Autonomous Vehicle Obstacle Avoiding and Goal Position Reaching by Virtual Obstacle

Ranka Kulic
Faculty of Maritime of Studies
Dobrota 36
Kotor
MONTENEGRO
rkulic2001@yahoo.com

Zoran Vukic
Faculty of Electrical Engineering and Computing
Unska 3
Zagreb
CROATIA
zoran.vukic@fer.hr

Abstract – **The problem of dynamic path generation for the autonomous vehicle in environments with unmoving obstacles is presented. Generally, the problem is known in the literature as the vehicle motion planning. In this paper the behavioural cloning approach is applied to design the vehicle controller and virtual obstacle is used also in the goal position reaching. In behavioural cloning, the system learns from control traces of a human operator. To learn from control traces the machine learning algorithm and neural network algorithms are used. The goal is to find the controller for the autonomous vehicle motion planning in situation with infinite number of obstacles.**

I. INTRODUCTION

Techniques for the automatic planning of the motion have advanced substantially during the last twenty years [1], [3]. In its simplest form, the motion planning problem can be defined as follows. Let B be the autonomous vehicle consisting of collection of rigid subparts having a total k degrees of freedom, and let B be free to move in two - or three - dimensional space V, avoiding obstacles whose geometry is known. For a given initial position S and a desired target position G of B, the task is to determine whether there exist a continuous obstacle - avoiding motion of B from S to G, and if so, to find such a motion. The simplest collision avoidance algorithm fall into the generate and test paradigms. A simple path from S to G, usually a straight line, is hypothesized and then it is tested for potential collisions between B and obstacles. If collision is detected, a new path is proposed using information about detected collision. This process repeats until no collision is detected. But in spite of its simplicity these methods have not found significant application. They have couple of fundamental drawbacks. One of these is inability to propose a radically different and better path from local information about potential collision. Another is that collection methods are based on a configuration space approach [2], [3], [4]. The configuration of rigid body is set of independent parameters that characterize the position of every point of it. For the vehicle B some regions represent illegal configuration space because there are obstacles. So the find path (the vehicle motion planning) approach means that the vehicle have to be shrinked to dimension of a reference point and to grow obstacles, i.e. to compute forbidden regions for the reference point. Finding the path of the vehicle is in this way transformed in finding the path of the reference point, moving in configuration space and avoiding obstacles. For the vehicle B moving from position S to position G the desired path is the shortest path which takes into account all constraints of the position of the reference point of B. It is possible to obtain this path by generating an appropriate graph (visibility graph, connectivity graph,…) and finding a path from graph node S to graph node G. Fundamental problem arising during the implementation of these methods is concerned with the obstacle growing and graph searching. For both cases the problem complexity is very large. For example, while in the planar case the shortest path can be found in time that is in the worst case the quadratic in the number of obstacle and edges, finding the shortest path between two points in three dimensions, which avoids a collections of polyhedral obstacles is NP – hard (has exponentional complexity) [2], [3]. This is a specially very large problem if the world model changes. Other classes of approaches are developed as alternative to the traditional ones. A typical such approach [5] regards the obstacles as the sources of repelling potential field, while the goal position G of the vehicle is considered as a strong attractor. The vehicle B follows potential gradient vector field. These approaches try to find the local minimum only. As the next, we can consider the direction which assumes problem solution capability of the vehicle motion planning based on the transfer of skill into controllers of the vehicle [6], [7]. A more detailed survey of these methods is given in [11]. In our case the behavioural cloning approach is applied to design the vehicle controller and virtual obstacle is used also in the goal position reaching. In behavioural cloning, the system learns from control traces of a human operator. To learn from control traces the machine learning algorithm and neural network algorithms are used. The goal is to find the controller for the autonomous vehicle motion planning in situation with infinite number of obstacles

II. THE AUTONOMOUS VEHICLE MOTION PLANNING BASED ON THE BEHAVIORAL CLONING

Skill is often defined as an ability to perform a high quality sensory-motor coordination and control in real time. Humans exhibit such a skill as a result of training over a period of time. It would be especially useful if we can also provide systems with the capability of acquiring such a skill. In this sense two approaches have been known. The former treats the skill as something that could be acquired in a dialogue with an operator. In that process it is expected from

ISBN: 978-960-89028-5-5

the operator to describe skills he has been governed over the control of the vehicle. Here arise some difficulties because the skill is human subconscious action and so cannot be completely consciously and reliably described. An alternative approach is to start from the assumption that the skill can be reconstructed, using learning algorithms, from the manifestation trace of it [6], [7], [8]. Sammut, Hurst, Kedzier and Michie [7] give a description of the solution belonging to the flight control area. Our idea [9], [10], [11] is to enable an intelligent system to learn from the examples (operator's demonstrations) to control a vehicle avoiding obstacles, like the human operator does. In section III the intelligent controller concept is given. In section IV the results in controller development are presented. In section V the conclusion is given and possibilities of further development are discussed.

III. ELEMENTS OF CONCEPT CONTROLLER DEVELOPMENT

A. Introduction

The idea of development of controller by cloning the human operator [6] is illustrated in Fig. 1.

In the problem of the obstacles avoiding, this idea could be interpreted in the following way. During one of the simulation phases, called the *training* phase, operator controls the vehicle avoiding unmoving obstacles located in its working space. In that phase variables that are evaluated as relevant have been written into LOG FILE. In the second simulation phase, called the *learning* phase, the machine learning program, taking data from LOG FILE, generates differential equations that define the operator's trajectory. In the third phase, called the *verifying* phase, operator is excluded from the vehicle control process and the vehicle is controlled solely by a clone induced in the learning phase. This development of process phases are needed for the repeated changing of both problem domain representation and/or learning system regarding cloning success criterion. Using "several" vehicle and environment models (problem domain representations) and "several" machine learning systems, it is attempted to find an appropriate domain model and an appropriate machine learning system in order to control the vehicle to avoid obstacles according to cloning success criterion. Now, according to the previous case [17 – 23], the environment model, which uses the virtual obstacle, enables application the same relation to avoid real obstacles and to reach the goal position of the vehicle.

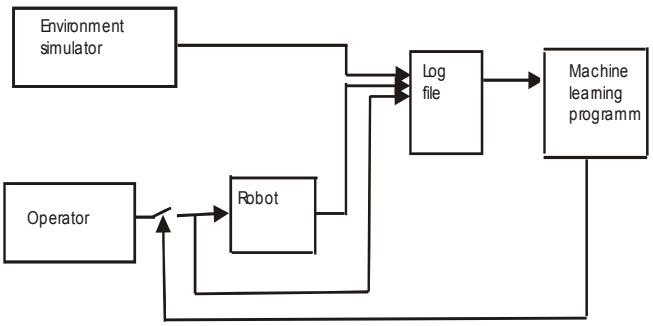


Fig. 1. Behavioral cloning process.

B. Learning Systems

Two learning systems were used. *GoldHorn* is based on algorithm of multiparameter optimization called *ameba*. It attempts to identify the operator's trajectory by finding constraints among the state variables. To induce such constraints GoldHorn uses the form of ordinary differential equations ([12]). To do that GoldHorn first introduces new variables by repeatedly applying operators, a such as multiplication, sin (x), cos(x), log(x), sign(x), |x|, e^x, to the state variables. Differential equations are generated from these variables using linear regression. The significance of the equations is judged by two measures: the multiple correlation coefficient and the normalized deviation. GoldHorn does structural synthesis of new forms of equations. GoldHorn treats noisy data, which makes it applicable to the modeling of real world systems. The operator's trajectory is learned by giving GoldHorn a subset of variables (attributes), contained in the execution trace. Control action is deliberately left out, since it can be computed if the desired next state and an approximate system model are known. GoldHorn then finds the constraints among the named variables in the form differential equations and ranks them according to their significance (error estimates). One or more most significant equations are used as the constraints to define the operator's trajectory.

The radial basis function (RBF) network is motivated by biological neurons, with locally tuned response. These nerve cells have response characteristics that are selective for some finite range of the input signal. These models are described by more authors ([15], [24], [25]).The model is commonly referred to as the *radial basis function (RBF) network.* The most important attribute that distinguishes the RBF network from earlier radial based models is its adaptive nature. It generally allows to utilize a relative small number of locally tuned units. RBF network were independently proposed by several authors ([15], [24], [25]).The following is a description of the basic RBF architecture Figure 2. The RBF network has a feedforward structure consisting of a single hidden layer of Q locally tuned units, which are interconnected to an output layer of L linear units. All hidden units simultaneously receive R dimensional real –valued input vector p. Notice the

absence of hidden layer weights. Each hidden unit output a_j is obtained by calculating the closeness of the input p to n dimensional parameter vector μ_j (IW in Figure 2). This parameter is associated with jth hidden units. The response characteristics of the jth hidden units are given by

$$a_j[p]=K(-\frac{\|p-\mu_j\|}{2\sigma_j^2}),$$

where K is a strictly positive radially symmetric function (kernel) with a unique maximum at its center μ_j and which drops off rapidly to zero away from the center. The parameter σ_j is the width of the receptive field of the input space for unit j. This means that a_j has an appreciable value only when distance $\| p- \mu_j\|$ is smaller than the width σ_j. A specially but commonly used RBF network assumes a Gaussian basis function for the hidden units, i.e.:

$$a_j[p] = \exp(-\frac{\| p-\mu_j \|^2}{2\sigma_j^2}),$$

where σ_j and μ_j, are the standard deviation and mean of the jth unit. The norm is the Euclidian norm. The output of the RBF network is the L dimensional vector a2, which is given by:

$$a2 = \sum_{j=1}^{L} LW_{ij} a_j \ .$$

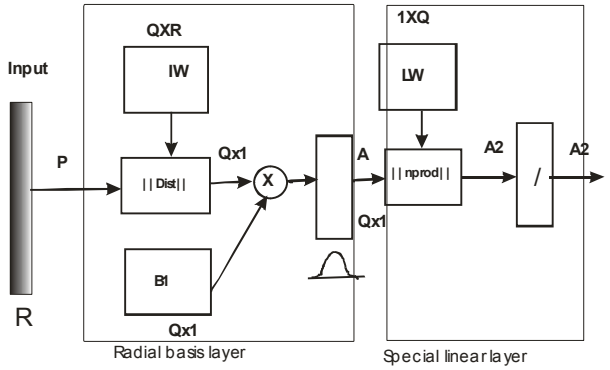


Fig. 2 A radial basis function neural network consisting of a single hidden layer of locally tuned units that is fully interconnected to an output layer of linear units.

RBF networks are best suited for approximating continuous real valued mappings f : $R^n \rightarrow R^L$, where n is sufficiently small. According to the previously named equations the RBF network may be as approximating a desired unction f(p).The degree of accuracy can be controlled by three parameters the number of basis functions to be used, their location and their width. The RBF networks are considered as universal approximators. Consider training set of m labeled pairs $\{x_j, y_j)$ which are represent samples of a continuous multivariate function. The criterion function is an error function E to be minimized over the given training set. It is desired to develop a training method that minimizes E by updating the free parameters of the RBF. These parameters are σ_j, μ_j and w_{ij}. One of the first training methods that comes to mind is a fully supervised gradient descent methods over E, as it is given in [21].

A neural network algorithm as an advancement according to the RBF is given in [20] and that model produces quite good results, as it is given in section IV.D.

C. The Autonomous Vehicle Kinematical Model

The following model of the vehicle is used: $\psi(n)=\psi(n-1) + \Delta t\, r(n)$, $x(n) = x(n-1) + \Delta t\, v \cos(\psi(n))$, $y(n) = y(n-1) + \Delta t\, v \sin(\psi(n))$,where: ψ is rotation angle of the vehicle (ψ=0 if the vehicle is oriented parallel to x-axis); r and v are control variables meaning desired rotation speed and translation speed; x, y are position coordinates, Δt is sample time and n is the time index. The vehicle is represented as a geometrical figure. Its dimensions are not neglected. It is necessary to point out this as a very important fact. The selection of the vehicle model is inspired by conventional methodology that is used in control systems for a given path [17].

D. Environment Models

Environment model amounts to the distances of the vehicle gravity center from the goal position (dx_G and dy_G) and from the obstacles (d_i). dx_G and dy_G are calculated as: $dx_G= x - x_G$, $dy_G= y - y_G$, where x_G, y_G are the goal position coordinates. Obstacles are represented by its characteristic values, as illustrated in Fig. 3. Obstacle area is divided into sub-areas.

A procedure, for the simulation purpose, calculating the vehicle distance d_i from *i-th* obstacle is explained for a triangle obstacle as: SubArea-4: *if((x>=x_1)and(y>=y_B)) then* $d_i = ((x-x_1)^2 + (y-y_1)^2)^{1/2}$, SubArea5: *if((x>=$x_1$)and(y<$y_B$)and(y>=$y_C$)) then* $d_i = |[y+[(y_1-y_2)/(x_2-x_1)]x + [(y_2-y_1)/(x_2-x_1)]x_1-y_1] / [[(y_1-y_2)/(x_2-x_1)]^2+1]^{1/2}$ |, ……y_B and y_C are lines that are normal onto the line BC at points B and C. According to the previous cases the environmental model has additional characteristics as it is given in Section IV.B and IV.C.

E. Cloning Success Criterion

Performance error is very important for the evaluation of the quality of clone that was constructed. Regarding the ideal case the goal concept and the approximation concept of the vehicle trajectory are identical and the performance error is

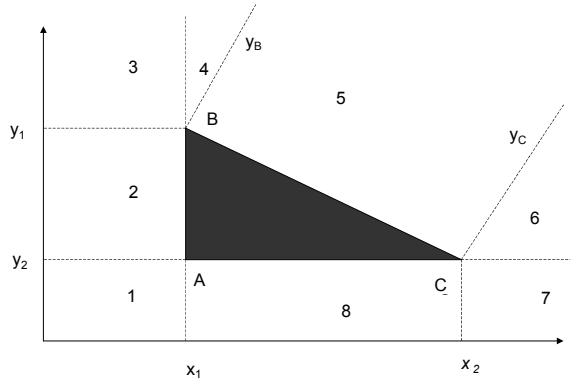


Fig. 3. Triangular obstacle.

equal to zero. Ideal trajectory in x-y plane without obstacles is, for example, a straight line between the start S and the goal G positions of the vehicle, as it is specified in Section IV.C. Operator, in a training phase, mostly does not manage to realize this trajectory. Position error E_{xy} is based upon a distances $d_{op(i)}$ and $d_{cl(i)}$ of operator and clone trajectories, respectively, from the named straight line. Our problem is to avoid obstacle and so we can consider only E_{perf} as:

$$E_{xy}=\frac{\sum_{j=1}^{N}\frac{\left|d_{op}(j)-d_{cl}(j)\right|}{\max(d_{op}(j)-d_{cl}(j))}}{N} \quad (1)$$

For avoiding n obstacles we have to find $(d)_m = \min\{ d_i, i=1,..n\}$ in order to define:

$$E_{xy}=\frac{\sum_{j=1}^{N}\frac{\left|(d_{cl}(j))_m-(d_{cl}(j-1))_m\right|}{\max(d_{cl}(j))_m-(d_{cl}(j-1))_m}}{N-1} \quad (2)$$

Eventually, we can say that between two clones more successful is the one which produces lower performance error regarding equations (1) and (2).

IV. EXPERIMENTS AND RESULTS

To form training instances the task was *only to avoid obstacles. The vehicle start position was its goal position. The minimal number of the vehicles traveling from the start to the goal position was to chosen be one.* It is the framework for selecting appropriate training instances. The idea is to combine this control strategy with a control strategy without obstacles in order to form *a full controller*, which enables to avoid obstacles and to reach the goal position of the vehicle. When an obstacle was included, the vehicle trajectory for the training scene is illustrated in Fig. 4.

A. Experiments using GoldHorn

Treating the problem without obstacles, GoldHorn found the next equation for the desired rotation angle:

$$\psi_{desired}=\frac{0.55311*(y-y_G)-0.01679}{x-x_G} \quad (3)$$

So, we defined: $(d)_m = \min\{ d_i, i=1,..n\}$. The training process was repeated (Fig. 4.) and GoldHorn managed to find the next equation of the desired rotation angle:

$$\psi_{desired}=\frac{-d_m+0.13735}{0.0064} \quad (4)$$

Now, the scenes created by three obstacles have been observed, (see Fig. 5.), using the control strategy:

$$r=\frac{\psi_{desired}-\psi}{\Delta t} \quad (5)$$

and $v=0.1$.

For the clone based on equations (4) and (5), it is obtained E_{xy} = 0.000141. Eventually, we can say, there is a strong reason to use relations (4), (5) and relation $(d)_m = \min\{ d_i, i=1,..n\}$ to solve the problem of avoiding n obstacles. The clone based on equations (4) and (5) enables only the obstacle avoiding. However, it is needed to combine these equations with the equation (3), [11]. To avoid the vehicle traveling through the narrow passages between obstacles it is

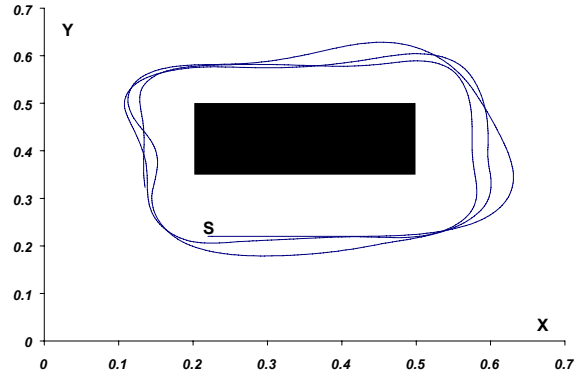


Fig. 4. The vehicle trajectory used to gather training examples.

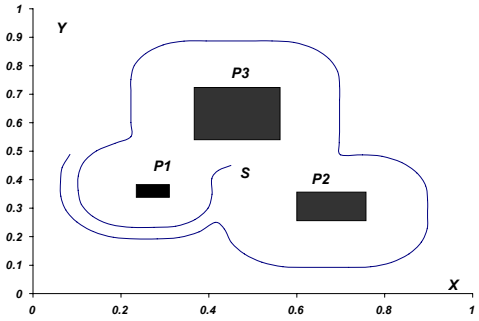


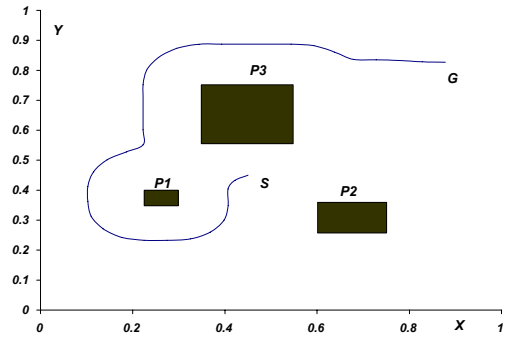
Fig. 5. Test solved by clone based on equations (4) and (5).

needed to define a rule regarding their dimensions. For the control variable *v* of the vehicle, a tested control strategy is given in [11].

B. Experiments with the Vehicle Goal Position as an Obstacle

In the next experiments the vehicle goal position is treated as an obstacle. The fashion is to take the distances (d_{goal}) of the vehicle gravity center from the goal position as the distance from i – th obstacle. Then expanding distance set we defined: $(d)_m = \min\{d_{goal}, d_i, i=1,..n\}$. To define the vehicle trajectory is defined a rule: if ($d_{min} = d_{goal}$) then $\Psi_{desired} = \Psi_{line}$ else $\Psi_{desired} = \Psi_{avoid}$, where Ψ_{line} is defined by equation (3) and Ψ_{avoid} is defined by equation (4). The described method is tested using scene a) S=(0.45, 0.45), G= (0.8,0.8), P1=(0.23,0.35,0.3,0.4), P2=(0.6,0.25,0.75,0.35), P3=(0.35,0.55,0.55,0.75), Fig. 5a); b) S=(0.45, 0.45), G=(0.85,0.8), P1=(0.23,0.35,0.3,0.4), P2=(0.6,0.25,0.75,0.35), P3=(0.35,0.55,0.55,0.75), Fig. 6b).

Scenes in Fig. 6a) and in Fig. 6b) are different only according to the x – axis of the vehicle goal position. According to the terminal position of the vehicle, the differences are relatively small, but the trajectory in Fig. 6b) is greater then from the trajectory in Fig. 6a) and they are different to all appearances. Obviously, the exposed method generates the vehicle trajectory which is not an optimal



a)

b)

Fig. 6. Test solved by clone using the goal position as an obstacle.

vehicle trajectory. It is the disadvantage. But the method is very simple and it enables to the vehicle to reach the goal position using only equation (4). It is the advantage.

C. Experiments with the Virtual Obstacle Based on Line Connecting the Goal and the Start Position

In the next experiments the line which connect the goal and the start position serves to be formed a virtual obstacle as it is illustrated in Fig. 7. According to the Fig. 7 it is posible to form relations:

$$\begin{aligned} &X1=XS+0.05, Y1=YS;\\ &X2=XS, Y2=YS+0.05;\\ &X3=XG-0.05, Y3=YG;\\ &X4=XG, Y4=Y4-0.05; \end{aligned} \quad (6)^1$$

$$YP1=Y1+\frac{Y2-Y1}{X2-X1}(X-X1) \quad (7)$$

$$YP2=Y2+\frac{Y3-Y2}{X3-X2}(X-X2) \quad (8)$$

$$YP3=Y3+\frac{Y4-Y3}{X4-X3}(X-X3) \quad (9)$$

$$YP4=Y1+\frac{Y4-Y1}{X4-X1}(X-X1) \quad (10).$$

It is known that the normal distance of the point M(X,Y) from the line aX+bY+c=0 is:

$$d=\frac{|aX+bY+c|}{\sqrt{a^2+b^2}} \quad (11).$$

A procedure, for the simulation purpose, calculating the vehicle distance from the virtual obstacle is given as: SubArea-1: *if((Y<=YP1)and(Y<=YP2)) then* $d_v = ((X-X1)^2 + (Y-Y1)^2)^{1/2}$, SubArea2: *if((Y>=YP4)and(Y<YP2)and(Y<=YP3)) then* $d_v = |Y+[(Y1-Y2)/(X2-X1)]X+[(Y2-Y1)/(X2-X1)]X1-Y1]|/[[(Y1-Y2)/(X2-X1)]^2+1]^{1/2}$,..... and so on for all of from the eight areas. This approach enables the vehicle moving from the start to the goal position using only equation (4), for example, (without equation (3)), expanding distance set by defining: $(d)_m = \min\{d_v, d_i, i=1,..n\}$.

.

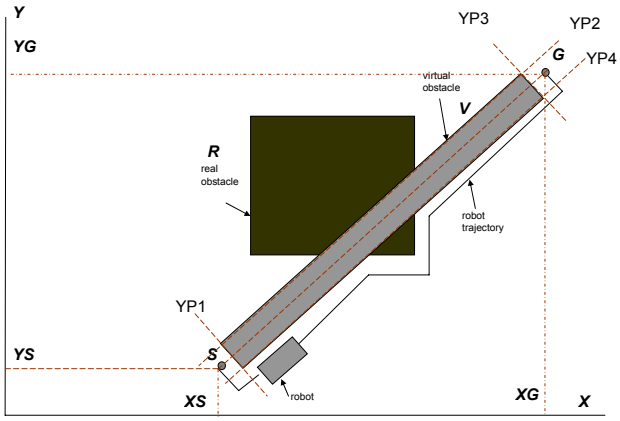


Fig. 7. The virtual obstacle based on the start - the goal line.

D. Experiments Using the Virtual Obstacle

[1] 0.05 *2 means that the virtual obstacle is not only the line but has dimensions which is randomly determined. Equations (6) – (10) define the line which is connected with two determined points.

As the first, the methodology which is exposed in section IV- C is tested using GoldHorn learning system and equations (4) and (5) for situation as it is illustrated in Figure 5. The result is given in Fig. 8.

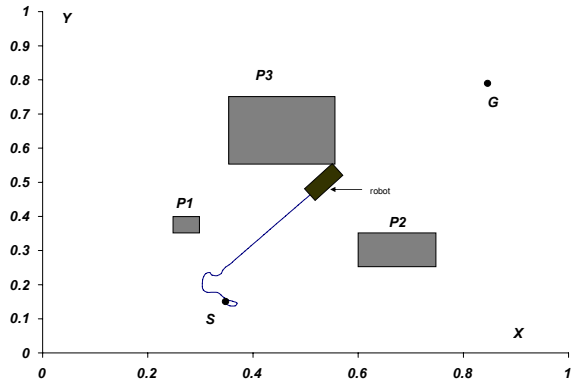


Fig. 8. The virtual obstacle based on the start - the goal line and GoldHorn clone.

The virtual obstacle is treated and according to all of obstacles obviously the touch is detected because reaction time of GoldHorn clone is inappropriate, as it was observed the previously [22].

An application of RBF network in autonomous vehicle motion planning is described in [18], [19]. A relatively small corresponding training set is given in Table I.

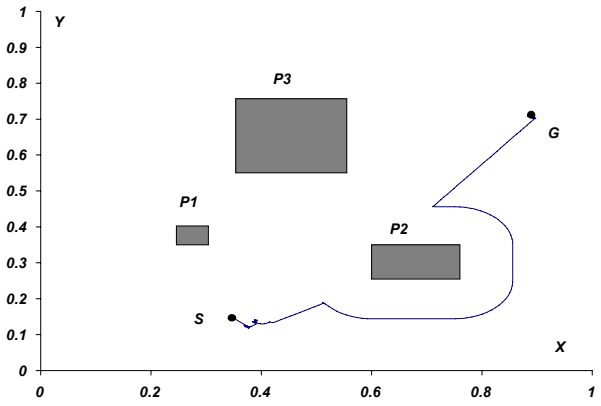
TABLE I
THE ORIGINAL AND THE MODIFIED TRAINING SET

Index J	Distance (IW[j,1]) d_{min}	Angle Ψ (LW[1,j])	Angle modified Ψ_m (LW[j,1])	Factor f_m
1	0.04	10.906	10.906	1
2	0.0494	13.8025	13.8025	1
3	0.0591	11.425	11.425	1
4	0.08	0.0035	81.025	23151
5	0.08	9.513	85.617	9
6	0.081	12.070	12.070	1
7	0.0894	12.692	12.692	1
8	0.0923	12.556	12.556	1
9	0.1006	7.6405	7.6405	1
10	0.12	0.6895	3.4475	5

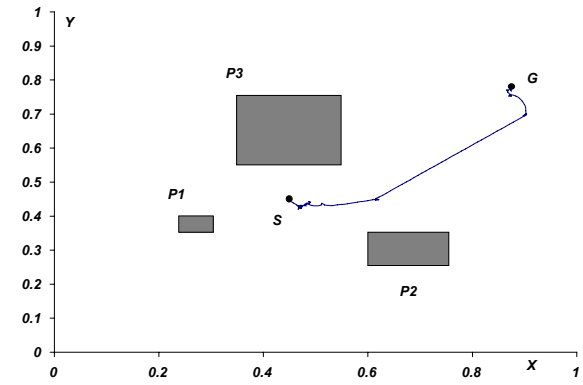
According to the problem of the autonomous vehicle motion planning, E_{xy} is treated as an appropriate criterion. First, spread factor is tuned to σ_j. For step k=1 spread =1, and then spread decreased in 0.01. Process was stopped for spread=0.002. The performance error E_{xy} has minimum for that spread value. While spread factor decreased, total time T of the autonomous vehicle moving out of reach of the obstacles increased:(spread = 0.005, T = 534 s; spread = 0.003, T = 600 s; spread = 0.002, T = 637 s. In order to decrease the performance error for small vehicle distance from obstacles edges ($d_{min} \approx 0.08$), LW[1,j] is multiplied by f_m [j] as it is given in Table 1. For f_m [4] = 23151, f_m[5]=6 and another {f_m [j]=1,j=1,2,3,6,7,8,9,10}, it was T = 1248 s.

But for f_m [j] which have values as it was given in Table 1 performance error was E_{xy} = 0.00382. Scene with five static obstacles is given in Fig. 9c). In that case we have situation when the autonomous vehicle moves away from P1 obstacle to be close, until some critical distance from obstacles P2, P3, P4 and P5, and changes Ψ rapidly safely avoiding the obstacles touch for long time. Changing f_m[j] repeatedly takes a step in the direction of steepest decrease of E_{xy}. But some changes and values of E_{xy} are inappropriate, as it is given in situation with GoldHorn clone. GoldHorn clone based on equation (4), in the situation given in Fig. 8c), touched obstacles (obstacle P2) upon T=475s. Obviously, using set of training examples illustrated in Fig. 4, the GoldHorn performs generalization which is inappropriate for some scenes.

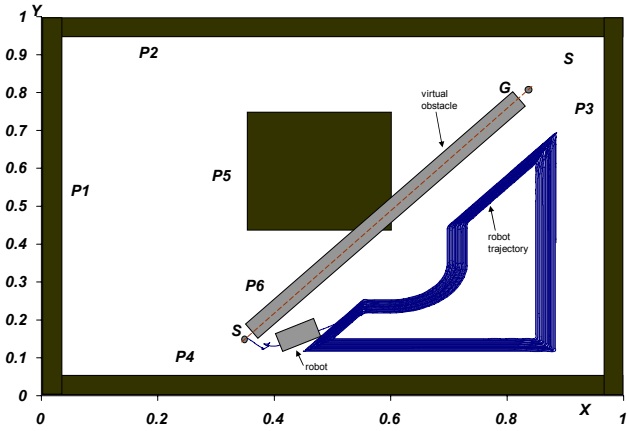
The obtained RBF network clone produced the results that are more appropriate and it could be included in all situations as it is included GoldHorn clone, but by now only according to the obstacle avoiding is used. We interested in the goal reaching as it is given in Section IV-C. A neural network algorithm as an advancement according to the RBF is given in [20]. That algorithm is applied in order to solve the situation which is given in Fig. 9a),b), c). Again, to avoid the vehicle traveling through the narrow passages between obstacles it is needed to define a rule regarding their dimensions [18], [19], [20], [22]. In situation illustrated by Fig. 9a) the vehicle moves from the start position S(0.35,0.15) to the goal position G(0.85, 0.8) avoiding the virtual obstacle also. The experiment is successful. Like that, the experiment is successful in Fig. 9b). Again the virtual obstacle is treated and the vehicle reaction time is quite appropriate. In Fig. 9c) is illustrated situation with five static obstacles P1 – P5, and virtual obstacle P6 which is illustrated also. After a great number of obstacle avoiding circles the vehicle attempts to move to the goal G, but some oscillations appears and the vehicle decides to avoid obstacles and reach the goal again and again, and has not the chance to stop. That process is not time limited, but it means that experiment is quite successful: we have obstacle avoiding and the goal position reaching using the same algorithm, i.e. relation.



a)



b)



c)

Fig. 9. The autonomous vehicle trajectory in xy plane for three and for five static obstacles. The autonomous vehicle is controlled by RBF neural network clone.

E. Conclusion

Attempting to find some elegant and general solution in order to avoid obstacles and in order to reach the goal position the advancement is appeared, Section IV.D. According to the situation which is the previously exposed in [23] and means that it is needed to construct sequence of the special rules to be reached the goal position of the vehicle, the situation which is given here means that it is needed to use the same relations to avoid obstacle and to reach the goal of the vehicle upon the virtual obstacle is defined. The proposed methodology has complexity $O(n^2)$, where n is the number of obstacles.

V. CONCLUSION

The problem of the vehicle motion planning in the environment with fixed obstacles was treated. The task for the given problem was to find a controller. Taking into account the examples which were obtained in training phase, when the vehicle was controlled by an operator to avoid rectangle unmoving obstacle, some acceptable guidance rules were synthesized using different learning algorithms. Their importance was subsequently extended in order to apply them in some situation with the infinite number of different unmoving obstacles. Attempting to find some elegant and general solution in order to avoid obstacles and in order to reach the goal position the advancement is appeared, Section IV.D. According to the situation which is the previously exposed in [23] and means that it is needed to construct sequence of the special rules to be reached the goal position of the vehicle, the situation which is given here means that it is needed to use the same relations to avoid obstacle and to reach the goal of the vehicle upon the virtual obstacle is defined. The proposed methodology has complexity $O(n^2)$, where n is the number of obstacles. Eventually, the vehicle kinematical and then a complete mathematical model is given by nonlinear equations describing a 12 state dynamical system simulated in Matlab or Simulink environment. In both case we manage to find the regulator for the named autonomous vehicle in 2D and 3D- space in situation with infinite number of obstacles, as it is given in [23]. The tendency of the regulator to guide the vehicle in such a fashion that its distance from obstacle edges increases was observed in both case. That fact is a good reason for the conclusion that these results, with sophisticated kinematical and dynamical model of the vehicle, quite confirm the results exposed early, when was used only kinematical model of the vehicle.

VI. REFERENCES

[1] J. C. Latombe, Vehicle Motion Planning, Kluwer Academic Publishers, Boston 1991.

[2] J. Reif, »Compexity of the generalized mover's problem«. Editors J. Schwartz, M. Sharir, and J. Hopcroft: Planning, Geometry, and Compexity of Vehicle Motion, Ablex, Norwood, Nj, 1987.

[3] J. Schwartz, M. Sharir, and J. Hopcroft, Planning, Geometry, and Compexity of Vehicle Motion, Ablex, Norwood, Nj, 1987.

[4] J. Schwartz, M. Sharir, »On the piano mover's problem: I. The case of twodimensional rigid polygonal body moving amidst polygonal barriers«, Editors J. Schwartz, M. Sharir, and J. Hopcroft: Planning, Geometry, and Compexity of Vehicle Motion, Ablex, Norwood, Nj, 1987.

[5] O. Khatib, »Real - time obstacle avoidance for manipulators and autonomous vehicles«, International Journal of Vehicleics Research, 5, 1986.

[6] D. Michie, R. Camacho, "Bilding symbolic representation of intuitive real - time skills from performance data." In: K. Furukawa, S. Muggleton (eds.) Machine Intelligence and Inductive learning, Oxford University Press, Oxford, 1994. C. Sammut, S.Hurst, D. Kedzier, D. Michie, "Learning to Fly", D. Sleeman, P. Edwards, (eds), Proceeding 9st

International Workshop on Machine Learning , Morgan Kaufmann, 1992.

[7] D. Michie, "Knowledge, learning and Machine Intelligence In:L.S. Sterling" (ed), Intelligent System, Plenum Press, New York 1993;

[8] R. Kulic, »Transfer of skill in obstacle avoiding domain«, The 2th IFAC-CIGR Symposium, Bali, Indonesia, August 2001.

[9] R. Kulic, »Behavioural cloning in obstacle avoiding«, The 7th International Symposium SAUM, Vrnjacka Banja, Yugoslavia, September 2001.

[10] R. Kulic, Z. Vukic, »Methodology of concept control synthesis to avoid unmoving and moving obstacles«, Journal of Intelligent and Robotic Systems, Vol. 37, Issue 1, May 2003, Pages: 21 – 41, Year of Publication: 2003,ISSN:0921-0296 ;

[11] A. Karalic, Regression trees learning from noice data, Master thesis, University of Ljubljana, Ljubljana, 1991. (in Slovenian);

[12] V. Krizman, The consideration of noice data using automatic modeling of dynamical systems, Master thesis, University of Ljubljana, Ljubljana, 1993. (in Slovenian).

[13] L. Breiman, J.H. Friedman, R.A. Olshen, C.J. Stone, Classification and Regression Trees, Wadsworth Int. Group, Belmont, California, USA, 1984;

[14] C, A. Micchelli, "Interpolation of scattered data: Distance and conditionally positive definite functions", Constructive Approximation, 2, 11-22, 1986.

[15] M. J.D. Powell, "Radial basis functions for multivariate interpolation: A review, in Algorithms for the Approximation of Functions and Data", J. C. Mason and M. G. Cox, eds. Clarendon Press, Oxford, England, 1987.

[16] R. Stojic, R. Kulic, M. Zivanovic, "Flight control for the given trajectory", Science -Technical observer, Vol. XI, No. 8-9, Belgrade, 1990 (in Serbian).

[17] R. Kulic,Trajectory design for the autonomous vehicle by operator cloning, PhD thesis, Faculty of Electrical Engineering and Computing, Zagreb, 2004. (in Croatian);

[18] R. Kulic, Z. Vukic, »Autonomous Vehicle Motion Planning by Behavioral Cloning«, Southeastern Europe, USA, Japan and European Community Workshop on Research and Education in Contrlol and Signal Processing REDISCOVER 2004, June 14-16, 2004, Cavtat, Croatia.

[19] R. Kulic, Z. Vukic, "Behavioral Cloning in the Autonomous Underwater Vehicle Path Generation", EDPE2005 Conference, Dubrovnik, 2005, Croatia.

[20] R. Kulic, Z. Vukic, „Autonomous underwater vehicle motion planning by behavioral cloning and Kohonen modified rule", Submitted to IEEE SMC, January 2006.

[21] R. Kulic, Z. Vukic, Methodology of concept control synthesis to avoid unmoving and moving obstacles (II), *(accepted, April 11, 2006; Online published June 14,2006)* Journal of Intelligent and Robotic Systems, Publisher Springer Netherlands, ISSN 0921-0296 (paper) 1573-0409 (Online), DOI: 10.1007/s 10846-006-9035-7, Issue: Volume 45, Number 3, Date:March 2006, Pages: 267 - 294.

[22] Kulic R., Vukic Z., Autonomous Vehicle Obstacle Avoiding And Goal Position Reaching By Behavioral Cloning, The 32nd Annual Conference of the IEEE Industrial Electronics Society (IECON-2006), November 7-10, 2006; Paris (France) .

[23] Kulic R., Vukic Z., Behavioral cloning and obstacle avoiding for two different autonomous vehicles, International EPE-PEMC 2006 Conference, 30 August – 01 September 2006, Portoroz, Slovenia.

[24] D. S. Broomhead, D. Lowe, Multivariate functional interpolation and adaptive networks, Complex Systems, 2, 321-355, 1988.

[25] M. Niranjan , F. Fallside, Neural Networks and Radial Basis Functions in Classifying static Speech Patterns , Technical Report CUEDIF –INFENG17R22, Engineering Department, Cambridge University, 1988.