NEURAL NETWORK LOCAL NAVIGATION OF MOBILE ROBOTS IN A MOVING OBSTACLES ENVIRONMENT

J. GOMEZ-ORTEGA, E. F. CAMACHO and J. QUERO

Dpto. Ing. de Sistemas y Automática, Univ. de Sevilla, Avd. Reina Mercedes s/n, Spain.
Fax: +34-5-4556849, E-mail:juango@esi.us.es

Abstract. This paper presents a local navigation method based on generalized predictive control. A modified cost function to avoid moving and static obstacles is presented. An Extended Kalman Filter is proposed to predict the motions of the obstacles. A Neural Network implementation of this method is analysed. Simulation results are shown.

Key Words— Mobile robots; guidance system; obstacle avoidance; neural Network; predictive real-time control; extended Kalman filter.

1. INTRODUCTION

One of the most important issues in the design and development of intelligent mobile robots is the navigation problem. This consists of the ability of a vehicle to plan and execute collision-free motions within its environment.

This problem can be divided into two hierarchical levels. The higher level, called global navigation or path planning, is concerned with the generation of a trajectory (space-time) from an initial configuration to a goal configuration, avoiding the known static and mobile obstacles in the environment. At this level, only those obstacles whose situation and motion are previously known are taken into account. Although some works presented in the literature consider the kinematic and dynamic models of the vehicle (Shiller and Gwo 1991), most of them only consider the geometric approach to the problem. Their computation time is not acceptable for real-time control of mobile robots. Some well known solutions are proposed by: Fujimura and Samet (1989), based on including time as one of the dimensions of the model world. This allow them to regard the moving obstacles as being stationary in the extended world; Kant and Zucker (1986) proposed a solution based on the decomposition of the trajectory planning problem (TPP) into two subproblems: the path planning problem (PPP), which is concerned with planning the path to avoid stationary obstacles, and the velocity planning problem (VPP), which is concerned with planning the velocities along the path to avoid moving obstacles. Although this reduces the complexity of the global problem, this solution does not change the predefined path and it cannot avoid moving obstacles with colinear trajectories to the robot's. Erdmann and Lozano-Perez (1987) proposed a solution based on a planner for moving objects that constructs a configuration space each time an object in the scene changes its velocity.

The lower level, called local navigation or guidance, is concerned with driving the vehicle through the trajectory generated by the global planner, now avoiding the unexpected obstacles (static or moving), and compensating the uncertainty of the configurations and motions data used by the global planner, using the real-time enviroment information provided by the sensor system. Usually, at this level, the vehicle kinematics and/or dynamics , and kinematics constraints (such as maximum velocity or acceleration) are considered by the system. Kant and Zucker (1988) have proposed a modification to their algorithm, adding a low control module, which compensates the uncertainty in the velocities of the moving obstacles considered by the planner. The solution proposed by Kathib (1989) is based on the *artificial potential field* (APF) approach, which is one of the most popular methods of real-time static obstacle avoidance. Another approach based on APF has been proposed by Borenstein and Koren (1989). They use the concept of the *certainty grid* to obtain the repulsive forces from the data for the sensors. Griswold and Eem (1990) ,based on Kant and Zuckers approach, propose a solution for unexpected moving objects. The uncertainty in velocity and direction of moving obstacles are considered simply as "noise". This noise suggests that

speed and direction angles of moving obstacles, relative to the robot, should be considered random variables, with a predetermined distribution, at a fixed time. Wang and Tsai (1991) use a modified least-mean-squared-error classification algorithm (used in pattern recognition) to compute a local collision-free navigation path among moving obstacles with no *a priori* position information, in an indoor corridor environment. The trajectories of moving obstacles are predicted by a real-time LMSE estimation algorithm, and the speed value of the robot is determined by the manoeuvering board technique used for nautical navigation. Papageorgiou and Steinkogler (1993) have proposed an optimal control approach to the problem of moving vehicles in changing environments. This approach is the most similar to the proposed one, although they give a numerical solution to the minimization problem while here, a neural network (NN) solution is presented. Papageorgiou and Steinkogler give times of less than a second to solve the optimal problem in some examples, using a very simple kinematic model (heading is not considered). A more complex model, which takes into account the heading of the robot and the velocities of both driving wheels, is considered here. With this model, the numerical solution requires too much computation for real-time.

2. GENERALIZED PREDICTIVE CONTROL

The *Generalized Predictive Control (GPC)* proposed by Clarke *et al.* (1987) is an optimal control technique, that has inspired much research work in the recent years. The objetive of the GPC is to drive future system outputs (in our case the robot position and orientation) close to their desirable values in some sense, bearing in mind the control activity required to do so. This is done using a *receding horizon* approach for which, at each sample instant, using a prediction model to generate a set of predicted outputs, some appropriate quadratic function J of the future errors and controls is minimized, assuming that after some *control horizon* H further increments in control are zero. Only the first control is applied, resulting in a control law that belongs to the class known as *Open-Loop-Feedback-Optimal control.* The cost function J can be of the form:

$$J(H,\Delta V)=\sum_{i=1}^{H}[X(t+i)-X_d(t+i)]^2$$

$$+\sum_{i=1}^{H}\lambda[\Delta V(t+i-1)]^2$$

where X is the vector of predicted outputs, X_d is the vector of desired values for X, V is the control variables vector, and λ is a weighting factor. Some research has been done aiming to apply this technique to the path tracking problem (Ollero and Amidi 1991).

This paper proposes a modification of the cost function J to include a term that penalizes the proximity to any obstacle (static or moving) in any of the H next sample instants. This leads to an obstacle avoidance with low control cost. The idea is that if it is noticed that a collision may be produced in the future, it will begin to avoid this situation with smooth control actions. The problem can be defined as follows: given a trajectory (space-time), drive the robot to follow it using the on-line sensor data, avoiding the unexpected obstacles found in the environment, and compensating the uncertainties in the data (positions of the obstacles) used by the global planner. The new cost function J is (see Fig. 1):

$$J(H,\Delta V)=\sum_{i=1}^{H}[X(t+i)-X_d(t+i)]^2$$

$$+\sum_{i=1}^{H}(\lambda_1([\Delta V_r(t+i-1)]^2+[\Delta V_l(t+i-1)]^2)$$

$$+\lambda_2[V_r(t+i-1)]-V_l(t+i-1)]^2)$$

$$+\sum_{j=1}^{NMO}(\sum_{i=1}^{H}\frac{\xi_1}{P_1(t+i)[dist(X(t+i),XMO^j(t+i))]^2})$$

$$+\sum_{j=1}^{NSO}(\sum_{i=1}^{H}\frac{\xi_2}{P_2(t+i)[dist(X(t+i),XSO^j(t+i))]^2})$$

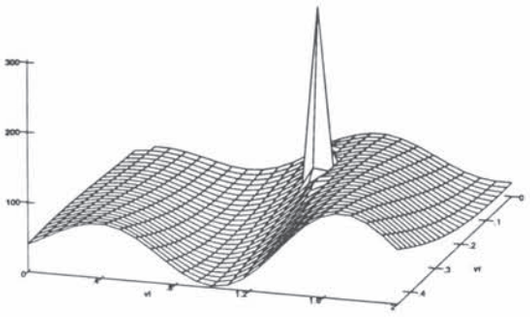

FIG. 1. Objetive Function $J(H,\Delta V)$ for H=1 and one static obstacle

where $X(i)=\{x(i),y(i),\theta(i)\}$ is the position and orientation vector of the robot in the sample instant i, $X_d(i)=\{x_d(i),y_d(i),\theta_d(i)\}$ is the desired position and orientation vector for the control horizon, and $XMO^j(i)=\{xmo^j(i),ymo^j(i)\}$ and $XSO^j(i)=\{xso^j(i),yso^j(i)\}$ are the positions of the moving and static obstacles in the sample instant i. V_r and V_l are the right and left

velocities of the two driving wheels, which are the control variables. NMO and NSO are the number of moving and static obstacles respectivly, and λ_1, λ_2, ξ_1 and ξ_2 are weighting factors. P_1 and P_2 are the covariance matrices of the predictions of the future positions of the obstacles, and *dist* is the euclidean distance between the robot and the obstacles.

For this formulation a model is needed to predict the future positions and orientations of the robot and a model to predict the positions of the moving obstacles. The following kinematic model (which corresponds to a differential-drive vehicle) is used for the first issue:

$$\theta(k+1) = \theta(k) + AT$$

$$x(k+1) = x(k) + \frac{V}{A}(\sin(\theta(k) + AT) - \sin(\theta(k)))$$

$$y(k+1) = y(k) - \frac{V}{A}(\cos(\theta(k) + AT) - \cos(\theta(k)))$$

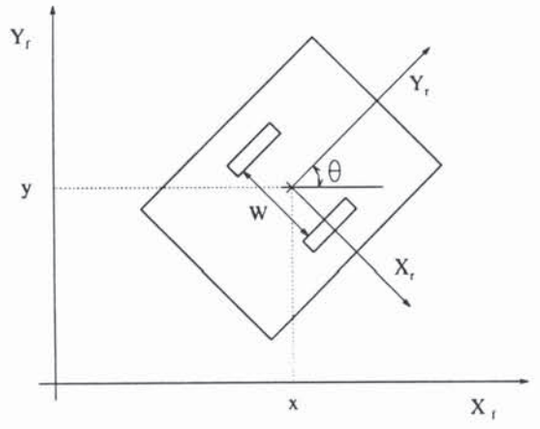

FIG. 2. Reference Frame

where x, y, θ are the position and orientation of the robot in a fixed reference frame, $A = \frac{V_r - V_l}{W}$ and $V = \frac{V_r + V_l}{2}$. T is the sample time and W is the distance between wheels (see Fig. 2).

3. MOBILE OBSTACLE MOTIONS PREDICTION

To predict the positions of the moving obstacles in the future, when the velocity of the obstacle is assumed to be constant between sampling intervals, the following non linear model can be considered:

$$x_o(k+1) = x_o(k) + V(k+1)cos(\theta_o(k+1))$$

$$y_o(k+1) = y_o(k) + V(k+1)sin(\theta_o(k+1))$$

$$V(k) = ((x_o(k) - x_o(k-1))^2 + (y_o(k) - y_o(k-1)^2)^{1/2}$$

$$\theta_o(k) = arctan(\frac{y_o(k) - y_o(k-1)}{x_o(k) - x_o(k-1)})$$

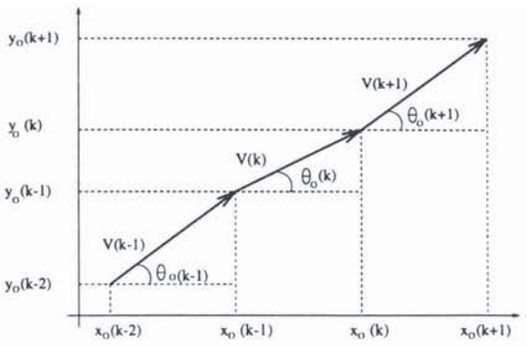

FIG. 3. Obstacle motion parameters

where x_o, y_o are the position of the mobile obstacle, V is the lineal velocity between two positions and θ_o is the angle of the velocity vector in respect to the horizontal x axis (see Fig. 3).

Now, the following hypothesis are made:

$$V(k+1) = V(k)$$

$$\Delta\theta_o(k+1) = \Delta\theta_o(k) = \theta_o(k) - \theta_o(k-1)$$

The *Extended Kalman Filter* approach is applied to this model to predict the future positions of the mobile obstacle and their covariance matrices $P_1(k+j|k)$. The result of the propagation cycle for H periods of prediction are the following equations:

$$\hat{x_o}(k+j|k) = \hat{x_o}(k|k) + V(k)T\sum_{i=0}^{j-1} cos(\mu(i))$$

$$\hat{y_o}(k+j|k) = \hat{y_o}(k|k) + V(k)T\sum_{i=0}^{j-1} \sin(\mu(i))$$

$$\mu(i) = 2arctan(\frac{\hat{y_o}(k+i|k) - \hat{y_o}(k+i-1|k)}{\hat{x_o}(k+i|k) - \hat{x_o}(k+i-1|k)})$$

$$- \arctan(\frac{\hat{y_o}(k+i-1|k) - \hat{y_o}(k+i-2|k)}{\hat{x_o}(k+i-1|k) - \hat{x_o}(k+i-2|k)})$$

With the actualization cycle only $\hat{x_o}(k+1|k+1)$, $\hat{y_o}(k+1|k+1)$ and the actualized covariance matrix $P_1(k+1|k+1)$ are calculated, because only measures of the $k+1$ instant are available; but these reduce the covariance matrix for the next predictions.

To take into account the prediction uncertainties, the distances between the robot and the obstacles are penalized with the covariance matrices obtained from the Kalman Filter equations. In the computation of the robot-obstacles distances, it will be assumed that the obstacles have a circular form.

4. THE NEURAL NETWORK APPROACH

The minimization of the cost function J cannot be obtained in real time with numerical methods. So, a Neural Network solution is proposed, which guarantees real time for the robot control. The Neural Network approach for robot guidance has been proposed by other researches (Pomerlau 1990), (Meng and Picton 1992).

The architecture of the NN controller consist of a single hidden layer backpropagation network.

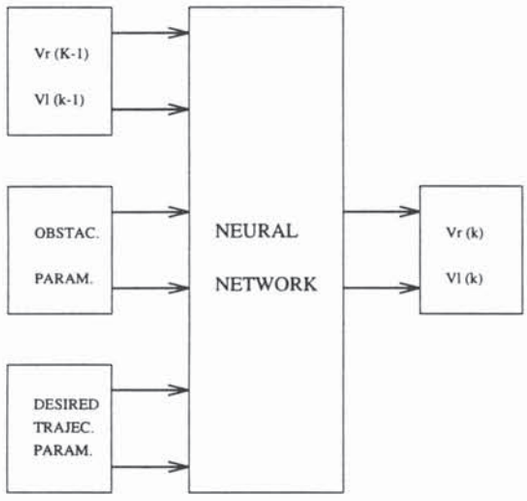

FIG. 4. Neural Network Scheme

The input layer consists of three modules (see fig. 4). The first one includes two network inputs associated with the control values in the last sample instant. The second module corresponds to the obstacle parameters: two inputs for each static obstacle (distance and orientation), and five inputs for each moving obstacle $(x_o(k), y_o(k), V(k+1), \theta_o(k+1), \Delta\theta_o(k+1))$. The third module includes ten network inputs which correspond to the parametrization of the desired trajectory in the next H sample instants. These parameters consist of the location and orientation of the first point of the local trajectory, the curvature of the next H points and an average desired velocity. The output layer consists of two nodes which correspond to the control command: the left and right wheels velocities.

The training module consists of a backpropagation scheme, shown in fig. 5, where the training patterns are solved by a GPC module which uses a numerical method to generate the outputs. The training patterns are selected properly to represent all the possible situations of driving among obstacles. Also, a local reference system is used to reduce the parameters range of variation.

The NN approach will be as follows:

- At time k, obtain the obstacles position parameters from the sensor system. Predict the future positions of the moving obstacles.

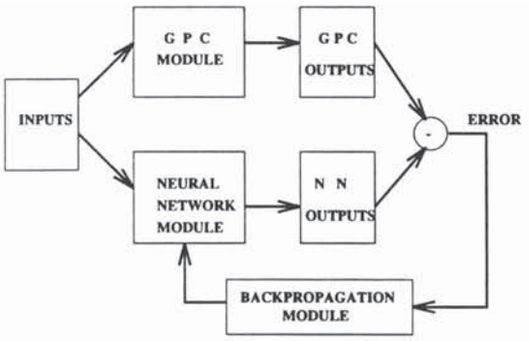

FIG. 5. Neural Network training Scheme

- Using the NN, predict the future H positions of the robot as if no obstacles where encountered.
- Detect if a collision may be produced in the next H sample instant.
- if not, apply the first NN control outputs to the vehicle.
- if a collision may be produced, then use the predicted positions and orientations of the robot as the new desired trajectory, and compute the NN again, now considering the obstacles.

The use of the predicted positions of the robot as a new desired trajectory is justified by the reduction of the number of training patterns necessary to obtain a good performance because a symetry analysis has can be made. In this analysis it has been supposed that the robot is always on the desired path.

Finally, it is important to notice that this method solves, at the same time, the problems of finding avoidance local trajectories and the control one. This guarantees that the trajectory followed by the robot is continuous in curvature, avoiding discontinuities in the wheels velocities, which produces errors between the predicted and the real position of the vehicle.

5. RESULTS

The proposed control structure has been tested by simulation with a model of the *Labmate* mobile robot (T. R. C. 1989). The NN used consisted of seventeen input neurons corresponding to the past control actions, past mobile obstacles positions and future refernce trajectory. The hidden layer was composed of 35 neurons and the output layer consisted of two neurons corresponding to the left and right wheel velocities for the mobile robot.

The NN was trained in a supervised manner as described previously. The control horizon choosen for the GPC was made equal to six. And the

weighting factors were given the following values: $\xi_1 = 3, \xi_2 = 3, \lambda_1 = 63, \lambda_2 = 10$. Where λ_1 and λ_2 correspond to the weights of the module of the velocity and to the angular velocity respectively. The high value of λ_1 is to make sure that the GPC will not choose the *easy* solution of stopping the robot and wait for the mobile object to pass. The relatively high value of λ_2 ensures a smooth trajectory. Fig. 6 shows the evolution of the error function $E(i) = \sum_{i=1}^{N}(O_d(i) - O(i))^2$ of the training phase, where N is the number of training patterns, and $O_d(i)$ and $O(i)$ are the desired and network output for the iteration i.

The simulated results obtained for three situations, different from the training cases, can be seen in fig. 7. The trajectories shown on the left hand side of fig. 7 correspond to the solutions obtained when applying the GPC controller while the trajectories shown on the right hand side correspond to the solutions obtained with the NN. Fig. 7.1 corresponds to an obstacle which is coming towards the mobile robot, while figures 7.2 and 7.3 correspond to obstacle trajectories crossing the mobile robot trajectory. The trajectories shown in fig. 7.3 correspond to a case where the initial position of the mobile robot is not sitting on the desired path. In all cases, the GPC and NN make the mobile robot take the necessary evasive control actions. As expected, the NN reproduces the behaviour of the GPC control quite well and takes only small fraction of the computation time required for solving the GPC which has to be solved using a numerical optimization algorithm (powell method has been used here).

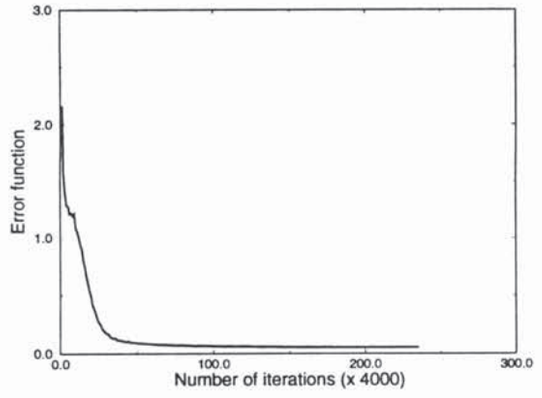

FIG. 6. Evolution of the error function

6. CONCLUSIONS

A local navigation method based on the GPC control algorithm has been proposed. A new cost function which penalizes the inverse of the distance between the robot and the moving and static obstacles has been presented. A prediction method for the future positions of the mobile obstacles has been studied. Finally a Neural Network implementation of the GPC method has been proposed to obtain real-time performance. Simulation results have been presented.

7. ACKNOWLEDGEMENT

The authors would like to acknowledge the *CICYT* for funding this work under grants TAP93-0408 and TAP93-0581.

REFERENCES

Borenstein, J. and Y. Koren (1989). *Real-Time Obstacle Avoidance for Fast Mobile Robots.* IEEE trans. on System, Man, and Cybernetics, vol. 19, n0 5, pp 1179-1187.

T. R. C. (1989). *Labmate reference manual.* (Transitions Research Corporation).

Clarke, D. W., C. Mohtadi and P. S. Tuffs (1987). *Generalized Predictive Control - Part I. The Basic Algorithm.* Automatica, vol 23, no 2, pp 137-148.

Erdmann, M. and T. Lozano-Perez (1987). *On multiple moving objects.* Algorithmica, vol. 2, no. 4 pp 477-521.

Fujimura, K. and H. Samet (1989). *A Hierarchical Strategy for Path Planning Among Moving Obstacles..* IEEE trans. on robotics and automation, vol. 5, no. 1, pp 61-69.

Griswold, N. C. and J. Eem (1990). *Control for mobile Robots in the Presence of Moving Objects.* IEEE trans. on Robotics and Automation, vol 6, no 2, pp 263-268.

Kant, K. and S. W. Zucker (1986). *Toward Efficient Trajectory Planning: The Path-Velocity Decomposition..* The Int. journal of Robotics Research, vol. 5, no. 3, pp 72-89.

Kant, K. and S. W. Zucker (1988). *Planning Collision-Free Trajectories in Time-varying environments: A Two-Level hierarchy..* Proc. IEEE Int. Conf. on Robotics and Automation, pp 1644-1649.

Kathib, O. (1989). *Real-time Obstacle Avoidance for Manipulators and Mobile Robots.* The Int. Journal of Robotics Research, vol.5, no. 1, pp 90-98.

Meng, H. and P. D. Picton (1992). *Obstacle Avoidance Using a Neural Network Controller and Visual Feedback.* Proc. SICICA 92, pp 563-568.

Ollero, A. and O. Amidi (1991). *Predictive Path Tracking of Mobile Robots. Aplications to the CMU Navlab..* Proc. IEEE Fifth Int. Conf. on Advanced Robotics (Pisa), vol 2, pp 1081-1086.

Papageorgiou, M. and A. Steinkogler (1993). *Real-Time Optimal Control of Moving Vehicles in Changing Environments.* Proc. of the 12th IFAC World Congress (Sydney), vol 3, pp 107-112.

Pomerlau, D. A. (1990). *Neural Network Based Autonomous Navigation, (Vision and Navigation. The Carnegie Mellon Navlab).* C.E. Thorpe Editor., Kluwer Academic Publishers, pp 83-93.

Sharma, R. (1992). *Locally Efficient Path Planning in an Uncertain, Dynamic Environment Using a Probabilistic Model.* IEEE trans. on Roboics and Automation, vol 8, no 1, pp 105-110.

Shiller, Z. and Y. Gwo (1991). *Dynamic Motion Planning of Autonomous Vehicles.* Trans. IEEE on Robotics and Automation, vol 7, no 2, pp 241-249.

Wang, L. and W. Tsai (1991). *Collision Avoidance by a Modified Least-Mean-Square-Error Scheme for Indoor Autonomous Land Vehicle Navigation.* Int. Journal of Robotics Systems, vol 8, no 5, pp 677-698.

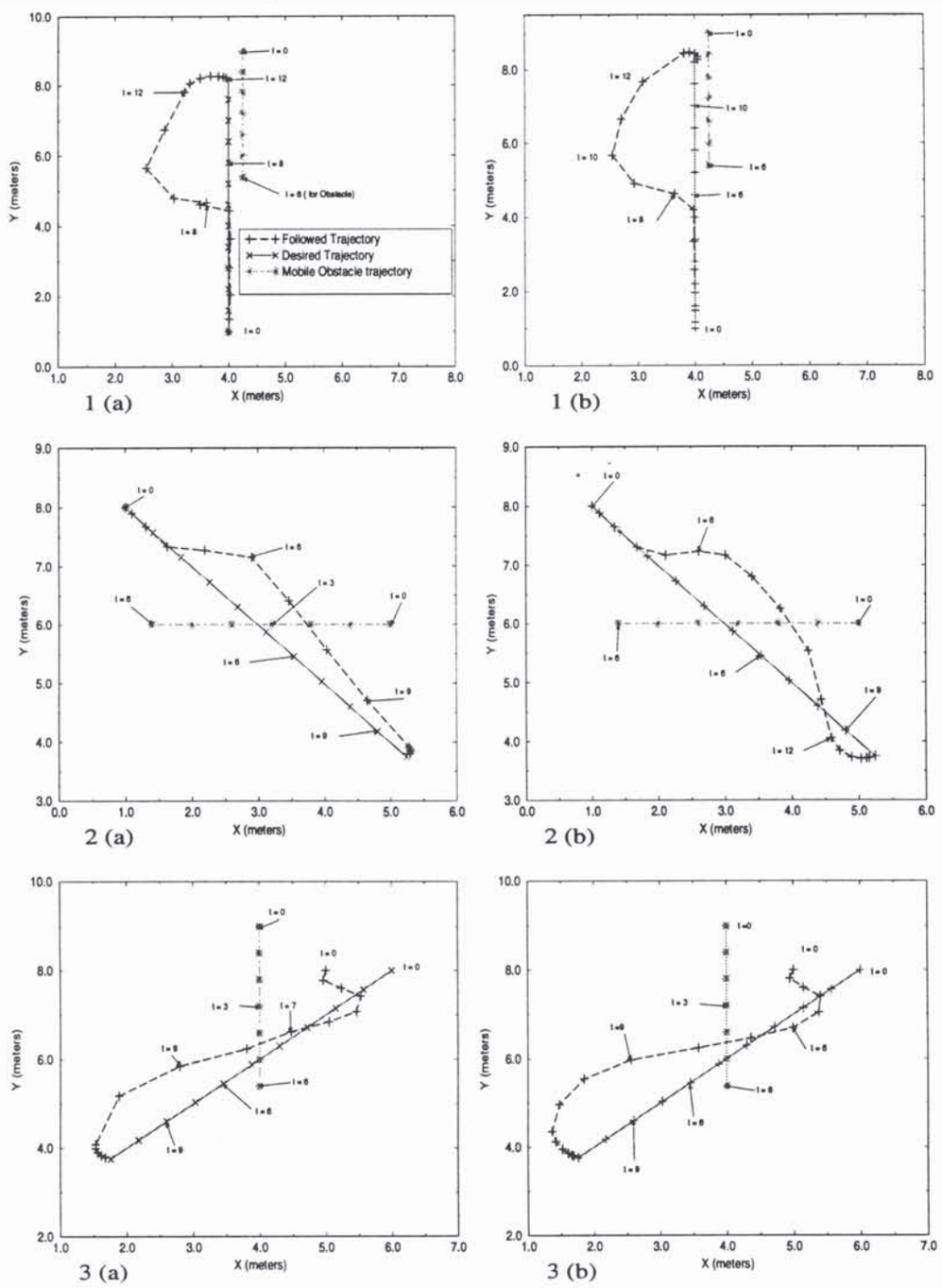

FIG. 7. Results.The left figures correspond to the numerical solved GPC. The right figures correspond to the Neural Network solution.