Hierarchies of Octrees for Efficient 3D Mapping

Kai M. Wurm Daniel Hennes Dirk Holz Radu B. Rusu Cyrill Stachniss Kurt Konolige Wolfram Burgard

***Abstract*— In this paper, we present a novel multi-resolution approach to efficiently mapping 3D environments. Our representation models the environment as a hierarchy of probabilistic 3D maps, in which each submap is updated and transformed individually. In addition to the formal description of the approach, we present an implementation for tabletop manipulation tasks and an information-driven exploration algorithm for autonomously building a hierarchical map from sensor data. We evaluate our approach using real-world as well as simulated data. The results demonstrate that our method is able to efficiently represent 3D environments at high levels of detail. Compared to a monolithic approach, our maps can be generated significantly faster while requiring significantly less memory.**

I. Introduction

As more and more systems for mobile manipulation emerge, the need for adequate three-dimensional models of the environment becomes evident as manipulation tasks typically require highly detailed models for grasping and navigation. In addition to providing the appropriate accuracy, these models should also be efficient and updatable so that objects can be added, removed, or rearranged in the model.

In general, a map of the environment cannot be assumed to be given and therefore need to be learned from sensor readings. Most existing mapping systems treat the environment as static and integrate all sensor readings into one monolithic map. Such monolithic maps, however, are unable to represent movable structures or objects in the environment. Furthermore, many of those approaches assume a uniform and fixed set of parameters such as the maximum level of detail and provide no means for adapting the representation locally according to the geometric or semantic structure of the environment.

In this paper, we propose a hierarchical data structure to model 3D environments. We model the environment as a tree of probabilistic multi-resolution maps. In this tree, each node represents a subspace of the environment. The subdivision applied in our system is based on a spatial relation that can be defined by the user. Fig. 1 gives an illustration of a hierarchy according to a relation based on *supporting planes*. We generate the maps from 3D range measurements taken from known sensor origins. Corresponding sensor modalities for acquiring such measurements include laser range finders and stereo cameras installed on a mobile robot. Estimating the poses of the sensor based on appropriate 3D SLAM methods is beyond the scope of this paper.

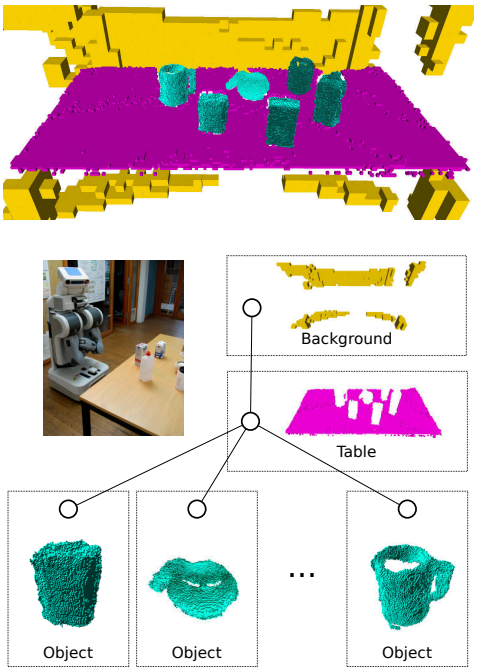


Fig. 1. Top: model of a table top scene. Bottom: illustration of the map hierarchy. Each map node is represented in a separate octree map.

Compared to a monolithic map of the environment our hierarchical approach shows a number of advantages. First, each submap is maintained independently and mapping parameters such as the resolution can be adapted for each submap. Second, submaps can be manipulated independently. For example, some submaps representing individual objects can be moved while others remain static. Third, nodes that share a common property can be combined in the hierarchy. For example, all objects on a table can be associated to this table and then are moved along when the table is moved.

The remainder of this paper is organized as follows. After discussing related work, we give a general description of our mapping framework and its components. We describe a specific implementation for tabletop manipulation in Sec. IV. In Sec. V, we present an exploration algorithm for autonomously acquiring models for tabletop manipulation using this implementation. In Sec. VI we present an experimental evaluation of the proposed approach.

K.M. Wurm, C. Stachniss and W. Burgard are with the University of Freiburg, Department of Computer Science, 79110 Freiburg, Germany. D. Hennes is with Maastricht University, Department of Knowledge Engineering, 6200 MD Maastricht, The Netherlands. D. Holz is with the Autonomous Intelligent Systems Group, University of Bonn, Germany. R.B. Rusu and K. Konolige are with Willow Garage Inc., Menlo Park, CA, USA

This work has partly been supported by the DFG under SFB/TR-8, by the European Commission under FP7-248258-First-MM, and by Microsoft Research, Redmond.

II. Related Work

Building three-dimensional models of the environment has been an active field of research in the robotics community for several decades. Popular representations include voxel grids, point clouds, octrees, and surfels.

Whereas point clouds provide high metric accuracy [2], [13], they cannot distinguish between or explicitly represent free space and unknown areas. They furthermore are memory-intensive and cannot be updated efficiently to adapt to changes in the environment.

Surfels [6], which recently have been used successfully in the context of object mapping [7], [22], only represent the surface of the environment and do not represent freespace or unknown volumes. For this reason, they are based on strong assumptions about the corresponding environment, e.g., in mobile manipulation scenarios where the knowledge about the freespace is essential for safe navigation.

Payeur *et al.* [15] used octrees [9] to adapt occupancy grid mapping from 2D to 3D and thereby introduced a probabilistic way of modeling occupied and free space. In contrast to our method, these previous approaches did not support multi-resolution hierarchies of maps.

A number of previous approaches use a hierarchy or collection of maps instead of one global map. Popular methods represent objects based on collections of 2D grid maps [1], [4]. Petrovskaya *et al.* [16] represent movable objects in a 2D map at a high resolution while the background is represented at a coarse resolution. Douillard *et al.* [3] combine a coarse elevation map for background structures with object voxel maps at a higher resolution to perform 3D segmentation. In contrast to our approach, they do not organize submaps in a hierarchy and they do not integrate multiple measurements into the model.

There exist several methods that consider semantic information in the context of 3D mapping. Nüchter *et al.* segment and classify 3D point clouds to improve scan registration [14] and to detect objects [12]. Rusu *et al.* [19] analyze segmented point clouds to derive functional properties of the environment in a household context. These approaches, however, are based on a point cloud representation, which has the disadvantages discussed above.

III. Hierarchical 3D Mapping Framework

In the following, we will introduce the proposed data structure, explain how to generate a hierarchy of maps based on a spatial relation, and show how such a hierarchical representation can be updated consistently. We assume that the robot always has an accurate pose estimate. This, for example, can be achieved by applying MCL followed by an ICP-based local registration. Such an approach can lead to high-precision pose estimates [11].

A. Map Hierarchy

In our representation, a map M consists of a tree of map nodes

$$M = \{n_1, \ldots, n_k\}, \tag{1}$$

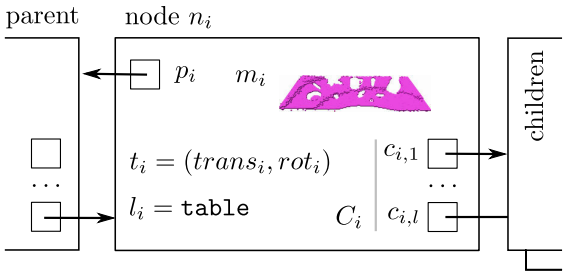


Fig. 2. Illustration of a map node.

where k is the number of map nodes in the tree. A node n_i is defined as

$$n_i = (m_i, t_i, p_i, C_i, l_i) \tag{2}$$

(see Fig. 2 for an illustration). Each node stores a 3D map m_i. Additionally, it holds a reference p_i to its parent node and a set of references to child nodes

$$C_i = \{c_{i,1}, \ldots, c_{i,l}\}. \tag{3}$$

A semantic label l_i is stored in each map node. We will discuss possible uses for such a label in Sec. III-E.

Map m_i is interpreted with respect to the node's reference frame t_i. The 6D transform t_i denotes the relative transform from the parent's reference frame to the node's local reference frame. To recover the global reference frame t_i^{global} of node n_i, the tree is traversed upwards from n_i to the root of the tree using the corresponding parent references. This leads to the following recursive formulation

$$t_i^{global} = \begin{cases} t_{p_i}^{global} \circ t_i & \text{, if } p_i \text{ defined} \\ t_i & \text{, } n_i \text{ is root node} \end{cases}, \tag{4}$$

where $t_{p_i}^{global}$ denotes the global reference frame of n_i's parent p_i and $\circ$ is the composition operator.

There are certain situations which cannot be modeled using a tree structure (e.g., one object resting on two tables). Assuming this structure, however, allows us to update the reference frames of map nodes recursively. The tree can also be used to locate map nodes more efficiently.

B. Construction of the Hierarchy

Throughout this paper we assume that 3D measurements are given as a pair (z, o_z) consisting of a cloud of endpoints z and the corresponding sensor origin o_z.

We assume that techniques exist to analyze point cloud measurements for meaningful structures. More specifically, we assume that there exists a method $S(z)$ which computes a segmentation of a point cloud measurement z so that

$$S(z) = \{z_1, \ldots, z_m\},\ z = \bigcup_i z_i, \tag{5}$$

where each segment z_i corresponds to a structure such as surface planes, geometric primitives, or point clusters.

The map hierarchy is based on a spatial relation that is provided by the user. This relation $r(n_i, n_j) \subseteq M \times M$ determines whether node n_i is at a higher level in the

hierarchy than n_j. Real-world examples of such relations include the relations *supports* or *contains*.

Whenever a new node n_{new} is added to the hierarchy M, relation r is evaluated for each existing node and a set A of candidate ancestors is generated

$$A = \{n \in M \mid (n, n_{new}) \in r\} \tag{6}$$

In general, A can consist of more than one candidate. For example, a cup standing on a table *is supported by* the table but also by the floor the table is standing on. For this reason, n_{new} is added to the hierarchy as a child of the deepest candidate node a^* in the hierarchy

$$a^* = \operatorname*{argmax}_{a \in A} \operatorname{depth}(a), \tag{7}$$

where $\operatorname{depth}(a)$ denotes the depth of a in the tree.

More than one relation may be used to define the hierarchy as long as they are mutually exclusive, that is, no more than one relation contains the tuple (n, n_{new}) for a given node n. In this case, all relations $r \in R$ in a set of relations are evaluated in Eq. (6). Using more than one relation allows the hierarchy to model, for example, that objects are *supported by* a table and that this table is *contained in* a specific room. Note that an object supported by the table could not be *contained in* the table at the same time.

C. Map Update

We now describe how a 3D measurement is integrated into the hierarchical map. To integrate a measurement into an existing hierarchy, we assume that an association function f_A exists. Given a point cloud measurement z, this function returns the lowest node $n \in M$ in the tree that z falls into or the empty set if no such existing node can be found. Given this association function and a segmentation function S, the model update follows three basic steps:

1) Segment the point cloud $\{z_i\} = S(z)$
2) Associate each segment z_i to an existing map node using f_A (multiple segments can be associated to the same map node).
3) Update submaps determined in previous step using the corresponding segments and the sensor origin o_z or create a new map node if no corresponding node could be found.

D. Transform Update

In addition to 3D range measurements, perceptions of object movements are integrated into the model. In our approach, a perceived relative 6D-transformation t (movement and change of orientation) of a submap n_i is integrated by updating the corresponding node transform $t_i \leftarrow t_i \circ t$. Such transforms can be measured using object detection methods, e.g., based on viewpoint feature histograms [18].

In general, there are also unobserved transformations of the environment and objects can be removed from the environment entirely. To cope with such cases, we estimate a probability of the existence for each node n. This probability $P(n \mid z_{1:t})$, where $z_{1:t}$ denotes the set of all measurements up to time t, is estimated using a binary Bayes filter [10]

$$P(n \mid z_{1:t}) = \left[1 + \frac{1 - P(n \mid z_t)}{P(n \mid z_t)} \frac{1 - P(n \mid z_{1:t-1})}{P(n \mid z_{1:t-1})} \frac{P(n)}{1 - P(n)}\right]^{-1}. \tag{8}$$

The inverse sensor model $P(n \mid z_t)$ is specific to the sensor and the segmentation used for mapping.

E. Semantic Annotation

Semantic annotations can be used, for instance, to facilitate data association or to adapt certain map properties. For this reason, a label l_i is stored in each map node n_i. To determine the label, we assume that a labeling function $L(z)$ exists that returns a label given a point cloud z. L can be implemented using object detection algorithms (e.g., the approach of Ruhnke et al. [17]) or scene analysis methods (e.g., the approach presented in [12]).

IV. Representation for Tabletop Manipulation

We will now describe a specific implementation of a map hierarchy H' along with a segmentation S', labeling function L', relation r', and association function f'_A. This implementation is used to represent an indoor environment for tabletop manipulation tasks.

In a tabletop manipulation scenario, objects on a tabletop need to be represented separately so that they can be updated and manipulated independently. In our implementation of H', we differentiate between objects, tables and the remaining structures, such as the floor and the walls, which are part of the scene background (see Fig. 1 for an illustration). Our model is designed to represent the entire working space of the robot. At the same time it represents objects at a fine resolution to facilitate, e.g., grasp computation without relying on predefined object models.

A. Construction of the Hierarchy

The map hierarchy H' is based on a supporting planes assumption, i.e., we follow the common assumption that objects rest on flat tabletops. To identify such supporting structures we analyze surface elements in the 3D point clouds.

To implement the segmentation $S'(z)$, we first locate horizontal planes at approximate table height (e.g., 0.5 m-1.0 m). The sets of table inliers are stored in segments $z_{table,1}$ to $z_{table,n}$.

These planes are then used to segment object points from the measurement point cloud. Objects are defined as measurements above a detected table plane. To segment those points into individual object measurements $z_{obj,1}$ to $z_{obj,m}$, the object points are clustered based on a threshold on the Euclidean point distance (e.g. 0.01 m). All points that do not belong to an object cluster and are not table inliers, are stored in a background segment z_{bg}. S' is then defined as

$$S'(z) = \{z_{bg}, z_{table,1}, \ldots, z_{table,n}, z_{obj,1}, \ldots, z_{obj,m}\}. \tag{9}$$

The labeling function $L'(z)$ is defined based on S':

$$L'(z) = \begin{cases} \texttt{table} & , z \in \{z_{table,1}, \ldots, z_{table,n}\} \\ \texttt{object} & , z \in \{z_{obj,1}, \ldots, z_{obj,m}\} \\ \texttt{background} & , \text{else} \end{cases} . \tag{10}$$

The spatial relation $r'(n_i, n_j)$ is defined based on the node label:

$$\begin{aligned} r' = \{\ & (n_i, n_j) \mid n_i, n_j \in M', \\ & (l_i = \texttt{table} \wedge l_j = \texttt{object}) \vee \\ & (l_i = \texttt{background} \wedge l_j = \texttt{table}) \vee \\ & (l_i = \texttt{background} \wedge l_j = \texttt{object})\ \}. \end{aligned} \tag{11}$$

When a new node n_i is added to the hierarchy, we use the first point cloud measurement that is integrated into the node to determine its reference frame t_i relative to the parent node p_i. The translational component is computed from the centroid of the measurement and we use principal component analysis to determine its orientation. While this cannot guarantee an optimal orientation of the reference frame with respect to all future measurements that are integrated into n_i, it did result in smoother object surfaces in our experiments.

B. Octree Maps

We maintain node maps using the OctoMap mapping framework [23] (available at `http://octomap.sf.net`). This octree-based system offers flexibility with regard to the mapped area and resolution. It performs a probabilistic occupancy estimation to ensure updatability and to cope with sensor noise. In this way, it models occupied as well as free volumes and implicitly also volumes that have not been measured. Furthermore, a lossless compression method ensures the compactness of the resulting models.

All map updates are local with respect to the map node's reference frame. Let (z, o_z) be a measurement and n_i the map node to be updated. The measurement and corresponding origin are transformed using the inverse global transform $(t_i^{global})^{-1}$ of n_i:

$$(z', o'_z) = ((t_i^{global})^{-1}(z), (t_i^{global})^{-1}(o_z)). \tag{12}$$

Map m_i is then updated using (z', o'_z).

To efficiently determine those voxels that need to be updated, we perform a ray-casting operation from o'_z to each measurement endpoint in z'. We update volumes along the beam using occupancy grid mapping as introduced by Moravec and Elfes [10]. For the sake of efficiency, we update only those freespace voxels along the beam that fall within the oriented bounding box of m_i. For further details we refer the reader to the work of Wurm *et al.* [23].

An important advantage of the proposed map hierarchy is the ability to adapt the mapping resolution based on the semantic class. In the context of mobile manipulation, for instance, objects usually need to be modeled at very fine resolutions (e.g., millimeters) while a table top can be modeled at a coarser resolution (e.g., centimeters) and walls and the floor can be modeled even coarser. As we will show in the experiments, this multi-resolution approach results in a significant reduction in both memory consumption and computation time.

Algorithm 1 Autonomous Model Acquisition.

```
M_0 ← ∅, t ← 0
z_0 ← getMeasurement
M_1 ← updateModel (M_0, z_0)
repeat
  table ← findNearestUnexploredTable(M_t)
  m_trav ← computeTraversable(M_t)
  V ← sampleViewpoints (m_trav, table, radius)
  for all v ∈ V do
    u(v) ← α I(M_t, v) − (1 − α) cost(m_trav, x_t, v)
  end for
  v* ← argmax_{v∈V} u(v)
  if u(v*) > τ then
    moveBase(v*)
    z_t ← getMeasurement
    M_{t+1} ← updateModel (M_t, z_t)
  end if
  t ← t + 1
until u(v*) ≤ τ
```

C. Node Association

To implement an association module f'_A for table top manipulation, we make use of oriented bounding boxes (OBBs).

Given a point cloud segment z, we first determine the OBB of z using principle component analysis. We then use its semantic label $L'(z)$ to find existing map nodes in the model M' with the same label. For each node n_i out of this set we compute its OBB. This can be done efficiently using the reference frame t_i and the map extend of m_i. We then test the bounding box of z for intersection with the bounding boxes of the map nodes. This test can be performed efficiently using Gottschalk's algorithm [5].

V. Autonomous Model Acquisition for Tabletop Manipulation

The previous sections explained how to generate a model from given sensor data. In this section, we consider the problem of learning a model of a tabletop environment in an autonomous fashion. Our exploration system is an information-driven approach. It takes into account the expected information of potential, future observations that are obtained when carrying out a certain action. Our approach can be seen as an extension of the exploration system of Stachniss et al. [20] towards 3D environments.

We assume that the robot consists of a movable platform that is equipped with a 3D range sensor and that the environment contains at least one table with some objects to be explored. We furthermore assume that a table can be detected in an initial measurement and that all relevant objects can be sufficiently measured by moving the robot around the table.

The key idea of our exploration approach is to sample potential target location from which the area of interest, i.e., the table, can be observed. Then, our approach estimates the exploration cost and the expected information gain for each target location. The robot selects the target location that provides the best trade-off between travel cost and expected information gain. The algorithm for this procedure is given as pseudo code in Alg. 1.

We initialize the exploration by performing a measurement of the robot's surroundings from its current location and generating an initial model of the world M_1 as described in Sec. IV. From this initial model, we select the closest unexplored table as our area of interest.

After initialization, we execute an iterative process of generating, evaluating, and selecting possible targets. To generate exploration targets around the selected table, we first estimate the traversable area. Since the robot's base moves on the ground only, we project all obstacles from the 3D model with which the robot could collide onto a 2D traversability grid map m_{trav}. This is similar to the approach presented by Strand *et al.* [21]. We then sample a set V of robot poses from the traversable area around the selected table in a given radius (e.g., 2 m).

Next, the potential view points are evaluated with respect to their exploration cost relative to the current pose x_t and with respect to the expected information gain, i.e. the expected reduction of uncertainty in M_t caused by the potential measurements to be obtained at the view point. For each potential view point $v \in V$, we determine its utility as

$$u(v) = \alpha\ I(M_t; v) - (1-\alpha)\ \mathrm{cost}(m_{\text{trav}}, x_t, v), \qquad (13)$$

where $I(M_t; v)$ is the expected information gain of v, M_t being the current hierarchical model, and $\mathrm{cost}(m_{\text{trav}}, x_t, v)$ the function that estimates the corresponding exploration cost; α is a constant factor to trade off cost against gain which is determined heuristically. In our approach, the exploration cost considers the constant time for performing a measurement plus the expected travel time from x_t to v. The second quantity can be estimated using the traversability map m_{trav} and a path-planning algorithm such as A^*.

The expected information gain is defined as the expected reduction of the entropy H in the model caused by considering the potential observations that are obtained at the target location. We refer to [8] for a compact and precise description of information-theoretic concepts. The information gain $IG(M_t; z)$ for a given measurement z is given by

$$IG(M_t; z) = H(M_t) - H(M_t \mid z). \qquad (14)$$

Since the measurement that will be obtained at the target location is not known, one has to integrate over all possible measurements. This results in the expected information gain of a target location v as

$$I(M_t; v) = \int_z p(z \mid M_t, v)\ IG(M_t; z)\, dz. \qquad (15)$$

Since integrating over all possible observations is infeasible, strong assumptions have to be made to approximate Eq. (15).

We can substantially simplify the computation by assuming first that all measurements pass through free space cells in M_t, measure an obstacle when reaching an occupied cell in M_t, and are reflected with a uniform probability when traversing unmapped cells. Second, we assume that only previously unknown cells contribute to a change of entropy. This is clearly not the case but the uncertainty reduction of such cells typically is the dominant effect compared to cells which are already known well. This strong assumption, however, makes the estimation of the expected information gain efficiently so that the it can be computed online—which is important for exploration tasks.

Under these two assumptions, we can efficiently approximate Eq. (15) by performing ray-casting operations from the potential view point towards the area of interest. We can actually compute $IG(M_t; z)$ in closed form by considering the number of unknown cell along each ray together with the change in entropy ΔH that is caused by measuring an unknown cell. We can derive ΔH directly from the sensor model.

Finally, we can determine the best next viewpoint v^* based on the utility of the sampled viewpoints:

$$v^* = \operatorname*{argmax}_{v \in V}\ u(v). \qquad (16)$$

As long as there are unexplored targets, the robot base is moved to v^* and a 3D measurement z_t is taken and integrated into the model M_t.

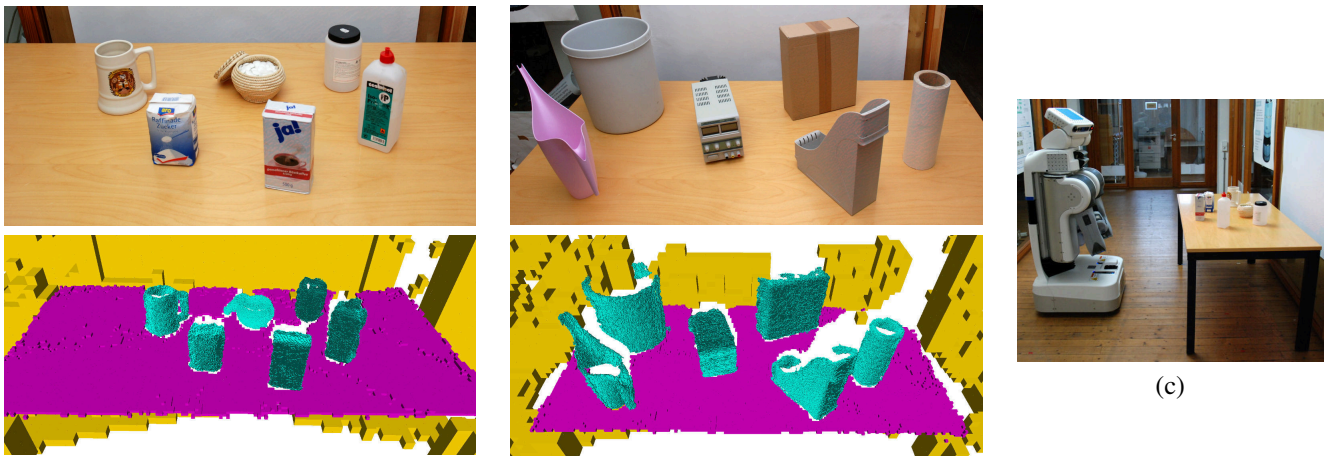
VI. Experiments

We evaluated approach described above in a number of experiments based on real and simulated data. In the experiments, we compared the hierarchical structure to a monolithic 3D map with respect to memory consumption, runtime, and mapping results in the presence of changes in the environment. We also evaluated the exploration system introduced in Sec. V in a simulated environment.

A. Tabletop Mapping

The first experiment is designed to show that the proposed model is able to efficiently represent real-world scenes. In this experiment, we used a PR2 robot to acquire 3D measurements. The robot is equipped with a stereo camera and a texture projector to improve stereo quality. The experimental setup can be seen in Fig. 3 (c). Please note that the presented approach is not limited to stereo images and could be applied using any 3D range sensor such as a tilting laser scanner or similar devices.

Two sets of objects were scanned by sweeping the stereo camera across the scene (see Fig. 3), taking 20 overlapping dense stereo images each. The measurements were integrated into a 3D model as introduced in Sec. IV. In the hierarchical map, the resolution was adapted using the semantic label. In all experiments, submaps with the label `table` and `background` were mapped at a resolution of 0.01 m and 0.05 m respectively. These values are commonly used in navigation. Nodes with the label `object`, were mapped at various different resolutions of 1 mm, 2 mm, 4 mm, 8 mm,



(a) (b) (c)

Fig. 3. (a),(b) Photos of objects and visualizations of models. In the visualizations, objects are shown in cyan, the table is displayed in magenta and yellow voxels represent the background. An object mapping resolution of 2 mm is shown. Voxels with an occupancy probability of less than 0.5 are considered freespace and are not visualized. (c) Experimental setup.

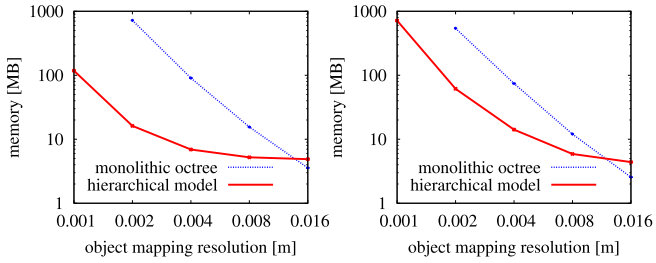


Fig. 4. Memory usage of models. Left: small objects (Fig. 3a), right: big objects (Fig. 3b). Note that a logarithmic scale is used in the plots. At an object resolution of 16 mm the monolithic map consumes slightly less memory since our implementation of the hierarchical map represents the table class at a fixed resolution of 10 mm.

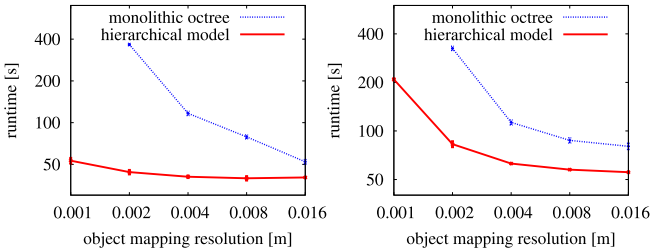


Fig. 5. Runtime to integrate all measurements. Left: small objects (Fig. 3a), right: big objects (Fig. 3b). Note that a logarithmic scale is used in the plots.

and 16 mm. A visualization of the resulting models can be seen in Fig. 3.

For comparison, all measurements were also integrated into a single monolithic octree map at the resolution used for the `object` class. In this case, segmentation was ignored.

Each mapping experiment was repeated five times on an Intel® Core™ i7-based desktop computer. A comparison of memory consumption and overall runtime can be seen in Fig. 4 and Fig. 5. Please note that a logarithmic scale is used in the plots. From the results it can be seen that the hierarchical model consumes significantly less memory and is updated significantly faster. At very fine resolutions the monolithic map is about one order of magnitude bigger and it takes about one order of magnitude longer to compute the model compared to our hierarchical map. A mapping resolution of 1 mm could not be evaluated in the case of monolithic maps since they would require in the order of 10 GB of memory.

B. Scene Update Under Occlusion

The second experiment is designed to show that our representation is able to adapt to changes in the environment. In this experiment, two objects are mapped in a sequence of measurements illustrated in Fig. 6. The first object (power supply) is measured and integrated into the map. Then the second object (tube) is placed in the scene so that it partially occludes the first and then the scene is measured again. The last measurements are taken after the first object was removed from the scene. All measurements are integrated into a hierarchical model (object resolution: 2 mm) and into a monolithic map at a resolution of 2 mm.

From the visualization of the resulting models in Fig. 6 it can be seen that the monolithic map is unable to fully adapt to the changes in the environment. Since all voxel occupancy probabilities are estimated independently, the model is not able to remove the object cells of the first object that are occluded by the second object. The use of submaps, as they are employed in our model, allows us to remove the corresponding map node completely after its existence probability falls below a threshold of 0.5 according to Eq. (8).

C. Object Exploration

To evaluate the exploration algorithm introduced in Sec. V, a simulated PR2 robot is used. The simulated environment contains a number of objects on a table (see Fig. 7 (a)).

An exemplary run of the system is depicted in Fig. 7 (b). After an initial scan, the table plane was detected and a series of view points v_i were chosen to autonomously model the objects on top of the table. The resulting model is visualized in Fig. 7 (c). The only remaining unmapped space within

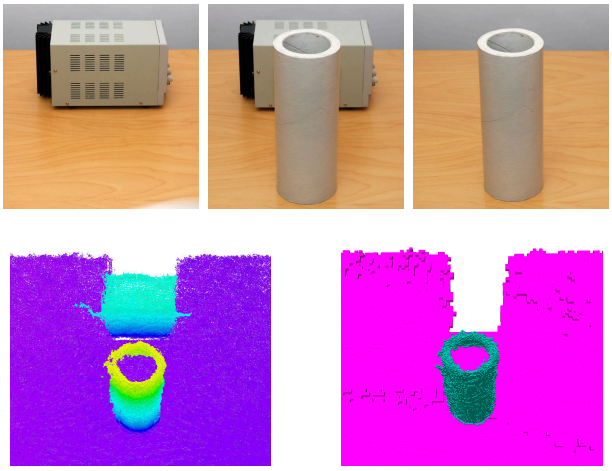


Fig. 6. Top row: scenes measured in experiment VI-B, from left to right. Bottom left: visualization of monolithic model at a resolution of 2 mm. Colors correspond to voxel height. Occluded object parts could not be removed from the model. Bottom right: visualization of hierarchical model with an object resolution of 2 mm. The object node is removed after its probability falls below 0.5.

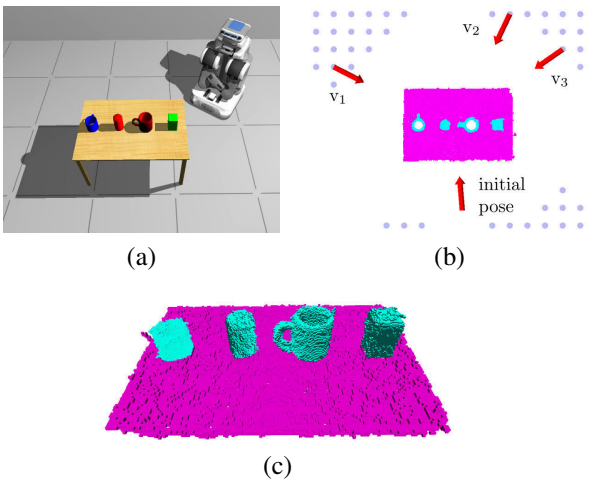


Fig. 7. (a) Simulated environment in exploration experiment. (b) View points generated during exploration (discs) and chosen (arrows). (c) Model generated during exploration. Objects are modeled at a resolution of 5 mm, the table top is modeled at 10 mm. Background voxels are omitted in the visualization for clarity.

the simulated cups (see top view) could not be measured by moving the robot around the table due to its sensor setup.

VII. Conclusion

In this paper, we presented an approach for efficiently modeling environments in 3D using a hierarchy of octree maps. Our model is compact and supports multiple resolutions. At the same time it has a probabilistic nature. In addition to the formal description of the system, we presented an implementation for tabletop manipulation tasks and an information-driven exploration algorithm that allows to learn the proposed model autonomously. We evaluated our approach using real world data as well as simulated data. The results demonstrate that our approach is able to efficiently represent 3D environments in a hierarchy of probabilistic maps at high levels of detail. Compared to a monolithic map, our approach is significantly faster and consumes significantly less memory.

References

[1] D. Anguelov, R. Biswas, D. Koller, B. Limketkai, S. Sanner, and S. Thrun. Learning hierarchical object maps of non-stationary environments with mobile robots. In *Proc. of the Conf. on Uncertainty in Artificial Intelligence (UAI)*, Edmonton, Canada, 2002.

[2] D.M. Cole and P.M. Newman. Using laser range data for 3D SLAM in outdoor environments. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2006.

[3] B. Douillard, J. Underwood, N. Melkumyan, S. Singh, S. Vasudevan, C. Brunner, and A. Quadros. Hybrid elevation maps: 3D surface models for segmentation. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2010.

[4] C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, JA Fernández-Madrigal, and J. González. Multi-hierarchical semantic maps for mobile robotics. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2005.

[5] S. Gottschalk. *Collision Queries using Oriented Bounding Boxes*. PhD thesis, The University of North Carolina, 2000.

[6] M. Habbecke and L. Kobbelt. A surface-growing approach to multi-view stereo reconstruction. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[7] M. Krainin, P. Henry, X. Ren, and D. Fox. Manipulator and object tracking for in hand model acquisition. In *Proc. of the Workshop on Best Practice in 3D Perception and Modeling for Mobile Manipulation at the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2010.

[8] D.J.C. MacKay. *Information theory, inference, and learning algorithms*. Cambridge Univ Press, 2003.

[9] D. Meagher. Geometric modeling using octree encoding. *Computer Graphics and Image Processing*, 19(2):129–147, 1982.

[10] H.P. Moravec and A.E. Elfes. High resolution maps from wide angle sonar. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 116–121, St. Louis, MO, USA, 1985.

[11] M. Müller, H. Surmann, K. Pervolz, and S. May. The accuracy of 6d slam using the ais 3d laser scanner. In *IEEE Int. Conf. on Multisensor Fusion and Integration for Intelligent Systems*, 2006.

[12] A. Nüchter and J. Hertzberg. Towards semantic maps for mobile robots. *Robotics & Autonomous Systems*, 56(11):915–926, 2008.

[13] A. Nüchter, K. Lingemann, J. Hertzberg, and H. Surmann. 6D SLAM—3D mapping outdoor environments: Research articles. *J. Field Robot.*, 24(8-9):699–722, 2007.

[14] A. Nüchter, O. Wulf, K. Lingemann, J. Hertzberg, B. Wagner, and H. Surmann. 3D mapping with semantic knowledge. *RoboCup 2005: Robot Soccer World Cup IX*, pages 335–346, 2006.

[15] P. Payeur, P. Hebert, D. Laurendeau, and C.M. Gosselin. Probabilistic octree modeling of a 3-d dynamic environment. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 1997.

[16] A. Petrovskaya and A.Y. Ng. Probabilistic mobile manipulation in dynamic environments, with application to opening doors. In *Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*, 2007.

[17] M. Ruhnke, B. Steder, G. Grisetti, and W Burgard. Unsupervised learning of 3d object models from partial views. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2009.

[18] R.B. Rusu, G. Bradski, R. Thibaux, and J. Hsu. Fast 3d recognition and pose using the viewpoint feature histogram. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2010.

[19] R.B. Rusu, Z.C. Marton, N. Blodow, M. Dolha, and M. Beetz. Towards 3D Point cloud based object maps for household environments. *Robotics & Autonomous Systems*, 56(11):927–941, 2008.

[20] C. Stachniss, G. Grisetti, and W. Burgard. Information gain-based exploration using rao-blackwellized particle filters. In *Proc. of Robotics: Science and Systems (RSS)*, pages 65–72, Cambridge, MA, USA, 2005.

[21] M. Strand and R. Dillmann. Using an attributed 2D-grid for next-best-view planning on 3D environment data for an autonomous robot. In *Int. Conf. on Information and Automation (ICIA)*, 2008.

[22] T. Weise, T. Wismer, B. Leibe, and L. Van Gool. In-hand scanning with online loop closure. In *ICCV Workshops*, 2009.

[23] K.M. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, and W. Burgard. Octomap: A probabilistic, flexible, and compact 3d map representation for robotic systems. In *ICRA 2010 Workshop on Best Practice in 3D Perception and Modeling for Mobile Manipulation*, Anchorage, USA, 2010.