Preprints of the 19th World Congress
The International Federation of Automatic Control
Cape Town, South Africa. August 24-29, 2014

Autonomous Exploration of Large Unknown Indoor Environments for Dense 3D Model Building

Ivan Maurović * **Marija Đakulović** * **Ivan Petrović** *

* *University of Zagreb, Croatia, Faculty of Electrical Engineering and Computing, Department of Control and Computer Engineering (email: ivan.maurovic@fer.hr, marija.dakulovic@fer.hr, ivan.petrovic@fer.hr)*

Abstract: Autonomous exploration and mapping of indoor environments is important task for building inspections. Mapping of large environments in 3D requires high memory and computational consumptions. In this paper, we present a 3D exploration strategy for a mobile robot equipped with a 3D laser scanner. Our strategy does not require a map of the environment and ensures on-line exploration of large unknown spaces. We propose a room detection algorithm and focus on the room-by-room exploration keeping the memory and computational requirements low. We evaluated our strategy by simulations and experimentally using a real mobile robot.

Keywords: Autonomous exploration, 3D model, 2D exploration, 3D exploration, Room detection

1. INTRODUCTION

In this paper we present a 3D exploration strategy for large indoor environments. A mobile robot equipped with a 3D laser scanner autonomously navigates through the environment with the aim to build its dense 3D model. The exploration strategy refers to obtaining exact, discrete scanning positions from which an unknown environment is completely modeled. The number of positions should be as minimal as possible to decrease the exploration time. The process of environment exploration finds the applications in the area of modeling indoor environments for different purposes such as building inspections, structural inspections or in a combination with a thermal camera, the model can provide complete thermal information and inspect energy losses in the building. 3D modeling is also useful in various robotics applications where motion planning in 3D is necessary.

A lot of work has been done in the area of environment exploration Ekman et al. (1997); Đakulović et al. (2011); Yamauchi (1997); González-Baños and Latombe (2001) where the aim is to get a model of environment (either 2D or 3D) using only 2D information captured from the certain height level. Surmann et al. (2003) take 3D scan based on 2D map generated during the exploration. These methods can be used for 3D exploration in simple environments or when the dense model is not necessary. The obtained model would probably contain also unexplored volumes caused by the assumption that 2D exploration provides 3D coverage of the environment. Blaer and Allen (2007) present an 3D approach method for large outdoor environments in which the 2D map of the environment is needed in advance. After exploration based on 2D map the process continues with the 3D exploration. The typical runtime between two scans was about 15 minutes. Dornhege and Kleiner (2011) use a frontier based method extended to 3D exploration. This method requires high computational effort and operating environment is limited to small workspaces. Shen et al. (2012) proposed a stochastic differential equation-based exploration algorithm to enable exploration in 3D with limited onboard sensing and processing constraints for micro-aerial vehicle.

The main contribution of this paper is a new exploration strategy that addresses the following challenges of modeling large indoor environments. To overcome the problem of enormous memory consumption while exploring the environments and to reduce the computation effort the proposed exploration strategy divides the environment into enclosed spaces and explores until the whole environment is covered. While exploring a room only a local map of the room is in use and computational effort depends only on the size of currently explored room instead on the size of the explored environment as with existing 3D exploration strategies. Furthermore, the proposed strategy prevents the robot to jump between far away scanning positions since the exploration stays inside the detected enclosed space until it is fully modeled.

The concept of the proposed exploration strategy is shown in Fig. 1. The exploration starts with the empty map of the environment. In the beginning the robot explores 3D environment using a 2D based exploration algorithm (Section 2). When the robot detects a room the algorithm switches to 3D based exploration inside detected room. For that purpose, we have developed a new room detection algorithm (Section III). Based on 3D exploration method (Section 4) the room is being explored until the whole space inside the room is captured by the 3D sensor and the exploration process is switched back to 2D based

⋆ This research has been supported by the European Community's Seventh Framework Programme under grant No. 285939 (ACROSS).

Copyright © 2014 IFAC

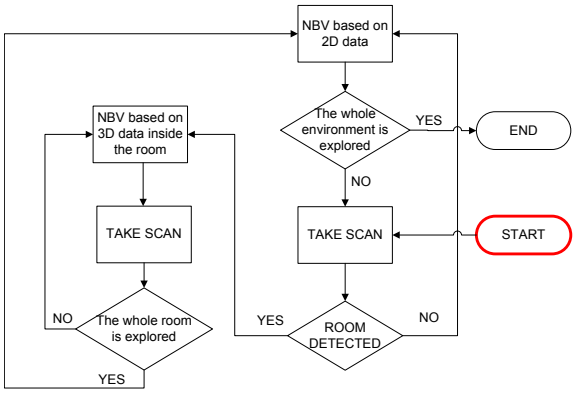

Fig. 1. The concept of the proposed exploration strategy - next best view planning (NBV).

exploration, which moves the robot out of the modeled room. The final result of the proposed algorithm is a complete 3D model of the indoor environment.

2. SENSOR PLACEMENT PLANNING IN 2D

The proposed sensor placement planning algorithm in 2D is based on our previous approach presented in Đakulović et al. (2011). The algorithm is adopted to 3D exploration by using 3D instead of 2D laser scanner. The inputs are range values in n uniformly distributed directions from the 360° field of view extracted from the 3D point cloud such that all range data lie in the plane parallel to the floor plane at the laser scanner height level. It is assumed that the environment model is unknown and incrementally built at each scanning position. The model is represented at three abstraction levels. At the lowest level the occupancy grid map is used for storing static and dynamic obstacle information needed for path planning and obstacle avoidance. The next abstraction level contains the polygonal representation of the environment which stores real environment edges extracted from the range data. The top, most abstract, level contains candidate scanning positions which are searched to find the best goal position for exploration. The goal positions are delivered to the path planning module which navigates the robot between scanning positions, while we assume that the robot localization is resolved, e.g. using the GMAPPING module under ROS [1] . We use a motion planning algorithm based on the D* algorithm and the Dynamic Window obstacle avoidance method described in Seder and Petrović (2007). The sensor placement algorithm is composed of three steps which are executed at each scanning position: (1) vectorization – extracting lines from the range data, (2) creation of the exploration polygon – building the most recent 2D model of the environment, and (3) selection of the next sensor position – searching the next goal for the path planning module. These three steps are briefly explained in the following (details can be found in Đakulović et al. (2011)).

Vectorization. The main goal of vectorization is grouping the range data $\mathcal{R} = \{r_k \mid k = 1, \ldots, n\}$ into M subsets, where M is not known a priori. Each subset contains almost collinear points within some error limit. For each subset the line segment is obtained by the least squares method. First, the Progressive Probabilistic Hough Transform (PPHT, from the OpenCV library Matas et al. (1998)) is carried out over the range data. PPHT calculates initial estimations of line segments which are then used to group all range data around the calculated line segments according to their distances $\delta\rho_k$ from the lines. Lines are represented in polar coordinates $L = [\rho \;\; \alpha]^T$, where ρ defines the normal distance to the origin and α defines the angle between normal and positive x-axis. For every measurement point the distance $\delta\rho_k$ is obtained as

$$\delta\rho_k = |d_k \cos(\alpha - \Theta_k) - \rho|, \tag{1}$$

where $r_k = [d_k \;\; \Theta_k]^T$ refers to k-th laser point in polar coordinates. If the distance $\delta\rho_k$ is below some treshold value $\Delta\rho$ the point belongs to set $\mathcal{R}_p \subset \mathcal{R}$ for the corresponding line segment L_p.

The next step calculates more precisely line parameters by the least squares line fitting algorithm which solves the following minimization problem

$$L = \underset{\alpha,\rho}{\operatorname{argmin}} \sum_{k=1}^{N} \delta\rho_k^2, \tag{2}$$

where N is the number of points in each subset of range data. Line segments are deterimined by trimming the given line at extreme endpoints from the subset of range data. At the end of the vectorization process real environment edges (walls, obstacles) are represented with line segments.

Creation of Exploration Polygon. At each scanning position p_i two polygons are calculated: (i) the measurement polygon P_i created from the vectorization of the newest range data, and (ii) the exploration polygon EP_i representing explored area, updated incrementally with each new measurement polygon P_i.

The line segments calculated in the vectorization step are sorted such that their ending points have increasing angles when transformed into the polar coordinate system from $[-\pi, \pi]$. Connecting the ending points defines the polygon P_i at the scanning position p_i. Now, the polygon P_i is composed of real line segments and artificial edges, i.e., jump edges, between them. Jump edges separate explored and unexplored regions of the environment. However, some jump edges from the new scan might fall into already explored area from the previous scans. To discard those jump edges in each step we use the union of the new polygon P_i (from the last scan) and the old exploration polygon EP_{i-1} (from previous scans) as a representation of the currently explored area $EP_i = EP_{i-1} \bigcup P_i$, where $EP_0 = \emptyset$ initially. The jump edges within the union of polygon (from GPC library Vatti (1992)) are discarded. Jump edges that are larger than the preset value Δr are considered for the selection of the next sensor position. The minimal length of the jump edge Δr is chosen in accordance to the robot dimensions and enables discarding too short jump edges (i.e., robot cannot pass through too narrow passages). If the nonempty extended exploration polygon EP_i contains no jump edges, then it is considered as the reliable polygonal description and the exploration process is finished.

[1] Robot Operating System, http://www.ros.org

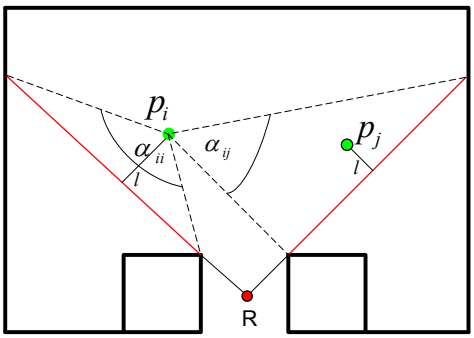

Fig. 2. Selection criterion based on angles of visibility. Jump edges are denoted by red lines.

Selection of the Next Sensor Position. For each jump edge one candidate scanning position is assigned. It is an obstacle free position near the mid point of the jump edge at distance l in front of the jump edge (Fig. 2). The distance is chosen to be equal to the robot's dimensions to ensure safety in case the jump edge is close to an obstacle that has not yet been detected. Additionally, l must be larger than the minimal sensor range, which enables range detection of all points inside unexplored area. The next sensor position is chosen by maximizing the criterion that estimates the amount of unexplored regions seen from each potential position:

$$I_j = k_1 \frac{1}{d_j} + k_2 \sum_{i=1}^{N} \alpha_{ij}, \tag{3}$$

where d_j is the length of the shortest path between the current robot position and the j-th candidate position calculated by the path planning module, N is the number of candidate positions and α_{ij} is the angle in the triangle defined by the j-th candidate position and the i-th jump edge. Two parameters k_1 and k_2 are used as weighting parameters of angle and distance estimations, respectively. The parameters are set in order to treat both terms in (3) equally. Numerous experiments in simulation and with the real robot showed good performance with k_1 set to the maximal range distance and k_2 set to $\frac{1}{N}$ which averages over all angles.

3. ROOM DETECTION ALGORITHM

Algorithm based on 2D information described in Section 2 could be efficiently used for 3D modeling of a simple indoor environments where rooms and obstacles inside are simply shaped. Applying 2D algorithm for exploration of a more complex environment can result with the incomplete 3D map model which contains holes as unknown volumes. That can be caused by occlusions generated from obstacles inside the environment like chairs, tables, wardrobes and others, since the area is covered only at certain height level. If the aim is to have a dense 3D model a method that seeks for unexplored area in 3D is necessary.

Implementing and applying a complete 3D exploration in large indoor environments would need protectively large memory area and computational effort due to the amount of information stored for the whole 3D environment. To avoid the complexity problem in this paper we propose the approach that combine both 2D exploration and 3D exploration, enabling tracking of three dimensional information of large environment to find unexplored volumes. The main idea relies on a typical indoor environment structure, compounded of enclosed spaces like rooms, halls, corridors and others enclosed structures. Starting with exploration based on only 2D measurements and following jump edges the robot detects the enclosed space, i.e. room, and switches to the 3D exploration which takes into account the whole 3D environment information captured by 3D laser point scanner. The 3D exploration algorithm is focused on exploration of the detected room as a small unit of the large environment. Therefore, a 3D algorithm can be used for exploring the single room, while keeping the computational and memory requirements low. When the room is fully explored, the 3D exploration algorithm terminates and exploration continues back with the 2D based strategy until it detects the new unexplored enclosed space and switches again to 3D exploration.

From the above discussion it is obvious that the room detection algorithm is the crucial step of the proposed combined 2D and 3D exploration process. Most of the room detection approaches rely on a grid map of the workspace. A fuzzy gridmap detector in Fabrizi and Saffiotti (2000) is extended in Buschka and Saffiotti (2002) to make it useful in on-line room detections. The fuzzy mathematical morphology is used to gather information about the shape of the empty space. Some approaches works fine on architectural floor planes like Ahmed et al. (2012), however they need precise maps in advance and they are not appropriate for on-line room detection process. In our case, since we use polygonal map in order to explore the environment, a line based room detection algorithm is developed, which can be used as an on-line room detector capable to detect room while exploring the environment.

3.1 Algorithm description

The room detection algorithm includes two steps: (i) vectorization and map building; and ii) room isolation process. Vectorization of the range data is described in section 2 where the output is polygonal map of the explored environment composed of real edges and jump edges. The input for the room isolation process are only real edges, while the jump edges are ignored. The algorithm is given by the pseudo-code in Alg. 1.

After taking the scan and before starting the next best view planning in 2D the room detection algorithm is called. 2D range data from the last scan are used in vectorization process of getting the line model of the potential room in 2D. The data are taken at some height level above the floor where the obstacles are rare and the room walls could be easily seen and detected. Typically, it is chosen at the height more than 2 meters.

Let us define a set A, representing all real edges detected from the first scan until the current scan. The order of the lines in A is not relevant. The detection starts from the first line a_1 in the set A. The algorithm introduces two sets, R and N, which refer to the set of lines which are part of the room and lines with no adjacent lines, respectively. The two lines are adjacent if the distance between their nearest line endings is smaller than the doorway length parameter λ. The two sets are organized as a stack with standard

Algorithm 1 Room detection

Require: 2D range data, doorway length λ, minimal room area $A_{\min}$
Ensure: Detected room

```
vectorization
initialization: A = detected lines, R = ∅, N = ∅, roomDetected = FALSE
for all a_i ∈ A do
  R.push(a_i)
  while 1 do
    y = findAdjacent(R.last(), A, N)
    if y ≠ ∅ then
      if y = a_i then
        if RoomArea(R) > A_min then
          roomDetected = TRUE
          break
        else
          N.push(R.last)
          R.pop
        end if
      else
        R.push(y)
      end if
    else
      N.push(R.last)
      R.pop
      if R = ∅ then
        BREAK
      end if
    end if
  end while
  if roomDetected = TRUE then
    break
  end if
end for
```

push and pop operations where $R.last$ refers to the last element in the stack R. Starting from the last element in R, function **findAdjacent** searches for the adjacent line from the the set of all lines A. The result of the **findAdjacent** can be the adjacent line or empty set in the case of which the adjacent line from the last line in R does not exist. Two reasons can cause empty set as a result: (i) no line exists in the vicinity defined by λ; (ii) lines in the vicinity are part of the N set, i.e. lines around have already been detected as not appropriate for closing a room. In the case of empty set the last element in R is deleted and pushed in the set N. If the adjacent line exists the set R is augmented with it and the whole procedure is repeated for the new line just added in R. The algorithm finishes if the adjacent line is the same as the starting one in R. That means the loop is closed and represents an enclosed space. The final step is to check the area of the detected room. If the area is smaller than the threshold $A_{\min}$, e.g. the loop encloses the corner of the bigger room, the process is repeated until all lines are grouped into the sets R and N. The returned value is set R.

The room detection was tested in simulation under the ROS environment using stage simulator. The laser sensor field of view was set to 360°. Figure 3 shows a simulated standard office environment with severals rooms, halls and doorways. The robot starting position is marked with the rectangle at the upper left. Since the simulation is in 2D and serves only as a test for the room detection, the 3D exploration was disabled. The result of the room detection is shown in Fig. 4, where each detected room is filled with different color. By comparing the real map of the office environment and detected rooms while simulating exploration, it can be seen that rooms are nicely recognized and the environment is divided into smaller units suitable for 3D exploration inside each of them.

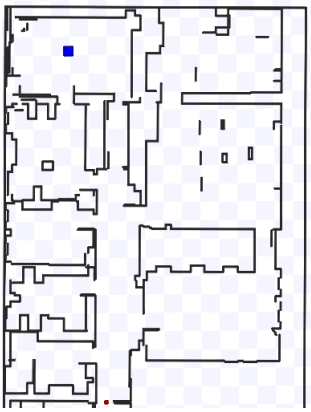

Fig. 3. Office environment used in the simulation

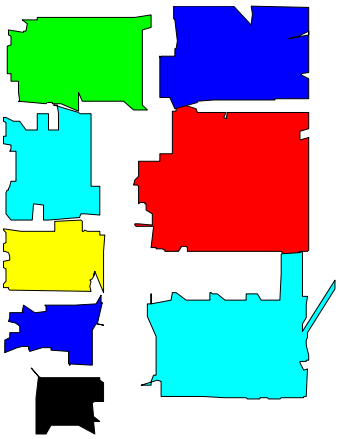

Fig. 4. Detected rooms of the office environment in Fig. 3

4. 3D EXPLORATION INSIDE THE DETECTED ROOM

The room detection algorithm provides the room parameters including the boundary area and the coordinates of the room. When the room is detected at least one scan inside the room is available since the robot has already entered the detected room. From the available scans inside the room the initial 3D model of the room is built and exploration continues by using the 3D exploration algorithm. The main idea of the 3D exploration algorithm used is described hereafter.

In order to obtain the next sensor position, the room 3D model needs to hold information of the explored and unexplored area inside the room. The room 3D model we use is voxel based where each voxel has one of the following marks: *occupied* if the volume within the voxel is occupied, *unseen* if occupied status of the voxel is unknown or *empty* if the voxel is empty with no obstacles inside.

Dimensions of the room refers to the number of voxels that should be used to cover the whole area of the room, i.e. memory allocation for the room. Since the shape of the room (enclosed space) could be arbitrary the 3D voxel model which describes the room is chosen to be cuboid where detected room volume fits entirely inside the cuboid. Although there could be voxels which are not part of the room, in the process of the next best sensor position they are ignored. When initial 3D model is created the next step is to detect areas with the most *unseen* voxel status and steer exploration towards that areas. The algorithm details can be found in Blaer and Allen (2007).

In the beginning, voxel map is initialized to unknown state with all voxels set to *unseen*, since we do not have any information on the environment. While scanning with 2D algorithm each scan is stored in memory together with the pose location where it was taken. Going through all scans only the scans inside the room update the initial map. Using ray tracing algorithm a ray is traced from the position where the scan has been taken to each data point in the scan. Each voxel that is crossed with the ray is marked as *empty* and the voxel joined to data scan is marked as *occupied*.

The potential next sensor position candidates are all voxels at the height of laser scanner with the status *empty* and reachable for the robot, which is tested by the path planning module. The aim is to choose the candidate position from where the most *unseen* voxels can be seen. For each candidate position we count the number of *unseen* voxels by tracing a ray from candidate to *unseen* voxel and if all crossed voxels are *empty* increase the counter for one increment. Since counting for every *unseen* voxel would be unnecessarily expensive only *unseen* voxels with at least one *empty* neighbor around are taken into account, as proposed by Blaer and Allen (2007). In that way the set of all voxels that need to be tested is decreased and voxels outside the room boundaries are not considered. The approach is similar to the jump edges in 2D because *unseen* voxels taken into account actually corresponds to jump planes which divide explored and unexplored regions. When a location that maximizes the number of unseen voxels is found, robot moves to the chosen position, takes the scan, and the whole procedure is repeated. The algorithm stops when the number of *unseen* voxels is bellow some predefined value V_{min}. Additionally, since the laser sensor introduces constraints in the field of view and range properties, the constraints are also included into account by checking the range and angle between potential candidate position and *unseen* voxels.

5. EXPERIMENTAL RESULTS

The overall exploration algorithm was tested experimentally on a differential drive mobile robot equipped with 2D SICK LMS100-10000 laser mounted on a pan-tilt unit FLIR PTU46 that enables 3D scanning of the environment. Figure 5 shows the part of the voxel based model of the environment used in 3D exploration algorithm. The *occupied* voxels are colored red, potential position voxels (PP) are colored blue, while white voxels represent *potentially seen* voxels (PS) that could be seen from at least one position in the workspace and which are subset of all *unseen* voxels (U) colored green. The specific green cone in the figure is a consequence of the laser field of view constraints. The other green voxels are situated under the tables and chairs inside the room representing unexplored area. The voxel volume used in the experiment was set to 0.008 m^3.

Table 1. The number of unseen (U), potential position (PP), and potential seen (PS) voxels at each scan inside the detected room

Scan	Scan 1	Scan 2	Scan 3	Scan 4	Scan 5
U (green)	930	604	567	340	303
PP (blue)	170	517	530	520	525
PS (white)	340	35	0	42	8

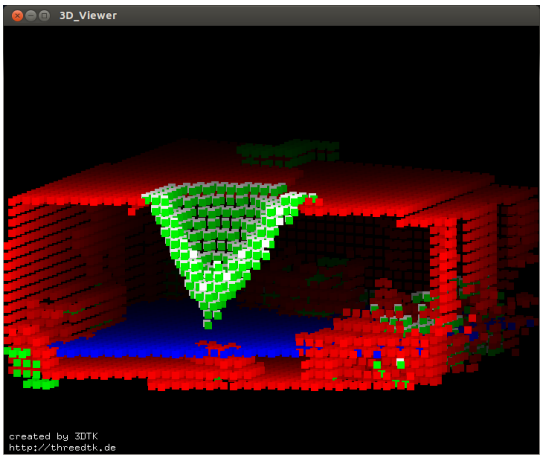

Fig. 5. The part of the voxel model of the experimental environment.

The experimental environment, together with the robot's path, consists of two rooms showed in the upper left image in Fig. 6. As the exploration process took 5 scans until the whole environment was explored, the numbers in the image refers to the scan order. The rest of the images in Fig. 6 show the 2D exploration polygon after each of five scans. Jump edges are colored red, the exploration polygon is colored blue and the candidate positions generated by 2D planner are noted by asterisks, while the best next position according to the 2D exploration is marked by a red circle.

Starting from the smaller room and taking the first scan, the enclosed space was detected and the algorithm was switched to 3D exploration. The next two scans (2 and 3) were taken based on the voxel map of the detected room and 3D planning process. After the third scan the smaller room was fully covered and the algorithm switched back to 2D exploration. The next scan position (scan 4) was obtained by 2D exploration after which the bigger room was detected. One additional scan position (scan 5) was chosen and the number of *unseen* voxels was bellow the treshold value (15 voxels). The algorithm finished since all rooms had been explored and, based on 2D exploration, no jump edges were found outside of the explored area. Table 1 summarizes the environment exploration process by showing the number of unseen (U), potential position (PP), and potentially seen (PS) voxels at each scan inside the detected room. The difference between PS and U number of voxels relies on the fact that some unseen voxels can not be seen from any potential position in the space.

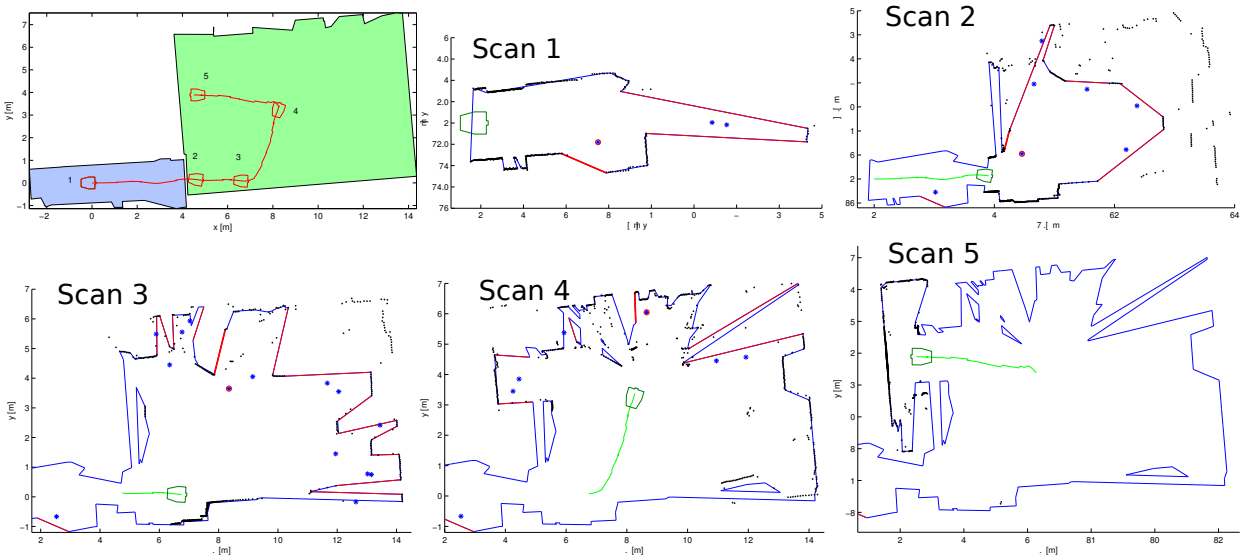

Fig. 6. A sequence of snapshots for the five scanning positions of the exploration strategy in the experimental environment.

6. CONCLUSION AND FUTURE WORK

The whole exploration algorithm is implemented and experimentally tested on the differential drive robot under ROS. Computational resources required for the exploration are decreased when the model of environment needed for 3D exploration is only local referring to the last room detected. The room detection algorithm can be used on a line map of the workspace or can work with range data together with vectorization process. The simulation results show the capability of the algorithm to be used in an indoor environment where the area is connected with rooms. The algorithm is able to detect when the robot is inside a room and to obtain room parameters such as coordinates of the room vertices and the room area. The experimental result shows that algorithm could be used in real environments under the sensor constraints and erroneous data. Although the experimental workspace is not large enough to prove the usability of the proposed algorithm, the results shows the way of exploring 3D environment room by room. For the future work we plan to test the algorithm in a large environment what would include many rooms and large areas.

REFERENCES

Ahmed, S., Liwicki, M., Weber, M., and Dengel, A. (2012). Automatic room detection and room labeling from architectural floor plans. In *Document Analysis Systems (DAS), 2012 10th IAPR International Workshop on*, 339–343. IEEE.

Đakulović, M., Ileš, S., and Petrović, I. (2011). Exploration and Mapping of Unknown Polygonal Environments Based on Uncertain Range Data. *AUTOMATIKA: Journal for Control, Measurement, Electronics, Computing and Communications*, 52(2), 118–131.

Blaer, P. and Allen, P. (2007). Data acquisition and view planning for 3-D modeling tasks. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, 417–422. IEEE.

Buschka, P. and Saffiotti, A. (2002). A virtual sensor for room detection. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 1, 637–642. IEEE.

Dornhege, C. and Kleiner, A. (2011). A Frontier-Void-Based Approach for Autonomous Exploration in 3D. In *Proceedings of the 2011 IEEE International Symposium on Safety,Security and Rescue Robotics*, 351–356. IEEE.

Ekman, A., Torne, A., and Stromberg, D. (1997). Exploration of polygonal environments using range data. *Systems, Man, and Cybernetics, Part B: Cybernetics*, 27(2), 250–255.

Fabrizi, E. and Saffiotti, A. (2000). Extracting topology-based maps from gridmaps. In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, volume 3, 2972–2978. IEEE.

González-Baños, H. and Latombe, J. (2001). A randomized art gallery algorithm for sensor placement. *SCG'01: Proceedings of the Seventeenth Annual Symposium on Computational Geometry*, 232–240.

Matas, J., Galambos, C., and Kittler, J. (1998). Progressive probabilistic hough transform. *Proc British Machine Vision Conference BMVC98*, 256–265.

Seder, M. and Petrović, I. (2007). Dynamic window based approach to mobile robot motion control in the presence of moving obstacles. In *Proceedings of the 2007 IEEE International Conference on Robotics and Automation (ICRA '07)*, 1986–1991.

Shen, S., Michael, N., and Kumar, V. (2012). Autonomous indoor 3d exploration with a micro-aerial vehicle. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, 9–15. IEEE.

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

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

Yamauchi, B. (1997). A frontier-based approach for autonomous exploration. In *Computational Intelligence in Robotics and Automation, 1997. CIRA'97., Proceedings., 1997 IEEE International Symposium on*, 146–151. IEEE.