Proceedings of the First

Euromicro Workshop on Advanced Mobile Robots (EUROBOT '96)

Kaiserslautern, Germany
October 9 – 11, 1996

Sponsored by Deutsche Forschungsgemeinschaft (DFG)

IEEE COMPUTER SOCIETY 50 YEARS OF SERVICE · 1946-1996

The Institute of Electrical and Electronics Engineers, Inc.

Multi-strategic Approach for Robot Path Planning

Bianco G., Cassinis R.
Dept. of Electronic for Automation - University of Brescia - Italy
Via Branze 38, I-25123 Brescia
Tel. +39-30-3715.453
Fax +39-30-3715.469
E-Mail: bianco@chiostro.univr.it, cassinis@bsing.ing.unibs.it

Abstract

The paper presents a proposal for an autonomous robot path planning system that uses several strategies to reach a target also in an a-priori unknown environment.
The proposed method has learning capabilities that allow the robot to take advantage of previous experience, thus improving its performance during successive traveling in the same environment.
The proposed multistrategic approach has been tested both on a software simulator and on a real robot

1.- Introduction

Due to the high computational complexity of the General Mover's Problem [Reif 79], the whole motion planning research community had to constrain the problem resolution to specific robot and environmental hypotheses [Hwang, Ahuja 92].

In this way, every strategy belongs to a particular classical group: Classical Mover's Problem, Circular (or Point) Robot, Manipulator Motion Planning, Multi Mover's Problem, Time Varying Environment, Constrained Motion Planning, Movable Object.

Therefore, each navigation strategy works well in a particular environment and with a particular robot

We can then assume that, if a given strategy has to operate in an environment and/or on a robot different from the ones it was originally designed for, there will always be another strategy that has better performance in the new situation.

For these reasons we introduce the concept of multi-strategic approach: always use the best available strategy to plan the path taking into account environmental and robot characteristics. During the path, the multi-strategy remembers which strategy has been used to plan the specific path measuring also its efficiency; this is quite different from the classical approach since instead of a "quantitative" learning (i.e. learning the geometric path) we use a "qualitative" learning (i.e. learning the used strategy).

To test this theory we wrote a multi-strategic simulator in C++, and we tested the multi-strategic approach on the field on a Robuter robot.

2.- The state of the art

The classical approach to path planning considers several strategic groups as described in [Hwang, Ahuja 92] formalized by the tree of figure 1. Each group has particular characteristics:

- *Classical Mover's Problem*: it is the problem of a rigid robot traveling in a known environment among stationary obstacles of fixed shape. The following papers, [Lozano-Perez, Wesley 79], [Brooks 83] are important contributions.
- *Point or Circular Robot*: the robot shape is circular, therefore its orientation is unimportant and the robot can be studied as a point. Classical works are in [Lumelsky 89] and [Krogh, Thorpe 86].
- *Manipulator*: the robot is an open kinematics chain with a fixed base. Important strategies are in [Barraquand, Latombe 90] and [Lozano-Perez 87].
- *Multiple Robot (Multimover's Problem):* in this group the strategies have to plan the path in the presence of several robots. Important works are due to [Schwartz, Sharir 83] and [Erdmann, Lozano-Perez 86].
- *Time Varying Environment:* the environments considered in this group change as time passes. In [Reif, Sharir 85] and [Kant, Zucker 88] we can find an approach to this situation.
- *Constrained Motion Planning*: Kino-dynamic and

0-8186-7695-7/96 $5.00 © 1996 IEEE
Proceedings of EUROBOT '96

holonomic constraints affect path planning. In [Dubin 57] and [Fortune, Wilfong 88] there are important contributions to holonomic path planning while in [Krogh, Thorpe 89] and in [Canny et al. 88] we find contributions to kino-dynamic path planning.

- *Movable Object Problem*: in this group the robot can move obstacles to plan the path. In [Lozano-Perez et al. 89] we can find a classical approach.

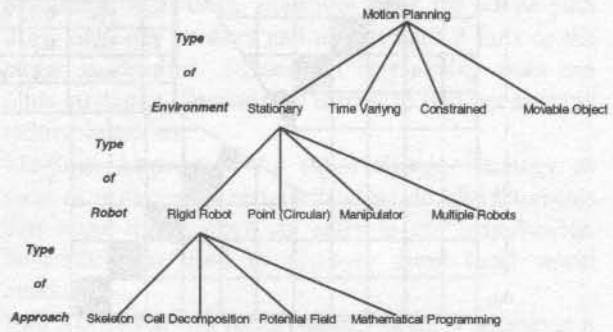


Figure 1: the classification tree

3.- The proposed multi-strategic solution

As stated before, every strategy has been designed to operate with specific characteristics in mind. For example, in [Lozano-Perez, Wesley 79] the described strategy operates in stationary environments with polygonal obstacles, a single robot (not a manipulator) with polygonal base and with total knowledge of the environment.

In [Lumelsky 89] the strategy assumes stationary environments, arbitrary obstacle shape, only one robot (not a manipulator) with a circular base shape and with a local knowledge of the environment.

Most likely, Lumelsky's strategy could operate in the same environments as those of Lozano-Perez and Wesley ones but, probably, with reduced efficiency: this is because the Lozano-Perez and Wesley strategy can optimize the path since it globally knows the environment.

With these considerations in mind we could argue the non-existence of an "always optimal path planning strategy". This is also stronger if we consider the strategy not only as a set of movements but also in term of sensoriality.

To formalize our hypotheses, we define a set S whose elements are path planning strategies; we also define a set C whose elements represent, in ways we do not formalize, all the possible world characteristics (environment, obstacle shape, etc.). Finally, we introduce a function η, whose values range from 0 to 1, representing the strategy efficiency to travel from a starting point to a target one. We assume $\eta(Start\ to\ Target)=0$ whenever the strategy applied in this path fails (it does not converge toward the target position or it loops); with $\eta(Start\ to\ Target)=1$ the strategy reaches the target in an optimal way (following the shortest path at the greatest possible speed). In other cases ($0<\eta(Start\ toTarget)<1$) the strategy always reaches the target but not in an optimal way.

Let us assume b as starting point and t as the target one; if $s_n(c,b,t)$ $(s_n \in S, c \in C)$ is the effective planned path, our hypothesis can be formalized as:

$$\forall b,t,\forall s_n \in S \Rightarrow \exists \bar{c} \in C, s_m \in S : \eta(s_m(\bar{c},b,t)) > \eta(s_n(\bar{c},b,t)), m \neq n$$

that means that for each strategy there exists at least one world characteristic whose existence makes another strategy more efficient than the former one. In other words, we assume that there is no absolute optimal strategy.

To operate with the multi-strategic approach we have to divide the environment into local contexts represented by a set of square cells (A_i) whose dimensions are approximately the same as the size of the robot base (see Figure 2).

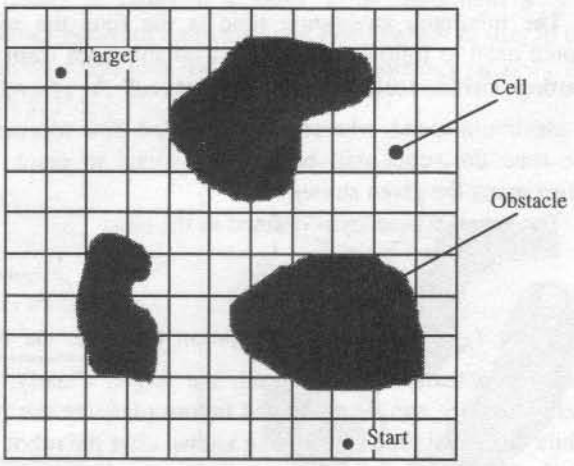


Figure 2. - Partitioning the environment in cells

The whole path, from the starting position to the final target, can be considered as an ordered sequence of not necessarily neighboring cells

$$A_b \rightarrow A_1 \rightarrow \ldots \rightarrow A_n \rightarrow A_t$$

therefore it consists of a sequence of elementary paths

$$A_i \rightarrow A_j$$

where each one can be planned using a different strategy.

The choice of a particular strategy for planning each elementary path is left to the robot, that chooses the most efficient one according to some given information.

The parameters we associate to each strategy, that enable us to compare them each other, are: applicability in

a given local environment, presumable and actual efficiencies.

Applicability criteria for a given strategy are very difficult to define. On the other hand, it is quite easy to state that, for any strategy s_n there is at least one set of characteristics c_{ij} whose existence guarantees that strategy s_n will certainly fail (for example, it is obvious that in a dark environment all vision-dependent navigation algorithms will fail), i.e.

$$\eta(s_n(c_{ij}, A_i, A_j)) = 0$$

where c_{ij} refers to the world characteristics that belong to the effective path, planned by strategy s_n, starting from cell A_i and ending in cell A_j.

The presumable efficiency is defined as the ratio between the minimum navigation time (t_μ) and the estimated navigation time (t_σ). So, for any path $A_i \rightarrow A_j$ a minimum time $t_{\mu_{ij}}$ and an estimated time $t_{\sigma_{ij}}$ will be identified together with the ratio

$$\frac{t_{\mu_{ij}}}{t_{\sigma_{ij}}}$$

The minimum navigation time is the time the robot would need to follow the shortest available path from the starting position (cell A_i) to the target (cell A_j) traveling at maximum speed, whereas the estimated time represents the time the robot will presumably spend to reach the target using the given strategy.

The actual efficiency is defined as the ratio

$$\frac{t_{\mu_{ij}}}{t_{\eta_{ij}}}$$

where $t_{\eta_{ij}}$ is the actual navigation time, i.e. the time that was actually spent to reach the target. Clearly, the estimated time can be computed before planning the path while the actual time can only be known after the robot has reached the target position.

The robot, using the information just described, can make a choice among the strategies it knows, and will choose the strategy that seems the most efficient to go from cell A_i to cell A_j. So, for each known applicable strategy ($\eta(s_n(c_{ij}, A_i, A_j)) \neq 0$) the robot will choose the highest actual efficiency strategy or, if the actual efficiency is unavailable, the highest presumable efficiency strategy.

As it was said before, the robot makes its choice according to the information included in an environmental representation that, in our case, is not an analogue representation of the world.

Our environmental representation is a square matrix whose rows and columns are cells identifiers, respectively starting cells and final position cells; each matrix element is linked with a table that contains a list of all the robot known strategies, together with specific information, as described in the sequel (see figure 3).

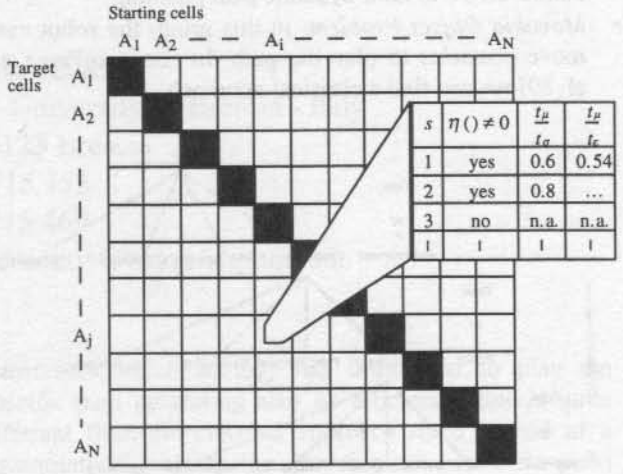


Figure 3. - The matrix for representing paths

As stated before, the robot needs three pieces of information from the surrounding environment; we can use three virtual sensors to compute the data:

- **VSA (Virtual Sensor of Applicability)**: provides a binary response about the applicability of a given strategy in the local environment context.
- **VSPE (Virtual sensor of Presumable Efficiency)**: provides a numeric measure ranging from 0 to 1 given by the ratio

$$\frac{t_\mu}{t_\sigma}$$

- **VSAE (Virtual Sensor of Actual Efficiency)**: Like VSPE, it provides a numeric output given by the ratio

$$\frac{t_\mu}{t_\eta}$$

The virtual sensors can be implemented using general knowledge about the world and/or by using active sensing

In order to compute t_σ and output of VSPE, we might relate it to the most time-consuming robot sensing activities. This can be done either formally (computing the time needed for each sensing activity in a given environmental situation) or statistically, using data from previous applications of the same strategy. To simplify the problem using the simulator, the output of VSPE is provided by a human operator.

We might consider VSA as a set of rules derived from the experience of human operators; but in the simulator, it derives from strategy failure or success. VSAE output is

computed exactly as stated above.

Multi-strategic path planning can be achieved in three different ways, according to the philosophy the robot uses to collect sensory information about the local environment. These different approaches imply different path generation methods, therefore, the planned paths are different in terms of complexity and efficiency. The three possible approaches are:

- **Minimum Approach**: the robot plans the whole path using only one strategy and uses it until it fails or the target is reached. Subsequent re-planning may use other strategies. Sensoriality is used to find out strategy failure situations.
- **Medium Approach**: the robot changes strategy as soon as it discovers certain local world characteristics that make the strategy currently in use inapplicable. Sensoriality is used to discover these local world conditions.
- **Full Approach**: the robot changes strategy whenever it finds, in any local environment, that another strategy has a better presumable efficiency than the one currently in use. Sensoriality is heavily used to compute, in each cell, the efficiency of each robot known strategy.

The chosen approach for the software simulator and the robot test is the minimum one.

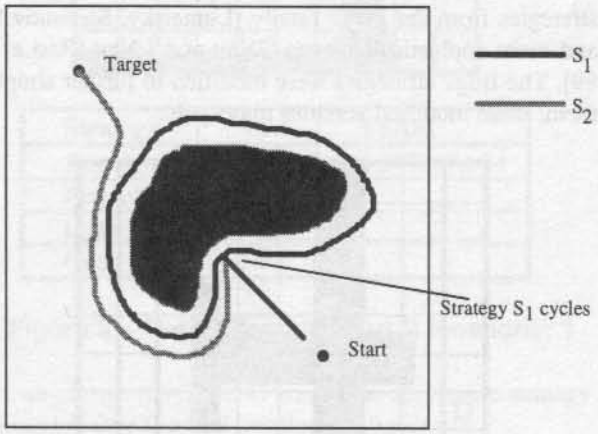


Figure 4. - Changing a strategy when a loop is discovered

It is obvious that once data are stored in any matrix element, they can be used each time the robot traverses that particular element, provided that environmental changes are limited.

In figures 4, 5, 6 we consider three examples about multi-strategic approaches. In figure 4 strategy s_1 is used until it fails (in this case it falls into an endless loop); then the robot uses another strategy (s_2).

In figure 5 strategy s_1 is used until discovers that it is not longer applicable; then the robot uses another strategy (s_2).

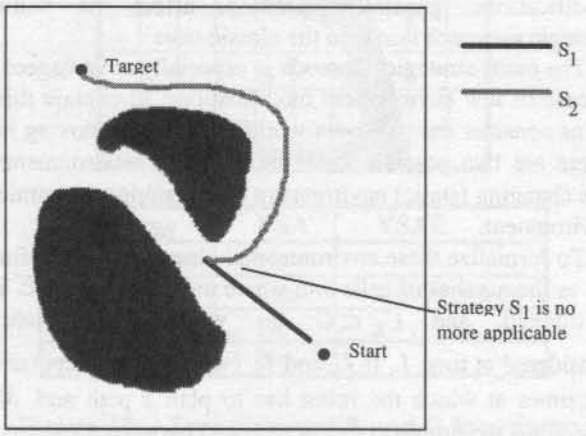


Figure 5. - Changing strategy when it fails

Lastly in figure 6 strategy s_1 is used until another strategy becomes more efficient than the one in use; in this case the robot uses the most efficient strategy (first s_2 and then s_3).

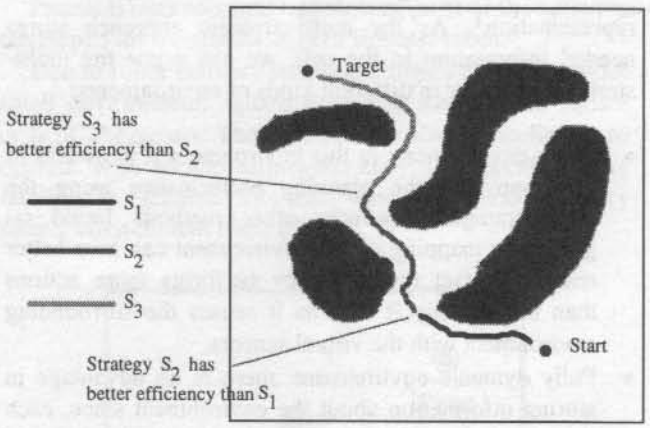


Figure 6. - Using the best strategy

4.- Issues on the multi-strategic proposed solution

Referring to above considerations, we have to compare the multi-strategic approach with other improvement methods and not with particular path planning strategies.

The main difference between the multi-strategic approach and the classic ones is that for each cell we define the *strategy* and not the *path* needed to reach the

target. This is the same difference we can find between Perceptive Learning (the classical approach) and Behavioral Learning (our approach). Behavioral learning allows the robot to consider qualitative rather than quantitative information about the surrounding environment; therefore obstacle movements and world modifications, generally speaking, affect the multi-strategic approach less than the classic ones.

The multi-strategic approach is especially advantageous in case of few environment modifications; to explain this, let us consider the unknown world the robot is moving in. There are two possible kinds of unknown environments: non changing (static) environment and changing (dynamic) environment.

To formalize these environmental classes we can define N as the number of cells into which an environment E is partitioned, and $C_{E_t} \subset C$ the world characteristics considered at time t. If t_1 and t_2 (with $t_2 > t_1$) represent the times at which the robot has to plan a path and M represents the number of cells changed between t_1 and t_2, we can define:

- Static environment: $M = 0$.
- Fully dynamic environment: $M \to N$.
- Partially dynamic environment: $M \to 0$.

This way, we have established a link between the environmental changes and the used environmental representation[1]. As the multi-strategic approach stores needed information in the cell, we can argue the multi-strategic behavior in different kinds of environments:

- Static environment: in this environment it is useless to try improving the planning performance using the multi-strategic approach; other methods based on geometric mapping of the environment can give better results: in fact multi-strategy performs more actions than the strategy it uses as it senses the surrounding environment with the virtual sensors.
- Fully dynamic environment: there is no advantage in storing information about the environment since, each time the robot traverses it, it is completely different from the previous time. For this class of environments the multi-strategic approach is a good method to face the uncertainty of the world, but learning capabilities of the system are completely useless.
- Partially dynamic environment: some parts of environment do not change, and previously acquired experience can be useful for further path-planning.

[1]For the sake of simplicity, we do not take into account changes that may occur *during* the navigation. It should, however, be kept in mind that several navigation algorithms are able to cope with moving obstacles.

5.- The simulator

In order to test the multi-strategic approach we wrote in C++ a path planning simulator. The software simulates a robot having 12 sonars, an odometric system, classical movement capabilities and a set of known path planning strategies. This approximates well the Robuter robot used in practical experiments.

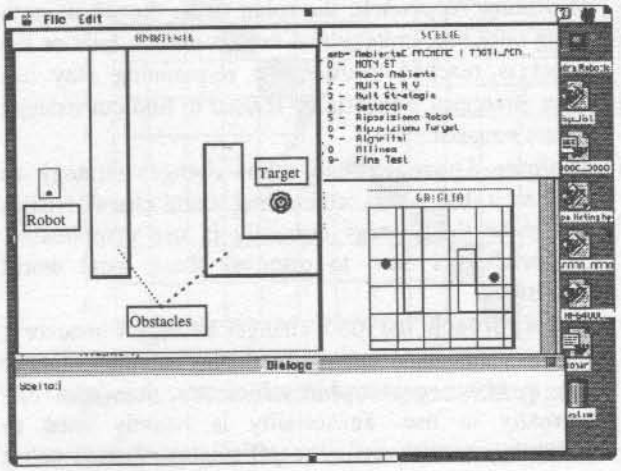
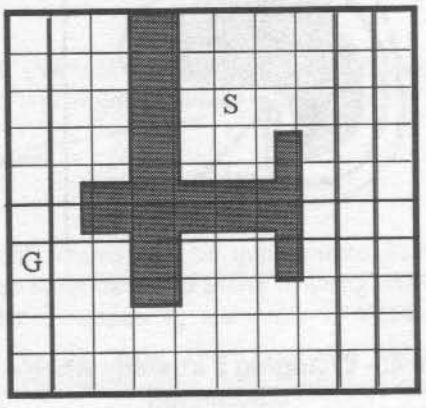


Figure 7. - The simulator

At this time, the implemented simulator runs with the "Minimum approach"; to test this approach we use simple strategies from the Bug2 family [Lumelsky, Stepanov 86] and more sophisticated ones GNav and LNav [Rao et al. 89]. The Bug2 strategies were modified to further simplify them; these modified versions may cycle.



Strategy	VSA	VSAE
Bug right	No	-
Bug left	Yes	4.163579e-04
Lnav	Yes	2.765488e-04
Gnav	Yes	6.089761e-04

Figure 8. - Test several applicable strategies

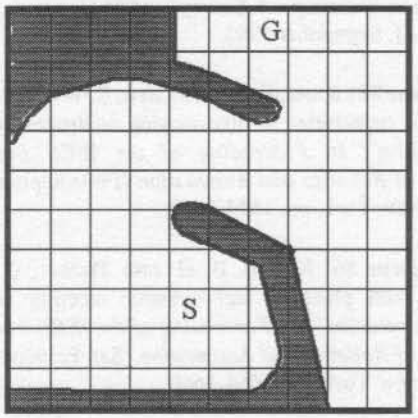
We have collected several data about the proposed path-planning method. The following figures and tables summarize the tests performed. The S and G point are respectively the Starting and the Goal points.

We can see that, assumed that the tests were performed with the minimum approach:

- for the same environment and same G and S points we generally have strategies that works better than others;
- for the same environment with changed S and G points in two different tests we could have different applicable strategies.

As we can see in figure 8 the best strategy for that environment and with those G and S points is the GNav.

In the test performed in the environment of figure 9, the LNav and GNav strategies are not applicable due to the non-polygonal obstacle shape.



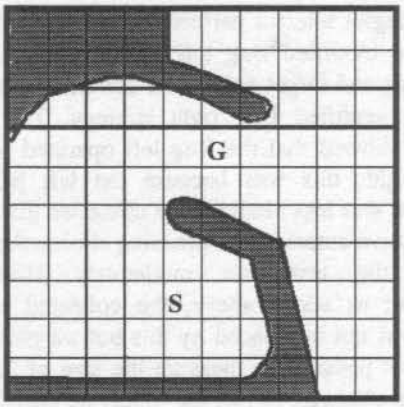
Strategy	VSA	VSAE
Bug right	Yes	4.343257e-04
Bug left	No	-
Lnav	No	-
Gnav	No	-

Figure 9. - Test different S and G locations: 1

If we change the S and G points the applicable strategy changes too (see G and S locations of figure 10)

After having performed several tests in a number of different, we can point out that:

- we have collected much evidences to confirm our initial assertion about the non-existence of an always optimal strategy.
- the way we use to collect experience is adequate for successive re-planning
- moderate changes of the environment do not affect significantly the quality of the collected experience.



Strategy	VSA	VSAE
Bug right	No	-
Bug left	Yes	4.697563e-04
Lnav	No	-
Gnav	No	-

Figure 10. - Test different S and G locations: 2

The last consideration is very important: the qualitative learning may cope with moderate changes of the environment. In our tests we have considered "moderate" the change of 15% of the environment cells.

6.- Tests on a real robot

Practical tests required transferring parts of the software developed for the simulator on a Robuter robot.

Due to robot memory size limitations we had to choose small environment: square of 10*10 meters with square cells of 2*2 meters. These limitations also compelled us to operate with simple strategies: only the modified Bug family was included. The tests were performed in a very simple environment (figure 11).

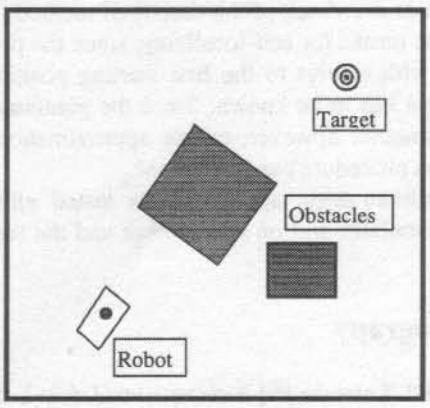


Figure 11. -The simple environment used to test the robot

The strategies selector performed well: at the beginning it chose the modified Bug left strategy. Then, with the same starting and target points, we compelled the selector to use the modified Bug right strategy. The collected experience showed that the Bug left operated better than the Bug right; this was because the left part of the environment was less cluttered by obstacles than the right one. Successive automatic re-planning choose the Bug left.

During the tests we moderately changed the environment; as stated above, the collected experience generally was not influenced by this but we pointed it out as it was not possible to measure the size of "moderate" environment changes: in fact the moderate changes of size strictly depend on the strategies used to plan the path.

7.- Conclusions

Our work has still to be completed following two main directions:

- for the "true" multi-strategic approach we have to study and to implement the Full Approach described above; only at that time the selector will always choose the best strategy by continuously sensing the environment;
- it is very difficult to implement the virtual sensors; in fact their input assumes also the strategy to be used; we probably have to formalize the strategies according to their operating hypotheses.

This time, we have shown a method for planning robot navigation which is capable of:

- selecting the apparently best suited algorithm among a library of navigation strategies;
- learning about failures or successes of previously used strategies.

The major drawback of the described methods is that it needs some means for self-localizing, since the position of the robot with respect to the first starting position in the environment has to be known. Since the planning method is not geometric, however, coarse approximations in the localization procedure can be allowed.

The multi-strategic approach was tested either on a software simulator and on a real robot and the tests prove the theory.

8.- Bibliograpy

[Barraquand, Latombe 90] BARRAQUAND, J. AND LATOMBE, J. C. 1990. "A Monte Carlo algorithm for Path Planning with many degrees of freedom". In *Proceeding of the IEEE International Conference of Robotics and Automation* (Cincinnati, May 13-18, 1990). IEEE New York, pp. 1712-1717.

[Brooks 83] BROOKS, R. A. 1983. "Solving the findpath problem by good representation of free space". *IEEE Trans. Syst., Man, and Cybernetics SMC-13*, 3 (Mar./Apr.), 190-197.

[Canny et al. 88] CANNY, J. F., DONALD, B., REIF, J., AND XAVIER, P. 1988. "On the complexity of kino-dynamic planning". In *Proceedings of the 29th Annual Symposium on Foundations of Computer Science* (White Plains, New York, Oct. 24-26). IEEE, New York, pp. 306-315.

[Erdmann, Lozano-Perez 86] ERDMANN, M. AND LOZANO-PEREZ, T. 1986. "On multiple moving objects". In *Proceeding of the IEEE International Conference of Robotics and Automation* (San Francisco, Apr. 7-10). IEEE, New York, pp. 1419-1424.

[Fortune, Wilfong 88] FORTUNE, S. AND WILFONG, G. 1988. "Planning constrained motion". In *The Symposium on the Theory of Computer Science* (Chicago). pp. 445-459.

[Hwang, Ahuja 92]. HWANG, Y. K. AND AHUJA, N. 1992. "Gross Motion Planning - A Survey". *ACM Computing Surveys*, Vol. 24. No. 3. September 1992.

[Kant, Zucker 88] KANT, K. AND ZUCKER, S. W. 1988. Planning collision free trajectories in time-varying environments: A two-level hierarchy". In *Proceeding of the IEEE International Conference of Robotics and Automation* (Philadelphia, Apr. 24-29). IEEE, New York, pp. 1644-1649.

[Krogh, Thorpe 86] KROGH, B. H. AND THORPE, C. E. 1986. "Integrated path planning and dynamic steering control for autonomous vehicles ". In *Proceeding of the IEEE International Conference of Robotics and Automation* (San Francisco, Apr. 7-10). IEEE, New York, pp. 1664-1669.

[Lozano-Perez 87] LOZANO-PÈREZ, T. 1987. "A simple motion planning algorithm for general robot manipulation". *IEEE J. Robotics Auto.* RA-3, 3 (June), pp. 224-238.

[Lozano-Perez et al. 89] LOZANO-PÈREZ, T., JONES, J. L., MAZER, E., O'DONNEL, P. A. 1989. "Task level planning of pick and place robot motions". *IEEE Comput.22*, 3 (Mar.), pp. 21-29.

[Lozano-Perez, Wesley 79] LOZANO-PÈREZ, T. AND WESLEY, M. A. 1979 "An algorithm for planning collision-free paths among polyhedral obstacles". *Commun. ACM. 22*, 10 (Oct.), pp. 560-570.

[Lumelsky 89] LUMELSKY, V. 1989 "On the connection between maze-searching and robot planning algorithms", *Proc. of the 28th IEEE Conf. on Decision and Control*, Vol. 2 December 1989.

[Lumelsky, Stepanov 86] LUMELSKY, V. AND STEPANOV, A. A. 1986 "Dynamic path planning for a mobile automaton with limited information on the environment", *IEEE Trans. on Automatic Control*, vol. AC-31, No. 11, 1986.

[Rao et al. 88] RAO, N., IYVENGAR, S., E DESAUSSURE, G., 1988.

"The visit problem: visibility graph-based solution". *In Proceedings of the IEEE International Conference on Robotics and Automation* (Philadelphia, Apr. 24-29). IEEE, New York, pp. 1650-1655, 1988.

[Reif 79] REIF, J. H. 1979. "Complexity of the mover's problem and generalizations, extended abstract". In *Proceeding of the IEEE Symposium on Foundation of Computer Science* (San Juan, Puerto Rico, Oct. 29-31). IEEE, New York, pp. 421-427.

[Reif, Sharir 85] REIF, J. H. AND SHARIR, M. 1985. "Motion planning in the presence of moving obstacles". In *Proceeding of the 26th Annual IEEE Symposium on Foundation of Computer Science* (Portland, Oreg., Oct. 21-23). IEEE, New York, pp. 144-154.

[Schwartz, Sharir 83] SCHWARTZ, J. H. AND SHARIR, M. 1983. "On the piano movers' problem: II. Coordinating the motion of several independent bodies". In *Int. J. Robotics Res.* 2, 3 (Fall), pp. 41-46.