A Consistent and Long-term Mapping Approach for Navigation

Handuo Zhang, Hasith Karunasekera and Han Wang*

School of Electrical and Electronic Engineering, Nanyang Technological University, 637335, Singapore

Abstract: The construction and maintenance of a robocentric map is key to high-level mobile robotic tasks like path planning and smart navigation. But the challenge of dynamic environment and huge amount of dense sensor data makes it hard to be implemented in a real-world application for long-term use. In this paper we present a novel mapping approach by incorporating semantic cuboid object detection and multi-view geometry information. The proposed system can precisely describe the incremental 3D environment in real-time and maintain a long-term map by extracting out moving objects. The representation of the map is a collection of sub-volumes which can be utilized to perform pose graph optimization to address the challenge of building a consistent and scalable map. These sub-volumes are first aligned by localization module and refined by fusing the active volumes using co-visible graph. With the proposed framework we can obtain the object-level constraints and propose a consistent obstacle mapping system combining multi-view geometry with obstacle detection to obtain robust static map in a complex environment. Public dataset and self-collected data demonstrate the efficiency and consistency of our proposed approach.

Keywords: Robocentric mapping, Navigation, Obstacle detection, Map fusion.

1. INTRODUCTION

Mapping the 3D surroundings is one of the basic abilities of mobile robots and it is always a challenging task due to the huge magnitude of sensor data and different types of noises. In addition to proper visualization of the environment, mapping should provide sufficient information to assist obstacle avoidance, planning and navigation. In this paper we aim to provide a consistent and scalable mapping approach which is also lightweight and can provide long-term references.

Simultaneous Localization and Mapping (SLAM) has been a hot research topic in the last decades by achieving satisfactory accuracy in GPS-denied environment, and to increase localization robustness and reduce data amount to be processed, most SLAM methods [1] convert the observation data into a collection of sparse features. The map of collections serves for localization by providing the source of bundle adjustment [2], but they also limit the function only for localization. The sparse 3D landmarks cannot meet the need of tasks like path planning or long-term navigation. In recent years some robot techniques stemming from dense 3D reconstruction [3] are adopted into robot applications [4, 5]. However, the mediocre localization accuracy makes drift accumulate rapidly and thus makes the mapping result inconsistent.

The rapid development of deep neural network makes the recognition and localization of 2D [6, 7] and 3D [8] objects accurate and fast enough for real-time robot applications. This paper proposes a robocentric mapping approach combining the clues of 2D pixel labeling and 3D geometric structure of the environments, which can be applied in various mobile robot applications. 2D semantic information can recognize and extract movable objects like vehicles and pedestrians, which we model as map cuboids, representing temporary map elements. The 3D geometric observations remain the structure of the whole scene, so we can abstract them and maintain a consistent map after refinement. The differentiation of temporary dynamic objects and static surroundings ensures the effectiveness of navigation, especially for long-term use.

The contributions of the proposed approach can be summarized as following.

- We propose a real-time approach combining the environment geometry and 2D semantic information to leverage a 3D incremental volumetric representation that can be easily integrated into the pose graph optimization of SLAM process.
- We implement the mapping method into an efficient system that can handle dynamic obstacles and generate a consistent map for both short-term and long-term use.
- We evaluate the method using public dataset and demonstrate the usability and efficiency in urban areas.

*Address correspondence to this author at the NTU, 61 Nanyang Drive, Singapore, 637335; Tel: +65 67904506;
E-mail: {hzhang032, karu0009}@e.ntu.edu.sg, hw@ntu.edu.sg.

E-ISSN: 2409-9694/18

2. RELATED WORKS

For exploration in unknown environments metric maps are needed to perform path planning and obstacle avoidance. GMapping [9] and Hector Mapping [10] are widely used for laser range finders in indoor applications.

Many researchers chose to use the technique of sub-mapping [11, 12] to split the global estimation into many smaller mapping regions and compute individual estimation for each part, and then analyze the relationship between these sub-maps. However, during this process, there comes a lot of issues rising by sub-mapping, including map overlap, data duplication, map fusion, map alignment and global coordinate unification.

Dense mapping usually requires a large amount of computation and memory, which brings big trouble in real-time robotics application. Obstacle mapping can reduce the resources needed and meanwhile keep most of the information in dense maps. Obstacle detection and mapping is an essential task for robot to avoid collision and other dangers. The first task for autonomous vehicles is obstacle detection from raw sensor data like laser, radar or pure visual signals, then map of the environment is generated [13].

The map representation method can be divided into four categories. The first category is the probabilistic occupancy grid map-based approaches [14] which represents the scene as a 2D lattice, with each cell a certain area, and maintains the occupancy status whether the cell is occupied by obstacles. The second category is digital elevation map (DEM) based methods which stores the height information of the 3D point cloud. In [15], the obstacles are detected and fused with DEM. The third category utilizes scene flow segmentation approaches by merging the depth and motion information [16]. The fourth category is simply the full reconstruction of obstacles by geometric and color information [17].

Of the many 3D map representation methods, volumetric occupancy grids are the most effective approach for robotics [18]. While point clouds and dense surface representations are relevant for many applications, occupancy grids methods, have the advantage of providing definite and indefinite regions of space. One famous implementation is *OctoMap* [19], which exploits an octree-based data structure to accumulate data probabilistically with the advantage of low storage and maintaining the distinction between occupied, unoccupied, and unknown cells.

3. PROPOSED METHOD

In this section the proposed multiple layer mapping approach is introduced.

3.1. Problem Statement

For large scenarios, voxels with adjustable resolutions are widely used. Therefore, each obstacle can be subdivided by multiple volumes, with different heights.

The scene representation is a set of cells $\mathcal{M}$ following the notation in [20], so each obstacle $\mathcal{M}_i$ has the following attributes: position $\mathbf{p} \in \mathbb{R}^3$, normal vector $\mathbf{n} \in \mathbb{R}^3$, colour model $\mathbf{I} \in \mathbb{N}^3$, width $\omega \in \mathbb{R}$, height $h \in \mathbb{R}$ and the last updated timestamp t.

In our proposed framework, the occupied map $\mathcal{M}$ has three layers: movable cuboids ($\mathcal{M}_{temp}$) representing the temporarily existing objects, active volumes ($\mathcal{M}_{active}$) representing the local map sub-voxels for further refinement, and inactive collection of history volumes ($\mathcal{M}_{inactive}$) representing the global map voxels.

Assuming the source of data comes from only one type of sensor, we formulate the problem: Given 3D geometric measurement $\mathcal{G}=\{\mathcal{G}k\}^m$, collection of co-visible key-frames, estimated camera trajectory $\mathcal{T}_{GC^i} \in SE(3)$ and semantic object information $S = \{S_k\}_{k=1}^m$, the task is to calculate the global occupied map $\mathcal{M}$.

Our proposed approach combines the grid-based mapping and obstacle representation into mapping to describe large-scale environments by using small sub-volumes that cover only the essential parts of the environment, which implies that we remove the ground plane data and obstacle too far away. The overall system is depicted in Figure **1** and is composed of four modules: 2D object detection and tracking, active volume integration, temporal map update, and sub volume fusion. In the following sections we will explain these modules in detail.

3.2. Real-time Object Extraction and Tracking

As for a wide angle camera, an object detection $S_k^1 = \left(c_k^1, s_k^1, b_k^1\right) \in S_k^1$ extracted from keyframe k with detected class label c_k^1, detection confidence score $s_k^1 \in C$ where C is a predesigned class label set, and a

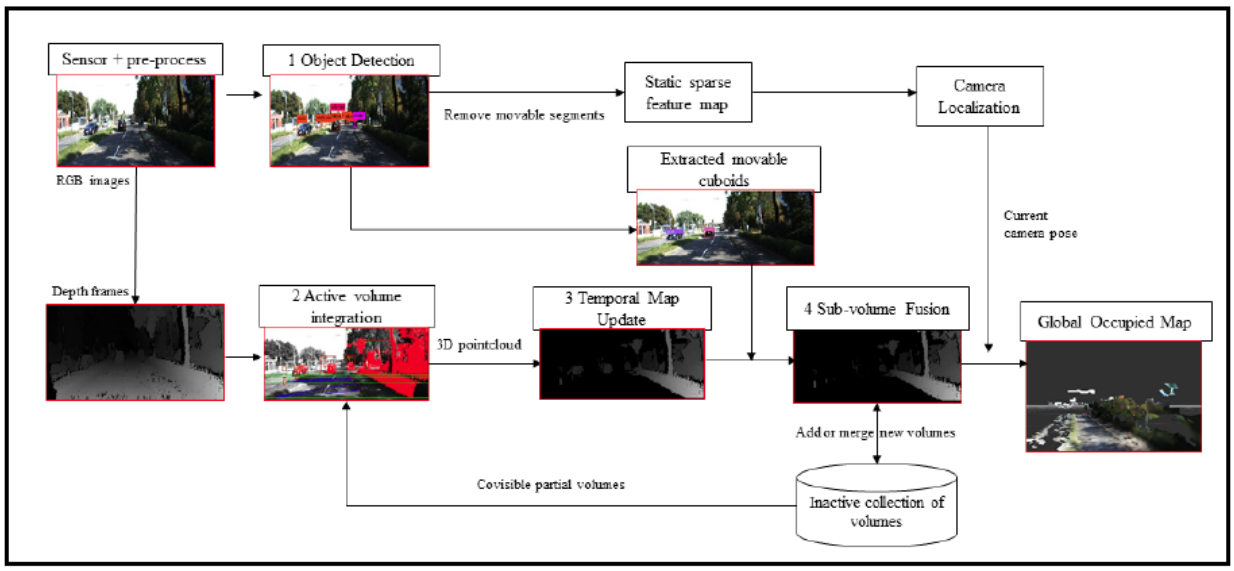


Figure 1: Block diagram of the proposed mapping approach. The global occupied map is generated and updated by the measurements of RGB-D or stereo cameras.

bounding box b_k^1. Such objects can be retrieved from many state-of-the-art approaches for object recognition, such as [6, 7, 21]. Some are based on CNN framework which must run on GPU and some like deformable parts model (DPM) can run on CPU in real time. As in most of these methods, an object proposal is needed, one advantage is that we have already the bounding boxes generated, so the search space has been dramatically reduced and the speed performance is much better. We utilize YOLOv3 framework [22] here to implement object classification and localization, as shown in sub figure (**a**) of Figure **2.**

After the extraction, to ensure the robustness we need to consider the previous n frames to keep track on the extracted objects. We utilize the newly developed *MASS*[1] technique to track the detected objects (yet the publication of this method is still under review). This method can guarantee efficiency and robustness. Thus, the temporary cuboid map layer $\mathcal{M}_{temp}$ is acquired, as shown in sub figure (**e**) of Figure **2.**

3.3. Active Volume Integration

The active volumes $\mathcal{M}_{active}$ consist the local sub volumes of the detected obstacles. "Local" here refers to the landmarks can be observed by a graph called *co-visible graph*. We detect all the obstacles in the scene with the help of u-v-disparity image space, which is equivalent to locating the peak response regions in the u-disparity image.

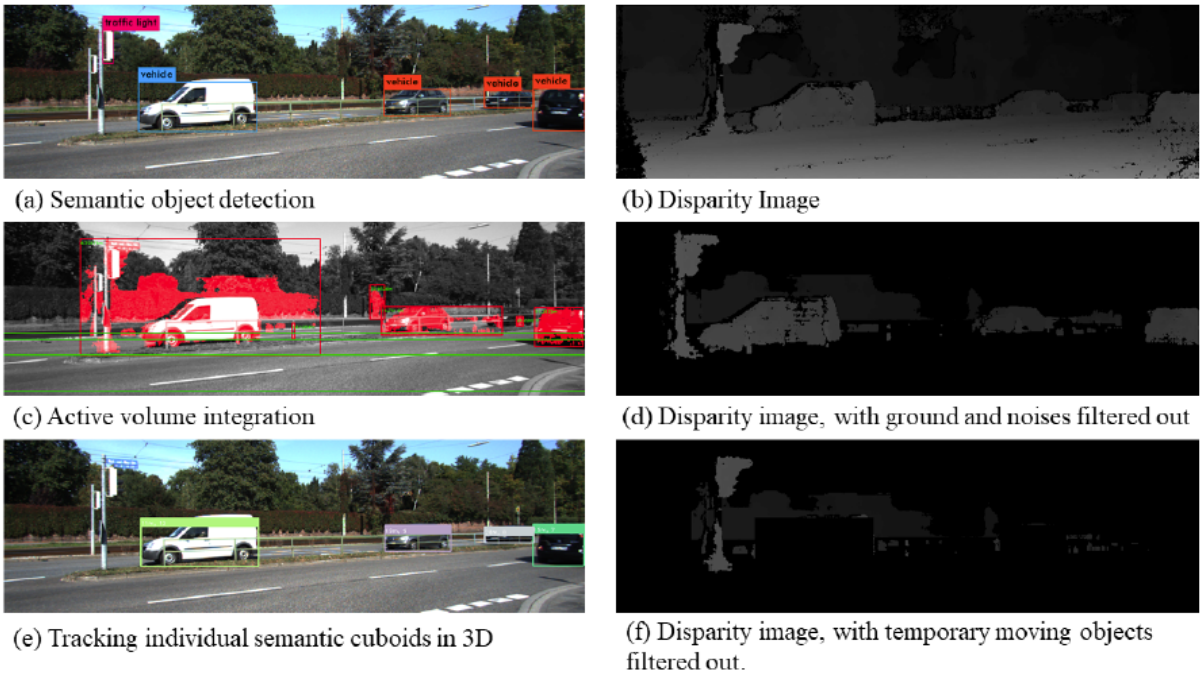
We firstly generate the region of interest (ROI), namely exclude all the objects that the vehicle will never collide in the current frame, like the sky or tree canopy which is too high or the road surface plane. Therefore, the search space of obstacles is greatly reduced, and many outliers can be rejected.

After the generation of ROI and statistical image preprocessing method like bilateral filter, we implement a depth based connected component approach to detect the obstacles. Based on the peak information of the u-disparity map with only ROI data, we exploit the method proposed in [23]. A hysteresis threshold is applied to remove the noises. As shown in sub figures (**c**) and (**d**) of Figure **2,** we are able to generate the active volume integration map thus obtain the corresponding obstacle disparity image as the refined map input.

3.4. Temporal Obstacle Map Update

The temporal obstacle mapping process utilizes N neighbor keyframes at keyframe k and each obstacle is searched along the epipolar line, generating N depth and normal vector hypotheses. Each depth hypothesis follows Gaussian distribution $N\left(\rho_s, \sigma_{\rho s}^2\right)$ due to observation noises containing feature extraction, matching and disparity resolution limitation. Each

[1] Multiple Object Tracking with Attention to Appearance, Structure, Motion and Size: http://www.cvlibs.net/datasets/kitti/eval_tracking_detail.php?result=e43ccf042b18819f740252b73de3d3da957d7273.



(a) Semantic object detection (b) Disparity Image

(c) Active volume integration (d) Disparity image, with ground and noises filtered out

(e) Tracking individual semantic cuboids in 3D (f) Disparity image, with temporary moving objects filtered out.

Figure 2: Illustration of obstacle detection process and active volume fusion process using object detection and depth information. (**a**) 2D object detection using YOLO [23] on the image from KITTI sequences 0014. (**b**) The corresponding disparity image generated using SGM method. (**c**) Obstacle segmentation using v-disparity and ground point triangulation techniques (Red regions are the obstacle masks within 30 meters). (**d**) The obstacle disparity image after (**c**). (**e**) The tracking method makes the detection result more robust. (**f**) The static obstacle disparity image after (**d**) and (**e**).

obstacle s_k is characterized by its average intensity value I, height h, width w and class label l, and the goal is to find the correspondence on image j.

The observed data is composed of the semantic obstacle measurements $\mathbf{s}_{1:Q}$, landmark geometric constraints $\mathbf{p}_{1:p}$ and camera poses $\boldsymbol{\xi}_{1:T}$. In each frame we have Q number of obstacles and P number of landmarks.

$$\begin{aligned} P(\mathcal{M}|\mathcal{G},\mathcal{S},\mathcal{X}) &\propto P(\mathcal{M}|\boldsymbol{\xi}_{1:T})P(\mathbf{p}_{1:P},\mathbf{s}_{1:Q}|\mathcal{M},\boldsymbol{\xi}_{1:T}) \\ &= P(\mathcal{M})\prod_{i=1}^{P}P\left(p_i|\mathcal{M},\xi_p\right)\prod_{j=1}^{Q}P\left(s_j|\mathcal{M},\xi_j\right) \\ &= \underbrace{P(\mathcal{M})}_{\text{prior}}\prod_{i=1}^{P}\underbrace{P(p_i|\mathbf{m}_i,\xi_i)}_{\text{geometric measurement}}\prod_{j=1}^{Q}\underbrace{P\left(s_j|\mathbf{m}_j,\xi_j\right)}_{\text{obstacle measurement}} \end{aligned} \tag{1}$$

where the conditional independence assumption is applied to obtain the equation. Then according to Bayes rules, we get the inverse sensor model:

$$P(\mathcal{M}|\mathcal{G},\mathcal{S},\mathcal{X}) \propto P(\mathcal{M})\prod_{i=1}^{P}P\left(\mathbf{m}_i|p_i,\xi_i\right)\prod_{j=1}^{Q}P\left(\mathbf{m}_j|s_j,\xi_j\right) \tag{2}$$

So finally, we are able to use the maximum posterior (MAP) estimate $\mathcal{M}^* = \text{argmax}_{\mathcal{M}}P(\mathcal{M}|\mathcal{G},\mathcal{S},\mathcal{X})$ to fuse the existing global map with the new added sub-map $\Delta\mathcal{M}_k$. Maximizing the posterior consists in finding the configuration of the nodes $\mathbf{x}*$ that minimizes the energy of all the edges co-visible locally in the map graphs with obstacle observation and pose vertices. Then we try to seek for the most likely configuration to assign the newly added graph nodes by minimizing the following cost function:

$$\begin{aligned} \mathbf{F}(\mathcal{X}) = &\sum_{<i,j>\in\mathcal{C}_r}\mathbf{e}_{ij}^{\top}\boldsymbol{\Omega}_{ij}\mathbf{e}_{ij} + \sum_{<i,j>\in\mathcal{C}_m}\mathbf{e}_{ij}^{\top}\boldsymbol{\Omega}_{ij}\mathbf{e}_{ij} \\ &+ \sum_{<i,j>\in\mathcal{C}_{inter}}\mathbf{e}_{ij}^{\top}\boldsymbol{\Omega}_{ij}\mathbf{e}_{ij} \end{aligned} \tag{3}$$

where $\mathcal{C}_r$ and $\mathcal{C}_m$ are respectively the edges in the reference and matched graphs $\mathcal{G}_r$ and $\mathcal{G}_m$, while $\mathcal{C}_{inter}$ are the edges connecting the two graphs. Therefore, to merge the graphs, we set the output a set of edges $\mathcal{M}$ connecting the vertices of $\mathcal{G}_r$ to vertices of $\mathcal{G}_m$. Each time we expand a node $\mathbf{x}_c$ in the current graph and seek for neighbors in the reference and we try to match the observations.

We display the pseudo-code for the whole procedure in Algorithm **1**.

Algorithm 1: Temporal and Spatial Map Merging

Require: $\mathcal{G}_r$: reference graph, $\mathcal{G}_m$: graph to be merged, $\boldsymbol{x}_m$: initial vertex in $\mathcal{G}_m$, d: minimum distance for matching.

Ensure: $\mathcal{M}$: edges connecting vertices of $\mathcal{G}_r$ to those of $\mathcal{G}_m$.

$\mathcal{M} := \emptyset$; // Initialize the output set of measurements.
while !queue.empty() **do**
  $\mathbf{x}_c$ = queue.front(); // Extract the first node of the matchable graph.
  queue.popFront();
  $\mathcal{N}$ = findNeighbors($\mathcal{G}_r, \mathbf{x}_r$); // Find the neighbors of $\mathbf{x}_r$ in reference graph.
  $S := \emptyset$; //Clear the matching set.
  for all $n \in \mathcal{N}$ **DO** //Try to match each neighbor and put the results in $\mathcal{S}$.
    if $|n - x_c|$ **then** S.add(Match($n, \mathbf{x}_c$));
    endif
  endfor
  s = bestMatch(S); //Pick up the best match $\mathcal{M}$.
  if s.score()≥minScore **then**
    $\mathcal{M}$. add(edge($\mathbf{x}_c, n$, s.transform()));
    $\mathbf{x}_c = n + s$.transform(); //Initialize $\mathbf{x}_c$ by transforming s to n;
  endif
  for all $\mathbf{x}_n$ ∈neighbours($\mathbf{x}_c$) **do**
    if $\mathbf{x}_n$.parent()==null **then**
      $\mathbf{e}$ = Edge($\mathbf{x}_c, \mathbf{x}_n$);
      $\mathbf{x}_n = \mathbf{x}_c + \mathbf{e}$.transform();
      queue.pushback($\mathbf{x}_n$)
    end if
  end for
end while

4. EXPERIMENTS

Since we are jointly estimating both the mapping and structure of self-localization, it is expected that we improve both. In this section we define the evaluation method for measuring the above and show results to validate our claim. We demonstrate the results of our proposed approach in public KITTI dataset and real dataset captured in NTU campus. Both these datasets involve difficult fast forward moving stereo cameras.

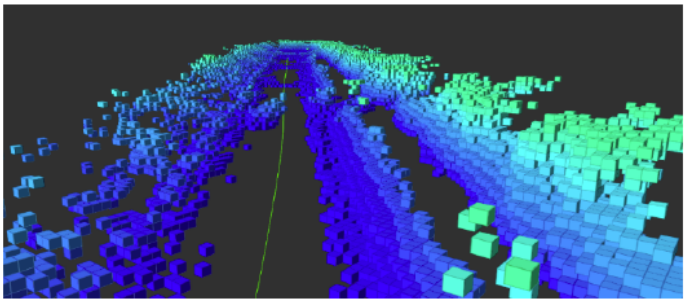
The NTU experimental datasets were recorded using a stereo camera set in outdoor environment (NTU campus). The data include images captured at 30 Hz. We have performed all experiments in a desktop with an Intel i5-8400 processor with 6 threads and a NVIDIA 1080 GPU (for both parallel stereo matching and object detection). For ego-motion estimation and mapping, we just use the power of CPU. We set the minimum disparity pixel 12 as the threshold to disregard all the obstacles 50 meters away from the sensor.

4.1. Map Merging with Visual Data

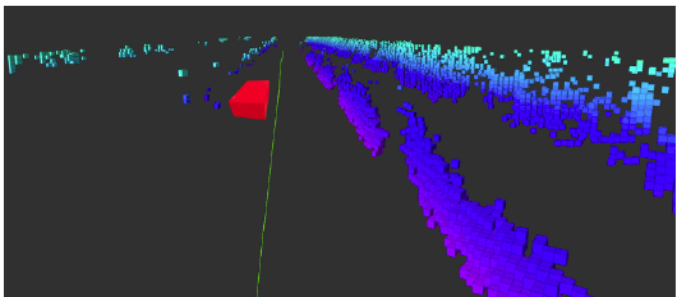
As shown in Figure **3**, we generate an *obstacle map* to annotate where are the obstacles and where it is free to travel, facilitating robots the ability to avoid obstacles. From our 3D obstacle map, we can obtain the probability, size and position of the detected individual obstacles, in both local and global coordinates.

The advantage over occupancy grid map is that the obstacle mapping consumes low memory and calculation resources, which is favorable applied in large scale environment. Table **1** shows the average run time for each module and total process for each frame, including object detection, active volume integration, disparity generation and sub-map fusion, running in optimized multi-thread framework (excluding the camera localization module, which is not tightly coupled with our mapping task). It is obvious that the system operates in real-time as the total time spent is much less than the total sequence length. However, it

(**a**) Original image



(**b**) Octomap from raw point cloud



(**c**) Our proposed mapping results

Figure 3: Global map generated by a scenario in KITTI dataset (**a**) Original image. (**b**) Octomap after removing road surface. (**c**) Our proposed system. (The slim green line in the middle is the robot trajectory. The cuboid rendered in red represents the vehicle removed from our inactive map but remained in the temporary map.

Table 1: Average runtime (ms) of our Proposed Method and that of OctoMap [19], with Around 1000 Runs. The Total Process Time is Run Under Multi-thread Architecture to Ensure Efficiency. The Resolution of output Map which has an Impact on the Mapping Process Time is Set to 0.2 Meter. "-" here Means *not utilized*

Dataset	Method	Object Detection	Disparity Generation	Volume Integration	Map Fusion	Total
Customized camera	**Proposed Semantic**	18.62	20.57	13.71	12.80	54.83
	Proposed no class	-	16.77	13.70	**6.74**	**37.21**
	Octomap	-	16.71	-	25.01	42.72
Camera for KITTI	**Proposed Semantic**	20.34	22.43	14.52	14.73	58.66
	Proposed no class	-	16.96	14.50	**7.45**	**38.91**
	Octomap	-	16.94	-	24.79	42.73

is noteworthy that the mapping module is always around 30 milliseconds delay permitting the generation of keyframe pose estimation.

4.2. Semantic Temporary Cuboids

In this section, the performance of temporary moving cuboid is presented qualitatively to verify the

accuracy. All the maps are created online and displayed without any post-processing. Note that different from the existing dense reconstruction methods taking a large amount of memory or sparse structure mapping methods that provide only localization information, our system can handle obstacle mapping in real time spending limited resources and providing 3D structure information of the environment. In Figure **4** the obstacle map of the outdoor scene is presented. According to Table **2** illustrated in Appendix, the average displacement error is within 0.5 m and the average bearing angle error is about 1 degree, which validates the performance of our proposed approach.

CONCLUSION

In this paper we carry out a new framework to build a robust and accurate 3D global occupied map on mobile robotic system, which can limit map growth, and run in real time with the help of a GPU. Firstly, we extract out semantic moving objects out into the first layer: temporal moving map layer. Then we fuse depth information with the RGB image data to leverage a collection of 3D points without the moving objects, ground plane and noises. The points are converted to sub-volumes and redundant sub-volumes are disregarded during the process of fusing and updating, to limit map growth amount. The second map layer, collection of active sub-volumes, is produced and ready to be updated until they become the third kind of map layer, inactive volume collection, when they are not in the current key frame sliding window. The sliding window maintains a covisibility graph inside the visual SLAM framework, so the map is coupled with the localization module.

We focus on the spatial and temporal obstacle merging based on the covisibility property of visual SLAM system and update map with the help of ego-motion estimation and joint optimization with multiple depth hypotheses.

In our evaluation of public KITTI and self-collected datasets, the results show that our proposed approach can maintain a consistent global map. The three kinds of map layer have their own usages, and the inactive sub-volume map layer can be further stored for long-term use.

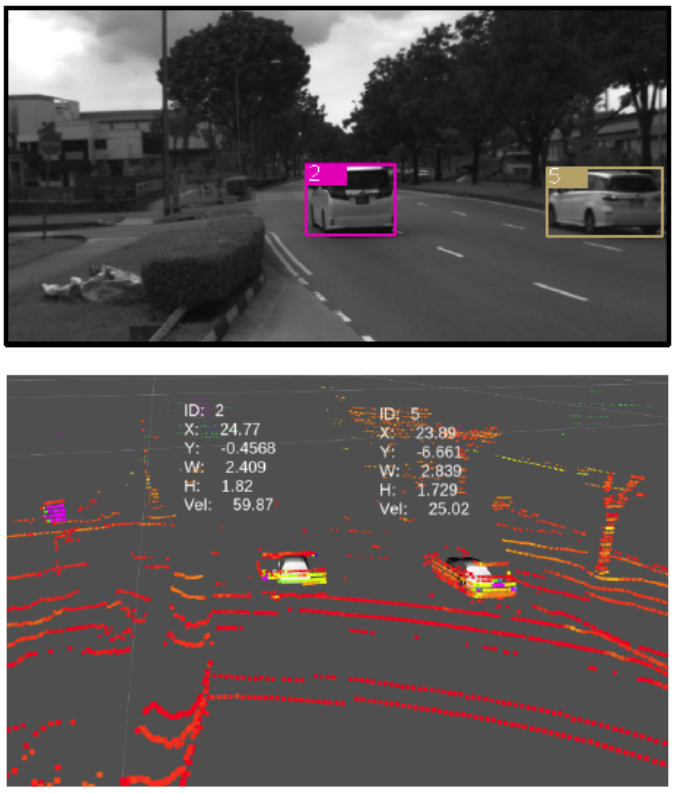


Figure 4: The 3D temporal obstacle mapping results in outdoor self-collected dataset. The ID number displayed represents the tracking identity of individual vehicles. The top figure denotes the semantic object detection and tracking results. The bottom one represents Temporary moving map layer evaluated by projecting 3D point cloud (red sparse points) from Velodyne sensor.

APPENDIX

The dataset comes from self-collected data in NTU campus. We hand-picked multiple points from Velodyne point cloud and compared them with our proposed obstacle detection results.

Table 2: The Accuracy of Obstacle Detection Compared with 3D Velodyne Lidar Sensor Measurements with 10 Samples

No.	Range of X Direction (meters)			Range of Y Direction (meters)			Bearing (degree)			X Direction Measurement		Object Width (meters)			Y Direction Measurement		Object Height (meters)		
	Est	Meas	Error	Est	Meas	Error	Est	Meas	Error	Xmin	Xmax	Est	Meas	Error	Ymin	Ymax	Est	Meas	Error
01	13.38	12.04	0.62	-2.89	-2.27	0.26	-24.38	-21.35	3.02	-1.95	-4.04	2.80	2.09	0.71	-0.72	-2.22	1.84	1.50	0.34
02	24.77	23.00	1.05	-0.46	-0.05	0.05	-2.13	-0.25	1.88	0.54	-1.75	2.41	2.29	0.12	-1.20	-2.79	1.82	1.59	0.23
03	23.89	22.90	0.27	-6.67	-6.36	0.05	-31.20	-31.04	0.16	-5.98	-7.95	2.84	1.97	0.87	-1.33	-2.42	1.73	1.09	0.64
04	44.59	43.31	0.56	5.39	5.42	0.33	13.78	14.27	0.48	6.46	4.71	2.02	1.75	0.27	-1.48	-2.97	1.98	1.49	0.49
05	39.35	36.97	1.66	-1.58	-1.55	0.33	-4.60	-4.80	0.20	-0.81	-2.55	2.56	1.74	0.82	-0.95	-2.78	1.70	1.83	0.13
06	25.73	26.41	1.40	0.55	0.78	0.13	2.45	3.38	0.93	1.73	-0.15	1.84	1.88	0.04	-1.83	-3.26	1.58	1.43	0.15
07	35.20	36.96	2.48	-1.68	-1.41	0.09	-5.47	-4.37	1.10	-0.02	-2.52	2.85	2.50	0.35	-0.29	-2.26	3.16	1.97	1.19
08	31.85	30.49	0.64	-3.48	-2.99	0.13	-12.47	-11.20	1.27	-1.37	-3.82	2.03	2.45	0.42	-2.20	-3.90	1.51	1.70	0.19
09	18.58	19.27	1.41	-0.68	-0.33	0.01	-4.19	-1.96	2.23	0.55	-1.35	1.85	1.90	0.05	-1.68	-2.45	1.51	0.77	0.74
10	26.76	19.59	6.45	8.09	6.16	2.29	33.64	34.91	1.27	6.48	5.89	0.98	0.59	0.39	-1.05	-2.70	2.34	1.65	0.69
	Camera to Velodyne Offset		**0.72**	**Camera to Velodyne Offset**		**0.36**													

Note: The metrics of accuracy can be calculated by:

$$Error\ Range_x = Estimation + Camera\ to\ Velodyne\ Offset - Measurement$$

$$Error\ Range_y = Estimation - Camera\ to\ Velodyne\ Offset - Measurement$$

REFERENCES

[1] JMM. Mur-Artal Raúl, Montiel and JD. Tardós, "ORB-SLAM: a Versatile and Accurate Monocular SLAM System," IEEE Transactions on Robotics 2015; 31(5): 1147-1163. https://doi.org/10.1109/TRO.2015.2463671

[2] B. Triggs, PF. McLauchlan, RI. Hartley, and AW. Fitzgibbon, "Bundle Adjustment - A Modern Synthesis," in Vision Algorithms: Theory and Practice, vol. 1883, B. Triggs, A. Zisserman, and R. Szeliski, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000; 298-372. https://doi.org/10.1007/3-540-44480-7_21

[3] S. Izadi *et al.*, "Kinect Fusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera," in Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology, New York, NY, USA, 2011, pp. 559-568. https://doi.org/10.1145/2047196.2047270

[4] R. Wagner, U. Frese, and B. Bäuml, "Graph SLAM with signed distance function maps on a humanoid robot," in 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems 2014; 2691-2698. https://doi.org/10.1109/IROS.2014.6942930

[5] T. Whelan, S. Leutenegger, R. Salas-Moreno, B. Glocker, and A. Davison, "ElasticFusion: Dense SLAM without a pose graph," 2015. https://doi.org/10.15607/RSS.2015.XI.001

[6] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in Advances in neural information processing systems 2015; 91-99.

[7] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proceedings of the IEEE conference on computer vision and pattern recognition 2016; 779-788. https://doi.org/10.1109/CVPR.2016.91

[8] A. Mousavian, D. Anguelov, J. Flynn, and J. Kosecka, "3D Bounding Box Estimation Using Deep Learning and Geometry," in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI 2017; 5632-5640. https://doi.org/10.1109/CVPR.2017.597

[9] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," IEEE transactions on Robotics 2007; 23(1): 34-46. https://doi.org/10.1109/TRO.2006.889486

[10] S. Kohlbrecher, J. Meyer, O. von Stryk, and U. Klingauf, "A Flexible and Scalable SLAM System with Full 3D Motion Estimation," in Proc. IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR), 2011. https://doi.org/10.1109/SSRR.2011.6106777

[11] M. Bosse, P. Newman, J. Leonard, M. Soika, W. Feiten, and S. Teller, "An atlas framework for scalable mapping," in Robotics and Automation. Proceedings. ICRA'03. IEEE International Conference on 2003; 2: 1899-1906.

[12] P. Piniés and JD. Tardós, "Scalable SLAM building conditionally independent local maps," in Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on, 2007; 3466-3471. https://doi.org/10.1109/IROS.2007.4399302

[13] S. Thrun and others, "Robotic mapping: A survey," Exploring artificial intelligence in the new millennium, 2002; 1: 1-35.

[14] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," Computer 1989; 22(6): 46-57. https://doi.org/10.1109/2.30720

[15] F. Oniga and S. Nedevschi, "Processing dense stereo data using elevation maps: Road surface, traffic isle, and obstacle detection," IEEE Transactions on Vehicular Technology 2010; 59(3) 1172-1182. https://doi.org/10.1109/TVT.2009.2039718

[16] N. Bernini, M. Bertozzi, L. Castangia, M. Patander, and M. Sabbatelli, "Real-time obstacle detection using stereo vision for autonomous ground vehicles: A survey," in Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on, 2014; 873-878. https://doi.org/10.1109/ITSC.2014.6957799

[17] A. Broggi, S. Cattani, M. Patander, M. Sabbatelli, and P. Zani, "A full-3D voxel-based dynamic obstacle detection for urban scenario using stereo vision," in Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on 2013; 71-76. https://doi.org/10.1109/ITSC.2013.6728213

[18] K. Konolige, "Improved occupancy grids for map building," Autonomous Robots 1997; 4(4): 351-367. https://doi.org/10.1023/A:1008806422571

[19] A. Hornung, KM. Wurm, M. Bennewitz, Cyrill Stachniss, and W. Burgard, "OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees," Autonomous Robots, 2013. https://doi.org/10.1007/s10514-012-9321-0

[20] M. Keller, D. Lefloch, M. Lambers, S. Izadi, T. Weyrich, and A. Kolb, "Real-time 3d reconstruction in dynamic scenes using point-based fusion," in 3DTV-Conference, 2013 International Conference on, 2013; 1-8. https://doi.org/10.1109/3DV.2013.9

[21] CR. Qi, L. Yi, H. Su, and LJ. Guibas, "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space," arXiv:1706.02413 [cs], Jun. 2017.

[22] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," arXiv preprint arXiv:1804.02767, 2018.

[23] M. Wu, C. Zhou, and T. Srikanthan, "Robust and low complexity obstacle detection and tracking," in Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on, 2016; 1249-1254. https://doi.org/10.1109/ITSC.2016.7795717

Received on 17-5-2018 Accepted on 28-5-2018 Published on 20-12-2018

DOI: http://dx.doi.org/10.15377/2409-9694.2018.05.4