Vision-Based Autonomous Navigation Using Supervised Learning Techniques

Jefferson Souza, Gustavo Pessin, Fernando Osório, Denis Wolf

Vision-based Autonomous Navigation Using Supervised Learning Techniques

Jefferson R. Souza, Gustavo Pessin, 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
{jrsouza,pessin,fosorio,denis}@icmc.usp.br

Abstract. This paper presents a mobile control system capable of learn behaviors based on human examples. Our approach is based on image processing, template matching, finite state machine, and template memory. The system proposed allows image segmentation using neural networks in order to identify navigable and non-navigable regions. It also uses supervised learning techniques which work with different levels of memory of the templates. As output our system is capable controlling speed and steering for autonomous mobile robot navigation. Experimental tests have been carried out to evaluate the learning techniques.

Keywords: Robotic Vehicles Navigation, Trapezoidal Algorithm, Finite State Machine, Supervised Learning Techniques

1 Introduction

Human driver errors are a major cause of accidents on roads. Frequently people get injuried or even die due to road traffic accidents (RTA). Also, bad road and weather conditions increase the risk of RTA. Autonomous vehicles could provide safer conditions in roads for individual or collective use. They also could increase efficiency in freight and provide some degree of independence to people unable to drive.

Several works in the literature have been focusing on navigation in outdoor environments. Competitions like DARPA Challenges [4] and ELROB [5] have been pushing the state of the art in autonomous vehicle control. 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 [4], [11]. Although there are several interesting applications for such technology, the cost of such systems is very high, which is prohibitive to commercial applications.

In this paper we propose a vision-based navigation approach based on a low cost platform. Our system uses a single camera to acquire data from the environment. It detects 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 robot in a safe path, and finally, control steering and accelerating of the robot.

Fig. 1 shows our test platform. The images are acquired and processed using an Artificial Neural Network (ANN) that identifies the road ahead of the robot.

Fig. 1. Pioneer 3-AT (P3-AT) test platform used in the experiments.

We use two ANNs. The first one identifies navigable regions in which a template-based algorithm classifies the image and identifies the action that should be taken by P3-AT. 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 ANN 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. We analyze six levels of template memory on the ANN searching to obtain the topology which provides the more reliable ANN. Also, we analyze many supervised ML algorithms to compare with this ANN in order to find the best among them.

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.

2 Related Works

Autonomous Land Vehicle in a Neural Network (ALVINN) [12] is an ANN based navigation system that calculates a steer angle to keep an autonomous vehicle in the road limits. In this work, the gray-scale levels of a 30 x 32 image were used as the input of an ANN. In order to improve training, the original road image and steering were generated, allowing ALVINN to learn how to navigate in new roads. The disadvantages of this work are the low resolution of a 30 x 32 image (gray-scale levels) and 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. Regarding that issue, this problem requires real time decisions therefore this topology is not efficient.

Later, the EUREKA project Prometheus [7] for road-following was successfully performed, which provided trucks with an automatic driving system to reproduce drivers in repetitious long driving situations. The 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.

Chan et al. [3] presents an Intelligent Speed Adaptation and Steering Control (ISASC) that allows the vehicle to anticipate and negotiate curves safely. This system uses Generic Self-Organizing Fuzzy Neural Network (GenSoFNN-Yager) which include the Yager inference scheme [10]. GenSoFNN-Yager has as main feature their ability to induce from low-level perceptual information in form of fuzzy IF-THEN rules. Results show the robustness of the system in learning from example human driving negotiating new unseen roads. The autonomous driver demonstrate that anticipation is not always sufficient yet, also large variations in the distribution of the rule were observed which imply a high complexity of the system, beyond the system be tested on a driving simulator.

The work [8] focus on the task of lane following, where a robot-car learns anticipatory driving from a human and visual sensory data. During the learning step the robot associates visual information with human actions. This information is derived from the street lane boundary that is detected in each image in real-time (based in [2]). In this work two modules were used, a reactive controller (RC) and a planner, which the former maps short-term information to a single steering control value, and the latter generates action plans, i.e. sequences for steering and speed control. The final steering command is a combination of planner and RC output. The advantages of this approach are react to upcoming events, cope with short lacks of sensory information, and use these plans for making predictions about its own state, which is useful for higher-level planning. Despite many advantages, due to the inertia of the robot it is less visible than what could be expected from the plotted signal. Also the system is not able to predict future states.

A more recent work, Markelic et al. [9], proposes a system that learns driving skills based on a human teacher. Driving School (DRIVSCO) is implemented as a multi-threaded, parallel CPU/GPU architecture in a real car and trained with real driving data to generate steering and acceleration control for road following. Besides, it uses an algorithm for detecting independently moving objects (IMOs) for spotting obstacles with stereo camera. A predicted action sequence is compared to the driver actions and a warning is issued if they are differing too much (assistance system). The IMO detection algorithm is more general in the sense that it will respond not only to cars, but to any sufficiently large (11 x 11 pixels) moving object. The steering prediction is very close to the human signal, but the acceleration is less reliable.

3 Proposed Method

Our approach (Fig. 2) 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, 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.

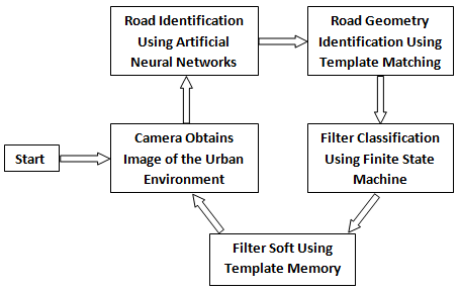

Fig. 2. The proposed method.

3.1 Image Processing Step

We adopted the proposed method of Shinzato [13], which proposes to use ANNs to be applied into a road identification task. Based on the results, a system composed by six Multilayer Perceptron (MLP) ANNs was proposed to identify the navigable regions in outdoor environments (Fig. 3 (a)). The real image of the environment can be seen on Fig. 3 (b). The result of this ANNs output combination is a navigability map (as shown on Fig. 3 (c)). The image processing step divides the image into blocks of pixels and evaluates them as single units.

The ANNs are used to classify the blocks considering their attributes (output 0 to non-navigable and 1 to navigable). Each ANN contains an input layer with the neurons according to the image input features (see Table 1), one hidden layer with five neurons, and the output layer which has only one neuron (binary classification). However, after the training step, the ANN returns real values between 0 and 1, as outputs. These real values can be interpreted as the classification certainty degree of one specific block. The difference between the six ANNs is the set of image attributes used as input for each one. All these sets of attributes (see Table 1) are calculated during the block-segmentation of the image. The choice of these attributes was based on the results presented in the work [13].

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 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 frame. 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.

Table 1. Input attributes of the ANNs (average = av, normalized = norm, entropy = ent, energy = en and variance = var).

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

3.2 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 Fig. 3 (d)). Each template is composed by a mask of 1s and 0s [15]. 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. Only one template can obtain a high score, because we use probabilities as the decision criteria.

3.3 Finite State Machine Step

The 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). 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 [15].

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

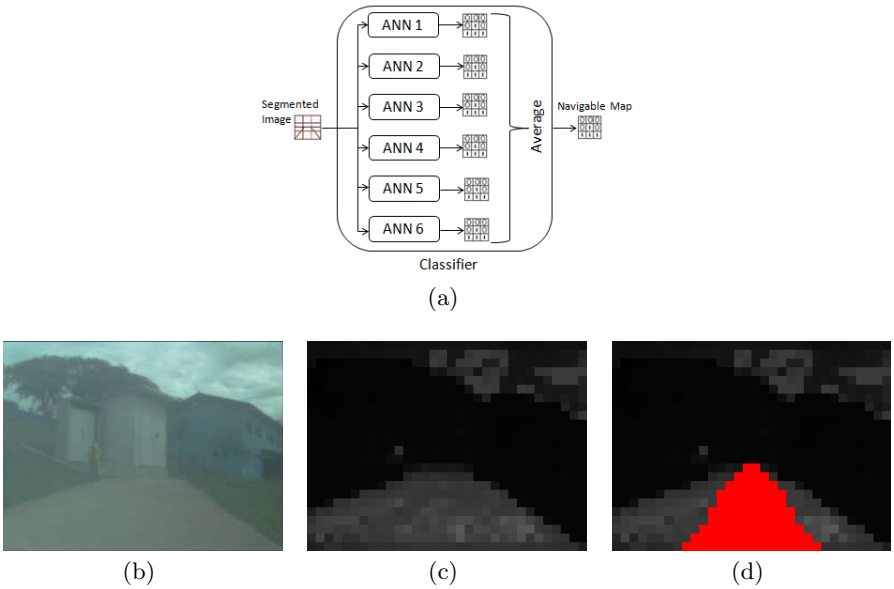

Fig. 3. Classifier structure (a), image real step (b), image processing step (c) and template matching (d).

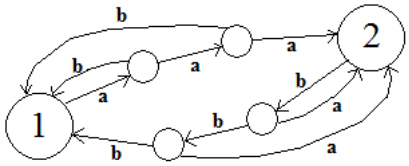

Fig. 4. Transition between 2 states with 2 intermediate states.

In this step, an ANN second is used differently of the ANNs used in the image processing step. The basic network structure (Fig. 5) 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. We compare the result of best ANN topology with others ML in order to find the best among them. Then, apply the best algorithm on the P3-AT robot to autonomous navigation in real-time.

4 Experimental Results

The experiments were performed using the P3-AT shown on Fig. 1. It was equipped with a VIDERE DSG video camera. The GPS was used only to visu-

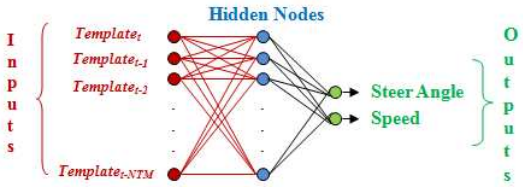

Fig. 5. The structure of the ANN.

alize the robot trajectory (Fig. 6). 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) [6], the ANN in the template memory step used Stuttgart Neural Network Simulator (SNNS) [14], and the supervised learning techniques used Waikato Environment for Knowledge Analysis (WEKA) [16]. For the development of the image acquisition, image processing and template matching algorithms, we used the OpenCV [1] library.

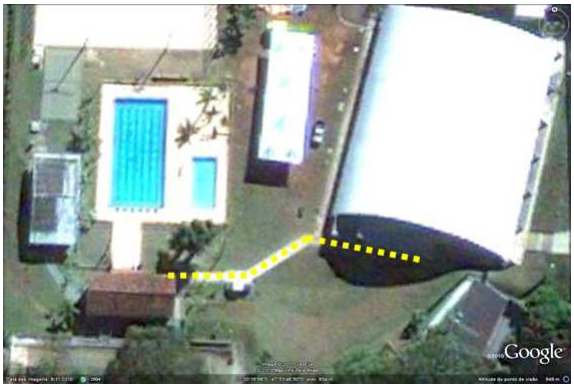

Fig. 6. GPS presents the performed path by P3-AT robot.

In Table 2, we analyze six Levels of Memory of the Templates (LMT), which represent the architecture of the second ANN used in our proposed system. The ANN topology represents the numbers that we tested randomly in order to develop a well-defined architecture which is one of the goals 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 a Rprop MLP, where tested the architectures: 3-1-2 (Half), 3-3-2 (Equal) and 3-6-2 (Double), obtaining the cycle of the optimal point of generalization (OPG) and the values of mean squared error (MSE) for the validation set.

Fig. 7 shows the dispersion of the validation set on the 3D plan. We can observe that this data set is simple, but contains data very close. s1, s2, s3, s4

Table 2. Results of ANN validation using different hidden layers.

	Mean Squared Error (MSE) (10^{-3})				
ANN Topology	Test 1	Test 2	Test 3	Test 4	Test 5
1x0x2	7.79376	7.79377	7.79376	7.79376	7.79376
1x1x2	6.58673	6.58674	6.58673	6.58672	6.58664
1x2x2	0.00249	0.00271	0.00259	0.00235	0.00254
3x1x2	6.54093	6.54091	6.54093	6.54094	6.54093
3x3x2	**0.00006**	0.00195	0.00011	0.00573	0.00573
3x6x2	0.00047	0.00015	0.00637	0.05327	0.00189
5x2x2	0.00648	0.00619	0.00794	0.00679	0.00614
5x5x2	0.00076	0.00043	0.00008	0.00036	0.00769
5x10x2	0.00053	0.00243	0.01529	0.00132	0.00122
8x4x2	0.07995	0.09705	0.05762	0.00039	0.02225
8x8x2	0.02109	0.05366	0.04026	0.01900	0.15480
8x16x2	0.25874	0.06981	0.01869	0.12832	0.00054
10x5x2	0.03305	0.02345	0.03621	0.01560	0.01411
10x10x2	0.01385	0.01721	0.10243	0.06503	0.00952
10x20x2	0.07953	0.02662	0.00606	0.01197	0.00243
15x7x2	0.02513	0.00116	0.00884	0.00151	0.02031
15x15x2	0.00519	0.01556	0.01363	0.01643	0.06065
15x30x2	0.22430	0.01377	0.13015	0.00816	0.05283

and s5 are the Rprop ANN outputs on the best ANN topology (3-3-2) from Table 2 (Test 1). These outputs represent the steering and speed control system of the P3-AT robot, for example, s1 (steering = 0.0000 and speed = 0.4000), s2 (steering = -0.0435 and speed = 0.2000), s3 (steering = 0.0435 and speed = 0.2000), s4 (steering = -0.0870 and speed = 0.1000) and s5 (steering = 0.0870 and speed = 0.1000). These values were obtained by the training set when the robot made an autonomously collect using only the assistance of the templates. We did not use the data collection from the human, but we obtained a response of the templates as shown Fig. 3 (c), to really make a safe collect, not allowing that the human supervisor to do interference in the behavior of the robot.

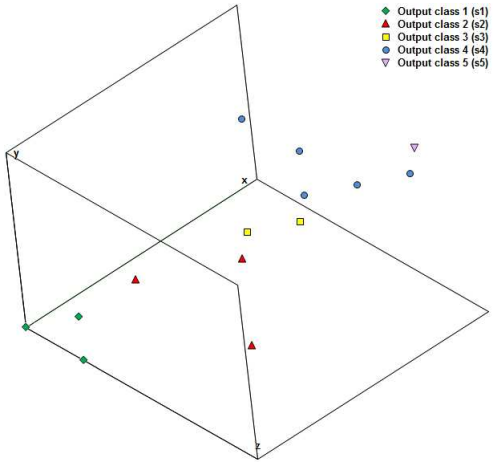

Fig. 7. Data dispersion of the validation set on 3D plan (most points are overlaped).

Table 3 shows the classification error of the supervised learning techniques using WEKA [16]. We can observe that the Bayes Network, AdaBoost and Sequential Minimal Optimization (SMO) algorithms presents the classification error similar (around 3%). Naives Bayes shows lower error compared with previous techniques. RBF network present the lower classification error compared Naive Bayes. Also shows the comparision with Support Vector Machine (SVM) for different kernel functions. Linear and Polynomial SVM present same values, but different incorrectly instances classified (see Table 4). The Sigmoid kernel function showed a larger error compared with previous techniques. Rprop shows a best classification compared Linear and Polynomial SVM (0% of error).

Table 3. Results of supervised learning techniques using the validation set.

Supervised Learning Techniques	Instances Classification Error
Rprop MLP	**0,0000**%
Bayes Network	3,7152%
AdaBoost	2,7864%
Naive Bayes	1,5480%
SMO	2,7864%
RBF Network	0,6192%
Linear SVM	0,3096%
Polynomial SVM	0,3096%
RBF SVM	0,9288%
Sigmoid SVM	67,182%

Table 4 presents the confusion matrix of the three best techniques using the validation set from Table 3, this set contains 323 instances. We can observe that the Rprop, Linear SVM and Polynomial SVM classifiers are very similar, because the main difference between these techniques are only one classification. Linear and Polynomial SVM showed only an error of classification compared Rprop (Linear - error s2 classified and Polynomial - error s4 classified). Therefore, such algorithms are very close, but Rprop MLP correctly classified all the instances.

Table 4. Confusion matrix of the best techniques using the validation set.

Rprop MLP					Linear SVM					Polynomial SVM					
a	b	c	d	e	a	b	c	d	e	a	b	c	d	e	Classified
88	0	0	0	0	88	0	0	0	0	88	0	0	0	0	s1
0	**106**	0	0	0	0	105	0	1	0	0	106	0	0	0	s2
0	0	**120**	0	0	0	0	120	0	0	0	0	120	0	0	s3
0	0	0	**1**	0	0	0	0	1	0	0	1	0	0	0	s4
0	0	0	0	**8**	0	0	0	0	8	0	0	0	0	8	s5

Experimental tests showed that the Rprop MLP output presents a better peformance in the robot behavior (see Video [1]).

[1] Experiment video available in the Internet:
http://www.youtube.com/watch?v=H5UJu2JMljk

5 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 a mobile robot tested in outdoor road following experiments. The robot was able to navigate autonomously in this environment in straight line, soft turn, or hard turn left and right since one of our goals is to find the best architecture of the ANN to be applied in different environments. Our quantitative analysis also obtained reasonable results for the learning of ANNs with the respective architectures.

As future work, we plan to evaluate other classification methods and decision making algorithms. We also planning to held in other urban environments the proposed method with GPS, in addition to integrate camera and LIDAR laser information in order to better deal with obstacles, bumps and depressions.

References

1. Bradski, G. and Kaehler, A. Learning OpenCV: Computer Vision with the OpenCV Library. O'Reilly, Cambridge, MA, 2008.
2. Canny, J. F. A Computational Approach to Edge Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, v. 8, n. 6, 679-698, 1986.
3. Chan, M., Partouche, D. and Pasquier, M. An Intelligent Driving System for Automatically Anticipating and Negotiating Road Curves. In IROS, 117-122, 2007.
4. Dahlkamp, H., Kaehler, A., Stavens, D., Thrun, S. and Bradski, G. Self-Supervised Monocular Road Detection in Desert Terrain. Robotics: Science and Systems, 2006.
5. European Land-Robot, http://www.elrob.org/, Access on 05 january, 2011.
6. Fast Artificial Neural Network, http://leenissen.dk/fann/, Access on 02 june, 2010.
7. Graefe, V. Vision for Intelligent Road Vehicles. In Proc. IEEE IVS, 135-140, 1993.
8. Markelic, I., Kulvicius, T., Tamosiunaite, M. and Worgotter, F. Anticipatory Driving for a Robot-Car Based on Supervised Learning. In Lecture Notes in Computer Science: Anticipatory Behavior in Adaptive Learning Systems, 267-282, 2009.
9. Markelic, I., Kjaer-Nielsen, A., Pauwels, K., Jensen, L. B. W., Chumerin, N., Vidugiriene, A., Tamosiunaite, M., Rotter, A., Hulle, M. V., Kruger, N. and Worgotter, F. The Driving School System: Learning Automated Basic Driving Skills from a Teacher in a Real Car. IEEE Trans. on Intelligent Transp. Systems, 2011.
10. Oentaryo, R. J. and Pasquier, M. GenSoFNN-Yager: A Novel Hippocampus-Like Learning Memory System Realizing Yager Inference. In IJCNN, 1684-1691, 2006.
11. Petrovskaya, A. and Thrun, S. Model Based Vehicle Tracking in Urban Environments. IEEE International Conference on Robotics and Automation, 1-8, 2009.
12. Pomerlau, D. A. ALVINN: An Autonomous Land Vehicle In a Neural Network. Advances In Neural Information Processing Systems, 1989.
13. Shinzato, P. Y. and Wolf, D. F. A Road Following Approach Using Articial Neural Networks Combinations. Journal of Intelligent and Robotic Systems, 2010.
14. SNNS, http://www.ra.cs.uni-tuebingen.de/SNNS/, Access on 20 november, 2010.
15. Souza, J. R., Sales, D. O., Shinzato, P. Y., Osório, F. S. and Wolf, D. F. Template-Based Autonomous Navigation in Urban Environments. In: 26th ACM Symposium on Applied Computing (SAC'11), TaiChung, Taiwan, 1381-1386, 2011.
16. WEKA, http://www.cs.waikato.ac.nz/, Access on 17 april, 2011.