Vision-based Autonomous Navigation Using Neural Networks and Templates in Urban Environments

Jefferson R. Souza, Gustavo Pessin, Patrick Y. Shinzato, Fernando S. Osório and Denis F. Wolf
Mobile Robotics Laboratory, University of São Paulo (USP)
Av. Trabalhador São-Carlense, 400 - 13.560-970 - São Carlos, SP, Brazil
Email: {jrsouza, pessin, shinzato, fosorio, denis}@icmc.usp.br
http://www.lrm.icmc.usp.br/

***Abstract*—The aim of this work is to develop a vehicle control system capable of learn behaviors based on examples obtained from human drivers and analyze different levels of memory of the templates (LMT). Our approach is based on image processing, template matching, finite state machine, and template memory. The proposed system allows to train an image segmentation and neural networks which works with LMT in order to identify navigable and non-navigable regions generating as output the steering control and speed for an Electric Autonomous Vehicle, that should stay replicating or improving the human behavior. Several experimental tests have been carried out under different environmental conditions to evaluate the proposed techniques.**

I. Introduction

Making autonomous vehicles to operate in non-structured environments is a very challenging research topic in mobile robotics. The research with autonomous vehicles provides safer conditions in road traffic, both for individual or collective use. It could increase efficiency in freight and provide some degree of independence to people unable to drive.

Several works have been improving on navigation in outdoor environments. Competitions like DARPA Challenges [2] and ELROB [3] have been pushing the state of the art in autonomous vehicle control.

The most relevant results obtained in such competitions combine information obtained from a large number of complex sensors. Some approaches use five (or more) laser range finders, video cameras, radar, differential GPS, and inertial measurement units [2], [6]. Although there are several interesting applications for such technology, the cost of such systems is very high, which is certainly prohibitive to commercial applications.

In this paper we propose a vision-based navigation approach for urban environments, based on a low/medium cost platform. Our system uses a single camera to acquire data from the environment, to detect the navigable regions (roads), estimates the best trapezium on an image, acquires and trains different levels of memory of the templates that should be done in order to keep the vehicle in a safe path, and finally, steering and accelerating the vehicle.

Fig. 1 presents our Electric Autonomous Vehicle (EAV) test platform. The images are acquired and then processed using an Artificial Neural Network (ANN) that identifies the road ahead of the vehicle.

Fig. 1. Test platform used in the experiments.

We use two neural networks. The first one identifies navigability regions in which a template-based algorithm classifies the image and identifies the action that should be taken by the EAV. After that, a Finite State Machine (FSM) is used to filter some input noise and reduce classification and/or control errors. In this paper noise is considered as variations in the road color, such as: dirt road (mud or dust), shadows, and depressions. So, after obtaining the current state (template), which is the input of a new neural network that works with levels of memory of the templates. This ANN aims to learn the driver's behavior, providing smoother steering and levels of speed in the same way as the driver. In this paper we analyze six levels of template memory on the ANN searching to obtain the topology which provides the more reliable ANN. This second neural networks, as it learn the driver behavior, may provide a way to make this technology more attractive to people in general.

This paper is organized as follows. Section 2 presents the related works. Section 3 describes the proposed method. Section 4 shows the experimental results and discussion. Finally, Section 5 presents the conclusion and future works.

II. Related Works

Autonomous Land Vehicle in a Neural Network (ALVINN) [7] is an ANN based navigation system, that calculated a steer angle to keep an autonomous vehicle inside the road limits. In this work, the gray-scale levels of a 30 x 32 image were used as the input of a neural networks. In order to improve training, the original road image and steering were generated, allowing ALVINN to quickly learn how to navigate in new roads. A disadvantage of this work is the high computational time. The architecture has 960 input units fully connected to the hidden layer to 4 units, also fully connected to 30 units in output layer. This ANN topology requires larger computational use. Regarding that issue, this problem requires real time decisions therefore these topology are not efficient.

Later, the EUREKA project Prometheus [5] for road-following was successfully performed, which provided trucks with an automatic driving system to reproduce drivers in repetitious long driving situations. In this project, the developed system also included a function to warn the driver in dangerous situations. A limitation of this project was an excessive number of heuristics created by the authors to limit the false alarms caused by shadows or discontinuities in the color of the road surface.

Another interesting work developing an autonomous vehicle control system, by Shihavuddin et al. [8], approaches the path map generation of an unknown environment using a proposed Trapezoidal Approximation (TA) of road boundary. At first, a blind map of the unknown environment is generated in computer, then the image of the unknown environment is captured by the vehicle and sent to the computer using a radio frequency transmitter module. After that, the image is preprocessed and the road boundaries are detected using the TA. So, the vehicle operates independently avoiding all obstacles, and the issue with this approach is the dependency of the camera tilt angle, because the vehicle moves through the trapezium and reaches the next approximated trapezium having a previously tilt angle.

A more recent work, from Stein and Santos [12], proposes a method to compute the steering of an autonomous robot, moving in a road-like environment. The proposed system used artificial neural networks (ANNs) to learn behaviors based on examples from human drivers, replicating and sometimes even improving human-like behaviors. To validate the created ANNs, real tests were performed and the robot successfully completed several laps of the test circuit showing good capacities for recovery and for generalization with relatively small training data sets. One of the issues in this work is the impossibility of validating a network training without actually testing it with the real robot.

III. Proposed Method

In this work, a vision-based navigation system is proposed (Fig. 2). Our approach is composed by 4 steps. In the first step an image is obtained and the road is identified using ANNs classification. In the second step, a template matching algorithm is used to identify the geometry of the road ahead of the robot. In the third step, a FSM is used to filter noisy inputs and any classification error. Finally, a template memory is used in order to define the action that the robot should take to keep on road. These steps will be described in the next sub-sections.

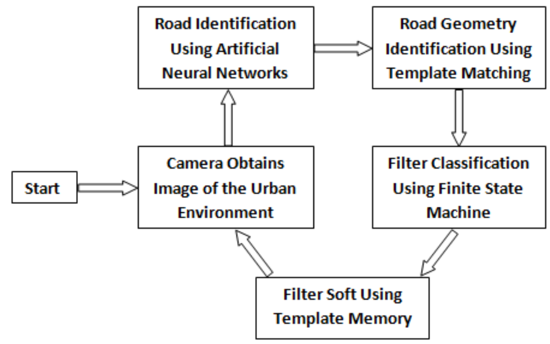


Fig. 2. The proposed method.

A. Image Processing Step

In the work [9], the performance of ANNs was evaluated when applied into a road identification task. Based on the results obtained, a system composed by six Multilayer Perceptron (MLP) ANNs was proposed to identify the navigable regions in urban environments (Fig. 3 (a)). The result of this ANNs output combination is a navigability map (as shown on Fig. 3 (b)). The image processing step divides the image into blocks of pixels and evaluates them as single units.

Each ANN is used to classify the block considering their attributes (output 0 to non-navigable and output 1 to navigable). In this work, each ANN contains an input layer with the neurons according to the image input features (see Table I), one hidden layer with five neurons, and the output layer which has only one neuron (binary classification). However, after the training step, the ANN return real values between 0 and 1, as outputs. These real values can be interpreted as the classification certainty degree of one specific block. The main difference between the six ANNs is the set of image attributes used as input for each one. All these sets of attributes (see Table I) are calculated during the block-segmentation of the image. The choice of these attributes was based on the results presented in the work [9].

After obtaining the six outputs of the ANNs referring to each block, the classifier calculates the average of these values to compose a single final output value. These values representing each block obtained from the original image form together the navigability map matrix. This matrix is used to locate the most likely navigable region. It is important to

ANNs	Input attributes
ANN1	U av, V av, B norm av, H ent, G norm en and H av
ANN2	V av, H ent, G norm en, G av, U av, R av, H av, B norm av, G norm av and Y ent
ANN3	U av, B norm av, V av, B var, S av, H av, G norm av and G norm ent
ANN4	U av, V av, B norm av, H ent, G norm en and H av
ANN5	V av, H ent, G norm en, G av, U av, R av, H av, B norm av, G norm av and Y ent
ANN6	U av, B norm av, V av, B var, S av, H av, G norm av and G norm ent

TABLE I
INPUT ATTRIBUTES OF THE ANNS (AVERAGE = AV, NORMALIZED = NORM, ENTROPY = ENT, ENERGY = EN AND VARIANCE = VAR).

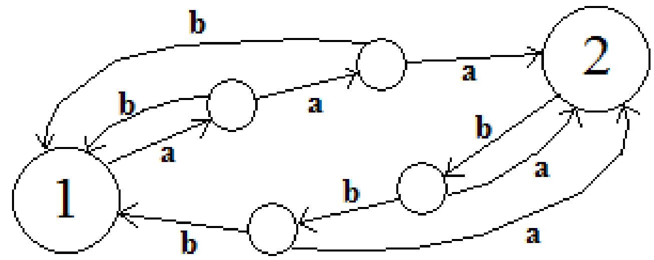


Fig. 4. Transition between 2 states with 2 intermediate states.

mention that the ANN is previously trained using supervised examples of navigable and non-navigable regions selected by the user one time on an initial image. After that, the trained ANN is integrated into the vehicle control system and used as the main source of information to the autonomous navigation control system.

B. Template Matching Step

After obtaining the ANN classification, 7 different road templates are placed over the image in order to identify the road geometry. One of them identifies a straight road ahead, two identify a straight road in the sideways, two identify soft turns, and two identify hard turns (e.g. a straight road ahead as shown on Fig. 3 (c)). Each template is composed by a mask of 1s and 0s [11]. The value of each mask is multiplied by the correspondent value into the navigability matrix (values obtained from the ANN classification of the correspondent blocks of the image). The total score for each template is the sum of products. The template that obtains the higher score is selected as the best match of the road geometry. In this work, only one template can obtain a high score, because we use probabilities as the decision criteria.

C. Finite State Machine Step

The developed FSM uses the result of the template matching step as input, which carries out a classification for the road detected in each captured frame. This classification is defined by the template which best fits the matrix and its position. The developed FSM is composed by 5 states (straight road, soft turns left and right, and hard turns left and right) as shown on Fig. 4. Fig. 4 represents a state change of ’a’ to ’b’. For example, ’a’ represents a straight road state and ’b’ soft turn left. To change the state in the FSM there must happen three consecutive equal states. In this work, we use the FSM with only 2 intermediate transitions between the states and have produced reasonable results. Detailed information can be seen in [11].

D. Template Memory Step

After obtaining the current state (e.g. template) by FSM, this current template is used as input in the template memory step. In this step, the levels of memory of the templates are stored in a queue, as shown on Fig. 5. In this work, the $Template_t$ represents the current template, $Template_{t-1}$ the previous template, $Template_{t-2}$ one template before the previous. This is done successively, until the number of template memory (NTM) is reached, where t represents the time.

$$Template_t, Template_{t-1}, Template_{t-2}, \ldots, Template_{t-NTM}$$

Fig. 5. Representation of the template memory step.

In this step, the basic network structure used is a feed-forward MLP, the activation function of the hidden neurons is the sigmoid function and the ANN learning is the resilient backpropagation (RPROP). The inputs are represented by templates memory and the outputs are the steer angle and speed, as shown on Fig. 6.

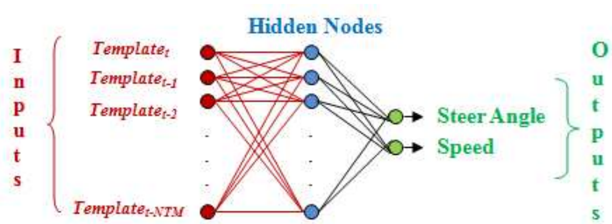
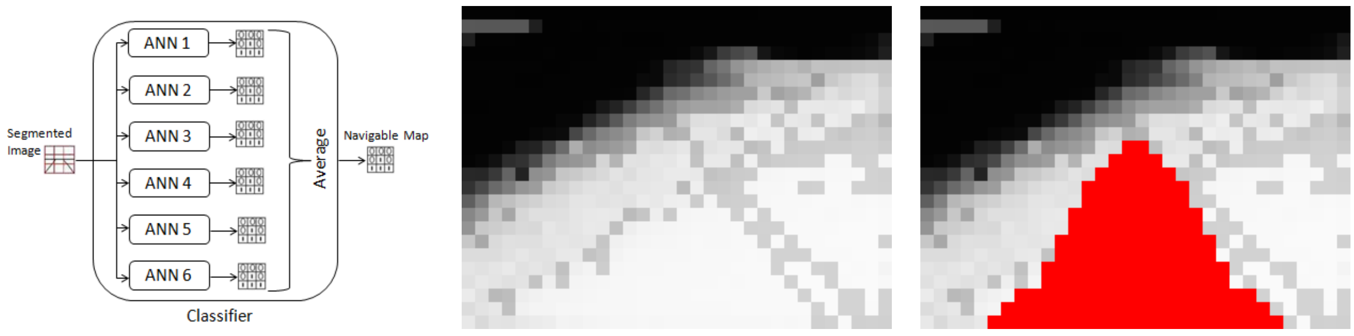


Fig. 6. The structure of the ANN.

IV. EXPERIMENTAL RESULTS

The experiments were performed using the EAV shown on Fig. 1, an electric vehicle capable of autonomous navigation in a urban road, equipped with a VIDERE DSG video camera, a ROBOTEQ AX2580 motor controller for steering control, an ARDUINO DUEMILANOVE is a microcontroller board based on the ATmega168 used for vehicle speed and a GARMIN 18X-5Hz GPS. The GPS was used only to register the log of the vehicle (robot) trajectory (Fig. 7), where the log data (it was not used as input of the system) were entered in Google Earth 6 to acquire the 3D image. The image acquisition resolution was set to (320 x 240) pixels. The ANNs of the image processing step were executed using Fast Artificial Neural Network (FANN) [4], which is a free



(a) (b) (c)

Fig. 3. Classifier structure (a), image processing step (b) and template matching step (c).

open source neural network library which implements MLP ANNs in C, and the ANN in the template memory step was used Stuttgart Neural Network Simulator (SNNS) [10], which is a software simulator for neural networks and creates an efficient and flexible simulation environment for research on application of ANNs (simulator kernel written in C). For the development of the image acquisition, image processing and template matching algorithms, we used the OpenCV [1] library that has been widely used in computer vision algorithms. Finally, the template memory step and other complementary developments were also made in the C Language.

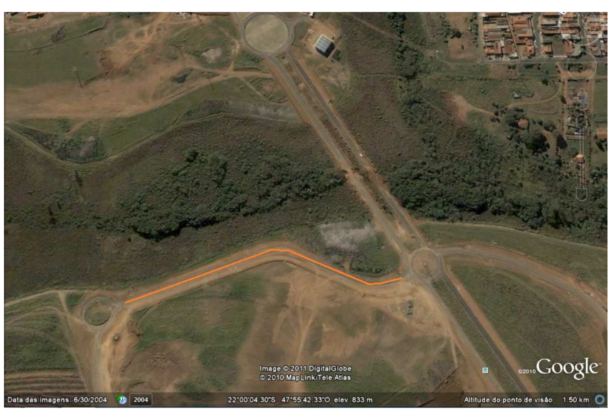


Fig. 7. GPS shows the path that the EAV used to learn (training data).

Table II shows the obtained values of the performed path by electric vehicle for supervised learning in order of the proposed method to replicate human behavior and analyze different levels of memory of the templates.

In Table II, we analyze six Levels of Memory of the Templates (LMT), which represent the architecture of the second neural network used in our proposed system. The numbers of column LMT represents the numbers that we tested randomly in order to develop a well-defined architecture which is one of the objectives of the paper. Half, Double and Equal shows the different architectures tested in this work, for example, LMT = 3 changes occur in the number of neurons in the intermediate layer of an MLP basic architecture RPROP (a learning heuristic for supervised learning in feedforward ANNs), where were tested the architectures: 3-**1**-2 (**Half**), 3-**3**-2 (**Equal**) and 3-**6**-2 (**Double**), obtaining the values of mean squared error (MSE), cycle (optimal point of generalization (OPG), i.e., minimum training error and capacity of maximum generalization) and the number of better neural network.

LMT	Half			Equal			Double		
	M	C	A	M	C	A	M	C	A
3	**45.599**	25	4	45.837	40	1	45.847	55	3
5	51.441	45	3	51.404	25	2	**51.362**	25	5
8	**43.601**	15	1	45.067	60	2	44.381	15	1
10	**48.969**	15	3	51.334	55	4	49.431	10	5
15	43.481	60	5	42.941	15	5	**42.233**	20	4
20	**52.133**	20	4	52.253	45	1	52.268	50	2

TABLE II
THE LOWEST VALUES OF MSE, CYCLE AND ANN FOR THE SIX LEVELS OF TEMPLATE MEMORY (MSE = M, CYCLE = C AND ANN = A).

After the analysis of data on Table II, the architecture that best learned from data presented in the input layer was 15-30-2, as it showed the lowest MSE for the cycle 20.

Table III shows the initial values of MSE for all LMT, in order to analyze the reduction of errors on the time, based on Table II shows the MSE lowest values. Thus, analyzing which ANNs best learned, reduced the MSE until the OPG. After that, the most arquitectures Half (3-1-2, 5-2-2, 8-4-2, 10-5-2 and 15-7-2) showed the higher initial values of the MSE.

	MSE		
LMT	Half	Equal	Double
3	**1060.730**	929.475	762.547
5	**1004.350**	825.228	606.680
8	**858.441**	662.772	463.422
10	**847.994**	622.916	472.747
15	**714.645**	467.323	693.573
20	603.171	463.863	**1385.200**

TABLE III
THE INITIAL VALUES OF MSE FOR ALL THE LEVELS OF MEMORY OF THE TEMPLATES.

Table IV shows the percentage of the MSE for all architectures of the second ANN proposed in this paper based on Tables II and III. The best architecture (15-30-2) is shown on Table IV, presenting the smallest MSE (approximately error

5.9%) on the set of architectures (15-7-2, 15-15-2 and 15-30-2). The lowest MSE of the Table IV is defined as being the architecture 20-10-2 (approximately error 3.7%), but after analyzing the data on Table II this architecture had the largest MSE (52.133) compared to other architectures. Therefore, the architecture 15-30-2 is the best in this work.

	MSE		
LMT	Half	Equal	Double
3	**4.298%**	4.321%	4.322%
5	5.121%	5.118%	**5.114%**
8	**5.079%**	5.250%	5.170%
10	**5.774%**	6.053%	5.829%
15	6.084%	6.008%	**5.909%**
20	**3.763%**	3.772%	3.773%

TABLE IV
THE PERCENTAGE OF MSE FOR ALL THE LEVELS OF MEMORY OF THE TEMPLATES.

Fig. 8 and 9 illustrate the steer angle and speed of EAV using the architecture 15-30-2, showing the human driver values and the obtained by learning of the ANN. Small oscillations are present in the data learned by ANN, since the FSM used in the proposed system maintains the current state to 2 intermediate transitions, resulting in a linearity in the data (the vehicle keeps on the road with steer and constants speed).

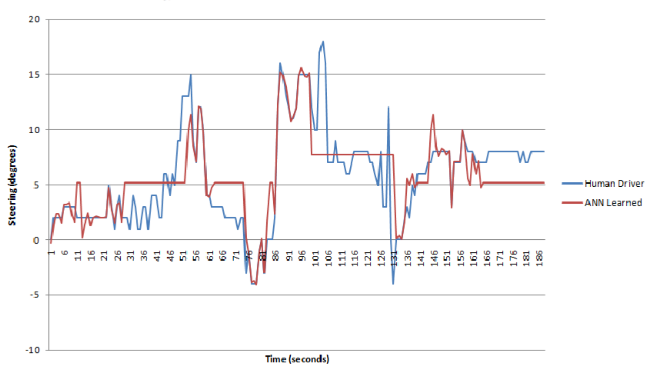


Fig. 8. Steering wheel of the EAV using the training data.

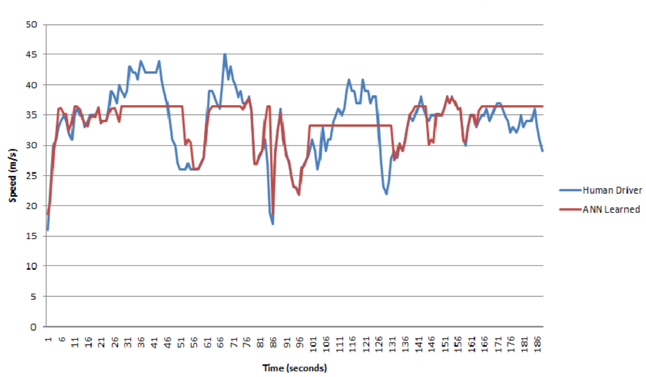


Fig. 9. Speed of the EAV using the training data.

Fig. 10 and 11 illustrate two tests performed by EAV obtained by the second ANN used in our proposed system. The architecture 15-30-2 was used in our experiments, showing the steer angle and speed for the test data.

Fig. 10 illustrates the steering performance of the ANN trained with the RPROP algorithm. It shows a better performance, with little oscillation as shown in Test 1 (the degree of steering between -5 and 15 as learned by ANN). The main difference when compared with Test 2 is small oscillation during the straight line used by EAV and it is quite similiar to what has been learned by the ANN. Unlike Test 2, the degree of steering is more than 15 in some periods on the time.

The speed performance of the RPROP case is shown on Fig. 11 for the test data. In this case, the speed for Test 1 was obtained by ANN, showing a better performance when compared to Test 2, because it showed a better response to the learning of ANN and also generalized the data to navigate the vehicle on a safe path on the road unlike Test 2. However, it stopped momentarily because of decision making performed by ANN in real time.

Experimental tests showed that the Test 1 generated as ANN output a better peformance in the robot behavior, as shown on Fig. 10 and 11 (see Experiment Video [1]).

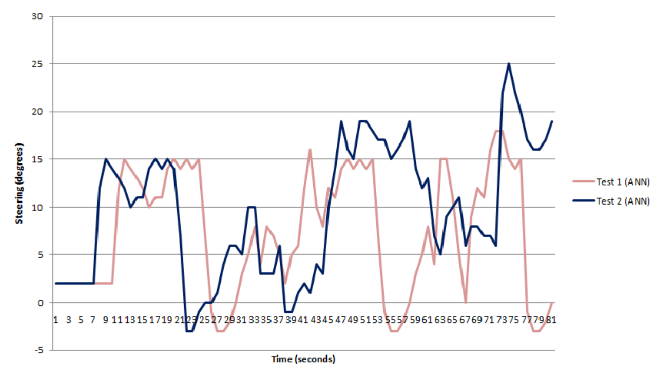


Fig. 10. Steering wheel of the EAV using the test data.

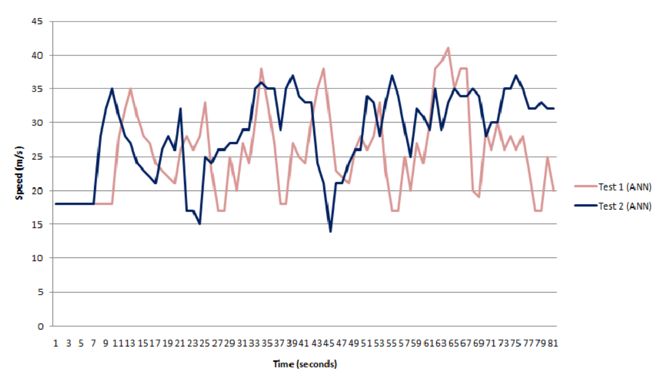


Fig. 11. Speed of the EAV using the test data.

[1]Experiment video available in the Internet:
Test 1 - http://www.4shared.com/video/rq9sGqnb/

V. Conclusion and Future Works

Autonomous vehicle navigation is a very important task in mobile robotics. This paper presented a vision-based navigation system which can be trained to identify the road and navigable regions using ANNs, template matching classification and a template memory algorithm. Our approach was evaluated using an Electrical Vehicle tested in outdoor road following experiments. The vehicle was able to navigate autonomously in this urban environment in straight line, since one of our goals is to find the best architecture of the ANN to be applied in different urban environments. Our quantitative analysis also obtained good results for the learning of ANNs with the respective architectures (levels of memory of the templates).

As future work, we plan to evaluate other classification methods and decision making algorithms. We also planning to held in other urban environments with some accented curves to analyze system behavior in such situations, and to integrate camera and LIDAR laser information in order to better deal with obstacles, bumps and depressions in the road.

Acknowledgment

The authors acknowledge the support granted by CAPES, CNPq and FAPESP to the INCT-SEC (National Institute of Science and Technology - Critical Embedded Systems - Brazil), and FAPESP doctoral grant (process 2009/11614-4).

References

[1] G. Bradski and A. Kaehler. Learning OpenCV: Computer Vision with the OpenCV Library. O'Reilly, Cambridge, MA, 2008.

[2] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. Bradski. Self-Supervised Monocular Road Detection in Desert Terrain. In Robotics: Science and Systems (RSS), 2006.

[3] European Land-Robot, http://www.elrob.org/, Access on 05 january, 2011.

[4] Fast Artificial Neural Network Library, http://leenissen.dk/fann/, Access on 02 june, 2010.

[5] V. Graefe. Vision for Intelligent Road Vehicles. In Proceedings of the IEEE Symposium of Intelligent Vehicles, 135-140, 1993.

[6] A. Petrovskaya and S. Thrun. Model Based Vehicle Tracking in Urban Environments. IEEE International Conference on Robotics and Automation, Workshop on Safe Navigation, vol. 1, pp. 1-8, 2009.

[7] D. A. Pomerlau. ALVINN: An Autonomous Land Vehicle In a Neural Network. Advances In Neural Information Processing Systems, 1989.

[8] A. S. M. Shihavuddin, K. Ahmed, M. S. Munir, and K. R. Ahmed. Road Boundary Detection by a Remote Vehicle Using Radon Transform for Path Map Generation of an Unknown Area. International Journal of Computer Science and Network Security (IJCSNS), vol. 8, n. 8, pp. 64-69, 2008.

[9] P. Y. Shinzato and D. F. Wolf. A Road Following Approach Using Articial Neural Networks Combinations. Journal of Intelligent and Robotic Systems, 2010.

[10] Stuttgart Neural Network Simulator, http://www.ra.cs.uni-tuebingen.de/SNNS/, Access on 20 november, 2010.

[11] J. R. Souza, D. O. Sales, P. Y. Shinzato, F. S. Osório, and D. F. Wolf. Template-Based Autonomous Navigation in Urban Environments. 26th ACM Symposium on Appied Computing, 2011.

[12] P. S. Stein and V. Santos. Visual Guidance of an Autonomous Robot Using Machine Learning. 7th IFAC Symposium on Intelligent Autonomous Vehicles, 2010.