In Proc. of the KTISIVIOS Panhellenic Conf. in Robotics and Automation, Santorini, Greece, Jul. 28-29, 2001.

PYTHEAS: an Integrated Robotic System with Autonomous Navigation Capabilities

K. Bekris[1,2], K. Hatzopoulos[1,2], G. Kazazakis[1,2], G. Kontolemakis[1], M. Masvoula[1], N. Tsivourakis[1], A. Argyros[1,2], P. Trahanias[1,2]

[1]Department of Computer Science, University of Crete, Heraklion, Crete, Greece

[2]Institute of Computer Science, Foundation for Research and Technology – Hellas, Heraklion, Crete, Greece

{bekris, hatzop, kazaz, kontolem, marisa, tsivour}@csd.uoc.gr
{argyros, trahania}@ics.forth.gr

Abstract

PYTHEAS is an integrated robotic software system that offers advanced navigation capabilities, which include localization, workspace mapping, path planning and tracking and obstacle avoidance. PYTHEAS facilitates mapping of an unknown indoors environment by exploiting information extracted from a laser scanner. Based on this acquired environment representation, the system is able to navigate autonomously in the mapped environment, while, at the same time, avoiding dynamic obstacles such as moving persons, etc. All the required competences are coupled in an integrated system, which can be controlled through a user-friendly interface over the web. Extensive experimental results demonstrate the capability of the developed system to map complicated environments and support navigation in dynamic worlds.

1. Introduction

PYTHEAS is an integrated system, which provides the ability to a user to remotely control a robotic platform, in order to map an unknown environment and effectively navigate in it, avoiding dynamic obstacles. Mapping is the procedure of extracting information from the physical environment by using the robot's sensors and transforming it to an appropriate internal representation of the environment. The accuracy of a map depends crucially on the alignment of the robot with the map. Therefore, estimating and constantly updating the robot's position, is an important issue in map building. This procedure is called localization. Assuming that the robot has an abstract description of the environment (a map) and that it knows its current position and the position of its goal, the robot must search the map for a path leading to the goal that is optimal under certain criteria, avoids static obstacles and that is wide enough to let the robot fit through. This phase is called path planning. Its output could be a set of sub-goal positions that have to be executed by the robot in a second phase, called plan execution. Motion planning relies on accurate, static models of the environments and therefore they often fail to function if unpredictable obstacles block their path. Autonomous mobile robots have to perceive their environments and re-plan dynamically in order to achieve their missions. The tasks mentioned above constitute very important issues in the mobile robotics field and many algorithms have been proposed for tackling them.

Recent research has produced two fundamental paradigms for modeling indoor robot environments, the grid-based, metric paradigm and the topological paradigm. Grid-based approaches, such as occupancy grids proposed by Moravec and Elfes [1], represent environments by evenly spaced (2D) grids. Each grid cell may, for example, indicate the presence of an obstacle in the corresponding region of the environment. Topological approaches represent robot environments by graphs, as initially proposed by Kuipers, Byun, Mataric and others [2,3].

A number of approaches that cope with uncertainty in robot's position are based on Markov models [4,5]. In [6] Thrun and Bucken confront the same problem by combining three kinds of information, wheel encoder readings, correlation between a local and the constructed global map and wall orientation. Usually the solution of the position tracking problem results in an independent module with accuracy that depends on the available computation time.

It is hard to analyze the state of the art in path planning since there is no commonly accepted general method for robot navigation. A large variety of methods have been published, that are special solutions for specific strategic tasks, machines and types of environments. A notable fraction of this work is either theoretic or has been tested only in simulations and additional work has to be done in order to test whether these methods operate satisfactorily in real robots. According to Latombe [7] all these methods can be categorized as “Roadmap Methods”, “Cell Decomposition Methods” and “Artificial Potential Field Approaches”. Another category of methods is recently under development, called “Navigation under Uncertainty”. This class of methods takes into consideration different types of uncertainties such as: uncertainty of sensory data, uncertainty of kinematics and of map information.

The motion planning approaches are usually unable to cope with the dynamics of real-world workspaces and a number of reactive approaches have been proposed to overcome this limitation. Dynamic window approach [8,9] is an example of a reactive collision avoidance strategy based on sequences of circular arcs. The Virtual Force Field method (VFF) [10], lies on the integration of two concepts, certainty grids for obstacle representations and potential fields for navigation, and enables continuous motion of the robot without stopping in front of obstacles. The Vector Field Histogram method (VFH) [11] is an extension of the VFF and uses a two-dimensional Cartesian histogram grid as a world model, which is updated continuously with range data.

The PYTHEAS system is an integration of various methods based on the approaches mentioned above. With respect to environment mapping, the grid-based approach has been selected. Metric maps are easy to build, represent and maintain even for complex environments. Laser readings are integrated over time, to yield a single consistent map according to a probabilistic model. Our approach for position tracking, is a variation of the method proposed by Thrun [12]. Odometry information is not used as a source of information because it becomes unreliable in the long run. For the task of path planning, the initial grid-based map is transformed to a topological one. As part of this transformation, a triangulation procedure has been used to find a set of control points, in order to cover the free space of the map. In this way, the problem of finding a path between current and goal position is turned into a shortest path graph-searching problem. The described approach exhibits small computational overhead, because the majority of computations are performed off-line. The motion planning method is a combination of "Navigating under uncertainty" approaches since it uses probabilistic models on the grid-based map to track the robot's position and of "Roadmap Planners" because of the use of intermediate points in a topological version of the map. In order to dynamically avoid objects that block the robot’s path to the goal position, the VFH algorithm is selected, because it results in smooth and continuous motion during obstacle avoidance. Those subsystems are controlled over a user interface that can issue commands to the robot even from a remote location over the network.

PYTHEAS consists of independent components, which can also be used separately to address each described sub-task of the application. It is possible to replace a certain component of the system without having to alter any other component. Moreover, PYTHEAS is a complete system, which totally handles mapping and navigation in an unknown environment, relieving the user from low-level commands. An important feature of PYTHEAS is that it can be remotely controlled over the web, with an easy to use interface.

The PYTHEAS system can be used in a variety of possible applications. The ability of web control allows the system to be used in applications that require remote control of the system, such as remote “representative” of the user in exhibition areas, trade shows, etc. PYTHEAS may also be employed in order to achieve navigation in hazardous environments, such as high radiation areas. Furthermore, a robot using PYTHEAS could play the role of a moving visual surveillance system. Especially, with the use of a panoramic camera, a large field of view is provided at all times. Since mapping is an independent module, PYTHEAS can also be used to map unknown indoor environments. Last but not least, it can offer services to people with special needs, since it can be used to control a wheelchair’s motion. The user can navigate through specific parts of a vocational or residential environment by selecting the desired destination from the extracted map, avoiding moving obstacles.

In the following section, the PYTHEAS system is described in detail. In section 3, experimental results are presented. The paper is concluded with a discussion presented in section 4.

2. The navigation system PYTHEAS

PYTHEAS is a fully developed navigation system that permits a user to control a mobile robot through the web. It is divided in three modules. The first module serves the need for mapping an unknown environment while the second is responsible for planning a path in this environment and avoiding obstacles during the execution of the path. In both modules, there is the need for the robot to estimate its position within the known world (localization). The final module offers to the user an interface for communicating with the entire system. The following sections, describe each module in detail. Moreover aspects of their integration are also presented.

2.1 Mapping

This module constructs a map of an unknown workspace in order to use it later for navigational purposes. The metric maps considered here are discrete, two-dimensional occupancy grids. Each grid-cell <x,y> in a map is associated with a value that measures the subjective belief that this cell is occupied. Occupancy values are determined based on sensor readings. Map building is an iterative procedure; each movement command initiates a new step, which is divided into the following four parts.

In the first part, the robot picks up a laser reading, which returns the distance of the robot from every obstacle in the front field of view. The laser scanner can only observe the half-plane in front of the robot, using a 180 reading sweep with angular accuracy of 1°. These readings are then translated into a local map. Let $s^{(t)}$ the sensor reading taken at time t. $P(occ_{x,y}|s^{(t)})$ is the probability that a grid cell <x,y> is occupied conditioned on the sensor reading $s^{(t)}$, ranging from 0 to 1, which corresponds to empty and occupied space, respectively. The desired probability can be computed in the following way:

$$1-\left(1+\frac{P(occ_{x,y}|s^{(T)})}{1-P(occ_{x,y}|s^{(t)})}\prod_{t=1}^{T-1}\frac{P(occ_{x,y}|s^{(t)})}{1-P(occ_{x,y}|s^{(t)})}\right)^{-1} \tag{1}$$

For the construction of the map, we used the midpoint algorithm for every laser beam in order to find the grid-cells best approximating the beam's line equation. For the cell corresponding to the laser reading we raise the probability of being occupied by setting $P(occ_{x,y}|s^{(t)})$ close to 1. For every other cell on the line with distance less than the laser reading, we lower it, by setting $P(occ_{x,y}|s^{(t)})$ close to 0. In this way, a local probability map is constructed.

In the second part, the position of the robot has to be estimated. For that purpose, the localizer procedure, explained later in section 3.2, is called with arguments the previous position of the robot and the movement instruction.

In the third part, after having estimated the robot's position, we add the local map, to the so far integrated global map, as shown in eq. 1. The size of global map is changing dynamically, allowing thus the robot to explore very large areas. Eventually, the local map, the so far created global map and the position-orientation of the robot are being communicated to the interface, facilitating the on-line evaluation of the mapping results by the user.

2.2 Localization

To navigate reliably in indoor environments, a mobile robot must know where it is. Localization is the problem of determining the position of a mobile robot from sensor data. Three methods are combined to achieve this capability.

In the first method, we consider a paralleloplane in the robot's configuration space, which acts as a search area for the other methods and we displace the probabilities from the previous call of localization accordingly. The previous movement command is employed in order to decrease the uncertainty about the robot's position. Adding the command to the previous result of localization, we get the centre of the search area. In this way, we completely ignore the odometry sensor of the robot, which, because of a number of reasons such as the morphology of the ground, may return erroneous measurements. The result of the odometry could also be used, but a larger search area is required.

The second method is the *correlation* between the measurements and the distances from the obstacles based on the current map. For every point in the search space <x,y,θ> we define the vector l(<x,y,θ>) which stores the 180 measurements that the laser sensor would have returned if the robot was on position <x,y,θ> and the map was accurate. The method computes the sum of the 180 differences between a laser measurement and l(<x,y,θ>)[i]. The smaller the value of this result, the greater the probability for cell <x,y,θ> to be the real position of the robot.

The last method used exploits *wall orientation*. This method is based on the fact that in indoor environments, walls are either parallel or form right angles. First, we assume that we only have one line segment constructed from the laser readings with a starting point corresponding to the first reading and an ending point to the last one. The algorithm searches for the point that has the greatest distance from the assumed line segment and splits the line in two. The algorithm is recursive and finishes when the distance of every reading from a line falls below a threshold or the number of lines exceeds a limit. Based on the assumption for the walls, the robot's angle is recalculated. The variation from Thrun's approach is that we assign weights to each wall according to the number of successive readings that compose the wall.

The values calculated from these three methods are normalized. We assign a weight for each method taking under consideration several experimental results and we update the probabilities according to:

$$J = (W_{corr}*corr[x,y,\theta] + W_{wall}*wall[x,y,\theta] + W_{move}*move[x,y,\theta]) / (W_{corr}+W_{wall}+W_{move}) \tag{2}$$

where W_{corr} is the weight of the correlation method, W_{wall} is the weight of the wall orientation method and W_{move} is the weight of the probabilities displacement. Finally, we calculate the maximum value of the J table which corresponds to the $\langle x,y,\theta \rangle$ point returned by localization. Figure 1 shows an example of how laser readings match the profile of the environment during concurrent localization and mapping.

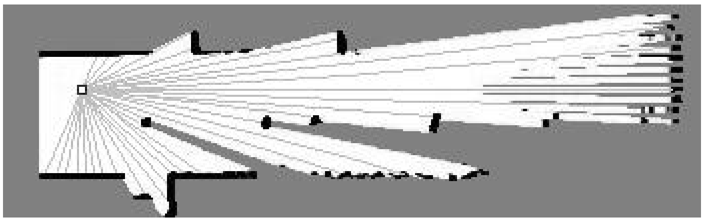


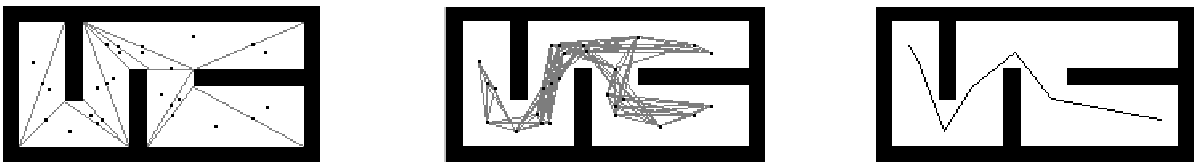
Figure 1. *Matching of the laser readings with global map*

2.3 Global path planner

The planner takes as input a grid-based map, a starting point and a goal point and it returns the shortest, non-occluded path from the starting to the goal point. Initially the grid-based map is thresholded and cells with high probability are considered obstacles. A smoothing operation on the grid-based map is performed using the morphological operators opening and closing. Afterwards, the borders of the free space and of all the obstacles are found and their polygonal approximation is computed. As a result, the free space is modeled as a non-convex polygon with holes. A triangulation procedure is then employed in order to fully cover the free space with a dense population of points, which are the mass centers and the middle points of triangle edges.

The next step is to create a graph, using as nodes the output of the triangulation procedure. In this graph, two nodes are connected with each other if a line segment, which is not occluded by obstacles, can connect their corresponding points in the map. The weight of a possible connection between two nodes is the distance between their corresponding points in the map. After the creation of the graph, the Warshal algorithm computes the shortest path from any node of the graph to any other. The complexity of this algorithm is $O(n^3)$, but it is called only once as an off-line procedure. Then, the problem of finding the shortest path from a random point to another is transformed to connecting these points to the existing graph and using the results of the Warshal algorithm.

The map positions that correspond to the nodes of the graph, which form the path from the initial to the target position, consist a set of control points. These points are intermediate targets passed to the obstacle avoider in order to reach the final destination. Figure 2 shows an example of the performance of the basic steps in a simple, synthetic map.



(a) (b) (c)

Figure 2: Global planner (a) Triangulated map including the centers of mass of the extracted triangles and the midpoints of the triangle's sides, (b) the connectivity graph and (c) the shortest path computed between two points.

2.4 Obstacle Avoidance

This part of the navigation software system is based on the Vector Field Histogram (VFH) algorithm. Its main purpose is to execute the plan provided by the path-planning module, and simultaneously to enable the robot to react properly in the presence of dynamic obstacles. As long as VFH is active, the robot is forced to navigate along the designed path until the desired target is reached. During robot's detours due to obstacles, its speed is optimally adjusted.

VFH uses as input the robot's laser readings and creates a two-dimensional grid-based local map, as described in section 3.1. Abrupt edges into the map are smoothed out and are then divided in sectors. Tempering each sector's grid values creates the corresponding sector repulsive values. They are named as Polar Obstacle Densities (PODs), and determine a quantum Cartesian object force histogram. All values in the object force histogram below a predefined threshold are considered as non-obstacle areas. The histogram is converted into a polar one, named Vector Histogram, which contains sector valleys that must be wide enough so as to be safe for the robot to pass through them. These valleys comprise a set of possible local steering directions. Taking into consideration the target course and the specified directions, a safe traveling drift relative to the robot, is determined (see Fig. 3). Based on the information from the global map and the localizer, it is converted into a global traveling direction, and an optimal steering speed is defined.

(a) (b)

Figure 3: (a) A sample 2D grid-based local map computed from laser readings. White color corresponds to free space, black color corresponds to obstacles and gray color corresponds to lack of information. (b) The vector histogram created from Fig 5(a). Light gray sector defines a non-safe (narrow) space and the dark gray sector defines a safe (wide) space. Although the target direction is along line 1, PYTHEAS will follow the direction along line 2.

2.5 Integration

PYTHEAS consists of mapping and navigation modules, which run autonomously. The user coordinates both procedures; therefore another system module is always working on a user host, since the user operates the system through the web. The overall module interconnection is shown in Fig. 4.

In the present version, the user interface module acts as a server, while the other two modules are its clients. Communication is implemented with two separate sockets, one for each client. The client–server communication supports either predefined or event driven message exchange procedures. For each client, a communication unit (COM) is assigned. In the case of navigation client, COM receives user commands to reach certain target positions or command to suspend operation. It then transmits the messages to the navigation module. In the case of the mapping client, COM receives movement commands from the user and transmits them to mapping module. Furthermore, all client modules can call COM functions to send information to the user interface.

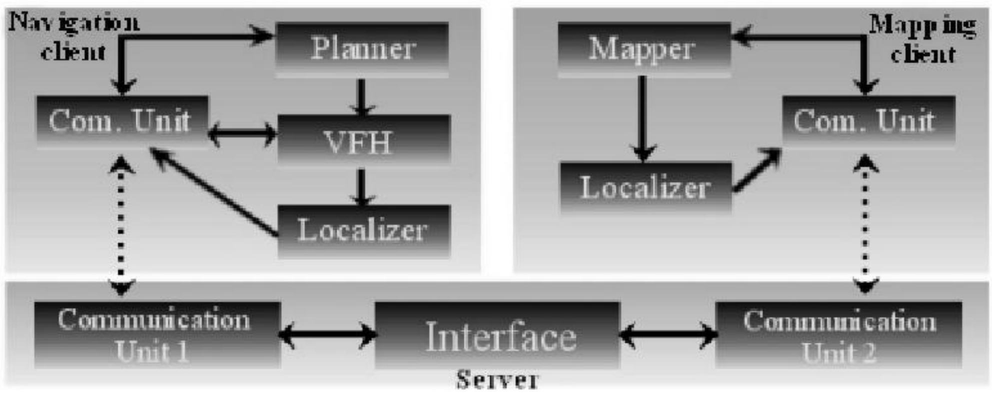


Figure 4: Overview of the integration architecture.

Whenever the planner is called to design a new path, it finds a set of control points that form a path trajectory to the target position. Then it calls the VFH to reach each one of them. VFH calls the Localizer periodically to retrieve the current position of the robot and the control returns to the planner if a control point is reached or a time limit is exceeded. On the other hand, mapping calls its localizer routines to retrieve the robot position, whenever a specific motion command has ended, or periodically, if mapping is autonomous.

2.6 Interface

The interface developed for the navigation system PYTHEAS offers the ability to the user to control map construction and navigation. A snapshot of this interface during operation, is shown in Fig. 5. The programming language used for developing the interface is Java 1.2 and the communication with the rest of the system is based on sockets.

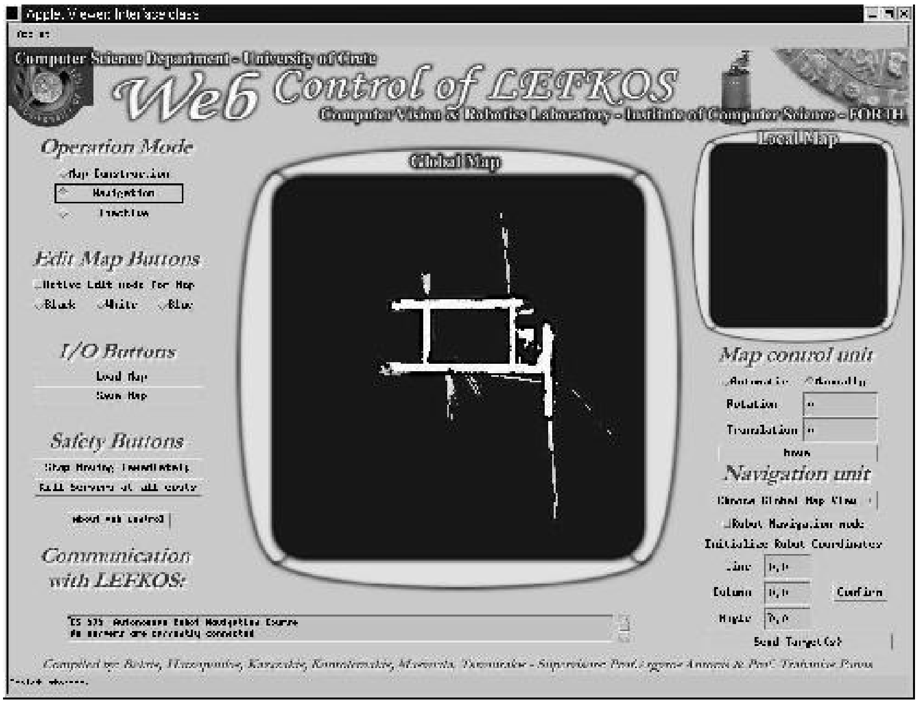


Figure 5: *The user interface.*

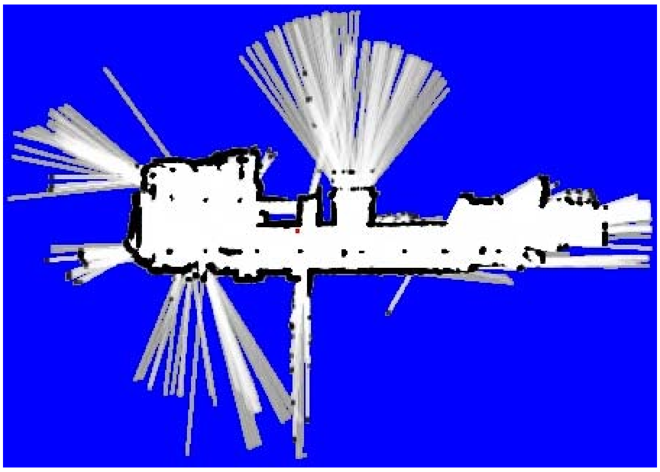
During map construction, one can either instruct the robot to automatically create a map or to manually control the robot's movement. In any point of this phase, the user can switch between these two modes of operation. For easing experimentation and for aesthetic purposes, a map-editing tool is also attached to the interface. The "Global Map" window shows the evolution of the map building procedure and the "Local Map" window informs about the robot's laser measurements in its current position. After this phase has ended the user can save the map in the robot's file system.

In the navigation mode, the user selects initially whether to use the currently loaded in the system map or to load another one. Available to the user are many intermediate results of the navigation module, such as the polygonal approximation of the map borders and all the available paths between the intermediate targets. After the user has defined the robot's position, direction and targets, either by clicking on the map or by filling the appropriate fields, the execution of the specified paths initiates. It should be noted that a text field continuously informs the user about the state of the communication between the interface and the rest of the system. It also offers guidance for an efficient control of the PYTHEAS system.

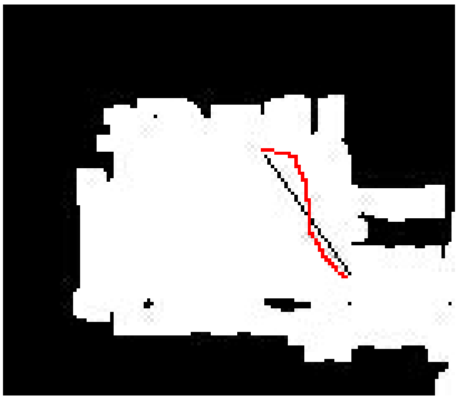
3. Experimental results

The proposed navigation system has been implemented on Lefkos, a mobile robotic platform available at ICS-FORTH. Several experiments were conducted in order to assess the performance of the integrated system. Due to space limitations only one characteristic experiment is presented.

Figure 6(a) shows the map constructed for the ground level of the main building of FORTH. Figure 6(b) shows a detail of the map where a simple path (straight line in black color) connects an initial position and a target one. Because of obstacles (moving persons), the robot finally followed the path shown in gray color. Snapshots of this navigation session are presented in Fig. 7.



(a)



(b)

Figure 6: (a) the map constructed for the ground level of ICS FORTH (b) Detail of the map where the black line depicts an original motion plan between two positions in the map and the gray line depicts the path followed by the robot in order to avoid obstacles.

4. Discussion

This paper presented PYTHEAS, an integrated system capable of acquiring maps of the environment autonomously and using them for navigating in the workspace. PYTHEAS is the integration of a number of state-of-the-art techniques in autonomous robot navigation, coupled together under a user-friendly interface that enables a user to remotely control it. The experimental results from extensive testing are very promising. Future research and development will address issues related to the expansion of the capabilities of PYTHEAS.

References

[1] H. Moravec and A. Elfes. High-resolution Maps from Wide Angle Sonar. *IEEE International Conference on Robotics and Automation*, pp116-121, St. Louis, MO, 1985

[2] B.Kuipers and Y.-T. Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. Journal of Robotics and Autonomous Systems, 8:47-63,1991

[3] M. J. Mataric A distributed model for mobile robot environment-learning and navigation. Master's thesis, MITm Cambridge, MA, January 1990, also available as MIT AI Lab Tech Report AITR-1228.

[4] Cassandra, A., Pack Kaelbling, L., and Kurien, J. Acting under uncertainty: Discrete Bayesian models for mobile-robot navigation. In Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS `96) (1996)

[5] Simmons, R., and Koenig, S. Probabilistic robot navigation in partially observable environments. In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI `95) (San Mateo, CA, 1995), Morgan Kaufmann, pp. 1080-1087. (Montreal, Canada, August 20-25, 1995)

[6] S. Thrun, 1998 Learning metric-Topoligical maps for Indoor Mobile Robot Navigation, Artificial Intelligence 99(1), 21-71

[7] Latombe, J.-C., Ed. Robot Motion Planning, vol. Fourth Printing 1996. Kluwer Academic Publishers, Boston, MA, 1991.

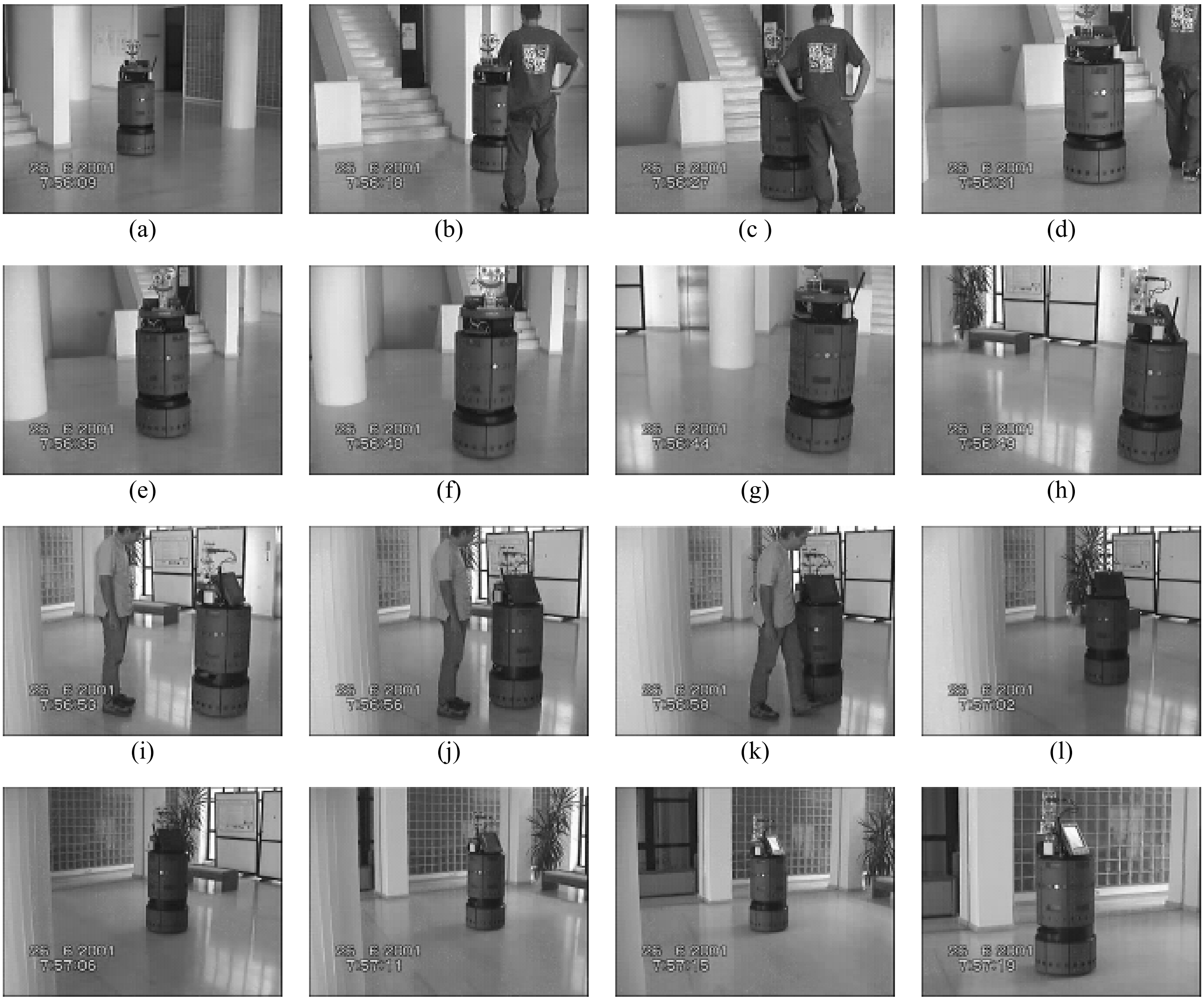
[8] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. Controlling synchro-drive robots with the dynamic window approach to collision avoidance. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 1996.

[9] Dieter Fox, Wolfram Burgard, and Sebastian thrun. The dynamic window approach to collision avoidance. IEEE Robotics and Automation Magazine, to appear.

[10] Borenstein,J. and Koren, Y.,1989,"Real-time Obstacle Avoidance for Fast Obstacle Avoidnce for Fast Mobile Robots." IEEE Transactions on Systems, Man, and Cybernetics, Vol. 19,No. 5,Sept./Oct.,pp,1179-1187.

[11] Borenstein,J. and Koren,Y.,1991,"The Vector Field Histogram -- Fast Obstacle-Avoidance for Mobile Robots. "IEEE Journal of Robotics and Automation,Vol.7,No. 3,June 1991,pp. 278-288.

[12] Thrun, S., and Bucken, A. Learning maps for indoor mobile robot navigation. Tech.Rep. TR CMU-CS-96-121, Carnegie Mellon University, Pittsburgh, PA 15213, April 1996.



(a) (b) (c) (d)
(e) (f) (g) (h)
(i) (j) (k) (l)
(m) (n) (o) (p)

Figure 7: Snapshots from a navigation session. The maneuvers that the robot performs in order to avoid moving obstacles in space can be observed in this experiment.