2011 11th International Conference on Control, Automation and Systems
Oct. 26-29, 2011 in KINTEX, Gyeonggi-do, Korea

Real-time building of a 3D Model of an Indoor Environment with a Mobile Robot

Eunchul Jeon and Sungho Jo

Department of Computer Science, KAIST, Daejeon, 305-701, Korea
(Tel : +82-42-350-3540; E-mail: {jsharp, shjo}@kaist.ac.kr)

Abstract: This paper presents a real-time and autonomous algorithm for generating low-complexity multi-planar 3D models of indoor environments with a mobile robot equipped with range finders and a panoramic camera. In contrast with previous studies, our algorithm relies on neither iterative computations nor manual processing but, instead, is incremental online. At each time step, a line of range finder measurements is linearly segmented, and the data line segments extract planar surfaces of indoor environments. The structuralization procedure does not use polygon models for the measurements. Our algorithm can build a compact 3D map without requiring dense point clouds. Thus, the visualization through the graph is simple and quick, and experimental results verify the feasibility and power of our algorithm.

Keywords: 3D environmental model, Laser range finder, Mobile robot

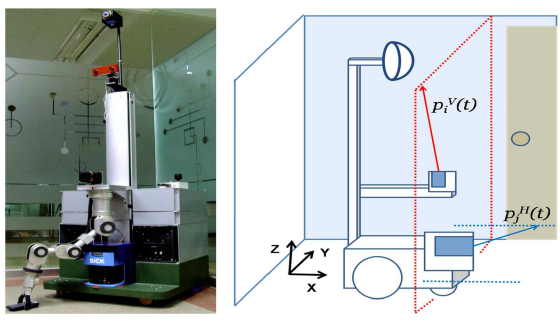
1. INTRODUCTION

This paper describes the problem of visualizing low-complexity, multi-planar 3D maps of indoor environments autonomously in real-time with mobile robots by acquiring range and camera measurements. As pointed out in previous studies [1]-[8], 3D models of environments can have important advantages over 2D maps because they provide richer information and less ambiguity [6-7]. 3D models are useful especially for remote interface. 3D information could be significant for implementing various robotic tasks, including navigation. Furthermore, 3D applications extend beyond robotics into fields such as architecture, emergency rescue, design, virtual reality, and exploration [8-9].

Even though various algorithms for 3D models of indoor environments have been proposed, most of them require polygonal models and are inappropriate for real-time execution. A well-known real-time executable algorithm proposed so far is based on an online variant of the Expectation-Maximization (EM) algorithm [9]. This EM-based algorithm operates by acquiring low-complexity, multi-planar 3D models of indoor environments. The online EM variant estimates the number of surfaces and their locations simultaneously. Although the approach is amenable for real-time execution, the EM algorithm principally relies on iterative computation. Therefore, its convergence rate influences the speed of execution. Depending on the environmental conditions, the map generation speed may be inconsistent. In contrast to previous studies, our algorithm uses a new incremental online approach with no iterative computation. Our algorithm does not rely on heavy data cloud points.

This work was supported in part by the Korea government (MEST) under the KRF grant (No. 2010-0015226) and in part by the Korea government (MKE) under Human Resources Development Program for Convergence Robot Specialists.

This work presents experimental results of building 3D models of indoor environments using the robot shown in Fig. 1(a). Our approach aims to build a 3D map that consists of planar surfaces under the assumption that such low-complexity map expression is both concise and sufficient for robotic applications, especially in indoor environments. A forward-pointed laser range finder (LRF) within the robot scans the environment horizontally with a field of view of 270° to localize itself during navigation. We use the MRPT library [18] to perform the localization based on the approach explained in [19]. This work presumes that the localization is sufficiently precise. The robot is also equipped with an upward-pointed LRF and a panoramic camera to obtain measurements of the environment. The panoramic camera is forwardly located on top of the robot, allowing the camera to detect side and top views as well as the front view. The upward-pointed LRF detects the environmental structure as the robot moves (see Fig. 1(b)). The robot is steered with differential drives. The entire robot is 60 cm × by 60 cm at the base and 160 cm high, and it can move at speeds of 0.1~ 0.6 m/s. The robot is operated by an onboard PC with a Core 2 Duo 2.53GHz processor and real-time Linux.



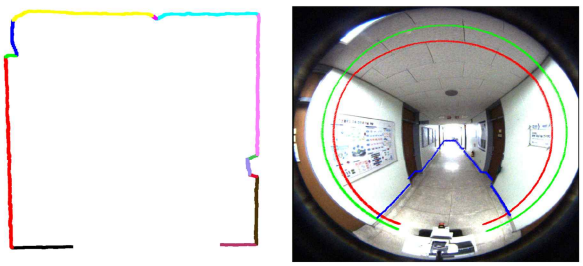
(a) (b)

Fig. 1.(a) Robot platform used in this work and (b) panoramic camera and LRF locations on the robot.

978-89-93215-03-8 98560/11/$15 ⓒICROS

2. DATA SEGMENTATION

The upward-pointed LRF acquires a line of data points extended over the walls, the ceiling and partially over the ground, as in Fig. 1(b). Because the data points are on a virtual flat plane, the line is the intersection between the flat plane and the environmental structure. Hence, each data point position can be described by a 2D vector $p^V(t) = \{p_i^V(t)\}$, $i=1,\ldots, N$, with respect to the upward-pointed LRF. The total number of data points, N, is fixed over time. The line of data points is then segmented into several line components by applying the split-and-merge line extraction algorithm[20][21](see Algorithm 1), which is most popularly used. This algorithm splits a line when the farthest point from line is bigger than threshold value T. End of this algorithm, we get a bunch of points in *SegmentList*. Each point where a new line segment begins is included in *SegmentList* together with the starting and end points of $p^V(t)$. For example, when a new line segment begins at $p_{15}^V(t)$, and $p_{37}^V(t)$, *SegmentList* stores (1, 15, 37, *N*).



(a) (b)

Fig. 2.(a) Linearly segmented vertical LRF data points and (b) LRF data points projected onto the image.

Algorithm 1 LRF line segmentation

Input: $p^V(t)$

$N \leftarrow$ number of points in $p^V(t)$
$SegmentList$.append(1)
$SegmentList$.append(N)
$T \leftarrow$ Threshold value
repeat
  for $i = 0$ to $SegmentList$.size()-2 **do**
    $start \leftarrow SegmentList[i]$
    $end \leftarrow SegmentList[i+1]$
    $maxDist = 0$, $index = 0$
    $L \leftarrow$ line through two points $p_{start}^V(t)$ and $p_{end}^V(t)$
    for $j = start$ to end **do**
      $dist \leftarrow$ minimum distance between a point $p_j^V(t)$ and a line L
      if maxDist < dist **then**
        $maxDist = dist$, $index = j$
      end if
    end for
    if maxDist > T **then**
      $SegmentList$.append(index)
    end if
  end for
until there is no appended index to SegmentList

Output: $SegmentList$

The LRF data points from a line segment are assumed be to be on an identical planar surface. Fig. 2(a) illustrates an example of the line segmentation. When the LRF data points are projected onto the image plane obtained from the panoramic camera at time *t*, they are shown to be circular, as in Fig. 2(b). The red line indicates the data points $p^V(t)$ on the image at current time t, and the green line indicates $p^V(t-1)$ at previous time *t*-1. The blue lines represent the LRF data points $p^H(t)$ at time *t* obtained by the forward-pointed LFR that are used for localization. The panoramic camera provides visual information on the environment. Due to its orientation and location, the ceiling, the walls on either side and the ground (with a partial occlusion of the robot's base) are captured in the image.

3. 3D STRUCTURE BULDING

3.1 Planar surface extraction

At time *t*, $p^V(t)$ are linearly segmented. Each linear segment is evaluated to be clustered into planar surfaces obtained at time *t*-1 and is used to update the planar information. If any surface is not searched satisfactorily, then the linear segment is regarded as being part of a new planar surface. Algorithm 2 explains the procedure. Let $C_i(t)$ identify each line segment$p^V(t)$includes. For example, with *SegmentList* (1, 15, 37, *N*), three segments are identified as (1, 15), (15, 37), and (37, *N*) by checking the *start* and *end* of each segment.

For points in each segment, the same value of $C_i(t)$ is assigned. Different line segments should be on different planar surfaces. Therefore, each $C_i(t)$ represents a surface index equivalently. For example,

Algorithm 2 Surface index allocation

Input: $p^V(t)$, $p^V(t-1)$, $C(t-1)$

$SegmentList \leftarrow$ Segment vertical laser sensor($p^V(t)$)
for $i = 0$ to $segmentList$.size()-2 **do**
  $start \leftarrow segmentList[i]$
  $end \leftarrow segmentList[i+1]$
  $counter < segmentIndex, num > \leftarrow \emptyset$
  for $j = start$ to end **do**
    $p_k \leftarrow$ the nearest point from $p_j(t)$ in $t-1$ frame.
    $counter[C_k(t-1)] \leftarrow counter[C_k(t-1)] + 1$
  end for
  $c \leftarrow \emptyset$
  for all $c_j \in counter$ by ascending order **do**
    if OnSurface(c_j, $start$, end, $p^V(t)$) **then**
      $c \leftarrow c_j$
      break
    end if
  end for
  if $c = \emptyset$ **then**
    $c \leftarrow$ CreateNewSurface($p^V(t)$, $start$, end)
  end if
  for $j = start$ to end **do**
    $C_j(t) \leftarrow c$
  end for
  UpdateSurface($p^V(t)$, c, $start$, end)
end for

$C_6(5) = 10$ implies that $p_6^V(5)$ is on the 10th planar surface. Suppose that we have m planar surfaces up to the t-1th time step. Then, m surface indexes exist in $C(0)$ to $C(t-1)$. $C(t-1)$ indicates surfaces on which line segments of $p^V(t-1)$ lies. With the *SegmentList* from $p^V(t)$, each line segment at t attempts to be associated with $C(t-1)$. If a line segment is determined to be on one of the existing planar surfaces, then the data information in the line segment is used to update the planar surface estimation as explained in section *C*. In addition, the related $C_i(t)$ is declared to indicate the surface by taking the corresponding index number from $C(t-1)$. To evaluate the association of a line segment with an existing plane, a line segment at time t-1 is selected that is nearest to an evaluated line segment at current time t. By checking a surface index that is most highly assigned to$C(t-1)$ of the nearest line segment, the current line segment is examined to determine whether it is associated with the plane the surface index indicates. If the line segment is not associated, then the second most highly assigned surface index in$C(t-1)$ is tested. The procedure is repeated until an existing planar surface is corresponded to or all surface indexes assigned to the nearest line segment in $C(t-1)$ are tested. If a line segment is not associated with any of the existing planar surfaces, then a new planar surface is designated based on the data information. $C_i(t)$stores a new index that indicates the new planar surface. The new index is numbered simply by adding one to the existing highest surface index. Therefore, the highest surface index informs the total number of planar surfaces in the 3D map up to the current time.

3.2 Correspondence with existing planar surface

To examine whether a line segment is on a planar surface, the estimated error variance of the plane is used (see Algorithm 3). It is presumed that all points from a line segment should be on an identical plane. For each point, the distance to the preexisting planar surface is computed, and the distance is compared with the error tolerance. If the distance is out of an allowed range, then the corresponding point is expected to not lie on the planar surface. Otherwise, the point is regarded as corresponding to the planar surface. The number of data points from the line segment within a tolerance range is counted. If the number is greater than the criterion, then the line segment is regarded as lying on the plane correspondingly. K in line 6 and Eh in line 10 of Algorithm 3 determine the strictness of the correspondence rule. The noise of the measurements exists intrinsically. Furthermore, during navigation, a robot, along with its sensors, could vibrate slightly due to the unevenness of the ground or other factors. Such fine vibrations can affect the measurements and, therefore, the surface correspondence. The relevant parameters are selected based on these issues.

Algorithm 3 Surface association check

FUNCTION: OnSurface(c, $start$, end, $p^V(t)$)

1: $count \leftarrow 0$
2: **for** $i = start$ to end **do**
3: $S_c \leftarrow SurfaceList[c]$
4: $\sigma_c^2 \leftarrow$ variance of S_c
5: $Err \leftarrow$ distance from $p_i^V(t)$ to surface S_c
6: **if** $K \cdot \sigma_c^2 < Err$ **then**
7: $count \leftarrow count$ +1
8: **end if**
9: **end for**
10: **if** $count < Eh$*$(end - start + 1)$ **then**
11: return true
12: **else**
13: return false
14: **end if**

3.3 Planar surface estimation

Once the segment allocation is determined, its data information is used to update the estimation of a planar surface a line segment lies on. Each planar surface is identified by a set of parameters $\{m_1, m_2, m_3\}$ where $m_1x + m_2y + m_3z = 1$. Let $p_i^G = (x_i, y_i, z_i)$ denote each LRF point in terms of the global coordinate. Suppose k LRF points are regarded as being on an identical plane up to now and are expressed by $\{p_i^G\}$, i=1,..., k. Then, l LRF points, $\{p_i^G\}$, i=k+1,..., k+l, in a line segment assigned to the same planar surface are used to update the parameters as follows.

$$X_{new} = X_{old} + PA_k(B - A_{k+l}^T X_{old}) \quad (1)$$

where $A_k = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \vdots & \vdots & \vdots \\ x_k & y_k & z_k \end{bmatrix}$, $B = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$, and $X = \begin{bmatrix} m_1 \\ m_2 \\ m_3 \end{bmatrix}$.

X_{old} and X_{new} are the parameter values before and after the update.

After the computation, P is updated by $P = P - PA_{k+l}A_{k+l}^T(I + A_{k+l}^T PA_{k+l})^{-1}P$ for the future computation.

Then, a surface normal vector of the plane is estimated to be $n = X/\|X\|$, and $d = 1/\|X\|$ is the distance of the surface to the origin of the global coordinate. The points on the surface hold the following equality.

$$n \cdot p^G = d \quad (2)$$

Assuming Gaussian measurement noise, the error variance σ^2 is also updated according to

$$\sigma_{new}^2 = \frac{1}{k+l}\Big((k\sigma_{old}^2 + (n_{new} - n_{old})^T H(n_{new} - n_{old}) - 2(d_{new}n_{new} - d_{old}n_{old})^T D + k(d_{new}^2 - d_{old}^2)) + \sum_{i=k+1}^{k+l}(n_{old} \cdot p_i^G - d_{old})^2\Big) \quad (3)$$

where n_{old} and d_{old} are surface parameters before the update, and n_{new} and d_{new} are after the update. After the computation, H and D are updated by $H = H + \sum_{i=k+1}^{k+l} p_i^G p_i^{G^T}$ and $D = D + \sum_{i=k+1}^{k+l} p_i^G$, respectively, for the future computation.

3.3 Planar surface estimation

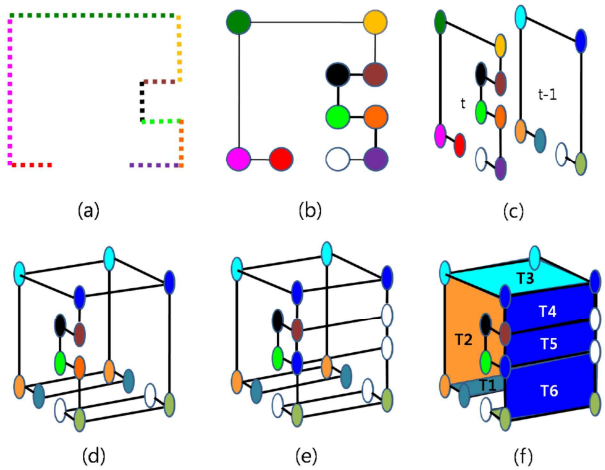


Fig. 3.The procedure of the structuralization and visualization

Fig. 3 illustrates the procedure of 3D structure building using a graph. Once $p^V(t)$ is measured, it is segmented into line components ((a)) through Algorithm 1. Each component's start and end points are designated as vertices in a graph, and the surface indexes are given through Algorithm 2 ((b)-(c)). Other points in a line segment are implicitly on the edge between the two vertices. The vertex indexes are allocated by checking the correspondence to the surfaces indicated with the vertex indexes from $p^V(t-1)$. Vertices identically indexed between t-1 and t (with the same colors) are connected to clarify each region, which an image texture is mapped onto ((d)). Sometimes, additional vertices are supplemented to assign some planar surfaces ((e)). The panoramic camera provides images as shown in Fig. 2(b), and an image strip that corresponds to vertices is collected at every time step. The image strip is circular between two sequential LRF measurement lines. Hence, the width of the image strip is determined by the distance between the two lines. The strip is stretched out and properly cut according to the vertex locations. Each image texture is superimposed onto the planar surfaces ((f)). T1 to T7 in Fig. 3(f) are an example of the image texture allocation. T4 to T6 are on the same plane, but they are mapped separately. Some edges in Fig. 3(f) are unassociated with planes. These edges indicate the appearance of new planes. The new planes will be built once after the next data $p^V(t+1)$ are measured. The overall procedure is repeated to execute the structuralization and visualization over the time steps.

$p^V(t)$is measured over time step t, and the distance between two sequential LRF lines depends on the mobile robot's speed. Sometimes, it is possible to miss important structural information between two lines. To minimize this loss, measurement $p^H(t)$ from the forward-pointed LRF (see Fig. 1(b)) is used. Fig. 4 shows an example of the structuralization based on

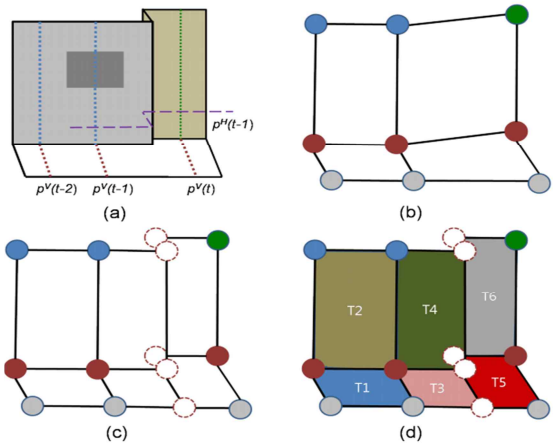


Fig. 4. The structualization using $\boldsymbol{p^H(t-1)}$

$p^H(t)$. In Fig. 4(a), $p^H(t-1)$ indicates that $p^V(t-1)$ and $p^V(t)$ are measured from different planar surfaces. Without the information, the graph would express the structure as in Fig. 4(b). However, the incorporation of the information supplements the additional vertices, as in Fig. 4(c). Then, the image textures are pasted on the planar surfaces, as in Fig. 4(d). Therefore, corners not detected by measurements $p^V(t)$ can be constructed. This process helps maintain the mobile robot's speed to some extent during real-time execution.

4. EXPERIMENTAL RESULTS

Experiments were conducted to demonstrate the main characteristics of the real-time 3D visualized map constructed by our algorithm. The mobile robot platform in Fig. 1(a) was used for the experiments. A series of experimental tests was designed to evaluate the capability of the overall algorithm. First, a map constructed with fine-grained polygons applied to raw data is compared with our approach. Second, structuralizations considered without and with $p^H(t)$ are compared. Third, the robotic testbed is run at different speeds to verify the mobile speed's effect on map construction. Fig. 5 compares views from the four experimental tests. In the top row, the fine-grained polygonal map is clearly less accurate due to noise in the raw data. The visual accuracy of textures on planar surfaces in our approach is obviously higher than in the polygon-based construction. In the second row, the effect of relying on $p^H(t)$ is demonstrated. The left figure clearly shows that the door's contour is less accurate by including the near wall area when measurements $p^H(t)$ are not taken into account. The accuracy of map construction does not seem to be very sensitive to the robot's speed. The left figure in the bottom row illustrates a built map, whereas the robot moves at a speed of 0.1 m/s, and the right figure at speed of 0.4 m/s. The higher speed causes a widening of each image strip cut. However, 3D structures are not much different from each other. In our algorithm, image

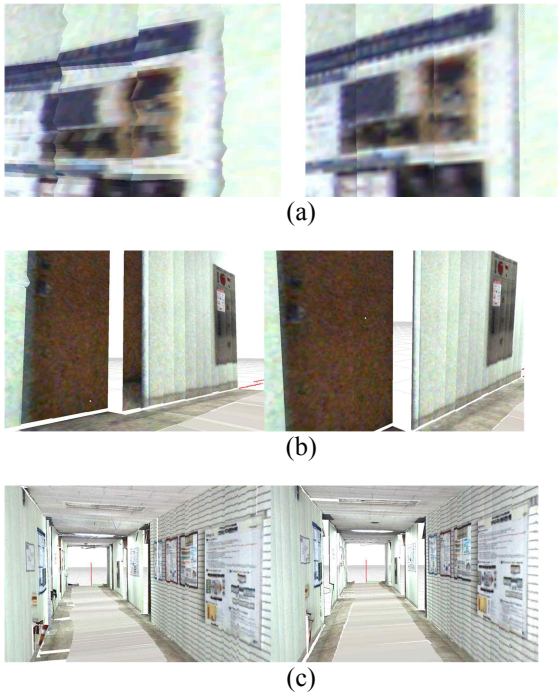


Fig. 5.(a) A view of a 3D visualized map produced by (left) fine-grained polygons and (right) our approach; (b) structuralization (left) without and (right) with the information in our approach; (c) map construction at robot's moving speed of (left) 0.1 m/s and (right) 0.4 m/s.

superimposition is executed at every time step in real-time. As the robot moves, the image strips are obtained at different angles from the light sources on the ceiling, which causes subtle color differences in the image strips, as shown in Fig. 5.

In the second experiment, the real-time map building was carried out in indoor corridor environments at the KAIST computer science building. Fig. 6 shows various views of a 3D visualized map obtained in real-time. The ceiling is removed for clarity in the top figure. The bottom figure illustrates a snapshot of map construction by rendering a robot's instant position. The built map is compact due to the multi-planar surface-based construction, and the visual detail is maintained at the image texture level. The map is extended every 300 msec, on average, including the built-in 2D localization operation while the robot moves at a speed of 0.4 m/s. The code programming is not optimized.

Some regions are not drawn; there are several reasons for this. First, some empty areas have no measurements. For example, if the surface between a door and a wall is located vertically to the robot during sensing, no measurement on the surface will be acquired. If desired, the surface could be interpolated. Second, the ceiling lights can cause incorrect LRF measurements of their areas, and the data will identify the wrong surface location. Third, surface estimation over LRF measurements is sometime not sufficiently precise, which leads to small cracks in the textured map. Furthermore, non-flat surfaces are not estimated.

Fig. 6. Several views of real-time 3D visualized map while the robot explores in corridor of KAIST computer science building.

5. CONCLUSION AND FUTURE WORKS

This work has presented a real-time and autonomous algorithm for visualizing compact 3D maps of indoor environments. Our implementation of structuralization and visualization is incremental online. At every time step, a consistent number of measurements is collected and processed. Neither iterative operation nor human manual input are required. Consequently, the overall processing is fast and, therefore, appropriate for real-time implementation. Experimental results demonstrated that our algorithm enables a mobile robot to draw compact and sufficiently accurate 3D maps of corridor-style indoor environments.

Although we assume that the localization is precisely performed, the localization algorithm we used is not perfectly precise in reality; its effect on3D visualization is sometimes not trivial. As already mentioned in [10], further investigation of interleaving localization and map building is an interesting future direction. Most similar studies have operated mobile

robots much more slowly to obtain LRF data points that are dense enough. In our case, lines of upward-pointed LRF data points are sparsely measured depending on the robot's moving speed. Even with sparse information of the structure, reasonably accurate maps can be acquired. We believe that our algorithm has practical applications for extracting compact structural descriptions well-suited for real-time robotic applications while allowing for realistic mobile robot speeds. In addition, the structural model is expressed compactly by a graph, as explained in section III.*D*. Therefore, our implementation is appropriate for representing a large structure efficiently.

Finally, it is worth noting the problem of non-flat surface construction, even though it was outside the scope of this work. As previous studies have shown [3, 8, 10], such non-flat objects could be handled locally by fine-gained polygonal models as long as sufficient measurements are acquired. Hence, a robot should move slowly to obtain dense measurements of non-flat objects. In practical scenarios, a robot may not need to detect detailed, small non-flat objects. In cases where it is necessary to recognize such objects, a robot could focus on acquiring their detailed measurements by briefly standing still, as if staring at them.

REFERENCES

[1] P.Biber, S.Fleck, and W.Strasser, "A mobile platform for acquisition of 3D-models of large environments-*The Wägele*", 2006.

[2] R. Treibel, W. Burgard, and F. Dellaert, "Using hierarchical EM to extract planes from 3D range scans",

[3] D. Hähnel, W.Burgard, and S. Thrun, "Learning compact 3D models of indoor and outdoor environments with a mobile robot", *Robotics and Autonomous Systems*, vol.44, pp.15-27, 2003.

[4] A. Nüchter, K.Lingemann, J.Hertzberg, and H. Surmann, "Heuristic-based laser scan matching for outdoor 6D SLAM", *LNAI*,pp.313-328, 2005.

[5] H. Surmann, A. Nüchter, and J.Hertzberg, "An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments", *Robotics and Autonomous Systems*, vol.45, pp.181-198, 2003.

[6] J. Weingarten, G. Gruener, R. Siegwart, "Probabilistic Plane Fitting in 3D and an Application to Robotic Mapping", *Proceedings of ICRA*, 2004, New Orleans.

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

[8] J. Weingarten, and R. Siegwart, "EKF-based 3D SLAM for structured environment reconstruction", In *ProcIEEE/RSJIntConf Intelligent Robotics Systems*, 2005.

[9] S. Thrun, S. Thayer, W. Whittaker, C. Baker, W. Burgard, D. Ferguson, D. Hähnel, M. Montemerlo, A. Morris, Z. Omohundro, C. Reverte, and W. Whittaker. "Autonomous exploration and mapping of abandoned mines",*IEEE Robotics and Automation*, 11(4), 2005.

[10] S. Thrun, C. Martin, Y. Liu, and D. Hahnel, "A real-time expectation maximization algorithms for acquiring multi-planar maps of indoor environments with mobile robots", *IEEE Trans. Robotics and automation*,

[11] R. Hartley and A.Zisserman, "Multiple view geometry in computer vision", Cambridge Universisty Press, ISBN: 0521623049, 2000.

[12] J. Diebel, and S. Thrun, "An application of Markov random fields to range sensing",

[13] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz,"Towards 3D Point Cloud Based Object Maps for Household Environments," *Robotics and Autonomous Systems* (Special Issueon Semantic Knowledge), 2008

[14] S. Thrun, M. Diel, and D. Hähnel, "Scan alignment and 3-D surface modeling with a helicopter platform", *ProcIntConf Field and Service Robotics*, Lake Yamanka, Japan, 2003.

[15] R. Triebel, P. Pfaff, and W. Burgard, "Multi-level surface maps for outdoor terrain mapping and loop closing", in *Proc IEEE/RSJ IntConf Intelligent Robots and Syst*, 2006.

[16] I. Kown, and T. Kanade, "High-resolution terrain map from multiple sensor data", *IEEE Trans Pattern Analysis and Machine Intelligence*, vol. 14, No. 2, pp. 278-292, 1992.

[17] http://www.mrpt.org/

[18] J. Marinez et al., "Mobile robot motion estimation by 2D scan matching with genetic and iterative closest point algorithms", Journal of Field Robotics, vol. 23, no. 1, pp. 21-34, 2006.

[19] D. Scaramuzza, A. Martinelli, and R. Siegwart, "A Flexible echnique for Accurate Omnidirectional Camera Calibration and Structure from Motion", in Proc *IEEE Int. Conf. on Computer Vision Systems(ICVS*), 2006.

[20] B.T.Nguyen, A. Martinelli, N. Tomatics, R. Siegwart, "A comparison of line extraction algorithms using 2D laser rangefinder for indoor mobile robotics", In *ProcIEEE/RSJIntConf Intelligent Robotics Systems*, 2005.

[21] L. Zhang and B. K. Ghosh, "Line Segment Based Map Building and Localization Using 2D Laser Rangefinder", *Proceedings of ICRA*, 2000