Available at www.ComputerScienceWeb.com
POWERED BY SCIENCE DIRECT®

Robotics and Autonomous Systems 45 (2003) 181–198

Robotics and Autonomous Systems

www.elsevier.com/locate/robot

An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments

Hartmut Surmann*, Andreas Nüchter, Joachim Hertzberg

Fraunhofer Institute for Autonomous Intelligent Systems (AIS), Schloss Birlinghoven, D-53754 Sankt Augustin, Germany

Received 8 January 2003; received in revised form 22 August 2003; accepted 22 September 2003

Abstract

Digital 3D models of the environment are needed in rescue and inspection robotics, facility managements and architecture. This paper presents an automatic system for gaging and digitalization of 3D indoor environments. It consists of an autonomous mobile robot, a reliable 3D laser range finder and three elaborated software modules. The first module, a fast variant of the Iterative Closest Points algorithm, registers the 3D scans in a common coordinate system and relocalizes the robot. The second module, a next best view planner, computes the next nominal pose based on the acquired 3D data while avoiding complicated obstacles. The third module, a closed-loop and globally stable motor controller, navigates the mobile robot to a nominal pose on the base of odometry and avoids collisions with dynamical obstacles. The 3D laser range finder acquires a 3D scan at this pose. The proposed method allows one to digitalize large indoor environments fast and reliably without any intervention and solves the SLAM problem. The results of two 3D digitalization experiments are presented using a fast octree-based visualization method.

Keywords: Autonomous mobile robots; 3D laser range finder; Scan matching; Next best view planning; 3D digitalization; 3D gaging; Robot relocalization; SLAM

1. Introduction

The increasing need for rapid characterization and quantification of complex environments has created challenges for data analysis. This critical need comes from many important areas, including industrial automation, architecture, agriculture, and the construction or maintenance of tunnels and mines. On one hand, precise 3D data of environments are necessary for factory design, facility management, urban and regional planning. Especially mobile systems with 3D laser scanners that automatically perform multiple steps such as scanning, gaging and autonomous driving have the potential to greatly advance the field of environment sensing. On the other hand, 3D information available in real-time enables autonomous robots to navigate in unknown environments, e.g., in the field of inspection and rescue robotics.

The problem of automatic environment sensing and modeling is complex, because a number of fundamental scientific issues are involved in this research. First is the control of an autonomous mobile robot and the environment scanning with a 3D sensor. Second is how to create a volumetric consistent scene in a common coordinate system from multiple views. Third is the computation of next view points to provide efficiently full coverage of the scene and to eliminate

* Corresponding author.
E-mail addresses: surmann@ais.fraunhofer.de (H. Surmann), nuechter@ais.fraunhofer.de (A. Nüchter), hertzberg@ais.fraunhofer.de (J. Hertzberg).

0921-8890/$ – see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/j.robot.2003.09.004

occlusion under the constraint of minimizing the total path length between these points. This paper addresses these three problems. A robot equipped with the AIS 3D laser range finder and no prior knowledge of its surroundings explores the world and creates reliably a precise and consistent 3D volumetric representation in real-time. A consistent volumetric model requires a solution to the simultaneous localization and map building problem (SLAM problem).

Some groups have attempted to build 3D volumetric representations of environments with 2D laser range finders. Thrun et al. [1,2], Früh and Zakhor [3] and Zhao and Shibasaki [4] use two 2D laser range finder for acquiring 3D data. One laser scanner is mounted horizontally and one is mounted vertically. The latter one grabs a vertical scan line which is transformed into 3D points using the current robot pose. Since the vertical scanner is not able to scan sides of objects, Zhao and Shibasaki [4] use two additional vertical mounted 2D scanner shifted by 45° to reduce occlusion. The horizontal scanner is used to compute the robot pose. The precision of 3D data points depends on that pose and on the precision of the scanner. All these approaches have difficulties to navigate around 3D obstacles with jutting out edges. They are only detected while passing them. Furthermore, exploration schemes for environment digitalization are missing. The published 2D probabilistic localization approaches, e.g., Markov models or Kalman filters work well in flat and structured 2D environments but an extension in the third dimension is still missing since the algorithm do not scale with additional dimensions.

A few other groups use 3D laser scanners [5–10]. A 3D laser scanner generates consistent 3D data points within a single 3D scan. The RESOLV project aimed to model interiors for virtual reality and telepresence [5–7]. They used a RIEGL laser range finder on two robots, called EST and AEST (Autonomous Environmental Sensor for Telepresence). They use the Iterative Closest Points (ICP) algorithm [11–13] for scan matching and a perception planning module for minimizing occlusions. A closed-loop, stable motor controller for autonomous positioning of the robot is not given. The AVENUE project develops a robot for modeling urban environments [8–10] using a CYRAX laser scanner. They employ a planning module [14], which calculates set intersections of volumes to calculate occluded volumes and to create a solid model. A closed-loop, stable motor controller for autonomous positioning of the robot is missing.

The paper is organized as follows. Section 2 describes the autonomous mobile robot and AIS 3D laser range finder. Sections 3–5 present the registration algorithms, the next best view planner and the motor controller, respectively. Section 6 shows experiments and results. Section 7 concludes the paper.

2. The autonomous mobile robot and the 3D laser range finder

2.1. The Ariadne robot

The Ariadne robot (Fig. 1) is an industrial DTV and is about 80 cm × 60 cm large and 90 cm high. The mobile platform can carry a payload of 200 kg at speeds of up to 0.8 m/s (about half the speed of a pedestrian). The right and left driving wheels are mounted on a suspension on the center line of the mobile platform. Passive castors on each corner of the chassis ensure stability.

The core of the robot is a Pentium-III-800 MHz with 384 MB RAM and real-time Linux. One embedded PC-104 system is used to control the motor, internal display and numerical keyboard and radio link of the robot. The platform is rigged with two 2D safety laser scanners as bumper substitutes, one on the front and the other on the rear of the robot. Each laser scans a horizontal plane of 180° of the environment. The robot has a weight of 250 kg and operates for about 8 h with one battery charge. Our motor controller is described in Section 5.

2.2. The AIS 3D laser range finder

The AIS 3D laser range finder [5] is built on the basis of a 2D range finder by extension with a mount and a servomotor. The 2D laser range finder is attached to the mount for being rotated. The rotation axis is horizontal (pitch). A standard servo is connected on the left side (Fig. 1) and is controlled by the computer running RT-Linux, a real-time operating system which runs Linux as a task with lowest priority [15,16]. The 3D laser scanner operates up to 5 h (scanner: 17 W, 24 V with batteries of 4.5 A h, servo: 0.85 W, 4.5 V with batteries of 4.5 A h).

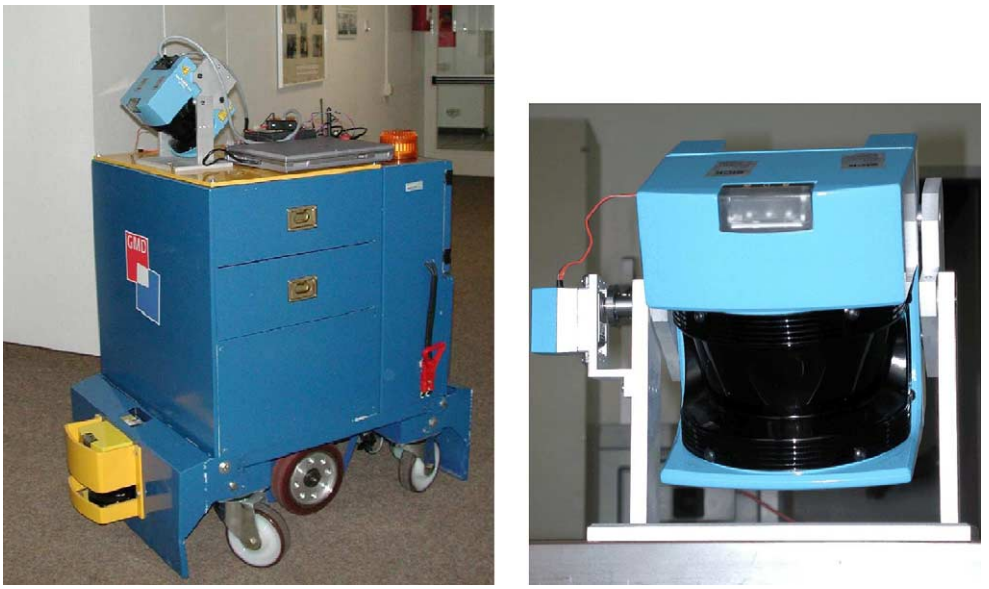

Fig. 1. Left: the Ariadne robot platform equipped with the 3D scanner; right: the AIS 3D laser range finder.

The area of 180° (h) × 120° (v) is scanned with different horizontal (181, 361, 721) and vertical (128, 256) resolutions. A plane with 181 data points is scanned in 13 ms by the 2D laser range finder (rotating mirror device). Planes with more data points, e.g., 361, 721, duplicate or quadruplicate this time. Thus a scan with 181 × 256 data points needs 3.4 s. In addition to the distance measurement, the 3D laser range finder is capable of quantifying the amount of light returning to the scanner. Fig. 2 (top row) shows an example of a reflectance image of the GMD Robobench, a standard office environment for the evaluation of autonomous robots. The left image gives a distorted view of the scene: one scan line of the figure corresponds to a slice of the 2D scanner, the rotation of the scanner is not considered. The right image shows the scene, with the distortions corrected.

2.2.1. Scanner software

The basis of the scan matching algorithms and the next best view planner are algorithms for reducing points, line detection, surface extraction and object segmentation. Next we give a brief description of these algorithms. Details can be found in [15,16].

The scanner emits the laser beams in a spherical way, such that the data points close to the source are more dense. The first step is to reduce the data. Therefore, data points located close together are joined into one point. The number of these so-called *reduced points* is one order of magnitude smaller than the original one (Fig. 5 (middle)). Furthermore, noise within the data is reduced [16].

Second, a simple length comparison is used as a line detection algorithm. Given that the anticlockwise ordered data of the laser range finder (points $a_0, a_1, \ldots, a_n$) are located on a line, then for a_{j+1} the algorithm has to check if $||a_i, a_{j+1}||/\sum_{t=i}^{j}||a_t, a_{t+1}|| > \epsilon(j)$ to determine if a_{j+1} is on line with a_j.

The third step is surface detection. Scanning a plane surface, line detection returns a sequence of lines in successive scanned 2D planes approximating the shape of surfaces. Thus a plain surface consists of a set of lines. A single line or the last line of an already existing surface is merged into a surface if the following three criteria are fulfilled: the first and last points of the new line have to be located within an ϵ-area of a previously found line. The angle between the two lines should be small and the new line has to be on the plane of the existing surface. Polygons are created from the detected surfaces. Since surfaces may overlap, we use Vatti's Polygon clipping algorithm [17] to calculate the union of the overlapping surfaces [16].

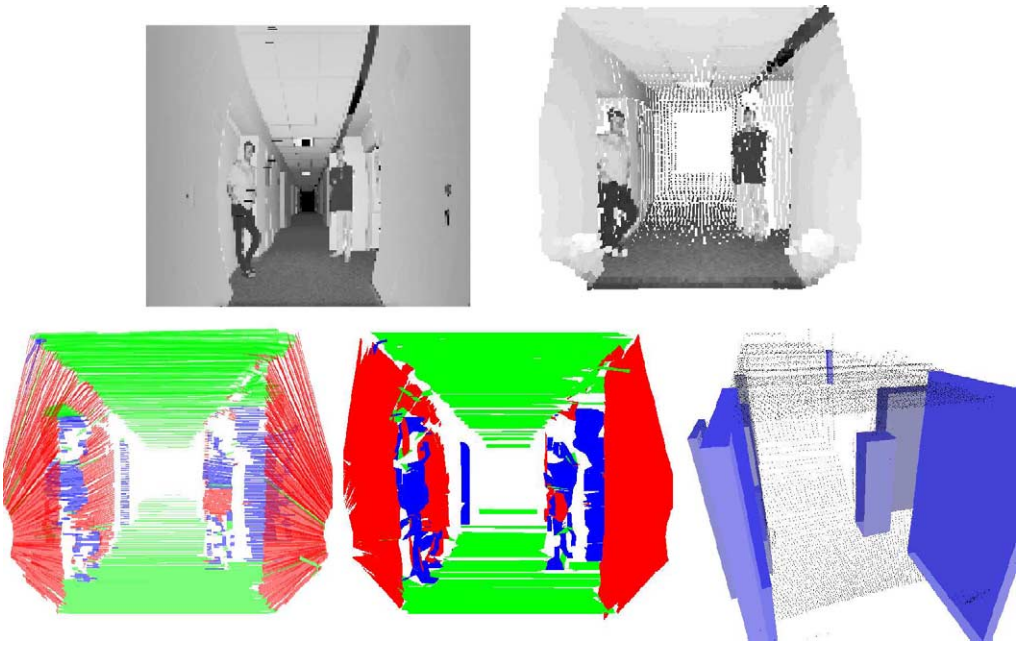

Fig. 2. Two persons standing in a corridor of an office building (GMD Robobench) – top left: reflectance image (distorted); top right: a 3D view of the scene (corrected reflectance image); bottom left: line detection; bottom middle: surface extraction; bottom right: object segmentation.

The fourth and final step computes occupied space. For this purpose, conglomerations of surfaces and polygons are merged sequentially into objects. Two steps are necessary to find bounding boxes around objects. First a bounding box is placed around each large surface and polygon. In the second step objects close to each other are merged together, e.g., one should merge objects closer than the size of the robot, since the robot cannot pass between such objects.

Data reduction, line and surface detection are online algorithms and run in parallel to the 3D scanning process whereas polygon creation and object segmentation run after the 3D scan. For a typical indoor scene the offline algorithms need around 1 s on a Pentium-III-600.

3. Range image registration and pose estimation

Multiple 3D scans are necessary to digitalize environments without occlusions. To create a correct and consistent model, the scans have to be merged in one coordinate system. This process is called registration. If the localization of the robot with the 3D scanner were precise then the registration could be done based on the robot pose. Due to the robot's sensors, the self-localization is usually erroneous and unprecise, so the geometric structure of overlapping 3D scan has to be considered for registration. Scan matching approaches can be classified into two categories:

- *Matching as optimization problem* uses a cost function for the quality of the alignment of the scans. The range images are registered by determining the rigid transformation (rotation and translation) which minimizes the cost function.
- *Feature based matching* extracts distinguishing features of the range images and uses corresponding features for calculating the alignment of the scans.

3.1. Matching as optimization problem

The following method for registration of point sets is part of many publications, so only a short summary is given here. The complete algorithm was invented 1991 and can be found, e.g., in [11–13]. The method is called *Iterative Closest Points* (*ICP*) *algorithm*.

Given two independently acquired sets of 3D points, M (model set, $|M| = N_m$) and D (data set, $|D| = N_d$), which correspond to a single shape, we want to find the transformation consisting of a rotation $\mathbf{R}$ and a translation $\mathbf{t}$ which minimizes the following cost function:

$$E(\mathbf{R}, \mathbf{t}) = \sum_{i=1}^{N_m} \sum_{j=1}^{N_d} w_{i,j} ||\mathbf{m}_i - (\mathbf{R}\mathbf{d}_j + \mathbf{t})||^2, \quad (1)$$

$w_{i,j}$ is assigned 1 if the ith point of M describes the same point in space as the jth point of D. Otherwise $w_{i,j}$ is 0. Two things have to be calculated: first, the corresponding points, and second, the transformation $\mathbf{R}$ and $\mathbf{t}$ that minimizes $E(\mathbf{R}, \mathbf{t})$ on the base of the corresponding points.

The ICP algorithm calculates iteratively the point correspondence. In each iteration step, the algorithm selects the closest points as correspondences and calculates the transformation $(\mathbf{R}, \mathbf{t})$ for minimizing Eq. (1). It is shown that the iteration terminates in a (local) minimum [11]. The assumption is that in the last iteration step the point correspondences are correct.

In each iteration, the transformation is calculated by the quaternion based method of Horn [18]. A unit quaternion is a four vector $\dot{q} = (q_0, q_x, q_y, q_z)^{\mathrm{T}}$, where $q_0 \geq 0, q_0^2 + q_x^2 + q_y^2 + q_z^2 = 1$. It describes a rotation axis and an angle to rotate around that axis. A 3×3 rotation matrix $\mathbf{R}$ is calculated from the unit quaternion according the following scheme:

$$\mathbf{R} = \begin{pmatrix} (q_0^2 + q_x^2 - q_y^2 - q_z^2) & 2(q_x q_y + q_z q_0) & 2(q_x q_z + q_y q_0) \\ 2(q_x q_y + q_z q_0) & (q_0^2 - q_x^2 + q_y^2 - q_z^2) & 2(q_y q_z - q_x q_0) \\ 2(q_z q_x - q_y q_0) & 2(q_z q_y + q_x q_0) & (q_0^2 - q_x^2 - q_y^2 + q_z^2) \end{pmatrix}.$$

To determine the transformation, the mean values of the paired points (centroid vectors) c_m and c_d are subtracted from all points in M and D, respectively, resulting in the sets M' and D'. The rotation expressed as quaternion that minimizes Eq. (1) is the largest eigenvalue of the cross-covariance matrix

$$\mathbf{N} = \begin{pmatrix} (S_{xx} + S_{yy} + S_{zz}) & (S_{yz} + S_{zy}) & (S_{zx} + S_{xz}) & (S_{xy} + S_{yx}) \\ (S_{yz} + S_{zy}) & (S_{xx} - S_{yy} - S_{zz}) & (S_{xy} + S_{yx}) & (S_{zx} + S_{xz}) \\ (S_{zx} + S_{xz}) & (S_{xy} + S_{yz}) & (-S_{xx} + S_{yy} - S_{zz}) & (S_{yz} + S_{zy}) \\ (S_{xy} + S_{yx}) & (S_{yz} + S_{zy}) & (S_{zx} + S_{xz}) & (-S_{xx} - S_{yy} + S_{zz}) \end{pmatrix}$$

with $S_{xx} = \sum_{i=1}^{N_m} \sum_{j=1}^{N_d} w_{i,j} m'_{ix} d'_{jx}$, $S_{xy} = \sum_{i=1}^{N_m} \sum_{j=1}^{N_d} w_{i,j} m'_{ix} d'_{jy}, \ldots$. After the calculation of the rotation $\mathbf{R}$, the translation is $\mathbf{t} = \mathbf{c}_m - \mathbf{R}\mathbf{c}_d$ [18]. Fig. 3 shows three steps of the ICP algorithm.[1]

Table 1
Computing time of the different 3D scan matching implementations for two scans of the GMD Robobench (Fig. 3)[a]

Points used	Time	Number of iterations
All points and brute force search	3 h, 47 min	27
Reduced points and brute force search	3 min, 6 s	25
All points and *k*D-tree	6 s	27
Reduced points and *k*D-tree	<1.4 s	25

[a] The number of all points is 46 336 (181×256) and the number of the reduced points is 4910 in this case.

The time complexity of the algorithm mainly depends on the determination of the closest points (brute force search $O(n^2)$). Several enhancements have been proposed [11,19]. We implement *k*D-trees as proposed by Simon et al. Combined with the above-described *reduced points*. Table 1 summarizes the results of different experiments on a Pentium-III-800. The starting point for optimization is given by the robot odometry.

3.2. Matching multiple 3D scans

To digitalize environments without occlusions, multiple 3D scans have to be registered. After registration, the scene has to be globally consistent. A straightforward method for aligning several 3D scans is *pairwise matching*, i.e., the new scan is registered against the scan with the largest overlapping areas. The latter one is determined in a preprocessing step.

[1] For an animation of this result please refer to the following web site: http://www.ais.fhg.de/ARC/3D/videos.

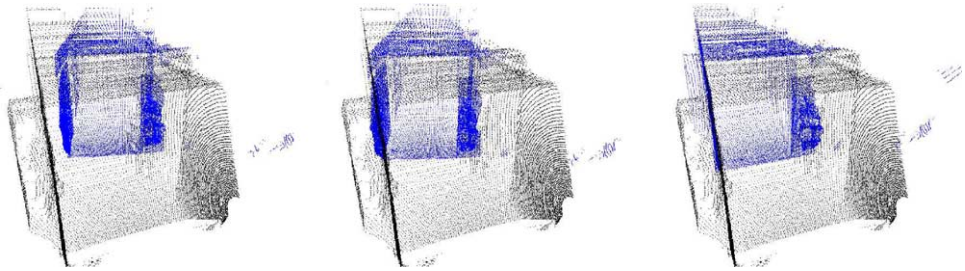

Fig. 3. Registration of two 3D scans with the ICP algorithm – left: initial alignment based on odometry; middle: alignment after five iterations; right: final alignment after 25 iterations. The scans were taken in the GMD Robobench; the first scan corresponds to Fig. 2. The number of sampled 3D points is 46 336 per 3D scan and the number of *reduced points* is 4910.

Alternatively, Chen and Medoni [12] introduced an *incremental matching* method, i.e., the new scan is registered against the so-called *metascan*, which is the union of the previously acquired and registered scans. Each scan matching has a limited precision. Both methods accumulate the registration errors such that the registration of many scans leads to inconsistent scenes (Fig. 4) and to problems with the robot localization.

Pulli [20] presents a registration method that minimizes the global error and avoids inconsistent scenes. This method distributes the global error while the registration of one scan is followed by registration of all neighboring scans. Other matching approaches with global error minimization have been published, e.g., by Benjemaa and Schmitt [21] and Eggert et al. [22].

Based on the idea of Pulli we designed a method called *simultaneous matching*. Thereby, the first scan is the master scan and determines the coordinate system. This scan is fixed. The following steps register all scans and minimize the global error:

(1) Based on the robot odometry, pairwise matching is used to find a start registration for a new scan. This step speeds up computation.
(2) A queue is initialized with the new scan.
(3) Three steps are repeated until the queue is empty:
   (a) The current scan is the first scan of the queue. This scan is removed from the queue.
   (b) If the current scan is not the master scan, then a set of neighbors (set of all scans that overlap with the current scan) is calculated. This set of neighbors forms one point set M. The current scan forms the data point set D and is aligned with the ICP algorithms.
   (c) If the current scan changes its location by applying the transformation (translation or rotation), then each single scan of the set of neighbors that is not in the queue, is added to the end of the queue.

Notes: One scan overlaps with another, iff more than 250 corresponding point pairs exist. To speed up the matching, *k*D-trees and *reduced points* are used (see Table 1).

In contrast to Pulli's approach, the proposed method is totally automatic and no interactive pairwise alignment has to be done. Furthermore the point pairs are not fixed [20]. Fig. 4 shows results of the scan matching using 20 scans taken in the GMD Robobench. Pairwise matching (a) works sufficiently, incremental matching shows most outliers (b), and simultaneous matching (d) reconstructs the corridor perfectly. The matching method works in six dimensions, i.e., the robot position is a vector (x, y, z) as well as its heading $(\theta_x, \theta_y, \theta_z)$. Probabilistic approaches do not scale good with the number of dimensions.

3.3. Feature based matching

Sappa et al. [23] suggest the extraction of edge points and use them for creating point pairs. Based on our line representation of the scene (see Section 2.2.1 and Fig. 2 (bottom left)) the end points of every line are used to create an edge-based representation. Fig. 5 (left) shows an edge-based representation in comparison with the reduced points (middle). Fig. 4(c) shows the result of the registration process with the edge points (pairwise matching). The registration speed is good due to the lower number of points. Unfortunately,

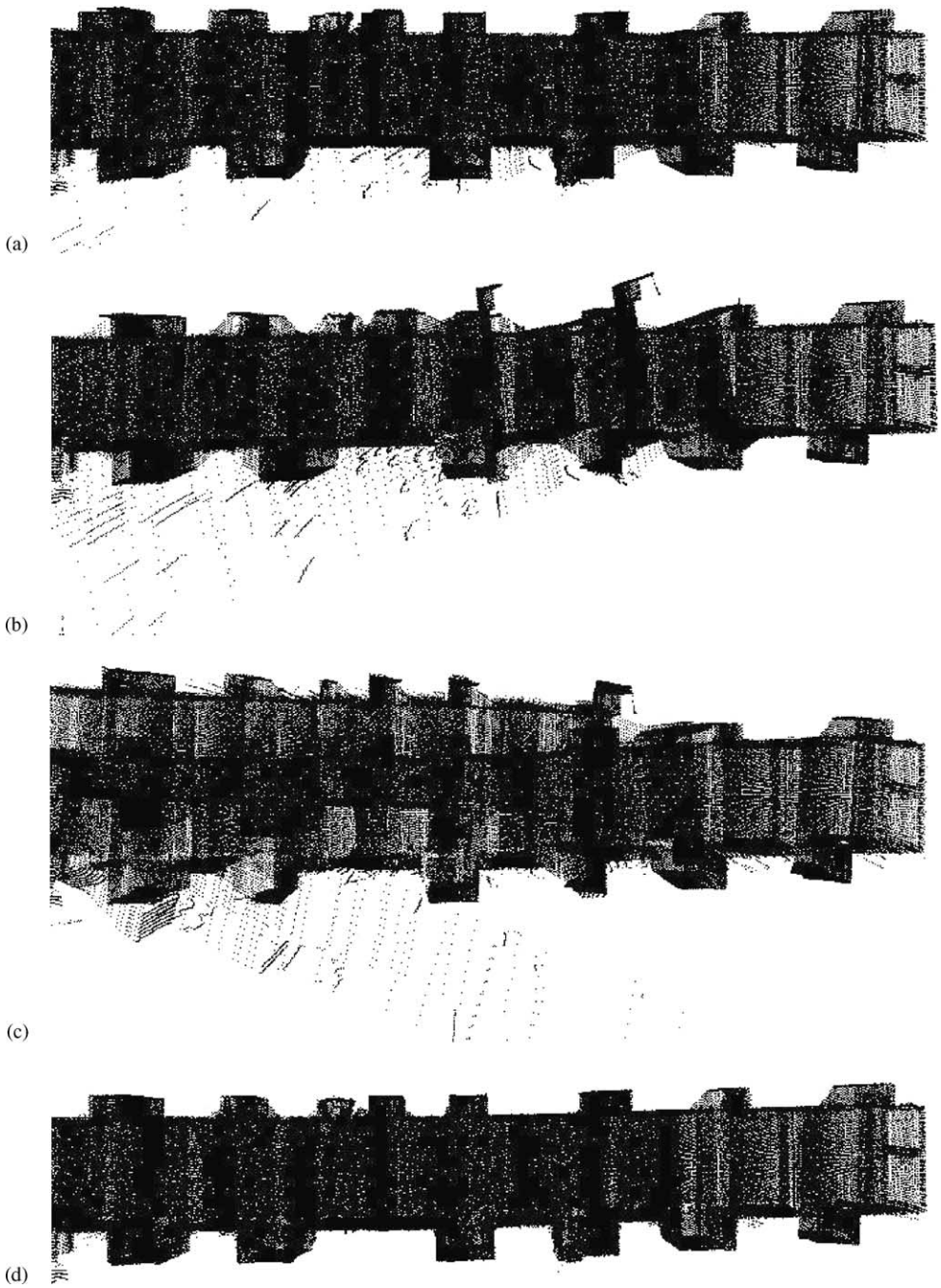

Fig. 4. Results of the scan matching of 20 scans (top view). All 3D scans were taken in the GMD Robobench, the first one corresponds to Fig. 2. (a) Pairwise matching, (b) incremental matching, (c) 3D scan matching with edge points and (d) simultaneous matching.

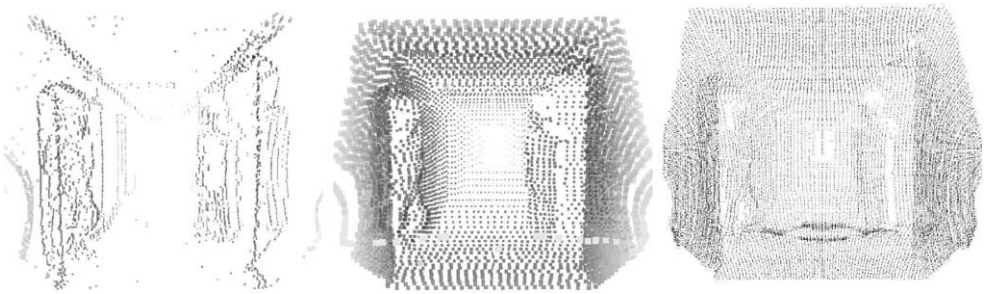

Fig. 5. Left: the scanned scene of the GMD Robobench (see Fig. 2) in an edge-based representation using the simple line detection routine; middle: 4910 *reduced points* (enlarged); right: all points.

the matching results are insufficient for office environments, because of the simple structure of the scanned scene. The office environment (corridors) mainly consists of floor, ceiling and walls.

3.4. Scanning in dynamic environments

Dynamic objects lead to errors in the resulting 3D volumetric model with artifacts or misalignments. Misalignments result in an incorrect self-localization of the mobile robot. To eliminate these errors, the robot monitors the environment with its other sensors, e.g., the horizontal mounted 2D laser range finders or cameras. If the sensors detect dynamic objects with the method of the differential frames, the robot simply repeats the 3D scan. Data points belonging to dynamic objects are not yet identified and removed.

4. The next best view

As mentioned above, multiple 3D scans from different positions are necessary to digitalize environments such that occlusions are resolved. The autonomous robot has to plan and to drive to these positions to efficiently generate a complete model. The next best view is the pose (x, y, θ) which has a high information gain, is accessible by the robot, and the overall robot path is short. Traditionally next best view algorithms assume that the sensor head can freely move around some object [24]. In mobile robotics, the sensor head has lower degrees of freedom, e.g., in our case even a fixed height. The sensor is inside the scene and the accuracy of the pose is limited. Thus Latombe and coworkers conclude that traditional next best view algorithms are not suitable [25,26].

The calculation of viewpoints, that is, the question of where shall we place the 3D sensor to scan the whole building without occlusions, is similar to the art *gallery problem*, i.e., given a map of a building, where guards be placed to watch the whole building [27]. Map building with mobile robots requires a competitive online strategy for finding the locations from which the whole environment is seen. The next part describes an approximation of the art gallery problem and derives an online greedy version based on the algorithm of Gonzalez-Banos et al. [25]. The art gallery is modeled by a horizontal plane (2D map) through the 3D scene. The approach is extended by considering several horizontal planes at different heights to model the whole 3D environment.

4.1. Solving art galleries problems by approximation

Suppose a 2D map of an art gallery is given by a polygon P, having n corners (vertices) and n walls (lines). If a watchman sees everything that lies on a straight line from his position, then the maximal number of guards needed is $\lfloor n/3 \rfloor$ [27]. Finding the minimum number of watchmen needed for this task is NP hard, since it can be reduced to the 3-SAT problem [27]. Gonzalez-Banos et al. reduce the art gallery problem to set cover and approximate the latter one. Set cover is defined as follows: given a pair (X, F), where X is some finite set. F is a subset of the power set of X, i.e., $(F \subset P(X))$ and $X = \bigcup_{s \in F} s$. Find the $C \subset F$, such that $X = \bigcup_{s \in C} C$ and the set C has a minimum number of elements. The reduction of the

art gallery to the set cover problem is randomized and can be stated as follows:

(1) Generate randomly c candidate positions in the interior of the polygon (map).
(2) Calculate for all candidate positions the visible part of the polygon.
(3) Approximate the set cover problem, i.e., find a minimal subset of the c candidate positions from which the entire polygon is visible.

A greedy algorithm approximates the set cover problem fast and with a good approximation ratio [28].

4.2. Planing the next best view

Exploration starts with a blank map, i.e., the environment is unknown. The robot's first action is to acquire a 3D scan. Based on the first scan, an initial ground plane is generated (Fig. 6 (top left)). The Hough transformation is used to detect horizontal lines in the ground plane (Fig. 6 (top middle)). The found lines are transformed into polygons, where the edges are either labeled as *seen* or *unseen* (Fig. 6 (top right)). The seen edges are the detected lines, which are connected by unseen edges. The seen lines have to be sorted for connecting them. The sorting criterion is the

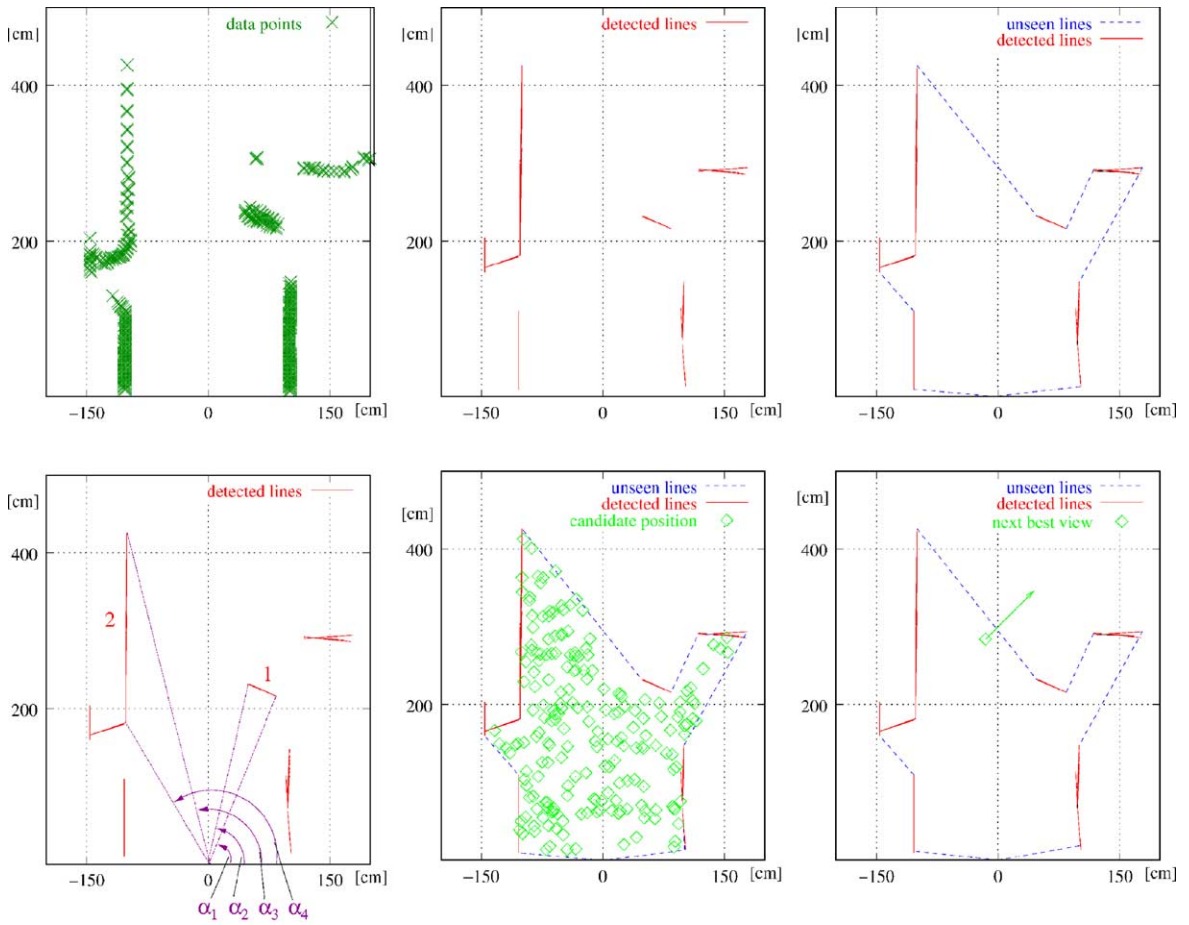

Fig. 6. A horizontal plane of the scene inside the floor of the GMD Robobench (Fig. 2) taken at a height of 150 cm ± 2 cm—top left: the scanned data; top middle: the detected lines; top right: the polygon; bottom left: sorting of the lines is done by using the smallest angle between the lines and the scanner position; bottom middle: random candidate positions within the polygon; bottom right: the next best view pose.

smallest angle between the end points of a line and the scanner position. Fig. 6 (bottom left) shows how to connect line 1 with line 2: $\alpha_2 < \alpha_3$ and no other α exists between them.

The second step of the next best view algorithm generates randomly candidate positions (x, y) within the polygon with equal probability (Fig. 6 (bottom middle)).

The candidate with the highest information gain is selected according to the position to the unseen lines, i.e., the direction of the next measurement has to be towards the unseen lines, since the area behind these lines is unexplored area. For estimating the information gain, a horizontal laser scanner with an apex angle of 180° and a distance range, e.g. [0.5 m, 15 m], dependent on the slice height, is simulated. The number of intersections with the polygon determines the information gain $V(\vec{p})$ and the direction θ_p of the candidate location $\vec{p} = (x, y)^{\mathrm{T}}$:

$$V(\vec{p}) = \max \left\{ \sum_{i=\varphi}^{\varphi+\pi} f(\vec{p}, \varphi) | 0 \le \varphi < 2\pi \right\},$$

$$\theta_p = \operatorname{argmax} \left\{ \sum_{i=\varphi}^{\varphi+\pi} f(\vec{p}, \varphi) | 0 \le \varphi < 2\pi \right\} + \frac{\pi}{2} = \varphi_{\max} + \frac{\pi}{2}$$

with $f(\vec{p}, \varphi) = 1$ if the laser beam hits an unseen edge, $f(\vec{p}, \varphi) = 0$ otherwise. Fig. 6 shows 200 candidate positions inside a polygon (bottom middle) and the result of the evaluation (bottom right).

A view pose (x, y, θ) has three properties:

(1) The evaluated information gain value $V(\vec{p})$ of the candidate position $\vec{p}$.
(2) The distance from the current position.
(3) The angle to the current position.

The next best view pose is an optimum of

$$\hat{V}(\vec{p}) = V(\vec{p}) \exp(-c_1 ||\vec{r} - \vec{p}||) \exp(-c_2 ||\theta_r - \theta_p||),$$

where $\vec{r} = (x, y)^{\mathrm{T}}$ is the current robot position and θ_r the current orientation. The first exponential item prevents the robot from oscillating between distant areas of high information gain. The second exponential part considers rotation and also prevents oscillation. The constants c_1 and c_2 weight the three terms and affect the optimum, e.g., $c_1 = 0.05\,\mathrm{cm}^{-1}$, $c_2 = 0.01$. To plan the next best view in 3D, several positions in different slices are computed and compared. In our experiments, we used one to five slices and it turned out that one slice is sufficient in the majority of cases.

4.3. Extension to multiple 3D scans – map building

The calculation of the next best view from multiple 3D scans is an extension to the method described above. Scan matching is used to align the 3D scans as described in Section 3 to calculate the precise scan position. Polygons are calculated from each single 3D scan and aligned with the precise position. A modified version of Vatti's polygon clipping algorithm [17] is used to merge the polygons. The modification is necessary to ensure the labeling of the edges (seen, unseen), while creating the union of the polygons (Fig. 7). A new acquired 3D scan is clipped against the union of the previous scans.

The performance of the proposed planning module can be estimated as follows: the first step converts data points into lines. The Hough transform runs in $O(d^2)$ (d is the maximal distance of a data point in a single 3D scan). Generating candidate points is being done in $O(n_c n_l)$ and their evaluation is also in $O(n_c n_l)$ (n_c is the number of candidates and n_l the number of lines). Vatti's polygon clipping algorithm runs in $O(n_p)$ (n_p is the sum of edges of both polygons). The whole planning algorithm takes up to 2 s on scenes of 20 m × 30 m, running on a Pentium-III-800.

5. Navigation in 3D

Once the next best view is determined, the robot has to move to that pose. The current pose has to be connected to the target pose by a trajectory, i.e., a continuous path. The robot's wheel encoders are used to calculate and estimate the robot pose. This estimation is the start pose for the scan matching as described in Section 3. The computed transformation for registering the 3D scans are used to correct the prior robot pose estimation.

The non-holonomic robot vehicle is controlled by a closed-loop, time invariant and globally stable motor controller, developed by Indiveri [29]. The target configuration is always approached on a continuous

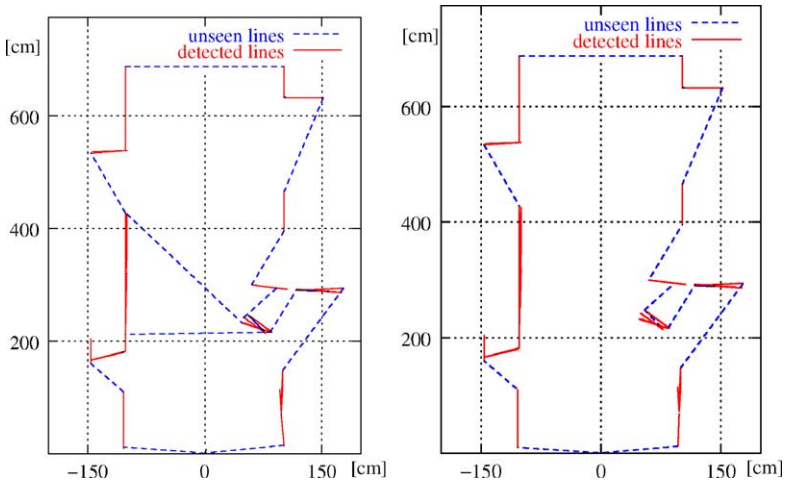

Fig. 7. Left: two polygons with edge labels prior merging; right: their union.

trajectory that ends in a straight line. The vehicle is controlled to move in only one specified forward direction, thus avoiding cusps in the paths and satisfying a major requirement for the implementation on non-holonomic vehicles. This section summarizes the controller. Details can be found in [29].

5.1. Computing the nominal trajectory

Let (x_G, y_G, ϕ) be the robot pose in the target centered coordinate system (Eq. (2)). The controller is based on a Cartesian kinematic model described by

$$\dot{x}_G = u\cos\phi, \qquad \dot{y}_G = u\sin\phi, \qquad \dot{\phi} = \omega = uc. \tag{2}$$

(0, 0, 0) is the final position and corresponds to the next best view. u is the robot's linear velocity, ω the angular velocity and c the (bounded) curvature. The transformation of the Cartesian coordinates into polar like coordinates results in (refer to the parameter as in Fig. 8)

$$\begin{aligned} e &= \sqrt{x_G^2 + y_G^2} & \dot{e} &= -u\cos\alpha \\ \theta &= \mathrm{ATAN2}(-y_G, -x_G) \quad \Rightarrow & \dot{\alpha} &= u\left(c - \frac{\sin\alpha}{e}\right). \\ \alpha &= \theta - \phi & \dot{\phi} &= u\frac{\sin\alpha}{e} \end{aligned} \tag{3}$$

A Lyapunov-like based control law synthesis results in

$$c = \frac{\sin\alpha}{e} + h\frac{\theta}{e}\frac{\sin\alpha}{\alpha} + \beta\frac{\alpha}{e} \quad \text{with } h > 1,\ 2 < \beta < h + 1, \tag{4}$$

using the quadratic Lyapunov function $V = (1/2)(\alpha^2 + h\theta^2)$ $(h > 0)$ and $u = \gamma e$ with $\gamma > 0$ for the velocity. The control law for u is motivated through the state

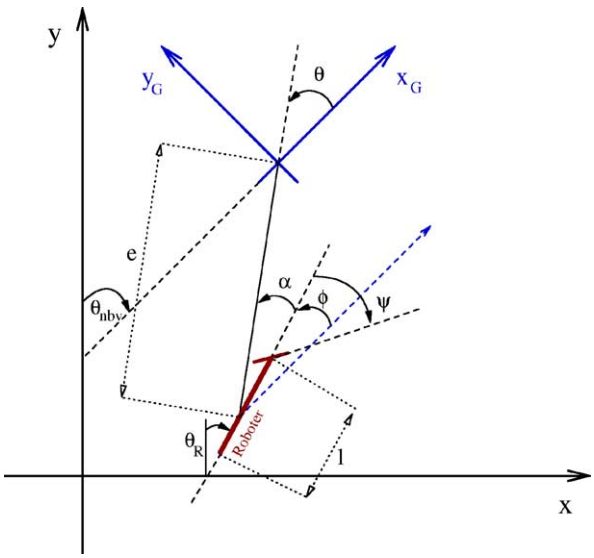

Fig. 8. The local Cartesian kinematic model (x_G, y_G, ϕ) embedded into a global coordinate system.

derivative of formula (3), that is 0, if $u = 0$, and through avoiding sign changes in the linear velocity.

Every vehicle has a maximum velocity $\bar{u}$. Therefore, the calculation of the velocity has to be saturated, i.e.,

$$u = \gamma e \,\mathrm{sat}(\gamma, e, \bar{u}) \quad \text{with } \gamma > 0,\ \mathrm{sat}(x, y) = \begin{cases} 1 & \forall x < y \,\forall x, y > 0, \\ \dfrac{y}{x} & \forall x \geq y \,\forall x, y > 0. \end{cases}$$

Indiveri [29] also proves closed-loop stability for the saturated case. Thus formula (4) guarantees that the robot pose (e, α, θ) converges to (0, 0, 0) with bounded curvature c. Fig. 9 shows different paths computed with the controller. Starting with the linear velocity u and the curvature c, values for the angular velocity ω and left resp. right motors are calculated.

5.2. Collision avoidance with arbitrarily formed objects

One major issue in mobile robotics is obstacle avoidance. Objects with jutting out edges, e.g., tables, are often not detected by standard sensors, and the mobile robots may hit these objects. The 3D laser scanner software as described in Section 2.2.1 (Fig. 2) computes bounding boxes around obstacles in a 3D scan. These bounding boxes are joined with a planned trajectory. The trajectory is calculated with the motor controller described above and the physical motion model of the mobile robot (Fig. 10). If no collision is detected the robot uses the motor controller to reach the target pose.

If a collision is detected, two simple robot behaviors are simulated to generate collision free paths. If these two additional trajectories fail, the path planning module restarts with next best target pose. The two simulated behaviors are:

(1) Turning the robot towards the target position and calculating the trajectory with the motor controller to the target pose. This behavior avoids large turning radii (Fig. 9).
(2) Turning the robot towards the target position and at the target position. Thus driving on a straight path is simulated.

The performance of the algorithm is $O(n_T \cdot n_O)$ (n_T is the number of points on the trajectory and n_O the number of objects in the scene, e.g., n_T is 10 values per meter and n_O is 10 objects per 3D scan).

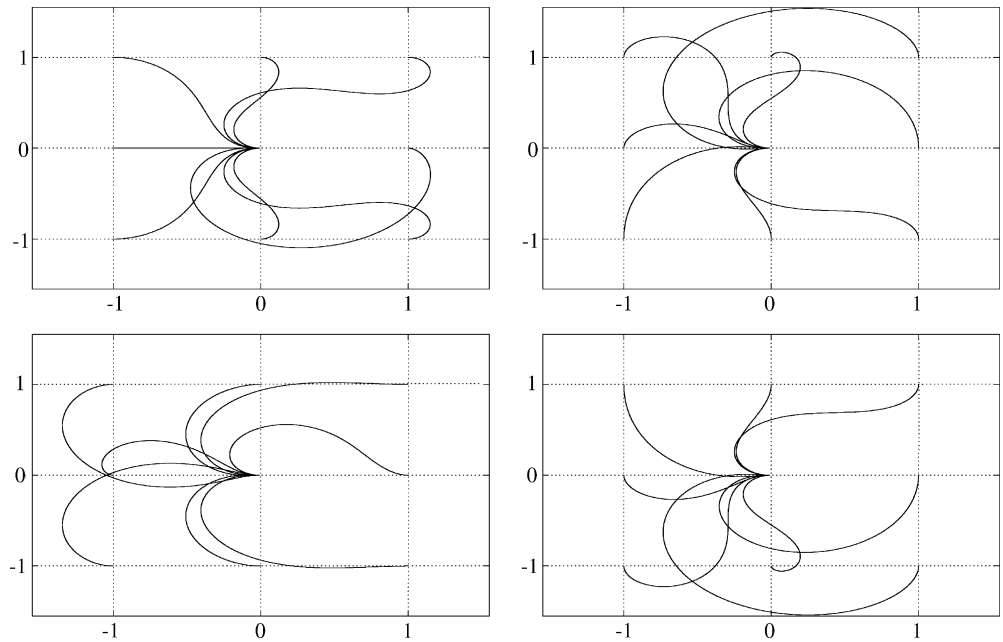

Fig. 9. Robot paths. The goal (0, 0, 0) is reached for each starting configuration. Used constants: $\gamma = 1$, $h = 2$ and $\beta = 2.9$.

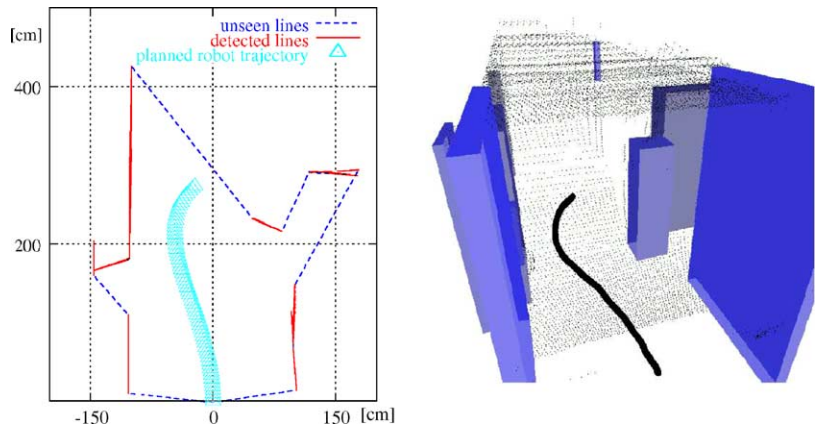

Fig. 10. Trajectory to the next best view pose—left: trajectory inside the polygon; right: trajectory around bounding boxes of objects.

5.3. Dynamic collision avoidance

The digitalization process is a stop, scan, plan and go setting. To drive around moving objects, e.g., humans, the planned trajectory is extended by a dynamic collision avoidance method, which is active during the driving to the target pose. The dynamic collision avoidance uses the 3D laser scanner in a

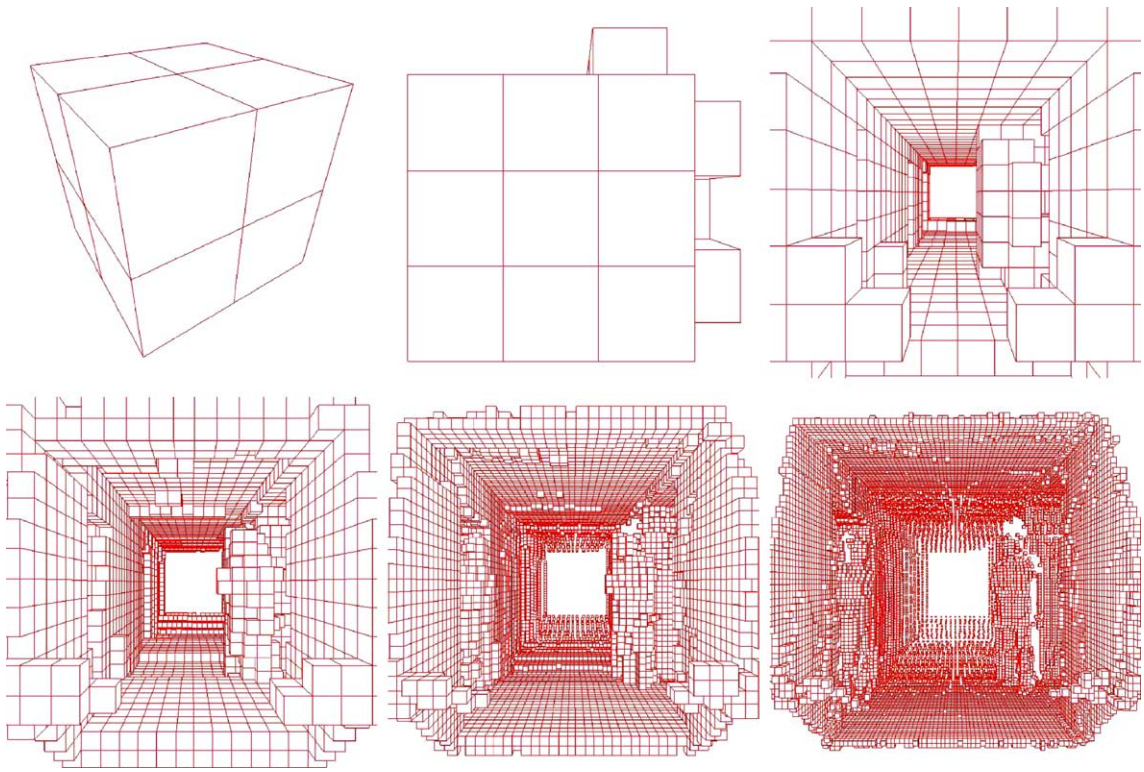

Fig. 11. Creation of a surface mesh using an octree. The octree of depth 1 (top left), 3 and 5–8 (bottom right) are plotted.

horizontal scanning mode and possibly additional sensors, e.g., the 2D safety laser range finder mounted at the bottom of the robot (Fig. 1 (left)), to detect dynamic obstacles. A vector pointing away from the obstacle is merged with the calculated angular velocity ω and the linear velocity u is reduced according to the distance to the obstacle. Since the motor controller is globally stable, the target is always reached by a smooth path (Fig. 9), if the dynamic obstacle has disappeared.

6. Results

After acquiring and merging several 3D scans from different robot locations, a further step is the output of the 3D volumetric model. The data is exported as a 2D point and line map as well as a 3D volumetric model in DXF and VRML format. Furthermore the data is exported as a 3D grid for an OpenGL based viewer application [15].

6.1. Visualization

An OpenGL-based application is used for projecting the 3D scene to a 2D image. Many methods have been proposed for generating triangle meshes from sorted [30,31] and unsorted 3D data [32–34]. Besides triangulation, Pulli et al. [33] present an octree-based approach for creating a mesh for the scene. An octree is a tree with up to eight successors, only the leaves of the octree contain the data. The first step of building an octree is to create a bounding box around all points. The axis of this cuboid are aligned along the axis of the coordinate system. The cuboid can be divided into eight equal sized smaller ones by introducing three

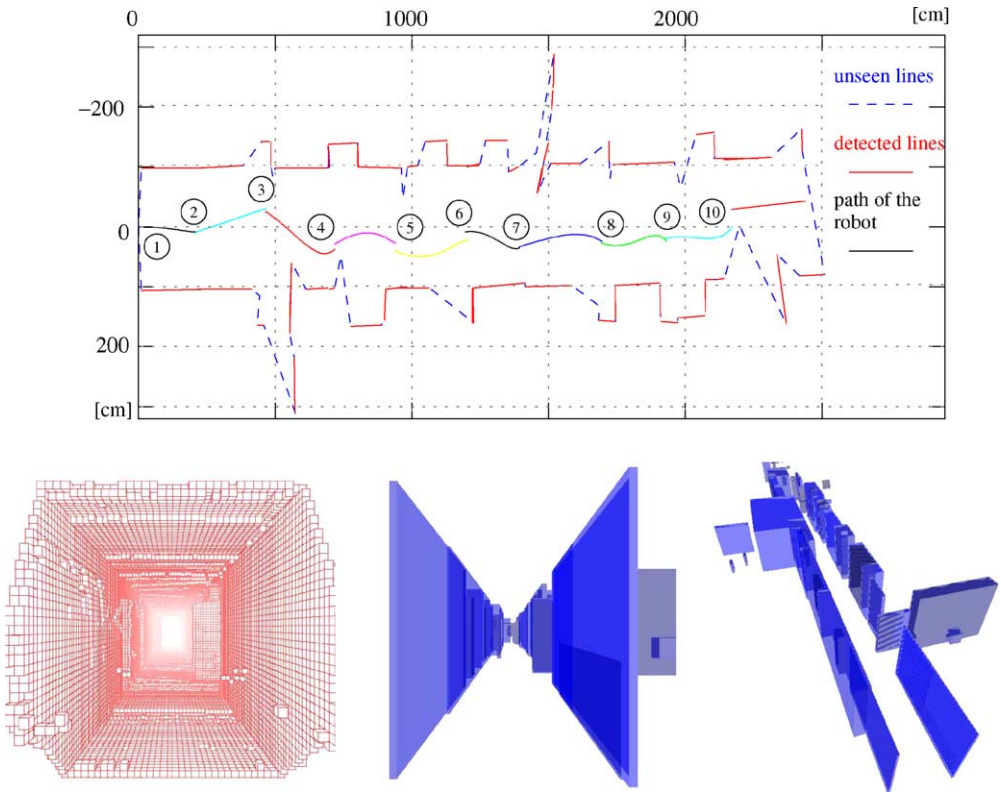

Fig. 12. Top: exploration path of the robot in a corridor. The floor was scanned at 10 different positions, marked with numbers; bottom left: view of the scanned scene as a mesh; bottom middle and right: object representation with bounding boxes.

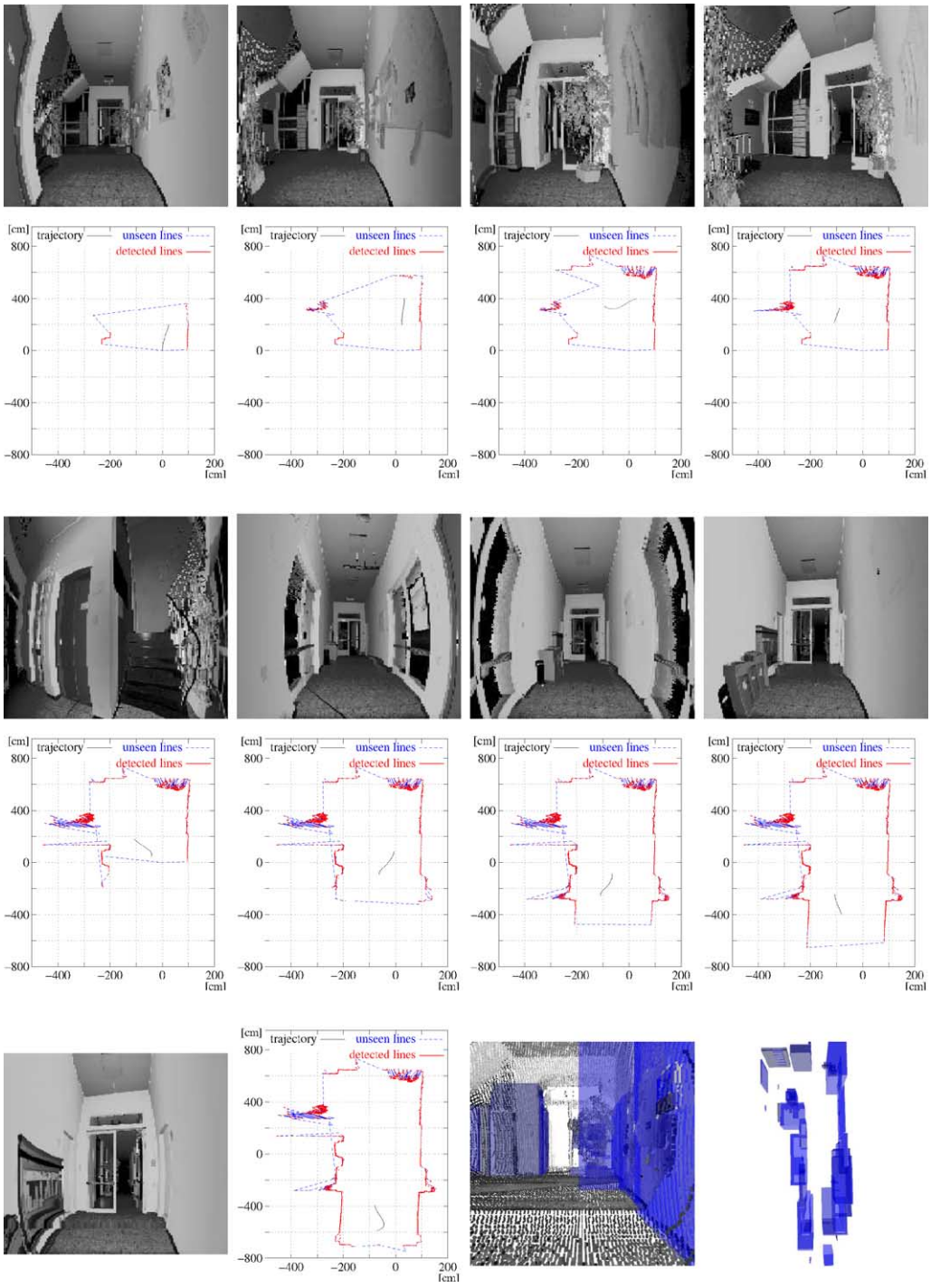

Fig. 13. The first nine scans and exploration path in an entrance hall with a staircase, elevators, flowers and a garbage can. The distorted reflectance images and polygons for the planning module, including a trajectory, are shown. The last two images show the colored bounding boxes of objects in this scene.

planes. The data is distributed to these child nodes. If a node is empty, the tree is pruned. Since only a few iterations are needed even for very large scenes, creating an octree is very fast (O(n)) and can be done in a fraction of a second. Recursion builds up the whole tree and the leaf nodes approximate the surface. Fig. 11 shows the creation of the mesh using an octree, Fig. 2 (top, right) shows the octree with colored voxels.

6.2. 3D digitalization

Five steps plus one global post-processing step are necessary to create a *complete* digital volumetric 3D model:

(I) *Data acquisition*. During the gaging of the range image the surface of parts of the scene is scanned.
(II) *Registration*. Each range image is registered in a common coordinate system. The odometry-based robot pose serves as a first estimate and is corrected by registration.
(III) *Planning*. The next best view is calculated and a collision free path is planned in 3D to that pose. Hereby, the information gain is optimized considering the cost of travel and turning.
(IV) *Robot control*. The robot moves to the target pose using the path computed in the previous step including dynamic collision avoidance.
(V) *Iteration*. The process continues with the next data acquisition until the complete environments is digitalized.
(VI) *Integration*. The acquired data are post-processed, e.g., 3D meshes are generated and stored.

The following two examples illustrate the performance of the proposed method. The first example demonstrates the approach in a simple office environment, with a long corridor (GMD Robobench) (Fig. 12) (see footnote 1). The second example shows a typical entrance hall with a staircase, elevators, flowers and garbage can (Fig. 13) (see footnote 1). The figure shows the 3D octree representation, including reflectance values, and the 2D planning map including the trajectory to the next pose.

7. Conclusion

This paper has presented an automatic system for gaging and digitalization of 3D indoor environments without any human intervention. An autonomous mobile robot together with a 3D laser range finder and three software modules form the system:

- The AIS 3D laser range finder acquires 3D scans at given poses, and a fast variant of the ICP algorithm registers the 3D scans in a common coordinate system and relocalizes the robot. The proposed method solves the simultaneous localization and map building problem (SLAM problem). The kidnapped robot problem, i.e., robot self-localization by a given map and a random start position, is not addressed in this paper. Nevertheless, the proposed method can be extended to handle the problem, e.g., by matching a 3D scan at randomly selected position in the 3D volumetric representation and the evaluation of the error function.
- The next best view planner computes next nominal poses under the conditions of:
  - maximizing the information gain,
  - reducing the overall robot path length,
  - minimizing the rotation angles, and
  - avoiding complicated obstacles, e.g., with jutting out edges.

  Instead of using grid-based approaches the planning module utilizes polygons in a continuous state space.
- A closed-loop, globally stable motor controller navigates the mobile robot to a nominal pose regarding dynamical obstacles.

The results of two 3D digitalization experiments were presented using a fast octree-based visualization method.

3D laser range finders on mobile robots enable the automatic acquisition, surveying and mapping of entire interiors. The potential of our approach lies in the areas of site survey, structural engineering, building restoration, open pit and underground mining, deformation monitoring, inspection and rescue robotics.

Future work will concentrate on four further aspects:

- Improving object classification while fusing camera information and an object database.

- Building semantic maps from the digitalized environment, i.e., a high-level description of the scanned scene. Some results are presented at the VMV2003 [35].
- Combining the presented registration methods with stochastic localization algorithms [1]. This would combine the precision of the ICP based methods with the flexibility of the localization [36].
- Using a 3D time of flight camera instead of a 3D laser range finder.

Acknowledgements

Special thanks to Kai Lingemann for creating the surface and object detection algorithms and to Stefan Materne for implementing the VRML output during the project. Many thanks to the unknown reviews for hints and corrections.

References

[1] S. Thrun, D. Fox, W. Burgard, A real-time algorithm for mobile robot mapping with application to multi-robot and 3D mapping, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'00), San Francisco, CA, April 2000.

[2] D. Hähnel, W. Burgard, S. Thrun, Learning compact 3D models of indoor and outdoor environments with a mobile robot, in: Proceedings of the Fourth European Workshop on Advanced Mobile Robots (EUROBOT'01), Lund, Sweden, September 2001.

[3] C. Früh, A. Zakhor, 3D model generation for cities using aerial photographs and ground level laser scans, in: Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR'01), Kauai, Hawaii, December 2001.

[4] H. Zhao, R. Shibasaki, Reconstructing textured CAD model of urban environment using vehicle-borne laser range scanners and line cameras, in: Proceedings of the Second International Workshop on Computer Vision System (ICVS'01), Vancouver, Canada, July 2001, pp. 284–295.

[5] The ACTS RESOLV (REconstruction using Scanned Laser and Video) Project, 1999. http://www.comp.leeds.ac.uk/resolv/.

[6] V. Sequeira, J. Goncalves, M. Ribeiro, 3D environment modelling using laser range sensing, Robotics and Automation 16 (1995) 81–91.

[7] V. Sequeira, E.W.K. Ng, J. Goncalves, D. Hogg, Automated 3D reconstruction of interiors with multiple scan-views, in: Proceedings of the SPIEs 11th Annual Symposium on Electronic Imaging'99, The Society for Imaging Science and Technology, San Jose, CA, January 1999.

[8] AVENUE (autonomous vehicle for exploration and navigation in urban environments) Project, 2001. http://www.cs.olumbia.edu/robotics/projects/avenue/.

[9] P. Allen, I. Stamos, A. Gueorguiev, E. Gold, P. Blaer, AVENUE: automated site modelling in urban environments, in: Proceedings of the Third International Conference on 3D Digital Imaging and Modeling (3DIM'01), Quebec City, Canada, May 2001.

[10] A. Gueorguiev, P. Allen, E. Gold, P. Blaer, Design, architecture and control of a mobile site-modelling robot, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'00), San Francisco, CA, April 2000.

[11] P. Besl, N. McKay, A method for registration of 3D shapes, IEEE Transactions on Pattern Analysis and Machine Intelligence 14 (1992) 239–256.

[12] Y. Chen, G. Medoni, Object modelling by registration of multiple range images, in: Proceedings of the IEEE Conference on Robotics and Automation (ICRA'91), Sacramento, CA, April 1991, pp. 2724–2729.

[13] Z. Zhang, Iterative point matching for registration of free-form curves, Technical Report No. RR-1658, INRIA, Sophia Antipolis, Valbonne Cedex, France, 1992.

[14] P. Allen, M. Reed, I. Stamos, View planning for site modeling, in: Proceedings of the DARPA Image Understanding Workshop, Monterey, CA, November 1998.

[15] H. Surmann, K. Lingemann, A. Nüchter, J. Hertzberg, A 3D laser range finder for autonomous mobile robots, in: Proceedings of the 32nd International Symposium on Robotics (ISR'01), Seoul, Korea, April 2001, pp. 153–158.

[16] H. Surmann, K. Lingemann, A. Nüchter, J. Hertzberg, Fast acquiring and analysis of three-dimensional laser range data, in: Proceedings of the Sixth International Fall Workshop Vision, Modeling, and Visualization (VMV'01), Stuttgart, Germany, November 2001, pp. 59–66.

[17] B. Vatti, A generic solution to polygon clipping, Communications of the ACM 35 (7) (1992) 56–63.

[18] B. Horn, Closed-form solution of absolute orientation using unit quaternions, Journal of the Optical Society of America A 4 (1987) 629–642.

[19] D. Simon, M. Hebert, T. Kanade, Real-time 3D pose estimation using a high-speed range sensor, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'94), vol. 3, San Diego, CA, May 1994, pp. 2235–2241.

[20] K. Pulli, Multiview registration for large data sets, in: Proceedings of the Second International Conference on 3D Digital Imaging and Modeling (3DIM'99), Ottawa, Canada, October 1999, pp. 160–168.

[21] R. Benjemaa, F. Schmitt, Fast global registration of 3D sampled surfaces using a multi-Z-buffer technique, in: Proceedings of the IEEE International Conference on Recent Advances in 3D Digital Imaging and Modeling (3DIM'97), Ottawa, Canada, May 1997.

[22] D. Eggert, A. Fitzgibbon, R. Fisher, Simultaneous registration of multiple range views satisfying global consistency constraints for use in reverse engineering, Computer Vision and Image Understanding 69 (1998) 253–272.

[23] A. Sappa, A. Restrepo-Specht, M. Devy, Range image registration by using an edge-based representation, in: Proceedings of the Ninth International Symposium on Intelligent Robotic Systems (SIRS'01), Toulouse, France, July 2001.
[24] J. Banta, Y. Zhieng, X. Wang, G. Zhang, M. Smith, M. Abidi, A best-next-view, algorithm for three-dimensional scene reconstruction using range images, in: D. Casasent (Ed.), Proceedings of the SPIE on Intelligent Robots and Computer Vision. XIV. Algorithms, vol. 2588, 1995, pp. 418–429.
[25] H. Gonzalez-Banos, E. Mao, J. Latombe, T. Murali, A. Efrat, Planning robot motion strategies for efficient model construction, in: J. Hollerbach, D. Koditschek (Eds.), Proceedings of the Ninth International Symposium on Robotics Research (ISRR'99), Springer, Berlin, 2000, pp. 345–352.
[26] H. Gonzalez-Banos, J. Latombe, A randomized art-gallery algorithm for sensor placement, in: Proceedings of the ACM Symposium on Computational Geometry (SoCG'01), Medford, MA, June 2001.
[27] J. O'Rourke, Art Gallery Theorems and Algorithms, Oxford University, Oxford, 1987.
[28] T. Cormen, C. Leiserson, R. Rivest, Introduction to Algorithms, McGraw-Hill, New York, 1997.
[29] G. Indiveri, Kinematic time-invariant control of a 2D nonholonomic vehicle, in: Proceedings of the 38th Conference on Decision and Control (CDC'99), Phoenix, AZ, December 1999.
[30] G. Turk, M. Levoy, Zippered polygon meshes from range images, in: Proceedings of the ACM SIGGRAPH'94, Orlando, FL, July 1994, pp. 311–318.
[31] B. Curless, M. Levoy, A volumetric method for building complex models from range images, Computer Graphics 30 (Annual Conference Series) (1996) 303–312.
[32] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, W. Stuetzle, Surface reconstruction from unorganized points, Computer Graphics 26 (2) (1992) 71–78.
[33] K. Pulli, T. Duchamp, H. Hoppe, J. McDonald, L. Shapiro, W. Stuetzle, Robust meshes from multiple range maps, in: Proceedings of the IEEE International Conference on Recent Advances in 3D Digital Imaging and Modeling (3DIM'97), Ottawa, Canada, May 1997.
[34] N. Amenta, S. Choi, R. Kolluri, The power crust, Communications of the ACM 35 (7) (2001) 56–63.
[35] A. Nüchter, H. Surmann, K. Lingemann, J. Hertzberg, Semantic scene analysis of scanned 3D indoor environments, in: Proceedings of the Eighth International Fall Workshop on Vision, Modeling, and Visualization (VMV'03), Stuttgart, Germany, in press.
[36] M. Hebert, M. Deans, D. Huber, B. Nabbe, N. Vandapel, Progress in 3D mapping and localization, in: Proceedings of the Ninth International Symposium on Intelligent Robotic Systems (SIRS'01), Toulouse, France, July 2001.

Hartmut Surmann heads a research group within the Fraunhofer Institute for Autonomous Intelligent Systems (AIS). He received his Diploma in computer science from the Department of Computer Science, University of Dortmund and his Ph.D. in electrical engineering from the Faculty of Electrical Engineering, University of Dortmund, Germany, in 1989 and 1994, respectively. His primary research interests are autonomous robotics and computational intelligence. He has published several papers in the area of design, architecture, hardware and application of robotics and fuzzy logic controllers. He is a member of the GI and VDE and received several awards, e.g., the FUZZ-IEEE/IFES'95 robot intelligence award and the Ph.D. award for his thesis from the German AI institutes in 1996. He gives lectures in control theory and robotics at FH Bonn-Rhein-Sieg.

Andreas Nüchter is a Ph.D. student at the Fraunhofer Institute for Autonomous Intelligent Systems (AIS). He received his Diploma degree in computer science from University of Bonn in 2002 and was awarded with the best paper award by the German Society of Informatics (GI) for his thesis. His research interests include reliable robot control, 3D vision, and laser scanning technologies. He is a member of the GI and the IEEE.

Joachim Hertzberg heads the Robot Control Architectures (ARC) Department within the Fraunhofer Institute for Autonomous Intelligent Systems (AIS). He holds a Diploma (1982) and a Doctorate degree (1986) in computer science, both from the University of Bonn. Past affiliations and stays were with the International Computer Science Institute (ICSI), Berkeley, CA, and with the Universities of Bonn (Germany), Dortmund (Germany), and Auckland (New Zealand). He is permanently affiliated with the Computer Science Department of Bonn University as an External Lecturer ("Privatdozent"). His area of research is in artificial intelligence, in particular robotics and planning. He is currently a member of the Executive Council of the International Conference on Automated Planning and Scheduling (ICAPS) and a member of the Comité de Pilotage (Steering Committee) of the Interdisciplinary National French Program in Robotics Robotique et Entités Artificielles (Robea).