Exploration of an Indoor-Environment by an Autonomous Mobile Robot

Thomas Edlinger
edlinger@informatik.uni-kl.de

Ewald von Puttkamer
puttkam@informatik.uni-kl.de

Department of Computer Science, University of Kaiserslautern
Erwin-Schrödinger-Straße, P.O.Box 3049, D-67653 Kaiserslautern, Germany
Tel.: +49 631 205 2624 Telex: 45627 unikl d Telefax: +49 631 205 2803

Abstract

This paper describes the exploration component of MOBOT-IV, an autonomous mobile robot for indoor-applications. MOBOT-IV has no preloaded world-model, but builds up an internal map, while exploring the environment using its optical range finder. A method is presented for the autonomous and systematic construction of a 2D-map of the vehicles environment. Besides a line based geometric representation of the obstacles a topological abstraction of the robots environment is generated representing the room structure.

1. Introduction

Autonomous Mobile Robots (AMRs) are systems which can move and perform useful operations without any external support or human intervention. The ability to operate in an unknown or partially known environment is essential for an AMR to be considered fully autonomous. Fundamental components are an actuator system to perform a given task and a perception system to build up a world model of the operating environment. If no explicit world model is given to the mobile robot it is more flexible in the case of different or changing environments. To achieve this flexibility the robot needs control software, which is able to plan exploration tours based on an incomplete description of the environment in order to get an complete and consistent world model. There are different approaches known from literature. They use idealized sensors of infinite measurement range [1] or built up only a topological representation of the environment [2] which is very critical, because a geometrical description is needed for path planning during the exploration tour. Approaches, which use a grid based method to distinguish between the known and unknown parts of the environment [3] will produce problems concerning the memory usage in case of large areas. In this paper a method is presented, which uses a real range finder to explore the environment and to build up a geometrical and topological representation of the vehicles environment suitable for large areas.

2. MOBOT-IV

The exploration method presented in this paper was developed for the autonomous mobile robot MOBOT-IV [4], [5], which

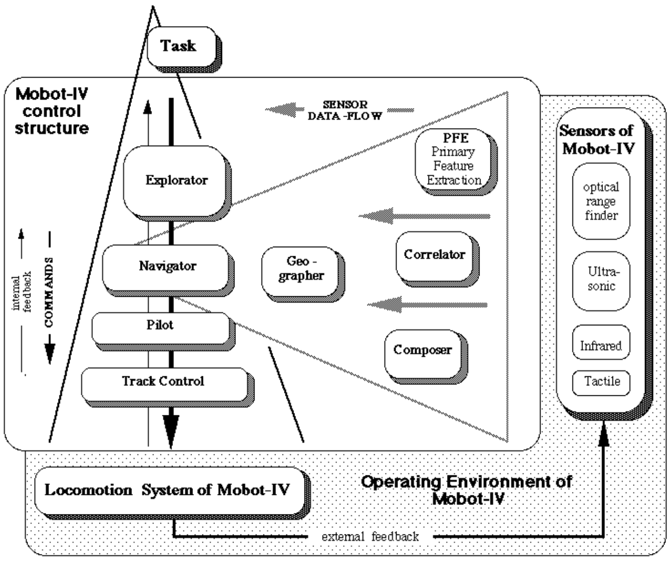

Figure 1: Control structure of MOBOT-IV

was designed to operate in indoor-environments. The control structure of MOBOT-IV consists of a vertical command tree and a horizontal sensor data processing tree, as shown in Fig. 1. The basic hardware and software components are described below.

2.1 The hardware

MOBOT-IV is a vehicle with one driven and steered front wheel and two passive rear wheels. Amongst other sensors it is equipped with an optical range finder as the primary sensor for environment perception. The rotating device produces 720 range measurements per revolution. These range values are the main input of the sensor data processing software. Together with the other software components its is running on a multi-processor VME-bus system.

2.2 The control architecture

As shown in Fig.1 the software components of MOBOT-IV are divided into two groups. The sensor data processing units (horizontal triangle) are responsible for the compression and verification of the raw sensor data in order to produce abstract and reliable information for the motion control components (vertical triangle). A unit called PFE (Primary Feature Extraction) produces the input for the Explorator by converting the scans of the range finder into a line based obstacle map called the Radar Map [6]. Besides this geometrical abstraction of the sensor data, it performs also a statistical abstraction, which is used for the external position estimation of the robot [7]. The robot knows thus its position in a global coordinate system. The Navigator is able to plan an obstacle free path from the vehicles actual position to a specified goal using a line based model of the environment as far as it is constructed at this time [8]. Such a component is necessary to move the robot collision free to the locations planned by the Explorator.

3. The algorithm

This section describes the details of the exploration-algorithm presented in this paper. The hierarchical method is divided into a local exploration which operates only in one room and a global exploration which tries to search the whole reachable area. One of the problems arising from this approach is to decide where a room begins and where it does end.

3.1 The concept

The basic idea is that the robot stays in a virtual *bubble* initially limited by the maximal measurement distance of the sensors. While the robot is moving the bubble deforms into the direction of the movement, so that the whole area scanned by the sensor remains always inside the bubble. Whenever the sensor detects an object, the objects surface will become a *real border* of the bubble in contrast to *virtual borders* which are determined by the maximal measurement distance of the sensor. The exploration phase terminates, when the bubble has no more virtual borders.

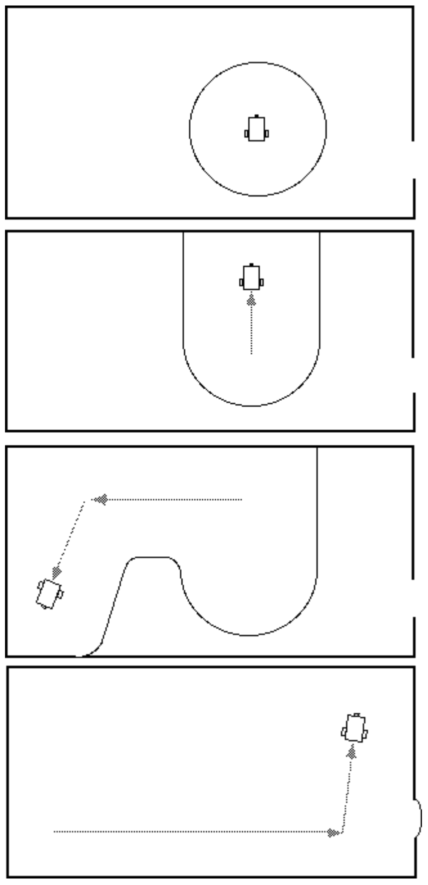

Figure 2: Construction of the bubble

Figure 2 shows the incremental construction of the bubble for a very simple environment. It shows a robot with a sensor, whose scanning range is relative small in relation to the dimensions of the room. At the first position it gets no information about obstacles. The bubble is a circle with the sensors scan distance as the radius. When the robot begins to move towards a wall the bubble is deformed until it reaches the wall. In order to change all virtual borders of the bubble into real borders, a very simple exploration strategy could be implemented: the robot always moves towards virtual borders. With this strategy the robot in the example of figure 2 explores the whole room. In order to achieve a hierarchical exploration, that means a room-by-room-exploration, the bubble is not allowed to deform through holes, which have less than a certain width. So the bubble cannot expand through the door on the right side, although the sensor has scanned through it into the next room. This has the effect, that a virtual border will remain in the bubble which is interpreted as a door. This information will be used in the global exploration for the construction of a topological way-net graph once the room under inspection has been explored.

So far the exploration strategy inside a room can be characterized by:

- **Criterion of termination**

  The exploration terminates, when the bubble consists only of real borders or virtual borders, which belong to doors.

- **Modelling of the knowledge**

  The knowledge about the environment and its objects is represented in the real borders of the bubble.

- **Modelling of the 'unknown'**

  Everything outside the bubble has not been explored jet.

- **View-point-planning**

  The robot always moves towards virtual borders of the bubble.

It should be noticed, that the 'unknown' is implicitly modelled. The strategy itself ensures, that everything inside the bubble was reached by the scanner. Even the view-point-planning is easy to realise. It only needs a criterion for selecting the next virtual border to drive to in the case, that more than one virtual border limits the bubble.

3.2 Representation of the bubble

One of the critical points in this algorithm is the insertion of a new sensor map into the already existing bubble. In the case, that the bubble should be modelled exactly, circle segments must be handled because of the shape of the scanning area of the sensor. Since the rest of the map is built of line segments, it turned out that it is difficult to work with two sorts of geometric primitives. So one could think of modelling the circles with small line segments. In both cases we have two kinds of virtual borders, which have to be handled in different ways.

- virtual borders, which connect real borders (between doors or small passages): virtual doors
- virtual borders, which limit the horizon of knowledge (scan area of the sensor)

To overcome this disadvantages the bubble is represented by two substructures. One data structure holds the real borders, the virtual doors and the virtual lines connecting them (*Sensorbubble*). The second structure represents the area in which the sensor has collected information about the environment (*Horizonbubble*). While the first data structure consists of lines, the Horizonbubble consists of a intersection of circles and is hard to handle. A solution of this problem is to represent the Horizonbubble not as a collection of all areas inside which the sensor has scanned, but by the positions at which the scanning took place. For every point in the robots environment it can be determined whether it is in the already scanned area or not. We only have to search for a scanning position in the Horizonbubble with a distance to the point under question less than the maximal measurement distance of the sensor. For reducing the amount of data which has to be stored and searched not every scanning position is stored but only positions, which have a certain distance to the last stored position. Experiments have shown, that a distance of about 30% of the scanners maximal measurement distance is a good compromise between the accuracy of the representation of the Horizonbubble and the amount of data which has to be stored.

3.3 Bubble fusion

The sensor data processing unit PFE produces a set of obstacle lines called the Radar Map for the exploration component. They are sorted by the sequence in which they have been detected by the sensor. The position from which the scan has been made is known, so that we can determine on which side of the line is free space and on which side is obstacle-space. A direction is given to each obstacle line, so that looking into this direction, free space is to the left side. Neighboured obstacle lines, which have no common endpoint, are connected by

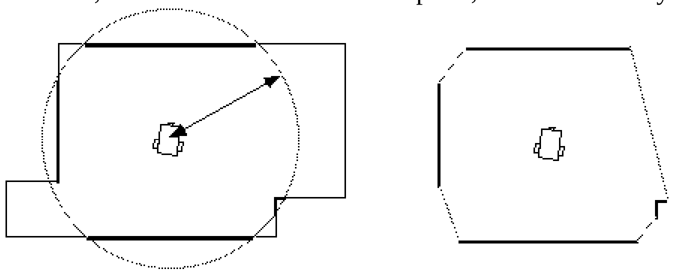
Figure 3: Transformation of the Radar Map

virtual lines, so that we now have transformed the Radar Map into the data structure of a Sensorbubble (Actual Sensorbubble) (see Fig. 3). For the incremental construction of the environment map, this transformed Radar Map has to be fused with the already existing Sensorbubble (Global Sensorbubble). The fusion of the actual and the global Sensorbubble have to fulfil several conditions:

a) the fusion must keep the relations of neighbourhood between obstacle lines
b) the length of all virtual lines in the resulting Sensorbubble must be minimal

Condition a) is easy to fulfil because in the actual and in the global Sensorbubble the information about neighbourhood of lines is stored. While fusing both Sensorbubbles we have to avoid that those neighbourhoods are destroyed and we have to detect new ones between lines from the two different Sensorbubbles. Condition b) is more critical, because for an exact solution of the problem no algorithm with polynomial complexity is known. In real-time-systems like an autonomous mobile robot such algorithms are not tolerable and so an algorithm with quadratic complexity is used, which produces a suboptimal solution of the problem. The Algorithm could be formulated as follows:

```
for every obstacle line $l_i$ in the actual Sensor-
    bubble do
  begin
   Find an optimal insertion point p for $l_i$ in the
       global Sensorbubble
   Insert $l_i$ at p in the global Sensorbubble
  end
```

This method is much more efficient than the exact algorithm and produces sufficient results in practical tests.

3.4 View-point-planning

In principle the view-point-planning could be divided into two phases:

- Finding points of interest (*POI*) from the actual knowledge about the environment
- Determine the ‘best’ point of interest by evaluating several criteria

A first kind of POIs is found near virtual lines in the Sensorbubble. One strategy could be to drive to the middle of a virtual line. In case of a very long virtual line this strategy will lead the robot away from the real lines, which is not intended.

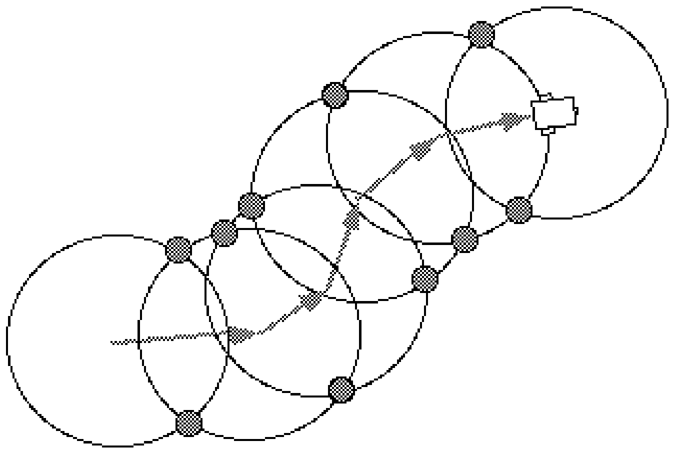

Figure 4: POIs from Horizonbubble

So every virtual lines produces three POIs. One in the middle and one at each end of the line. Points near real borders are shifted into free space so that the robot can actually reach them. To avoid the robot driving to the POI in the middle of the virtual line, a lower weight is asserted to this POI than to the POIs at the end of the virtual line. An obvious advantage of this strategy, that it realises a wall-following without explicit programming.

A second kind of POIs is computed from the Horizonbubble by intersecting two circles around two neighboured scan positions in the Horizonbubble (see Fig. 4). The intersecting points are POIs, which guarantee, that the whole area of a room is scanned. POIs of the last kind could be eliminated if they are in the already scanned area.

To determine the next view-point out of a set of POIs, a weight is assigned to each POI. This weight is computed from several features of the point. These features can be the distance of the POI to the actual position of the robot or in case of the first kind of POIs the length of the real lines next to the corresponding virtual line. The kind of features that are taken into account and their priority determine the exploration strategy of the vehicle.

3.5 Global exploration

The global exploration determines the sequence in which the rooms are explored. After the first room, which is determined by the robots start position, has been explored the system extracts the doors found in this room. If this door has been seen only from one side, it is put into the set of POIs of the global exploration. If more than one door is in this set, the door with the minimal distance to the robots momentary position is chosen. Then the way to the next door is planned hierar-

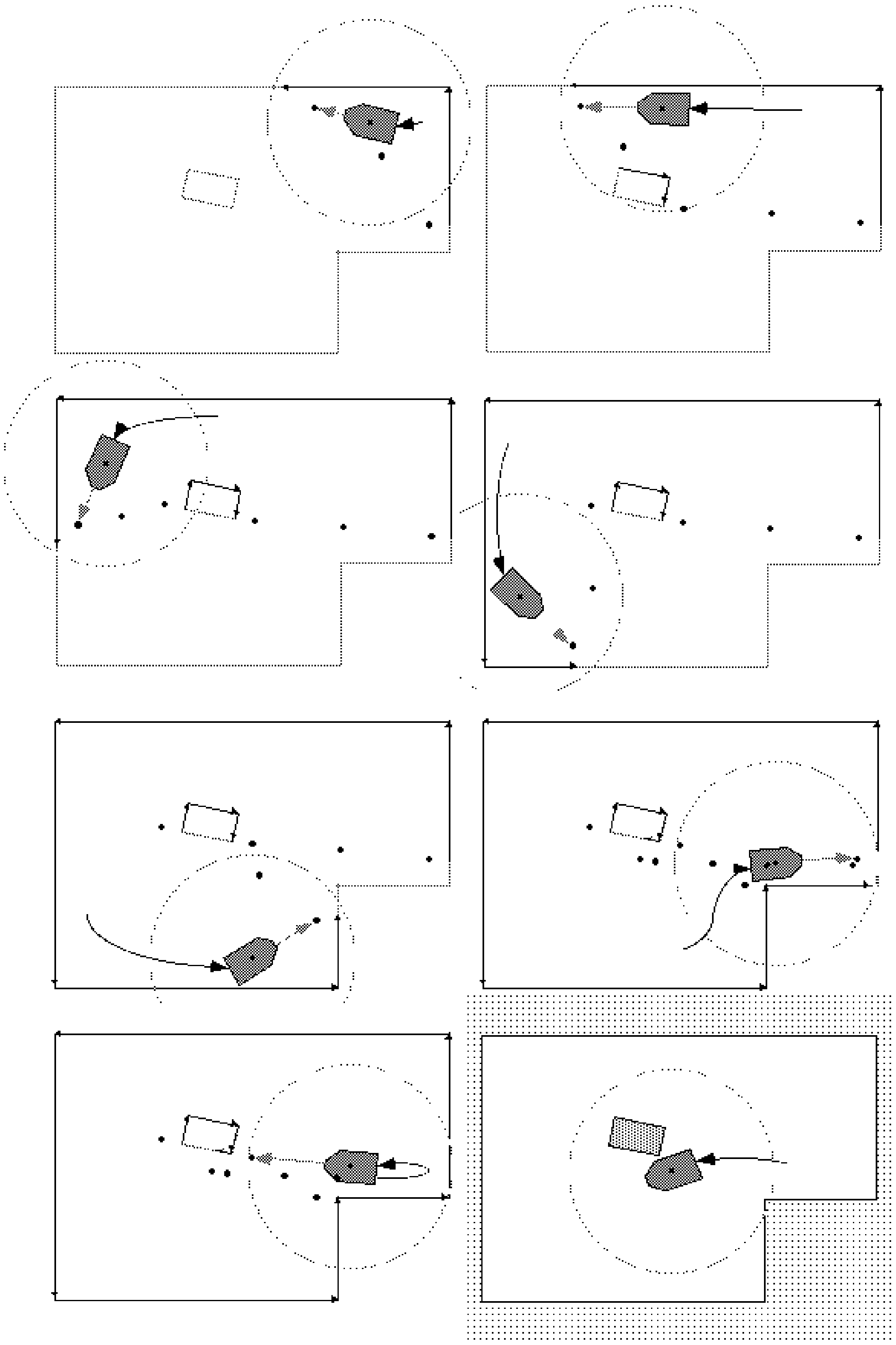
Figure 5: Test run of local exploration

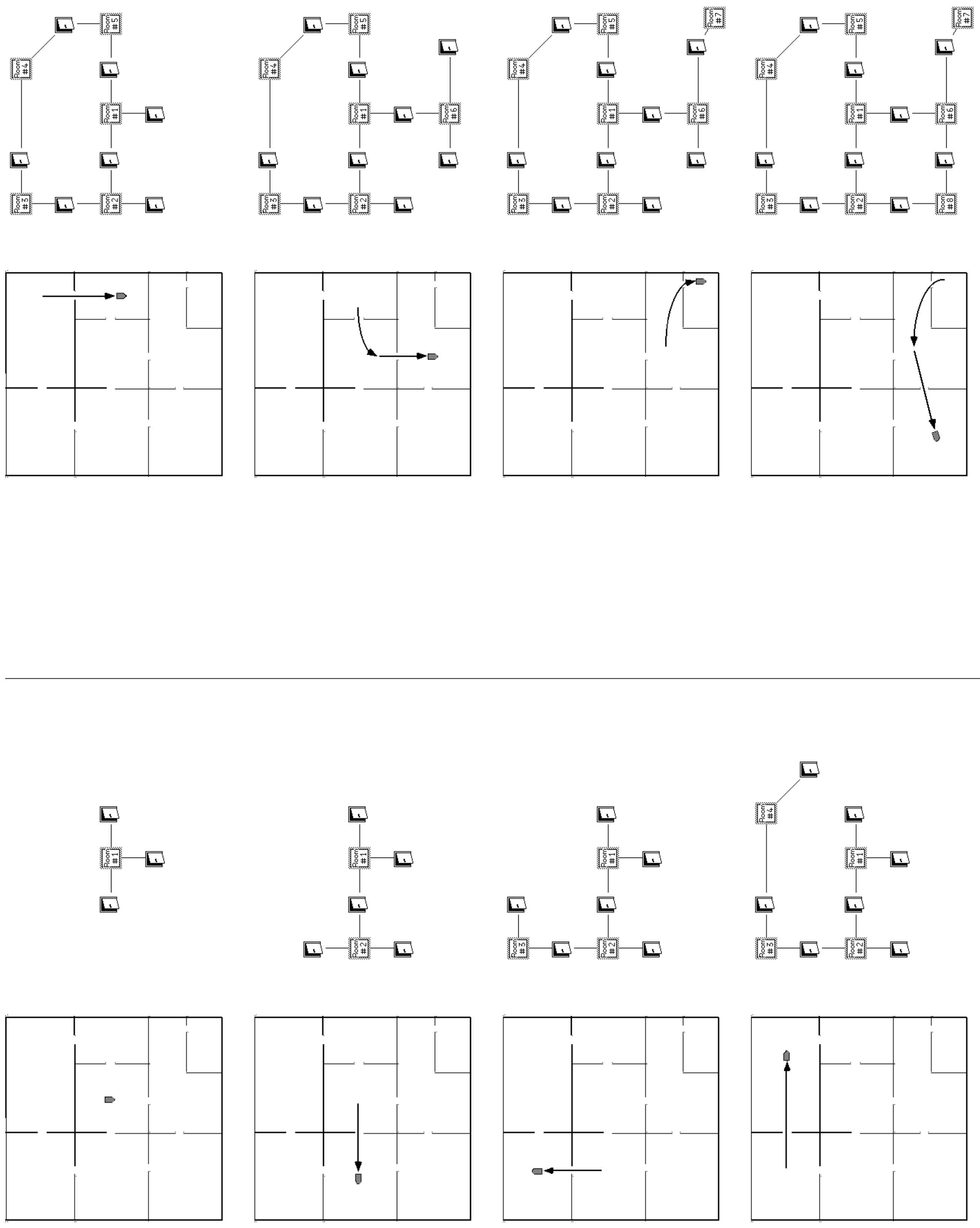

Figure 6: Test run of global exploration

chically. First the rooms which have to be passed are computed, based on the topological graph which the global explorator has built up to now. The navigation inside a single room is done by the geometrical navigator mentioned earlier.

4. Test results

The figures 5 and 6 on the previous pages show test results from a simulation of the exploration strategy. The first sequence of figures shows a local exploration-tour of the robot in a simple room with a table in it. The dots represent the points-of-interest produced by the system. The circle around the vehicle is the range of the robot's sensor.

The second sequence of figures shows a global exploration in a more complex environment. Each of the figures shows the environment and the path planned by the global explorator after the local exploration of one room has finished. In addition the incremental construction of the topological graph is shown, which represents the connectivity of the detected rooms as seen by the system.

5. Conclusions

In this paper a strategy for exploring an indoor-environment with an optical range finder is described. The results achieved with the simulation environment have shown that the method presented works for most indoor-environments. The algorithms for view-planning and map-building can be executed in real-time and are therefore suitable for the installation on an autonomous mobile robot.

References

[1] N. S. V. Rao, S. S. Iyengar, B. J. Oommen, R. L. Kashyap, “On Terrain Acquisition by a Point Robot Amidst Polyhedral Obstacles”, IEEE Journal of Robotics and Automation, Vol. 4, No. 4, Tsukuba, Japan, Aug. 1988, pp. 450-455

[2] B. J. Kuipers, Y. T. Byun, “A Qualitative Approach to Robot Exploration and Map-Learning”, AAAI Workshop on Spatial Reasoning and Multi-Sensor Fusion, St. Charles, Illinois, Oct. 1987, pp. 390-404

[3] J. Iijima, S. Asaka, S. Yuta, “Searching Unknown Environment By A Mobile Robot Using Range Sensor - An Algorithm And Experiment”, IEEE/RSJ International Workshop on Intelligent Systems 89, Tsukuba, Japan, Sep. 1989, pp. 46-53

[4] R. Hinkel, T. Knieriemen, E. v. Puttkamer, “MOBOT-III An Autonomous Mobile Robot for Indoor Applications”, ISIR 88, 19th International Symposium and Exposition on Robots, Sydney, Australia, Nov. 1988, pp. 489-504

[5] T. Knieriemen, “Autonome Mobile Roboter-Sensordaten-interpretation und Weltmodellierung zur Navigation in unbekannter Umgebung”, BI Wissenschaftsverlag, Mannheim/Wien/Zürich, 1991

[6] T. Knieriemen, E. v. Puttkamer, J. Roth, “Auswertung von Radarbildern mit Krümmungsalgorithmen im Vergleich mit Linienalgorithmen”, Autonome Mobile Systeme, 6. Fachgespräch der FG ‘Robotik’ in der GI (FA 1.2), Karlsruhe, Germany, 1990, pp. 185-195

[7] T. Edlinger, E. v. Puttkamer, R. Trieb, “Accurate Position Estimation for an Autonomous Mobile Robot Fusing Shaft Encoder Values and Laser Range Data”, IARP 91, 2nd Workshop on sensor fusion and Environmental Modelling, Oxford, UK, Sep. 1991

[8] P. Hoppen, T. Knieriemen, E. v. Puttkamer, “Laser-Radar based Mapping and Navigation for an Autonomous Mobile Robot”, IEEE International Conference on Robotics and Automation, Cincinnati, May 1990, pp. 948–953