An Efficient Extension to Elevation Maps for Outdoor Terrain Mapping and Loop Closing

Patrick Pfaff Rudolph Triebel Wolfram Burgard

Department of Computer Science, University of Freiburg, 79110 Freiburg, Germany
{*pfaff,triebel,burgard*}*@informatik.uni-freiburg.de*

Abstract

Elevation maps are a popular data structure for representing the environment of a mobile robot operating outdoors or on not-flat surfaces. Elevation maps store in each cell of a discrete grid the height of the surface at the corresponding place in the environment. However, the use of this $2\frac{1}{2}$-dimensional representation, is disadvantageous when utilized for mapping with mobile robots operating on the ground, since vertical or overhanging objects cannot be represented appropriately. Furthermore, such objects can lead to registration errors when two elevation maps have to be matched. In this paper, we propose an approach that allows a mobile robot to deal with vertical and overhanging objects in elevation maps. Our approach classifies the points in the environment according to whether they correspond to such objects or not. We also present a variant of the ICP algorithm that utilizes the classification of cells during the data association. Additionally, we describe how the constraints computed by the ICP algorithm can be applied to determine globally consistent alignments. Experiments carried out with a real robot in an outdoor environment demonstrate that our approach yields highly accurate elevation maps even in the case of loops. We furthermore present experimental results demonstrating that our classification increases the robustness of the scan matching process.

1 Introduction

The problem of learning maps with mobile robots has been intensively studied in the past. In the literature, different techniques for representing the environment of a mobile robot prevail. Topological maps aim at representing environments by graph-like structures, where nodes correspond to places, and edges to paths between them. Geometric models, in contrast, use either grid-based approximations or geometric primitives for representing the environment. Whereas topological maps have the advantage to better scale to large environments, they lack the ability to represent the geometric structure of the environment. The latter, however, is essential in situations, in which robots are deployed in unstructured outdoor environments where the ability to traverse specific areas of interest needs to be known accurately. However, full three-dimensional mod-

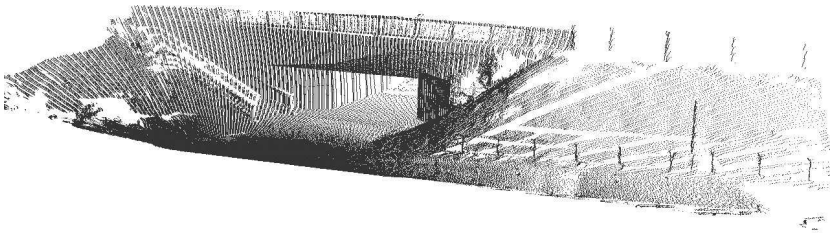

Figure 1: Scan (point set) of a bridge recorded with a mobile robot carrying a SICK LMS laser range finder mounted on a pan/tilt unit.

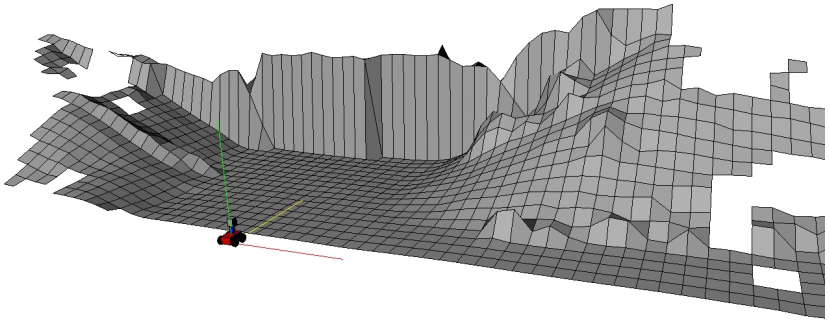

Figure 2: Standard elevation map computed for the outdoor environment depicted in Figure 1. The passage under the bridge has been converted into a large un-traversable object.

els typically have too high computational demands for a direct application on a mobile robot.

Elevation maps have been introduced as a more compact, $2\frac{1}{2}$-dimensional representation. An elevation map consists of a two-dimensional grid in which each cell stores the height of the territory. This approach, however, can be problematic when a robot has to utilize these maps for navigation or when it has to register two different maps in order to integrate them. For example, consider the three-dimensional point cloud shown in Figure 1, which has been acquired with a mobile robot located in front of a bridge. The resulting elevation map, which is computed from averaging over all scan points that fall into a cell of a horizontal grid (given a vertical projection), is depicted in Figure 2. As can be seen from the figure, the underpass has completely disappeared and the elevation map shows a non-traversable object. Additionally, when the environment contains vertical structures, we typically obtain varying average height values depending on how much of this vertical structure is contained in a scan. When two such elevation maps need to be aligned, this can lead to large registration errors.

In this paper, we present a system for mapping outdoor environments with ele-

vation maps. Our approach classifies locations in the environment into four classes, namely locations sensed from above, vertical structures, vertical gaps, and traversable cells. The advantage of this classification is twofold. First, the robot can represent obstacles corresponding to vertical structures like walls of buildings. It also can deal with overhanging structures like branches of trees or bridges. Second, the classification can be applied to achieve a more robust data association in the ICP algorithm [4]. In this paper we also describe how to apply a constraint-based robot pose estimation technique [33], similar to the one presented by Lu & Milios [17], to calculate globally consistent alignments of the local elevation maps. We present experimental results illustrating the advantages of our approach regarding the representation aspect as well as regarding the robust matching in urban outdoor environments also containing loops.

This paper is organized as follows. After discussing related work in the following section, we will present our extension to the elevation maps in Section 3. In Section 4 we then describe how to incorporate our classification into the ICP algorithm used for matching elevation maps. Section 5 will introduce our constraint-based pose estimation procedure for calculating consistent maps. Finally, we present experimental results in Section 6.

2 Related Work

The problem of learning three-dimensional representations has been studied intensively in the past. Recently, several techniques for acquiring three-dimensional data with 2d range scanners installed on a mobile robot have been developed. A popular approach is to use multiple scanners that point towards different directions [29, 9, 30]. An alternative is to use pan/tilt devices that sweep the range scanner in an oscillating way [25, 20]. More recently, techniques for rotating 2d range scanners have been developed [13, 36].

Many authors have studied the acquisition of three-dimensional maps from vehicles that are assumed to operate on a flat surface. For example, Thrun *et al.* [29] present an approach that employs two 2d range scanners for constructing volumetric maps. Whereas the first is oriented horizontally and is used for localization, the second points towards the ceiling and is applied for acquiring 3d point clouds. Früh and Zakhor [7] apply a similar idea to the problem of learning large-scale models of outdoor environments. Their approach combines laser, vision, and aerial images. Furthermore, several authors have considered the problem of simultaneous mapping and localization (SLAM) in an outdoor environment [6, 8, 31]. These techniques extract landmarks from range data and calculate the map as well as the pose of the vehicles based on these landmarks. Our approach described in this paper does not rely on the assumption that the surface is flat. It uses elevation maps to capture the three-dimensional structure of the environment and is able to estimate the pose of the robot in all six degrees of freedom.

One of the most popular representations are raw data points or triangle meshes [1, 15, 25, 32]. Whereas these models are highly accurate and can easily be textured, their disadvantage lies in the huge memory requirement, which grows linearly in the number of scans taken. Accordingly, several authors have studied techniques for simplifying point clouds by piecewise linear approximations. For example, Hähnel *et al.* [9] use

a region growing technique to identify planes. Liu *et al.* [16] as well as Martin and Thrun [18] apply the EM algorithm to cluster range scans into planes. Recently, Triebel *et al.* [34] proposed a hierarchical version that takes into account the parallelism of the planes during the clustering procedure. An alternative is to use three-dimensional grids [21] or tree-based representations [27], which only grow linearly in the size of the environment. Still, the memory requirements for such maps in outdoor environments are high.

In order to avoid the complexity of full three-dimensional maps, several researchers have considered elevation maps as an attractive alternative. The key idea underlying elevation maps is to store the $2\frac{1}{2}$-dimensional height information of the terrain in a two-dimensional grid. Bares *et al.* [3] as well as Hebert *et al.* [10] use elevation maps to represent the environment of a legged robot. They extract points with high surface curvatures and match these features to align maps constructed from consecutive range scans. Parra *et al.* [24] represent the ground floor by elevation maps and use stereo vision to detect and track objects on the floor. Singh and Kelly [28] extract elevation maps from laser range data and use these maps for navigating an all-terrain vehicle. Ye and Borenstein [37] propose an algorithm to acquire elevation maps with a moving vehicle carrying a tilted laser range scanner. They propose special filtering algorithms to eliminate measurement errors or noise resulting from the scanner and the motions of the vehicle. Lacroix *et al.* [14] extract elevation maps from stereo images. Hygounenc *et al.* [12] construct elevation maps with an autonomous blimp using 3d stereo vision. They propose an algorithm to track landmarks and to match local elevation maps using these landmarks. Olson [23] describes a probabilistic localization algorithm for a planetary rover that uses elevation maps for terrain modeling. Wellington *et al.* [35] construct a representation based on Markov Random Fields. They propose an environment classification for agricultural applications. They compute the elevation of the cell depending on the classification of the cell and its neighbors. Compared to these techniques, the contribution of this paper lies in two aspects. First, we classify the points in the elevation map into horizontal points seen from above, vertical points, and gaps. This classification is important especially when a rover is deployed in an urban environment. In such environments, typical structures like the walls of buildings cannot be represented in standard elevation maps. Second, we describe how this classification can be used to improve the matching of different elevation maps.

Recently, several authors have studied the problem of simultaneous localization and mapping by taking into account the six degrees of freedom of a mobile robot operating on a non-flat surface. For example, Davison *et al.* [5] presented an approach to vision based SLAM with a single camera moving freely through the environment. This approach uses an extended Kalman Filter to simultaneously update the pose of the camera and the 3d feature points extracted from the camera images. More recently, Nüchter *et al.* [22] developed a mobile robot that builds accurate three-dimensional models. In this approach, loop closing is achieved by uniformly distributing the estimated odometry error over the poses in a loop. In contrast, the work described here employs elevation maps to obtain a more compact representation of the 3d data. Our approach also includes a technique to globally optimize the pose estimates for calculating consistent maps. The loop-closing technique is also an extension to our previous work [26] in which the ICP algorithm was used for incremental mapping.

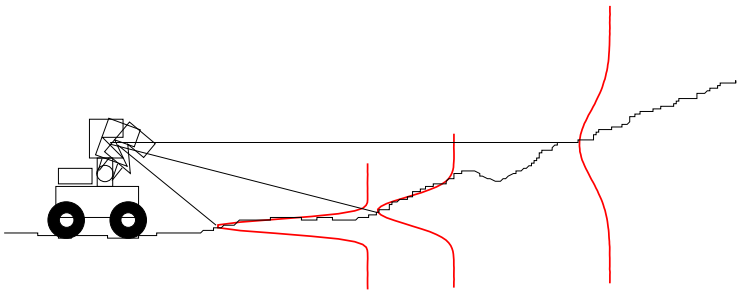

Figure 3: Variance of the height measurements depending on the distance of the beam.

3 Extended Elevation Maps

As already mentioned above, elevation maps are a $2\frac{1}{2}$-dimensional representation of the environment. They maintain a two-dimensional grid and store in every cell of this grid an estimate about the height of the terrain at the corresponding point of the environment. To correctly reflect the actual steepness of the terrain, a common assumption is that the initial tilt and roll of the vehicle are known.

When updating a cell based on sensory input, we have to take into account, that the uncertainty in a measurement increases with the measured distance due to errors in the tilting angle. In our current system, we a apply a Kalman filter to estimate the parameters $\mu_{1:t}$ and $\sigma_{1:t}$ about the elevation of points in a cell and its standard deviation. We apply the following equations to incorporate a new measurement z_t with standard deviation σ_t at time t [19]:

$$\mu_{1:t} = \frac{\sigma_t^2 \mu_{1:t-1} + \sigma_{1:t-1}^2 z_t}{\sigma_{1:t-1}^2 + \sigma_t^2} \quad (1)$$

$$\sigma_{1:t}^2 = \frac{\sigma_{1:t-1}^2 \sigma_t^2}{\sigma_{1:t-1}^2 + \sigma_t^2} \quad (2)$$

Note that the application of the Kalman filter allows us to take into account the uncertainty of the measurement. In our current system, we apply a sensor model, in which the variance of the height of a measurement increases linearly with the distance of the corresponding beam. This process is illustrated in Figure 3. Although this approach is an approximation, we never found evidence in our practical experiments that it causes any noticeable errors.

In addition we need to identify which cells of the elevation map correspond to vertical structures and which ones contain gaps. In order to determine the class of a cell, we first consider the variance of the height of all measurements falling into this cell. If this value exceeds a certain threshold, we identify it as a point that has not been observed from above. We then check whether the point set corresponding to a cell contains gaps exceeding the height of the robot. This is achieved by maintaining a set of intervals per grid cell, which are computed and updated upon incoming sensor data. During this process we join intervals with a distance less than 10cm. Accordingly,

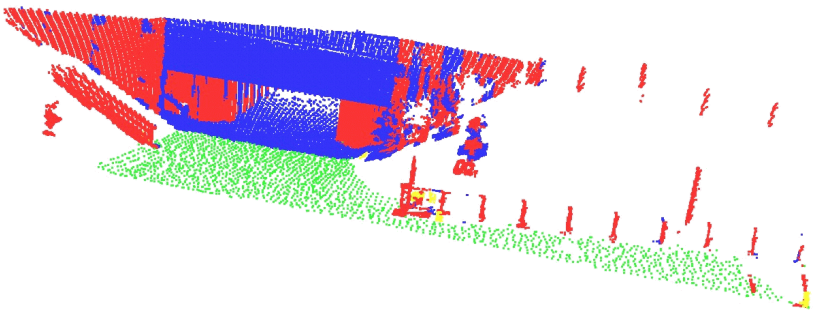

Figure 4: Labeling of the data points depicted in Figure 1 according to their classification. The four different classes are indicated by different colors/grey levels.

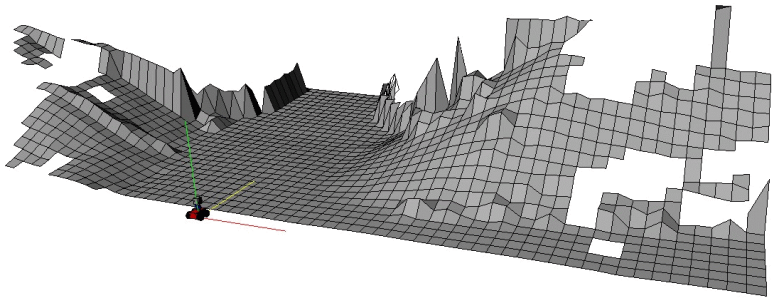

Figure 5: Extended elevation map for the scene depicted in Figure 1.

it may happen, that the classification of a particular cell needs to be changed from the label 'vertical cell' or 'cell that was sensed from above' to the label 'gap cell'. Additionally, it can happen in the case of occlusions that a cell changes from the state 'gap cell' to the state 'vertical cell'. When a gap has been identified, we determine the minimum traversable elevation in this point set.

Figure 4 shows the same data points already depicted in Figure 1. The classes of the individual cells in the elevation map are indicated by the different colors/grey levels. The blue/dark points indicate the data points which are above a gap. The red/medium grey values indicate cells that are classified as vertical. The green/light grey values, however, indicate traversable terrain. Note that the non-traversable cells are not shown in this figure.

A major part of the resulting elevation map computed from this data set is shown in Figure 5, in which we only plot the height values for the lowest interval in each cell. As a result, the area under the bridge now appears as a traversable surface. This allows the robot to plan a safe path through the underpass.

4 Matching Elevation Maps in 6 Dimensions

To calculate the alignments between two local elevation maps generated from individual scans, we apply the Iterative Closest Point (ICP) algorithm. The goal of this process is to find a rotation matrix R and a translation vector $\mathbf{t}$ that minimize an appropriate error function. Assuming that the two scans are represented by point sets $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_{N_1}\}$ and $\mathcal{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_{N_2}\}$, the algorithm first computes a set of C index pairs or *correspondences* $(i_1, j_1), \ldots, (i_C, j_C)$ such that the point $\mathbf{x}_{i_c}$ in $\mathcal{X}$ corresponds to the point $\mathbf{y}_{j_c}$ in $\mathcal{Y}$, for $c = 1, \ldots, C$. Then, in a second step, the error function

$$e(R, \mathbf{t}) \quad := \quad \frac{1}{C} \sum_{c=1}^{C} (\mathbf{x}_{i_c} - (R\mathbf{y}_{j_c} + \mathbf{t}))^T \Sigma^{-1} (\mathbf{x}_{i_c} - (R\mathbf{y}_{j_c} + \mathbf{t})) \tag{3}$$

is minimized. Here, Σ denotes the covariance matrix of the Gaussian corresponding to each pair $(\mathbf{x}_i, \mathbf{y}_i)$. In other words, the error function e is defined by the sum of squared Mahalanobis distances between the points $\mathbf{x}_{i_c}$ and the transformed point $\mathbf{y}_{j_c}$. In the following, we denote this Mahalanobis distance as $d(\mathbf{x}_{i_c}, \mathbf{y}_{j_c})$.

In principle, one could define this function to directly operate on the height values and their variance when aligning two different elevation maps. The disadvantage of this approach, however, is that in the case of vertical objects, the resulting height strongly depends on the view point. The same vertical structure may lead to varying heights in the elevation map when sensed from different points. In practical experiments, we observed that this introduces serious errors and often prevents the ICP algorithm from convergence. To overcome this problem, we separate Equation (3) into four components each minimizing the error over four different classes of points. These four classes correspond to cells containing vertical objects, gap cells, traversable cells, and edge cells. In this context, traversable cells are those for which the elevation of the surface normal obtained from a plane fitted to the local neighborhood exceeds a threshold of 83 degrees. Edge cells are cells that lie more than 20cm above their neighboring points.

Let us assume that $\mathbf{u}_{i_c}$ and $\mathbf{u}'_{j_c}$ are corresponding vertical points, $\mathbf{v}_{i_c}$ and $\mathbf{v}'_{j_c}$ are vertical gap cells, $\mathbf{w}_{i_c}$ and $\mathbf{w}'_{j_c}$ are edge points, and $\mathbf{x}_{j_c}$ and $\mathbf{x}'_{j_c}$ are traversable cells. The resulting error function then is

$$e(R, \mathbf{t}) = \frac{1}{C} \Bigg(\underbrace{\sum_{c=1}^{C_1} d_v(\mathbf{u}_{i_c}, \mathbf{u}'_{j_c})}_{\text{vertical objects}} + \underbrace{\sum_{c=1}^{C_2} d(\mathbf{v}_{i_c}, \mathbf{v}'_{j_c})}_{\text{vertical gaps}} + \underbrace{\sum_{c=1}^{C_3} d(\mathbf{w}_{i_c}, \mathbf{w}'_{j_c})}_{\text{edge cells}} + \underbrace{\sum_{c=1}^{C_4} d(\mathbf{x}_{i_c}, \mathbf{x}'_{j_c})}_{\text{traversable cells}} \Bigg). \tag{4}$$

In this equation, the distance function d_v calculates the Mahalanobis distance between the lowest points in the particular cells. Note that the sum in Equation (3) has been split into four different sums of distances and that $C = C_1 + C_2 + C_3 + C_4$.

To increase the efficiency of the matching process, we only consider a subset of these cells by sub-sampling. In case there are not enough cells in the individual classes, we randomly select an appropriate number of cells (approximately 1,000). This way we obtain an approach that allows to match elevation maps even in the absence of features. In such a situation, our technique becomes equivalent to a standard approach, in which the complete or a random subset of all cells is used for matching.

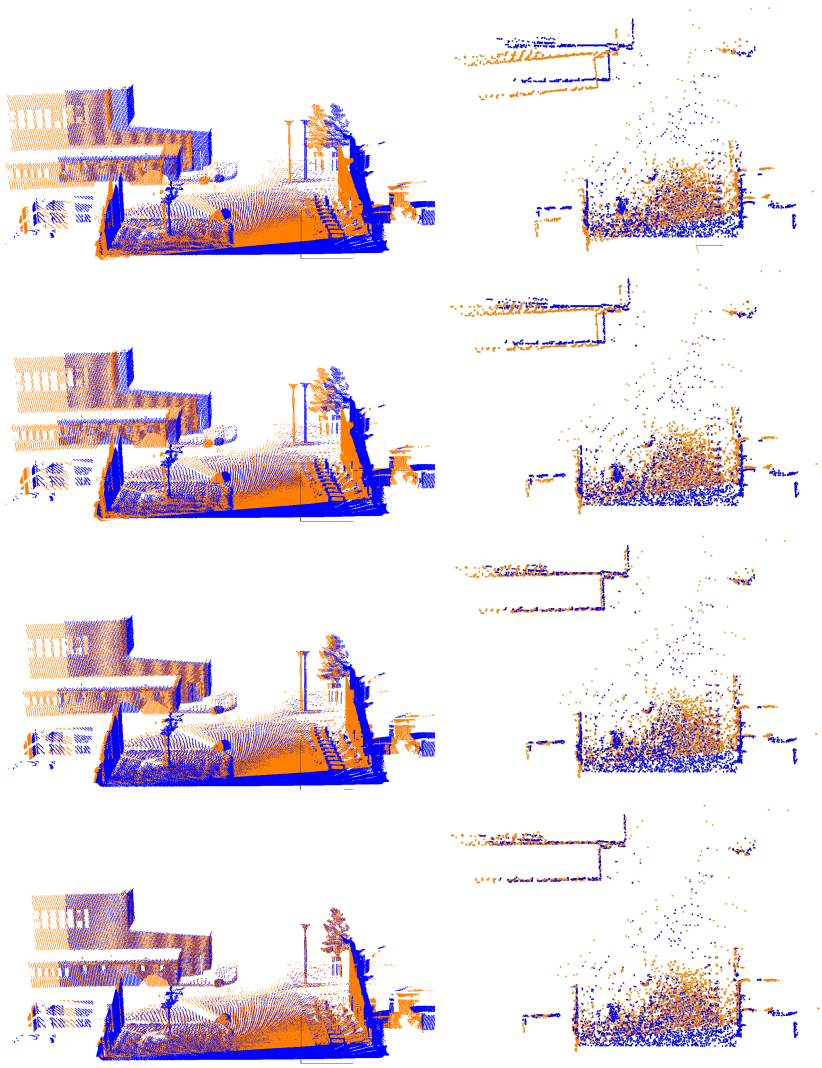

Figure 6: Incremental registration of two elevation maps. The left column depicts the original point clouds (see also Extension 1). The right column shows the 'vertical', 'gap', 'edge', and 'traversable' cells of the elevation maps used by the ICP algorithm. The individual rows correspond to the initial relative pose (top row), alignment after 2 iterations (second row), after 5 iterations (third row) and the final alignment after 15 iterations (last row).

Figure 6 illustrates how two elevation maps are aligned over several iterations of the minimization process. Whereas the left column shows the point clouds, the right column shows a top view on the cells in the elevation map used for minimizing Equa-

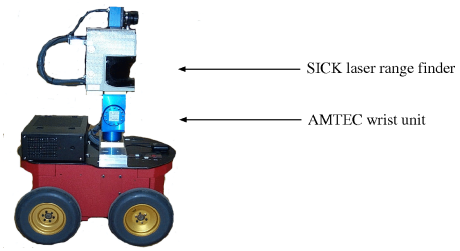

Figure 7: Robot Herbert used for the experiments.

tion (4). In our current implementation, each iteration of the ICP algorithm usually takes between 0.1 and 0.2 seconds on an 800 MHz laptop computer. The time to acquire a scan by tilting the laser is 5 seconds.

5 Loop Closing

The ICP-based scan matching technique described above is known to work well for the registration of single robot poses into one global reference frame. The advantage of our variant based on the four different point classes lies in its efficiency so that it can be carried out while the robot travels from one scan position to the next. It quickly allows the robot to correct potential odometry errors encountered while traveling over uneven terrain. However, the scan matching processes may result in small residual errors which quickly accumulate over time and prevent the robot from building globally consistent maps. In practice, this typically becomes apparent when the robot encounters a loop, i.e., when it returns to a previously visited place. Especially for large loops, this error may result in inconsistencies that prevent the map from being useful for navigation. Accordingly, techniques for calculating globally consistent maps are necessary. In the system described here, we apply a technique similar to the one presented by Lu & Milios [17] to correct for the accumulated error when closing a loop.

5.1 Network-based Pose Optimization

Suppose the robot recorded 3D scans at N different positions and then detects that the first and the last position are so similar that a loop can be closed. As described above, our current system calculates at each position a local elevation map including the classification into vertical objects, vertical gaps, edge cells, and traversable cells. In principle, this process can be considered as extracting point features of four different classes from each local elevation map. The collection of all feature points for a given local elevation map will be called a *partial view* $\mathcal{V}_n$ where $n = 1, \ldots, N$. This means, a partial view $\mathcal{V}_n$ consists of $3D$ feature points of four different kinds. We denote the number of feature points in a view $\mathcal{V}_n$ as s_n and all its points as $\mathbf{z}_1^n, \ldots, \mathbf{z}_{s_n}^n$. Note that each feature point $\mathbf{z}_i^n$ belongs to one of four feature classes. However, for the sake of clarity we assign them all the same symbol $\mathbf{z}$. In fact, the distinction of the feature

classes only improves the data association and has no effect on the following equations. Finally, we define a robot position as a vector $\hat{\mathbf{p}}_n \in \mathbb{R}^3$ and its orientation by the Euler angles $(\varphi_n, \vartheta_n, \psi_n)$. We refer to the *robot pose* $\mathbf{p}_n$ as the tuple $(\hat{\mathbf{p}}_n, \varphi_n, \vartheta_n, \psi_n)$. The goal now is to find a set of robot poses that minimizes an appropriate error function based on the observations $\mathcal{V}_1, \ldots, \mathcal{V}_N$.

Following the approach described by Lu and Milios [17], we formulate the pose estimation problem as a system of error equations that are to be minimized. We represent the set of robot poses as a constraint network, where each node corresponds to a robot pose. A link l in the network is defined by a pair of nodes and represents a *constraint* between the connected nodes. This framework is similar to graph based approaches like the ones presented by Allen *et al.* [2] or Huber and Hebert [11] with the distinction that in the constraint network all links are considered, while in a pose graph only the most confident links are used, either using a Dijkstra-type algorithm [2] or a spanning tree [11]. This means, the network based approach uses all the available information about links between robot poses and not only a part of it.

5.2 Constraints between Robot Poses

A constraint between two robot poses $\mathbf{p}_n$ and $\mathbf{p}_m$ is derived from the corresponding views $\mathcal{V}_n$ and $\mathcal{V}_m$. Assuming that we are given a set of C_{nm} point correspondences $(i_1, j_1), \ldots, (i_{C_{nm}}, j_{C_{nm}})$ between $\mathcal{V}_n$ and $\mathcal{V}_m$ as described above, we define the constraint between poses $\mathbf{p}_n$ and $\mathbf{p}_m$ as the sum of squared errors between corresponding points in the global reference frame

$$l(\mathbf{p}_n, \mathbf{p}_m) \quad := \quad \sum_{c=1}^{C_{nm}} \|(\hat{R}_n \mathbf{z}_{i_c}^n + \hat{\mathbf{t}}_n) - (\hat{R}_m \mathbf{z}_{j_c}^m + \hat{\mathbf{t}}_m)\|^2. \tag{5}$$

Here, the transformation between the local coordinates $\mathbf{z}^n$ and the global coordinates is represented as a global rotation matrix $\hat{R}_n$, which is computed from the Euler angles $(\varphi_n, \vartheta_n, \psi_n)$, and the global translation vector $\hat{\mathbf{t}}_n$, which is equal to the robot position $\hat{\mathbf{p}}_n$. These transforms $(\hat{R}_n, \hat{\mathbf{t}}_n)$ into the global reference frame are different from the local transforms $(R, \mathbf{t})$ from Equation (4). In fact, the local transforms obtained after convergence of our modified ICP algorithm are not needed any more, because we only need to consider the correspondences resulting from the last ICP step. Also note that in Equation (5), the number of correspondences C_{nm} is equal to the sum $C_1 + C_2 + C_3 + C_4$ from Equation (4), because the distinction of the different feature classes is not needed at this point.

Let us assume that the network consists of L constraints $l_1, \ldots, l_L$. Note that the number of links is not necessarily equal to the number N of robot poses, because links can also exist between non-consecutive poses. The robot pose estimation can then be formulated as a minimization problem of the overall error function

$$f(\mathbf{p}_1, \ldots, \mathbf{p}_N) \quad := \quad \sum_{i=1}^{L} l_i(\mathbf{p}_{\nu_1(i)}, \mathbf{p}_{\nu_2(i)}). \tag{6}$$

Here, we introduced the indexing functions ν_1 and ν_2 to provide a general formulation for any kind of network setting, in which links can exist between any pair of robot

poses. In the simplest case, in which we have only one loop and only links between consecutive poses, we have $\nu_1(i) = i$ and $\nu_2(i) = (i + 1) \bmod N$.

To solve this non-linear optimization problem we derive f analytically with respect to all positions $\mathbf{p}_1, \ldots, \mathbf{p}_N$ and apply the Fletcher-Reeves gradient descent algorithm to find a minimum. In general, this minimum is local and there is no guarantee that the global minimum is reached. However, in our experiments the minimization always converged to a value that was at least very close to the global minimum. We also found that the convergence can be improved by restarting the scan matching process with the new, optimized robot poses as initial values. In this way, we obtain an iterative algorithm that stops when the change in the robot poses drops under a given threshold or no improvement can be achieved over several iterations.

It should be noted that in general the global minimum for the error function f is not unique. This is because both local and global constraints are only defined with respect to the relative displacements between the robot poses and the global minimum of f is invariant with respect to affine transformations of the poses. In practice, this problem can be solved by fixing one of the robot poses at its initial value. The other poses are then optimized relative to this fixed pose.

6 Experimental Results

The approach described above has been implemented and tested on a real robot system and in simulation runs with real data. The overall implementation is highly efficient. Whereas the scan matching process can be carried out online, while the robot is moving, the loop closing algorithm typically required between 3 and 10 minutes for the data sets described below and on a standard laptop computer. The robot used to acquire the data is our outdoor robot Herbert, which is depicted in Figure 7. The robot is a Pioneer II AT system equipped with a SICK LMS range scanner and an AMTEC wrist unit, which is used as a pan/tilt device for the laser. The experiments described in this section have been designed to illustrate that our approach yields highly accurate elevation maps even containing large loops and that the consideration of the individual classes in the data association leads to more accurate matchings.

6.1 Learning Large-scale Elevation Maps with Loops

To evaluate our approach on large-scale data sets, we steered our robot Herbert through two loops on our university campus and acquired 135 scans consisting of 35,500,000 data points. The area scanned by the robot spans approximately 160 by 120 meters. During the data acquisition, the robot traversed two nested loops. Throughout the evaluations described below, the inner loop, which has a length of 188m and consists of 58 scans, is referred to as loop 1. The outer loop, which has a length of 284m and consists of 77 scans, is denoted as loop 2. The map has been computed according to the pose estimates calculated with our SLAM algorithm. To allow a quantitative evaluation of the results, we always let the robot return to its starting position after closing each loop. Figure 8 shows top-views of three different elevation maps obtained from this data set. Whereas the leftmost image shows the map obtained from raw

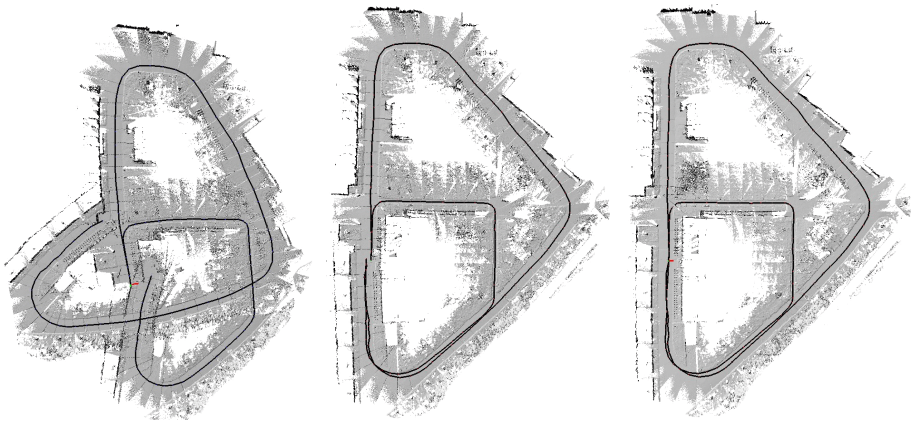

Figure 8: The leftmost image shows the map obtained from raw odometry, the middle image depicts the map obtained from the pure scan matching technique described in our previous work [26]. The rightmost image shows the map obtained from our SLAM algorithm described in Section 5. In these maps, the size of each cell of the elevation maps is 10 by 10cm. The lines show the estimated trajectory of the robot. The size of all the maps is approximately 160 by 120 meters.

Figure 9: Triangulated mesh representation of the outer loop including data points from 77 laser scans. Extension 2 shows a virtual walk through this model on the trajectory taken by the robot. Figure 10 depicts the elevation map of the same scene.

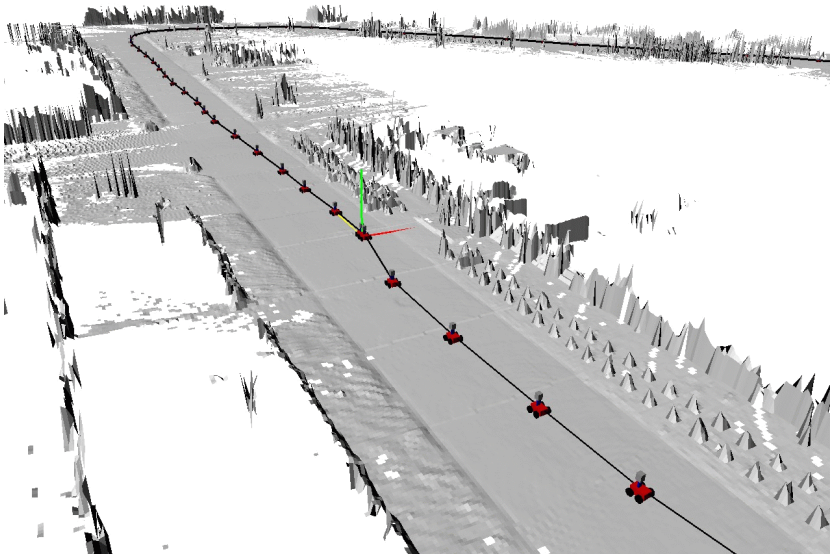

Figure 10: Elevation map corresponding to the data set shown in Figure 9.

odometry, the middle image depicts the map obtained from the pure scan matching technique described in our previous work [26]. The rightmost image shows the map obtained from our SLAM algorithm described in Section 5. In these maps, the size of each cell of the elevation maps is 10 by 10cm. The lines show the estimated trajectory of the robot. As can be seen from the figure, the scan matching process substantially decreases the odometry error but fails to correctly close the loop. Using our SLAM algorithm, in contrast, the loop has been closed correctly. Figure 9 shows a triangulated mesh representation of the entire scan point data set of loop 2. Extension 2 shows a virtual walk through this model on the trajectory taken by the robot. Figure 10 depicts the elevation map of the same scene.

Figure 11 depicts the area around the starting location after closing both of the loops. The two images also show the trajectories estimated by odometry, our scan matching procedure, and our loop closing algorithm. As can be seen from the figure, the network-based pose optimization algorithm yields the smallest localization error in both cases.

To quantitatively evaluate the accuracy of our loop closing procedure, we determined the estimated pose of the vehicle, given that the initial position was set to zero. Table 1 shows the estimates of the odometry after closing the loop. As can be seen, the translational error exceeds several meters and the angular error is 13 and 60 degrees respectively. Table 2 shows the positions as they were calculated by our incremental scan matching algorithm. This procedure substantially reduces the error in the odometry, especially the angular deviation. However, the pose error is still too large to correctly close the loop. Finally, Table 3 shows the pose estimates obtained with our loop closing procedure, which uses the results of the scan-matching process to calculate the

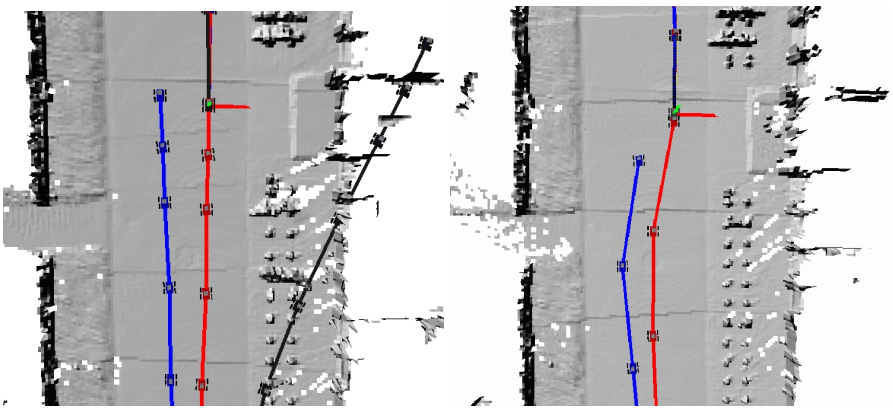

Figure 11: Partial views of the elevation maps containing the area where the two loops start and end. Both maps were obtained using our loop closing algorithm. The different lines indicate the trajectories estimated by odometry (black), using the scan-matching procedure (dark/blue), and our SLAM algorithm (grey/red). As can be seen, the scan matching process is highly accurate but still fails to correctly close the loop. Our SLAM algorithm correctly identifies that the robot has returned to the position where it started.

loop	length	x	y	ψ
1	$188m$	$-7.968m$	$2.3676m$	$13.126°$
2	$284m$	$-6.919m$	$24.678m$	$59.583°$

Table 1: Poses estimated by odometry after closing the loops.

loop	length	x	y	z	ϕ	θ	ψ
1	$188m$	$-1.767m$	$0.353m$	$-0.231m$	$1.235°$	$0.235°$	$0.751°$
2	$284m$	$-1.375m$	$-1.916m$	$-0.464m$	$1.201°$	$0.435°$	$2.956°$

Table 2: Poses estimated by the incremental online scan matching algorithm after closing the loops.

loop	length	x	y	z	ϕ	θ	ψ
1	$188m$	$0.006m$	$0.064m$	$-0.010m$	$0.097°$	$0.008°$	$0.631°$
2	$284m$	$0.007m$	$-0.303m$	$-0.006m$	$0.206°$	$0.057°$	$1.257°$

Table 3: Poses estimated by the loop closing algorithm.

constraints of the network. As can be seen, the angular error drops below one degree and also the pose error is seriously reduced.

To further evaluate our method in non-flat environments, we steered the robot through an underpass and then uphill on a ramp. Figure 12 shows a photograph of the corresponding area. The map obtained with our algorithm is depicted in Figure 13.

Figure 12: Photograph of the area where the map depicted in Figure 13 has been built.

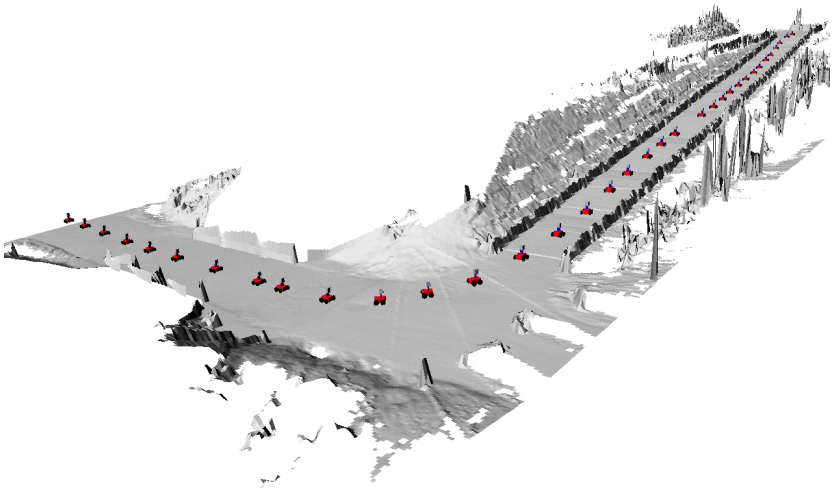

Figure 13: Elevation map generated from 36 local elevation maps. The size of the map is approximately 70 by 30 meters.

It has been obtained from 36 scans with an overall number of 9,500,000 data points. The size of each cell in the elevation map is 10 by 10cm. The whole map spans approximately 70 by 30 meters. As can be seen from the figure, the map clearly reflects the details of the environment. Additionally, the matching of the elevation maps is quite accurate. The figure also shows the individual positions of the robot where the scans were taken.

Additionally, we compared our approach to the standard elevation map algorithm. Figure 14 shows a typical scenario in which our algorithm yields more accurate maps than the standard approach. In this situation the robot traveled along a paved way and scanned a tree located in the foreground of the scene (see left image of Figure 14). Whereas the middle image shows the map obtained with the standard elevation map approach, the right image shows the map obtained with our method. Again, the figure

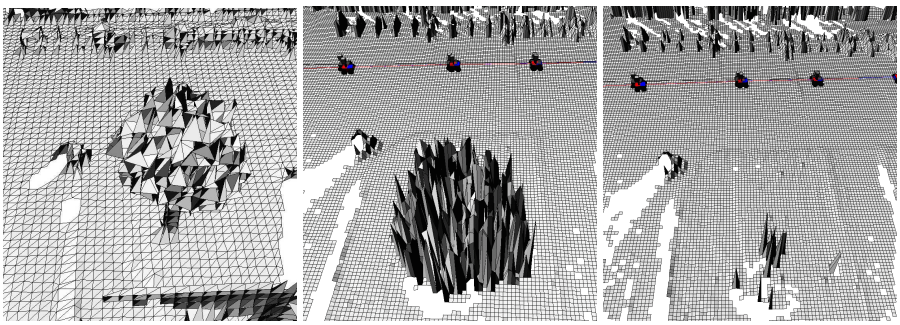

Figure 14: Comparison of our elevation maps to the standard elevation map technique. The left image depicts a triangulated mesh of the whole scan data. The image in the middle shows a standard elevation map build from four scans. The right image depicts the map obtained with our approach from the same data. The peak in the foreground of the scene corresponds to the tree depicted in the left image. As can be seen from the right image, the errors introduced by the treetop are substantially reduced using our representation.

displacement class	max. rotation	max. translation
1	±5 degrees	$\pm 0.5m$
2	±5 degrees	$\pm 1.0m$
3	±5 degrees	$\pm 1.5m$
4	±5 degrees	$\pm 2.0m$
5	±5 degrees	$\pm 2.5m$

Table 4: Displacement classes used to evaluate the performance of the ICP algorithm on the classified and unclassified points extracted from the elevation maps.

shows the positions of the robot where the scans were taken. As can be seen, our method results in more free space around the trunk of the tree compared to the standard elevation map approach. This is due to the fact that the cells containing the treetop are classified as overhanging and only the value for the lowest interval in our elevation map is shown.

6.2 Statistical Evaluation of the Feature-based Registration

Additionally, we performed a series of experiments to get a statistical assessment as to whether the classification of the data points into normal, vertical, and gap points combined with the sub-sampling of the normal points leads to better registration results. To perform these experiments we considered two different elevation maps for which we computed the optimal relative pose using several runs of the ICP algorithm. We then randomly added noise to the pose of the second map and applied the ICP algorithm to register both maps. We performed two sets of experiments to compare the registration

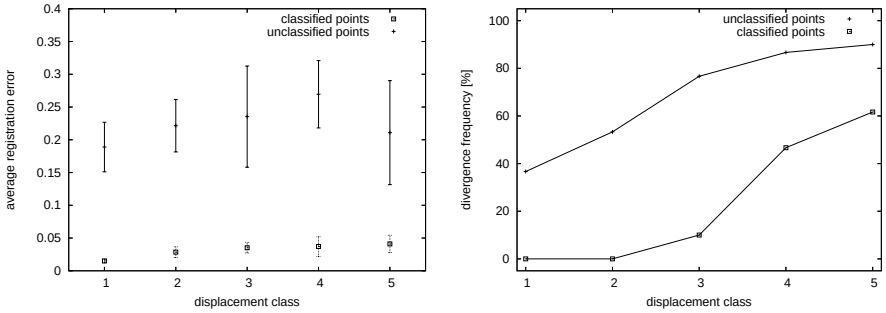

Figure 15: The left image shows the average registration error for the individual types of initial displacement. The results have been averaged over 200 runs. The right image depicts the number of times the ICP algorithm diverges for the individual initial displacements.

results for the unclassified and the classified point sets. Table 4 shows the individual classes of noise that were added to the true relative pose of the two maps before we started the ICP algorithm. In the experiment described here, we only varied the pose error of the maps and kept the error in the rotations constant. In particular, we randomly chose rotational displacements from ± 5 degrees around the real relative angle as well as varying random displacements in the x and y direction.

The resulting average displacement errors after convergence of the ICP algorithm are depicted in the left column of Figure 15. As can be seen from the figure, the ICP algorithm performed significantly better on the classified point sets. The error bars indicate the $\alpha = 0.05$ confidence level. The results have been averaged over 200 runs.

Additionally, we evaluated how often the ICP algorithm failed to accurately register the two maps. The right column of Figure 15 depicts the normalized divergence frequencies in percent for the individual displacement classes. As this plot illustrates, the utilization of the individual classes in the ICP algorithm leads to a substantially better convergence rate. In additional experiments not reported here, we obtained similar results for the different orientational errors.

7 Conclusions

In this paper we presented an approach to solve the SLAM problem with elevation maps generated from three-dimensional range data acquired with a mobile robot. Our approach especially addresses the problem of acquiring such maps with a ground-based vehicle. On such a system one often encounters situations, in which certain objects, such as walls or trees, are not seen from above. Accordingly, the resulting elevation maps contain incorrect information. The approach in this paper classifies the individual cells of elevation maps into four classes representing parts of the terrain seen from above, vertical objects, overhanging objects such as branches of trees or bridges, and traversable areas. We also presented an extension of the ICP algorithm that takes this

classification into account when computing the registration. Additionally we use a technique for constraint-based robot pose estimation to learn globally consistent elevation maps.

Our algorithm has been implemented and tested on outdoor terrain data containing two loops. In practical experiments our constraint-based pose optimization yielded highly accurate maps. Additionally, the consideration of the individual classes during the data association in the ICP algorithm provides more robust correspondences and more accurate alignments.

Acknowledgment

This work has partly been supported by the German Research Foundation (DFG) within the Research Training Group 1103 and under contract number SFB/TR-8.

A Index to Multimedia Extensions

The multimedia extensions to this article can be found online by following the hyperlinks from www.ijrr.org.

Extension	Type	Description
1	Video	Incremental registration of two original point clouds
2	Video	Triangulated mesh representation of the outer loop including data points from 77 laser scans. The extension shows a virtual walk through this model on the trajectory taken by the robot.

References

[1] P. Allen, I. Stamos, A. Gueorguiev, E. Gold, and P. Blaer. Avenue: Automated site modeling in urban environments. In *Proc. of the 3rd Conference on Digital Imaging and Modeling*, pages 357–364, 2001.

[2] Peter K. Allen, Ioannis Stamos, Alejandro Troccoli, Benjamin Smith, M. Leordeanu, and Y. C. Hsu. 3D modeling of historic sites using range and image data. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, Tapei, 2003.

[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] P.J. Besl and N.D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:239–256, 1992.

[5] A.J. Davison, Y. Gonzalez Cid, and N. Kita. Real-time 3d SLAM with wide-angle vision. In *Proc. of the 5th IFAC Symposium on Intelligent Autonomous Vehicles (IAV)*, 2004.

[6] G. Dissanayake, P. Newman, S. Clark, H.F. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localisation and map building (SLAM) problem. *IEEE Transactions on Robotics and Automation*, 17(3):229–241, 2001.

[7] C. Früh and A. Zakhor. An automated method for large-scale, ground-based city model acquisition. *International Journal of Computer Vision*, 60:5–24, 2004.

[8] J. Guivant and E. Nebot. Optimization of the simultaneous localization and map building algorithm for real time implementation. *IEEE Transactions on Robotics and Automation*, 17(3):242–257, 2001.

[9] D. Hähnel, W. Burgard, and S. Thrun. Learning compact 3d models of indoor and outdoor environments with a mobile robot. *Robotics and Autonomous Systems*, 44(1):15–27, 2003.

[10] 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.

[11] Daniel Huber and Martial Hebert. Fully automatic registration of multiple 3D data sets. In *IEEE Computer Society Workshop on Computer Vision Beyond the Visible Spectrum(CVBVS 2001)*, December 2001.

[12] E. Hygounenc, I.-K. Jung, P. Souères, and S. Lacroix. The autonomous blimp project of laas-cnrs: Achievements in flight control and terrain mapping. *International Journal of Robotics Research*, 23(4-5):473–511, 2004.

[13] P. Kohlhepp, M. Walther, and P. Steinhaus. Schritthaltende 3D-Kartierung und Lokalisierung für mobile inspektionsroboter. In *18. Fachgespräche AMS*, 2003. In German.

[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. *International Journal of Robotics Research*, 21(10-11):917–942, 2002.

[15] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk. The digital michelangelo project: 3D scanning of large statues. In *Proc. SIGGRAPH*, pages 131–144, 2000.

[16] Y. Liu, R. Emery, D. Chakrabarti, W. Burgard, and S. Thrun. Using EM to learn 3D models with mobile robots. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2001.

[17] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4:333–349, 1997.

[18] C. Martin and S. Thrun. Online acquisition of compact volumetric maps with mobile robots. In *IEEE International Conference on Robotics and Automation (ICRA)*, Washington, DC, 2002. ICRA.

[19] P.S. Maybeck. The Kalman filter: An introduction to concepts. In *Autonomous Robot Vehicles*. Springer Verlag, 1990.

[20] M. Montemerlo and S. Thrun. A multi-resolution pyramid for outdoor robot terrain perception. In *Proc. of the National Conference on Artificial Intelligence (AAAI)*, 2004.

[21] H.P. Moravec. Robot spatial perception by stereoscopic vision and 3d evidence grids. Technical Report CMU-RI-TR-96-34, Carnegie Mellon University, Robotics Institute, 1996.

[22] A. Nüchter, K. Lingemann, J. Hertzberg, and H. Surmann. 6d SLAM with approximate data association. In *Proc. of the 12th Int. Conference on Advanced Robotics (ICAR)*, pages 242–249, 2005.

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

[24] 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.

[25] K. Pervölz, A. Nüchter, H. Surmann, and J. Hertzberg. Automatic reconstruction of colored 3d models. In *Proc. Robotik*, 2004.

[26] P. Pfaff and W. Burgard. An efficient extension of elevation maps for outdoor terrain mapping. In *Proc. of the International Conference on Field and Service Robotics (FSR)*, pages 165–176, 2005.

[27] H. Samet. *Applications of Spatial Data Structures*. Addison-Wesley Publishing Inc., 1989.

[28] S. Singh and A. Kelly. Robot planning in the space of feasible actions: Two examples. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 1996.

[29] S. Thrun, W. Burgard, and D. Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2000.

[30] S. Thrun, D. Hähnel, D. Ferguson, M. Montemerlo, R. Triebel, W. Burgard, C. Baker, Z. Omohundro, S. Thayer, and W. Whittaker. A system for volumetric robotic mapping of abandoned mines. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2003.

[31] S. Thrun, Y. Liu, D. Koller, A.Y. Ng, Z. Ghahramani, and H. Durant-Whyte. Simultaneous localization and mapping with sparse extended information filters. *International Journal of Robotics Research*, 23(7-8):693–704, 2004.

[32] S. Thrun, C. Martin, Y. Liu, D. Hähnel, R. Emery Montemerlo, C. Deepayan, and W. Burgard. A real-time expectation maximization algorithm for acquiring multi-planar maps of indoor environments with mobile robots. *IEEE Transactions on Robotics and Automation*, 20(3):433–442, 2003.

[33] R. Triebel and W. Burgard. Improving simultaneous mapping and localization in 3d using global constraints. In *Proc. of the National Conference on Artificial Intelligence (AAAI)*, 2005.

[34] R. Triebel, F. Dellaert, and W. Burgard. Using hierarchical EM to extract planes from 3d range scans. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2005.

[35] Carl Wellington, Aaron Courville, and Anthony Stentz. Interacting markov random fields for simultaneous terrain modeling and obstacle detection. In *Proceedings of Robotics: Science and Systems*, Cambridge, USA, June 2005.

[36] O. Wulf, K-A. Arras, H.I. Christensen, and B. Wagner. 2d mapping of cluttered indoor environments by means of 3d perception. In *ICRA-04*, pages 4204–4209, New Orleans, apr 2004. IEEE.

[37] C. Ye and J. Borenstein. A new terrain mapping method for mobile robot obstacle negotiation. In *Proc. of the UGV Technology Conference at the 2002 SPIE AeroSense Symposium*, 1994.