Navigation in Combined Outdoor and Indoor Environments using Multi-Level Surface Maps

P. Pfaff* R. Kümmerle* D. Joho* C. Stachniss* R. Triebel+ W. Burgard*

*Department of Computer Science, University of Freiburg, 79110 Freiburg, Germany
+Autonomous Systems Lab, Swiss Federal Institute of Technology, 8092 Zurich, Switzerland

Abstract— **Whenever mobile robots are used in real world applications, the ability to learn an accurate model of the environment and to localize itself based on such a model are important prerequisites for reliable operation. Whereas these problems have been successfully solved in the past for most indoor tasks, in which the robot is assumed to operate on a flat surface, such approaches are likely to fail in combined indoor and outdoor environments in which the three-dimensional structure of the world needs to be considered. In this paper, we consider the problem of localizing a vehicle that operates in 3D indoor as well as outdoor settings. Our approach is entirely probabilistic and does not rely on GPS information. It is based on so-called multi-level surface maps which are an extension of the well-known elevation maps. In addition to that, we present a technique that allows the robot to actively explore the environment. This algorithm applies a decision-theoretic approach and considers the uncertainty in the model to determine the next action to be executed. In practical experiments, we illustrate the properties as well as advantages of our approach compared to other techniques.**

I. Introduction

Robots that are able to acquire an accurate model of their environment and to localize themselves based on such a model are regarded as fulfilling a major precondition of truly autonomous mobile vehicles.

The problem of mobile robot localization with range sensors in outdoor environments arises whenever GPS signals are missing due to occlusions caused by buildings, bridges, or trees. Furthermore, in case of combined outdoor and indoor environments, relying on GPS information will obviously lead to failure in the pose estimate. In such situations, a mobile robot typically has to estimate its position in the environment using its exteroceptive sensors and a map of the environment. However, when a robot attempts to perceive its environment to localize itself, the choice of the direction of the perception can substantially influence the accuracy of the position estimate. The localization task requires a given map of the environment. In case such a model is not available, it has to be learned by the robot. This problem is also known as autonomous exploration. So far, most approaches to mobile robot exploration assume that the robot lives in a plane. They typically focus on generating motion commands that minimize the time needed to cover the whole terrain [13], [24]. A frequently used technique is to build an occupancy grid map since it can model unknown locations efficiently. The robot seeks to reduce the number of unobserved cells or the uncertainty in the grid map. In the three-dimensional space, however, such approaches are not directly applicable. The size of occupancy grid maps in 3D, for example, prevents the robot from exploring an environment larger than a few hundred square meters.

The contribution of this paper are solutions to the localization and to the autonomous exploration problem in three-dimensional, combined outdoor and indoor environments. Both techniques use multi-level surface maps to provide an appropriate model of the environment. The MCL-based localization technique does not require GPS information and uses only proximity data from a laser range finder as well as odometry information. Our exploration technique extents existing exploration approaches used in 2D to the three-dimensional space. It selects actions that reduce the uncertainty of the robot about the world. It does so by reasoning about potential measurements that can be obtained when selecting an action. Our approach is able to deal with negative obstacles like, for example, abysms, which is a problem of robots exploring a three-dimensional world. Experiments carried out in simulation and on a real robot show the effectiveness of our techniques.

II. Related Work

The problem of localizing a mobile robot in indoor and outdoor environments with range sensors or cameras has been studied intensively in the past. In indoor environments, Monte-Carlo localization (MCL) [5] is one of the current state-of-the-art approaches. Outdoors, Adams *et al.* [1] extract predefined features from range scanners and apply a particle filter for localization. Davison and Kita [4] utilize a Kalman filter for vision-based localization with point features on non-flat surfaces. Recently, Agrawal and Konolige [2] presented an approach to robot localization in outdoor terrains based on feature points that are tracked across frames in stereo images. Lingemann *et al.* [15] recently described a method for fast localization in in- and outdoor environments. Their system operates on raw data sets, which results in huge memory requirements. Additionally, they apply a scan-matching routine for localization, which does not facilitate global localization. To reduce the memory requirements of outdoor terrain representations, several researchers applied elevation maps [3], [12], [14], [17]. A probabilistic approach to localize a planetary rover in such elevation maps has been described by Olson [16]. In this system, elevation maps were sufficient to robustly localize the vehicle, mainly because

the number of vertical and overhanging objects is negligible in environments like on Mars. However, environments on earth contain many objects like buildings or trees which have vertical or even overhanging surfaces. To address this issue, we use multi-level surface (MLS) maps [22] to represent the environment in this paper. MLS maps discretize the environment into cells and store for each cell a list of patches representing the individual layer in the environment as well as vertical structures.

So far, most approaches to mobile robot exploration assume that the robot lives in a plane. They typically focus on generating motion commands that minimize the time needed to cover the whole terrain [13], [24]. A frequently used technique is to build an occupancy grid map since it can model unknown locations efficiently. The robot seeks to reduce the number of unobserved cells or the uncertainty in the grid map [24], [18]. In the three-dimensional space, however, such approaches are not directly applicable. The size of occupancy grid maps in 3D, for example, prevents the robot from exploring an environment larger than a few hundred square meters.

Whaite and Ferrie [23] presented an exploration approach in 3D that uses the entropy to measure the uncertainty in the geometric structure of objects that are scanned with a laser range sensor. In contrast to the work described here, they use a fully parametric representation of the objects and the size of the object to model is bounded by the range of the manipulator. Surmann *et al.* [20] extract horizontal planes from a 3D point cloud and construct a polygon with detected lines (obstacles) and unseen lines (free space connecting detected lines). They sample candidate viewpoints within this polygon and use 2D ray-casting to estimate the expected information gain. In contrast to this, our approach uses an extension of 3D elevation maps and 3D ray-casting to select the next viewpoint. González-Baños and Latombe [9] also build a polygonal map by merging safe regions. Similar to our approach, they sample candidate poses in the visibility range of frontiers to unknown area. But unlike in our approach, they build 2D maps and do not consider the uncertainty reduction in the known parts of the map. Fournier *et al.* [8] present an 3D exploration approach utilizing an octree structure to represent the environment. However, it is unclear if the presented approach is able to explore on multiple levels.

The contribution of this paper are techniques for autonomously learning MLS maps with a mobile robot based on laser range finder and odometry only. We furthermore describe how a robot can utilize such a model to track its own pose and to globally localize itself. Our approach does not rely on GPS information and thus allows a robot to operate in combined indoor and outdoor scenarios.

III. 3D Model of the Environment

Our exploration system uses multi-level surface maps (MLS maps) as proposed by Triebel *et al.* [22]. MLS maps use a two-dimensional grid structure that stores different elevation values. In particular, they store in each cell of a discrete grid the height of the surface in the corresponding area. In contrast to elevation maps, MLS maps allow us to store multiple surfaces in each cell. Each surface is represented by a Gaussian with the mean elevation and its uncertainty σ. In the remainder of this paper, these surfaces are referred to as patches. This representation enables a mobile robot to model environments with structures like bridges, underpasses, buildings, or mines. They also enable the robot to represent vertical structures by storing a vertical depth value for each patch. Figure 1 shows two example maps from the same environment. The left image shows that it is not possible to represent an underpass , overhanging and vertical objects correctly using elevation maps. On the other hand the right image illustrates the ability of the MLS map approach to represent all these structures correctly.

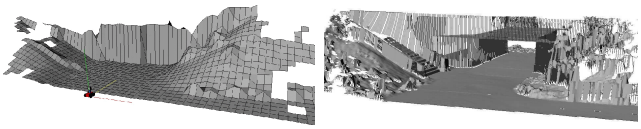

Fig. 1. Standard elevation map (left) which is not able to represent the underpass under the bridge correctly, and multi-level surface map (right) that correctly represents the height of the vertical objects and is able to model multiple levels.

IV. GPS-free Localization using MLS Maps

In this chapter, we assume that the robot already has a multi-level surface map available for localization. In the next chapter, we then present a technique to autonomously learn a MLS map.

To estimate the pose $\mathbf{x} = (x, y, z, \varphi, \vartheta, \psi)$ of the robot in its environment, we consider probabilistic localization, which follows the recursive Bayesian filtering scheme. The key idea of this approach is to maintain a probability density $p(\mathbf{x}_t \mid \mathbf{z}_{1:t}, \mathbf{u}_{0:t-1})$ of the robot's location $\mathbf{x}_t$ at time t given all observations $\mathbf{z}_{1:t}$ up to time t and all control inputs $\mathbf{u}_{0:t-1}$ up to time $t-1$. This posterior is updated as follows:

$$p(\mathbf{x}_t \mid \mathbf{z}_{1:t}, \mathbf{u}_{0:t-1}) = \alpha \cdot p(\mathbf{z}_t \mid \mathbf{x}_t) \cdot \int p(\mathbf{x}_t \mid \mathbf{u}_{t-1}, \mathbf{x}_{t-1}) \cdot p(\mathbf{x}_{t-1})\, d\mathbf{x}_{t-1}. \quad (1)$$

Here, α is a normalization constant ensuring that $p(\mathbf{x}_t \mid \mathbf{z}_{1:t}, \mathbf{u}_{0:t-1})$ sums up to one over all $\mathbf{x}_t$. The terms to be described in Eqn. (1) are the *prediction model* $p(\mathbf{x}_t \mid \mathbf{u}_{t-1}, \mathbf{x}_{t-1})$ and the *sensor model* $p(\mathbf{z}_t \mid \mathbf{x}_t)$. One major contribution of this paper is an appropriate computation of these models in the case that an MLS map is given.

For the implementation of the described filtering scheme, we use a sample-based approach which is commonly known as *Monte Carlo localization* [5]. Monte-Carlo localization is a variant of particle filtering [6] where each particle corresponds to a possible robot pose and has an assigned weight w_i. The *belief update* from Eqn. (1) is performed by the following two alternating steps:

1) In the **prediction step**, we draw for each particle with weight w_i a new particle according to w_i and to the prediction model $p(\mathbf{x}_t \mid \mathbf{u}_{t-1}, \mathbf{x}_{t-1})$.

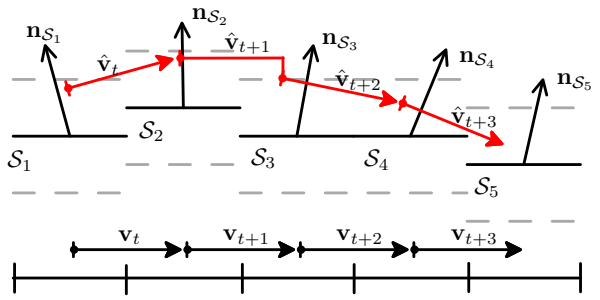

Fig. 2. Application of our prediction model to a series of 2D motion vectors (black). They are rotated to estimate the 3D motion vectors (red). The dashed line indicates the tolerance interval for the z-coordinate.

2) In the **correction step**, a new observation $\mathbf{z}_t$ is integrated. This is done by assigning a new weight w_i to each particle according to the sensor model $p(\mathbf{z}_t \mid \mathbf{x}_t)$.

A. Prediction Model for MLS Maps

The prediction model $p(\mathbf{x}_t \mid \mathbf{u}_{t-1}, \mathbf{x}_{t-1})$ we use is based on an approach introduced by Eliazar *et al.* [7]. It reflects systematic errors such as drift, as well as the uncertainty in the execution of an action $\mathbf{u} = (x_u, y_u, \theta_u)$, where (x_u, y_u) is the translation and θ_u the rotation angle. To incorporate this 2D motion into our 3D map we proceed as follows. First, we obtain a possible outcome (x_v, y_v, θ_v) of the action by applying the probabilistic model. Then, we adapt the motion vector $\mathbf{v} = (x_v, y_v)$ to the shape of the 3D surface traversed by the robot. This surface is obtained from the given MLS map and consists of planar square patches. To adapt the motion vector, we discretize it into segments of length c, which is the cell size of the MLS map, in our case 0.1 m. For each segment, we determine the corresponding surface patch $\mathcal{S}$ and rotate the segment according to the orientation (φ_S, ϑ_S) of the patch, where φ_S is the rotation about the x-axis and ϑ_S the rotation about the y-axis. The patch orientation is computed from the normal vector $\mathbf{n}_S$ of the patch $\mathcal{S}$, which in turn is obtained by fitting a plane into the local vicinity of $\mathcal{S}$. The normal vector computation is done beforehand and constitutes an extension to the framework of MLS maps. In general, it is not robust against noise and small errors in the MLS map, which results in an uncertainty of the patch orientation. In our approach, we model this uncertainty by adding Gaussian noise to the orientation parameters φ_S and ϑ_S. Thus, our prediction model expresses the uncertainty in 5 out of 6 position parameters – x, y and ψ by the 2D motion model and φ and ϑ by our 3D extension. For the last one – the height value z – we have the constraint that the robot must stay on the ground. Therefore, we adjust the z-value manually whenever it is too high or too low. This is illustrated in Figure 2. Finally, after concatenating all transformed motion vector segments, we obtain a new 3D motion vector $\hat{\mathbf{v}}$ which is added to the current estimate of the robot position $\mathbf{x}_{t-1}$ to obtain a new position estimate $\mathbf{x}_t$.

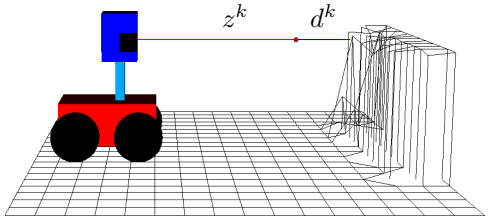

Fig. 3. Example of a single beam which ends close to vertical object in the MLS map. In the end point model, the probability $p_{hit}(z_k \mid \mathbf{x})$ only depends on the distance d^k between the end point of the k-th laser beam and the closest obstacle in the map.

B. Endpoint Sensor Model for MLS Maps

In our sensor model, we treat each beam independently and determine the likelihood of a whole laser scan by factorizing over all beams. Thus, we have

$$p(\mathbf{z} \mid \mathbf{x}) = \prod_{k=1}^{K} p(z^k \mid \mathbf{x}) \tag{2}$$

where K is the number of beams in each laser measurement $\mathbf{z}$. In Eqn. (2) and in the following, we drop the index t for convenience. Our sensor model $p(z^k \mid \mathbf{x})$ is based on an approach that has been introduced by Thrun [21] as *likelihood fields* (LF) or *end point model*. In particular, we formulate the sensor model $p(z^k \mid \mathbf{x})$ for each particular beam as a mixture of three different distributions:

$$\begin{aligned} p(z^k \mid \mathbf{x}) = \alpha_{hit} p_{hit}(z^k \mid \mathbf{x}) + \\ \alpha_{rand} p_{rand}(z^k \mid \mathbf{x}) + \alpha_{max} p_{max}(z^k \mid \mathbf{x}), \end{aligned} \tag{3}$$

where p_{hit} is a normal distribution $\mathcal{N}(0, \sigma^2)$ that models situations in which the sensor detects an obstacle. Random measurements are modeled using a uniform distribution $p_{rand}(z^k \mid \mathbf{x})$. Maximum range measurements are covered by a point mass distribution $p_{max}(z^k \mid \mathbf{x})$. These three distributions are weighted by the non-negative parameters α_{hit}, α_{rand}, and α_{max}, which sum up to one. The values for α_{hit}, α_{rand}, α_{max}, and σ^2 used in our current implementation have been determined empirically.

In the end point model, the probability $p_{hit}(z_k \mid \mathbf{x})$ only depends on the distance d^k between the end point of the k-th laser beam and the closest obstacle in the map. Figure 3 shows an example of a single beam z^k which ends close to vertical object in the MLS map. Thus, the physical property of the laser beam is ignored, because the model just uses the end point and does not consider the beam characteristic of the laser. Therefore, we need to calculate the global coordinates for a beam end point. If we denote the angle of the k-th beam relative to the zero angle with ζ^k, then the end point $\tilde{\mathbf{p}}^k = (\tilde{x}^k, \tilde{y}^k, \tilde{z}^k)^T$ of that beam in the robot's own coordinate frame is calculated as

$$\begin{pmatrix} \tilde{x}^k \\ \tilde{y}^k \\ \tilde{z}^k \end{pmatrix} = \begin{pmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \end{pmatrix} + R z^k \begin{pmatrix} \cos(\zeta^k) \\ \sin(\zeta^k) \\ 0 \end{pmatrix}, \tag{4}$$

where $(\hat{x}, \hat{y}, \hat{z})^T$ denotes the position of the sensor at time t and R is a rotation matrix that expresses the 3D sensor

orientation in the robot's coordinate frame. For a given robot pose $\mathbf{x} = (x, y, z, \varphi, \vartheta, \psi)$ at time t we can compute the global coordinates $\mathbf{p}^k = (x^k, y^k, z^k)^T$ of the k-th beam end point $\mathbf{p}^k$ as follows

$$\begin{pmatrix} x^k \\ y^k \\ z^k \end{pmatrix} = R(\varphi, \vartheta, \psi) \begin{pmatrix} \tilde{x}^k \\ \tilde{y}^k \\ \tilde{z}^k \end{pmatrix} + \begin{pmatrix} x \\ y \\ z \end{pmatrix}, \tag{5}$$

where $R(\varphi, \vartheta, \psi)$ denotes the rotation matrix for the given Euler angles φ, ϑ, and ψ. In MLS maps, obstacles are represented as *vertical surface patches*, which can be seen as vertical segments of occupied space. Unfortunately, there is no efficient way to find the closest of all vertical segments to a given beam end point. Therefore, we use an approximation by uniformly sampling a set $\mathcal{P}$ of 3D points from all vertical patches. The distance d^k of the k-th beam end point $\mathbf{p}^k$ to the closest obstacle is then approximated as the Euclidean distance $d(\mathbf{p}^k, \mathcal{P})$ between $\mathbf{p}^k$ and $\mathcal{P}$. This distance can be efficiently calculated by storing all points from $\mathcal{P}$ in a kD-tree.

Equations. (4) and (5) describe a 3D transform $T(z^k; \mathbf{x})$ of the measurement z^k at position $\mathbf{x}$. Using this and the fact that p_{hit} is Gaussian, we can compute p_{hit} as

$$p_{hit}(z^k \mid \mathbf{x}) \approx \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2}\left(\frac{d(\mathbf{p}^k, \mathcal{P})}{\sigma}\right)^2\right), \tag{6}$$

where $\mathbf{p}^k = T(z^k; \mathbf{x})$. Plugging this into Eqn. (3) and the result into Eqn. (2), we obtain the entire sensor model.

V. Autonomous Exploration in Three-dimensional Environments

The previous section covered the problem of localizing a vehicle in a MLS map. In this section, we relax the assumption that such a model is provided and present an approach to autonomously learn a MLS map with our mobile robot.

In order to autonomously explore the environment, we first need to perform a traversability analysis, thereby avoiding positive and negative obstacles. Then we determine candidate viewpoints in the vicinity of unexplored areas and evaluate those candidate viewpoints by considering the travel costs to a particular viewpoint and the expected information gain of a measurement at this viewpoint.

A. Traversability Analysis

A grid based 2D traversability analysis usually only takes into account the occupancy probability of a grid cell – implicitly assuming an even environment with only positive obstacles. In the 3D case, especially in outdoor environments, we additionally have to take into account the slope and the roughness of the terrain, as well as negative obstacles such as abysms which are usually ignored in 2D representations.

Each patch p will be assigned a traversability value $\tau(p) \in [0, 1]$. A value of zero corresponds to a non-traversable patch, a value greater zero to a traversable patch, and a value of one to a perfectly traversable patch. In order to determine $\tau(p)$, we fit a plane into its local 8-patch neighborhood by minimizing the z-distance of the plane to the elevation values of the neighboring patches. We then compute the slope and the roughness of the local terrain and detect obstacles. The slope is defined as the angle between the fitted plane and a horizontal plane and the roughness is computed as the average squared z-distances of the height values of the neighboring patch to the fitted plane. The slope and the roughness are turned into traversability values $\tau_s(p)$ and $\tau_r(p)$ by linear interpolation between zero and a maximum slope and roughness value respectively. In order to detect obstacles we set $\tau_o(p) \in \{0, 1\}$ to zero, if the maximum squared z-distance of a neighboring patch exceeds a threshold, thereby accounting for positive and negative obstacles, or if the patch has less than eight neighbors. The latter is important for avoiding abysms in the early stage of an exploration process, as some neighboring patches are below the edge of the abysm and therefore are not visible yet.

The combined traversability value is defined as $\tau(p) = \tau_s(p) \cdot \tau_r(p) \cdot \tau_o(p)$. Next, we iteratively propagate the values by convolving the traversability values of the patch and its eight neighboring patches with a Gaussian kernel. For non-existent neighbors, we assume a value of 0.5. The number of iterations depends on the used cell size, the robot's size and a safety margin. In order to enforce obstacle growing, we do not perform a convolution if one of the neighboring patches is non-traversable ($\tau = 0$), but rather set the patch's traversability directly to zero in this case.

B. Viewpoint Generation

We follow the popular frontier-based approach to exploration [24] and adapt it to the needs of a 3D environment. In our approach, a patch is considered as explored if it has eight neighbors and its uncertainty, measured by the entropy in the patch, is below a threshold. Additionally, we track the entropy as well as the number of neighbors of a patch. If the entropy or number of non-existing neighbors cannot be reduced as expected over several observations, we consider it to be explored nonetheless since further observations do not seem to change the state of the patch.

A frontier patch is defined as an unexplored patch with at least one explored neighboring patch. Most of these patches have less than eight neighbors and therefore are considered as non-traversable, since they might be at the edge of an abysm. Therefore, we cannot drive directly to a frontier patch. Instead, we use a 3D ray-casting technique to determine close-by candidate viewpoints. A patch is considered as a candidate viewpoint, if it is reachable and there is at least one frontier patch that is likely to be observable from that viewpoint. Instead of using ray-casting to track emitted beams from the sensor at every reachable position, we use a more efficient approach. We emit virtual beams from the frontier patch instead and then select admissible sensor locations along those beams (Figure 4). This will reduce the number of needed ray-casting operations as the number of frontier patches is much smaller than the number of reachable patches.

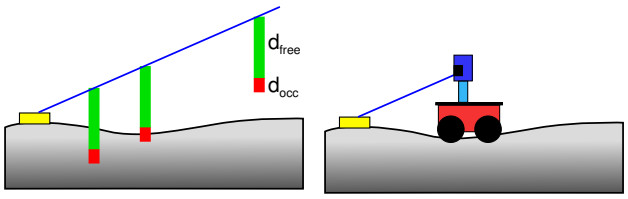

Fig. 4. To generate viewpoints, we emit laser beams from viewpoints and determine admissible sensor positions along those beams. The interval d_{free} needs to be free and the interval d_{occ} has to contain a reachable patch.

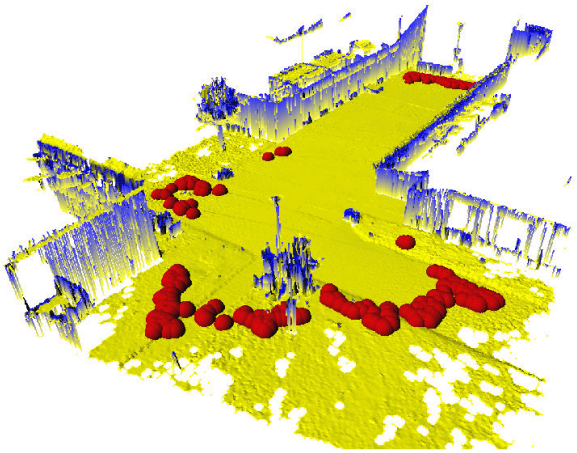

Fig. 5. Outdoor map showing sampled candidate viewpoints as red (dark gray) spheres.

In practice, we found it useful to reject candidate viewpoints, from which the expected information gain is below a threshold. We also cluster the frontier patches by the neighboring relation, and prevent patches from very small frontier clusters to generate candidate viewpoints. This will lead to a more reliable termination of the exploration process. Candidate viewpoints of an example map are shown in Figure 5.

C. Viewpoint Evaluation

The utility $u(v)$ of a candidate viewpoint v, is computed using the expected information gain $E\{I(v)\}$ and the travel costs $t(v)$. As the evaluation involves a costly 3D ray-casting operation, we reduce the set of candidate viewpoints by sampling uniformly a fixed number of viewpoints from that set.

In order to simultaneously determine the shortest paths to all candidate viewpoints, we use a deterministic variant of the value iteration. The costs of moving from a patch p to p' can be defined as

$$c(p, p') = d(p, p') + w(1 - \tau(p')) \tag{7}$$

where $d(p, p')$ describes the Euclidian distance and $\tau(p')$ the traversability of p'. The constant w is used to weight the penalization for traversing poorly traversable patches. The travel costs $t(v)$ of a viewpoint v is defined as the accumulated step costs of the shortest path to that viewpoint.

The expected information gain considers the uncertainty reduction in the known parts of the map as well as the information gain caused by new patches that are expected to be discovered.

To determine the patches that are likely to be hit by a laser measurement, we first perform a ray-cast operation similar to [19]. We determine the intersection points of the cell boundaries and the 3D ray projected onto the 2D grid. In a second step, we determine for each cell the height interval covered by the ray and check for collisions with patches contained in that cell by considering their elevation and depth values.

Let the sequence $L = \langle l_1, \ldots, l_m \rangle$ be an equidistant discretization of the maximum laser range. If the simulated laser ray hits a patch in distance that falls into l_h, we can divide L into three subsequences L^f, L^h, and L^n, whereas $L^f = \langle l_1, \ldots, l_{h-1} \rangle$ contains the collision free traversed distances, $L^h = \langle l_h \rangle$ contains the above mentioned discretized distance to the patch that has been hit, and $L^n = \langle l_{h+1}, \ldots, l_m \rangle$ contains the non-traversed distances. Accordingly, if the simulated ray does not hit a patch, this will result in three subsequences $L^f = L$ and $L^h = L^n = \langle\rangle$. For each traversed distance $l \in L^f \cup L^h$ we expect the ray during a real measurement to end after distance l with probability $p(l)$. If $l \in L^f$, then this corresponds to the discovery of a new patch, which implies an information gain $I_f(l)$. If $l \in L^h$, then this corresponds to a measurement of an already known patch, which implies an information gain $I_h(l)$. The expected information gain of ray r then is defined as

$$E\{I(r)\} = \sum_{l \in L} p(l) I(l) = \sum_{l \in L^f} p(l) I_f(l) + \sum_{l \in L^h} p(l) I_h(l). \tag{8}$$

Here we assume $p(l) = 0$ for $l \in L^n$, as we do not expect the ray to travel through a known patch.

To assess the probabilities $p(l)$, we created statistics through simulated measurements in a large outdoor map which yielded a conditional probability distribution $p_s(d \mid \alpha_v)$ denoting the probability of hitting an obstacle after distance d when the elevation angle of the ray is α_v. The intuition behind this is, that it is much more likely for downward pointing rays to hit a patch than for upward pointing rays. Secondly, the probability to hit an obstacle is not equally distributed along the laser range, especially not for downward pointing rays. Using this distribution, we can define

$$p(l) = \begin{cases} p_s(l \mid \alpha_v) & l \in L^f \\ \sum_{l_i \in L^h \cup L^n} p_s(l_i \mid \alpha_v) & l \in L^h \\ 0 & l \in L^n \end{cases} \tag{9}$$

with α_v being the elevation angle of the current ray r.

The information gain I_h is defined by the uncertainty reduction in the known map. We therefore temporary add a new measurement m_h into the grid cell of the hit patch p_h with a corresponding mean and variance that depends on the distance l_h of the simulated ray. The mean and variance of

the patch p_h will then be updated by using a Kalman filter. As a patch is represented as a Gaussian, we can compute the entropy $H(p)$ of a patch as

$$H(p) = \frac{1}{2} \log \left(2e\pi\sigma^2\right). \quad (10)$$

The information gain $I_h(l)$ is then defined as the difference

$$I_h(l) = H(p_h) - H(p_h \mid m_h) \quad l \in L^h. \quad (11)$$

between the entropy $H(p_h)$ of the patch p_h before and the entropy $H(p_h \mid m_h)$ after the temporary incorporation of the simulated measurement m_h.

For the information gain I_f we will proceed similarly. As a newly discovered patch p_f will be inserted with an uncertainty σ proportional to the distance $l \in L^f$ of measurement m_f, we can thereby compute $H(p_f \mid m_f)$ as in Eqn. 10. We assume that the uncertainty σ_b of the patch before it has been measured, is bounded by the distance d_p to the nearest patch in that cell and choose, as a heuristic, an uncertainty so that $3\sigma_b = d_p$. Using σ_b we can define $H(p_f)$ and finally compute

$$I_f(l) = H(p_f) - H(p_f \mid m_f) \quad l \in L^f. \quad (12)$$

The expected information gain $E\{I(v)\}$ of a viewpoint v is then defined as the sum $E\{I(v)\} = \sum_{r \in R} E\{I(r)\}$ of the expected information gains of all casted rays $r \in R$.

Finally, the utility $u(v)$ of each candidate viewpoint is computed by a relative expected information gain and travel costs as

$$u(v) = \alpha \frac{E\{I(v)\}}{\max_x E\{I(x)\}} + (1-\alpha) \frac{\max_x t(x) - t(v)}{\max_x t(x)}. \quad (13)$$

By varying the constant $\alpha \in [0,1]$ one can alter the exploration behavior by trading off the travel costs and the expected information gain.

D. Overlap

As explained before, we choose the viewpoint with the best utility as the next goal point. However, to ensure that we can construct a globally consistent map, we have to continously track the position of the vehicle. We construct a network of constraints between poses according to the observations. We then apply an efficient global optimization approach [10], [11] to correct the poses.

To ensure that the relations between poses can be accurately determined, a certain overlap between consecutive 3D scans is required. We perform several 3D scans along the way to ensure this sufficient overlap. We use the 3D ray-casting technique to simulate a 3D scan and estimate the overlap of a real scan at each patch p_i of the planned path $\langle p_1, \ldots, p_n \rangle$. The estimated overlap $\hat{o}(p_i) = r_l/|R|$ is ratio of the number of rays r_l that hit a patch of the last local map to the number of all casted rays $|R|$ for a simulated scan at patch p_i. The patch p_i with the highest index $i \in \{1, \ldots, n\}$ whose overlap $\hat{o}(p_i)$ is above a threshold is chosen as the subgoal for the next 3D scan.

Based on the map estimate so far, we apply the localization approach described in the previous chapter. Based on the most likely pose reported by the localization module, we perform scan-matching to refine the estimate. The relation between poses that are determined in this way are then added to the constraint network. The exploration ends, if the set of candidate viewpoints is empty.

VI. Experiments

In this section, we present experiments designed to illustrate the properties of the presented techniques as well as their advantages compared to other techniques. First, we present experiments that evaluate the GPS-free localization approach using laser range finder only. Then, we investigate the properties of our uncertainty-driven exploration approach.

A. Localization

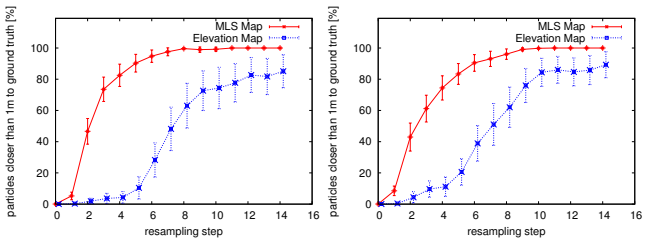

Fig. 6. Convergence of the particles to the true position of the robot with 500,000 (left) and 1,000,000 (right) particles. The x-axes depict the number of resampling steps, while the y-axes show the percentage of particles that are closer than $1m$ to the true position.

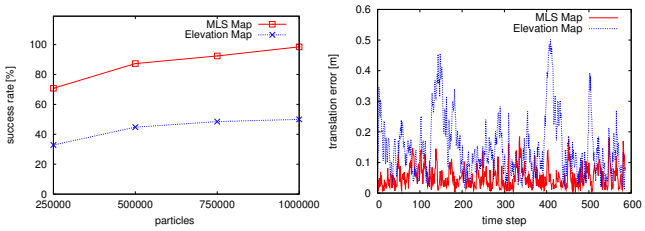

Fig. 7. The left image depicts the number of successful localizations after 15 resampling steps for the two different map representations for particle numbers from 250,000 up to 1,000,000. The right image shows the average localization error over all particles for a tracking experiment with 1,000 particles. In average the use of the MLS maps leads to smaller errors.

The first set of experiments is designed to evaluate the performance of the MLS map approach in the context of a global localization task. Figure 6 depicts the convergence of the particles to the true position of the robot with $500,000$ and $1,000,000$ particles. Whereas the x-axis corresponds to the resampling step, the y-axis shows the number of particles in percent that are closer than $1m$ to the true position, which has been computed by a tracking experiment with $100,000$ particles. Shown are the evolutions of these numbers when the MCL is applied on standard elevation maps and on MLS maps. Note that the elevation map does not reach 100%. This is due to the fact that the sensor model for the standard elevation map relies on a highly smoothed likelihood function, which is good for global localization but does not achieve maximal accuracy during tracking. The application

of a more peaked sensor model in the case of the standard elevation map would lead to much higher divergence rates. In both cases, a t-test showed that it is significantly better to apply the MLS maps than the standard elevation maps for the global localization task. Experiments with 250,000 and 750,000 particles showed the same behavior. The left image of Figure 7 shows the number of successful localizations for the two different map representations and for different numbers of particles. Here, we assumed that the localization was achieved when every particle differed by at most $1m$ from the true location of the robot. We can see that the global localization performs more robust on the MLS map than on the standard elevation map.

Fig. 8. MLS map used for the localization experiments. The area represented by this map spans approximately 195 by 146 meters. The blue / dark gray line shows the localized robot poses. The yellow / light gray line shows the pure odometry. The traversed trajectory has a length of 284 meters.

As a second set of experiments we carried out experiments, in which we analyzed the accuracy of the MLS map approach in the context of a position tracking task. To obtain the corresponding data set, we steered along a loop in our campus environment. The traversed trajectory has a length of 284 meters. Figure 8 depicts a top view of the MLS map of our test environment. The blue / dark gray line shows the localized robot poses. The yellow / light gray line shows the pure odometry. The right image of Figure 7 depicts the average localization error for a tracking experiment with 1,000 particles. As can be seen from the figure, the MLS map

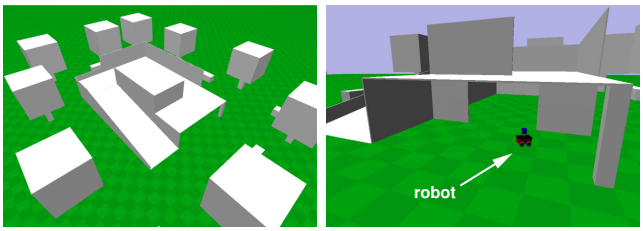

Fig. 9. Overview of the simulation environment and a detailed view of the entrance on the first floor with the robot in front of it.

approach outperforms the standard elevation map approach. The tracking experiments have been computed online on a standard PC with an AMD Athlon 64 3200+ processor. In the practical experiments we found that the use of the MLS maps results in a computational overhead of no more than 10% compared to elevation maps.

B. Exploration

The first exploration experiment is designed to show the ability of our exploration technique to take full advantage of the capabilities that MLS maps provide, e.g. representing multiple surface layers on top of each other. In a simulation environment with realistic rigid body physics we constructed a two-story building (Figure 9). It consists of two rooms located on top of each other, each 12 by 8 meters in size, and an unsecured balcony, where the robot is initially located. The house is surrounded by some trees and bushes, which are approximated by cuboids. We restricted the location of possible viewpoints to a rectangular area around the house in order to focus on the exploration of the house rather than the free space around the house. The robot explored the balcony, traversed the upper room and proceeded down a ramp that connects the upper room with the ground floor. The robot drove around the house and then entered the entrance to the room in the first floor. During the exploration of the lower room several 3D loops with positions at the upper room have been closed. He then visited a last viewpoint at the back of the house and then the exploration ended. The robot visited 18 viewpoints, performed 29 3D scans and traveled a distance of 212 meters. The final map consists of 185,000 patches. We demonstrated with this experiment, that we are able to deal with several challenges that simple mapping approaches are not able to deal with, e.g. negative obstacles and multiple surface layers. A 2D approach would simply have fallen down the unsecured balcony, and simple 3D mapping approaches like, for example, elevation maps, would not support the exploration of the two storys on top of each other. Figure 10 shows the constructed map with a detailed view of the entrance to the lower room.

To demonstrate the ability to explore real environments, we performed an experiment on the campus of the University of Freiburg using an ActivMedia Pioneer 2-AT equipped with a SICK laser range scanner mounted on a pan-tilt unit. To give the exploration an initial direction, we restricted the generation of viewpoints to the half-plane in front of the initial location of the robot. The robot followed a path

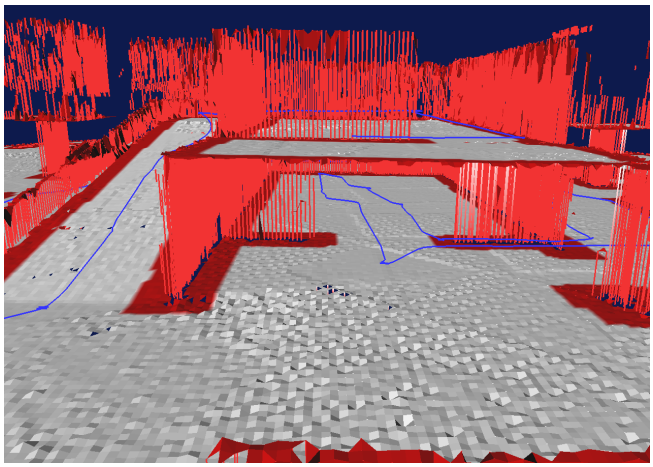

Fig. 10. Detailed view of the final traversability map showing the robot's trajectory as a blue line.

bordered by the wall of a house on the left side and grassland on the right side (Figure 11). Then he entered a small courtyard on the left, which was sufficiently explored after a few scans. He then proceeded to explore the rest of the campus until he reached the border of the defined half-plane. The figure shows four snapshots of the exploration process. In the last image, the robot traveled 186 meters, visited 18 viewpoints and performed 26 3D scans. The corresponding map including the traversability information contains about 410,000 patches and is depicted in Figure 12. In both experiments, we set $\alpha = 0.5$ in order to equally consider the travel costs and expected information gain.

VII. Conclusion

In this paper, we considered the problem of autonomously learning a three-dimensional model for combined outdoor and indoor environments with a mobile robot. We furthermore demonstrated how to localize a mobile vehicle based on such a model without requiring GPS information. Our approach uses proximity data from a laser range finder as well as odometry. Using our three-dimensional model of the environment, namely multi-level surface maps, we obtain significantly better results compared to elevation maps. We also presented an algorithm to actively acquire such maps from an unknown environment. This approach is decision-theoretic and trades off the cost of carrying out an action with the expected information gain of future observations. The approach also considers negative obstacles such as absyms which is an important prerequisite for robots operating in 3D environments.

Acknowledgment

This work has partly been supported by the DFG within the Research Training Group 1103 and under contract number SFB/TR-8, by the EC under contract number FP6-IST-34120-muFly, action line: 2.5.2.: micro/nano based subsystems, and under contract number FP6-004250-CoSy.

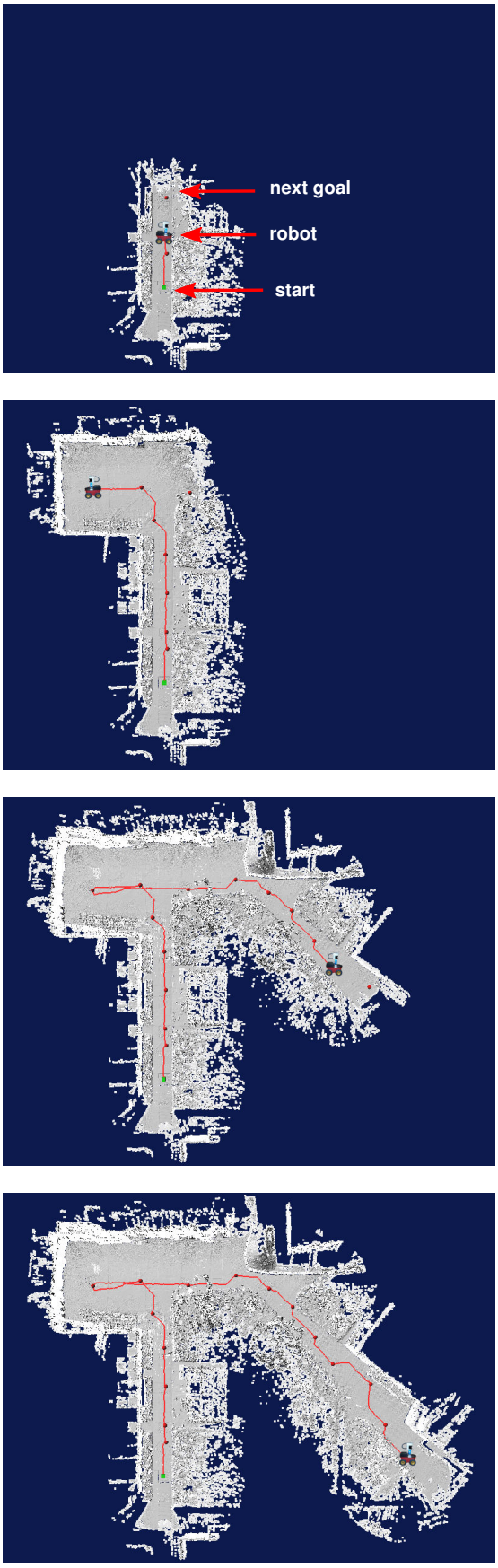

Fig. 11. Course of the exploration process on the university campus.

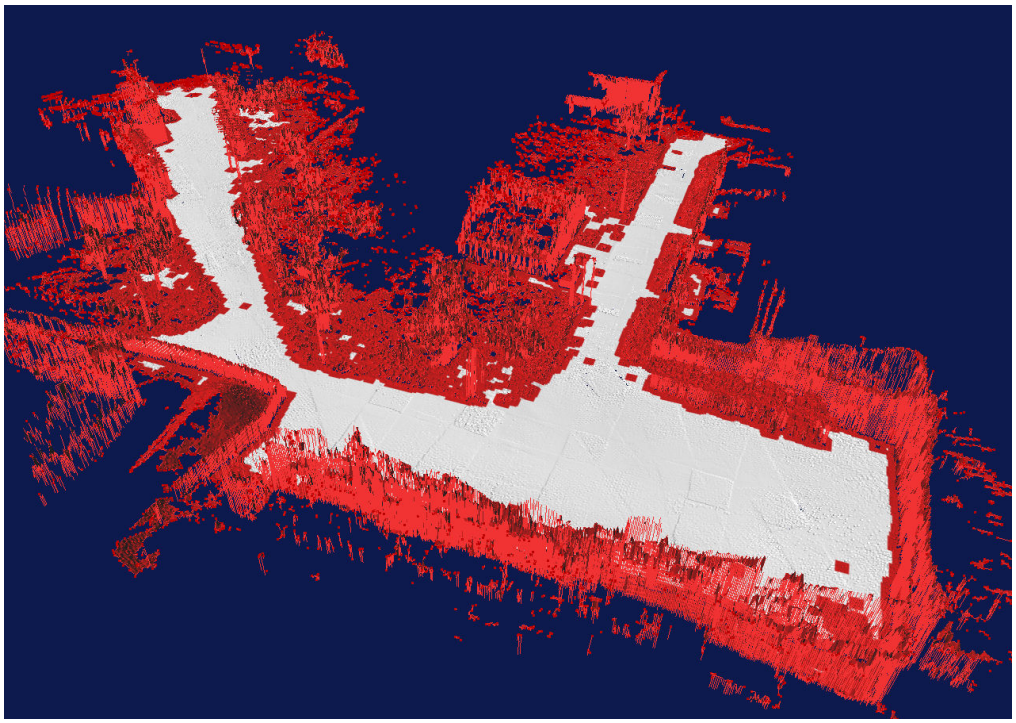

Fig. 12. Traversability map of the university campus.

REFERENCES

[1] M. Adams, S. Zhang, and L. Xie. Particle filter based outdoor robot localization using natural features extracted from laser scanners. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2004.

[2] M. Agrawal and K. Konolige. Real-time localization in outdoor environments using stereo vision and inexpensive gps. In *International Conference on Pattern Recognition (ICPR)*, 2006.

[3] J. Bares, M. Hebert, T. Kanade, E. Krotkov, T. Mitchell, R. Simmons, and W. R. L. Whittaker. Ambler: An autonomous rover for planetary exploration. *IEEE Computer Society Press*, 22(6):18–22, 1989.

[4] A Davison and N. Kita. 3d simultaneous localisation and map-building using active vision for a robot moving on undulating terrain. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition, Kauai*. IEEE Computer Society Press, December 2001.

[5] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte carlo localization for mobile robots. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, Leuven, Belgium, 1998.

[6] A. Doucet, N. de Freitas, and N. Gordan, editors. *Sequential Monte-Carlo Methods in Practice*. Springer Verlag, 2001.

[7] A. Eliazar and R. Parr. Learning probabilistic motion models for mobile robots. In *Proc. of the International Conference on Machine Learning (ICML)*, 2004.

[8] Jonathan Fournier, Benoit Ricard, and Denis Laurendeau. Mapping and exploration of complex environments using persistent 3D model. In *Proc. of the Canadian Conf. on Computer and Robot Vision (CRV)*, pages 403–410, Montreal, Canada, 2007.

[9] H.H. González-Baños and J.-C. Latombe. Navigation strategies for exploring indoor environments. *Int. Journal of Robotics Research*, 21(10-11):829–848, 2002.

[10] G. Grisetti, S. Grzonka, C. Stachniss, P. Pfaff, and W. Burgard. Efficient estimation of accurate maximum likelihood maps in 3d. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, San Diego, CA, USA, 2007. To appear.

[11] G. Grisetti, C. Stachniss, S. Grzonka, and W. Burgard. A tree parameterization for efficiently computing maximum likelihood maps using gradient descent. In *Proc. of Robotics: Science and Systems (RSS)*, Atlanta, GA, USA, 2007.

[12] M. Hebert, C. Caillas, E. Krotkov, I.S. Kweon, and T. Kanade. Terrain mapping for a roving planetary explorer. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 997–1002, 1989.

[13] S. Koenig and C. Tovey. Improved analysis of greedy mapping. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Las Vegas, NV, USA, 2003.

[14] S. Lacroix, A. Mallet, D. Bonnafous, G. Bauzil, S. Fleury and; M. Herrb, and R. Chatila. Autonomous rover navigation on unknown terrains: Functions and integration. *Int. Journal of Robotics Research*, 21(10-11):917–942, 2002.

[15] K. Lingemann, H. Surmann, A. Nüchter, and J. Hertzberg. High-speed laser localization for mobile robots. *Journal of Robotics & Autonomous Systems*, 51(4):275–296, 2005.

[16] C.F. Olson. Probabilistic self-localization for mobile robots. *IEEE Transactions on Robotics and Automation*, 16(1):55–66, 2000.

[17] C. Parra, R. Murrieta-Cid, M. Devy, and M. Briot. 3-d modelling and robot localization from visual and range data in natural scenes. In *1st International Conference on Computer Vision Systems (ICVS)*, number 1542 in LNCS, pages 450–468, 1999.

[18] C. Stachniss and W. Burgard. Exploring unknown environments with mobile robots using coverage maps. In *Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*, pages 1127–1132, Acapulco, Mexico, 2003.

[19] 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.

[20] H. Surmann, A. Nüchter, and J. Hertzberg. An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments. *Journal of Robotics & Autonomous Systems*, 45(3-4):181–198, 2003.

[21] S. Thrun. A probabilistic online mapping algorithm for teams of mobile robots. *International Journal of Robotics Research*, 20(5):335–363, 2001.

[22] R. Triebel, P. Pfaff, and W. Burgard. Multi-level surface maps for outdoor terrain mapping and loop closing. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2006.

[23] P. Whaite and F. P. Ferrie. Autonomous exploration: Driven by uncertainty. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(3):193–205, 1997.

[24] B. Yamauchi. Frontier-based exploration using multiple robots. In *Proc. of the Second International Conference on Autonomous Agents*, pages 47–53, Minneapolis, MN, USA, 1998.