IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received April 22, 2019, accepted May 26, 2019, date of publication June 3, 2019, date of current version June 18, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2920511

Efficient Planar Surface-Based 3D Mapping Method for Mobile Robots Using Stereo Vision

BINGHUA GUO[1], **HONGYUE DAI**[1], **ZHONGHUA LI**[2], **AND WEI HUANG**[1]

[1]Department of Electronics and Information Engineering, University of Zhaoqing, Zhaoqing 526061, China
[2]School of Information Science and Technology, Sun Yat-sen University, Guangzhou 510006, China

Corresponding author: Wei Huang (weihzqu@163.com)

This work was supported in part by the Foundation of Guangdong Educational Committee under Grant 2014KTSCX191, and in part by the National Natural Science Foundation of China under Grant 61201087.

ABSTRACT Environmental mapping plays an important role in the field of robotics. Conventional voxel-based occupancy grid models largely reduce system mapping efficiency for huge quantities of grid cells. This paper presents an efficient method of 3D grid modeling using stereo vision based on planar surfaces. This method first uses matching key feature points with a multi-random sample consensus algorithm to estimate plane parameters and then clusters pre-processed point cloud data located on the same plane. Next, a split and combining algorithm is used to generate 3D planar grid approximation representations of the environment. The occupancy probabilities of grid cells are estimated and updated by using the Kullback–Leibler divergence. Finally, a series of experiments including map qualitative analysis and performance tests, are adopted to evaluate the presented method in indoor and outdoor environments. The results of the experiments and performance evaluation illustrate the capabilities of our approach in generating efficient 3D maps.

INDEX TERMS 3D mapping, planar surface, mobile robots, stereo vision, occupancy grid models.

I. INTRODUCTION

Environmental mapping plays an important role in the application of robotics, such as environment exploration, simultaneous localization and mapping (SLAM), and path planning. The robots use sensors (e.g., 3D lasers, RGB-D cameras, stereo vision systems, omni-directional thermal camera, etc.) [1]–[3] to build environment maps, which provide them essential information.

In the last few decades, with the application of 3D sensors, especially low-cost stereo vision systems that use disparity to obtain depth information [4], many researchers have proposed 3D mapping approaches [5]–[7] to expand robot applications that will enable robots to adapt to complex environments and perform other useful tasks (e.g., local localization). The major 3D approaches for the robots are point cloud methods and voxel-based grid methods, each with their advantages and disadvantages. Point cloud models present higher accuracy of environment modeling than voxel-based grid methods but involve the processing of huge amounts of data, which greatly reduces the system efficiency. Point cloud data are also easily influenced by noise [8]. By contrast, voxel-based occupancy grid methods discretize the environment into voxel cells and seek to determine whether each cell is occupied, unoccupied, or unknown. Their computational resource and memory requirements are less than those of point cloud models. However, when robots work in complex or outdoor environments, the same fixed-size voxel models greatly reduce computational efficiency and increase memory cost. Adaptive rectangular cuboids [9] improve mapping efficiency to a certain extent, but they use axis-aligned combining rectangular cuboids instead of the standard cubic grid cell assumption common to all occupancy grid cells. If the grid cells do not follow this rule of axis alignment (e.g., surfaces of objects in the environment are tilted at a certain angle from the coordinate axis), it would be incapable of dealing with such combination processing. However, the points of the surface scanned by sensors are located on the same planes, and if planar cells with the same angles are used as representations, then they can be conveniently combined and the number of cells for representing the planes will be significantly reduced. In practice, many such objects locate in the working environment of mobile robots, especially structured indoor environments.

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

Intuitively speaking, this condition will significantly reduce the number of cells by using planar cells to model the 3D environment.

This paper presents a method for using planar cells partially instead of equal-sized cubic voxels and extends the standard occupancy for mapping a 3D environment. Specifically, this paper contributes the following aspects:

- Instead of the cubic voxel, a planar grid with different angles is proposed to partially represent the 3D environment.
- A planar grid combination algorithm is proposed to reduce the number of grid cells in the 3D environment and effectively improve the mapping efficiency.
- A local explicit modeling space is proposed for the occupancy grid approach. This modeling space brings more benefits to some applications of the robot.

The remainder of this paper is organized as follows. We first give an overview of related work in Section II. Then, we describe the proposed planar occupancy grid mapping method in Section III and provide a detailed explanation for each part in subsections III-A to F. We present some experiments to evaluate the performances of our proposed method in Section IV. Finally, we finish with a conclusion in Section V.

II. RELATED WORK

Volumetric occupancy mapping is a general approach for 3D world representation for mobile robots and is widely used to represent distributed spatial information. It discretizes the environment in cells with a given resolution, with each cell represented by a probabilistic belief about its state (e.g., free or occupied).

Some recent examples of volumetric occupancy mapping methods can be found in [10], [11], where the authors generate 3D occupancy maps (or evidence grid) from the RGB-D camera or stereo visual data, and a signed distance function is used to store a distance to a surface in a discretized voxel grid, which represents the world geometry. Occupancy maps represent areas of occupied, free, or unknown space and are well suited to fusing data from multiple sources. These maps have been extensively used in robotics because of their simplicity and suitability for decision-theoretic approaches. However, they are not scaled well with the environment's dimension, which consumes too much memory to represent high-image resolutions across all space, and their real-time performance often requires high-performance graphics hardware (e.g., GPU).

Traditional approaches of occupancy maps assume that the occupancy probability of each cell in a map is an independent random variable, an assumption that makes traditional methods simple and fast. Recently, a Gaussian process occupancy mapping has relaxed this assumption [12] and has taken into account the latent function variables of Gaussian processes, making it desirable for producing accurate and continuous occupancy maps. However, the computational complexity of predicting Gaussian processes for mean and variance is not directly applicable to large-scale environments. Kim *et al.* [13] proposed a framework for building occupancy maps and reconstructing surfaces for large-scale mapping using a Gaussian process classification. They partitioned the training data and the test data with a coarse-to-fine clustering method and applied local Gaussian processes to each small subset of data to reduce the computational time. However, the high-storage complexity is still its main drawback.

In [14], [15], an octree-based 3D map representation was proposed that shows the memory requirements of the voxel grid could be greatly reduced. Octree-based 3D map representation is currently the state-of-the-art 3D representation. The octree is a hierarchical data structure for spatial subdivision and is a collection of nodes that discretize a space into voxels, which are then recursively subdivided into eight sub-volumes or child nodes until a desired resolution is reached. Schauwecker and Zell [16] extended the application of octree-based 3D maps constructed from data captured using costly 3D laser scanners to a low-cost stereo vision system.

Most occupancy grid approaches use cubic grid cells for 3D representation. Khan *et al.* [9] presented a rectangular cuboid approximation framework (RMAP) for 3D mapping. They used axis-aligned rectangular cuboids to model the large-scale 3D environment, wherein computational and memory efficiencies were largely reduced compared with those in octree-based 3D maps. Later, in [17], they extended previous works by adding a fusion process that modifies the resolution of grid cells in an incremental manner. Their evaluation results show that the fusion process effectively reduces the number of grid cells required by the occupancy grid. However, this method does not adapt to combining non-axis-aligned surfaces of objects.

III. SYSTEM OVERVIEW

A. ROBOTIC MAPPING SYSTEM

Our robotic mapping systems are shown in Fig. 1. The mobile robot is equipped with a stereo camera pair and some motion control cards. The mapping structure consists of three main parts: stereo vision camera system, front-end visual processing system, and map construct system. The stereo vision camera system is used to scan the surrounding environment and acquire color and image pair information. The data captured are sent to the front-end visual processing system. A structure based on a digital signal processor (with DM642 of Ti Comp.) is adopted as the front-end visual processing system. The front-end visual processing system is mainly responsible for the binocular stereo visual computation and feature (e.g., color, intensity, etc.) extraction. The processed data are delivered to map construct system through the network, which is performed on an Intel Core i3 CPU with 2.40 GHz for 3D mapping.

B. APPROACH

In this paper, we propose a method using planar grid to represent the 3D environment. Fig. 2 shows the schematic

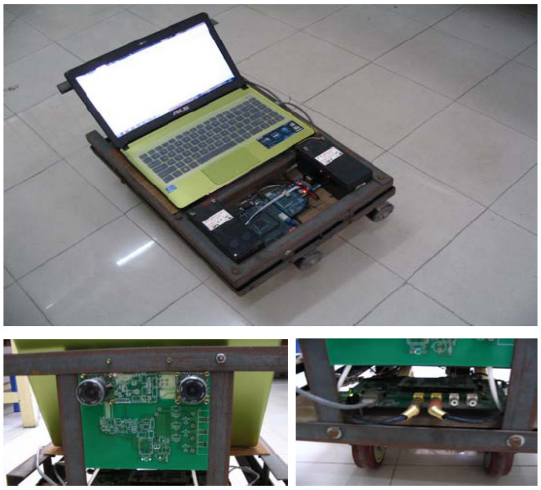


FIGURE 1. **(Top) Our mobile robot platform. (Bottom row) Binocular visual system and DM642 image processing card.**

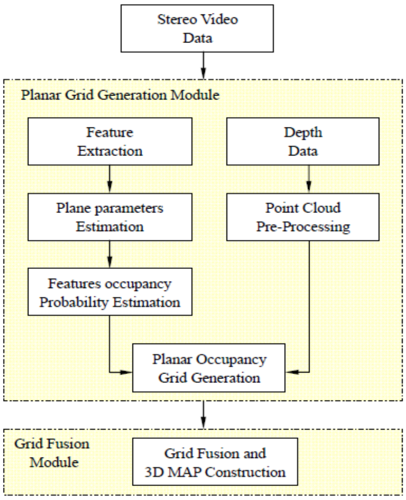

Stereo Video Data
Planar Grid Generation Module
Feature Extraction
Depth Data
Plane parameters Estimation
Point Cloud Pre-Processing
Features occupancy Probability Estimation
Planar Occupancy Grid Generation
Grid Fusion Module
Grid Fusion and 3D MAP Construction

FIGURE 2. **Schematic overview of our method. The system is divided into two main modules: generating occupancy planar grid and grid combining.**

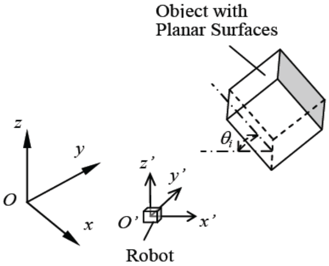


FIGURE 3. **Coordinate system of robots. The center of the robot is taken as the origin of the relative coordinate system.**

representation of the presented system. The system is divided into two main modules: generating planar grid and grid combining. In the generating planar grid module, the main task is to generate planar grid from point-cloud data. The front-end works consist of three parts: acquisition of point-cloud data, estimation of plane parameters, and estimation of features occupancy probability. The data from the three parts are transformed to planar occupancy grid using a suitable combining algorithm. The estimations of plane parameters and grid occupancy probability are based on the features extracted from frames to reduce the computation cost. In the grid combining module, grid cells located on the same plane are combined into large rectangular cuboids for 3D environment representations. The following sections describe the illustrated parts of the system.

C. PLANE ESTIMATION

The actual 3D world is composed of many objects with planar surface structures. However, the structures of non-axis-aligned surfaces (e.g., slope surfaces) require more grid cells to form if the standard cubic grid representation is used, which increases the access time. The planar grid representation is convenient for combining most of the planar surfaces of objects because it is located on the same plane and reduces the number of grid cells used. The planar grid cell can be described with a tilted angle relative to the ground and a grid cell size. As the tilted angles of the planar grid representation are consistent with the plane where the grid cells are located, the angles can be obtained by estimating the plane angle relative to the ground. The grid cell size is formed through the division and combination of the point cloud data (see subsection III-F "Grid Approximation and Combination Processing" for further details).

As shown in Fig. 3, we set up a ground coordinate system O-xyz and a robot coordinate system O'-x'y'z'. We consider the starting point of the robot motion as the origin of the ground coordinate system and the robot center as the origin of the robot coordinate system. The ith structure of the objects' surfaces is as follows:

$$f_i(x, y, z) = a_i x + b_i y + c_i z + d_i \tag{1}$$

where a_i, b_i, c_i, d_i are the parameters of the ith plane.

Plane angle relative to the ground θ_i can be expressed as

$$\theta_i = \arccos \frac{c_i}{\sqrt{a_i^2 + b_i^2 + c_i^2}} \tag{2}$$

where $0 \leq \theta_i < \pi$(Fig. 3). We normalize (2) as ϑ_i between $(-\pi/2, \pi/2]$, given by

$$\vartheta_i = \begin{cases} \theta_i, & c_i > 0 \\ 0, & \theta_i = 0 \\ \theta_i - \pi, & c_i < 0 \end{cases} \tag{3}$$

In (1), we adopt the multi-random sample consensus (multi- RANSAC) algorithm [18] to estimate the plane parameters of a_i, b_i, c_i, and d_i, and the inputs can be the 3D point cloud data. However, the large amount of data increases the computational complexity of the algorithm. We use the 3D positions of matching key feature points (e.g., SURF) as the input parameters to estimate the plane parameters. The number of matching feature points is significantly less than that of point cloud data, thereby improving the system's computational efficiency.

D. OCCUPANCY PROBABILITY ESTIMATION

The 3D grid models generally use a space occupancy grid to represent environments, and the occupancy of each independent cell is adopted to describe the position distribution of obstacles. Owing to their simplicity and suitability for various types of sensors, these models are extensively used in the field of robotics. In this section, we only discuss the static environment. Obstacles are defined as the static objects above the ground, such as indoor wall and table, outdoor tree, and pillar.

Our mobile robot is equipped with a stereo vision system, and point cloud data are first captured from the surrounding environment and then pre-processed. The estimation of occupancy probability of cells can be processed with the point cloud. However, due to the large amount of data, we substitute the point cloud with matching key feature points (e.g. SURF) to estimate the occupancy probability of cells and thus lower the computation load.

Consider $\mathbf{p}_t = \{p_{1\,t}, p_{2t}, \ldots, p_{Mt}\}$ as a set of matching feature points at time t. $p_{it} = \{\aleph_{it}, d_{it}\}$ is defined as the concatenation of its position $\aleph_{it} = \{x_{it}, y_{it}, z_{it}\}$ relative to the robot center, and d_{it} is its disparity. Suppose that the robotic environment workspace W is composed of a discretized grid cell, the occupancy state of obstacle grid cell v_t is defined as an independent random variable, and $p(v_t)$ denotes the occupancy probability of the grid cell occupied by an obstacle. Let $z_t = \{z_{1t}, z_{2t}, \ldots, z_{Nt}\}$ represent a set of image observations, and the occupancy probability $p(v_t\,|z_t, \boldsymbol{x}_t)$ of the grid cell can be obtained over the sensor observations z_t and robot poses$\boldsymbol{x}_t$. In particular, occupancy probability is defined as a sampled Gaussian, and the occupancy probability $p(v_t\,|z_t, \boldsymbol{x}_t)$ can be written as [19]

$$p(v_t|\mathbf{z}_t, \mathbf{x}_t) = \eta_1 \exp\left[-\frac{1}{2}\left(\frac{r_i - \bar{r}}{\Delta r}\right)^2\right] \tag{4}$$

where

$$\bar{r} = (\sum_{i=1}^{N} r_i)/N, \quad \Delta r = (\bar{r}/z_i)\Delta z.$$

N is number of features,r denotes the measurement distance of features, η_1 is the normalization factor and Δz is the depth measurement error.

For a stereo camera, the depth measurement error Δz increases quadratically with the measured depth [19]; that is,

$$\Delta z = \frac{z^2}{Bf}\Delta d$$

where z is the depth of feature point, and Δd is the disparity error. In general, we get $\Delta d = 0.5$.

E. OCCUPANCY PROBABILITY UPDATE

The occupancy probability of cells can be calculated by Equation (4). However, some key feature points may be scanned many times when the mobile robot moves forward. Here we use the Kullback-Leibler divergence [20] to update the occupancy probability of cells. The $D_{\mathrm{KL}}(p||q)$ quantifies the difference between two probabilities p and q, where p and q are the occupancy probabilities of the same key feature points in consecutive images (Fig. 4). The robot moves from one position to the next and supposes that p_t^i represents the occupancy probability of the key feature points scanned at time t and that q_{t+k}^i represents the matching key feature points in the next consecutive frames. $D_{\mathrm{KL}}(p||q)$ can be written as follows:

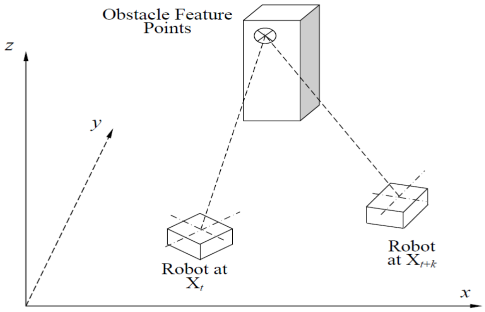


FIGURE 4. Update of the occupancy probability of cells. The same key feature points are captured in the next positions.

$$D_{KL}(p||q) = \sum_{i=1,k=1}^{N,M} p_t^i \log \frac{p_t^i}{q_{t+k}^i} \tag{5}$$

where N is the number of matches of the same key feature points in consecutive images, and M denotes the time at $t+M$.

In (5), p_t^i and q_{t+k}^i are as follows:

$$p_t^i = p(v_t|\mathbf{z}_t, \mathbf{x}_t) \tag{6}$$

$$q_{t+k}^i = \eta_2 p_{t+k}^i p(\mathbf{x}_{t+k}|\mathbf{x}_t) \tag{7}$$

where η_2 is a normalization term, and $\mathbf{x}_t$ and $\mathbf{x}_{t+k}$ can be depicted as follows:

$$\mathbf{x}_{t+k} = R_i\mathbf{x}_t + T_i \tag{8}$$

where R_i and T_i are the rotation and translation matrices between two sets of matching feature points and can be estimated by using the RANSAC [18] algorithm with the points. The estimation process involves two steps that are iteratively repeated. It initially selects the positions of some matching key feature points to calculate R and T by (8) and then checks that all other feature points are consistent with (8). The RANSAC algorithm will iteratively repeat until the obtained model in a specific iteration has sufficient inliers.

Equation (8) describes the motion information of the robot's pose (Fig. 4). We assume the motion errors follow a Gaussian distribution and model $p(\mathbf{x}_{t+k}|\mathbf{x}_t)$ as follows:

$$p(\mathbf{x}_{t+k}|\mathbf{x}_t) = N(\mathbf{x}_{t+k}; m_k, \sum\nolimits_k) \tag{9}$$

where $m_k = Q_k\mathbf{x}_t$ and $\sum_k = Q_k$, Q_k is the covariance of robots' motion error and can be obtained via experiments.

The updated probability can be written as follows:

$$p_{t+k}(v_{t+k}|\mathbf{z}_{t+k}, \mathbf{x}_{t+k}) = \eta_3 \exp\left[-E_{t+k}(v_{t+k})\right] \tag{10}$$

where

$$E_{t+k}(v_{t+k}) = \begin{cases} abs\left[D_{KL}(p||q)\right] + A, & p_t, q_{t+k} \leq 0.5 \\ abs\left[D_{KL}(p||q)\right], & others. \end{cases}$$

The A is a constant and is mainly utilized to ensure that the probability p_{t+k} does not become very large when both p_t and q_{t+k} take smaller values. We obtained $A = 1.07$ in our experiments. The abs[.] denotes absolute value and η_3 is a normalization term. Our motivation is that if the difference (for $p, q > 0.5$) is smaller at different locations, then the occupancy probability will be larger. Conversely, feature points may be mismatched or affected by noise, and their occupancy probability is small. If the feature point is not detected at the next position, its occupancy probability will remain the same as before.

All the occupancy grid cells are a priori initialized as free space and will be updated if cells are in the field of view of the camera. If the feature points are in a free space or an unknown space, then the occupancy probability will be calculated according to (4). Next, if some feature points remain in the field of view of the camera at the next robot locations, then these points are updated according to (10).

F. GRID APPROXIMATION AND PROCESSING

Standard grid mapping uses cubic volume to represent the objects in an environment. Grid combination helps significantly reduce the number of grid cells and improve the computational efficiency in some robots' applications, such as localization, navigation, and exploration.

Our method of grid approximation is based on point clouds obtained by the stereo vision system equipped on the robot. As the scene scanned by the vision system contains more information, we mainly aim at modeling the objects above-ground in the environment, and the visual point cloud data should be preprocessed to remove the ground information ahead of grid approximation. Our method makes two important assumptions: one is that the structures of objects in an environment contain planar surfaces, and the other is that the ground data of point cloud are removed. The first assumption is easy to satisfy, especially in structured indoor environments. While outdoor environments contain non-structured objects, many objects, however, such as buildings and stone steps, consist of planar surfaces. Removed ground data in the second assumption help further reduce the number of cells and improve the computational efficiency.

The standard occupancy grid representation is composed of occupied, free, and unmapped cells, as shown in Fig. 5(a). All grid cells inside the field of view of the cameras are considered as free or occupied, and the outside are unmapped areas. Free space assumption [9] significantly improves the computational and memory efficiency of mapping, specifically for large-scale mapping. In this assumption, all space is considered as free, except for obstacles, as shown in Fig. 5(b). However, it does not differentiate free and unknown space, and this may be a problem for some robotic applications, such as path planning and navigation.

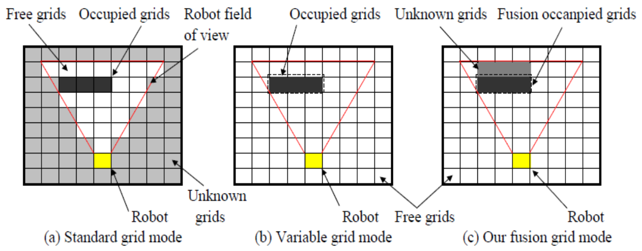
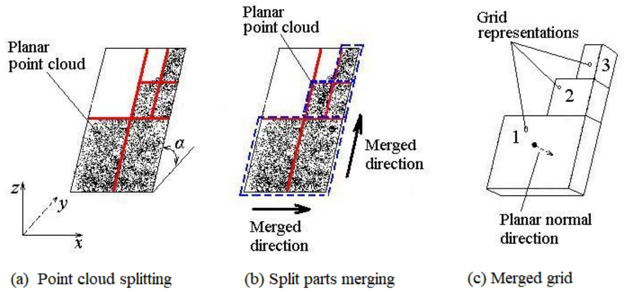


FIGURE 5. Mode of occupancy grid representation. (a) Standard occupancy grid representations are composed of occupied, free, and unmapped grid. (b) Variable grid representations are composed of occupied and free grid. (c) Our combination grid mode is composed of occupied, free, and unmapped grid, and the unknown grid is usually produced by the visual occlusion of obstacles.



(a) Point cloud splitting (b) Split parts merging (c) Merged grid

FIGURE 6. Example of clustering planar point cloud data splitting, merging, and grid forming. (a) The point cloud data are clustered and first split into four parts; if the splitting density threshold is not reached, the splitting will take place, as shown in the red lines. (b) The split parts are first merged along the horizontal direction of the plane and then the vertical direction. The merged parts are shown in the blue box. (c) The merged parts extend a certain size along the planar normal direction and form three grid approximations.

The current study proposes an occupancy grid approach with a local explicit modeling space, as shown in Fig. 5(c). All space a priori is initialized as free, but the grid cells inside the field of view of the cameras are marked as free, occupied, or unknown. The unknown cells are usually produced by the visual occlusion of obstacles. These unknown spaces inside the field of view of the camera are added to the map when robot mapping is carried out. If a robot trajectory is given for such a map during mapping, then the map with some unknown space will bring more benefits to some applications of the robot compared with the free space assumption.

Fig. 6 illustrates the grid approximation and combination processing. The proposed approach initially uses multi-RANSAC algorithm [18] to estimate the plane parameters and then clusters the point cloud data located on the same plane. With the noise measured by the sensors considered, the following equation is used to extract these planar point cloud data:

$$\frac{|a_i x_i + b_i y_i + c_i z_i + d_i|}{\sqrt{a_i^2 + b_i^2 + c_i^2}} \leq |d_T| \tag{11}$$

where d_T is a distance threshold and can be obtained by experiments. $d_T = 0.05$ in our experiments. (a_i, b_i, c_i, d_i) are the planar parameters as (1), and (x_i, y_i, z_i) are the positions of point cloud data.

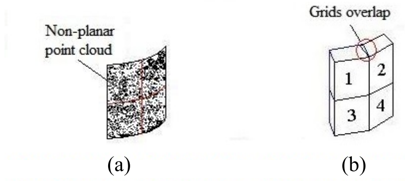


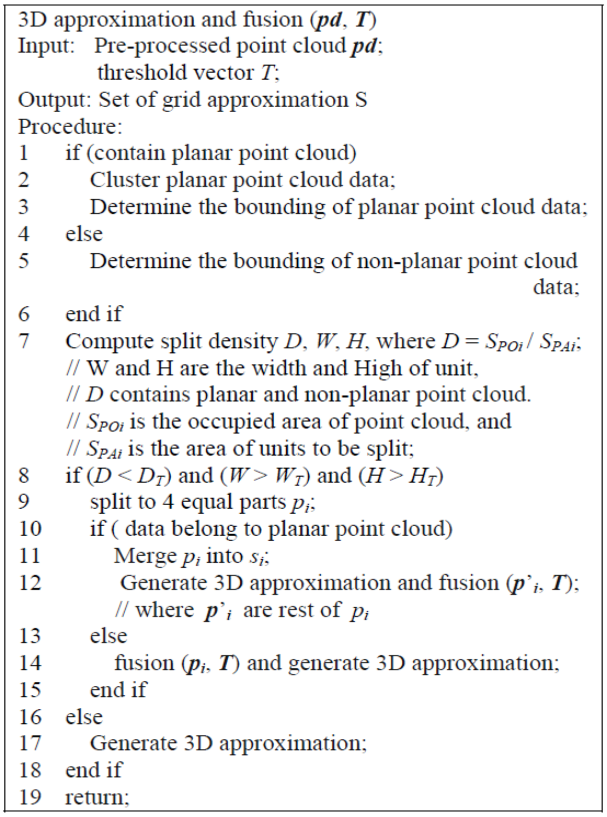
FIGURE 7. Example of non-planar point cloud splitting and grid forming. (a) Non-planar point clouds are split based on their center. (b) The split parts form four grid cells, which may have small partially overlapping. However, the overlaps are not located on the surfaces of objects but inside.

The clustering point cloud data located on the same plane are split and merged to approximation representations as shown in Fig. 6(a). These data split the plane point cloud data into four equal parts based on their center along the plane direction, which can be obtained by (3) until the areas of parts satisfy a certain threshold. The split parts are preferentially merged into a larger rectangle toward the horizontal direction of the plane and then continue to merge in the vertical direction, as shown in Fig. 6(b). The merged parts extend a certain size along the normal (back) of the plane and form rectangular cuboid approximations, as shown in Fig. 6(c). In the example of Fig. 6, three planar rectangular cuboid approximations are formed.

The remaining point cloud data not in the same plane are repeatedly processed as planar point cloud data to be split into four equal parts based on their center until they reach the given threshold. The split parts finally extend a certain size along the normal (back) of the plane, which is formed from the point cloud boundary, and produce rectangular cuboid approximations, as shown in Fig. 7(a). Unlike the split parts of the plane point cloud, these parts do not merge into larger rectangular cuboid approximations. In some cases, these grid approximations may have small partial overlapping, such as with objects that have large curved surfaces. However, this overlaps have little effect on the application of robots because they occur inside the objects, as shown in Fig. 7(b).

The pseudocode of the grid approximation and mergence algorithm is shown in Fig. 8. The input parameters contain pre-processed point cloud data, which removed the ground points, and some threshold vectors, $T = (D_T, W_T, H_T)$. D_T, W_T, and H_T are the thresholds of density, width, and height of split units, respectively. The output parameters contain a set of grid approximations $S(s_1, s_2, \ldots, s_n)$.

This algorithm first checks whether the input data contain planar point cloud data. Given these data, it adopts Equation (11) to cluster planar point cloud data and determine the bounding of planar and non-planar data. The algorithm then computes unit split density and determines whether or not it will split down further. If the unit split density is less than the specified threshold, then the unit will be further split into four parts. After the split is complete, the units where the planar point cloud is located process the mergence. It first merges the adjacent cells in the horizontal direction of the plane and then processes these cells in a vertical direction. The horizontal mergence only operates on adjacent cells with equal widths and equal length only for the vertical direction. For the units where the non-planar point cloud is located, the split continues until the size of the split cell reaches the specified threshold; however, these units do not merge. Finally, merged or split units extend a certain size to form the 3D approximation along the normal (back) of unit planes.



```
3D approximation and fusion ($\boldsymbol{pd}$, $\boldsymbol{T}$)
Input:   Pre-processed point cloud $\boldsymbol{pd}$;
         threshold vector $T$;
Output: Set of grid approximation S
Procedure:
    if (contain planar point cloud)
        Cluster planar point cloud data;
        Determine the bounding of planar point cloud data;
    else
        Determine the bounding of non-planar point cloud
                                                      data;
    end if
    Compute split density $D$, $W$, $H$, where $D = S_{POi} / S_{PAi}$;
    // W and H are the width and High of unit,
    // $D$ contains planar and non-planar point cloud.
    // $S_{POi}$ is the occupied area of point cloud, and
    // $S_{PAi}$ is the area of units to be split;
    if ($D < D_T$) and ($W > W_T$) and ($H > H_T$)
        split to 4 equal parts $p_i$;
        if ( data belong to planar point cloud)
            Merge $p_i$ into $s_i$;
             Generate 3D approximation and fusion ($\boldsymbol{p'}_i$, $\boldsymbol{T}$);
            // where  $\boldsymbol{p'}_i$  are rest of  $p_i$
        else
            fusion ($\boldsymbol{p}_i$, $\boldsymbol{T}$) and generate 3D approximation;
        end if
    else
        Generate 3D approximation;
    end if
    return;
```

FIGURE 8. Grid approximation and combination pseudocode.

The occupancy probability of grid is substituted with that of matching key feature points, which are located in this grid. Here, we take their average as the probability estimate of this grid. If the grid does not contain matching key feature points, then the average occupancy probability of adjacent grid cells is used as its probability estimate.

IV. EXPERIMENT RESULTS AND ANALYSES

In this section, we perform a series of experiments to evaluate the proposed method. The experiments are conducted in indoor and outdoor environments and are further divided into two main subsections: map qualitative analysis and performance evaluation. The robot platform is shown in Fig. 1. The installation of cameras is $h_c = 370$ mm, which is used to capture the visual data of the environment.

In our experiments, SURF [21] is used to estimate the plane parameters and occupancy probability as it is a robust feature extractor that works in environments with dense or sparse features. Given the computational efficiency, these feature extraction and matching processes are implemented on the DM642 card of the robot platform.

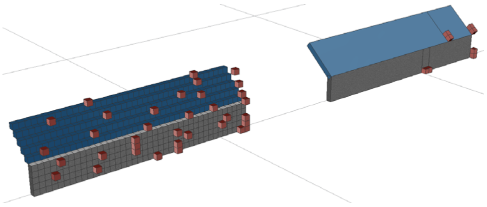


FIGURE 9. **Example of erroneous artifacts produced by our method and standard grid approximation. Left view shows the erroneous artifacts produced by the standard grid approximation (red grid cells). Right view shows that most of the erroneous artifacts are removed in our method (red grid cells). The occupied grid cells are represented with color for easy viewing.**

In building a map, we use (8) to locate the robot because accurate mapping depends on localization. Key points in the continuous frame are initially used to estimate the rotation matrix and the translation matrix, and then the position of the robot in the map is calculated by using the estimated parameters. When the robot moves to the next positions, it uses the matching key feature points to create registrations in the frames and then updates and merges the data. The new data structure can be stored in files to enable the robot to perform any tasks. We use the labels 00, 01, 10, and 11 to represent free, occupied, unknown, and inner node with child, respectively. These labels are encoded as a bit stream and saved as files.

In our experiments the mobile robot's speed is limited to $v \leq 0.5m/s$.

A. MAP QUALITATIVE ANALYSIS

In our map building experiments, the SURF visual features are used to cluster the planar point cloud data. The clustering of a large number of planar point cloud data is beneficial to discard some isolated error point cloud data and improve the mapping accuracy. As shown in Fig. 9, planes A and B are mapped using our method and the standard grid approximation. The standard grid method produces some erroneous artifacts (red grid cells) for some error point cloud data, as shown in the left view of Fig. 9. However, most of the erroneous artifacts are removed in our method, and the mapping quality is higher than the standard grid approximation, as shown in the right view of Fig. 9.

Fig. 10 is a part of an indoor corridor 3D grid map constructed by using our method, where the dashed line and arrow describe the path and moving direction of the mobile robot, respectively. The size of the minimum grid is 0.1 m × 0.1 m × 0.1 m, and the maximum size is 2.2 m × 0.7 m × 0.1 m. Owing to the occlusion of objects, when the robot is mapping along the corridor, some areas with gray will not update; these areas are marked as unknown, as shown in Fig. 10(b), which is the 2D projection view of Fig. 10(a). From this view, these unknown areas (with gray color) can be distinguished clearly.

Fig. 11 is an example of an outdoor 3D grid map created with our approach; the size of the scene is about 30 m × 10 m. Some structured objects, such as buildings, walls, and steps, are contained in the scene, as well as some non-structured objects, such as trees, stone columns, and iron pillars, and are marked in the map. Taking into account the complexity of the outdoor environment, the tree leaves may be a variable because of the effect of the wind; we limited the height of trees, only leaving the trunks with no leaves in the created map. The dashed line and arrow are the path and moving direction of the mobile robot. The unknown areas are not shown in the figure for ease of visualization.

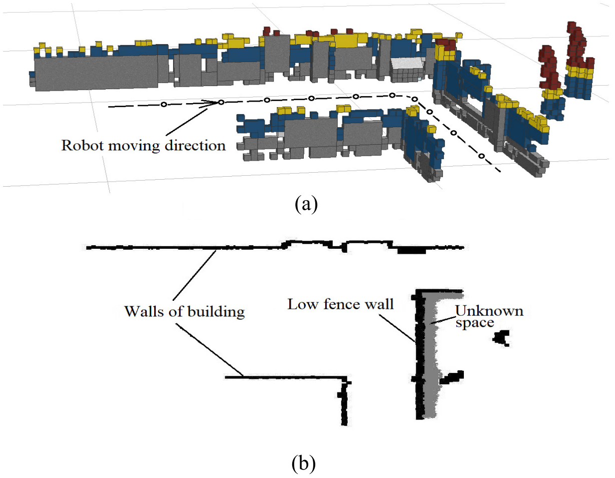


FIGURE 10. **Indoor corridor 3D grid map with the size of minimum grid cell and maximum grid cell are 0.1 m × 0.1 m × 0.1 m and 2.2 m × 0.7 m × 0.1 m, respectively. 10 places (circles) along the robot motion path direction are selected for performances evaluation. The occupied grid cells are represented with color for easy viewing. (a) Corridor 3D map. (b) The 2D projection view of the 3D map and the fields of gray represent the unknown.**

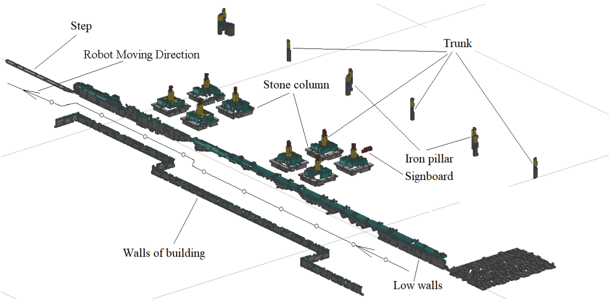


FIGURE 11. **Outdoor 3D map with the minimum and maximum grid cell sizes of 0.1 m × 0.1 m × 0.1 m and 0.7 m × 0.3 m × 1 m, respectively. 10 places (circles) along the robot motion path direction are selected for performances evaluation. The occupied grid cells are represented with color for easy viewing.**

B. PERFORMANCE EVALUATION

In this section, we evaluated our method in terms of grid quantity, erroneous artifacts, and memory efficiency in indoor and outdoor environments. The scenes of indoor and outdoor environments are as shown in Figs. 10 and 11, respectively. Given the inconvenience of huge number of grid cells, 10 places along the robot motion path direction (circles in Figs. 10 and 11) and 10 frames at each selected location were carefully selected to evaluate these performances. In our

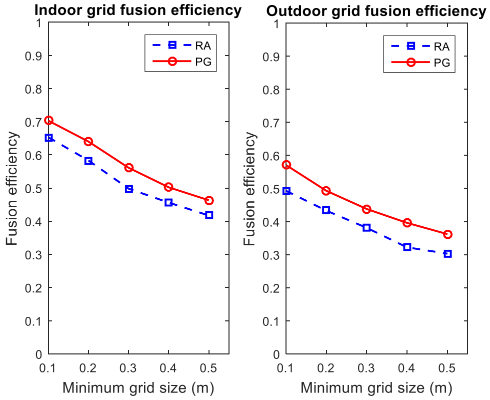


FIGURE 12. Comparison of grid combination efficiency in indoor and outdoor environments for RA and PA; these results take the standard grid as reference. The minimum grid cell size represents the minimum voxel size.

comparative experiments, our method (PG) was compared with the standard grid (SG), octree grid (vision-based) (OG) and rectangular cuboid approximation (RA) methods.

We first conducted comparison experiments of combination efficiency and computational cost. The experiments of combination efficiency were evaluated in indoor and outdoor environments, and the standard grid was used as reference in the evaluation results. Combination efficiency η is defined based on

$$\eta = 1 - (N_f / A_S)$$

where $A_S$and N_f are the number of combination grid cells and the total number of standard grid cells in some position, respectively. The results are shown in Fig. 12. The minimum grid cell size on the horizontal axis indicates the size of the voxel. The figures show the combination efficiency with different minimum grid cell sizes between 0.1 and 0.5 m. The results reveal that PG is more efficient than RA in combination efficiency mainly because RA only combines axis-aligned grid cells, whereas PGs can also combine planar objects that do not follow the rule of axis alignment. The structured indoor environment has higher combination efficiency than the outdoor environment, thereby significantly reducing the number of grid cells.

Evaluating all the mapping errors is a highly difficult task. 10 specific locations (as shown in Figs. 10 and 11) are selected to compute their mapping error and take the number of standard grid cells as reference. Here, the mapping error e is defined based on

$$e = (N_e / A_S) \times 100\%$$

where N_e and A_S are the number of erroneous artifacts and the total number of standard grid cells in some position, respectively. When the robot moves to a specified position, we take the occupied grid in a small range near the robot to calculate the erroneous artifacts against the ground truth. The experiment results are shown in Fig. 13. These figures clearly show that mapping erroneous artifacts produced by our method (PG) are significantly less than in the SG and rectangular cuboid approximation method (RA), and the mapping errors outdoor are higher than indoors.

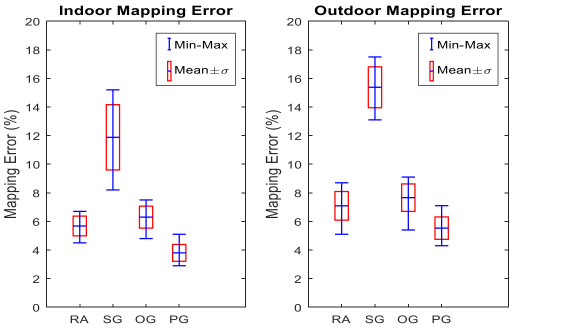


FIGURE 13. Evaluation of mapping error in indoor (left) and outdoor (right) environments. The plots were generated from 10 places marked on the trajectory of a robot with little circles in Fig. 10 and Fig. 11.

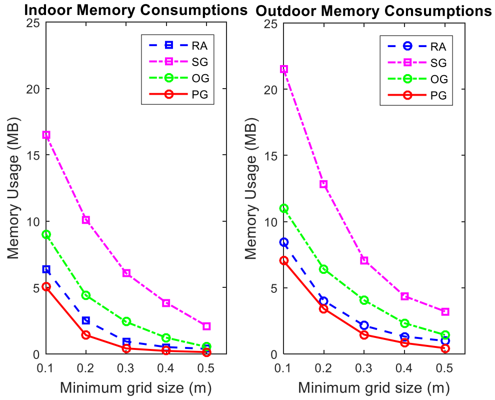


FIGURE 14. Evaluation of memory consumptions (left) indoor and (right) outdoor. The plots were generated from 10 places marked on the trajectory of the robot with little circles in Figs. 10 and 11. The minimum grid cell size represents the minimum voxel size.

In terms of memory consumption, we also compared our method (PG) to the standard grid (SG)čoctree grid (OG) and rectangular cuboid approximation (RA) methods in indoor and outdoor environments. In our experiments, the memory consumption is recorded when the robot moves to several specified positions. On average over all test runs, we evaluated the memory performance with different minimum grid sizes between 0.1 and 0.5 m (Fig. 14). These figures show that our method consumes less memory compared to the other two methods, especially in an indoor structured environment. Owing to the complexity of the outdoor environment, unstructured objects usually cause low grid combination efficiency and increase the memory. However, in our method (PG), the combination efficiency was high and only the surfaces of objects were mapped; the memory consumptions are still lower than those used by the standard grid (SG), octree grid (OG) and rectangular cuboid approximation (RA) methods.

V. CONCLUSION

This paper presents an efficient method for 3D grid modeling based on planar surface. Conventional voxel-based occupancy grid models largely reduce system mapping efficiency for huge quantities of grid cells. Although some methods, such as rectangular cuboid approximation models, improve the mapping efficiency using grid combination algorithm, the non-axis-aligned surface of objects still have to be mapped by using voxel occupancy grid and cannot be merged. This study uses planar grid, substitutes voxel grid, causes easier combination for the non-axis aligned grid cells, and further improves the mapping efficiency. In our method, matching key feature points are initially used to estimate the plane parameters, and then pre-processed point cloud data located on the same plane are clustered. In the next, a split and combining algorithm is used to generate 3D planar grid approximation representations of the environment. The occupancy probabilities of grid is obtained from those of matching key feature points to reduce the computation loads. Finally, a series of experiments that include map qualitative analysis and performance tests is adopted to evaluate the presented method in indoor and outdoor environments.

REFERENCES

[1] Y. Zhuang, N. Jiang, H. S. Hu, and F. Yan, "3-D-laser-based scene measurement and place recognition for mobile robots in dynamic indoor environments," *IEEE Trans. Instrum. Meas.*, vol. 62, no. 2, pp. 438–450, Feb. 2013.

[2] T. Whelan, H. Johannsson, M. Kaess, J. J. Leonard, and J. McDonald, "Robust real-time visual odometry for dense RGB-D mapping," in *Proc. IEEE Int. Conf. Robot. Automat.*, Karlsruhe, Germany, May 2013, pp. 5724–5731.

[3] E. Benli, J. Rahlf, and Y. Motai, "Dynamic 3-D reconstruction of human targets via an omni-directional thermal sensor," *IEEE Sensors J.*, vol. 18, no. 17, pp. 7209–7221, Sep. 2018.

[4] B. Tippetts, D. J. Lee, K. Lillywhite, and J. Archibald, "Review of stereo vision algorithms and their suitability for resource-limited systems," *J. Real-Time Image Process.*, vol. 11, no. 1, pp. 5–25, 2016.

[5] D. Hähnell, W. Burgard, and S. Thrun, "Learning compact 3D models of indoor and outdoor environments with a mobile robot," *Robot. Auton. Syst.*, vol. 44, no. 1, pp. 15–27, Jul. 2003.

[6] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments," *Int. J. Robot. Res.*, vol. 31, no. 5, pp. 647–663, 2012.

[7] L. Li, M. Yang, C. Wang, and B. Wang, "Hybrid filtering framework based robust localization for industrial vehicles," *IEEE Trans. Ind. Inform.*, vol. 14, no. 3, pp. 941–950, Mar. 2018.

[8] D. Ni, A. Song, X. Xu, H. Li, C. Zhu, and H. Zeng, "3D-point-cloud registration and real-world dynamic modelling-based virtual environment building method for teleoperation," *Robotica*, vol. 35, no. 10, pp. 1958–1974, Oct. 2017.

[9] S. Khan, A. Dometios, C. Verginis, C. Tzafestas, D. Wollherr, and M. Buss, "RMAP: A rectangular cuboid approximation framework for 3D environment mapping," *Auton. Robots*, vol. 37, no. 3, pp. 261–277, Feb. 2014.

[10] F. Endres, J. Hess, J. Sturm, D. Cremers, and W. Burgard, "3-D mapping with an RGB-D camera," *IEEE Trans. Robot.*, vol. 30, no. 1, pp. 177–187, Feb. 2014.

[11] O. Miksik, V. Vineet, M. Lidegaard, R. Prasaath, M. Nießner, S. Golodetz, S. L. Hicks, P. Pérez, S. Izadi, and P. H. Torr, "The semantic paintbrush: Interactive 3D mapping and recognition in large outdoor spaces," in *Proc. 33rd Annu. ACM Conf. Hum. Factors Comput. Syst.*, Apr. 2015, pp. 3317–3326.

[12] S. T. O'Callaghan and F. T. Ramos,"Gaussian process occupancy maps," *Int. J. Robot. Res.*, vol. 31, no. 1, pp. 42–62, Jan. 2012.

[13] S. Kim and J. Kim, "Occupancy mapping and surface reconstruction using local Gaussian processes with Kinect sensors," *IEEE Trans. Cybern.*, vol. 43, no. 5, pp. 1335–1346, Oct. 2013.

[14] M. Zeng, F. Zhao, J. Zheng, and X. Liu, "Octree-based fusion for realtime 3D reconstruction," *Graph. Models*, vol. 75, no. 3, pp. 126–136, 2013.

[15] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Auton. Robot.*, vol. 34, no. 3, pp. 189–206, 2013.

[16] K. Schauwecker and A. Zell, "Robust and efficient volumetric occupancy mapping with an application to stereo vision," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2014, pp. 6102–6107.

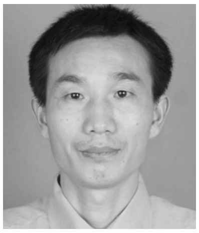
[17] S. Khan, D. Wollherr, and M. Buss, "Adaptive rectangular cuboids for 3D mapping," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2015, pp. 2132–2139.

[18] M. Zuliani, C. S. Kenney, and B. S. Manjunath, "The multiRANSAC algorithm and its application to detect planar homographies," in *Proc. IEEE Int. Conf. Image Process.*, Oct. 2005, p. 153.

[19] B. H. Guo, H. Y. Dai, and Z. H. Li, "A visual-attention-based 3D mapping method for mobile robots," *Acta Automatica Sin.*, vol. 43, no. 7, pp. 1248–1256, Jul. 2017.

[20] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, 1951.

[21] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Proc. Eur. Conf. Comput. Vis.*, May 2006, pp. 404–417.



BINGHUA GUO received the Ph.D. degree in control theory and control engineering from the South China University of Technology, Guangzhou, in 2003. He is currently an Associate Professor of control science with the University of Zhaoqing, Zhaoqing, Guangdong. His areas of interests include artificial intelligence and 3-D modeling.

HONGYUE DAI received the Ph.D. degree in circuits and systems from the South China University of Technology, Guangzhou, Guangdong, in 2007. He is currently a Faculty Member of the University of Zhaoqing, Zhaoqing, Guangdong. His research interests include robotics and image processing.

ZHONGHUA LI received the Ph.D. degree in control science and engineering from the South China University of Technology, in 2005. He is currently a Faculty Member of the School of Information Science and Technology, Sun Yat-sen University, Guangzhou, Guangdong, China. His research interests include artificial intelligence and robotics, RFID technologies, and the Internet of Things (IoT).

WEI HUANG received the B.S. degree in automation from Henan Polytechnic University, Jiaozuo, Henan, in 1990. He is currently a Faculty Member of the University of Zhaoqing, Zhaoqing, Guangdong. His research interests include robotics and artificial intelligence.