Fusing Lidar and Semantic Image Information in Octree Maps

Julie Stephany Berrio [1], James Ward [1], Stewart Worrall [1], Wei Zhou[1], Eduardo Nebot[1]

***Abstract*— Current autonomous driving applications require not only the occupancy information of the close environment but also reactive maps to represent dynamic surroundings. There is also benefit from incorporating semantic classification into the map to assist the path planning in changing scenarios. This paper presents an approach to building a multi-label semantic 3D octree map based on the Octomap mapping framework. Current state-of-the-art point cloud classification methods such as conditional random fields (CRF), random forest (RF) and support vector machines (SVM) use dense point clouds in order to train the model that assign a label to each point. This work utilizes images from a convolutional neural network to provide the semantic context of the local environment and projects the classification into a 3D lidar point cloud. The resulting point cloud feeds into the octree map building algorithm and computes the corresponding probabilities (occupancy and classification) for every 3D voxel. We also propose a method to incorporate the uncertainty of semantic labels based on the pixel distance to the label boundaries. The algorithm is validated using data collected by our mobile vehicle platform driving around the University of Sydney campus.**

构建multi-label semantic 3D octree map

这里列出了当前利用在 point cloud classification 的传统的machine learning的方法
1. CRF
2. RF
3. SVM

I. INTRODUCTION

Autonomous vehicles need more detailed information (e.g. street names and width, the location of traffic lights, tolls, and speed cameras) than traditional maps to localize with high accuracy and to operate reliably and efficiently in an environment shared with other vehicles. A more descriptive map is required, which should provide accurate positions and velocity of all mobile agents in proximity, position of poles, buildings, drivable and undrivable roads, and other components that form the vehicle's surroundings. These comprehensive maps are essential for path planning, driving decision making and vehicle control.

The ability to classify the elements of the local environment relies on the perception capabilities of the vehicle sensor system. Cameras have been extensively used for object classification and scene understanding due to their low cost and high information content [3]. Light Detection and Ranging (lidar) has also been used as a cost-effective and reliable tool for representing the urban environment [4]. Laser scanners have shown a promising capability in applications of intelligent transportation systems (ITS) with a large number of experimental deployments around the world, such as Google's Waymo [22], Uber's Otto, NuTonomy, WEpods, Sohja, Tesla, Volvo, etc.

传感器融合的好处

Sensor fusion approaches make it possible to overcome the inherent limitations of each individual sensing modality and at the same time exploit their best capabilities. The integration of laser range-finding technologies with existing vision systems enable a more comprehensive understanding of the 3D structure of the environment [7]. The level of abstraction in which sensor fusion is performed in this paper consists of using images of the environment to classify categories, designing a method to associate uncertainty to each classified label, and then projecting the labels with estimated uncertainty into a point cloud. This process can enable classification, clustering and positioning of relevant features in proximity. Our algorithm can also be optimized by providing high-precision sensor calibration, with intrinsic and extrinsic camera parameters being essential for successful projections.

The highly-developed Convolutional Neural Networks (CNNs) [6] for image-interpretation tasks are made possible due to the availability of highly parallelized network

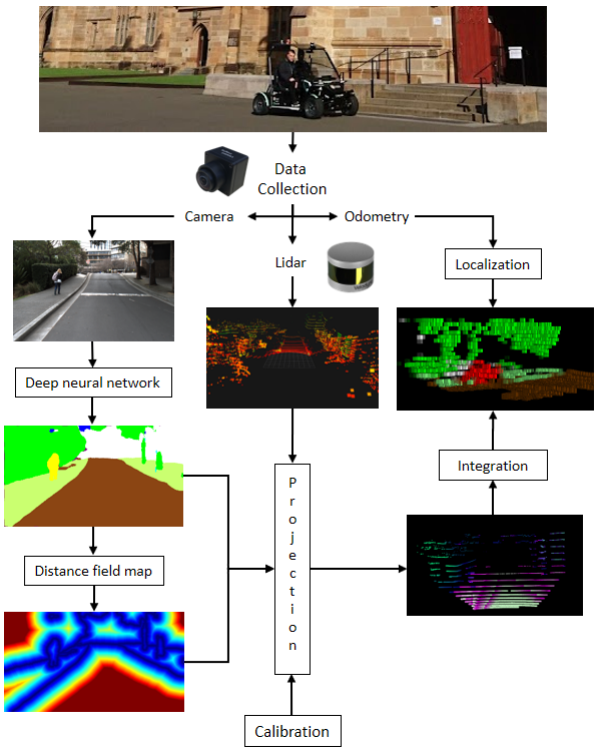

Fig. 1. Flow chart of the processing pipeline. Top image is our electrical vehicle for data collection and algorithm implementation. Data are collected by cameras and lidar. After semantic segmentation and uncertainty processing, they are fed into the Octomap algorithm to build a semantic map for our local environment.

[1]J. Berrio, J. Ward, S. Worrall, W. Zhou, E. Nebot are with the Australian Centre for Field Robotics (ACFR) at the University of Sydney (NSW, Australia). E-mails: j.berrio, j.ward, s.worrall, w.zhou, e.nebot at acfr.usyd.edu.au

architectures that facilitate training from millions of images on general purpose graphics processing units (GPUs), and the availability of vast public benchmark data sets [2]. The computational requirement for training is currently facilitated by the availability of pre-trained models and the transferred knowledge to adapt to different scenarios. This work is based on the results obtained from [5], where semantic segmentation model was trained with public datasets and fine-tuned with a small amount of locally annotated images to improve the performance for our local environment.

In this paper, we introduce a methodology to build a semantic octree map integrating the information of semantically labeled images, 16-beam lidar point cloud and odometry. The captured images are processed by a CNN model to produce an output image with class index for each point. Since all input/sensor measurements are affected by uncertainty, the labeled pixels near the boundary of each class tend to be less accurate than pixels localed close to the center. This motivates us to perform a post processing step and associate uncertainty to each label depending on the pixel's distance to the class boundary. After this process, the labeled image and its uncertainty are projected into the point cloud. The last step is to feed this semantic point cloud and odometry information to the semantic mapping algorithm which adopts an efficient 3D grid representation called Octomap [1]. The ultimate output is an Octree map with all its voxels labeled and associated with corresponding class probabilities. A flowchart of this methodology is shown in Fig. 1.

The paper is organized as follows: in Section 2, we explore the related work for point cloud classification and map representation. In Section 3, we describe the procedure of lidar-camera-odometry fusion and the derivation of per-voxel probabilities. The experiments and results are depicted in Section 4. We draw our conclusion and outlook to future work at the end.

II. RELATED WORK

Two distinct tasks have to be accomplished in order to build a semantic map. The first one is to select a method for semantic classification. The second one is to adopt an appropriate data structure to represent the map. This structure should be flexible to be used for autonomous operations and to allow the synthesis of the semantic labels [8].

Schnabel et al. [16] decompose the point cloud into a concise, hybrid structure of inherent shapes and a set of remaining points. Lafarge and Mallet showed in [17] an algorithm that from point clouds reconstructs simultaneously buildings, trees and topologically complex grounds with geometric 3D-primitives such as planes, cylinders, spheres or cones describing regular roof sections, and irregular roof components.

There are several approaches to representing 3D map as point clouds, voxel grids, octrees, surfels, Gaussian process, but not all satisfy all the requirements of being memory-efficient, allowing multi-resolution and being able to integrate semantic labels [8]. An efficient probabilistic 3D mapping framework based on octrees was developed in [1], this open source framework generates volumetric 3D environment models.

Semantic segmentation divides the image into semantically meaningful components, and categorizes each part into one of the pre-determined labels. Munoz et al. [11] present a Max-Margin Markov Network (M^3Ns) for contextual classification of 3D point clouds or images, by adapting a functional gradient approach in order to learn high-dimensional parameters of random fields to perform discrete, multi-label classification. [12] presented a system to recognize objects in 3D point clouds of urban environments, using hierarchical clustering to localize objects and a graph-cut algorithm to segment points. A feature vector is created to finally label the object using a Bayesian classifier.

[13] exploited the fact that morphological features can be retrieved from the echoes composing the lidar's waveforms. The authors investigated the potential of full-waveform data through the automatic classification (support vector machine) of urban areas using a set of labels to describe building, ground, and vegetation points. In [10] a method is presented to automatically convert the raw 3D point into a compact, semantically rich information model. In [8], Lang et al. presented a 3D semantic outdoor mapping system with multi-label and resolution octree map, using conditional random fields (CRF) to classify point clouds.

Sengupta et al. presented in [24] an algorithm for dense 3D reconstruction with associated semantic labellings by using stereo camera, based on truncated signed distance function (TSDF) and CRF. An approach to labeling objects in 3D scenes is introduced in [14], the authors developed the Hierarchical Matching Pursuit for 3D (HMP3D) which is a hierarchical sparse coding technique for learning features from 3D point cloud data. [15] a 3D point cloud labeling scheme based on 3D CNN is introduced.

This is in contrast to our approach, where the semantic information is extracted from the output of a CNN model, then the classification is projected into the point cloud. This approach is suitable for a platform with lidars and cameras with an overlapped field of view (FOV) within an environment with dynamic objects.

III. SEMANTIC 3D OCTREE MAPS

This section describes the proposed approach to building a semantic 3D octree map. We first explain the labeling process of the point cloud given an image. Then this labeled point cloud is the input for the map building algorithm. The map structures is based on the Octomap approach [1], which has been modified in order to incorporate the probabilistic representation of the labeling process to the voxels, as explained in the next subsection.

A. Semantic labeled point cloud

A CNN for pixel-level semantic segmentation presented in [5] was used in this paper. The network was trained with public Cityscapes dataset [20] and fine-tuned with locally annotated USYD dataset. The classes in Cityscapes

Fig. 2. Result from the fine-tuned model trained. Red is for vehicles, white is for buildings, brown is for roads, green is for vegetation, blue is for sky, neon green is for undrivable roads, yellow for pedestrians and riders, cyan for poles, gray is for fence and purple for misc or unlabeled pixels

dataset have been remapped and we used 12 categories which include 'sky', 'building', 'pole', 'road', 'undrivable road', 'vegetation', 'sign symbol', 'fence', 'vehicle', 'pedestrian', 'rider' and 'unlabeled' to train our models.

Fig. 2 shows the model result for one image, which demonstrates classification with common features.

The intended application of this research is for autonomous vehicles which require of real-time implementation. It is then very important to balance the efficiency and accuracy of the proposed methods. The training and inference speed was highly improved by downsampling the input image at an early stage of the process and using only a small number of feature maps since most of them were redundant.

By downsampling the image, we reduce the size of the input parameters and control model overfitting, but also lose a wealth of information, which can't be restoring by resizing output to the same resolution as the input. In the final resizing process the zones near to the boundaries are affected due to their Sharpness information content. However, the loss of accuracy in the model may result in noisy predictions near class boundaries.

The next task is to associate uncertainty to the labels. Since the noise is more likely to happen near to the class boundaries, an log-odds distance field of the labeled is formed to represent the probability of the pixel to belong to the class.

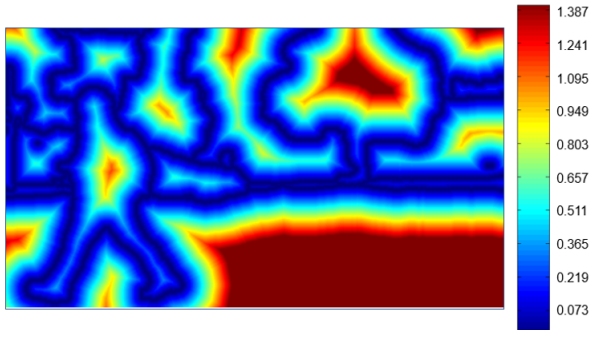

Fig. 3. Result of the log-odds distance map for the labeled image shown in Fig. 2. The color scale goes from Blue to Red which represents low (0.1) to high class (1.4) probability.

The distance field (map) changes the value of the intensities of the points inside the foreground regions of an image with their corresponding distances from the closest 0 value. The first step in our image processing algorithm is to extract the boundaries of our label through out of laplacian filter, with an approximation of a second derivative kernel. The images that contain the edges obtained as a result of applying the filter are dilated in order to make sure that every class boundary is closed. We then inverted the image to set the edges as background.

The next step is to build the distance map with the Euclidean distance of each pixel to the closest class edge. After this process the distance map is truncated and scaled using odd logs function Eq. (11), to map all the values between 0.1 and 1.4 as shown in Fig. 3.

The minimum and maximum values correspond to the max and min probability that a label can have, in this case, 0.51 on the pixels belonging to the boundary and 0.8 for those placed more inner the class.

An accurate calibration between the lidar and camera is needed for projecting labels of the visual classifier and its uncertainty to the point cloud. The calibration was performed using the Autoware calibration tool [9]. From this tool, we obtained the intrinsic parameters that provide the transformation between pixel coordinates and camera coordinates and extrinsic parameters that provide the transformation between camera coordinates and lidar coordinates.

The projection is performed by applying the coordinates transformations among lidar, camera, and image. We project and encode two pieces of information per each point which corresponding to the label and the odd-logs distance of the pixel where the 3D point was projected.

Our point grey camera has 56^o horizontal field of view, which makes the result of the projection a point cloud cover 56^o with 16 beams. Fig. 4. shows the result of our projection.

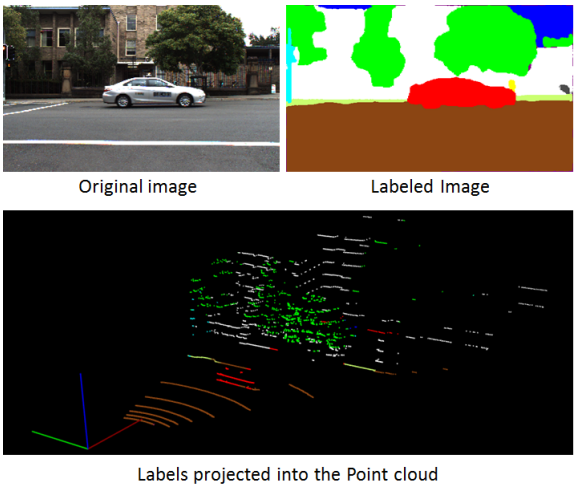

Fig. 4. The result of the label projection process to a point cloud. Colors in the point cloud correspond to the label colors assigned in Fig. 2.

B. Voxel occupancy and label probability

The OctoMap approach of Wurm et al. [1] is based on octrees and builds a 3D map of voxels (cubic volume unit) for a set of registered point clouds. The sensor readings integration is done by using the occupancy grid mapping introduced by Moravec and Elfes [18]. The probability $P(n|z_{1:t})$ of the voxel v to be occupied given the sensor measurements $z_{1:t}$ is estimated according to

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

$P(v|z_{1:t})$ is particular to the sensor that produced z_t. The update equation depends on the current measurement z_t, a prior probability $P(n)$, and the previous estimate $P(v|z_{1:t-1})$ from time point 1 to t-1. The Octomap approach [1] takes the common assumption of a uniform prior probability that leads to $P(v) = 0.5$.

The interpretation of the multi-label octree depends on the premise that every 3D point of a point cloud can be classified into different semantic labels. In this work, the update of the probability of each label for the node v is performed as follows:

- z denotes the sensor reading.
- c is the current label of the reading z.
- $P(c|z_{1:t})$ is the posterior voxel's label probability of the label c given the current and past sensor readings.
- $P(c|z_t)$ denotes the probability of the 3D point to belong to the class c
- $P(c_n|z_t)$ denotes the probability of the 3D point to belong to the class c_n (where $n = 1 : 11$).

To calculate the posterior distribution $P(c|z_{1:t})$ from the corresponding posterior on time step earlier $P(c|z_{1:t-1})$ [19], first step is to apply the Bayes rule to the target posterior:

$$P(c|z_{1:t}) = \frac{P(z_t|c)P(c|z_{1:t-1})}{P(z_t|z_{1:t-1})} \tag{2}$$

Now applying Bayes rule to the measurement model $P(z_t|c)$:

$$P(z_t|c) = \frac{P(c|z_t)P(z_t)}{P(c)} \tag{3}$$

We obtain:

$$P(c|z_{1:t}) = \frac{P(c|z_t)P(z_t)P(c|z_{1:t-1})}{P(c)P(z_t|z_{1:t-1})} \tag{4}$$

Now, we obtain the posterior distribution for the opposite event $\neg c$, which corresponds to the sum of the individual posterior distribution of the remaining labels:

$$P(\neg c|z_{1:t}) = \sum_{i=1}^{n-1} \frac{P(c_i|z_t)P(z_t)P(c_i|z_{1:t-1})}{P(c_i)P(z_t|z_{1:t-1})}$$

$$= \frac{P(z_t)}{P(z_t|z_{1:t-1})} \sum_{i=1}^{n-1} \frac{P(c_i|z_t)P(c_i|z_{1:t-1})}{P(c_i)} \tag{5}$$

For practicality, we assume the remaining probability in all cases is going to be equally distributed among the remaining labels, and since the number of labels is $n = 11$, we obtain:

$$P(\neg c|z_{1:t}) = \tag{6}$$

$$= \frac{P(z_t)}{P(z_t|z_{1:t-1})} \sum_{1}^{10} \frac{\left(\frac{1-P(c|z_t)}{10}\right)\left(\frac{1-P(c|z_{1:t-1}))}{10}\right)}{\left(\frac{1-P(c)}{10}\right)}$$

$$P(\neg c|z_{1:t}) = \frac{(1 - P(c|z_t))P(z_t)(1 - P(c|z_{1:t-1})}{(1 - P(c))P(z_t|z_{1:t-1})} \tag{7}$$

Now, our problem is reduced to a Binary Bayes Filter. By computing the ratio of (4) and (7) we obtain:

$$\frac{P(c|z_{1:t})}{P(\neg c|z_{1:t})} = \frac{\frac{P(c|z_t)P(z_t)P(c|z_{1:t-1})}{P(c)P(z_t|z_{1:t-1})}}{\frac{(1-P(c|z_t))P(z_t)(1-P(c|z_{1:t-1})}{(1-P(c))P(z_t|z_{1:t-1})}} \tag{8}$$

$$\frac{P(c|z_{1:t})}{P(\neg c|z_{1:t})} = \frac{P(c|z_t)P(c|z_{1:t-1})(1 - P(c))}{(1 - P(c|z_t))(1 - P(c|z_{1:t-1}))P(c)} \tag{9}$$

Log odds are an alternate way of expressing probabilities, which simplifies the process of updating them with new evidence. Log odds ratio is defined as:

$$l(x) = \log \frac{p(x)}{1 - p(x)} \tag{10}$$

Denote the log odds ratio of the belief $bel_t(x)$ by $l_t(x)$

$$l_t(c) = \log \frac{P(c|z_t)}{1 - P(c|z_t)} + \log \frac{P(c|z_{1:t-1})}{1 - P(c|z_{1:t-1})} + \log \frac{1 - P(c)}{P(c)} \tag{11}$$

The product turns into a sum

$$l_t(c) = l(P(c|z_t)) + l(P(c|z_{1:t-1})) - l(P(c)) \tag{12}$$

From Eq. (12) it is clear that to adjust the label of a voxel we need to integrate as many observations of the same label as have been integrated to define its current state. As an example, if k 3D point are placed in the same voxel with the label 'vegetation' and a constant probability p_l, then we need $k + 1$ points of a particular label (with the same p_l value) inside the voxel label to consider the voxel as this last label.

$l_t(c)$ is initialized with the value -2.3026 which corresponds to the log odds probability of $1/n$ $n = 11$.

$P(c)$ can be retrieved by

$$P(c) = 1 - \frac{1}{1 + exp(l_t(c))} \tag{13}$$

The sum of probabilities from the 11 labels in a single voxel must be equal to 1:

$$\sum_{i=1}^{11} P(c_i) = 1 \tag{14}$$

The probability $P(n, c_{max})$ of the most likely class c_{max} of each node n is calculated as follows:

$$P(n, c_{max}) = \underset{c}{\text{argmax}}[P(n, c_1), P(n, c_2), ..., P(n, c_{11})] \quad (15)$$

Fig. 5. demonstrates the label's probability update of a single voxel when a sequence of six labeled 3D points is located inside the voxel. The update 0 corresponds to the initial state of the label's probability, that are all initialized with the same value of 9.09%.

TABLE I
SEQUENCE OF LABELED POINTS TO UPDATE A SINGLE VOXEL

Update	Label	Probability
1	Vegetation	0.71
2	Vegetation	0.75
3	Vegetation	0.57
4	Pedestrian	0.59
5	Vehicle	0.59
6	Vegetation	0.65

Table I, shows the sequence of points used for updating the voxel, specifying the order, label and its probability. The first point entered inside of the voxel belongs to the label 'vegetation', with a label probability of 0.71. From Fig. 5. we can corroborate the increase of the probability for the label 'vegetation' (green) and the decrease of the remaining labels for the update 1. The same behavior can be seen with the update 2 till 3, where the voxel's belief of belonging to the label 'vegetation' keeps increasing. Update 4 is performed by a 3D point labeled as 'pedestrian' with a label probability of 0.59, then ochre bar gets larger. The Update 5 is due to a point labeled as 'vehicle', so the dark gray bar in the figure is larger in this case. The last update, another 'vegetation' labeled point is placed inside the voxel and the green bar is increased again.

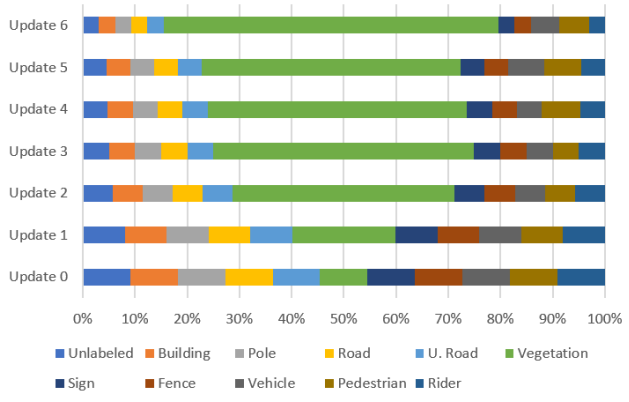

Fig. 5. Voxel probability variation for 6 updates

In all cases the value of the voxel probability of the label corresponding to the 3D point increases while the remaining values are decreasing to make sure all the time that the sum of probabilities is always 1.0.

IV. EXPERIMENTS AND RESULTS

We tested our algorithm on dataset collected around The University of Sydney. The data was collected by an electric vehicle (EV) equipped with one Velodyne VLP-16 sensor, a fixed lens Pointgrey camera with 56^o field of view (FOV) angle, an IMU with gyros, accelerometers and magnetometer, and encoders that provide accurate position of the vehicle.

The collected data was classified into the semantic labels through our methodology; first obtaining the semantic labels for the current frame, then we calculated the distance field map from the images and projected both pieces of information into the point cloud. The lidar provides us with a point cloud of 16 beams with 360^o FOV, since the camera's FOV is 56^o, after the projection we obtained around 15% of the original point cloud with label information. This portion of the original point cloud corresponds to the section overlapped between the VLP-16 and the camera, but the size of point cloud projected can vary depending the scenario since the lidar has a limited range of 100 meters.

The lidar has a vertical angular resolution of 2^o and the vertical field of view is 30^o, for this reason the point cloud becomes sparse at when the detected obstacle is further, this behavior is also reflected in the map, with sparse voxels located far from the vehicle's last position.

The labeled point cloud associated with the current position of the vehicle feeds our adapted Octomap algorithm. This component evaluates the occupancy and calculates the update the belief of the labels in every voxel. For visualization purposes the alpha channel of the octree map was affected with the probability of its label, being 1 a totally opaque color that represents the highest value of a class belief.

Fig. 6 shows the results of our algorithm for three different scenarios recorded at 5 frames per second, the first scenario corresponds to a road with vehicles parked at the sides, the voxel resolution was set to 0.4 m. The second scenario has just one vehicle parked on the road, the voxel resolution was set to 0.3 m. For the next three scenarios the voxel resolution was 0.5 m. We chose road environments for data collection since our primary objective is to use this algorithm as part of the path planning method for a self driving car.

The same environments were mapped using different voxel sizes, Fig. 7 displays a fourth and fifth surroundings. We set the voxel size as 0.5 m for the top image and 0.2 m for the bottom one. We can notice three major characteristics that are affected by the voxel's size:

- Contiguity of the ground plane: In the Octomap approach individual range measurements are integrated using raycasting. This updates the end point of the measurement as occupied while all other voxels along a ray to the sensor origin are updated as free [1]. However, discretization results of the ray-casting process can prompt to undesired results when using a sweeping lidar. During a sensor sweep over flat surfaces at shallow angles, volumes measured occupied in one 2D scan may be marked as free in the ray-casting of following scans

Fig. 6. Octomap representation. Red is for vehicles, white is for buildings, brown is for roads, green is for vegetation, blue is for sky, neon green is for undrivable roads, yellow for pedestrians and riders, cyan for poles, gray is for fence

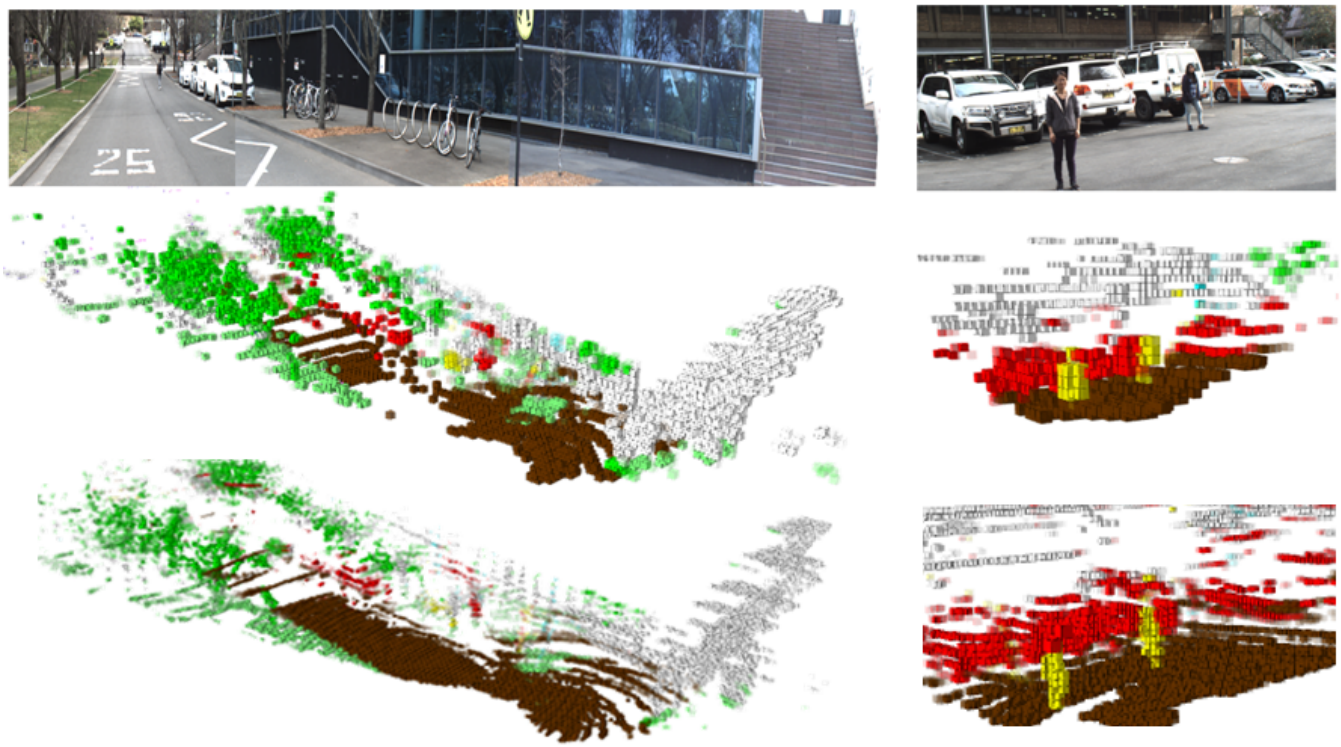

Fig. 7. Comparison of the Octomap representation for two different voxel sizes (0.5 m and 0.2 m). Red is for vehicles, white is for buildings, brown is for roads, green is for vegetation, blue is for sky, neon green is for undrivable roads, yellow for pedestrians and riders, cyan for poles, gray is for fence

[1]. This undesired updates commonly generate holes in the modeled surface. The larger the voxel size, the greater the likelihood of obtaining holes in the floor, as it can be seen at the Fig. 6.

- Details in obstacles: As the size of the voxel becomes smaller, the shape of the obstacles becomes more faithful to the original form, as it approaches the size of the points that make up the ground truth defined by the raw point cloud.
- Number of voxels: the larger the voxel size, the more points belonging to the cloud of points, will be within every voxel, therefore a smaller number of voxels will be required to represent the original data. A map made of bigger voxel needs less memory capacity and lower computational cost to perform its updated but sacrificing accuracy.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a new methodology to build an octree semantic map based on the projection of semantic labels from images to a point cloud. This methodology allows us to build a semantic map on the way based on labeled images provided by a CNN and a accumulation of a lidar

point cloud of the environment without the need of a dense point cloud.

Every time we obtain the current readings of the sensors (camera, lidar, odometry), the image is labeled and we estimate uncertainty to every label, under the premise that pixels near to the class boundary are less accurate using a distance field map. Labels and uncertainty are then projected to the collected point cloud, to be the input of our algorithm that builds probabilistically the semantic 3D octree map. The final map provides a good start to be incorporated in the processing and setting of driving behaviors and path planning algorithms.

It is also noticed the presence of sparse noise, this is due to minor inaccuracies in the lidar - camera calibration where some wrong labels are assigned to 3D points, which their projection is near to the label boundary. Nevertheless, our algorithm has shown robustness in updating class probabilities when the same area is repeatedly seen by the sensors from different angles. Tighter calibration of the lidar - camera relationship will improve performance further.

The quality of the resulting map depends on the size of the voxel, larger voxels produce maps that need less memory and are less faithful to the original forms of the environment. Smaller voxels generate more detailed maps, but since more volumetric units are needed to represent the scenario, the required memory space will be larger. The size of the voxel, then depends on the application and the geometric characteristics of the environment.

The frequency of our algorithm input data depends on the slowest sensor, in our case the VLP-16 works at 10 Hz and the Pointgrey camera at 5 Hz. Processing the collected data results in a labeled point cloud at 5 Hz. Quality of the results of our algorithm is then susceptible to the driving speed given we require redundant information to update label's probabilities.

For the next stage of this project, we plan to use 6 fast cameras around the vehicle for extending the field of view of our labeled point cloud to 360^o.

ACKNOWLEDGMENT

This work has been funded by the Australian Centre for Field Robotics and the University of Sydney through the Dean of Engineering and Information Technologies PhD Scholarship (South America). Research partially funded by arc by Australian research council discovery grant dp160104081 And university of Michigan collaboration.

REFERENCES

[1] A. Hornung, K.M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees" in Autonomous Robots, 2013; DOI: 10.1007/s10514-012-9321-0. Software available at http://octomap.github.com.

[2] T. Hackel, N. Savinov, L. Ladicky, J. D. Wegner, K. Schindler and M. Pollefeys, "Semantic3D.net: A new Large-scale Point Cloud Classification Benchmark" in ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Science, 2017.

[3] G. Ros, S. Ramos, M. Granados, A. Bakhtiary, D. Vazquez, and A. M. Lopez, "Vision-based off-line perception paradigm for autonomous driving", in 2015 IEEE Winter Conference on Applications of Computer Vision. IEEE, 2015. pp. 231-238.

[4] V. Vo, L. Truong-Hong, D. F. Laefer and M. Bertolotto, "Octree-based region growing for point cloud segmentation", in ISPRS Journal of Photogrammetry and Remote Sensing, Volume 104, 2015, pp. 88-100.

[5] W. Zhou, R. Arroyo, A. Zyner, J. Ward, S. Worrall, E. Nebot and L. M. Bergasa, "Transferring visual knowledge for a robust road environment perception in intelligent vehicles", in EEE 20th International Conference on Intelligent Transportation Systems, 2017

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks", in Advances in Neural Information Processing Systems (NIPS), December 2012, pp. 1106-1114.

[7] H. J. Chien, R. Klette, N. Schneider and U. Franke, "Visual odometry driven online calibration for monocular lidar-camera systems", in 23rd International Conference on Pattern Recognition (ICPR), Cancun, 2016, pp. 2848-2853.

[8] D. Lang, S. Friedmann and D Paulus, "Semantic 3D Octree Maps based on Conditional Random Fields", in International Conference on Machine Vision Applications (MVA2013), Kyoto, May 2013, pp. 185 -188.

[9] S. Kato, E. Takeuchi, Y. Ishiguro, Y. Ninomiya, K. Takeda, and T. Hamada. "An Open Approach to Autonomous Vehicles", IEEE Micro, Vol. 35, 2015, No. 6, pp. 60-69.

[10] X. Xiong, A. Adan, B. Akinci and D. Huber, "Automatic creation of semantically rich 3D building models from laser scanner data", in Automation in Construction, Volume 31, May 2013, pp 325-337.

[11] D. Munoz, J. A. Bagnell, N. Vandapel and M. Hebert, "Contextual classification with functional Max-Margin Markov Networks," in "2009 IEEE Conference on Computer Vision and Pattern Recognition", Miami, FL, 2009, pp. 975-982.

[12] A. Golovinskiy, V. G. Kim and T. Funkhouser, "Shape-based recognition of 3D point clouds in urban environments", in 2009 IEEE 12th International Conference on Computer Vision, Kyoto, 2009, pp. 2154-2161.

[13] C. Mallet, F. Bretar, M. Roux, U. Soergel and C. Heipke, "Relevance assessment of full-waveform lidar data for urban area classification", in ISPRS Journal of Photogrammetry and Remote Sensing, Volume 66, Issue 6, 2011, pp. S71-S84.

[14] K. Lai, L. Bo and D. Fox, "Unsupervised feature learning for 3D scene labeling", in 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, 2014, pp. 3050-3057.

[15] J. Huang and S. You, "Point cloud labeling using 3D Convolutional Neural Network," in 23rd International Conference on Pattern Recognition (ICPR), Cancun, 2016, pp. 2670-2675.

[16] R. Schnabel, R. Wahl and R. Klein, "Efficient RANSAC for Point-Cloud Shape Detection", in Computer Graphics Forum, vol. 26, no. 2, 2007, pp. 214-226.

[17] F. Lafarge and C. Mallet, "Creating Large-Scale City Models from 3D-Point Clouds: A Robust Approach with Hybrid Representation", in International Journal of Computer Vision, Vol. 99, Issue 1, August 2012, pp. 69-85.

[18] Moravec H, Elfes A, "High resolution maps from wide angle sonar", in Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA), 1985, St. Louis, MO, USA, pp 116121.

[19] Thrun, Sebastian, Wolfram Burgard, and Dieter Fox. Probabilistic robotics. MIT press, 2005.

[20] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding" in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016,pp. 3213-3223.

[21] P. Felzenszwalb and D. Huttenlocher, "Distance transforms of sampled functions", Technical report, Cornell University, 2004.

[22] S. Verghese, "Self-driving Cars and lidar," in Conference on Lasers and Electro-Optics, OSA Technical Digest (online) (Optical Society of America, 2017), paper AM3A.1.

[23] S. Sengupta, E. Greveson, A. Shahrokni and P. H. S. Torr, "Urban 3D semantic modelling using stereo vision," 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, 2013, pp. 580-585.

[24] S. Song, F. Yu, A. Zeng, A. Chang, M. Savva, T. Funkhouser, "Semantic Scene Completion from a Single Depth Image," CoRR, abs/1611.08974, 2016.