PERGAMON

Engineering Applications of Artificial Intelligence 12 (1999) 543–554

Engineering Applications of
ARTIFICIAL
INTELLIGENCE

www.elsevier.com/locate/engappai

Mobile robot path planning: a multicriteria approach

Juan A. Fernandez[a], J. Gonzalez[a], L. Mandow[b,*], J.L. Pérez-de-la-Cruz[b]

[a]*Dpto. Ingeniería de Sistemas y Automática, Universidad de Málaga/Campus Teatinos, P.O.B. 4114, 29080, Málaga, Spain*
[b]*Dpto. Lenguajes y Ciencias de la Computación, Universidad de Málaga/Campus Teatinos, P.O.B. 4114, 29080, Málaga, Spain*

Received 1 July 1997; accepted 1 February 1999

Abstract

This paper addresses the problem of searching paths in a graph-based model of the environment for mobile robot navigation. Unlike conventional approaches, where just a scalar cost (or a scalar function combining several costs) is to be optimized, this paper proposes a multicriteria path planner that provides an efficient and natural way of both defining and solving problems in which conflicting criteria are involved. In particular, the multicriteria METAL-A* algorithm is used as the core of a mobile robot global path planner. This algorithm has been implemented and tested in the RAM-2 mobile robot for indoor navigation. The results presented demonstrate the performance of the algorithm when dealing with energy-consumption, temporal, and clearance restrictions on the paths.

Keywords: Path planning; Mobile robots; Multicriteria decision theory; Goal satisfaction

1. Introduction

To achieve an autonomous navigation capability, a mobile robot must be able to plan a suitable path between a start and a destination in the environment. In large-scale space, i.e. environments where spatial structures are on a significantly larger scale than the sensory horizon of the observer (i.e. buildings) (Kuipers et al., 1993), this process involves two different problems: first, to provide a global path in terms of intermediate subgoals and, second, to travel between consecutive subgoals while avoiding unexpected and moving obstacles along the way. The latter problem has received great attention in the literature, with important contributions ranging from completely reactive techniques (Khatib, 1986; Borenstein and Koren, 1991) to planned trajectories that take into consideration different constraints: non-holonomic kinematics (Canny, 1988; Lozano-Pérez, 1983), dynamics (Latombe, 1991), time (Hu et al., 1993), etc.

On the other hand, the problem of global path generation is tightly related to that of decision-making. Provided that a graph-based model of the environment is available, this problem is usually addressed on the basis of some kind of optimal graph search by using a certain cost function (Hart et al., 1968; Dijkstra, 1959).

Usually, the cost function depends on a unique cost variable, e.g. the distance travelled or the time spent. More sophisticated planners combine these with other factors; e.g. energy consumption, safety, or the robustness of the path indicated by its clearance (Hu and Brady, 1997; Stentz and Hebert, 1995), the probability of encountering moving obstacles (Fujimura, 1995; Simmons et al., 1997), etc. These approaches suffer from the following limitations:

- The set of factors or variables to be considered for the cost function are of such differing natures that, in practice, they cannot be combined in an easy and intuitive way. Typically, the function is built up as a weighted linear combination of them. The weights are chosen on an ad-hoc basis, and cannot be applied to a wide range of situations.
- In many missions it is not important for the mobile robot to optimize the value of the cost function, but to satisfy some restrictions on the cost variables, for

* Corresponding author. Fax: +34-5-213-13-97.
E-mail address: lawrence@lcc.uma.es (L. Mandow)

0952-1976/99/$ - see front matter
PII: S0952-1976(99)00018-4

Let s be the initial node and Γ the set of destination nodes.

1. **CREATE**
   - A list of paths OPEN with $(s, \boldsymbol{F}^s)$. ;where $\boldsymbol{F}^s$ may have arbitrary values
   - An empty list of nodes CLOSED.
   - An acyclic search graph SGRAPH with the initial node s at its root.

2. **NODE SELECTION.**

2.1. If OPEN is empty, then return failure.

2.2. Select a pair $(n_1\ \boldsymbol{F}_1)$ from OPEN which minimizes the value of $d_1(\boldsymbol{F})$. If there are several such pairs, break ties succesively selecting the ones that minimize $d_2(\boldsymbol{F})$, $d_3(\boldsymbol{F})$, ... $d_n(\boldsymbol{F})$. If several pairs still have equal values for all $d_i(\boldsymbol{F})$, arbitrarily choose one of them with non-dominated $\boldsymbol{F}$.

2.3. Close(n_1).

3. **BODY.**

3.1. If $n_1 \in \Gamma$

then return a path in SGRPAH from n_1 to s with cost vector equal to $\boldsymbol{F}_1$.

3.2. If $n_1 \notin \Gamma$

3.2.1. Expand node n_1, generating the set SUC of all its successors in G that will not produce cycles in SGRAPH.

3.2.2. For each node $n_2 \in$ SUC do the following,

a) If n_2 is a new node (i.e. it is not in SGRAPH), then
   - Set a pointer in SGRAPH from n_2 to n_1.
   - Calculate G-SET(n_2) and F-SET(n_2)
   - Open(n_2)

b) If n_2 is in SGRAPH, and new efficient paths to n_2 have been found, then
   - If no pointer exists in SGRAPH from n_2 to n_1, then set a pointer from n_2 to n_1.
   - For all nodes n^* pointed by n_2, such that all costs $\mathbf{G}(P)$ of paths P arriving at n_2 from n^* are now dominated, remove the pointer from n_2 to n^*.
   - Calculate the new G-SET(n_2) and update F-SET(n_2)
   - Update OPEN.
     - For all estimated cost vectors $\boldsymbol{F}$ newly added to F-SET(n_2) , add (n_2, $\boldsymbol{F}$) to OPEN.
     - For all estimated cost vectors $\boldsymbol{F}$ now discarded from F-SET(n_2), if (n_2, $\boldsymbol{F}$) is in OPEN then delete (n_2, $\boldsymbol{F}$) from OPEN.

3.3. Go back to step 2 (NODE SELECTION)

Fig. 1. The METAL-A* pseudocode algorithm.

example, to reach a place before a given time (not to optimize the arrival time). This is supported by the way in which human beings solve real-life planning problems. Often, the restrictions are in conflict with each other, and the path found by a classical graph search algorithm, although optimal, does not guarantee to satisfy all of them.

An intelligent mobile robot should be provided with more flexible and realistic mechanisms to plan global paths when operating in large-scale environments. To make the reasoning behind this statement clearer, consider the following examples.

Assume that a mobile robot path planner uses a cost function that is a combination of *path length* and *energy consumption*. These factors may become conflicting, since the shortest path may contain very power-demanding elements, such as ramps or a great number of turns. During a mission, the mobile robot may detect that the battery is below the emergency level, and decide to plan a path to the only available recharging station (the destination). A path provided by a classical optimization technique will be optimal with respect to a given combination of both *distance* and *energy* factors, but it will not guarantee to find a

(probably existing) path that enables the robot to reach the station before the battery becomes exhausted.

Another typical situation arises when a mobile robot must deal with time scheduling. In many real robot applications, there exists the need to coordinate the operation of the robot with other processes (or robots). This leads to temporal restrictions that must be prioritized in order to accomplish the mission successfully. For example, a mobile robot is sent to pick up an object at a given place, and carry it to a delivery point. If the object is not available until a certain time T, it is not worth being at the pick-up place before T, waiting for the object to be ready. Therefore, optimizing the arrival time at the pick-up place as soon as possible would not be required: just arriving at T is necessary. This provides the path planner with a margin within which to accomplish other important objectives such as energy consumption, safety of the path, vehicle degradation, etc.

These examples show that in many situations the robot must plan a path whose cost variables are below (or above) a given value. In Operations Research (OR) terminology, this value is called an "aspiration level", and the variable/aspiration-level pair is called the "goal". Following this terminology, one could say that "energy consumption $\leqslant K$" or "arrival time $\leqslant T_1$" are goals for the planned path. Notice that the word "goal" may also have other meanings, for example, the destination point of a path, etc. In the rest of this paper the term "goal" will always be used with its OR meaning. Formulating the problem as a set of *goals* is a more natural and flexible manner of specifying the requirements for optimizing conflicting criteria.

The approach presented here for global path planning relies on a general algorithm for search problems with lexicographic goals, named METAL-A*, which was first introduced in a previous work by the authors (Mandow et al., 1999). METAL-A* allows the different goals involved in a path-finding problem to be arranged into several prioritized groups, and the best possible solution path to be found according to the assigned priorities.

The organization of the paper is as follows. In Section 2 an informal survey of lexicographical goal path planning is given. Section 3 describes the application of the METAL-A* algorithm to mobile robot path planning (the algorithm is shown in Fig. 1 in pseudocode, and explained in Appendix A). Section 4 presents the experimental results and discussion. Finally, some conclusions are outlined.

2. Search problems with lexicographic goal satisfaction

Multi-Criteria Decision Theory (MDT) is a suitable framework in which to solve the problem of finding paths that must satisfy a set of possibly conflicting objectives. As discussed above, in this paper, objectives are considered in the form of goals, which are formulated as inequalities:

$$\text{function (variable)} \begin{matrix} \leqslant \\ = \\ \geqslant \end{matrix} \text{value}.$$

For example:

time $\leqslant 5$ min .;
distance $\leqslant 100$ m.;
safety $\geqslant 90\%$.

If these goals are grouped into a number of prioritized levels, the problem is called a *search problem with lexicographic goal satisfaction*.

A solution path P that connects a pair of start and destination node locations in the graph is a sequence of nodes $(n_1, n_2, \ldots, n_m)$ such that for two consecutive nodes n_i and n_{i+1} there is an arc between them. The total cost for P is the sum of the costs of each arc of P. The costs for each arc are represented as a vector of positive values:

$$\mathbf{K}(n_i, n_j) = \big(\text{cost}_1(n_i, n_j), \text{cost}_2(n_i, n_j), \ \ldots, \text{cost}_q(n_i, n_j)\big)$$

where $\text{cost}_k(n_i, n_j)$ refers to the cost of traversing the arc (n_i, n_j), measured with respect to the k-th cost variable (i.e. energy consumption, distance, time, etc.).

The total cost vector $\mathbf{K}(P)$ of a path P is defined as the addition of the costs of all the arcs in P,

$$\mathbf{K}(P) = \big(\text{cost}_1(P), \text{cost}_2(P), \ \ldots, \text{cost}_q(P)\big)$$

where

$$\text{cost}_k(P) = \Sigma\ \text{cost}_k(n_i, n_j).$$

A solution path P for a particular problem is said to be *dominated* if there exists another solution path P' that improves at least one component of the cost vector $\mathbf{K}$ of P while not making the others worse. A *lexicographic goal search problem* consists of finding a non-dominated solution path in the graph that satisfies a set of goals grouped into prioritized levels. Each level L is a set of weighted goals in the form

$$\text{cost}_{Li}(P) \leqslant t_{Li}\ ; w_{Li}$$

where $\text{cost}_{Li}(P)$ is some component of the total cost vector of P, $t_{Li} \geqslant 0$ is the corresponding aspiration level and $w_{Li} \geqslant 0$ is the associated weight, i. e., the relative importance given to the satisfaction of the goal in the level L. In this way, the complete problem

Fig. 2. The RAM-2 mobile robot. The METAL-A* algorithm has been implemented and tested in this platform for global path planning.

is stated as:

$$\begin{aligned}
&\text{Level 1: } \mathrm{cost}_{11}(P) \leqslant t_{11}\ ; w_{11}\\
&\qquad \cdots\\
&\qquad \mathrm{cost}_{1r}(P) \leqslant t_{1r}\ ; w_{1r}\\
&\text{Level 2: } \mathrm{cost}_{21}(P) \leqslant t_{21}\ ; w_{21}\\
&\qquad \cdots\\
&\qquad \mathrm{cost}_{2s}(P) \leqslant t_{2s}\ ; w_{2s}\\
&\text{Level } q\text{: } \mathrm{cost}_{q1}(P) \leqslant t_{q1}\ ; w_{q1}\\
&\qquad \cdots\\
&\qquad \mathrm{cost}_{qz}(P) \leqslant t_{qz}\ ; w_{qz}\\
&\qquad \forall i, j\ t_{ij} \geqslant 0.
\end{aligned}$$

The goals of level i are infinitely more important than those of level $i + 1$. This means that goals in level $i + 1$ are only taken into account once all the goals in level i have been fully satisfied.

In this work, an algorithm called METAL-A* is used to solve these type of problems within the field of mobile robot path planning. METAL-A* can be used to find one non-dominated solution path for lexicographic goal-satisfaction problems with an arbitrary number of goals at each priority level. Further details can be found in (Mandow et al., 1999). A brief explanation of the algorithm is given in Appendix A.

3. The RAM-2 multicriteria path planner

A path planner based on the METAL-A* algorithm has been implemented on the RAM-2 mobile robot for navigating in indoor, structured environments (Fig. 2). RAM-2 is equipped with a manipulator and a variety of sensors such as video-cameras, a radial laser rangefinder, a sonar ring, and an odometric system. Two onboard Pentium 120 MHz PC cards run the Lynx real-time operating system, as well as a software system called NEXUS (Fernandez and Gonzalez, 1998), that takes care of the communication and coordination between different modules of the robot's control architecture.

The RAM-2 navigation system relies on two modules:

A METAL-A* path planner, that generates a global path based on the information provided by a graph-based model of the world (Fernandez and Gonzalez, 1997). The nodes of the graph are distinctive places in the buildings where RAM-2 navigates.

A reactive navigator, that drives the robot between locations on the global path, based on the information provided by the sensors (Fernandez and Gonzalez, 1998; Gonzalez et al., 1995).

The RAM-2 path planner takes into account a number of factors when searching for paths in the graph model of the environment. They include the time elapsed and the energy consumed in traversing each arc, as well as an estimation of the probability that an arc will not be blocked. In the following paragraphs, these factors are considered.

3.1. Clearance

Due to the dynamic characteristics of the environment, some possible paths may be blocked with a certain probability, i.e., some doors may frequently be closed, while others are almost always open. Obviously, if the selected path is blocked along the way, the robot will need to replan its path and take more time to reach its destination. A probabilistic measure of the overall clearance of a path can be defined as the product of the clearance probability of

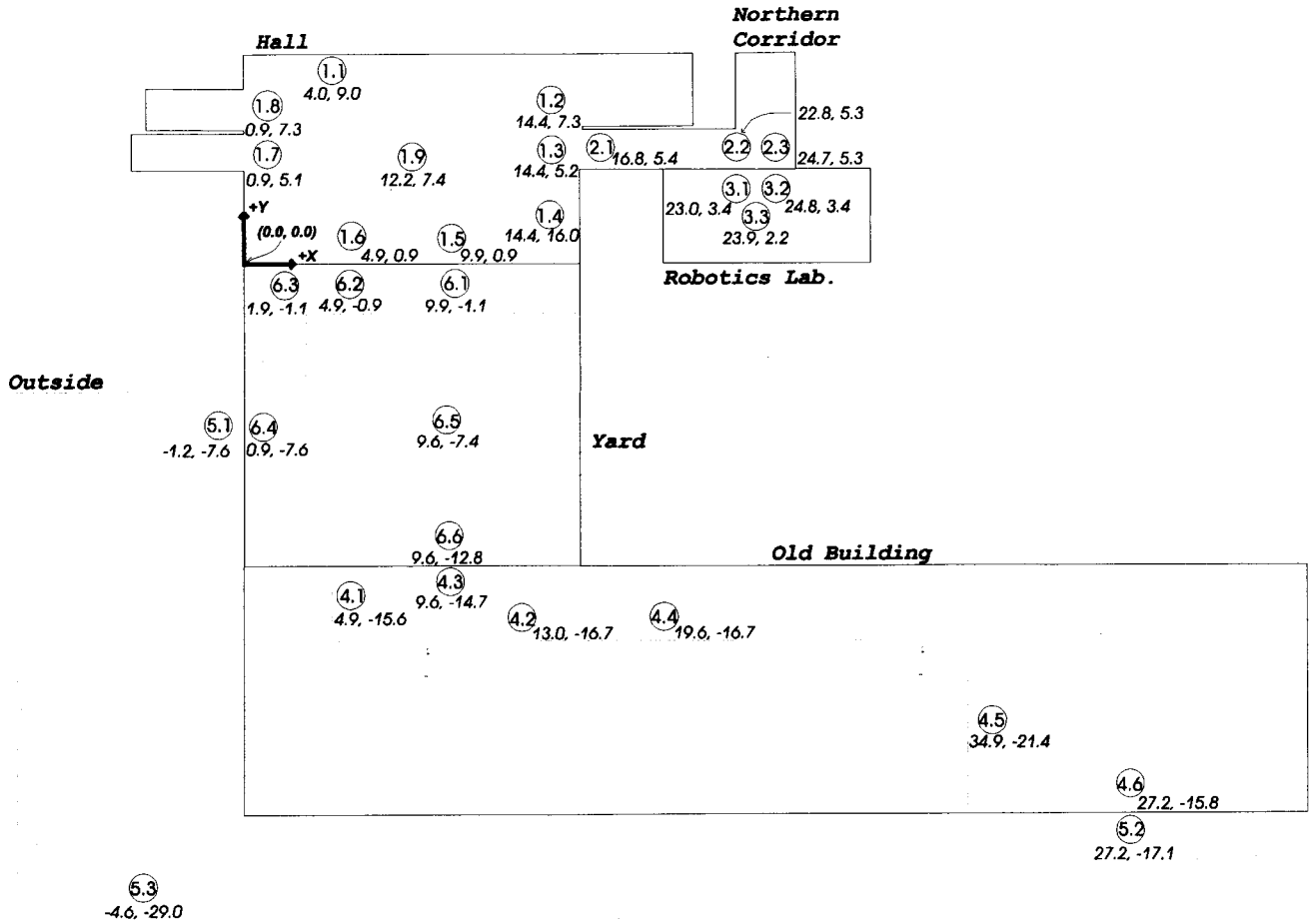

Fig. 3. A 2-D map of the first floor of the Electrical Engineering building, where RAM-2 navigates. The distinctive places that appear in the graph model of the environment are indicated as labelled circles. Their actual coordinates with respect to a global reference frame are also shown.

its component arcs, assuming conditional independence among them.

3.2. Time

Minimizing the time, or at least reducing it to reasonable limits, is an essential and straightforward requirement in any robotic task. It has been included as the second goal.

3.3. Energy consumption

Finally, an additional factor has been added to keep battery consumption low. This ensures that the robot will reduce its battery consumption whenever this reduction does not conflict with the satisfaction of previous goals. Thus the undesirable and slow battery recharge operation is postponed for as long as possible.

This can be summarized with a goal formulation as follows:

Priority level 1: $p_{\text{free}}(Path) \geqslant K_f$
Priority level 2: $\text{time}(Path) \leqslant K_t$
Priority level 3: $\text{battery\_consumption}(Path) \leqslant K_c$

i.e. try to ensure that reasonably safe paths will be found (above K_f% free), and then try to take the robot to its destination quickly enough (in less than K_t seconds) and consuming less than K_c units of energy. Notice that K_f, K_t and K_c are values that can be easily specified for each navigational task, depending on the internal status of the robot or on explicit operator commands.

The implementation of the first goal ($p_{\text{free}}(Path) \geq K_f$) requires some special considerations. In order to keep the clearance probabilities above a certain level, the planner should maximize a multiplicative cost function. However, since the implementation of METAL-A* has been designed following A*, it can only solve problems by minimizing

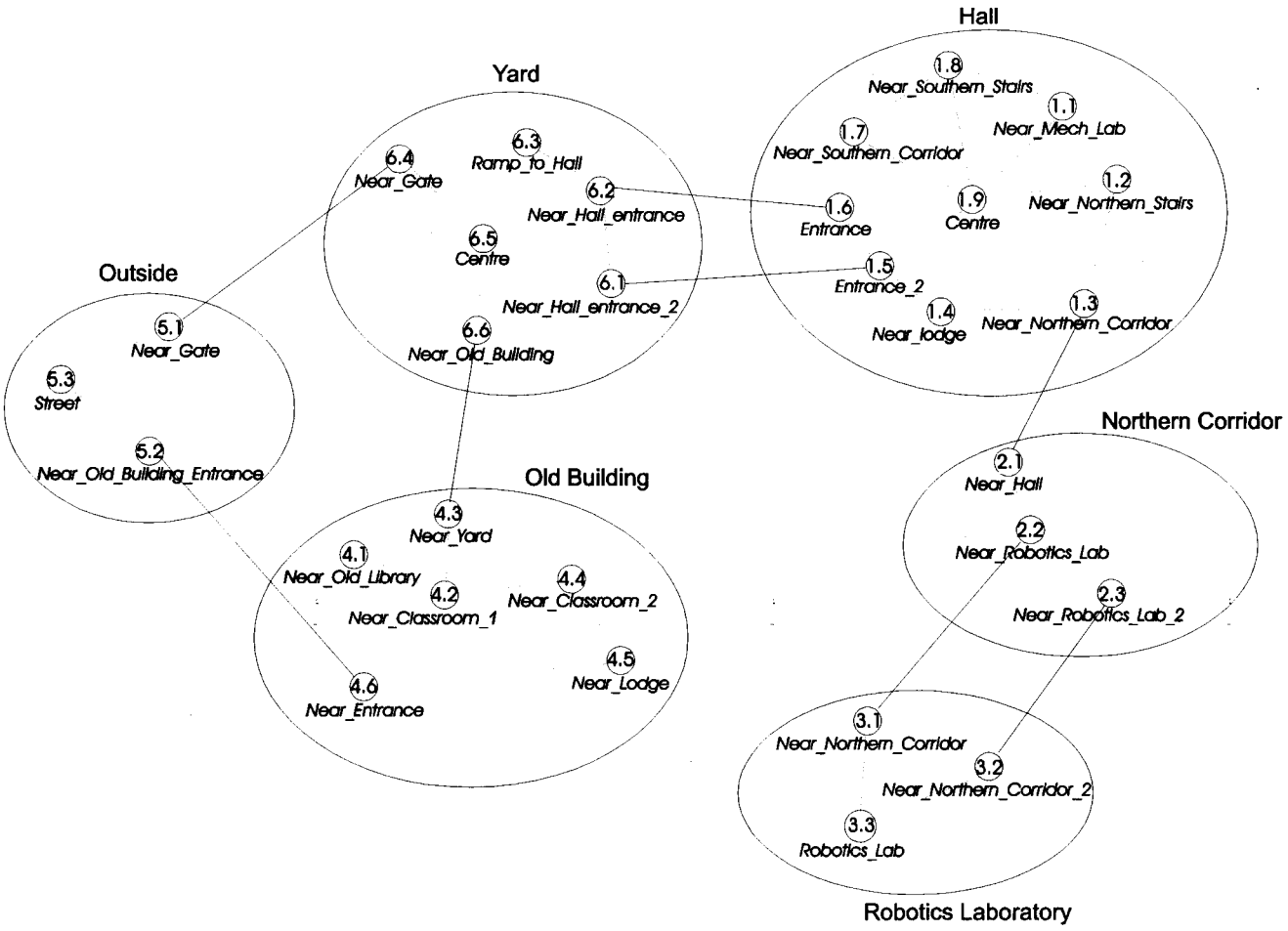

Fig. 4. Graph model of the building. The nodes represent distinctive locations for navigation.

additive cost functions. The original goal has been reformulated to fit the search algorithm.

Let $p_{\text{free}}(P)$ be the probability that the path P is free, and let $p_{\text{free}}(i)$ be the probability that each arc i of the path P is free. If the probabilites $p_{\text{free}}(i)$ are conditionally independent, then:

$$p_{\text{free}}(P) = \prod_i p_{\text{free}}(i) \quad \forall \text{ arc } i \in P.$$

The former multiplicative function can be changed to an additive cost function using logarithms:

$$\max\ p_{\text{free}}(P) \equiv \max \prod_i p_{\text{free}}(i) \equiv \max\ \exp\left[\sum_i \ln(p_{\text{free}}(i))\right]$$

Since $0 \leqslant p_{\text{free}}(i) \leqslant 1$, then $-\infty \leqslant \ln(p_{\text{free}}(i)) \leqslant 0$, and:

$$\max\ p_{\text{free}}(P) \equiv \min \sum_i \left|\ln(p_{\text{free}}(i))\right|.$$

This function can be used whenever the following reasonable assumption holds,

$$p_{\text{free}}(i) \neq 0 \quad \forall \text{ arc } i \in G$$

i.e. no arc in the graph G is permanently blocked.

If, for instance, $K_f = 90\%$, the problem is reformulated, so transforming the goal

$$p_{\text{free}}(P) \geqslant 0.9$$

to

$$\sum_i \left|\ln(p_{\text{free}}(i))\right| \leqslant \left|\ln(0.9)\right| = 0.10536$$

and using $\left|\ln(p_{\text{free}}(i))\right|$ as an additive cost measure for each arc i in the graph.

4. A real experiment

This section presents an example of the application of the METAL-A* algorithm to the navigation of a mobile robot. In this section a particular experiment is described, and its results are compared to those that

node1	node2	$\|\ln(p_{free})\|$	Time (seconds)	batt. cons. (Kjoules)
1.4	1.3	0	2.5	1.82
1.9	1.2	0	10.1	7.42
1.4	1.5	0	11	8.12
1.8	1.7	0	7	5.18
1.8	1.1	0	3.2	2.38
1.1	1.2	0	18.6	13.72
1.2	1.3	0	11.6	8.54
1.5	1.6	0	3.7	2.8
1.6	1.9	0	4.2	3.08
1.9	1.7	0	14.3	10.5
1.8	1.9	0	15	11.06
1.5	1.9	0	4.1	2.8
1.9	1.1	0	14.4	10.1
2.1	2.2	0	13.5	9.94
2.2	2.3	0	2.1	1.54
3.1	3.3	0	7.5	5.6
3.3	3.2	0	6.7	4.9
4.6	4.5	0	5.1	3.78
4.5	4.4	0	10.6	7.84
4.4	4.2	0	6.3	4.62
4.2	4.3	0	4	2.94
4.2	4.1	0	8.2	6.02
5.1	5.3	0	9.2	7
5.3	5.2	0	20.3	14.98
6.6	6.5	0	4.6	3.36
6.5	6.4	0	4.4	3.22
6.4	6.3	0	6.8	5.04
6.3	6.3	0	9.1	6.72
6.2	6.1	0	3.7	2.66
5.1	6.4	0.061875	3.9	2.8
6.2	1.6	0.051293	5.9	4.2
6.1	1.5	0.030459	5.9	4.2
5.2	4.6	0.020202	4	2.94
4.3	6.6	0.010050	6.8	5.04
1.3	2.1	0	1.9	1.4
3.1	2.2	0.030459	4.9	3.64
3.2	2.3	0.051293	5.5	4.06

Fig. 5. Table of costs for each arc in the graph. Clearance probability, time elapsed and energy consumption are shown.

would have been obtained by a conventional optimization-based path planner.

Figs. 3 and 4 show the 2-D map and its corresponding graph-based model, respectively, used by RAM-2 to navigate its way through the first floor of the Electrical Engineering buildings at the University of Malaga. The particular experiment presented here considers the navigation of RAM-2 from the location *Outside.Street* (node 5.3) to *Robotics_Lab.Robotics_Lab* (node 3.3).

Fig. 5 shows the costs associated with the arcs in the graph for each additive cost function. Values for the costs concerning time and battery consumption have been estimated by averaging a number of experiments where RAM-2 was manually guided along the arcs. On the other hand, the values for clearance probabilities were initially set to 1 for all the arcs except for those involving passing through doors, which were set to a priori values based on knowledge of the environment. All these values are updated dynamically by the robot as it traverses the buildings during its normal operation.

An ideal solution path should achieve the optimum value for all three individual factors considered (clearance probability, time, and battery consumption). Unfortunately, an ideal solution is rarely reached, since these objectives are usually conflicting. The aspiration level for the first cost variable has been set

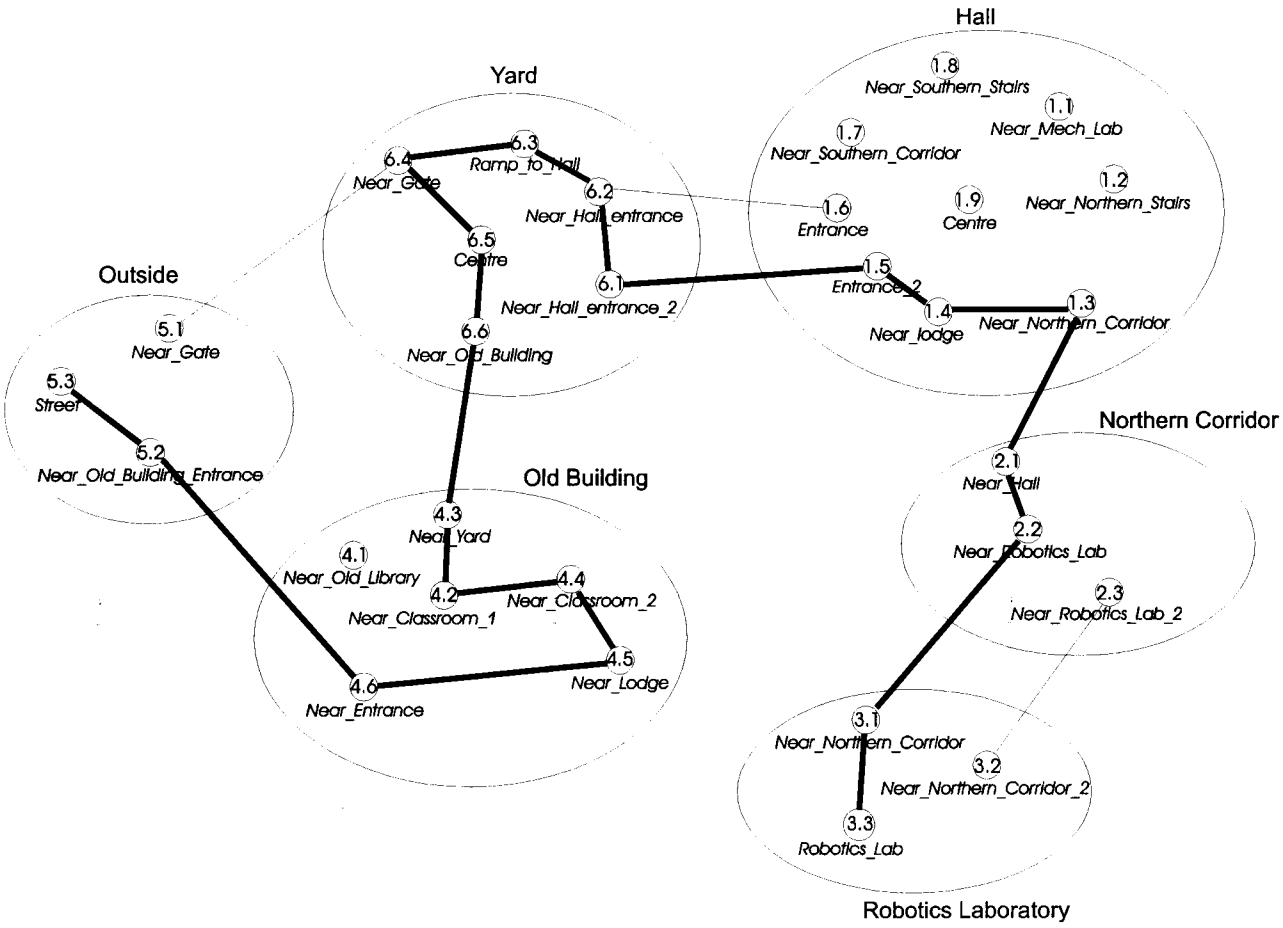

Fig. 6. Non-dominated solution path found by the METAL-A* algorithm going from the *Robotics Laboratory* to the *Street*.

to a 90% chance that the path will be clear. Reasonable levels for the time and battery consumption factors are obtained using the Euclidean distance from the origin to the destination. Both attributes are roughly proportional to the distance traversed by the robot, though significant variations are possible, and justify separate treatment.

More precisely, if the Euclidean distance from origin to destination is Dist (meters) and the robot's maximum speed is $\max_{speed}$, (m/s) then it will be impossible to reach the destination in less than $\text{Dist}/\max_{speed}$ (s). Similarly, if the robot's minimum battery consumption rate is $\min_{consumption}$, (kJ/m) then at least $\text{Dist} \times \min_{consumption}$ (kJ) are needed to reach the destination. Aspiration levels of 150% of these optimistic values are used for the time and battery consumption goals in the experiment. Euclidean distance is measured from the map of the buildings.

Thus, for the RAM-2 mobile robot and the particular experiment at hand,

$\max_{speed}$	$= 1.7$ m/s
$\min_{consumption}$	$= 0.7$ kJ/m
Eucl_dist(node-5.3, node-3.3)	$= 132.2$ m.

The problem goals are formulated as

Priority level 1:Path clearance probability	$\geqslant 0.9$
Priority level 2:Path time (in s)	$\leqslant 116.6$
Priority level 3:Path battery consumption (in kJ)	$\leqslant 138.8$.

A problem that always arises when using a heuristic search is the determination of the heuristics. If the robot had a geometric map of the environment, although approximate, it could employ the Euclidean distance to compute time and battery consumption heuristics. However, as RAM-2 is not intended to have such a detailed geometric representation, no heuristic information has been used to guide the planner, i.e. METAL-A* behaves as an uninformed search algorithm.

4.1. Results

The results obtained for solving the above sample problem can be summarized as follows.

The planner found the only non-dominated solution to this goal problem (see Fig. 6). The corresponding values for each variable are,

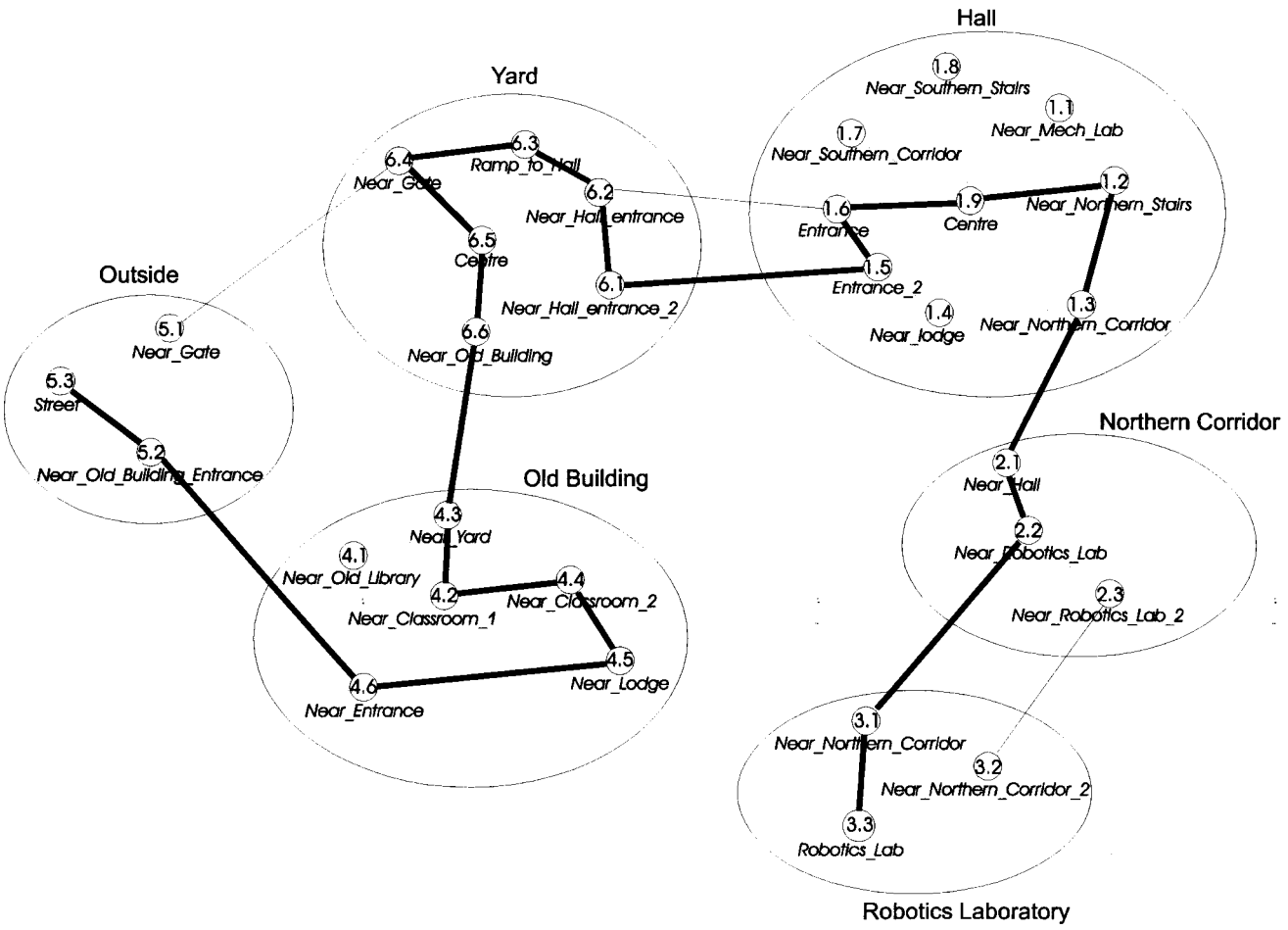

Fig. 7. An example of a path that minimizes the clearance probability.

Clearance probability	= 0.9128
Time	= 132.9 s
Batt. consumption	= 97.86 kJ.

The deviations achieved for each priority level are (0, 16.3, 0) respectively. It is clear that the time goal could not be achieved in 16.3 s, but nevertheless the planner finds the fastest path that preserves satisfaction of the clearance goal. Although there is little margin left for the third goal, METAL-A* recognizes that there are several paths that satisfy the first goal while taking 132.9 s, and returns the path with the least battery consumption.

The solution to this problem is clearly different from those found by path planners that optimize a single criterion.

- In the above problem there are many paths that maximize the clearance probability. These are equally good to an A*-based planner and hence any of them could be returned as a solution. Some of them, however, are clearly undesirable, like the one shown in Fig. 7.
- The paths that minimize the time for this problem are shown in Fig. 8. Depending on implementation considerations, A* may find either. These, however, have different battery consumption and clearance probabilities, a distinction that is necessarily lost in single-objective formulations. Analogous reasoning can also be used regarding the battery consumption criterion.

Note also that there is no solution to this problem that satisfies all the goals. If the problem criteria had been formulated as constraints and solved again using a conventional optimizing planner, the result would have been a failure.

Sometimes problem constraints are inherent to a problem, and cannot possibly be violated (i.e. physical constraints like "the robot's speed cannot be faster than the speed of light", or "the battery consumption for any mission can never be greater than the maximum battery capacity"). However, most everyday constraints can be viewed as "soft" in the sense that their violation only results in more inconvenient or less preferred solutions. METAL-A* provides a simple framework to incorporate these so-called "soft" constraints as goals (i.e. in the less important priority levels). As a

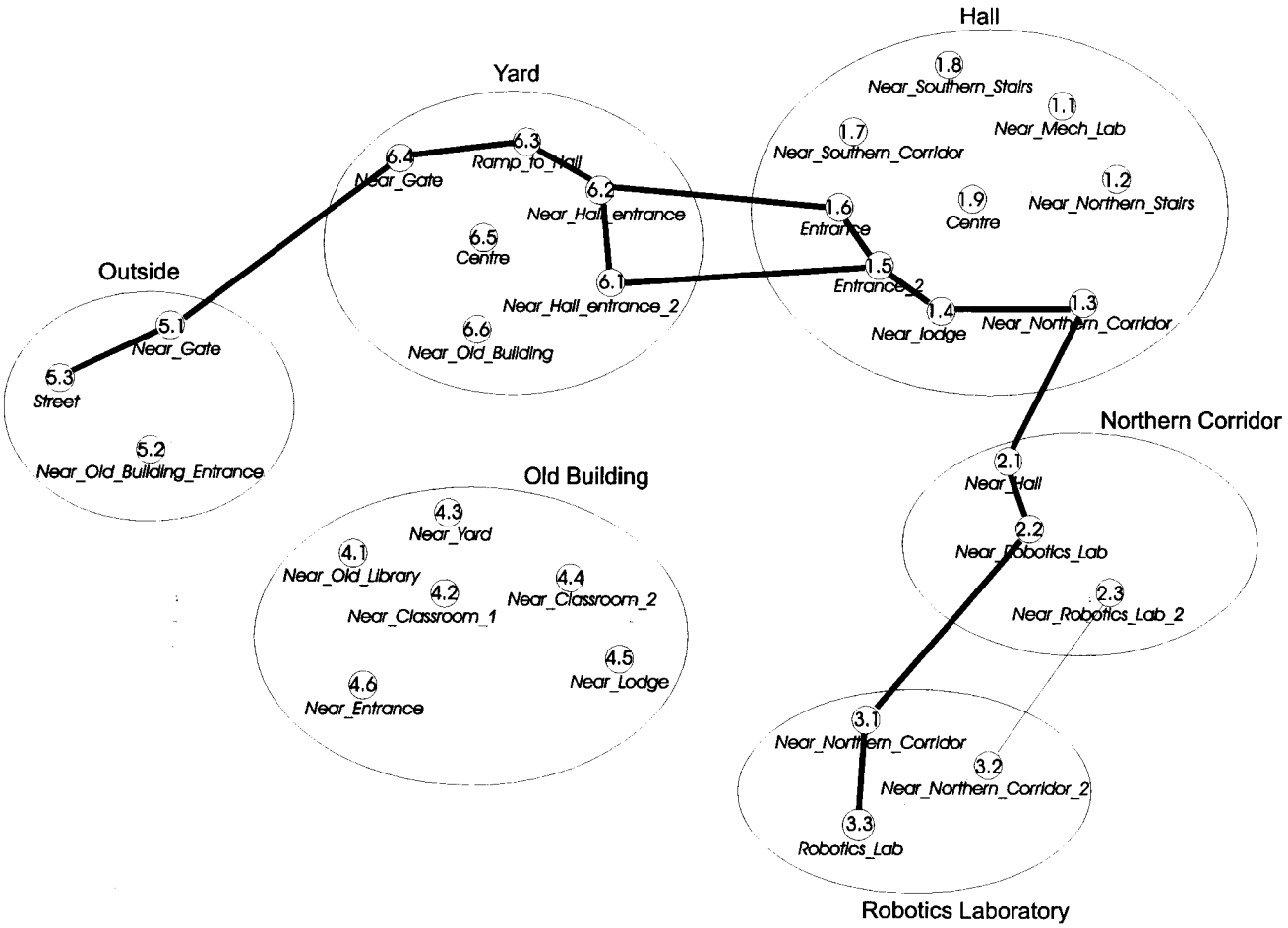

Fig. 8. Paths which minimize time, found by a conventional A* algorithm.

result, constraint violation never results in failure, but results in solutions that at least try to achieve the goals as well as possible, i.e. METAL-A* (and goal-satisfaction algorithms in general) enables elegant handling of overconstrained problems.

5. Conclusions

In many practical situations it may not be appropriate to state mobile robot path planning as a conventional optimization problem. What is important is to avoid fuel exhaustion, not to save a marginal joule; to reach the destination in time, not a millisecond before.

In this paper, a multicriteria path planner for mobile robot navigation in large-scale space has been presented. In particular, the METAL-A* algorithm has been implemented in the RAM-2 mobile robot. METAL-A* searches for optimal paths that satisfy the greatest number of a lexicographically ordered set of goals. This approach improves upon conventional optimization techniques in several ways. It allows conflicting objectives to be dealt with, and provides more comprehensive path planning. Besides this, it can also result in a more efficient procedure: the addition of new goals can play an important role in reducing the search effort while guiding the planner to the most convenient paths, even in the absence of heuristic information.

Appendix A

A1. METAL-A Algorithm*

METAL-A* is basically a "multicriteria best-first" algorithm: a partial solution path that "best" satisfies the goals of the problem is selected and expanded in each iteration until a solution path is selected. Only its fundamental details are described here. For a more complete reference see (Mandow et al., 1999).

The algorithm uses three typical data structures:

- SGRAPH: a directed acyclic search graph that records all non-dominated paths found from the start node to the expanded nodes.
- OPEN: a list of partial solution paths in SGRAPH

that can be further expanded.
- CLOSED: a list of nodes already explored.

A significant difference between A* and METAL-A* is the use of a vector cost evaluation function for each path. The cost of each path P is a vector $\mathbf{G}(P)=\mathbf{G}^P(n)=(g^P{}_1(n),\ g^P{}_2(n),\ldots,g^P_q(n))$. Each component in $\mathbf{G}(P)$ stands for some of the individual additive costs functions considered.

The graph SGRAPH records all non-dominated paths that reach known nodes. A set G-SET(n) is used for each node to record all non-dominated cost vectors $\mathbf{G}^P(n)$ corresponding to paths $P=(s,\ldots,n)$ included in SGRAPH.

METAL-A* allows the use of heuristic information to guide search. A multicriteria heuristic function $H(n)$ is a function that returns a nonempty finite set of cost vectors $\mathbf{H}(n)$ which estimate the cost of some path from n to a goal node.

$$H{:}n\rightarrow\{\mathbf{H}(n),\ \mathbf{H}'(n),\ \ldots\}.$$

A function $H(n)$ is said to be admissible if for all non-dominated paths from n to any destination node there is a cost vector $\mathbf{H}(n)$ in the returned set that dominates or equals its cost. Therefore, for each node n in SGRAPH, a set of vector cost estimates for all known non-dominated paths arriving at n can be calculated as follows:

$$\text{F-SET}(n)=\text{nodom}(\{\mathbf{F}(n)=\mathbf{G}(n)+\mathbf{H}(n)/\mathbf{G}(n)\in\text{G-SET}(n)\ \text{and}\ \mathbf{H}(n)\in H(n)\})$$

where nodom(A) gives the set of non-dominated vectors of the set A without repetitions.

The algorithm uses the estimated cost vectors to decide which node to expand at each iteration. All paths in SGRAPH that can be further expanded are represented in the OPEN list. Each element in the list is a pair of the form:

(n, $\mathbf{F}$), where n is a node in the graph and $\mathbf{F}\in$ F-SET(n),

i.e. any generated but not yet expanded node will appear in OPEN as many times as vectors $\mathbf{F}$ exist in F-SET(n). The operations OPEN(*node*) and CLOSE(*node*) are defined in the following way:

- OPEN(n_1): add to the list OPEN a pair (n_1, $\mathbf{F}$) for each vector $\mathbf{F}$ in F-SET(n_1).
- CLOSE(n_1): delete all pairs (n_1, ·) from OPEN and add node n_1 to the list CLOSED.

At each iteration a pair (n, $\mathbf{F}$) that minimizes goal deviations according to their priorities is selected from OPEN. METAL-A* accepts several ways to minimize the deviation from the problem goals. The one used here is:

$$d_i(F)=\sum_{j=1}^{q_i} w_{ij}\times\max\ (0,f_{ij}-t_{ij})$$

where q_i is the number of goals in the level i, w_{ij} is the weight of the goal j in the same level, f_{ij} the cost function for that goal, and t_{ij} is its aspiration level (see Section 2). Thus the deviation for each priority level i is measured as the weighted sum of positive deviations of each cost function f_{ij} from its aspiration level t_{ij}. Note that if $f_{ij}\leqslant t_{ij}$ then no deviation is measured, since the goal is satisfied.

It can be shown that METAL-A* is complete and admissible when used with an admissible multicriteria heuristic function, i.e. the first solution found is non-dominated and satisfies the goals in the best possible way according to their priorities.

References

Borenstein, J., Koren, Y., 1991. The vector field histogram-fast obstacle avoidance for mobile robots. IEEE Trans. R&A 7, 278–288.

Canny, J., 1988. The Complexity of Robot Motion Planning. The MIT Press, Cambridge, MA.

Dijkstra, E.W., 1959. A note on two problems in connexion with graphs. Numerische Mathematik 1, 269–271.

Fernandez, J.A., Gonzalez, J., 1997. A general world representation for mobile robot operations. In: Seventh Conference of the Spanish Association for the Artificial Intelligence (CAEPIA'97), Malaga, Spain.

Fernandez, J.A., Gonzalez, J., 1998. NEXUS: A flexible, efficient and robust framework for integrating the software components of a robotic system. In: IEEE International Conference on Robotics and Automation (ICRA'98), Leuven, Belgium.

Fujimura, K., 1995. Time-minimum routes in time-dependent networks. IEEE Trans. on Robotics and Automation 11, 343–351.

González, J., Stentz, A., Ollero, A., 1995. A mobile robot iconic position estimator using a radial laser scanner. Journal of Intelligent and Robotic Systems 13, 161–179.

Hart, P.E., Nilsson, N.J., Raphael, B., 1968. A formal basis for the heuristic determination of minimum cost paths. IEEE Transactions on System Science and Cybernetics SSC-4, 100–107.

Hu, H., Brady, M., 1997. Dynamic global path planning with uncertainty for mobile robots in manufacturing. IEEE Trans. on Robotics and Automation 13, 760–767.

Hu, T.C., Kahng, A.B., Robins, G., 1993. Optimal robust path planning in general environments. IEEE Trans. on Robotics and Automation 9, 775–784.

Khatib, O., 1986. Real-time obstacle avoidance for manipulators and mobile robots. Int. Journal Robotics Research 5, 90–98.

Kuipers, B.J., Froom, R., Lee, W-Y., Pierce, D., 1993. The semantic hierarchy in robot learning. In: Connell, J., Mahadevan, S. (Eds.), Robot Learning. Kluwer Academic Publishers, Boston, pp. 141–170.

Latombe, J.C., 1991. Robot Motion Planning. Kluwer Academic, Boston, MA.

Lozano-Pérez, T., 1983. Spatial planning: a configuration space approach. IEEE Trans. Comput C32, 108–120.

Mandow, L., Pérez de la Cruz, J.L., 1999. A heuristic search algorithm with lexicographic goals. Engineering Applications of Artificial Intelligence (in press).

Simmons, R.G., Goodwin, R., Haigh, K.Z., O'Sullivan, J., Veloso,

M., 1997. Xavier: experience with a layered robot architecture. Sigart Bulletin, Fall.
Stentz, A., Hebert, M., 1995. A complete navigation system for goal acquisition in unknown environments. Autonomous Robots 2 (2 August).

Juan A. Fernández-Madrigal was born in Córdoba (Spain) in 1970. He received the M.S. degree in computer science from University of Málaga, Spain, in 1994. He currently has a fellowship from the Ministerio de Educacion y Ciencia of the Spanish Government and is completing the Ph.D. degree in Computer Science at University of Málaga. His research interests include world modeling and control architectures for autonomous mobile robots.

Javier Gonzalez received the B.S. degree in Electrical Engineering from the University of Sevilla in 1987. He joined the Department of "Ingeniería de Sistemas y Automática" at the University of Málaga in 1988 and received the Ph.D. from this University in 1993. Since July 1990 until July 1991 he was at the Field Robotics Center, Robotics Institute, Carnegie Mellon University (USA) working on mobile robots. Currently he is an associate professor at the University of Málaga and is leading a project on mobile robots funded by the Spanish Government. His research interest includes mobile robot autonomous navigation, world modeling, computer vision and laser-based range sensing. Dr. Gonzalez is member of the IEEE, IFAC and IAPR.

Lawrence Mandow received his M.S. degree in Computer Science from the University of Málaga (Spain) in 1992. He was awarded a post-graduate research fellowship and joined the Department of Computer Sciences and Languages where he is currently Profesor Asociado and Ph.D. candidate. His research interests include the development of multicriteria problem solving procedures for Artificial Intelligence and their application to architectural and engineering design problems.

J.L. Pérez de la Cruz holds the degrees of Ingeniero and Doctor Ingeniero de Caminos, Canales y Puertos from the School of Madrid. He is currently an associate professor at the University of Málaga, where he lectures on Artificial Intelligence. He has co-authored more than a dozen papers in international journals and conferences. His main research interests are currently reasoning methods for design tasks and intelligent tutorial systems.