Proceedings of the 3rd.Portuguese Conference on Automatic Control
Coimbra, Portugal, September 1998

NAVIGATION ARCHITECTURE FOR THE RESOLV MOBILE ROBOT*

Vitor Santos[1], **José Castro**[2], **M. Isabel Ribeiro**[2]

[1]*Universidade de Aveiro, Departamento de Engenharia Mecânica,*
3810 Aveiro, PORTUGAL
e-mail: vitor@ua.pt
[2]*Instituto Superior Técnico/Instituto de Sistemas e Robótica*
Av.Rovisco Pais 1, P 1096 Lisboa CODEX, PORTUGAL
e-mail: {jcastro,mir}@isr.ist.utl.pt

Abstract: The RESOLV project requires a mobile robot to robustly move in partially or totally unknown environments. The robot needs to be autonomous and accomplish remote high level orders sent by a user or a computer application. This paper describes a multi-loop, modular navigation architecture adapted for this robot but usable on other platforms. A clear separation between local and global navigation is proposed. Both can run independently, but with an adequate alternation/competition between them trajectory execution is achieved successfully. That includes avoidance of unknown or, to a given extent, dynamic obstacles.

Keywords: Navigation Architecture, Mobile Robots, Perception Maps, Clothoids.

1. INTRODUCTION

The navigation architecture described in this paper is implemented on the AEST (Autonomous Environment Sensor for Telepresence) mobile platform displayed in Figure 1 built in the framework of the RESOLV project. The project aims at the 3D reconstruction of large and complex indoor environments based on range and video data. The transportation of the sensor head (Laser Range Scanner plus Vision Camera) between the successive acquisition positions is done according to a perception plan that defines the next goal to be reached by the AEST, (Sequeira *et al*.,1996). Within a universe where the environment is unknown until it is reconstructed and where autonomous motion is required both in reconstructed and yet unmodelled regions, a robust navigation architecture for the AEST is imperative.

In the context of RESOLV, a final goal has to be reached from the actual location. That requires the planning and execution of a designed trajectory, the avoidance of eventual unknown or unexpected static or dynamic obstacles, the emergency detection and handling, the localisation and the recovery or, should it be necessary, the recalculation of the path to reach the final goal.

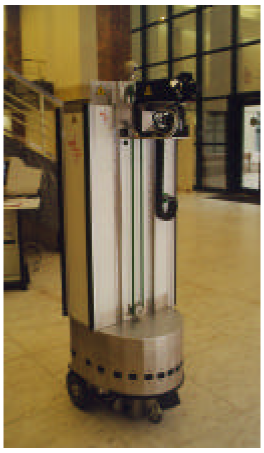


Figure 1 - AEST mobile platform

The architecture, composed by three enclosed loops implementing reflexive, reactive and functional procedures, is robust and modular achieving correct motion in dynamic environments. The major novelty is the competition of different navigation modes (sets of motion strategies) according to the robot status and task under execution.

* This work was supported by the project RESOLV-Reconstruction using Scanned Laser and Video of the ACTS Programme, EU.

2. NAVIGATION ARCHITECTURE

Most navigation architectures can be divided into two main categories: the **behavioural**, such as the subsumption proposed by (Brooks, 1986) and the **functional**. Behavioural architectures include the **reflexive** and the **reactive** types, depending on the degree of sensing data processing prior to action taking. A third category, the **hybrid** architecture, tries to fuse the pros of both the behavioural and functional categories ending up with larger capabilities.

A navigation architecture for a robot as complex as the one involved in the AEST's operation cannot be purely reflexive because that would certainly lead to instabilities due to the large number of sensors (24 ultrasonic for navigation purposes) and to the poor reliability of individual measurements. Therefore, a reactive structure, though appearing somehow limited, was a suited point to start defining an alternative. In the other hand, as some navigation actions were to be more elaborate than simple reactive behaviours, the functional component had certainly to be inserted in the architecture. Stated this, it is clear that the final architecture will be of the hybrid type.

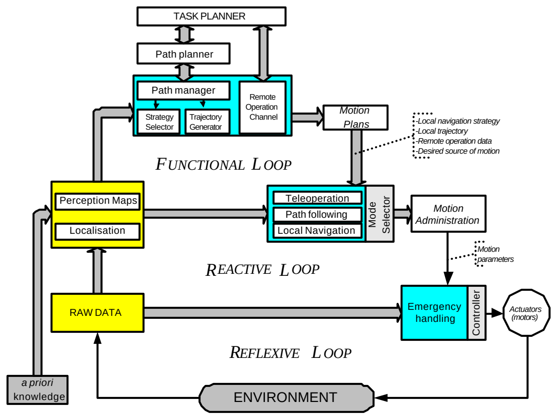


Figure 2 - AEST Navigation Architecture

The proposed architecture, which evolved from earlier work, (Santos *et al.*, 1994), (Santos *et al.*, 1996) is represented in Figure 2. It has as a set of three enclosing loops: the **reflexive loop,** the **reactive loop** and the **functional loop**.

The **reflexive loop** deals with imminent collision detection, loss of communications with a remote host, the Host PC, freshness of sensorial data, and very simple motion commands to evade traps or dead locks. Raw sensorial data is available within this loop.

The **reactive loop** deals with local motion, path following and localisation issues. Actions to be taken after data acquisition are more elaborate than simply reflexive and sensorial data can be more than simply raw. Processed, integrated or fused data is obtained at this level. Also at this level, eventual external *a priori* knowledge (processed data, e.g., a full or partial model of the environment) is supplied to the system for localisation purposes.

The main issue is to perform safe motion in the environment whatever it may be. Three navigation modes, **teleoperation**, **path following** and **local navigation** are implemented with motion achieved by alternation/competition among them. Each of these modes is permanently available and assured by specific modules. The modules attached to each navigation mode (essentially path follower and local navigation) continuously calculate what would be, from their point of view and with their inputs, the correct motion. Another module, the mode selector, acts as a referee by defining which is to be the "winning" motion according to some rules. Teleoperation, by definition, always wins unless path appears obstructed, or other safety concerns are to be taken into account. In the path following mode, the AEST performs motion along specified curves to sequentially cover a list of sub-goals. Local navigation wins when the path being executed risks the short-term collision with obstacles. Normally, the local navigation module actuates when the projected path is non-feasible due to obstacles. In this working mode the obstacle is avoided (contoured or deviated from, depending on the current local navigation strategy) and as soon as the next subgoal direction is free, the system returns to path following mode. Figure 3 illustrates the basic navigation modes and the main transitions among them.

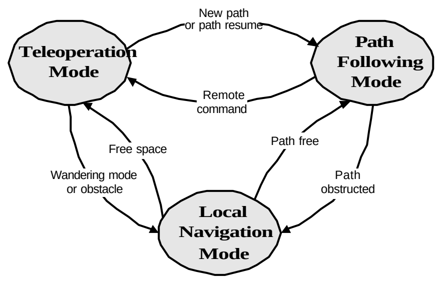


Figure 3 - Transitions of navigation modes

The **functional loop** is responsible for evaluating the path to next subgoal and which strategy is suited for local navigation, should it become active. It still verifies whether remote operator is demanding direct control of the robot. Whenever the path following mode is the desired one, the competition among modes occurring at the reactive loop may raise a path-recovery problem, i.e., the definition of a new path to reach the final goal. This problem is not directly solved by the competition among navigation modes given that the reactive nature of the modules involved determines conflicting behaviours competing for the control of the system and path recovering requires co-operation. It is only at the functional level that the problem is solved, by computing in real time appropriate motion plans to influence (not decide) the behaviour of the reactive loop. Among the functions assured by this functional loop there is still the verification whether subgoals have been reached, and the manipulation and rearranging of their sequence in order to fulfil or try to optimise the navigation task.

3. IMPLEMENTATION OF THE ARCHITECTURE

The navigation architecture is implemented within the AEST overall software architecture (Lopes *et al.*, 1998) distributed among three different processors as represented in Figure 4. Localisation, Path Planning and Mobile Robot Communication Interface (MRCI) modules run on the Host PC, while the remaining of the navigation algorithms are implemented on the Motorola 68040 processor board of the mobile platform with the real-time operating system Albatros. Teleoperation is provided through the Human-Computer Interface on a Notebook.

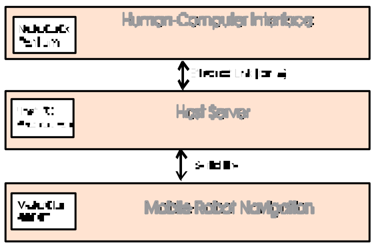


Figure 4 - Host Server Software Architecture

The navigation components running on the 68040 are organised in several modules event if almost all have one or more real time processes associated, as represented in Figure 5.

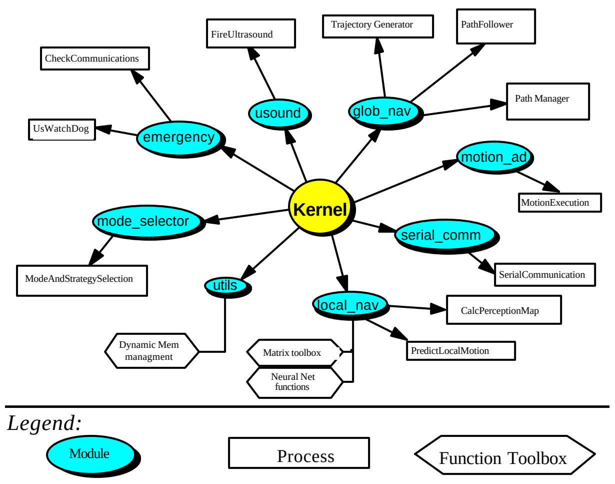


Figure 5 - Software architecture on the Robot

The system receives messages from the Host PC through the MRCI module. That is managed by the `SerialCommunication` process from the `serial_comm` module. This process has the ability of extracting full messages from incoming buffer, analyse their category and dispatch them to the several modules, by invoking appropriate specific functions provided by those modules.

4. MAIN BLOCKS OF THE ARCHITECTURE

A brief description of the principles underlying some of blocks in Figure 2 is presented next. An absolute localisation procedure, detailed described in (Gomes-Mota and Ribeiro, 1998), periodically sets odometry. Therefore, odometric readings are the best available estimate of the platform's location.

4.1. Emergency handling

This is a key block on the reflexive loop of the architecture, managing the lowest level of procedures. Its actions concern essentially the detection of **imminent collision** and platform stopping in case the velocity (intensity and direction) is considered unsafe for the free space perceived by one or more sensors. The sensorial data, provided by ultrasound sensors, is not processed at all, thus revealing the reflexive component. Another emergency is the **loss of communications** with the Host PC. The detection of no communications within a given period triggers a predefined action, such as "stop immediately".

4.2. Local Navigation

The local navigation is a block on the reactive loop, and also the name of a navigation mode. Local navigation provides motion to allow the robot to move with no given references. It exclusively uses the perception of the environment plus some motion behaviour: the **navigation strategy**. The use of raw data from 24 ultrasonic sensors to generate local motion is not suited due to the poor stability caused by erroneous and unstable sensorial measurements. Therefore, data must be processed in order to obtain a more robust and solid representation of space occupancy around the robot. This is achieved by the perception maps, (Santos *et al.*, 1994), (Santos *et al.*, 1996), that results from data integration and are organised as a radial robot-centred grid as displayed in Figure 6-a).

Among the developed methods to build the map, the most robust found was one based on Neural Networks. With these maps, (an example of occupancy is displayed in Figure 6-a) and obeying some simple strategy, such as "follow the free space", "follow the environment on the left/right", or "contour obstacles", motion is then generated using a dedicated algorithm (Santos, 1995), (Santos *et al.*, 1996).

4.3. Global Navigation

Given the robot actual location, the already reconstructed 3D model of the environment and the location of the next laser data acquisition (the desired goal), the **path planner** evaluates a set of subgoals. Each subgoal, $\mathbf{g}_i$, consists of a vector $\mathbf{p}_i = (x_i\ , y_i\ , \theta_i\ )^T$ defining its posture in the global frame, and a flag with three possible status: *stopped* (goal to reach at zero velocity), *forward* or *reverse* (goal to reach with the correct heading). A simple optimisation algorithm defines a set of line segments to the goal, keeping a safe distance from already modelled environment and avoiding, whenever possible, motion on unmodelled regions. Subgoals are taken from the intersection points of consecutive segments. In the framework of the RESOLV project no sophisticated optimisation is required at this level since partially modelled environments are to be dealt with most of the times. Instead, subgoals and trajectories may be dynamically recalculated at different modules and during task execution, as explained next.

4.4. Navigation in the global frame

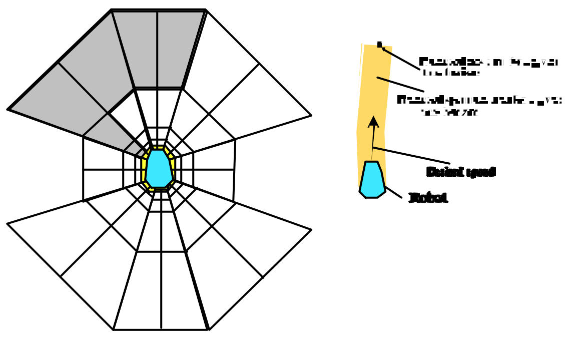
The navigation in the world frame is achieved by sequentially tracking the list of subgoals generated by the path planner, ended with the desired final goal. This list constraints the path to be executed, but leaves freedom on the navigation between them. The **trajectory generator** module computes, on-line, a smooth trajectory between the robot's current location and the next subgoal. Clothoid curves, (Kanayama and Miyake, 1986) are used aiming at smoothing motion to reduce the odometry errors produced by wheel slippage. The referred smooth trajectory is defined by a sequence of vectors $\mathbf{p}_k$ and an associated curvature, c_k, and velocity, v_k. Whenever the robot deviates more than a given amount from this trajectory a new one is computed from the actual location to the current subgoal. The **path follower** evaluates, in real time, the set $(\mathbf{p}_{k-1}, \mathbf{p}_k)$ with the smaller distance to the robot's actual location based on an ellipse inclusion criterion. Then, it dispatches $\mathbf{ref}_k = (\mathbf{p}_k^T\ c_k\ v_k)^T$ as the reference for a hybrid controller that implements a trajectory following control law.

The **path manager** module checks odometry regularly and when the current subgoal is reached, takes the next subgoal as the current one to attract motion to. A path is considered as having been executed when the final goal is reached within a pre-specified tolerance. Whenever an unexpected obstacle, which is checked out through the combined information of the perception maps and odometry blocks the current sub-goal, the **path manager** replaces the current subgoal and may, if necessary, insert additional subgoals.

4.5. Navigation mode selection

Through the Human-Computer Interface (Figure 4) the operator chooses the desired navigation mode which, however, might not be always active.

In fact, the **mode selector** performs the selection of the active mode. Considering the robot dimensions and the motion proposed by the desired navigation mode, the **mode selector** estimates, within certain tolerances and within a given **time horizon**, the spanned area needed to perform the desired motion (Figure 6-b). The occupancy of this area, currently enlarged on the vehicle sideways, is checked by analysing the perception maps (Figure 6-a). If it is occupied, local navigation is selected. Otherwise, the navigation mode will be the desired one.



(a) (b)
Figure 6 -a) Perception map - b) Spanned area

5. RESULTS

This section presents experimental results on the combined action of global navigation, local navigation and teleoperation. The trajectories correspond to the path described by the mid point between the drive wheels of the AEST, which has a width of about 50 cm. The sub-goals and the final goal are represented by its position, with the arrow displaying the orientation with which the goal/sub-goal should be reached.

5.1. Global Navigation and Teleoperation

Figure 7 illustrates situations of path recovery performed with the AEST. A sequence of subgoals, sg_i and a final goal are given as a path. Superimposed are the results of two experiments: one, represented by a dashed line, corresponds to the execution of a path exclusively in the path following mode. The AEST moves from the initial location, reaches each sub-goal within a given tolerance and stops at the final goal. On the second experiment, the robot starts at the same initial location and, during two distinct time intervals, an operator teleoperates the AEST. The solid grey line represents the path followed during teleoperation. When teleoperation is no longer active, a new path is evaluated to regain the original one. Yet, the path manager skips one intermediate subgoal.

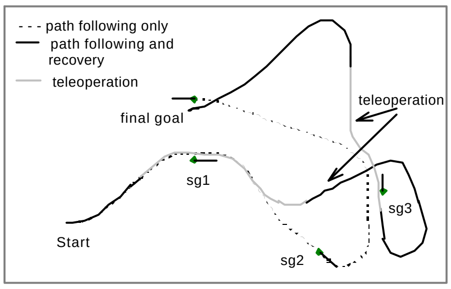


Figure 7 - Global Navigation and Teleoperation

5.2. Global and Local Navigation

Extensive results on local navigation as a standalone module (wondering mode), can be found in (Santos *et al.*, 1994), (Santos *et al.*, 1996). The new results obtained in the framework of RESOLV concern the interaction of the path follower and mode selector when global and local navigation modes are active. Three different experiments were carried out in the same environment with and without unexpected obstacles. The AEST location is referred to a world direct frame whose origin is the point (0,0) and where orientation is measured counter clockwise relative to the positive x axis. In all the experiments the initial location is (0,0,180º). In the figures, the thin arcs correspond to a path under the global navigation, while the **light grey** ones correspond to the activation of the local navigation.

- **Experiment 1**

The mission has only the final goal at (-2,1.5,180º) and the obstacle in the middle is not know a priori. Figures 8, 9 and 10 display the path described for different time horizon of the mode selector, 3, 4 and 5 seconds. The unexpected obstacle is avoided in the three cases and in all of them the final goal is reached within a given tolerance, but the trajectories change as a function of the mode selector time horizon. As described before, the mode selector checks the perception maps within a specified time horizon taking into account the desired motion defined according to the vehicle's velocity. With a short time horizon only a small environment region surrounding the robot is analysed and therefore the AEST only perceives the environment obstacles when it is close to them. On the contrary, with a large time horizon the analysed perception maps extends far apart from the AEST and therefore environment obstacles are sensed even if they are not close to the mobile robot. As a conclusion, an increase on the time horizon renders the AEST more sensitive to the environment obstacles and borders, and therefore the local navigation mode becomes active more often and remains active during longer periods.

These effects are visible on Figure 8, Figure 9 and Figure 10. In Figure 8, which corresponds to the smallest time horizon, the central obstacle is only dealt with when the AEST is close to it. Once avoided, by the activation of the local navigation mode, the remaining path corresponds to a global navigation with no interference from the environment borders. In this case the local navigation becomes active only when it is strictly necessary. With a time horizon of 4 and 5 seconds, the environment borders are sensed and taken into account by the mode selector. The local navigation becomes active more often and remains active during longer periods. For the time horizon of 4 seconds this is the reason why the mobile robot's behaviour is quite oscillatory with the AEST being alternatively repelled by the borders. For a time horizon of 5 seconds, even though this oscillatory behaviour is still evident, the trajectory is smoother because the AEST remains longer periods in the local navigation mode and therefore the transitions between local and global navigation are less frequent.

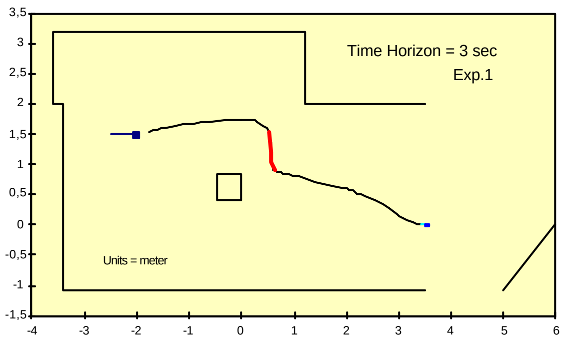


Figure 8 - Exp1: Time horizon = 3 sec

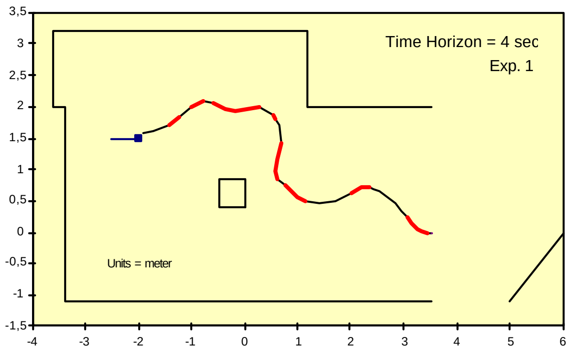


Figure 9 - Exp1: Time horizon = 4 sec

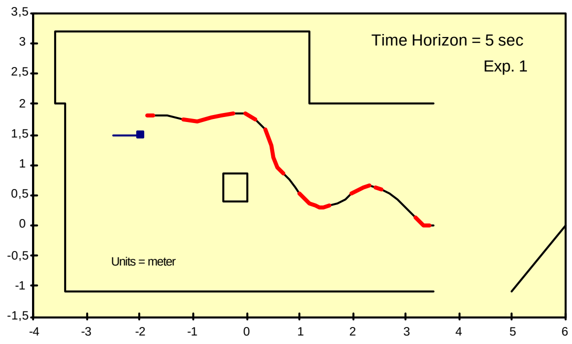


Figure 10 - Exp1: Time horizon = 5 sec

- **Experiment 2**

The location of the central obstacle is assumed known a priori. To reach the final goal at (-2,1.5,180º) a subgoal is defined with coordinates (0,1.5,180º). Figure 11, obtained for a time horizon of 3 seconds, displays the correspondent path. With the intermediate sub-goal, the paths defined at the global navigation level drive the AEST from its initial location towards the sub-goal and from this to the final goal, thus implicitly avoiding the central obstacle without requiring the local navigation activation. This mode is only activated for a short period near the sub-goal due to the central obstacle.

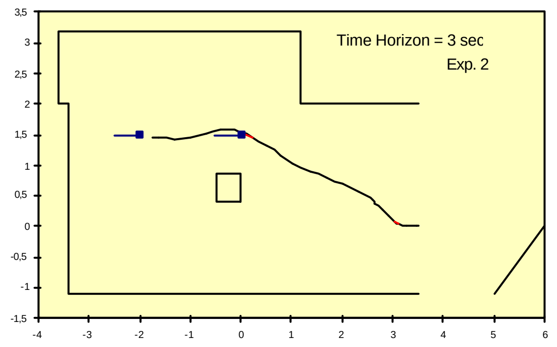


Figure 11 - Exp2: Time horizon = 3 sec

- **Experiment 3**

The position of the central obstacle assumed known a priori. Experiment 3 corresponds to a complete tour around the central obstacle: from the initial location at (3.5,0,180º), the AEST has to successively reach the sub-goals (0,2,180º), (-2,1.5,180º), (-1,0,0º) towards the final goal at (3.5,0,0º). The experiment was carried out for mode selector time horizons of 3 and 5 seconds.

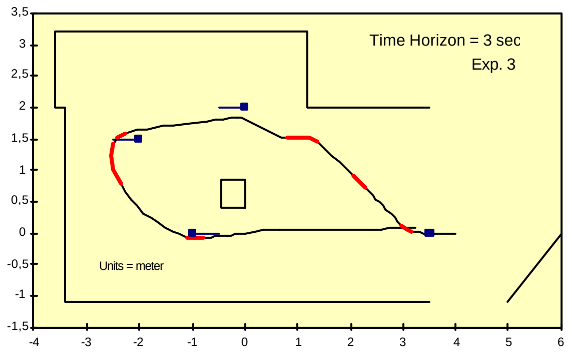


Figure 12 - Exp3: Time horizon = 3 sec

With a time horizon of 3 seconds (Figure 12), the path closely follows the sub-goals and the AEST is almost insensitive to the environment's borders, this resulting in a smooth trajectory where the local navigation becomes active only in short periods. On the contrary, with a time horizon of 5 seconds (Figure 13), the border is almost always perceived this resulting in a real competition and alternate activation of local and global navigation. This is particularly true on the last part of the mission, between the last sub-goal and the final goal, where motion is to be done along a straight wall. The displayed results were obtained using the "Follow Free Space" as the local navigator strategy. Should the "follow obstacle on the right" strategy been applied and the trajectory along the last part of the trajectory will be smoother, even for a time horizon of 5 seconds.

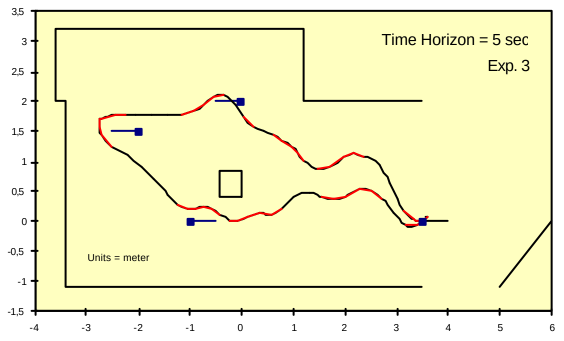


Figure 13 - Exp3: Time horizon = 5 sec

6. CONCLUSIONS AND FUTURE WORK

The developed multi-loop architecture has proved its robustness and the ability to fulfil the requirements of the project, that is, go from one point to another in space, coping with most types of unknown obstacles. However, its possibilities and usage go far beyond that: it is the case of dynamic obstacles, which are dealt using the same principles. Motion behaviours are built on the fly and therefore true autonomy is achieved. Extremely demanding experiments are to be carried on using moving and/or U-shaped obstacles, etc. This will show the possibilities or limitations on demanding circumstances as well as give the opportunity to tune some parameters of navigation, or to give rules to robot of how to do it by itself.

REFERENCES

Brooks, R.A. (1986). A Robust Layered Control System for a Mobile Robot. *IEEE Journal of Robotics and Automation*, **vol. 2, n. 1**, pp.14–23.

Gomes-Mota,J., M.I.Ribeiro (1998). Localisation of a Mobile Robot using a Laser Scanner on Reconstructed 2D Models. *In Proc. 3rd Portuguese Conference on Automatic Control*, Coimbra, Portugal.

Kanayama, Y., N. Miyake (1986). Trajectory Generation for Mobile Robots. *In Robotics Research, The MIT Press*, **vol. 3**, pp.333–340.

Lopes, F., P.Gil and J. Pereira (1998). Client/Server Software Architecture for a Robotic System. *In Proc. 3rd .Portuguese Conference on Automatic Control*, Coimbra, Portugal.

Santos, V., J. Gonçalves and F. Vaz (1994). Perception Maps for the Local Navigation of a Mobile Robot: a Neural Network Approach. *In Proc. IEEE Int. Conf. on R&A*, pp.2193-2198, San Diego, USA.

Santos,V.(1995). Autonomous Robot Navigation: Sensorial Data Interpretation and Local Navigation, *PhD Thesis*, University of Aveiro, Portugal (in Portuguese).

Santos, V., J. Gonçalves and F. Vaz (1996). Local Perception Maps for Autonomous Robot Navigation. *In Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp.821–827, Osaka, Japan.

Sequeira, V., J. G. M. Gonçalves and M.I.Ribeiro (1996). Active View Selection for Efficient 3D Scene Reconstruction. *In Proc. of the 13th ICPR*, pp.Vienna, Austria.