Semantic Scene Analysis of Scanned 3D Indoor Environments.

Semantic Scene Analysis of Scanned 3D Indoor Environments

Andreas Nüchter, Hartmut Surmann, Kai Lingemann, and Joachim Hertzberg

Fraunhofer Institute for Autonomous Intelligent Systems (AIS)
Schloss Birlinghoven, D-53754 Sankt Augustin, Germany
{andreas.nuechter|hartmut.surmann|kai.lingemann|joachim.hertzberg}@ais.fraunhofer.de

Abstract

Precise digital 3D models of indoor environments are needed in several applications, e.g., facility management, architecture, rescue and inspection robotics. This paper presents a new method that transforms a 3D volumetric model, acquired by a mobile robot equipped with a 3D laser scanner, into a very precise compact 3D map and generates semantic descriptions. The scanned 3D scene is matched against a coarse semantic description of general indoor environments. The matching is done by a Prolog program compiled from the scanned 3D scene and combined with clauses from the coarse semantic description. The generated scene specific knowledge produced by the unification in the Prolog program is used to refine the 3D model.

1 Introduction

Automatic and precise reconstruction of indoor environments is an important task in architecture and robotics. Autonomous mobile robots equipped with 3D laser range finders are well suited for gaging the 3D data. Due to odometry errors the self localization of the robot is an unprecise measurement and therefore can only be used as a starting point for registration of the 3D scans in a common coordinate system. Furthermore the merging of the views as well as the scanning process itself is noisy and small errors may occur. We overcome these problems by extending the reconstruction process with a new knowledge based approach for the automatic model refinement.

Since architectural shapes of environments follow standard conventions arising from tradition or utility [6] we exploit knowledge for reconstruction of indoor environments. The used knowledge describes general attributes of the domain, i.e., architectural features as plane walls, ceilings and floors. For various domains different knowledge is needed, e.g., for reverse engineering of CAD parts [17]. We show that applying general knowledge for recovering specific knowledge improves reverse engineering.

This paper presents algorithms for building compact and precise 3D models and extends our work in [12]. The proposed algorithm consists of three steps: First we extract features, i.e., planes from registered unmeshed range data. The planes are found by an algorithm which is a mixture of the RANSAC (Random Sample Consensus) algorithm and the ICP (Iterative Closest Point) algorithm [1, 3]. Second the computed planes are labeled based on their relative orientation. A predefined semantic net implementing general knowledge about indoor environments is employed to define these orientations. The semantic net is externalized through a representation by a set of Horn clauses. 3D analysis of the previously found planes compiles additional clauses. Prolog's unification and backtracking algorithms are used to derive a scene specific interpretation from the general knowledge. Finally architectural constraints like parallelism and orthogonality are enforced with respect to the gaged 3D data by numerical methods to refine the 3D model. To this end, two minimization algorithms are compared: Powell's method and the downhill simplex method.

The paper is organized as follows. After discussing the state of the art in the following part we present the 3D laser range finder that is mounted on an autonomous mobile robot. Then we start to describe algorithms for the 3D model based analysis and scene refinement. These algorithms run after data acquisition. The second section presents the feature extraction algorithm. The algorithms for semantic interpretation of the data is given in section three. In section 4 the model refinement is described. Section 5 concludes the paper.

1.1 Related Work

Automatic and autonomous reconstruction of environments has received much attention for several years. Some groups have attempted to build 3D volumetric representations of environments with 2D laser range finders [8, 11]. 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. 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.

A few other groups use 3D laser scanners [2, 9, 19]. 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 tele presence [19]. They used a RIEGL laser range finder on robots and the ICP algorithm for scan matching [3]. The AVENUE project develops a robot for modeling urban environments [2], using a CYRAX laser scanner. The research group of M. Hebert reconstruct environments using the Zoller+Fröhlich laser scanner and aim to build 3D models without initial position estimates, i.e., without odometry information [9].

1.2 The AIS 3D Laser Range Finder

The AIS 3D laser range finder [20] 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. A standard servo is connected on the left side (figure 1) [20].

The area of $180°$(h) $\times$ $120°$(v) is scanned with different horizontal (181, 361, 721) and vertical (128, 256, 512) 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 $\times$ 256 data points needs 3.4 seconds. In addition to the distance measurement the AIS 3D laser range finder is capable of quantifying the amount of light returning to the scanner.

1.3 The Autonomous Mobile Robot

KURT2 (figure 1) is a mobile robot platform with a size of 45 cm (length) $\times$ 33 cm (width) $\times$ 26 cm (height) and a weight of 15.6 kg. Equipped with the 3D laser range finder the height increases to 47 cm and weight increases to 22.6 kg. KURT2's maximum velocity is 5.2 m/s (autonomously controlled 4.0 m/s). Two 90W motors are used to power the 6 wheels, whereas the front and rear wheels have no tread pattern to enhance rotating. KURT2 operates for about 4 hours with one battery (28 NiMH cells, capacity: 4500 mAh) charge. The core of the robot is a Pentium-III-600 MHz with 384 MB RAM running Linux. An embedded 16-Bit CMOS microcontroller is used to control the motor.[1].

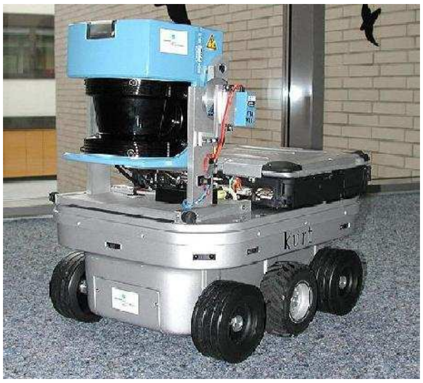


Figure 1: The robot platform KURT2 equipped with the AIS 3D laser range finder.

1.4 Range Image Registration

We use the well-known Iterative Closest Points (ICP) algorithm to calculate a rough approximation of the transformation while the robot is acquiring the 3D scans. 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 the equation

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

where N_m and N_d, are the number of points in the model set M or data set D, respectively and w_{ji} are the weights for a point match. The weights are

[1]Videos of the exploration with the autonomous mobile robot can be found at `http://www.ais.fhg.de/ARC/kurt3D/index.html` and `http://www.ais.fhg.de/ARC/3D/scanner/cdvideos.html`

assigned as follows: $w_{ji} = 1$, if $\mathbf{m}_i$ is the closest point to $\mathbf{d}_j$ within a close limit, $w_{ji} = 0$ otherwise.

It is shown that the iteration terminates in a minimum [3]. 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 [10].

To digitalize environments without occlusions, multiple 3D scans have to be registered. After registration, the scene has to be globally consistent. We designed a method called *simultaneous matching* to generate overall consistent scenes by minimizing the global error [12]. The computed transformations are applied to the robot pose and thus a relocalization of the robot is done after every 3D scan. Therefore the *simultaneous localization and mapping problem (SLAM)* is solved.

2 Feature Detection

A common technique for plane extraction is the region growing based approach, e.g., used by Hähnel et al. [8]. Starting from an initial mesh, neighbored planar triangles are merged iteratively. The drawback of this approach is the high computational demand.

Another well known algorithm for feature extraction from data sets is the RANSAC algorithm [1], used by Cantzler et al. [5]. RANSAC (Random Sample Consensus) is a simple algorithm for robust fitting of models in the presence of many data outliers. RANSAC first selects N data items randomly and uses them to estimate the parameters of the plane. The next step computes the number of data points fitting the model based on a user given tolerance. RANSAC accepts the fit, if the computed number exceeds a certain limit. Otherwise the algorithm iterates with other points [1].

Liu et al. proposes another technique for plane extraction from range data. They use expectation maximization (EM) for generating a surface model [11]. Their algorithm adjusts the number of planes and estimates the location and orientation, by maximizing the expectation of a logarithmic likelihood function. Plane parameters are efficiently calculated by reducing the problem to a computation of eigenvalues by introducing Lagrange multipliers. This approach is not inherently able to determine the number of planes in the data set [8].

Our algorithm is a mixture of the RANSAC and the ICP algorithm, and provides fast plane extraction for a point cloud. No prior meshing algorithms need to be applied. A plane p is defined by three 3D points ($\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3 \in \mathbb{R}^3$) or by one 3D point and the surface normal ($\mathbf{p}_1$, $\mathbf{n}$ with $||\mathbf{n}|| = 1$, $\mathbf{p}_1, \mathbf{n} \in \mathbb{R}^3$). To detect a surface, the algorithm randomly selects a point and estimates a plane through two neighbored data points. Now the data points $\mathbf{x} \in \mathbb{R}^3$ are calculated that fulfill:

$$|(\mathbf{x} - \mathbf{p}_1) \cdot \mathbf{n}| < \epsilon. \tag{2}$$

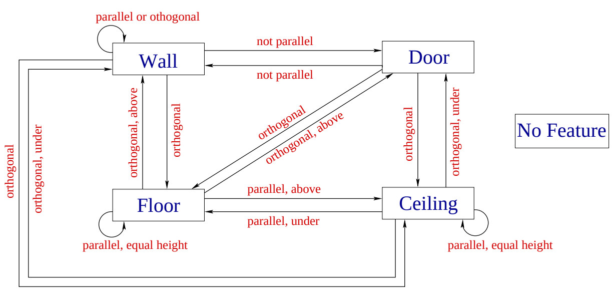
If this set of points exceeds a limit, e.g., 50 points, an ICP based optimization is started. All data points satisfying eq. (2) form the model set M and are projected to the plane to form the data set D for each iteration of the ICP algorithm. Minimizing the ICP error function (1) by transforming the plane with this point-to-plane metric takes only a few iterations. The time consuming search is replaced by direct calculation of the closest point and the transformation $(\mathbf{R}, \mathbf{t})$ is efficiently calculated [10]. Given the best fit, all plane points are marked and subtracted from the original data set. The algorithm terminates after all points have been tested according to eq. (2).

The extracted 3D planes are unbounded in size. Surfaces are finally extracted from the points by mapping them onto the planes. A quadtree based method generates the surfaces. Figure 4 shows an example with 7 extracted planes of a single 3D scan containing 58680 range data points.

3 Semantic Scene Interpretation

The scene interpretation uses the features, i.e., planes previously computed. The background for interpretation comprises generic architectural knowledge. A model of an indoor scene is implemented as a semantic net based on the idea of Grau et al. [7] that is also used by Cantzler et al. [5].

Nodes of a semantic net represent entities of the world. The relationship between the entities are encoded using different connections. Possible labels of the nodes are L = {`Wall`, `Floor`, `Ceiling`, `Door`, `No Feature`}. The relationships between the features are R = {`parallel`, `orthogonal`, `above`, `under`, `equalheight`}. The labels `above` and `under` are relative to their plane and hence not commutative. Figure 2 left shows the



```
parallel(floor,floor).
parallel(ceiling,floor).
parallel(ceiling,ceiling).
parallel(floor,ceiling).
parallel(wall,wall).
parallel(X,_) :- X == nofeature.
parallel(_,X) :- X == nofeature.
orthogonal(ceiling,door).
orthogonal(ceiling,wall).
...
equalheight(floor,floor).
equalheight(ceiling,ceiling).
equalheight(door,_).
...
```

Figure 2: Left: Semantic net for scene interpretation. Right: Externalized knowledge representation in Prolog with facts for every arc of the net and a condition for the `No Feature` label.

entities and their relations. The entity `door` represents an *open* door. The semantic net can easily be extended. Further entities have to be accompanied by a more sophisticated feature detection. This paper concentrates on plane detection so that the semantic net is a subset of all indoor environments.

Prolog is used to implement and externalize the semantic net. The net is encoded by definite Horn clauses [18]. The nodes of the net are arguments and the arcs define relations on the nodes. Figure 2 right shows a part of the semantic net as a Prolog program. All facts for the relation `parallel` are shown. For encoding the label `nofeature`, a condition is used. This prevents Prolog's unification algorithm from assigning planes with the label `nofeature`. In addition to the representation of the semantic net, a clause of the following form is compiled from the analysis of the planes. The planes are represented by variables `P0`, `P1`, etc.:

```
labeling(P0,P1,P2,P3,P4) :-
parallel(P0,P1),under(P0,P1),
orthogonal(P0,P2),under(P0,P2),
orthogonal(P0,P3),under(P0,P3),
parallel(P0,P4),above(P0,P4),
...
```

Prolog's unification algorithm is used to assign a consistent labeling to the planes:

```
consistent_labeling(P0,P1,P2,P3,P4) :-
labeling(P0,P1,P2,P3,P4).
```

The label `nofeature` is considered, iff the unification fails. In this case, an additional Horn clause is used to generate a consistent labeling with explicit unifying of one variable with `nofeature`. All combinations are computed to unify the variable:

```
consistent_labeling(P0,P1,P2,P3,P4) :-
comb([P0,P1,P2,P3,P4],[nofeature]),
labeling(P0,P1,P2,P3,P4).
```

The process is continued with assigning two variables the label to `nofeature`, and so on until a Prolog's unification succeeds:

```
consistent_labeling(P0,P1,P2,P3,P4) :-
comb([P0,P1,P2,P3,P4],[nofeature,nofeature]),
labeling(P0,P1,P2,P3,P4).
...
```

The order of the rules above is important, as in all Prolog programs. Prolog searches for clauses from top to bottom. The following three clauses are used to compute all combinations:

```
comb(_,[]).
comb([X|T],[X|Comb]) :- comb(T,Comb).
comb([_|T],[X|Comb]) :- comb(T,[X|Comb]).
```

Finally the following query is submitted:

```
consistent_labeling(P0,P1,P2,P3,P4).
```

and the automatic generated Prolog program starts and computes the solution. Table 1 shows the computation time (Pentium IV-2400, SWI-Prolog [15]) of the Prolog program and a previous complete depth first search implementation [12].

number of planes	backtracking	Prolog
5	93.51 ms	89.33 ms
7	155.14 ms	101.81 ms
13	589.11 ms	313.79 ms

Table 1: Computation time for matching the semantic net with the planes.

4 Model Refinement

Due to unprecise measurements or registration errors, the 3D data might be erroneous. These errors lead to inaccurate 3D models. The semantic interpretation enables us to refine the model. The planes are adjusted such that they explain the 3D data, and the semantic constraints like parallelism or orthogonality are enforced.

To enforce the semantic constraints, the model is first simplified. A preprocessing step merges neighboring planes with equal labels, e.g., two ceiling planes. This simplification process increases the point to plane distance, which has to be reduced in the following main optimization process. This optimization uses an error function to enforce the parallelism or orthogonality constraints. The error function consists of two parts. The first part accumulates the point to plane distances and the second part accumulates the angle differences given through the constraints. The error function has the following form:

$$\begin{aligned} E(P) &= \sum_{p_i \in P} \sum_{\mathbf{x} \in p_1} ||(\mathbf{x} - \mathbf{p}_{i1}) \cdot \mathbf{n}_i|| \\ &\quad + \gamma \sum_{p_i \in P} \sum_{p_j \in P} c_{i,j}, \end{aligned} \qquad (3)$$

where $c_{i,j}$ expresses the parallelism or orthogonality constraints, respectively, according to

$$\begin{aligned} c_{i,j} &= \min\{|\arccos(\mathbf{n}_i \cdot \mathbf{n}_j)|, \\ &\qquad |\pi - \arccos(\mathbf{n}_i \cdot \mathbf{n}_j)|\}, \text{ and} \\ c_{i,j} &= |\tfrac{\pi}{2} - \arccos(\mathbf{n}_i \cdot \mathbf{n}_j)|, \text{ respectively.} \end{aligned}$$

Minimization of $E(P)$ (eq. (3)) is a nonlinear optimization process.

The time consumed for optimizing $E(P)$ increases with the number of plane parameters. To speed up the process, the normal vectors $\mathbf{n}$ of the planes are specified by spherical coordinates, i.e., two angles α, β. The point $\mathbf{p}_1$ of a plane is reduced to a fixed vector pointing from the origin of the coordinate system in the direction of $\mathbf{p}_1$ and its distance d. The minimal description of all planes P consists of the concatenation of p_i, with $p_i = (\alpha_i, \beta_i, d_i)$, i.e., a plane p_i is defined by two angles and a distance.

4.1 Powell's Method

A suitable optimization algorithm for eq. (3) is Powell's method [13], because the optimal solution is close to the starting point. Powell's method finds search directions with a small number of error function evaluations of eq. (3). Gradient descent algorithms have difficulties, since no derivatives are available.

Powell's method computes directions for function minimization in one direction [13]. From the starting point P_0 in the n-dimensional search space (the concatenation of the 3-vector descriptions of all planes) the error function (3) is optimized along a direction $\mathbf{i}$ using a one dimensional minimization method, e.g., Brent's method [14].

Conjugate directions are good search directions, while unit basis directions are inefficient in error functions with valleys. At the line minimum of a function along the direction $\mathbf{i}$ the gradient is perpendicular to $\mathbf{i}$. In addition, the n-dimensional function is approximated at point P by a Taylor series using point P_0 as origin of the coordinate system. It is

$$\begin{aligned} E(P) &= E(P_0) + \sum_k \frac{\partial E}{\partial P_k} P_k + \sum_{k,l} \frac{\partial^2 E}{\partial P_k \partial P_l} P_k P_l \\ &\quad + \cdots \\ &\approx c - \mathbf{b} \cdot P + \frac{1}{2} P \cdot \mathbf{A} \cdot P \end{aligned} \qquad (4)$$

with $c = E(P_0)$, $\mathbf{b} = \nabla E|_{P_0}$ and $\mathbf{A}$ the Hessian matrix of E at point P_0. Given a direction $\mathbf{i}$, the method of conjugate gradients is to select a new direction $\mathbf{j}$ so that $\mathbf{i}$ and $\mathbf{j}$ are perpendicular. This selection prevents interference of minimization directions. For the approximation above the gradient of E is $\nabla E = A \cdot P - \mathbf{b}$. From the differentiation $(\delta(\nabla E) = \mathbf{A}(\delta P))$ it follows for directions $\mathbf{i}$ and $\mathbf{j}$ that

$$0 = \mathbf{i} \cdot \delta(\nabla E) = \mathbf{i} \cdot \mathbf{A} \cdot \mathbf{j}. \qquad (5)$$

With the above equation conjugate directions are defined and Powell's method produces such directions, without computing derivatives.

The following heuristic scheme is implemented for finding new directions. Starting point is the description of the planes and the initial directions $\mathbf{i}_l$, $l = 1, \ldots, n$ are the unit basis directions. The algorithm repeats the following steps until the error function (3) reaches a minimum [14]:

1. Save the starting position as P_0.
2. For $l = 1, \ldots, n$, minimize the error function (3) starting from P_{l-1} along the direction $\mathbf{i}_l$ and store the minimum as the next position P_l. After the loop, all P_l are computed.
3. Let $\mathbf{i}_l$ be the direction of the largest decrease. Now this direction $\mathbf{i}_l$ is replaced with the direction given by $(P_n - P_0)$. The assumption of the heuristic is that the substituted direction

includes the replaced direction so that the resulting set of directions remains linear independent.

4. The iteration process continues with the new starting position $P_0 = P_n$, until the minimum is reached.

Experimental evaluations for the environment test settings show that the minimization algorithm finds a local minimum of the error function (3) and the set of directions remains linear independent. The computed description of the planes fits the data and the semantic model.

4.2 Downhill Simplex Method

Another suitable optimization algorithm for eq. (3) is the downhill simplex method as used by Cantzler et al. [4]. A nondegenerate simplex is a geometrical figure consisting of $N+1$ vertices in N dimensions, whereas the $N+1$ vertices span a N-dimensional vector space. Given an initial starting point P_0, the starting simplex is computed through

$$P_i = P_0 + \lambda \mathbf{i}_l, \tag{6}$$

with $\mathbf{i}_l$ the unit basis directions and λ a constant that depends on the problem's characteristic length scale [14]. In our experiments λ is set to 0.15.

The downhill simplex method consists of a series of steps, i.e., reflections and contractions [14]. In a reflection step the algorithm moves the point of the simplex where the function is largest through the opposite face of the simplex to some lower point. If the algorithm reaches a "valley floor", the method contracts the simplex, i.e., the volume of the simplex decreases by moving one or several points, and moves along the valley [14].

Figure 3 shows the minimization of eq. (3) with the downhill simplex method in comparison with Powell's method. The downhill simplex method performs worse during the first steps, but reaches a better minimum than Powell's method. The peaks in $E(P)$ produced by Powell's method are the result of search directions $\mathbf{i}$ cross a "valley" in combination with large steps in Brent's line minimization algorithm.

4.3 Final Refinement and Results

The semantic description, i.e., the ceiling and walls enable to transform the orientation of the model along the coordinate axis. Therefore it is not necessary to transform the model interactively into a global coordinate system or to stay in the coordinates given by the first 3D scan.

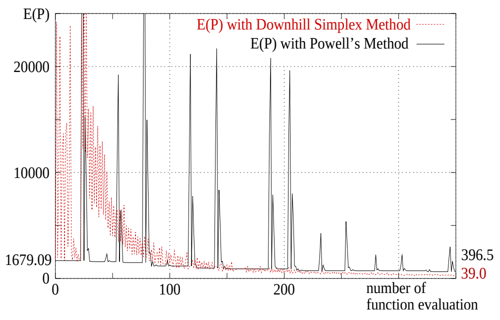


Figure 3: Minimization of the error function starting with $E(P) = 1679.09$. Powell's method finds a minimum at 396.5 and the downhill simplex reaches a minimum at 39.0.

The proposed methods have been tested in several experiments with our autonomous mobile robot in the GMD Robobench, a standard office environment for the evaluation of autonomous mobile robots. Figure 4 shows an example 3D point cloud (single 3D scan with 58680 points) and the semantic interpretation. The figure shows the reduction of the jitters at the floor, ceiling and walls (circled). The orientation of the model in the bottom image is transformed along the axis of the coordinate system and the meshing algorithm produces flat walls. Hereby an octree-based algorithm [16] generates the mesh (cube width: 5cm). The total computation time for the complete optimization is about 2.4 seconds (Pentium-IV-2400).

5 Conclusion

This paper has presented a new approach to sensor and knowledge based reconstruction of 3D indoor environments with autonomous mobile robots equipped with a 3D laser scanner. The proposed method consists of three steps and is applied after the 3D data is acquired:

- The first step is a fast feature extraction, i.e., plane detection. The presented algorithm is a combination of the ICP algorithm with the RANSAC approach.
- Second, the computed planes are labeled with a predefined semantic net. The semantic net

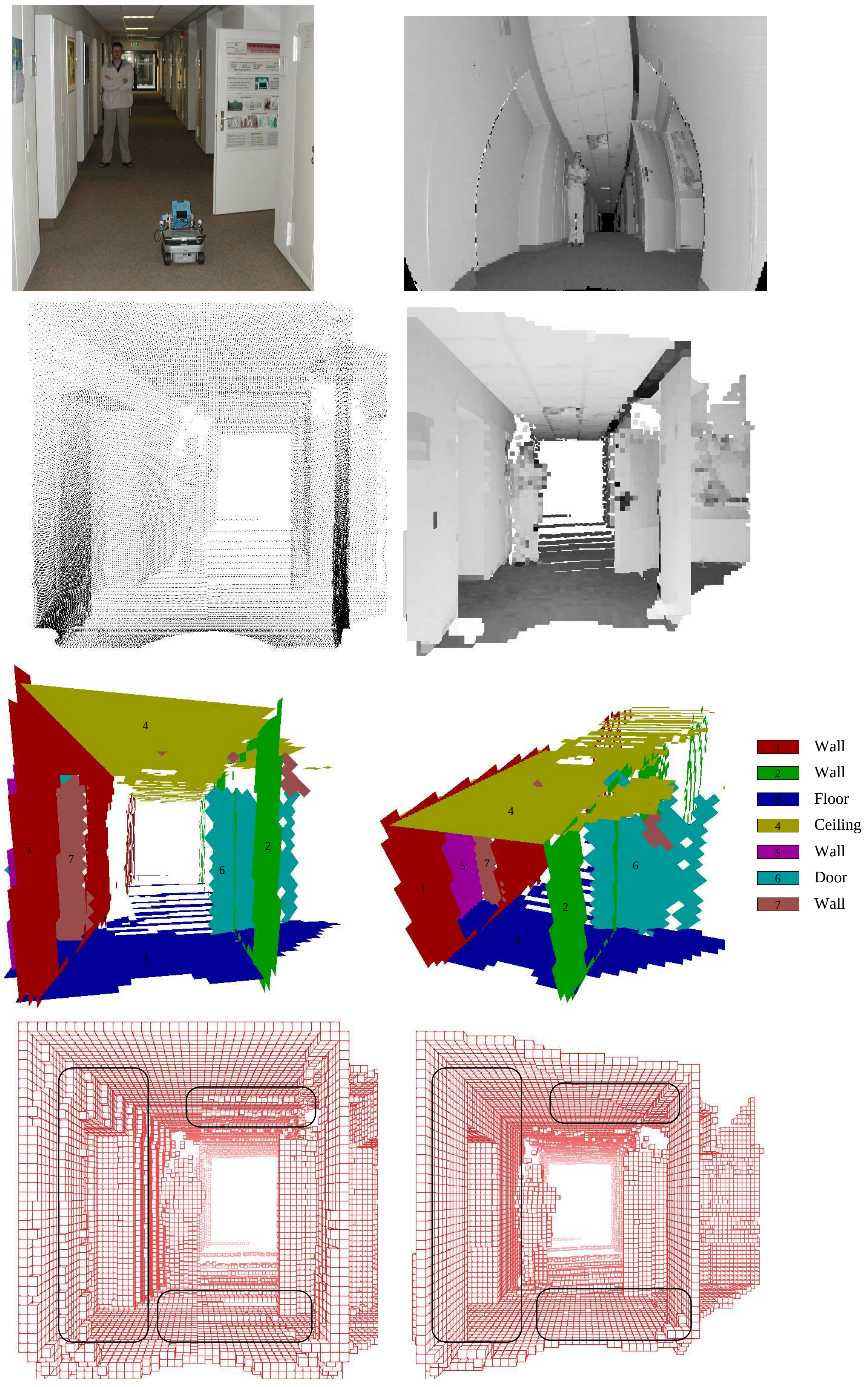


Figure 4: From top left to bottom right: Photo of the scanned scene; Reflectance values acquired by the AIS 3D laser range finder (distorted, since the rotation of the scanner is not considered [20]); 3D point cloud; rendered scene with reflectance values; extracted planes with semantic interpretation; unconstrained mesh, constrained mesh.

contains and implements general knowledge of indoor scenes. The semantic net is externalized and implemented as a Prolog program. 3D analysis of the extracted features compiles additional clauses and Prolog's backtracking and unification algorithm derives scene specific knowledge.

- Third the model is refined with the constraints arising from the semantic labeling. Numerical algorithms, i.e., Powell's method and the downhill simplex method are used for the 3D model improvement.

The proposed method will be included in the robot control architecture for the automatic gaging of indoor environments.

Future work will concentrate on the integration of two color cameras and enhancing the semantic interpretation by fusing color images with range data. The aperture angle of the camera will be enlarged using a pan and tilt unit to acquire color information for all measured range points. Furthermore the semantic net will be extended to more detailed features, i.e., non-planar features.

References

[1] The RANSAC (Random Sample Consensus) Algorithm. `http://www.dai.ed.ac.uk/CVonline/LOCAL_COPIES/FISHER/RANSAC/`.

[2] P. Allen, I. Stamos, A. Gueorguiev, E. Gold, and P. Blaer. AVENUE: Automated Site Modelling in Urban Environments. In *Proc. of the Third Int. Conf. on 3D Digital Imaging and Modeling (3DIM '01)*, Quebec City, Canada, May 2001.

[3] P. Besl and N. McKay. A method for Registration of 3–D Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239 – 256, February 1992.

[4] H. Cantzler, R. B. Fisher, and M. Devy. Improving architectural 3D reconstruction by plane and edge constraining. In *Proc. of the British Machine Vision Conf. (BMVC '02)*, pages 43 – 52, Cardiff, U.K., September 2002.

[5] H. Cantzler, R. B. Fisher, and M. Devy. Quality enhancement of reconstructed 3D models using coplanarity and constraints. In *Proc. of annual Symposium for Pattern Recognition (DAGM '02)*, pages 34 – 41, Zürich, Switzerland, September 2002.

[6] R. B. Fisher. Applying knowledge to reverse engeniering problems. In *Proc. of the Int. Conf.. Geometric Modeling and Processing (GMP '02)*, pages 149 – 155, Riken, Japan, July 2002.

[7] O. Grau. A Scene Analysis System for the Generation of 3-D Models. In *Proc. IEEE Int. Conf. on Recent Advances in 3D Digital Imaging and Modeling (3DIM '97)*, pages 221 – 228, Canada, 1997.

[8] D. Hähnel, W. Burgard, and S. Thrun. Learning Compact 3D Models of Indoor and Outdoor Environments with a Mobile Robot. In *Proc. of the fourth European workshop on advanced mobile robots (EUROBOT '01)*, Sweden, September 2001.

[9] M. Hebert, M. Deans, D. Huber, B. Nabbe, and N. Vandapel. Progress in 3–D Mapping and Localization. In *Proc. of the 9th Int. Symposium on Intelligent Robotic Systems, (SIRS '01)*, Toulouse, France, July 2001.

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

[11] Y. Liu, R. Emery, D. Chakrabarti, W. Burgard, and S. Thrun. Using EM to Learn 3D Models of Indoor Environments with Mobile Robots. In *Proc. of the 18th Conf. on Machine Learning*, July 2001.

[12] Andreas Nüchter, Hartmut Surmann, and Joachim Hertzberg. Automatic Model Refinement for 3D Reconstruction with Mobile Robots. In *Proc. of the 4th IEEE Int. Conf. on Recent Advances in 3D Digital Imaging and Modeling (3DIM '03)*, page (accepted), Banff, Canada, October 2003.

[13] M. J. D. Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *Computer Journal*, 7:155 – 162, 1964.

[14] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C : The Art of Scientific Computing*. Cambridge University Press, January 1993.

[15] SWI Prolog. `http://www.swi-prolog.org/`.

[16] K. Pulli, T. Duchamp, H. Hoppe, J. McDonald, L. Shapiro, and W. Stuetzle. Robust meshes from multiple range maps. In *Proc. IEEE Int. Conf. on Recent Advances in 3D Digital Imaging and Modeling (3DIM '97)*, Ottawa, Canada, May 1997.

[17] C. Robertson, R. B. Fisher, N. Werghi, and A. P. Ashbrook. Fitting of Constrained Feature Models to Poor 3D Data. In *Proc. of the Adaptive Computing in Design and Manufacture (ACDM '00)*, pages 149 – 160, Plymouth, UK, April 2000.

[18] U. Schöning. *Logik für Informatiker*. Spektrum Akademischer Verlag, 2000.

[19] V. Sequeira, K. Ng, E. Wolfart, J. Goncalves, and D. Hogg. Automated 3D reconstruction of interiors with multiple scan–views. In *Proc. of SPIE, Electronic Imaging '99, The Society for Imaging Science and Technology /SPIE's 11th Annual Symposium*, San Jose, CA, USA, January 1999.

[20] H. Surmann, K. Lingemann, A. Nüchter, and J. Hertzberg. Fast acquiring and analysis of three dimensional laser range data. In *Proc. of the of the 6th Int. Fall Workshop Vision, Modeling, and Visualization (VMV '01)*, pages 59 – 66, Stuttgart, Germany, November 2001.