Adaptive Rectangular Cuboids for 3D Mapping

Sheraz Khan[1], Dirk Wollherr[1] and Martin Buss[1,2]
[1]Chair of Automatic Control Engineering
Technische Universität München (TUM), 80333 München, Germany
[2]TUM Institute of Advanced Study
Technische Universität München (TUM), 85748 Garching, Germany
{sheraz.khan,dw,mb}@tum.de

***Abstract*— This paper presents an extension of the standard occupancy grid for 3D environment mapping. The presented approach adds a fusion process after the occupancy update which modifies the resolution of the grid cells in an incremental manner. Consequently, the proposed approach requires fewer grid cells for 3D representation in comparison to a standard occupancy grid. The resolution adaptation process is based on the occupancy probabilities of the grid cells and leads to the relaxation of the cubic grid cell assumption common to most 3D occupancy grids. The aim of this paper is to show the advantage of the proposed incremental fusion process which leads to the approximation of the 3D environment using rectangular cuboids. Evaluation on a large scale dataset and comparison to the state of the art shows that the proposed approach has faster access time for all occupied grid cells and requires a smaller number of cells for 3D environment representation.**

I. Introduction

An important component required by an autonomous agent to perform navigation and obstacle avoidance is an accurate 3D representation of the environment. A large amount of research has been carried out in the field of 2D mapping and localization in the last few decades. The environment representation generated by most 2D mapping approaches can be categorized into topological [1] or metric [2]–[4] maps. Topological maps use graph structures to represent the environment whereas metric maps capture its area or volume. The most commonly used representation for metric mapping is an occupancy grid [2]–[6] as it provides a probabilistic method to deal with noisy sensor observations and multisensor data fusion. Occupancy grids have been used extensively for navigation, localization and exploration in the field of robotics.

Recently the focus in the robotics research community has shifted from 2D to *large scale* 3D mapping. A majority of the occupancy grids in literature consist of fixed resolution grid cells. The main question raised by this paper is whether multiresolution representations based on rectangular cuboid grid cells can be useful for 3D environment representation. Figure 1 shows a fixed resolution representation in comparison to a multiresolution representation in a simplified 2D example. The multiresolution representation is created by allowing the square cells to fuse to form axis aligned rectangles. Intuitively speaking, this leads to a reduction in the number of grid cells required without any loss of information in the environment representation. Additionally, it allows faster access times as only 4 grid cells need to be accessed for a multiresolution representation to reconstruct the environment in contrast to 25 for a fixed resolution representation, as shown in the example in Figure 1. In this paper the term cells is used interchangeably for squares (2D), rectangles (2D), cubes (3D) and rectangular cuboids (3D) based on context. If the structure (occupied regions) of the actual 3D world is composed of planar axis aligned surfaces whereas free space does not have any definite shape, the question arises if there is any advantage in relaxing the assumption of 3D representation based on cubes (inherent to most occupancy grids) to allow axis aligned rectangular cuboids.

This paper presents an extension of the standard occupancy grid that adapts the initial resolution of its cells based on occupancy probabilities. The proposed approach is based on the Rtree data structure [7] which is composed of a hierarchy of axis aligned rectangular cuboids. Evaluation of the approach presented in this paper on a large scale dataset and a comparison to the state of the art shows that it requires a fewer number of grid cells for 3D environment representation.

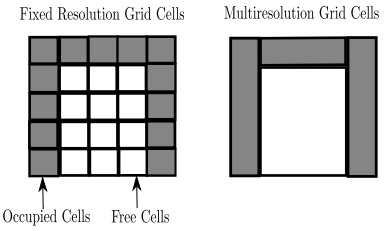


Fig. 1. Comparison of a fixed and a multiresolution representation. The multiresolution representation requires fewer number of grid cells in contrast to a fixed resolution representation.

II. Related Work & Contribution

A. Related work

2D occupancy grids [2], [3], [5] are considered as the de facto standard for mobile robotic mapping. The main drawback of 2D occupancy grids is that they are only applicable in case of planar environments. Besides 2D environment representations, some grid structures additionally store the height corresponding to each cell generating 2.5D height

maps [8]. However, a disadvantage of height maps is its inability to model the environment shape.

The advent of the Velodyne and the Kinect has shifted the focus of the robotics research community from 2D towards 3D environment representation. Occupancy grids and raw point clouds are the most commonly used approaches for 3D environment representation. Point cloud representation is useful, however, its disadvantage is that it requires a large amount of memory as all points are stored. Additionally, it does not allow data fusion in a probabilistic manner. Multi level surface maps [9] can also be used for 3D environment representation, however, it cannot represent arbitrary shapes. In [10], the authors present an approach titled Multi Volume Occupancy Grids (MVOG) for 3D mapping which groups observations in vertical volumes over a 2D occupancy grid. The vertical volumes represent positive (obstacle) and negative (free) readings. A 3D probabilistic occupancy grid is formed by merging these volumes. The work presented in [11] develops an N^d-tree based formulation which adapts the resolution of the grid in an online manner based on measurements. A fully probabilistic occupancy grid structure based on octrees titled Octomap [4], [6] has also been presented which allows multiresolution 3D environment representation. Recently, an extension of the 3D NDT (Normal Distribution Transform) [12] concept in the context of occupancy mapping titled NDT-OM (Occupancy Mapping) has been presented [13], [14] which models the point distribution in each cell using a Gaussian distribution.

All occupancy grid approaches mentioned above rely on cubic grid cells for 3D environment representation. The work presented in [4], [6] uses an Octree to represent the environment which is composed of a hierarchy of cubes. The approach in [11] utilizes a generalization of quadtree and octree that allows a subdivision of any d dimensional volume with N^d children, however, in principle still relies on cubes or hyper cubes for representation. This paper in contrast, augments the occupancy grid with an incremental fusion process to form multiresolution 3D environment representations based on axis aligned rectangular cuboids.

B. Contribution

The contribution of this paper is an incremental fusion process of grid cells using occupancy probabilities. Consequently, this fusion process leads to multiresolution 3D environment representations based on axis aligned rectangular cuboids instead of the standard cubic grid cell assumption common to all occupancy grids. The approach presented in this paper is an extension of the authors' previous work [15]. The differentiating factor is the addition of the fusion process which incrementally adapts the grid cells resolution (which remained fixed in the occupancy grid approach proposed in the authors prior work) and the hierarchy of axis aligned rectangular cuboids. In addition, the sensor model considered in this paper explicitly models free space. As a result of the fusion process the number of grid cells required by the occupancy grid are effectively reduced without any loss of information in the environment representation.

The rest of the paper is organized as follows: a brief description of the Rtree data structure is presented in Section III. The occupancy grid based on the Rtree data structure and the adaptation of cell resolution based on occupancy probabilities is discussed in Section IV. Section V presents results by conducting experimental evaluations. Conclusions are presented in Section VI.

III. Rtree Data Structure

This section provides only a brief overview of the Rtree datastructure, therefore, the reader is referred to [7], [15], [16] for a comprehensive description. The Rtree data structure [7] proposed by Antonin Guttman is composed of a hierarchy of minimum bounding axis aligned rectangles (MBR), or minimum bounding axis aligned rectangular cuboids (MBRC) for 3D. As shown in Figure 2, the Rtree nodes are labeled as R, L to denote root and leaf nodes, respectively. Inner nodes are not shown in Figure 2, however, as the hierarchy expands inner nodes are added as well. It is important to remark here that the tree structure depiction in Figure 2 is different than normal convention to facilitate the discussion of the Rtree based occupancy grid (Section IV) and the results in the experimental section (Section V). All branches connected to the leaf, inner and root node are termed leaf, inner and root branches, respectively. The root and inner branches contain information regarding the MBR, whereas the leaf branches contain the rectangle. The Rtree of order (d,M) has the following characteristics [7], [16]:

- A leaf node can have a maximum of M branches and a minimum of d where $d \leq \frac{M}{2}$. The leaf branches contain the tuple $(\text{rectangle}, \text{object})$. The object represents the occupancy probability of the rectangle in the context of this paper. As the Rtree is height balanced all leaf nodes are at the same height.
- An inner node contains a maximum of M branches and a minimum of d entries. Each inner branch consists of a MBR and a pointer to its child node.
- The root node can have a minimum of two branches unless it is a leaf node.

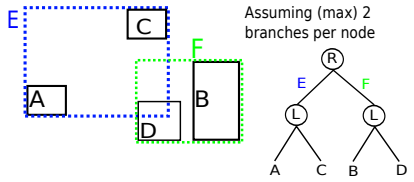


Fig. 2. An example of the Rtree data structure which is composed of a hierarchy of axis aligned rectangles (2D).

A rectangle is inserted into the Rtree structure through the process of *least expansion* which involves searching for inner branches in the hierarchy that lead to the minimum expansion of the MBR. If the number of branches in a node exceeds M during the insertion process, the node has to be split. An exemplary Rtree construction is shown in Figure 2 assuming that each node can have a maximum of 2 branches ($M = 2$). The Rtree structure initially consists of a single

node when rectangles A and B are inserted. Furthermore, if rectangles C and D are added the node splits increasing the height of the structure and forms overlapping MBR (E and F), as shown in Figure 2. The splitting process shown in the figure is just an illustration. The splitting strategy used in this paper is termed as quadratic splitting [7] and this strategy has been chosen due to its better quality of split in comparison to linear splitting. The important aspect is that the MBR of the inner branches in the Rtree structure can *overlap*. As a consequence of overlaps within the tree hierarchy multiple nodes might need to be searched during a spatial query. The number of branches allowed per node M is another important factor in the hierarchy construction process. For a fixed number of leaf branches increasing M generates tree structures containing fewer inner nodes but creates more overlaps. Consider the scenario shown in Figure 2 in which the assumption of 2 branches per node is considered. If the maximum number of branches per node is increased from 2 to 4, one node is required in the hierarchy to represent all leaf branches thereby reducing the number of nodes required for representation. The search for rectangles contained within a given arbitrary axis aligned query rectangle is carried out by performing overlap/containment tests throughout the Rtree hierarchy. The focus of this paper is on 3D mapping, hence the term rectangular cuboids will be used for leaf branches and MBRC for inner and root branches.

IV. Rtree based Adaptive Occupancy grid

This section is divided into two subsections. The first subsection deals with the description of the occupancy grid based on the Rtree data structure and the second describes the adaptation of the grid cell resolution.

A. Occupancy grid based on the Rtree data structure

The Rtree occupancy grid is probabilistic in nature and models the occupancy of its grid cells like a standard occupancy grid. However, in contrast to a standard occupancy grid in which the structure is predefined, the Rtree based occupancy grid *incrementally* generates the grid structure and the tree hierarchy composed of axis aligned rectangular cuboids as outlined in the previous section. Given a specific set of sensor observations, grid cells are initialized at the beam end points and along the beam path with a probability of 0.5 and updated. A grid cell once initialized is always updated. Regions in which observations are not received remain uninitialized. Figure 3 shows an example of the grid cell initialization for the Rtree based occupancy grid. Ray tracing is performed along the beam path to update the occupancy values of all initialized grid cells. Initially all grid cells i.e. leaf branches of the Rtree occupancy grid are of cubic volume based on the chosen resolution of the grid, axis aligned and do not overlap. However, the inner branches MBRC can overlap as discussed in Section III.

Let $\hat{z}_t$ represent the sensor observation where the subscript denotes the time index. Consider a grid $G_t = \{g_1, g_2, \ldots, g_n\}$ at time t consisting of n grid cells g_i, $i = 1, \ldots, n$ of cubic or multiple resolutions. Initially, the

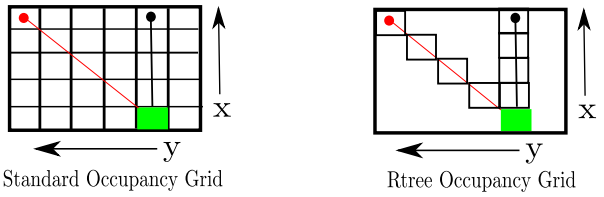


Fig. 3. The cell initialization process in the Rtree occupancy grid based on sensor observations. The standard occupancy grid has a predefined structure, i.e cubic grid cells are initialized with 0.5 occupancy probability in a fixed predefined region which is expanded dynamically. In contrast, the Rtree based occupancy grid incrementally generates the grid structure as sensor observations are obtained. The robot is shown as a solid green block.

occupancy grid is composed of cubic grid cells, however, as sensor observations are obtained the adaptation of grid cell resolution takes place (Section IV B) based on occupancy probabilities leading to multiresolution cells. The notation used in this section is valid for cubic and multiresolution representations. In the context of the Rtree occupancy grid, the grid cells g_i represent the leaf branches of the Rtree structure. The occupancy probability of any leaf branch g_i representing the i^{th} grid cell can be derived from the posterior distribution over the cells given all the sensor observations $\hat{z}_{1:t}$ and robot poses $x_{1:t}$

$$P(g_1, g_2, \ldots, g_n | \hat{z}_{1:t}, x_{1:t}).$$

A common assumption in the standard occupancy grid to reduce the dimensionality and computational complexity of the problem is

$$P(g_1, g_2, \ldots, g_n | \hat{z}_{1:t}, x_{1:t}) = \prod_{i=1}^{n} P(g_i | \hat{z}_{1:t}, x_{1:t}),$$

which states that the occupancy probability of a cell is calculated independently of other grid cells. Furthermore, by transforming the observations based on the pose estimates into the global frame of reference we can omit the pose information

$$\prod_{i=1}^{n} P(g_i | \hat{z}_{1:t}, x_{1:t}) = \prod_{i=1}^{n} P(g_i | z_{1:t}),$$

where $z_{1:t}$ represents the transformed observations in the global frame of reference. By using the Bayes rule and incorporating the Markov assumption in the first term of the numerator, i.e. the current observation z_t is conditionally independent of previous observations $z_{1:t-1}$ given the robot pose, each grid cell probability can be written as

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

Similarly, the first term of the numerator in (1) can be written as

$$P(z_t | g_i) = \frac{P(g_i | z_t) P(z_t)}{P(g_i)}. \tag{2}$$

By substituting (2) into (1),

$$P(g_i | z_{1:t}) = \frac{P(g_i | z_t) P(z_t) P(g_i | z_{1:t-1})}{P(g_i) P(z_t | z_{1:t-1})}, \tag{3}$$

(a) Sampling grid cells for fusion

(b) Example of cell expansion

Fig. 4. (Best viewed in color) (a) The process of sampling cells along the beam path to allow fusion in the occupancy grid. The randomly sampled cells are shown with thin dashes (in grey) and the corresponding cells of the grid are shown with a pattern of thick dashes (in red). (b) The cell expansion process for two cases i.e. cube and a rectangular cuboid, shown in (a). The search direction is defined by the cell width vector $\mathbf{w}_i$. The first preference is shown with dashes (in blue) followed by the second preference with a dash-dot pattern (in orange). If all sides of the cell are the same i.e. the cubic cell closest to the robot, a fixed search direction is employed (first along the x axis and then along the y axis). In case of the rectangular cuboid, the expansion is biased based on the larger side of the cuboid, as shown in the figure.

the equation defining the occupancy probability of cell g_i is derived. The probability that the cell g_i is free can then be similarly calculated as

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

Dividing (3) by (4) gives the odds,

$$\frac{P(g_i|z_{1:t})}{1 - P(g_i|z_{1:t})} = \frac{P(g_i|z_t)P(g_i|z_{1:t-1})(1 - P(g_i))}{(1 - P(g_i|z_t))(1 - P(g_i|z_{1:t-1}))P(g_i)},$$

which by simple algebraic manipulation can be transformed into [4]–[6]

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

which is a commonly used inverse sensor model in robotic mapping. $P(g_i|z_{1:t})$ represents the occupancy probability of the i^{th} grid cell given all observations. $P(g_i)$ represents the occupancy probability of a grid cell prior to any observations. $P(g_i|z_t)$ and $P(g_i|z_{1:t-1})$ represent the probability given the most current observation z_t and observations since the beginning of time until time $t-1$ respectively. In literature [4], [6], occupancy grids use a probability clamping threshold to prevent each cell of being over confident about its state. Following the same pattern the Rtree based occupancy grid defines a minimum and maximum probability threshold $\alpha_{\min}$, $\alpha_{\max}$ respectively after which a grid cell is no longer updated.

B. Adapting the grid cell resolution

The adaptation of the cell resolution is the process of reducing the number of cells required to represent the environment. Given a grid $G_t = \{g_1, g_2, \ldots, g_n\}$ at time index t consisting of n grid cells of cubic or multiple resolutions, the objective of the resolution adaptation process is to generate a grid $G_{t+1} = \{\bar{g}_1, \bar{g}_2, \ldots, \bar{g}_m\}$ (the bar indicating a modification of cell size) where $m \ll n$ by allowing the cells to fuse.

An important aspect within the fusion process is the selection of the cell g_i which is allowed to expand and fuse with its neighbourhood cells. In principle it is possible to allow all grid cells at the beam end point and along the beam path (given the sensor observations) to fuse. However, this strategy causes a substational increase in the computational cost, thus a different strategy is adopted. Consider the sensor observation $z_t = \{z_t^1, z_t^2, \ldots, z_t^n\}$ where z_t^i represents the i^{th} observation among the n point observations from a laser scanner at time index t. The occupancy grid updates the cell corresponding to the beam end point of the observation z_t^i and all cells that lie along the beam path. Given all beam end point observations (z_t^i, $i = 1, \ldots, n$) a set $T = \{g_1, \ldots, g_p\}$ composed of grid cells can be generated by randomly sampling cells along the beam path based on the beam length and always considering the beam end point. Figure 4(a) shows an illustration of the process of generating the set T.

The grid cells within the set T are allowed to expand and fuse with the neighbourhood cells. The pseudocode of the expansion and fusion process for any grid cell g_i in the set T is shown in Figure 5 and explained in detail here. The fusion process shown in Figure 5 is carried out after every sensor observation. As mentioned in Section III each grid cell g_i (or leaf branch g_i) contains the following

$$g_i = \begin{pmatrix}\mathbf{D}_i & P(g_i)\end{pmatrix},$$

where $\mathbf{D}_i = \begin{bmatrix}\mathbf{d}_i^{\min} & \mathbf{d}_i^{\max}\end{bmatrix}^T$ and $P(g_i)$ represents the occupancy probability. In the context of Rtree based occupancy grid, $\mathbf{d}_i^{\min} = \begin{bmatrix}x_i^{\min} & y_i^{\min} & z_i^{\min}\end{bmatrix}$ and $\mathbf{d}_i^{\max} = \begin{bmatrix}x_i^{\max} & y_i^{\max} & z_i^{\max}\end{bmatrix}$ represents the minimum and maximum values corresponding to the axis aligned rectangular cuboid in the global frame of reference. Given $\mathbf{d}_i^{\min}$ and $\mathbf{d}_i^{\max}$ the width vector $\mathbf{w}_i = \begin{bmatrix}w_i^x & w_i^y & w_i^z\end{bmatrix}$ can be easily extracted. The expansion process of the cell g_i in the Rtree based occupancy grid is defined (line 8 of Figure 5) as

$$\bar{\mathbf{D}}_i = \mathbf{D}_i +^j \bar{\mathbf{S}}_i,$$

for any specific search direction index j, where $\bar{\mathbf{S}}_i$ represents the search direction set. To explain the notation consider that $\bar{\mathbf{S}}_i = \{\mathbf{S}_i^x, \mathbf{S}_i^y, \mathbf{S}_i^z, \mathbf{S}_i^{-x}, \mathbf{S}_i^{-y}, \mathbf{S}_i^{-z}\}$, which states that the i^{th} grid cell should try to expand along the x axis, then along the y axis etc. The index j in ${}^j\bar{\mathbf{S}}_i$ represents the j^{th} element of

Fuse(g_i)

Input: g_i // cell g_i to be expanded
Outcome: $\bar{g}_i = (\bar{\mathbf{D}}_i \; P(\bar{g}_i))$ **or**
fusion not possible
// $\bar{g}_i$ is the fused grid cell

Procedure:

1 Determine the width vector $\mathbf{w}_i$ of g_i;
2 **If** (all elements of $\mathbf{w}_i$ of g_i are equal)
3 $\bar{\mathbf{S}}_i = \{\mathbf{S}_i^x, \mathbf{S}_i^y, \mathbf{S}_i^z, \mathbf{S}_i^{-x}, \mathbf{S}_i^{-y}, \mathbf{S}_i^{-z}\}$;
4 //first expand along x, then y etc.
5 **else**
6 Re-arrange $\bar{\mathbf{S}}_i$ based on $\mathbf{w}_i$;
7 **for-all** j ($j \leq |\bar{\mathbf{S}}_i|$) // $|\bar{\mathbf{S}}_i|$ is the number of elements in $\bar{\mathbf{S}}_i$
8 $\bar{\mathbf{D}}_i = \mathbf{D}_i +^j \bar{\mathbf{S}}_i$;
9 $\forall k$ such that $\mathbf{D}_k \in \bar{\mathbf{D}}_i$
10 **If** ($(P(g_i)$ **and** $P(g_k)) \leq \alpha_{min}$
or $(P(g_i)$ **and** $P(g_k)) \geq \alpha_{max}$)
11 Fuse cells to form $\bar{g}_i$;
12 Remove g_i and g_k from the grid;
13 **Fuse**($\bar{g}_i$); // recursive call
return;
14 **end for**;
15 return;

Fig. 5. The pseudocode describing the fusion process of the grid cells of the occupancy grid

the set $\bar{\mathbf{S}}_i$, hence the index $j = 1$ would correpond to $\mathbf{S}_i^x$ in the above example. The search direction set for any specific cell g_i is chosen based on the width vector $\mathbf{w}_i$. If all sides of the axis aligned rectangular cuboid are equal a fixed set of search directions (line 2-3 of Figure 5) are chosen otherwise it is biased based on the larger side of the rectangular cuboid (line 6 of Figure 5), as shown in Figure 4(b). The exact form of the search direction $\mathbf{S}_i^x$ is defined below

$$\mathbf{S}_i^x = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{W}_i^x \end{bmatrix} \begin{bmatrix} \mathbf{0}_{3\times1} \\ \boldsymbol{\sigma}_i^x \end{bmatrix} \tag{5}$$

where $\mathbf{0}_{m\times n}$ represents a zero matrix of m rows, n columns and $\mathbf{W}_i^x$ is a 3×3 matrix defined as

$$\mathbf{W}_i^x = \begin{bmatrix} \mathbf{w}_i \\ \mathbf{0}_{2\times3} \end{bmatrix},$$

and $\boldsymbol{\sigma}_i^x$ is a 3×1 unit vector (scaled based on width of rectangular cuboid) along the x dimension of the global reference frame. The basic operation being performed in (5) is the modification of the maximal x value of the axis aligned rectangular cuboid. In a similar manner other search directions such as $\mathbf{S}_i^y$, $\mathbf{S}_i^z$, $\mathbf{S}_i^{-x}$ etc. can be defined by replacing $\mathbf{W}_i^x$, $\boldsymbol{\sigma}_i^x$ and manipulating the structure of matrices (to change the maximum or minimal value of the rectangular cuboid). Given the expanded cell $\bar{\mathbf{D}}_i$ (line 8 of Figure 5) based on the search direction, fusion with *neighbouring cells* g_k is allowed if (line 9 of Figure 5)

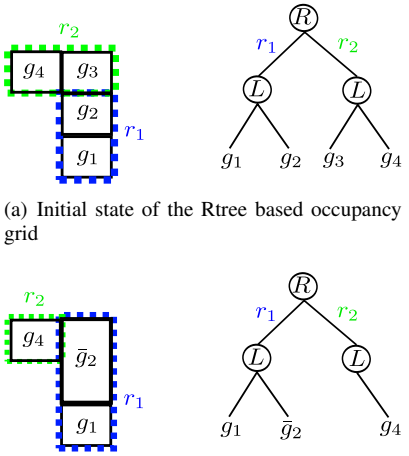
$$\forall k \text{ such that } \mathbf{D}_k \in \bar{\mathbf{D}}_i, \tag{6}$$

any of the following two conditions is satisfied (line 10 of Figure 5)

$$\forall k : P(g_k) \leq \alpha_{\min} \text{ and } P(g_i) \leq \alpha_{\min}, \tag{7}$$

or

$$\forall k : P(g_k) \geq \alpha_{\max} \text{ and } P(g_i) \geq \alpha_{\max}. \tag{8}$$



(a) Initial state of the Rtree based occupancy grid

(b) Final state after the fusion process

Fig. 6. An example scenario depicting the hierarchy adaptation of the Rtree occupancy grid based on the fusion process (colors have been added to aid visualization of the tree hierarchy and do not denote anything significant). The g_i are the grid cells in the hierarchy and r_1, r_2 represent the MBR. The following assumptions are made for the example scenario shown above: Firstly the probability of only cell g_2 and g_3 is above $\alpha_{\max}$ and the cell g_2 is chosen for expansion and tries to expand in the direction of cell g_3 (as discussed in Section IV-B). Secondly, the value of M is assumed to be 2 (as in Figure 2) a) The initial state of the hierarchy of the Rtree occupancy grid. b) The final state of the hierarchy after removal of expanding cell g_i (i = 2), g_k (k = 3) and insertion of the fused grid cell $\bar{g}_2$ (where $\mathbf{D}_2$, $\mathbf{D}_3 \in \bar{\mathbf{D}}_2$).

Expression (6) simply states that all rectangular cuboids $\mathbf{D}_k$ should be contained in the expanded rectangular cuboid $\bar{\mathbf{D}}_i$, whereas (7) and (8) state that the occupancy probability of each cell g_k should be below $\alpha_{\min}$ or above $\alpha_{\max}$ if the occupancy probability of cell g_i is below $\alpha_{\min}$ or above $\alpha_{\max}$ respectively. The objective of the constraints (7) and (8) is to limit the fusion to only those cells which have a high probability of being occupied or free and are no longer being updated as they are beyond the clamping probability thresholds ($\alpha_{\min}, \alpha_{\max}$). If the conditions stated above are satisfied the cells are fused to form $\bar{g}_i = (\bar{\mathbf{D}}_i \; P(\bar{g}_i))$. Additionally, cell g_i and cells g_k ($\forall k$ such that $\mathbf{D}_k \in \bar{\mathbf{D}}_i$) are removed from the grid. The probability of the fused cell is taken as an average probability of cells g_k ($\forall k$ such that $\mathbf{D}_k \in \bar{\mathbf{D}}_i$) that are contained in it (all occupancy probabilities are above $\alpha_{\max}$ or below $\alpha_{\min}$ based on (7) or (8)). The fusion function is called recursively (line 13 of Figure 5) after merging the cells to form $\bar{g}_i$. In case fusion is not possible (line 15 of Figure 5) the algorithm returns without any modification in the cell

size. This fusion process continues for all the elements of the set T. The description mentioned above discusses the fusion of grid cells on the level of leaf branches of the Rtree occupancy grid, however, the incremental adaptation process also causes a change in the tree hierarchy after every successful fusion. Figure 6 shows an example scenario depicting the hierarchy adaptation due to the incremental fusion process. Once a specific number of neighbouring cells g_k ($\forall k$ such that $\mathbf{D}_k \in \bar{\mathbf{D}}_i$) along with the expanding cell g_i have been chosen for fusion, they are first removed from the tree hierarchy (line 12) which causes a change in the size of MBRC being propagated up the hierarchy till the root. Additionally, a node might underflow (the number of branches might fall below d, see Section III) during this removal process; hence that specific node is removed and all leaf branches that are contained in the MBRC of that node are reinserted into the hierarchy based on the least expansion principle (see Section III). After the cells have been removed, the fused grid cell $\bar{g}_i$ is also inserted into the hierarchy based on the least expansion principle.

After the adaptation process, the new sensor observation z_{t+1} is used to update the occupancy values of the current grid G_{t+1} (Section IV A) followed by another fusion step (Section IV B). This recursive formulation of occupancy update and cell fusion continues for all sensor observations obtained by the robot. The search direction strategy shown in Figure 5 is chosen as it leads to the best results on the Freiburg campus dataset[1] which is used for evaluation in the experimental section. In principle the success of a specific search strategy for fusion is dependent on the structure of the environment which is unknown prior to the mapping process. The incremental fusion process presented in this paper is not restricted to any specific search strategy, rather it can be changed as per requirements or based on any prior information available about the environment. In the proposed approach $\alpha_{\min}$ and $\alpha_{\max}$ are set to very low and high occupancy probabilities respectively to ensure that stable regions of the occupancy grid are fused. Assuming a static environment this is a reasonable assumption. In addition, the fused regions of the Rtree based adaptive occupancy grid are constrained to be axis aligned as the proposed occupancy grid is composed of axis aligned rectangular cuboids. However, this axis aligned constraint is inherent to all occupancy grids as they are composed of axis aligned cubes.

V. Experimental Evaluation

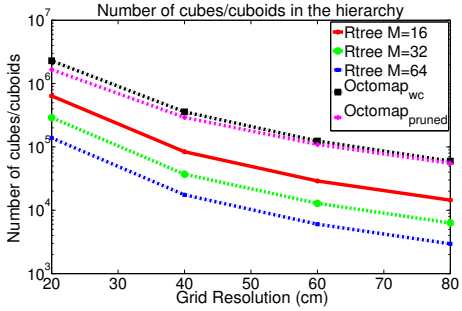
In this section the proposed adaptive occupancy grid is compared to the Octomap [4], [6] approach on the Freiburg campus dataset. The evaluation is based on the insertion, access time as well as the number of grid cells required for 3D representation. Octomap version 1.6.1 is used for this evaluation. The insertion time is defined as the time required to insert all laser scans into the grid. In context of the Rtree based adaptive occupancy grid it also includes the time taken by the fusion process of the grid cells. The access time of Octomap and the Rtree based adaptive occupancy grid corresponds to the time taken to access only the occupied grid cells given the entire hierarchy (consisting of occupied and free grid cells) after all scans have been inserted. The *graph2tree* tool (provided along with Octomap implementation) is used to determine the number of inner, leaf nodes and insertion time of Octomap. The access time of Octomap is determined by using the iterator based access method (on the pruned Octomap) after all scans were inserted. The parameter $\alpha_{\min}$ is set to 0.12, $\alpha_{\max}$ to 0.97 and the inverse sensor model is as follows

$$P(g_i|z_t) = \begin{cases} P_{\text{occ}} & \text{if beam is reflected within volume} \\ \\ P_{\text{free}} & \text{if beam traversed volume} \end{cases},$$

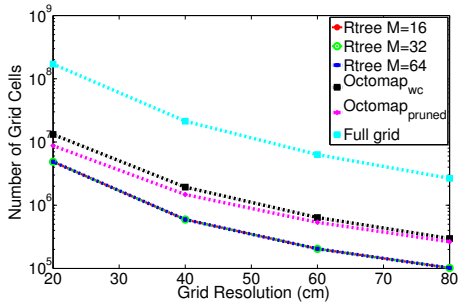
where P_{occ} is 0.7 and P_{free} is 0.4. These values ($\alpha_{\min}$, $\alpha_{\max}$, P_{occ}, P_{free}) are the same as mentioned in [4] and were fixed for all experiments discussed in this section. The evaluation is performed on a single core of an Intel i5-2500K 3.3 GHz processor with 16 GB RAM.

Figure 7(a) shows the number of cuboids required in the tree hierarchy by the Rtree based occupancy grid in comparison to the cubes required by Octomap (shown for two cases: *without compression (wc)* and *pruned*) whereas Figure 7(b) shows the number grid cells required by both approaches. It can be seen that the Rtree based adaptive occupancy grid requires fewer grid cells as well as cuboids in the tree hierarchy in comparison to the cubes and grid cells of the Octomap approach. Focusing on cubes/cuboids required in the tree hierarchy first, two main reasons can be attributed to this, firstly Octomap (based on Octrees) has a pre-defined hierarchy consisting of cubes with the number of children per node fixed to 8. In contrast, the nodes in the Rtree based occupancy grid can contain an arbitrary maximum number of children (M) which can effectively reduce the number of nodes required (as discussed in Section III). Additionally the MBRC in the Rtree based adaptive occupancy grid hierarchy can overlap and are not constrained to be cubic. Considering the number of grid cells required for representation as shown in Figure 7(b), the comparison between pruned Octomap and the Rtree based adaptive occupancy grid is interesting. The Octomap approach uses the $\alpha_{\min}$ and $\alpha_{\max}$ threshold to prune out regions of the Octree hierarchy (nodes and leaves) to achieve compression whereas the Rtree based adaptive occupancy grid approach uses these parameters for fusion of leaf branches only. The reduction in the number of grid cells required to represent the environment by the Rtree based adaptive occupancy grid in contrast to pruned Octomap is **28.51**% at 20 cm resolution grid. The amount of grid cells required by the full 3D grid (or standard occupancy grid) as shown in Figure 7(b) is calculated based on $\frac{x \times y \times z}{r^3}$ [4] where x, y and z is the minimal bounding box in each dimension (292 $\times$ 167 $\times$ 28 m for the Freiburg campus dataset) and r represents the resolution of the grid in meters. It is important to specify here that for a fixed grid resolution the branching factor M does not influence the number of

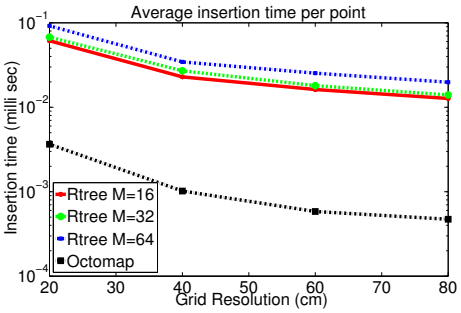
[1]Courtesy of B. Steder and R Kümmerle, available at http://ais.informatik.uni-freiburg.de/projects/datasets/octomap/



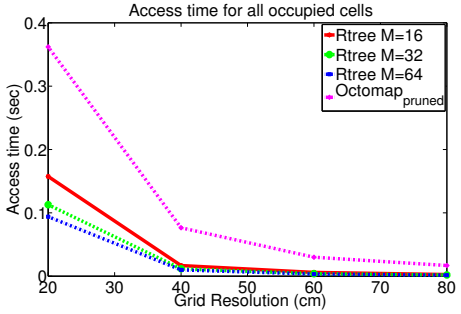
(a) Number of required cubes (by Octomap) and cuboids (proposed approach) in the tree hierarchy (Semilog plot)



(b) Number of grid cells (Semilog plot)



(c) Average insertion time per point (Semilog plot)

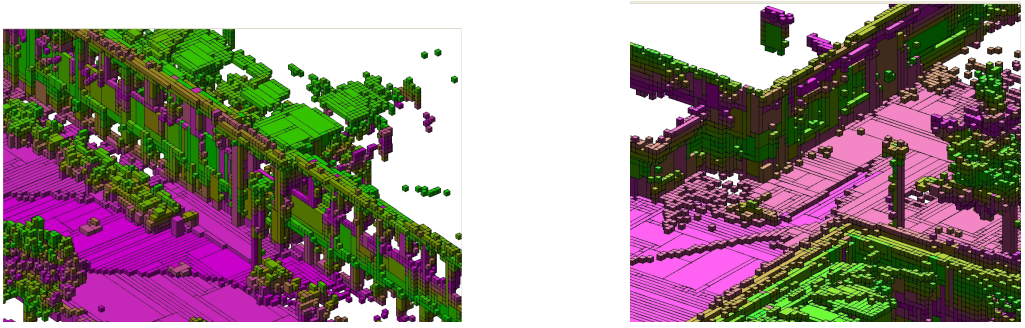


(d) Access time for occupied cells only, given the complete tree hierarchy consisting of occupied and free cells

Fig. 7. Results on all 81 scans of the Freiburg campus dataset. In context of Octomap the notation *wc* means without compression, pruned represents pruned compression. In context of the Rtree based adaptive occupancy grid M represents the maximum number of branches allowed per node. (a) The number of cubes and cuboids required by Octomap and the Rtree based adaptive occupancy grid in the tree hierarchy (not including the leaves). An increase in parameter M causes the Rtree to effectively reduce the number of cuboids required for representation in the hierarchy (see Section III). (b) The number of grid cells as a function of the grid resolution shown as a semilog plot. The grid cells (leaf branches) required by the Rtree based adaptive occupancy grid is less than Octomap due to the incremental fusion process which leads to axis aligned rectangular cuboids whereas in context of Octomap the cells are constrained to be cubic. A variation in parameter M for a fixed grid resolution does not cause any change in the number of grid cells required to represent the environment. The formula used to calculate the number of grid cells for the full 3D grid (or standard occupancy grid) is the same as in [4] (c) The average insertion time (per point) of the Octomap and the Rtree based adaptive occupancy grid. An increase in parameter M causes the insertion time to increase due to increased overlaps. (d) Access times for occupied grid cells only given the entire hierarchy consisting of free and occupied grid cells.

grid cells required for representing the environment nor the fusion process. A comparison with the maximum likelihood compression of Octomap is not performed in this paper as it involves thresholding (either occupied or free) all the nodes of the Octree. Due to its lossy nature, this thresholding process might lead to an inaccurate environment representation. In addition, it also causes a loss of all the probabilistic information thereby making the comparison unfair. This probabilistic information is essential for the Rtree based adaptive occupancy grid as the entire resolution adaptation process is based on it. Consequently, this would prevent probabilistic fusion of grid cells in case the robot receives additional sensor observations.

Figure 7(c) shows the normalized insertion time per point of the Rtree based adaptive occupancy grid and Octomap. The Rtree based adaptive occupancy grid is slower than Octomap due to multiple reasons. The Rtree based adaptive occupancy grid incrementally generates the tree hierarchy based on node splitting and least expansion as observations are obtained whereas Octomap has a predefined hierarchy consisting of cubes. Additionally, the fusion process of the grid cells in the Rtree based adaptive occupancy grid causes a variation in the tree hierarchy. The variation in grid cells (leaf branches) is propagated up the hierarchy leading to a change in the MBRC of the inner branches. Finally, the overlaps between the MBRC of the inner branches in the Rtree based occupancy grid can also slow down the query/search process. An increase in parameter M increases the tree width as well as the insertion time because of increased overlaps between MBRC. Figure 7(d) shows the time required to access all the occupied cells in the grid given the entire hierarchy composed of occupied and free grid cells. It can be seen that the Rtree based adaptive occupancy grid is capable of accessing the occupied cells faster than Octomap. An increase in the branching factor M reduces the number of nodes required in the Rtree hierarchy (see Section III) and causes the access time of occupied cells to decrease as can be seen in Figure 7(d). Figure 8 shows examples of the axis aligned rectangular cuboids generated by the Rtree based adaptive occupancy grid for the occupied regions on the Freiburg campus dataset. The fused free space regions are not shown in the figure for the ease of visualization.



(a) Visualization of fused occupied grid cells

(b) Visualization of fused occupied grid cells

Fig. 8. Visualization of fused occupied grid cells on the Freiburg campus dataset (colors have been assigned for the ease of visualization).

VI. Conclusion and Future Work

In this paper an extension of the standard occupancy grid has been presented. The proposed approach adds an incremental fusion process after the occupancy update and represents the environment using axis aligned rectangular cuboids instead of cubic grid cells. An evaluation of the proposed Rtree based adaptive occupancy grid is performed on a publically available dataset and additionally it is compared to the state of the art Octomap approach. The evaluation shows that the proposed approach requires fewer grid cells for 3D environment representation and has faster access times for occupied cells. The fusion process in the proposed occupancy grid is limited to axis aligned regions, however, this axis aligned constraint is inherent to all occupancy grids as they consist of axis aligned cubes. Future work will include an evaluation of the fusion process for dynamic environments as the scope of this paper has been limited to static environments. The proposed fusion process is easily extendable to environments containing dynamics. This extension is possible by splitting the fused cells of the dynamic region based on the chosen resolution of the grid if the occupancy probability falls below or goes above the clamping threshold. Additionally, future work will also include an evaluation of different search strategies for the grid cell fusion process of the Rtree based adaptive occupancy grid.

Acknowledgment

This work is supported in part within the ERC Advanced Grant SHRINE Agreement No. 267877 (www.shrine-project.eu). Support of the TUM Institute for Advanced Study (IAS), Technische Universität München (TUM), is hereby gratefully acknowledged, see also www.tum-ias.de.

References

[1] S. Thrun, "Learning metric-topological maps for indoor mobile robot navigation," *Artificial Intelligence*, vol. 99, no. 1, pp. 21–71, 1998.

[2] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

[3] S. Thrun, "Learning occupancy grid maps with forward sensor models," *Autonomous robots*, vol. 15, no. 2, pp. 111–127, 2003.

[4] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous Robots*, pp. 1–18, 2013.

[5] H. Moravec and A. Elfes, "High resolution maps from wide angle sonar," in *IEEE International Conference on Robotics and Automation*, vol. 2, 1985, pp. 116–121.

[6] K. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: A probabilistic, flexible, and compact 3d map representation for robotic systems," in *ICRA workshop on best practice in 3D perception and modeling for mobile manipulation*, 2010.

[7] A. Guttman, "R-trees: A dynamic index structure for spatial searching," *ACM*, vol. 14, no. 2, 1984.

[8] M. Herbert, C. Caillas, E. Krotkov, I. S. Kweon, and T. Kanade, "Terrain mapping for a roving planetary explorer," in *IEEE International Conference on Robotics and Automation*, 1989, pp. 997–1002.

[9] R. Triebel, P. Pfaff, and W. Burgard, "Multi-level surface maps for outdoor terrain mapping and loop closing," in *IEEE International Conference on Intelligent Robots and Systems*, 2006, pp. 2276–2282.

[10] I. Dryanovski, W. Morris, and J. Xiao, "Multi-volume occupancy grids: An efficient probabilistic 3d mapping model for micro aerial vehicles," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010, pp. 1553–1559.

[11] E. Einhorn, C. Schroter, and H.-M. Gross, "Finding the adequate resolution for grid mapping-cell sizes locally adapting on-the-fly," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 1843–1848.

[12] M. Magnusson, A. Nüchter, C. Lörken, A. J. Lilienthal, and J. Hertzberg, "Evaluation of 3d registration reliability and speed, a comparison of icp and ndt," in *IEEE International Conference on Robotics and Automation*, 2009.

[13] J. Saarinen, H. Andreasson, T. Stoyanov, J. Ala-Luhtala, and A. J. Lilienthal, "Normal distributions transform occupancy maps: Application to large-scale online 3d mapping," in *IEEE International Conference on Robotics and Automation*, 2013.

[14] J. P. Saarinen, H. Andreasson, T. Stoyanov, and A. J. Lilienthal, "3d normal distributions transform occupancy maps: an efficient representation for mapping in dynamic environments," *The International Journal of Robotics Research*, vol. 32, no. 14, pp. 1627–1644, 2013.

[15] S. Khan, A. Dometios, C. Verginis, C. Tzafestas, D. Wollherr, and M. Buss, "Rmap: a rectangular cuboid approximation framework for 3d environment mapping," *Autonomous Robots*, pp. 1–17, 2014.

[16] A. Nanopoulos, A. N. Papadopoulos, and Y. Theodoridis, "R-trees: Theory and applications," *Springer*, 2006.