A distributed architecture for autonomous navigation of robots

Vito Di Gesù, Biagio Lenzitti, Giosuè Lo Bosco, Domenico Tegolo
University of Palermo
Via Archirafi 34, 90123 Palermo, ITALY
(digesu,lenzitti,lobosco,tegolo@math.unipa.it)

Abstract

The paper shows a distributed architecture for autonomous robot navigation. The architecture is based on three modules that are implemented on separate and interacting agents: the target recognizer, the obsta90cle evaluator and the planner. An adaptive genetic algorithm has been studied to identify mechanisms for reaching the target and for manipulating the 2-diretions of the robot; the distributed architecture has been embedded in the DAISY (Distributed Architecture for Intelligent System). Experiments have been carried out using a LEGO intelligent brick.

1. Introduction

Autonomous Robot Navigation (ARN) is an interesting field of investigation because of its relevance in several risky applications such as environment surveillance, sea-bottom exploration, and atomic power plant. ARN systems are endowed with different kind of sensors, from which information flows to suitable modules to perform elaboration useful for decisions making.
The performance of an ARN system lies on the strategy of integration of information from several sensors. Optimized algorithms allow faster and correct actions such as obstacle avoiding and target achievement. Moreover, ARN systems should be able to adapt their behavior depending on the current goal and the nature of the input data. Such performance can be obtained in systems able to interact dynamically with the environment.

Previous considerations suggest the use of information-fusion techniques [1], implemented on distributed systems by using goals-oriented strategies and optimization techniques [2,3,4].

In this paper we propose a distributed architecture of an ARN systems for the autonomous navigation of a robot in an unknown environment towards a given target. Obstacle can be present between the robot and target. Data are collected from a video camera, placed on the head of the robot, and from two infrared sensors on the bottom. The video information is used to recognize the target; the infrared information allows the evaluation of distance information from the obstacles.

Our work has been also motivated by the increasing interest in studying learning and problem solving in artificial living systems and compare them with animal behavior (prey - predator method). The genetic algorithm (GA) proposed for the planner is related to that designed by Yamada [5]. In this paper the authors generate a description of the environment by using robot paths. A review on application of GA and neural network for robot navigation and goal-oriented problems can be found in [6].

Experiments have been carried out in simulation in order to evaluate the best assessment and system configuration on real laboratory environment. For this purpose a LEGO™-robot has been used, equipped with sensors (video camera and infrared), on board computer with radio Internet.
The paper is organized as follows. Section 2 describes the distributed architecture. The planning agent is described in Section 3. Section 4 describes experimental result and implementations details. Conclusions are made in Section 5.

2. The architecture

The distributed architecture has been implemented inside the DAISY system [7]. According to DAISY architecture the ARN system is based on agents too. Two agents are dedicated to the evaluation of the sensors information (*target recognizer* and *the obstacle evaluator*) a third agent (*the planner*) plan the robot movements according to an optimization strategy [3] (see figure 1). Two more modules are used to handle the I/O and to evaluate the robot movements. They are incorporated in the corresponding agents.

The choice of DAISY has been motivated because it allows reconfigurability and provides coupling functions

™ LEGO Group, DK-7190 Billund, Denmark

0-7695-0740-9/00 $10.00 © 2000 IEEE

between co-operating agent mechanism and the mobile units (the LEGO-robot).

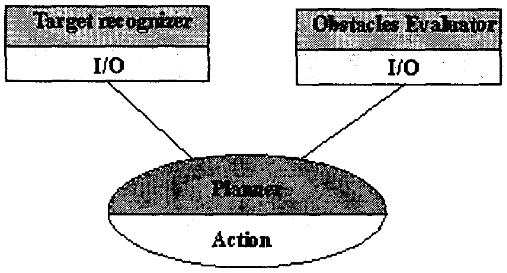


Figure 1. The agents of the ARN system.

This approach allows us to implement a co-operating agents strategy in our system. The system evolution is based on the *Active Information Fusion* (AIF) loop [8].

The AIF loop consists of five modules (see figure 2): *Observe, Evaluation, Optimization, Choose-Next,* and *Action. Observe* is responsible for the acquisition and preprocessing of sensorial data; *Evaluation* compute, on the basis of the observed parameters and the current state of the robot the next choose; the *evaluation* uses an optimization's method (in our case a genetic algorithm).

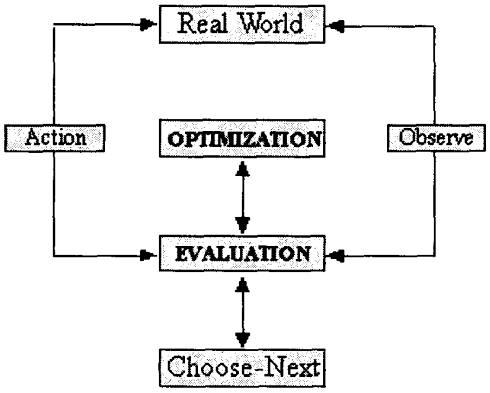


Figure 2. The Information Active Fusion Loop.

Choose-Next selects the new movement of the robot on the basis of the evaluation module. It should be noted that all processes may run in parallel mode and they may be executed on the network of the distributed. The result of the choose module determines the *Action* (next movement in the environment). After the movement the *Observe* module drives further sensor-explorations. The *Real World* represents the environment on which the ARN operates. The information, within the system, flows in a continuous active fusion loop.

3. The planning agent

GA's have been applied to behavioral studies, simulation and optimization. They have been applied for the control of instrumentation in risky environments (atomic energy cite, sea bed, space missions). GA's are useful whenever the complexity of the problem requires for approximated fast solutions.

The planning agent drives the robot toward the target. The problem can be stated, informally, as follows: *find the minimum number of steps to reach the target in position* (x_T, y_T) *starting from the position* (x_0, y_0) *and avoiding obstacles in the room.*

The navigation problem, before sketched, can be formulated, in the case of one rectangular obstacle of size *a*x*b*, as a dynamic programming problem (see figure 3):

$$x_i = S_x(\theta_{i-1}, x_{i-1}, y_{i-1}, x_{ob}, y_{ob}, x_F, y_T; a, b)$$
$$y_i = S_y(\theta_{i-1}, x_{i-1}, y_{i-1}, x_{ob}, y_{ob}, x_F, y_T; a, b)$$
$$\theta_i = S_\theta(\theta_{i-1}, x_{i-1}, y_{i-1}, x_{ob}, y_{ob}, x_F, y_T; a, b)$$
$$\min(\delta(x_0, y_0, x_T, y_T))$$

The variables in the equations represent the coordinates of the robots (x_i, y_i), the coordinates of the top-left corner of the obstacle (x_{ob}, y_{ob}), the coordinates of the target (x_T, y_T), the angle (θ) between the direction of the robot displacement and the vector robot target.

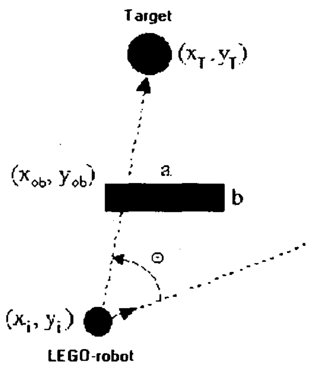


Figure 3. The robot target assessment.

Functions S_x, S_y, S_θ, describe the dynamical evolution of the system (for sake of simplicity it includes the bounds conditions of the obstacle), the function δ measure the length of the path made by the robot to reach the target in step units. The problem has exponential complexity, even in the case of a single obstacle with rectangular shape.

In real situation, the robot has not knowledge of the coordinate system and only visual sensors and distance evaluators drive its movements. In our case it can measure the distance from the obstacles and the direction of the target respect its displacement direction.

The GA, here described, has been designed to approximate the solution of the general optimal problem above stated. The planner, on the basis of the information given by the sensors agents, provides the direction of the displacement and the speed to the robot. The GA can be sketched as follows:

1) select a starting population of genomes (random choice of 200 samples directions for example "1010101111010111");
2) perform the crossover operation on randomly chosen genomes (the crossover is performed in a single point and the probability of crossover is uniform), for example: let us be two genomes "10010110 01011101" and "11110110 01000001", the new genomes are "10010110 01000001" "11110110 0101110";
3) perform a bit-mutation with probability 0.01 to change one or more bit randomly
4) compute the fitting function, $F_{h,k}$, for each pair of chromosomes (k,h);
5) select the new population (with the tournament technique);
6) **if** the global fitting, $\overline{F}$, satisfy the convergence criteria **goto** 7) **else goto** 2);
7) end

Data representation. One of the key points in GA is the representation of the data that constitute the *genome* on which to perform *genetic* transformations (*selection, crossover and mutation*). Our genomes are the displacement directions, α, in the interval [-π,π] (see figure 4). They are mapped in 16-bits word using the function: $R(\alpha) = (\alpha + \pi)(2^{15}-1)/2\pi$.

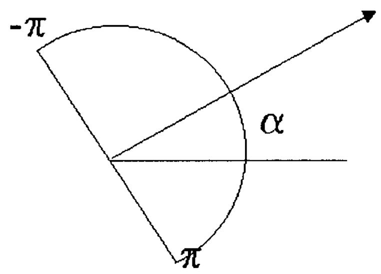


Figure 4. The direction of the displacement.

Fitness function. The mathematical definition of the fitness function has been formulated by considering obstacles and target as electric objects that generate an electrostatic field that is attractive in the case of the target and repulsive in the case of the obstacles:

$F_{itness} = \alpha \times F_{target} - \beta \times F_{obstacles}$ with $\alpha + \beta = 1$

Where $F_{target} = K \times (\pi - |\theta|)$ is a linear decreasing function of the angular distance, θ, of the robot from the target, K is a positive constant experimentally determined. $F_{obstacles}$ is function of the measures coming from the range detectors R_i (i=1,2):

$$F_{obstacles} = \sum_{i+1}^{2} \frac{C_i}{R_i^2}$$

C_i's are constants that depend on the infrared sensors direction. They have been computed during the calibration phase, described in the next section.

4. Implementation notes

The DAISY architecture has been designed to face perceptual problems; in this paper we keep in mind the control architecture for an autonomous robot. The distributed architecture is based on dynamic node that allows the flow of large number of data through Ethernet-radio. In our case, these information are used to correct the direction of the robot during its navigation and to detect the environment around it [8]. Figure 5 shows the structure of the dynamic node; its components are:

- *Video Camera to acquire digital images.* Predefined tasks, to their acquisition, send images to specialized tasks for navigating or for detecting.
- *Wheels and their controller.* This component allows to move the dynamic nodes along the direction fixed by the task on the distributed system, controller of wheel's speed is linked with portable personal computer and with electric-motor of the wheels.
- *Channels to communicate inside and outside.* A radio Ethernet is used for a better communication with the distributed system; a RJ45 Ethernet card is sufficient for this aim.
- *Remote sensing.* It includes all sensors that improve the performance of the dynamic unit (light sensor, echo, sonar, touch sensor).

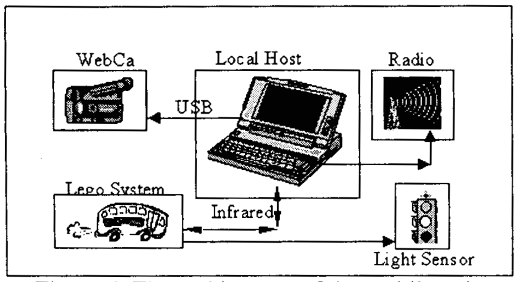


Figure 5. The architecture of the mobile unit.

The robot. The robot has been realized with a Lego brick and three electric motors that guarantee the movement of the mobile unit. A video camera (WebCam) on the head of the robot provides the acquisition of the visual information, two infrared sensors (Infrared) allow the detection of obstacles at short distance (see figure 6).

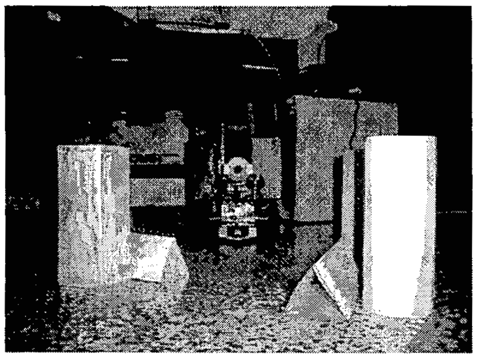


Figure 6. The environment whit mobile unit.

The local host. The dynamic node incorporates a portable personal computer (PPC) allowing temporary power independence. It supports standard operating system (Linux, Win98), standard programming language (Java, C++), standard network communication (internet), and standard communication library.

A Java server has been hosted on the PPC for allowing the communication between the local host and the remote computer. User, operating from an Internet connection, can handle the mobile unit from a remote computer.

The pilot system. Two different modalities have been considered to pilot the mobile unit. *The standard piloting system.* In this modality commands are sent using the Internet browser (see figure 7). The visual interface is organized in two sections, one to display image coming from video camera of the mobile unit, and the other one oriented to the manipulation of path of the unit and to guide the position of video camera. Two tasks have been defined to synchronize remote Internet browser, local host and intelligent brick. One of these receives data instructions from browser and send them to second one hosted on local host, the second one capture the data instructions from the network and, after their manipulation, send the new code to the intelligent Lego brick. This interface allows the shipping, with the MU, into the flat floor with or without obstacle.

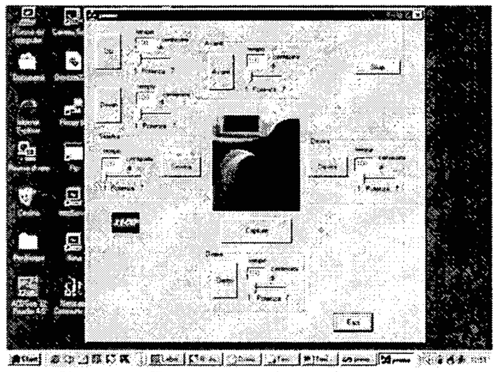


Figure 7. The standard piloting system.

The autonomous piloting system. In this modality, see figure 8, the driving of the mobile unit is established on the basis of the sensors information (video camera and Infrared) that are sent to the genetic planner.

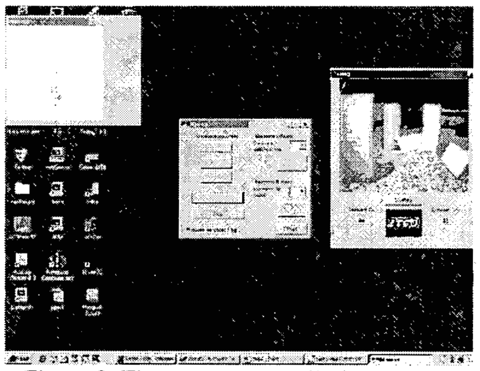


Figure 8. The autonomous piloting system.

The Robot software is composed by two programs, one is dedicated to the robot movements, it is written in Visual C++ and use the OCX technique to communicate with intelligent Lego brick and with the camera of the mobile unit, the other one is a robot monitoring interface. This interface has been realized for monitoring the evolution of robot, it has been divided in two sections: *image, landscape.* It allows to see both image coming from video camera of mobile unit and the landscape of the environment. Image had been sampled with a one image per second; otherwise landscape is update while the robot changes its position, and it is possible to see three different symbols representing: mobile unit, target and obstacles. Its implementation is organized by client/server architecture. The client part consists of a monitoring program (an applet JAVA) that display the information received from robot. In this way a user can look at the robot status from each computer of the distributed system with an Internet browser.

5. Experimental results

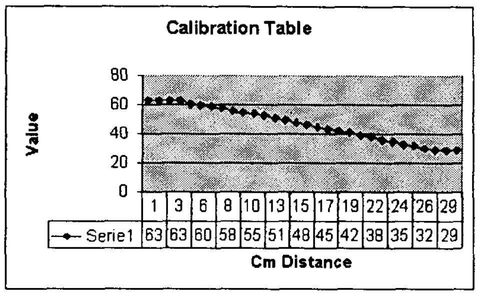
Experiments on autonomous driving have been performed. The purpose was to reach a given target in a scene, avoiding obstacles. The environment is limited and the position of the target is unknown. The number of obstacles and their position is also unknown. In this experiment, target and obstacles are static.

User can decide the starting position of the robot using the standard piloting system above described. After that, user can monitor both the path of the robot and the scene captured by the video camera.

The genetic algorithm, described in Section 3, has been used to pilot autonomously the robot to the target avoiding obstacles. The fitting function is computed on the basis of the information acquired from the video camera and the light sensors[10]. The information acquired from the video camera does not need special calibration and it is essentially normalized in the [0,1] range; this normalization has been obtained by dividing for the maximum intensity detected in the room. The light detectors need of a more accurate calibration phase, because their intensity values must be transformed in a distance value from the obstacle. figure 9 shows the relation between the light value and distance from the obstacle.

6. Conclusions

The distributed solution presented in this paper seems to solve in a very general way the problem of autonomous navigation of a robot in an unknown environment. Evolutionary algorithms are effective and simple to implement. Further studies will regards the stability of the solutions obtained (trajectories), the experimentation with more than one robot of cooperating strategies to reach static and dynamic targets.



	1	3	6	8	10	13	15	17	19	22	24	26	29
Serie1	63	63	60	58	55	51	48	45	42	38	35	32	29

Figure 9 Calibration table for the light sensors.

7. References

[1] V.Di Gesù, G.Gerardi, and D.Tegolo' "M-VIF: a Machine -Vision based on Information Fusion", in proceedings*5th CAIP'95 - Budapest, September* 13-15, 1995.

[2] P.E.Agre and D.Chapman, "What Are Plans for?", Robotics and Autonom. Systems, Vol.6, pp.17-34, 1990.

[3] J.M.Mataric, "Integration of representation into goal-driven behavior-based robot", in *IEEE Trans.on Robotics and Automation*, Vol.8, N.3, pp.14-23, 1992.

[4] S.R.Sutton, "Planning by Incremental Dynamic Programming", in Proc. of the *Eighth Int.Workshop on Machine Learning*, Morgan Kaufmann, pp.353-357, 1991.

[5] S.Yamada, "Learning Behaviurs for Environmental Modaling by Genetic Algorithm", in Evolutionary robotics Husbands, Meyer(Eds.), Lecture Notes in Computer Science, pp.179-191, Springer Verlag,1998

[6] I.Harvey and P. Husbands, "Evolutionary robotics", in Proc. Of IEE Colloquium on Genetic Algorithms for Control System Engineering, London 8 May 1992, 1992.

[7] A.Chella, V.Di Gesù, S.Gaglio, et.al., "DAISY: a Distributed Architecture for Intelligent System", in *Proc.CAMP-97*, IEEE Computer Soc., Boston, 1997.

[8] B. Lenzitti, G. Lo Bosco, D. Tegolo, "An integrated Environement for Dynamic Processes in Distributed Image Analysis System", in *Proc. MIPRO'98*, IEEE Computer Soc., Opatija (Croatia), May 1998, P. Biljanovic, K.Skala, S.Ribaric, L. Budin (Eds.), pp.14-21.

[9] V. Di Gesù, "Multi-agent System and Artificial Vision", in *Computer and Devices for Communication*, pp.393-402 , 1998.

[10] A. Azarbayejani, P. Pentland, "Recursive estimation of motion, structure and focal length", PAMI 562-575, 1995.