all children of a node are stable leafs with the same occupancy state, then they can be pruned leading to a considerable reduction in the total number of nodes.

III. DETECTION, CLASSIFICATION AND TRACKING OF MOVING OBJECTS

In this section, we describe our approach for the detection of moving objects from the octree map presented in previous section. We also summarize the method used for classification and tracking of the detected moving objects.

A. Detection of Moving Objects

After construction of a consistent local map of the environment, moving objects can be detected when new measurements arrive. The principal idea of our appraoch for the hypothesis of a moving object is based on the inconsistencies between observed free space and occupied space in the local grid map. This method borrows idea from background-subtraction methods in computer vision. Our process for the detection of moving objects is carried out in two steps: detection of dynamic voxels and their segmentation into individual dynamic objects.

The first step is to detect the voxels that might be containing measurements obtained from dynamic objects. This can be considered as a background modeling process. In this step, we first construct a 3D occupancy grid map incrementally from laser measurements as explained in previous section. Based on the constructed grid map, we are able to make a hypothesis about the voxels of the grid occupied by moving objects when new measurements arrive. For this, we maintain a list of the voxles whose states are inconsistent between the current and previous scan.

Let S_{t-1} and S_t be the states of a voxel in previous scan and current scan respectively. If the transition between these two states for a specific voxel of the grid is such that $S_{t-1} = free$ and $S_t = occupied$ then this is the case when an object is detected on a location previously seen as $free$ space and it is possibly a moving object. We add it to the list of possible dynamic voxels. In contrary, if $S_{t-1} = occupied$ and $S_t = free$, it means that the location which was previously being observed as $occupied$ is $free$ now. This can possibly be caused by a missed detection by the sensor or it was a voxel occupied by a dynamic object which may have displaced now. We search this voxel in our list of dynamic voxels maintained from previous scans. If it is found, we wait for the next few scans instead of removing it from the dynamic voxels list immediately. If it is observed as $free$ in next scans as well, then we delete it from the list.

If $S_{t-1} = occupied$ and $S_t = occupied$, it means that an object is observed on a location previously $occupied$ then it probably is static. If an object appears at a location which was previously unobserved, then we can say nothing about that object. For such measurements, a priori we will suppose that they are static until latter evidences come. As a result of this step, all the inconsistencies between the two measurements are identified as dynamic voxels. These include a large number of sparsely situated voxels generated as a noise. This issue is resolved in the subsequent processing.

Once we have maintained the list of all possible dynamic voxels, the next step consists of the segmentation of detected dynamic measurements into regions. It is carried out by clustering these dynamic voxels into seperate groups where each group represents a single object. The criterion used for deciding whether the voxels belong to the same cluster is the Euclidean distance between their centers.

We can intuitively expect that all voxels belonging to a specific cluster are neighboring or atleast spatially very close to each other. Thus, we do not require to exhaustively compare the voxels pairwise to check whether they belong to a cluster or not. The clustering can be performed using an approach similar to a region-growing algorithm. In this approach, we examine the neighboring voxels of initial "seed point" voxel and determine whether the neighbors should be added to the region or not.

As described above, all possible dynamic voxels are stored in a data list. Our clustering algorithm starts with stepping through this list. A voxel is defined by the position of its center and the length of its side. If the current voxel in the list is not yet assigned to any cluster, a new cluster is initialized. We find the set $Neighbor(v)$ of its neighboring voxels from the list. As criterion for adding a voxel to the cluster, we use the Euclidean distance between the center of the current voxel and the voxel in consideration. If this criterion is satisfied by the current voxel then it is added to the cluster. Now, we use this newly added voxel further and continue the search within its neighborhood in a recursive manner.

At the end, we apply a density threshold on the cluster to check if it has sufficient number of voxels to be identified as a moving object. A cluster which has the number of voxels fewer than this threshold is ignored and its corresponding voxels are removed from the list. In this way, we get rid of the false alarms and spurious elements in the environment which were wrongly identified as dynamic object voxels. The remaining dynamic voxels in the list have a higher probabliity of corresponding to moving objects which is further improved in the next section of classification. This reduces the number of detected voxels by a large amount.

Fig. 3 is an illustration to explain the two steps described above for the detection of moving objects. The leftmost image gives the point cloud corresponding to a situation where the vehicle is moving on the road having dynamic objects around it. The local static map of the environment is incrementally constructed from these point clouds after each

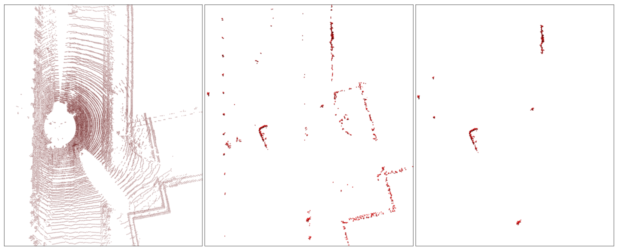

Fig. 3. An illustration of detection of moving objects from inconsistencies between scans and the results of clustering. See text for more details.

scan. The image in the center shows the voxels for dynamic object hypothesis detected on the base of the inconsistencies between the scans displayed in red. Here you can see the amount of noise detected as the sparse red points. The image on the right is the result of clustering of the dynamic voxels. Applying an appropriate threshold has eliminated quite a large number of wrongly detected dynamic voxels. But still, there remain some clusters which do no belong to dynamic objects. Those will be handled in the next step.

After detection of moving objects, the next step is to track them in order to estimate their states and predict their behaviors in the future. Tracking multiple moving objects is a classical problem in 2D. In the general case this problem is very hard, however it has been shown experimentally that simple methods are good enough to cope with urban scenarios. Classification of the dynamic objects has also proved to play a great role in improving the results of tracking. In our current work, we have initiated with the implementation of a simple object classification technique based on bounding box along with an object tracking scheme using Global Nearest Neighborhood (GNN) data association and Kalman filter to track detected objects in 3D.

B. *Classification of Moving Objects*

The classification of moving objects (specifically pedestrians, bicycles, vehicles etc) is of utmost importance for an intelligent vehicle. In recent years, considerable progress has been made in this field through supervised classification methods. A variety of approaches have been developed in this context but they require a sufficiently large number of training examples to ensure good performance in the test phase. For now, we have used only a simple technique for the classification of moving objects based on the bounding box.

The idea is to identify and track a cluster of dynamic cells provided by the previous step and fix a 3D bounding box to it. This bounding box is calculated for connected components of moving objects in 3D space. The classification is done based on the properties of these boxes such as the ratio between their length, width and height. A benefit of this method is that it is highly time efficient as compared to any complex classification technique and it works well for small numbers of moving objects. Its shortcoming is that the problem of occlusion cannot be solved properly with this approach in cluttered situations. Grouped regions for spatially close objects will form a combined blob. As a result, they will be wrongly identified as a single object and cause tracking errors. However, the results of tracking are still improved by using this simple classification technique which promises the potential for the feature-based classification.

As demonstrated in Fig. 4, the classes of the objects are identified on the base of their respective bounding boxes, which are further used to track these objects. Each cluster created in the previous step is used as a hypothesis for a dynamic object and to calculate the bounding box corresponding to it. Classification is then performed on the basis of the properties of each bounding box. A simple case, that we have experimented with, is the use of ratio between length, width and height of the bounding box as an indicator for its class. We have defined these parameters for the classes of car, bus/truck, bicycle/motorbike and pedestrian. For this step, we start with the results of clustering obtained from the previous step, as shown in the leftmost image of Fig. 4. The image in the center shows ths bounding boxes calculated for each cluster. The four small objects shown in this image do not fall in any of the defined classes therefore they are removed. The bounding box classification has identified two moving objects in the scene, namely a car and a bike or motorbike. The object which was identified as a motorbike did not appear in the next scans therefore it was assumed to be a wrong detection. It was discarded with the help of tracking.

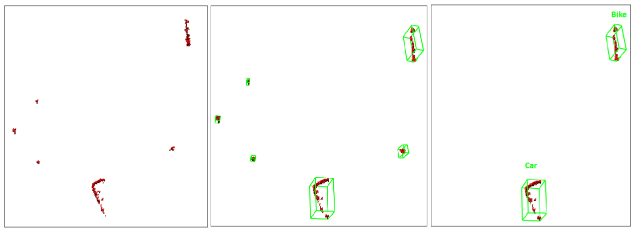

Fig. 4. An illustration of bounding-box based classification and tracking of moving objects. See text for more details.

C. *Tracking of Moving Objects*

The detected dynamic objects need to be tracked. The tracking of multiple moving objects is a complex problem which is generally divided into two parts: filtering and data association. Filtering is the sequential estimation of the state of a dynamic object. It is usually performed using Bayesian filters which require a specific motion model for tracked objects to predict their positions in environment. After predicting the positions of existing tracks, next we perform data association to assign the observations to the existing tracks. The first step in data association is to perform gating to find the likely observations to be associated with the existing tracks. We have used the fastest and simplest technique for data association called Global Nearest Neighbor (GNN). An important optimization that we have achieved in this step is related to the classification information provided by the previous step. While finding possible nearest neighbor association between a track and an observation, we ignore all the associations which consist of objects from different classes. For instance, if the nearest neighbor of an observation classified as a pedestrian is found to be a track corresponding to a vehicle, we ignore that and search for the next nearest neighbor within the specified gate.

After applying GNN, the observations that are not associated to any of the existing tracks can possibly be the potential candidates for new tracks or the false alarms. We create a new track for each of these unused observations but those tracks are not confirmed yet. If those newly created tracks get associated to the observations for next consecutive scans then those are marked as confirmed object tracks. In the other case, those observations are assumed to be false alarms and are suppressed subsequently. Similar to the track creation mechanism, a track deletion or suppression mechanism is also implemented in our technique. If at any stage, we find no observation associated to a track, we mark it as *not observed*. We predict the position of the object from its previous observations. This predicted position is considered to be the current position of that object. This deals with the cases when the object is temporarily occluded. If the object is not observed for next consecutive scans, we assume that it has moved out of the scanning area and its track is deleted.

IV. EXPERIMENTAL RESULTS

To verify the analysis and efficiency of the proposed method, it was tested on the real data acquired by a Velodyne HDL-64 High Definition Lidar scanner mounted on top of an experimental vehicle. It consists of a column of 64 single lasers, covering a pitch range of approximately 26 degrees. It rotates at a rate of 10 Hz sweeping the complete horizontal ground plane and producing 180000 points per turn. We have evaluated the proposed algorithm for different scans acquired by this sensor in inner city traffic scenes. There is no ground truth information available yet, therefore we have performed a qualitative evaluation of the performance.

The results of detection of dynamic objects with their classification and tracking are shown in Fig. 5. This is a relatively complex scenario with the vehicle in a crowded urban environment having a number of moving objects around it. The amount of the noise generated by the sensor is also very large. The two images in the second row show that the detection and clustering steps have done a segmentation of the dynamic objects good enough to be used in the next steps of classification and tracking.

The results of classification and tracking are demonstrated in the two images at the bottom of Fig. 5. The classes of the objects are identified on the base of their respective bounding boxes. Each cluster created in the previous step is used to caluculate the bounding box corresponding to it. Then the ratio between the length, width and height of each bounding box is used to perform the classification of the objects. For instance, if the bounding box of an object has the length very large as compared to its width, then it is less likely to be a vehicle and more likely to be a bicycle or motorbike. Similary, for a pedestrian, both width and legth of the bounding box should be very small as compared to the height.

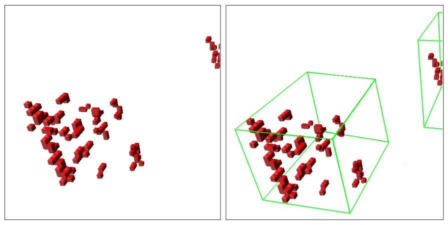
Fig. 6. A group of pedestrians is detected as a single object hence it cannot be classified correctly using the bouding box based classification.

The results obtained from the application of the proposed technique on the real data collected from an urban road scenario show that the moving objects are clearly and efficiently segmented from the environment despite a considerable amount of noise. The spurious elements are further minimized by use of classification and tracking. This results in an efficient and reliable representation of the dynamic environment in 3D. More results and videos can be found at *http://membres-liglab.imag.fr/aycard/html/Demos/iv2012/*.

V. CONCLUSION AND FUTURE WORK

We have presented an approach capable of performing detection, classification and tracking of moving objects from 3D range data. Experimental results have shown that our system can successfully perform the moving object tracking from a vehicle in different dynamic outdoor scenarios. The proposed approach uses an octree based occupancy grid mapping of the environment in 3D. After a map is built for each scan, moving objects are detected from the inconsistencies between this map and that of the previous scan.

For this work, we have performed a naive classification of the moving objects using bounding box. It is a time efficient technique with good performance in less cluttered situations but it has its limitations in cluttered environments. For instance, it cannot identify a group of pedestrians walking together as illustrated in Fig. 6. The detection and clustering step gives a single cluster corresponding to such a group due to the spatial proximity of the pedestrians. Thus calculating the bounding box results in an undefined class. Using a machine learning technique for feature-based classification can prove better in such scenarios. Therefore, the future works include augmentation of our approach by using such a technique for classification alongwith a more robust data association technique for tracking of multiple objects in cluttered environments.

REFERENCES

[1] D. M. Cole and P. M. Newman. Using laser range data for 3d slam in outdoor environments. In *IEEE International Conference on Robotics and Automation*, pages 1556–1563, 2006.

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

[3] J. Fournier, B. Ricard, and D. Laurendeau. Mapping and exploration of complex environments using persistent 3d model. In *Proceedings of the Fourth Canadian Conference on Computer and Robot Vision*, pages 403–410, Washington, DC, USA, 2007. IEEE Computer Society.

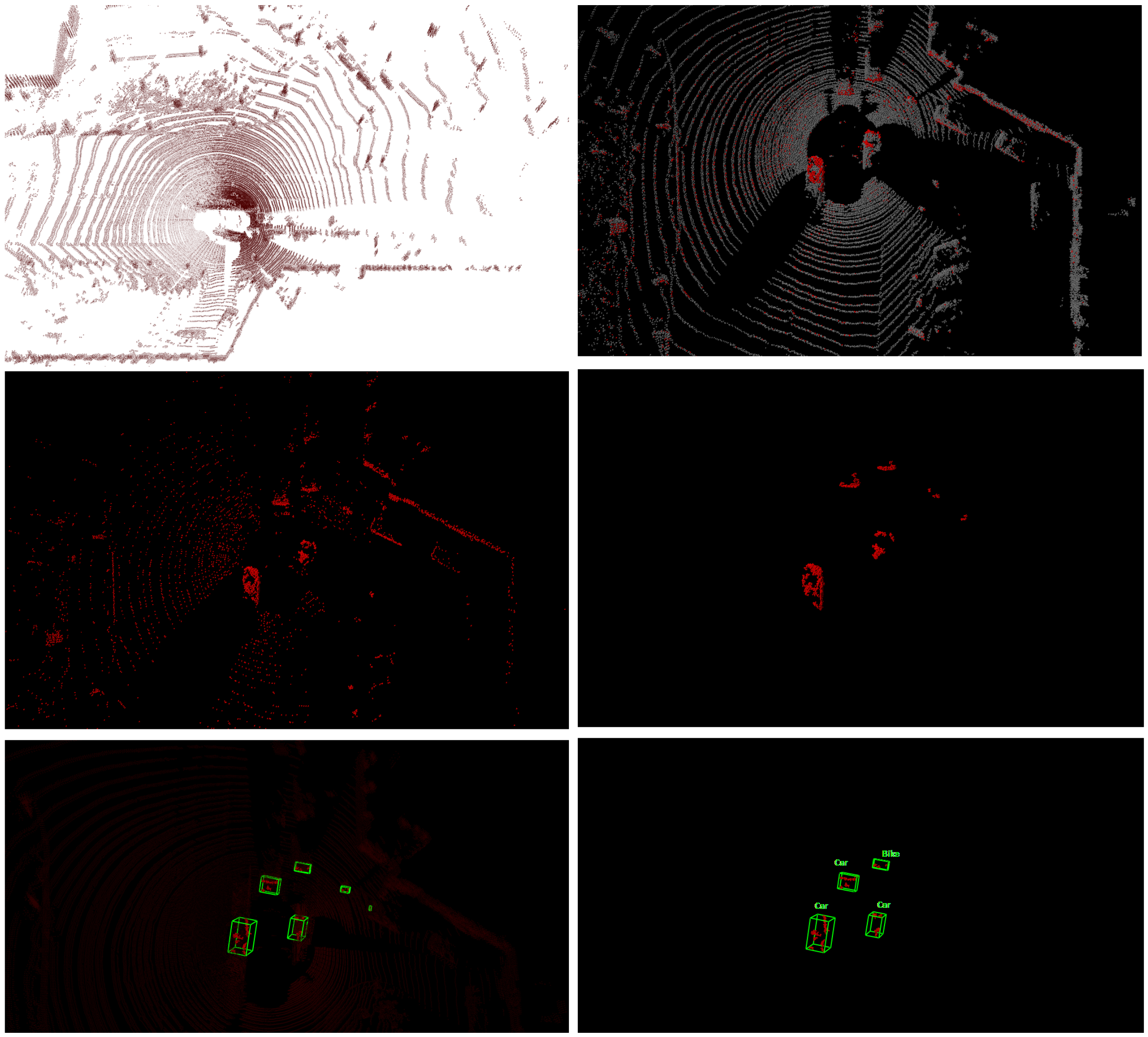

Fig. 5. Experimental results showing the detection, classification and tracking of moving objects, best viewed in color. (First row) The point cloud corresponding to a scan obtained from an urban road scene with multiple moving objects (left) and the occupancy grid corresponding to this scenario showing the static occupied voxels in grey and those detected as dynamic in red (right). (Second row) Results of detection; all the dynamic voxels detected from the inconsistencies between scans (left) and the dynamic voxels remained after clustering and application of threshold (right). (Third row) Results of classification and tracking; calculation of the bounding box corresponding to each cluster (left) and application of classification and tracking resulting in four dynamic objects identified (right).

[4] D. Hähnel, R. Triebel, W. Burgard, and S. Thrun. Map building with mobile robots in dynamic environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2003.

[5] D. Meagher. Geometric modeling using octree encoding. *Computer Graphics and Image Processing*, 19(2):129–147, June 1982.

[6] H. P. Moravec. Robot spatial perception by stereoscopic vision and 3d evidence grids. Technical report, 1996.

[7] A. Nüchter, K. Lingemann, J. Hertzberg, and H. Surmann. 6d slam-3d mapping outdoor environments: Research articles. *Journal of Field Robotics*, 24:699–722, August 2007.

[8] K. Pathak, A. Birk, and S. Schwertfeger. 3d forward sensor modeling and application to occupancy grid based sensor fusion. In *International Conference on Intelligent Robots and Systems*, 2007.

[9] Donald B. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, 24:843–854, 1979.

[10] Y. Roth-Tabak and R. Jain. Building an environment model using depth information. *Computer*, 22:85–90, June 1989.

[11] R. Triebel, P. Pfaff, and W. Burgard. Multi-level surface maps for outdoor terrain mapping and loop closing. In *International Conference on Intelligent Robots and Systems*, 2006.

[12] T.D. Vu, J. Burlet, and O. Aycard. Grid-based localization and local mapping with moving object detection and tracking. *Information Fusion*, 12:58–69, January 2011.

[13] C. C. Wang. *Simultaneous localization, mapping and moving object tracking*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 2004.

[14] S. Wender and K. Dietmayer. 3d vehicle detection using a laser scanner and a video camera. *Intelligent Transport Systems, IET*, 2(2):105 –112, june 2008.

[15] K. M. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, and W. Burgard. OctoMap: A probabilistic, flexible, and compact 3D map representation for robotic systems. In *Proceedings of the ICRA 2010 Workshop on Best Practice in 3D Perception and Modeling for Mobile Manipulation*, Anchorage, AK, USA, May 2010.

[16] M. Yguel, O. Aycard, and C. Laugier. Update policy of dense maps: Efficient algorithms and sparse representation. In *Field and Service Robotics*, volume 42, pages 23–33, 2007.