IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 28, 2020, accepted October 11, 2020, date of publication October 16, 2020,
date of current version December 21, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3031867

Semantic RGB-D SLAM for Rescue Robot Navigation

WENBANG DENG[1], KAIHONG HUANG[1], XIEYUANLI CHEN[2], ZHIQIAN ZHOU[1], CHENGHAO SHI[1], RUIBIN GUO[1], AND HUI ZHANG[1]

[1]Robotics Research Center, College of Intelligence Science and Technology, National University of Defense Technology, Changsha 410073, China
[2]Photogrammetry and Robotics Laboratory, University of Bonn, 53115 Bonn, Germany

Corresponding author: Kaihong Huang (kaihong.huang11@outlook.com)

This work was supported by the National Natural Science Foundation of China under Grant 61773393, Grant U1813205, and Grant U1913202.

ABSTRACT In this paper, we propose a semantic simultaneous localization and mapping (SLAM) framework for rescue robots, and report its use in navigation tasks. Our framework can generate not only geometric maps in the form of dense point-clouds but also corresponding point-wise semantic labels generated by a semantic segmentation convolutional neural network (CNN). The semantic segmentation CNN is trained using our RGB-D dataset of the RoboCup Rescue-Robot-League (RRL) competition environment. With the help of semantic information, the rescue robot can identify different types of terrains in a complex environment, so as to avoid specific obstacles or to choose routes with better traversability. To reduce the segmentation noise, our approach utilizes depth images to perform filtering on the segmentation results of each frame. The overall semantic map is then further improved in the point-cloud voxels. By accumulating results of multiple frames in the voxels, semantic maps with consistent semantic labels are obtained. To show the advantage of having a semantic map of the environment, we report a case study of how the semantic map can be utilized in a navigation task to reduce the arrival time while ensuring safety. The experimental result shows that our semantic SLAM framework is capable of generating a dense semantic map for the complex RRL competition environment, with which the arrival time of the navigation time is effectively reduced.

INDEX TERMS Deep learning, path planning, RoboCup, rescue robot, semantic SLAM.

I. INTRODUCTION

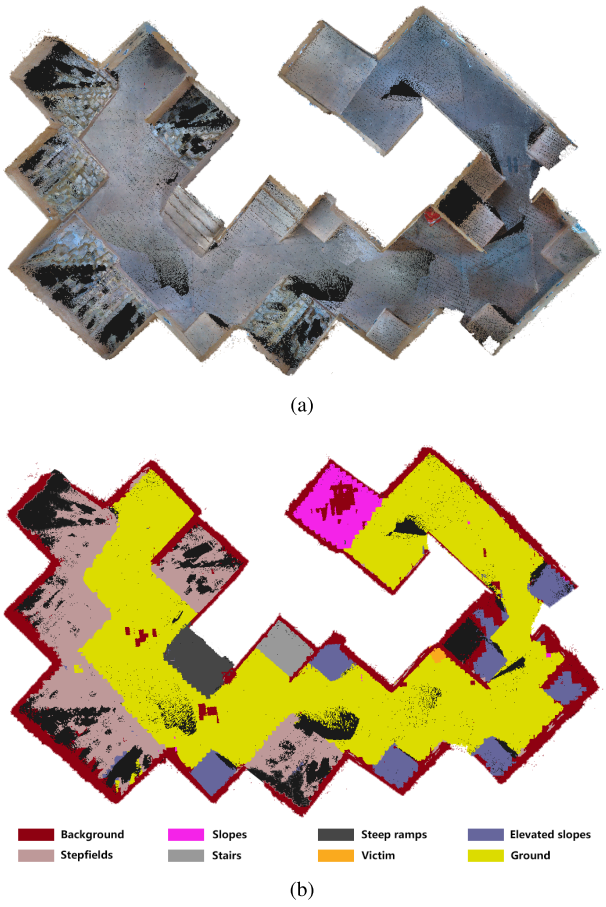
Semantic information representing classes of objects allows robots to understand their surroundings in a higher level other than geometry or appearance. With the help of semantic information, robots can perform better in tasks like path planning and human-robot interaction, etc. For example, in the RoboCup Rescue-Robot-League (RRL) competition – an international competition for evaluating the performance of rescue robots – the contestants need to autonomously traverse and generate maps for a maze consisting of challenging terrains, such as stairs, stepfields, elevated slopes, and steep ramps. It is crucial for the robot to understand what terrains it encountered and then act accordingly to avoid terrains that are difficult or dangerous to traverse.

In this paper, we propose a semantic simultaneous localization and mapping (SLAM) framework for rescue robots to better navigate through challenging RRL environments. Our framework combines the well-known ORB-SLAM2 [1] method and a convolutional neural network (CNN) to generate both geometric and semantic maps of dense point-cloud, using an RGB-D camera.

In order to reduce the adverse effect of segmentation error, our method performs filtering on both single frame and across multiple frames. For single frame filtering, the depth information is utilized to estimate whether neighboring pixels in semantic images belong to the same object, thereby reducing the mislabelled pixels. For multiple frame filtering, we accumulate local point-clouds of multiple frames into voxels and determine the most frequently appeared semantic label for each voxel. Fig. 1 depicts an example of mapping results.

With the point-wise semantic labels available to us, we improve the existing path planning algorithm by also considering the semantic information. For example, in the RRL competition environment, stepfields are notoriously hard to traverse as they are made up of wooden blocks with

The associate editor coordinating the review of this manuscript and approving it for publication was Yangmin Li.



(a)

(b)

FIGURE 1. Example dense semantic map generated by our approach. (a) Colored point-cloud map of an RRL test field (view from above). (b) Corresponding semantic map including eight types of terrain.

random heights. Our framework can be used to identify such regions.

We perform real-world experiments to validate the proposed method. Experimental results suggest that our approach is able to generate dense semantic maps for the complex environment. In the rest of the paper, we introduce related works in Sec. II and introduce the proposed methods in detail in Sec. III. We evaluate the effectiveness of the neural network training, semantic image and semantic map optimization by experiments in Sec. IV. Finally, we conclude our work in this paper and introduce our future work.

II. RELATED WORKS

A. SEMANTIC SEGMENTATION

With the development of CNN, semantic segmentation algorithms are widely based on deep learning because of its speed and accuracy. FCN [2] is the first successful approach that utilizes CNN to accomplish semantic segmentation. After that, SegNet [3] adds a decoder-encoder structure on the basis of FCN. To optimize segmentation result, EncNet [4] and DeepLab series algorithms [5], [6] proposed by Google consider context information when processing images. FastFCN [7] runs three times faster than EncNet without the accuracy loss by replacing dilated convolutions with Joint Pyramid Upsampling (JPU) model in ResNet-101 [8]. Apart from common RGB semantic segmentation algorithms mentioned above, there are also RGB-D based [9]–[13] and point-cloud based [14] approaches.

B. SIMULTANEOUS LOCALIZATION AND MAPPING

SLAM system enables robots to build maps of surroundings and locate their position in the environment. Appearance-based SLAM system RTAB-Map [15] achieves long-term mapping and loop closure detection by utilizing the memory management method. ElasticFusion [16] is a surfel-based dense SLAM system. To realize pose estimation, it fuses the iterative closest point (ICP) method with the direct method. However, GPU is needed when running this algorithm. ORB-SLAM2 is a lightweight SLAM system that utilizes ORB image features to achieve fast feature extraction and matching, and precise pose estimation. Because ORB-SLAM2 has a higher precision of pose estimation and can be easily deployed to robots, our approach employs ORB-SLAM2 as our SLAM system.

C. SEMANTIC SLAM

Most semantic SLAM methods fuse semantic labels obtained from semantic segmentation and maps generated by the SLAM algorithm to generate 3D maps with semantic information. According to the type of sensor they used, semantic SLAM algorithms can be classified as the monocular camera-based [17]–[19], stereo camera-based [20], [21], LiDAR-based [22]–[24], multiple sensors-based [25], [26], RGB-D camera-based approaches, and so on. This paper mainly considers semantic SLAM algorithms based on the RGB-D camera.

RGB-D camera can provide both color and depth information of the environment. SemanticFusion [27] uses both information throughout its system: it adds the depth channel into the RGB semantic segmentation neural network, so as to improve the performance of the network. Semantic labels are integrated into maps from ElasticFusion and updated by the recursive Bayesian method. Unlike SemanticFusion, Nakajima *et al.* [28] use RGB images as input to the neural network, and assign class probabilities to each segmentation label instead of each element (like voxel and surfel) to improve efficiency and reduce storage complexity. To improve accuracy, Antonello *et al.* [29] leverage multiple views to make the semantic segmentation result of a single frame more accurate. DS-SLAM [30] is based on the semantic segmentation method SegNet [3] and the SLAM system ORB-SLAM2. In order to make this approach accurate and robust in dynamic environments, moving consistency check is proposed to filter out moving objects in the scene. Unlike methods mentioned above, Sünderhauf *et al.* [31] utilize object detection algorithm SSD [32] to locate objects in the image before using depth images to extract boundaries and associating semantic labels of objects.

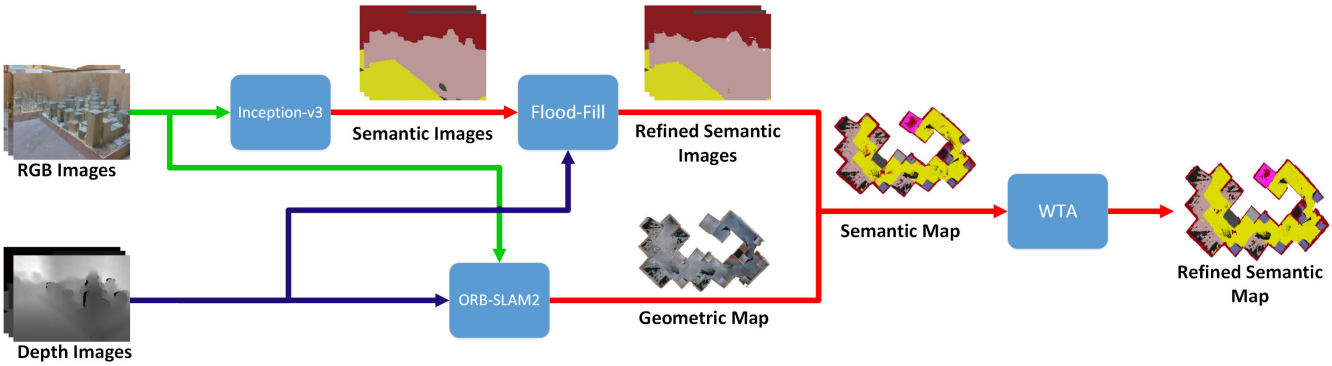


FIGURE 2. Overview of our approach. Semantic segmentation CNN Inception-v3 utilizes RGB images to generate semantic images. To optimize these semantic images, we leverage corresponding depth images to implement flood-fill, so that we can get refined semantic images. Meanwhile, in the process of generating the dense geometric point-cloud map, RGB images and depth images are used by ORB-SLAM2. After fusing refined semantic images and geometric point-cloud map to generate the semantic map, we use "winner-takes-all" (i.e. WTA) to split the semantic map into voxels and refine it by counting the number of each label in every voxel, so as to generate the refined semantic map at last.

D. SEMANTIC INFORMATION FOR PATH PLANNING

Common path planning algorithms often consider only the geometric information when generating feasible paths. These methods can distinguish whether there is an obstacle in front of the robot or not, whereas they cannot tell how difficult it is to traverse the obstacle. With semantic information, robots can have an understanding of what terrain is in front of them and what cost it takes to pass through. Wang *et al.* [33] utilize semantic information to distinguish rooms and corridors. Since corridors will be observed many times, robots can scan rooms first, so that robots do not have to go through repeated routes. Similarly, Sünderhauf *et al.* [34] distinguish offices and corridors as well. To avoid interrupting workers in the office during working hours, robots will choose the longer path and pass through the corridor. When at night, robots prefer the shortest route and pass through the office. Lin *et al.* [35] take the terrain dangerousness into account when planning the path. Wang *et al.* [36] leverage the semantic information to decide the possibility of managing to find target objects.

In this paper, we propose a semantic SLAM framework for rescue robots to better navigate through challenging environments that comprise complex terrains other than flat ground.

III. METHOD

To generate dense point-clouds augmented with point-wise semantic labels, our pipeline contains three parts: 1) an RGB-D SLAM frontend, 2) a convolutional neural network for semantic segmentation, and 3) filters for semantic labels and maps. Fig. 2 illustrates our pipeline.

A. SLAM FRONTEND

To obtain a dense point-cloud map and the accurate pose of our robot, we use ORB-SLAM2 as the SLAM frontend. ORB-SLAM2 runs mainly three threads in parallel, namely the tracking thread, the local mapping thread, and the loop closing thread. In the tracking thread, the system processes incoming images, extracts feature points, and roughly estimates the pose of the camera. The local mapping thread generates local maps and optimizes the camera pose with local bundle adjustment. The loop closing thread detects the loop and rectifies the accumulating drift of pose estimation.

We utilize the camera poses estimated by ORB-SLAM2 to generate dense point-cloud maps from consecutive RGB-D images. To be more specific, let us denote the coordinates of an RGB-D pixel as $[u, v, d]$, where u, v are the RGB image coordinates, and d is the pixel depth value. The 3D coordinates of a pixel under the camera frame $[x_c, y_c, z_c]$ can be recovered as

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} \frac{u - c_x}{f_x} d \\ \frac{v - c_y}{f_y} d \\ d \end{bmatrix}, \tag{1}$$

where c_x, c_y, f_x, f_y are the camera intrinsic parameters. Its world coordinates $[x_w, y_w, z_w]$ are

$$\begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} = T^{-1} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}, \tag{2}$$

where T is the ORB-SLAM2 estimated camera pose with R being a 3×3 rotation matrix (representing the orientation of the camera), and t being a 3×1 translation matrix (representing the position of the camera)

$$T = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}. \tag{3}$$

Fig. 1a demonstrates the geometric point-cloud map generated by the SLAM frontend.

B. SEMANTIC SEGMENTATION

To extract semantic labels from RGB-D images, we formulate it as a supervised-learning problem and train a convolutional neural network based on the Inception-v3 [37] architecture.

The Inception-v3 is chosen based on an empirical evaluation with the other two models, which shows that

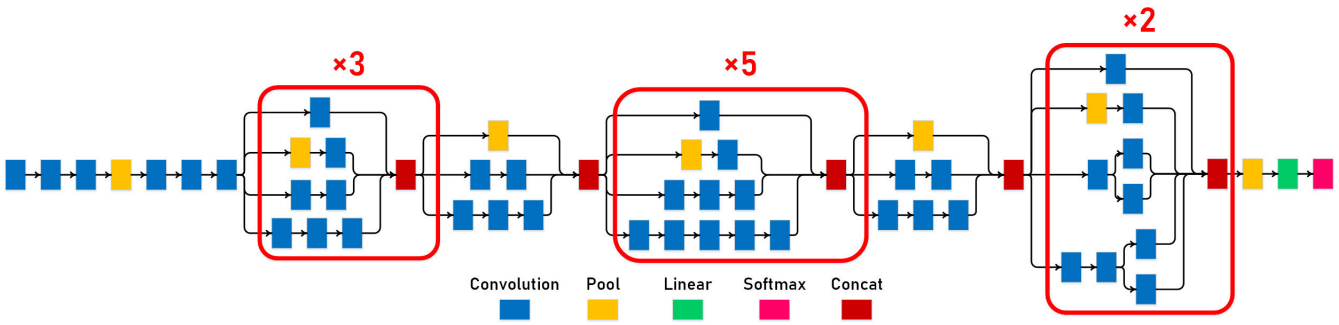


FIGURE 3. Structure of the semantic segmentation model Inception-v3.

Inception-v3 is best suited for this task and offers the best segmentation performance. Fig. 3 depicts the structure of Inception-v3. It utilizes various solutions to factorize convolutions and implement model regularization, so as to improve the computational efficiency and extract more features. For example, the Inception-v3 uses two small convolutions to replace a larger spatial filter, and it uses asymmetric convolutions to factorize convolutions into smaller ones. Also, label smoothing is used for regularization. All these efforts make Inception-v3 suitable for mobile robots with limited computing resources.

The dataset used to train the Inception-v3 network contains 5679 RGB-D images of an RRL test field, covering most of the interested terrains and objects with eight classes. The dataset contains eight classes: stairs, stepfields, slopes, elevated slopes, steep ramps, victim, ground, and background.

The RGB-D images in the dataset are collected with a realsense D435 camera mounted on our rescue robot. The resolution of the RGB-D images is set to 640 × 480. The file format of the dataset is the same as that of Cityscapes [38], which is a public and commonly used semantic segmentation dataset focused on city scenes. Fig. 4 shows some examples of our dataset.

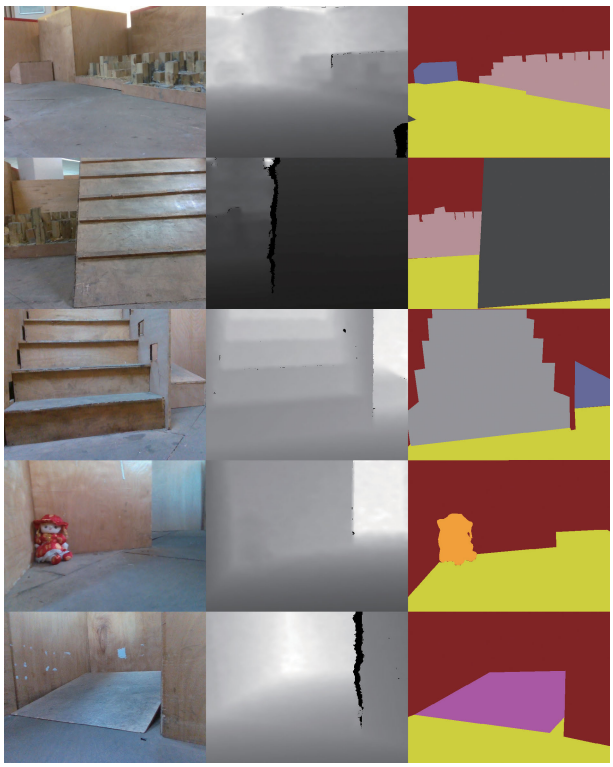


FIGURE 4. Dataset of the RRL competition environment. Left: RGB images, middle: depth images, right: semantic labels.

C. FILTERING SEMANTIC LABELS AND MAPS

Segmentation results provided by CNN often contain scattered labels of wrong classes. To filter out such labels, we perform a flood-fill operation with the depth images, similar to the work of Chen [22].

To identify the erroneously labeled pixels, we first perform an operation called ''erosion''. Let us denote the semantic label of a pixel as $L(x, y)$, where x, y is the pixel coordinate. If $L(x, y)$ is different from one of its nearby pixel labels, we re-label it as 0, assuming it belongs to no class.

Pixels labeled with 0 are then corrected by a procedure called ''flood-fill'', which determines the proper label by considering the depth value differences among the neighboring pixels. For a pixel $S(x, y)$ with label 0, if the depth value difference D_d between this pixel and its nearby pixel is less than a threshold d, the nearby pixel will be marked. After searching all pixels near $S(x, y)$, we modify the value of $L(x, y)$ to the label of the marked pixel with minimum D_d. As a result, most of the scattered labels of wrong classes are filtered out. Alg. 1 shows the detailed procedure of the flood-fill process.

Fig. 5 demonstrates the process of flood-fill. We can see that pixels in one object do not have great depth differences, so erroneously segmented pixels in one object should be marked as this object. As for pixels at the edges of objects, their depth differences with the same object are often smaller than those with neighboring objects. Therefore, their semantic labels will be corrected as well.

To further improve the consistency of the semantic labels, our approach performs temporal filtering by fusing the segmentation results of multiple frames. This is achieved by

Algorithm 1 Flood-Fill

Input: Semantic image I_s, corresponding depth image I_d
Output: Optimized semantic image I'_s
n is the scale of neighboring pixel.
$L(x, y)$ is the label of pixel $S(x, y)$ in I_s.
$D(x, y)$ is the depth value of pixel in I_d.
d is the threshold of depth difference.
for each pixel $S(x, y) \in I_s$ **do**
  for $(x - n) < x_0 < (x + n)$,$(y - n) < y_0 < (y + n)$ **do**
    if $L(x, y) \neq L(x_0, y_0)$ **then**
      $L(x, y) = 0$
      break
    end if
  end for
end for
for each pixel $S(x, y)$ with $L(x, y) = 0$ **do**
  for $(x - n) < x_0 < (x + n)$,$(y - n) < y_0 < (y + n)$ **do**
    if $d(x_0, y_0) = |D(x, y) - D(x_0, y_0)| < d$ **then**
      mark $S(x_0, y_0)$ and $d(x_0, y_0)$
    end if
  end for
  for each marked $d(x_0, y_0)$ **do**
    find the minimum $d(x_0, y_0)$, i.e. $d_{min}(x_0, y_0)$ and corresponding $S_{min}(x_0, y_0)$
  end for
  Let $L(x, y) = L_{min}(x_0, y_0)$, where $L_{min}(x_0, y_0)$ is the label of $S_{min}(x_0, y_0)$
end for

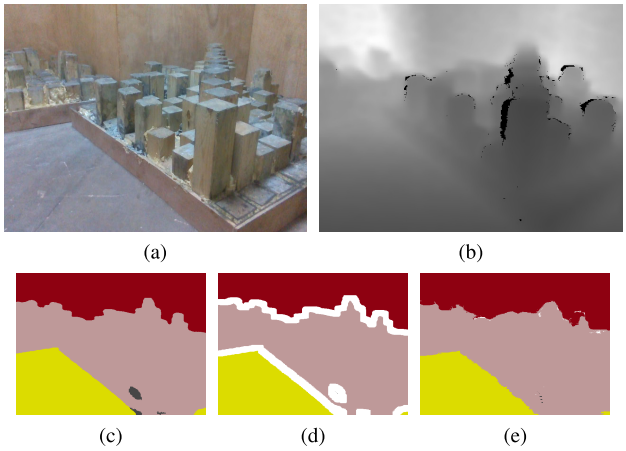


FIGURE 5. Flood-fill operation. (a) Raw RGB image for segmentation. (b) Corresponding depth image. (c) Direct segmentation result, which is referred as a semantic image. (d) Eroded segmentation result, where pixels at the boundary or labeled as erroneous classes are marked. (e) Result of flood-fill operation. By considering the depth difference between neighboring pixels, we can decide whether one pixel should be marked with the same label as its neighbors'.

firstly voxelizing the point-cloud into a grid with a voxel size of 0.08 m × 0.08 m × 0.08 m, then, performing a ''winner-takes-all'' operation for every voxel, i.e., all points in the same voxel will take the most frequently appeared label as their label.

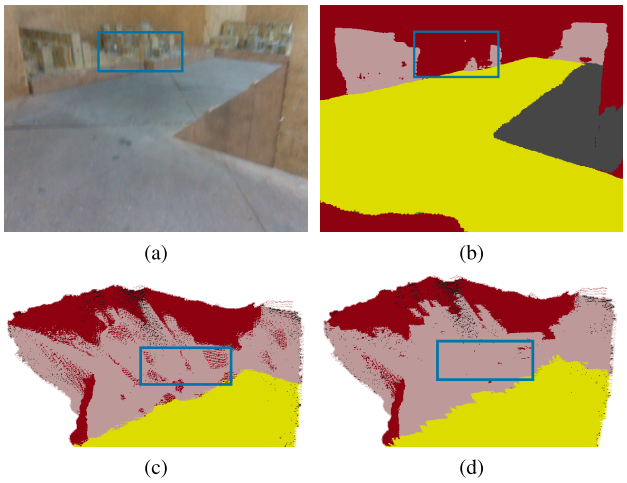


FIGURE 6. Multi-frame filtering. (a) The RGB image of the first frame. (b) The semantic segmentation result of the first frame. (c) A simple merging of ten point-clouds. (d) The resulting after filtering.

Fig. 6 provided an example result of filtering point-clouds of ten frames/timesteps. Objects in blue rectangular boxes belong to a stepfeild region. When simply merging the ten sub-point-clouds (as shown in Fig. 6c), the result is very noisy and contains lots of point-cloud labeled as the background instead of ''stepfield''. The mislabeled points mostly come from the first frame because the camera vibrates sharply when the record begins, leading to a blurry image and hence a noisy segmentation result. But the camera movements regain steadiness after the first frame, and so the segmentation accuracy of the next nine frames recovers thereafter.

Fig. 6d depicts the result after the voxelization and ''winner-takes-all'' operation. Since the inlier labels outnumbered the outliers, the incorrect labels are greatly reduced after the filtration.

D. PATH PLANNING WITH SEMANTIC INFORMATION

Standard path planning methods often consider trajectory length as one – and the only one – criterion, i.e. the shortest path is most preferred. However, in a complex environment consists of challenging terrains, factors like traversability of different terrains should also be considered. This can be achieved when the semantic label of every map points are available to us.

To perform path planning with the semantic augmented point-cloud, we consider the fast marching level set method (FFM) [39], which can generate global optimum solutions with low computational complexity. Given a velocity map, FFM works by simulating the front propagation of a curve moving in its normal direction, where the moving speed of the curve only depends on the local position. The arrival time from the start point to every position in the map can also be estimated.

To apply the FFM method, a velocity map which determines the traversing speed everywhere in a certain environment is needed. Inspired by the work of Gao et al [40], we propose an improved velocity function, which considers

both Euclidean signed distance field (ESDF) and the terrain types (i.e. semantic labels) to generate a velocity map for the RRL competition environment. To be more specific, the velocity function is set to be

$$f(d) = \begin{cases} (1 - c_i) \cdot v_m \cdot (tanh(d - e) + 1)/2, & 0 \leq d, \\ 0, & d < 0, \end{cases} \quad (4)$$

where

- d is the distance to its nearest obstacle;
- $c_i \in [0, 1]$ is the time cost factor of passing terrains of type i, which reflects the traversability of traversing such terrain. The higher the c_i is, the slower the robot will move through;
- e is the Euler's number;
- v_m is the maximum velocity of our robot;
- $tanh()$ is the hyperbolic tangent function.

Given this velocity function and a voxelized point-cloud of the environment, the corresponding velocity map is determined, which in turn determines the arrival time map using FFM. Then, a path connecting a given pair of starting and target points with minimal arrival time can be extracted by performing a gradient descent on the arrival time map.

IV. EXPERIMENTS

Our experiments are designed to validate that our method is able to 1) generate the semantic map of the RRL competition environment, 2) improve the path planning algorithm by using semantic information.

A. SEMANTIC SEGMENTATION

We use our RRL competition environment dataset to train semantic segmentation CNN Inception-v3. This dataset has 3407 images for training, 1136 images for validation, and 1136 images for testing, which involves eight classes in the RRL competition environment. The CNN model is pretrained with the Cityscapes dataset before we train it with our own dataset.

When training this model, the batch size is set to 4, the momentum rate is set to 0.9, and the learning rate is 1×10^{-4}. To avoid overfitting, we add 0.4 dropout rate and 5×10^{-5} weight decay rate to increase the generalization ability of models and decrease the complexity of model weights, respectively. The training equipment is a desktop with Intel Core i7-4790K 4.00GHz CPU, GeForce GTX 1070 GPU, and 15.6GB RAM.

After trained for 19 epochs, this model achieves the performance of 96.87 % Mean IoU, 98.99 % Mean Accuracy, and 98.93 % Mean Recall in the testing set. In the meanwhile, the inference time for each frame is 0.038 seconds. The results demonstrate that Inception-v3 is capable of implementing accurate and fast segmentation in our dataset.

We also attempt to train ERFNet [41] and MobileNetV2 [42] with our dataset, but the results are below that of Inception-v3. As for every model, we implement the same times of iteration, which is 16K in this experiment, and before the iteration stops, they have all converged, so we can assure that they have the best performance with the current training parameters. Table. 3 shows the training parameters of every model.

FIGURE 7. Example results of segmentation test. Left: RGB images, middle: segmentation result, right: ground truth.

TABLE 1. Training parameters.

model	*dropout*	*learning rate*	*weight decay*
ERFNet	0.4	1×10^{-4}	1×10^{-5}
Inception-v3	0.4	1×10^{-4}	5×10^{-5}
MobileNetV2	0.1	1×10^{-3}	1×10^{-6}

TABLE 2. Performance of each network.

model	*mean IoU%*	*mean ACC%*	*inference time (sec)*
ERFNet	0.9612	0.9879	**0.033**
Inception-v3	0.9687	0.9899	0.038
MobileNetV2	**0.9735**	**0.9912**	0.072

The accuracy and inference time of every model on the testing set are demonstrated in Table. 4. From this table, we can know that MobileNetV2 takes too much time for inference, while the segmentation accuracy of ERFNet is slightly lower. After taking both accuracy and inference time into account, we utilize Inception-v3 to train with our dataset and obtain satisfying segmentation results. Fig. 7 shows some examples of semantic segmentation results.

B. PATH PLANNING WITH SEMANTIC INFORMATION

To validate the effectiveness of the semantic information in path planning, we carry out an experiment in our RRL competition environment. As shown in Fig. 8a, on top of

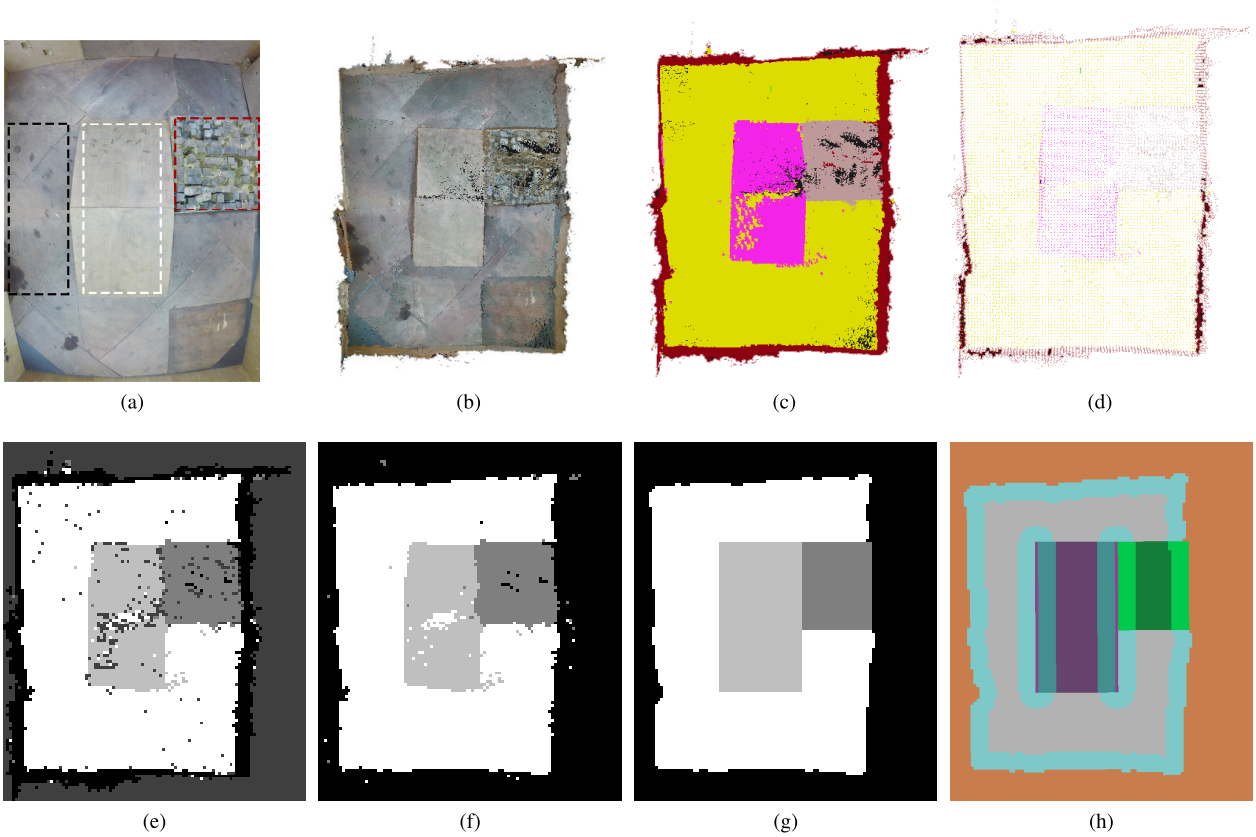


FIGURE 8. Semantic map preprocessing. (a) A picture of the test field (view from above). There is a slope/hill in the middle and a stepfield to the right. We are interested in three pathways marked with black, white and red boxes respectively, referred as path A, B, and C. (b) The geometric map in the form of a colored point-cloud. (c) The same point-cloud with semantic labels. (d) Downsampled point-cloud. (e) The original grid map. (f) The grid map after filling cells not scanned. (g) The grid map after filtering. (h) The grid map after inflation, where purple lines are edges of slopes that should not be traversed, and the blue part is the boundary of obstacles after inflation.

the flat ground, there are two continuous slopes next to a stepfield.

Before the experiment, a point-cloud (and its semantic labels) of the field is obtained using the proposed semantic SLAM framework (see Fig. 8c). To perform path planning on this point-cloud, we performed several preprocessing operations. First, the 3D point-cloud is downsampled by voxelization and projected to the ground plane to generate a 2D grid map, i.e. Fig. 8d and Fig. 8e. Such a grid map has a cell size of 0.05 m $\times$ 0.05 m. Then, salt and pepper noise of small scale in the grid map is removed using their neighboring cells, leading to a refined map as shown in Fig. 8g. Finally, we make the inflation at the edge of obstacles and some terrains in the grid map to broaden their ranges, so that the robot will not get too close to the edge and fall or collide. The inflation process is realized automatically by leveraging the function in costmap_2d package of Robot Operating System (ROS). For the slopes, the robot can only traverse the green lines and avoid the red lines in Fig. 9a, because the robot will fall and turn over from the top of the slopes at the red lines, see Fig. 9b. To find out red lines that need to be inflated, we utilize the normals of cells belonging to the slopes in the grid map, where the normals can be obtained from the 3D semantic map. If the edges of the slopes are parallel to the projection of normals onto a horizontal plane, the edges are red lines. Otherwise, the edges are green lines if they are perpendicular to the projection. Fig. 8h is the final semantic map where the path planning takes place.

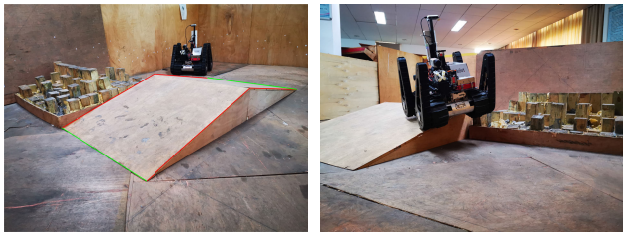
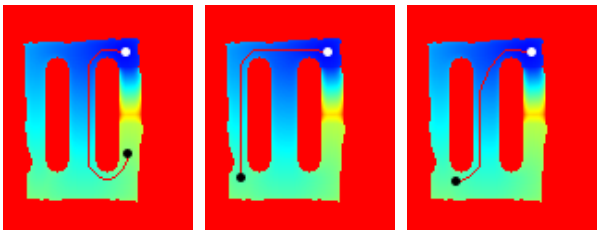


FIGURE 9. Path planning environment observed from different angles. (a) Environment observed from the top left corner. It is dangerous for the robot to traverse the red lines, so we must distinguish them from boundaries and avoid them. (b) Environment observed from the bottom right corner, and the robot will fall when traversing the red line of slopes.

Given the semantic grid map, a velocity map is generated using (4) with terrain time cost factors c_i from Table. 3, where the time cost factors are determined by the difficulty and the danger level for robots to traverse the corresponding terrains.

TABLE 3. Time cost factor of each terrain.

terrain	*background*	*slopes*	*steep ramps*	*elevated slopes*	*stepfields*	*stairs*	*victim*	*ground*
time cost factor	1	0.25	0.35	0.5	0.85	0.6	1	0



(a) (b) (c)

FIGURE 10. Three path planning trials with different goal points. In (a), the robot avoids the stepfield. In (b), the robot traverses the same distance in different paths to reach the goal point. And in (c), the robot takes less time when traversing slopes to reach the goal point. The white points and the black points are the start points and the goal points respectively.

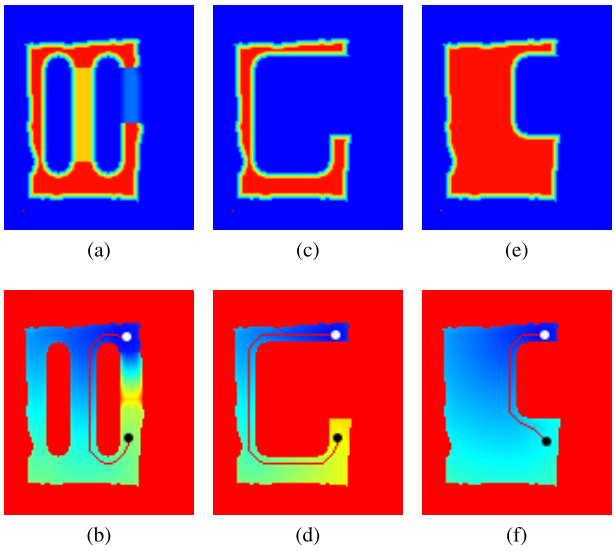
The resulting velocity map is shown in Fig. 11a, in which the maximum speed of the robot is 1 m/s. With the velocity map, the arrival time from the start to the goal can be calculated by using the FMM algorithm introduced in Sec. III-D. The path with the minimum time cost is also determined by looking for the steepest gradient descent direction.

Fig. 10 shows the result of three path planning trials with different goal points. We can see that the robot is able to choose a suitable path for different goal points. If the path lengths that the robot needs to traverse in different routes are the same, the robot will choose path A but not path B consisting of slopes because of the lower time cost factor of the ground, see Fig. 10b. As for the goal point in Fig. 10b, the robot will take 0.52 more seconds to reach by traversing the slopes than traversing path A. To reach the goal point in Fig. 10c, it will save 0.36 seconds to traverse the slopes because the route is shorter and the time cost factor of the slopes is relatively low.

In all three trials, the robot is aware of the terrain class given the semantic information, and hence can make use of the fact that the slopes are neither obstacles nor flat ground, but terrains with a slightly higher time cost factor. In the first trial, it is faster for the robot to traverse path B because path B has low time cost factor compared to the stepfield in path C. As for path A, it requires the robot to pass two more 1.2 m $\times$ 1.2 m blocks than traversing path B. Thus, after considering the high time cost factor of path C and the long path length of traversing path A, the robot chooses path B as the route to reach the goal point, see Fig. 10a.

As a comparison, we conducted an experiment with semantic information being neglected in two cases: i) both stepfields and slopes are regarded as obstacles and, ii) stepfields are regarded as obstacles, but slopes are regarded as passable paths the same as flat ground.

In the first case, the robot neglects the slopes for a shortcut and traverses path A with longer arrival time, as can be seen in Fig. 11c and Fig. 11d. Table. 4 shows the arrival time of



(a) (c) (e)

(b) (d) (f)

FIGURE 11. Velocity maps and arrival time maps using three different settings. The second row shows arrival time maps, where the warmer the color is, the longer time the robot needs to reach the cell from the start point to the goal point.

TABLE 4. The arrival time of different routes.

routes	*Fig. 10b*	*Fig. 10c*	*Fig. 11b*	*Fig. 11d*	*Fig. 11f*
arrival time	5.57s	5.63s	6.49s	8.38s	4.27s

different routes, where the arrival time of the route in Fig. 11d is 8.38 seconds, and it is 1.89 seconds longer than that of traversing the slopes (see Fig. 11b).

In the second case, the robot is too greedy and not aware of the dangerous edges. To traverse the path with minimum time cost, the robot may pass the red line and fall with a high chance, see Fig. 9b, Fig. 11e, and Fig. 11f. Although traversing the red line will reduce 2.22 seconds and 4.11 seconds than traversing the green line and path A respectively, the risk of the robot falling is high.

To sum up, from the perspective of topology, it is also beneficial to combine semantic information with path planning. With semantic information, the slopes and the stepfields will be regarded as terrains that are passable but different from the ground. In this way, paths that go through path A, B, and C in Fig. 8a respectively belong to three different homotopy classes. However, if we regard the slopes and the stepfields as the obstacles, the generated paths will only belong to one homotopy class, and the other classes are completely removed. Moreover, if we regard the slopes as the ground and the stepfields as the obstacles, the whole environment will become a connected graph, so there will be no difference between path A and path B. In this way, the number of homotopy classes will also be one, and it neglects the difference of terrains, so there is a strong possibility that the path goes

through the dangerous edges of terrains, just as Fig. 9b. Thus, semantic information is invaluable in path planning.

V. CONCLUSION

In this paper, we propose a semantic RGB-D SLAM system for rescue robots. This system can generate dense point-cloud maps with semantic information by fusing semantic segmentation CNN and RGB-D SLAM frontend. We utilize the depth information to determine whether neighboring pixels in the semantic image belong to the same object, so as to improve the precision of semantic segmentation. We also use ''winner-takes-all'' to improve the precision of the semantic map by leveraging point-cloud generated by multiple frames. Semantic information about surroundings plays an important role in helping robots understand the environment and implement path planning. We demonstrate how semantic information can optimize path planning and help to generate paths with the minimum time cost. To validate our semantic SLAM system, we generate an RGB-D semantic dataset of the RRL competition environment. Experimental results prove that our system can generate accurate dense semantic maps and leverage semantic information to improve the path planning results. In future work, we plan to fuse the semantic information and SLAM system tightly, such as utilizing semantic information to improve the accuracy of pose estimation and leveraging the point-cloud map to improve the semantic segmentation precision.

REFERENCES

[1] R. Mur-Artal and J. D. Tardos, ''ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras,'' *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[2] J. Long, E. Shelhamer, and T. Darrell, ''Fully convolutional networks for semantic segmentation,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.

[3] V. Badrinarayanan, A. Kendall, and R. Cipolla, ''SegNet: A deep convolutional encoder-decoder architecture for image segmentation,'' *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017.

[4] H. Zhang, K. Dana, J. Shi, Z. Zhang, X. Wang, A. Tyagi, and A. Agrawal, ''Context encoding for semantic segmentation,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7151–7160.

[5] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, ''Semantic image segmentation with deep convolutional nets and fully connected CRFs,'' 2014, *arXiv:1412.7062*. [Online]. Available: https://arxiv.org/abs/1412.7062

[6] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, ''DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs,'' *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2018.

[7] H. Wu, J. Zhang, K. Huang, K. Liang, and Y. Yu, ''FastFCN: Rethinking dilated convolution in the backbone for semantic segmentation,'' 2019, *arXiv:1903.11816*. [Online]. Available: http://arxiv.org/abs/1903.11816

[8] K. He, X. Zhang, S. Ren, and J. Sun, ''Deep residual learning for image recognition,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[9] Y. Xing, J. Wang, X. Chen, and G. Zeng, ''2.5D convolution for RGB-D semantic segmentation,'' in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 1410–1414.

[10] X. Qi, R. Liao, J. Jia, S. Fidler, and R. Urtasun, ''3D graph neural networks for RGBD semantic segmentation,'' in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5199–5208.

[11] Y. Nakajima, B. Kang, H. Saito, and K. Kitani, ''Incremental class discovery for semantic segmentation with RGBD sensing,'' in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 972–981.

[12] S. Lee, S.-J. Park, and K.-S. Hong, ''RDFNet: RGB-D multi-level residual feature fusion for indoor semantic segmentation,'' in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 4980–4989.

[13] Y. He, W.-C. Chiu, M. Keuper, and M. Fritz, ''STD2P: RGBD semantic segmentation using spatio-temporal data-driven pooling,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4837–4846.

[14] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, ''RangeNet++: Fast and accurate LiDAR semantic segmentation,'' in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 4213–4220.

[15] M. Labbé and F. Michaud, ''RTAB-map as an open-source LiDAR and visual simultaneous localization and mapping library for large-scale and long-term online operation,'' *J. Field Robot.*, vol. 36, no. 2, pp. 416–446, Mar. 2019.

[16] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger, ''ElasticFusion: Real-time dense SLAM and light source estimation,'' *Int. J. Robot. Res.*, vol. 35, no. 14, pp. 1697–1716, Dec. 2016.

[17] L. Xiao, J. Wang, X. Qiu, Z. Rong, and X. Zou, ''Dynamic-SLAM: Semantic monocular visual localization and mapping based on deep learning in dynamic environment,'' *Robot. Auton. Syst.*, vol. 117, pp. 1–16, Jul. 2019.

[18] K. Tateno, F. Tombari, I. Laina, and N. Navab, ''CNN-SLAM: Real-time dense monocular SLAM with learned depth prediction,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6243–6252.

[19] X. Li and R. Belaroussi, ''Semi-dense 3D semantic mapping from monocular SLAM,'' 2016, *arXiv:1611.04144*. [Online]. Available: https://arxiv.org/abs/1611.04144

[20] V. Vineet, O. Miksik, M. Lidegaard, M. Nießner, S. Golodetz, V. A. Prisacariu, O. Kahler, D. W. Murray, S. Izadi, P. Peerez, and P. H. S. Torr, ''Incremental dense semantic stereo fusion for large-scale semantic scene reconstruction,'' in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2015, pp. 75–82.

[21] Y. Yang, F. Qiu, H. Li, L. Zhang, M.-L. Wang, and M.-Y. Fu, ''Large-scale 3D semantic mapping using stereo vision,'' *Int. J. Autom. Comput.*, vol. 15, no. 2, pp. 194–206, Apr. 2018.

[22] X. Chen, A. Milioto, E. Palazzolo, P. Giguère, J. Behley, and C. Stachniss, ''SuMa++: Efficient LiDAR-based semantic SLAM,'' in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 4530–4537.

[23] R. Dubé, A. Cramariuc, D. Dugas, J. Nieto, R. Siegwart, and C. Cadena, ''SegMap: 3D segment mapping using data-driven descriptors,'' 2018, *arXiv:1804.09557*. [Online]. Available: http://arxiv.org/abs/1804.09557

[24] X. Chen, T. Läbe, A. Milioto, T. Röhling, O. Vysotska, A. Haag, J. Behley, and C. Stachniss, ''OverlapNet: Loop closing for LiDAR-based SLAM,'' in *Proc. Robot., Sci. Syst. (RSS)*, 2020.

[25] Y. Dang, P. Chen, R. Liang, C. Huang, Y. Tang, T. Yu, X. Yang, and K.-T. Cheng, ''Real-time semantic plane reconstruction on a monocular drone using sparse fusion,'' *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7383–7391, Aug. 2019.

[26] S. L. Bowman, N. Atanasov, K. Daniilidis, and G. J. Pappas, ''Probabilistic data association for semantic SLAM,'' in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2017, pp. 1722–1729.

[27] J. McCormac, A. Handa, A. Davison, and S. Leutenegger, ''SemanticFusion: Dense 3D semantic mapping with convolutional neural networks,'' in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2017, pp. 4628–4635.

[28] Y. Nakajima, K. Tateno, F. Tombari, and H. Saito, ''Fast and accurate semantic mapping through geometric-based incremental segmentation,'' in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 385–392.

[29] M. Antonello, D. Wolf, J. Prankl, S. Ghidoni, E. Menegatti, and M. Vincze, ''Multi-view 3D entangled forest for semantic segmentation and mapping,'' in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2018, pp. 1855–1862.

[30] C. Yu, Z. Liu, X.-J. Liu, F. Xie, Y. Yang, Q. Wei, and Q. Fei, ''DS-SLAM: A semantic visual SLAM towards dynamic environments,'' in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 1168–1174.

[31] N. Sunderhauf, T. T. Pham, Y. Latif, M. Milford, and I. Reid, ''Meaningful maps with object-oriented semantic mapping,'' in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 5079–5085.

[32] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, ''SSD: Single shot multibox detector,'' in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 21–37.

[33] C. Wang, D. Zhu, T. Li, M. Q.-H. Meng, and C. W. de Silva, ''Efficient autonomous robotic exploration with semantic road map in indoor environments,'' *IEEE Robot. Autom. Lett.*, vol. 4, no. 3, pp. 2989–2996, Jul. 2019.

[34] N. Sunderhauf, F. Dayoub, S. Mcmahon, B. Talbot, R. Schulz, P. Corke, G. Wyeth, B. Upcroft, and M. Milford, "Place categorization and semantic mapping on a mobile robot," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2016, pp. 5729–5736.
[35] Y.-H. Lin, Y.-S. Liu, G. Gao, X.-G. Han, C.-Y. Lai, and M. Gu, "The IFC-based path planning for 3D indoor spaces," *Adv. Eng. Informat.*, vol. 27, no. 2, pp. 189–205, Apr. 2013.
[36] C. Wang, J. Cheng, W. Chi, T. Yan, and M. Q.-H. Meng, "Semantic-aware informative path planning for efficient object search using mobile robot," *IEEE Trans. Syst., Man, Cybern. Syst.*, early access, Oct. 14, 2019, doi: 10.1109/TSMC.2019.2946646.
[37] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826.
[38] 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 *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3213–3223.
[39] J. A. Sethian, "A fast marching level set method for monotonically advancing fronts," *Proc. Nat. Acad. Sci. USA*, vol. 93, no. 4, pp. 1591–1595, Feb. 1996.
[40] F. Gao, W. Wu, Y. Lin, and S. Shen, "Online safe trajectory generation for quadrotors using fast marching method and bernstein basis polynomial," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2018, pp. 344–351.
[41] E. Romera, J. M. Alvarez, L. M. Bergasa, and R. Arroyo, "ERFNet: Efficient residual factorized ConvNet for real-time semantic segmentation," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 1, pp. 263–272, Jan. 2018.
[42] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.

WENBANG DENG received the B.E. degree from Central South University, China, in 2018. He is currently pursuing the M.E. degree with the National University of Defense Technology, China. His research interests include deep leaning, simultaneous localization and mapping, and mobile robots.

KAIHONG HUANG received the B.E. degree from the University of Electronic Science and Technology of China, in 2011, the M.E. degree from the National University of Defense Technology, China, in 2013, and the Ph.D. degree from the University of Bonn, Germany, in 2018. His research interests include robotics perception and state estimation, simultaneous localization and mapping, and sensor calibration.

XIEYUANLI CHEN received the bachelor's degree in electrical engineering and automation from Hunan University, China, in 2015, and the master's degree in robotics from the National University of Defense Technology, China, in 2017. He is currently pursuing the Ph.D. degree with the Photogrammetry and Robotics Laboratory, University of Bonn. His research interests include localization, mapping, and SLAM. He is also a member of the Technical Committee of RoboCup Rescue Robot League.

ZHIQIAN ZHOU received the B.E. degree from the National University of Defense Technology, China, in 2017, where he is currently pursuing the Ph.D. degree. His research interests include mobile robotics, especially on social robot, soccer robot, and rescue robot.

CHENGHAO SHI received the B.E. degree from the Nanjing University of Aeronautics and Astronautics, China, in 2017, and the M.E. degree from the National University of Defense Technology, China, in 2020, where he is currently pursuing the Ph.D. degree. His research interests include localization and mobile robots.

RUIBIN GUO received the B.S. degree in communication engineering, the M.S. degree in information and communication engineering, and the Ph.D. degree from the National University of Defense Technology, China, in 2012, 2014, and 2019, respectively. He is currently a Lecturer with the National University of Defense Technology. His research interests include computer vision, simultaneous localization and mapping, and 3D reconstruction.

HUI ZHANG received the B.E., M.E., and Ph.D. degrees from the National University of Defense Technology, China, in 1993, 1996, and 2000, respectively. He became a Professor at the National University of Defense Technology, in 2011. His research interests include mobile robots, simultaneous localization and mapping, and intelligent control.