Buletin Teknik Elektro dan Informatika
(Bulletin of Electrical Engineering and Informatics)
Vol. 3, No. 3, September 2014, pp. 195~200
ISSN: 2089-3191

Neural Network Navigation Technique for Unmanned Vehicle

Boumediene Selma[1], Samira Chouraqui[2]
Department of Computer Science, Faculty of Science, University of Science and Technology "Mohamed Boudiaf" USTO Oran.BP1505, Algeria
e-mail: Selma.boumediene@yahoo.fr[1], S_chouraqui@yahoo.fr[2]

Abstract

Using a neural network (ANN) for the brain, we want a vehicle to drive by itself avoiding obstacles. We accomplish this by choosing the appropriate inputs/outputs and by carefully training the ANN. We feed the network with distances of the closest obstacles around the vehicle to imitate what a human driver would see. The output is the acceleration and steering of the vehicle. We also need to train the network with a set of strategic input-output. The result is impressive, for a couple of neurons! The unmanned vehicle (UV) drives around avoiding obstacles, but some improvement or modification can be done to make this software work for a specific purpose.

Keywords: *unmanned vehicle, neural network, control*

1. Introduction

The idea is to have a vehicle that drives by itself and avoids obstacles in a virtual world. Every instant, the vehicle decides by itself how to modify its speed and direction according to its environment. In order to make it more real, the AI should only see what a person would see if it was driving, so the UV decision is only based on obstacles that are in front of the vehicle. By having a realistic input, the AI could possibly be used in a real vehicle and work just as well.

Gaming is the first use I think of when I hear "controlling a vehicle". Many driving games could use this technique to control vehicles, but there are a number of other applications that could be found for software that controls a vehicle in a virtual world, or in the real world, some of the techniques proposed include fuzzy control [1]-[3], adaptive control [4]-[6], neural networks [7], genetic algorithms [8], Lyapunov theory [9] and neuro-fuzzy control [10].

So how do we do this? There are many AI techniques out there, but since we need a "brain" to control the vehicle, neural networks would do the job. Neural networks are based on how our brain works; they seem to be the right choice. We will need to define what the input and output should be for this neural network.

2. Neural Networks

The Neural Network Artificial Intelligence comes from how brains work. This principle is applied to neural networks on a smaller scale. Today's computers don't have the power of computing that 20 billion neurons do, but even with a few neurons, we are able to have intelligent response from a neural network.

Neurons are organized in layers as Figure 1 shows. The input layer will have entries, and depending on the strength of connection to each neuron in the next layer, the input signal is sent to the next layer. The strength of the connection is called a weight. The value of each neuron in each layer will depend on the weight of the connection and the values of the neurons of the previous layer.

A driver could be compared to a "function". There are different inputs: what the driver sees. This data is processed by the brain as a function and the response of the driver is the output of the function.

A function $f(x)=y$ transforms a value x (one dimension) into y (one dimension).

Received May 3, 2014; Revised July 2, 2014; Accepted July 15, 2014

We use a back propagation neural network for the "brain" of the driver because such a neural network is capable of approximating any function with a domain and a range that each has multiple dimensions: $f(x_1, x_2,..., x_n)=y_1, y_2,..., y_m$.

That is exactly what we need here since we need multiple inputs and multiple outputs. When a neural network has just a couple of neurons, we can compute what the weight should be to get the desired response. But as we increase the number of neurons we use, we also increase the complexity of computing what the weight should be. A back propagation network lets us train the neural network which sets the weights for us. We just need to provide the desired outputs with the corresponding inputs.

After being trained, the neural network will respond close to the desired output when a known input is given and will "guess" a response for any input that doesn't exactly match the trained input-output.

The neural network used in this case has 3 layers (Figure 1). The input layer has 4 neurons, and the output layer has 3. I'll explain why later. The layer in between have 5 neurons each. When choosing the number of neurons, keep in mind that each layer and each neuron you add to the system will add computation time to calculate the weights.

2.1. The Input

What information is important in controlling a vehicle when we are driving? Firstly, we need the position of road and position of obstacles relative to us and velocity? If there are buildings on both side of the road, but none if front, we will accelerate. But if a car is stopped in front of us, we will break. Secondly, we need the distance from our position to the object. If an object is far away we will continue driving until it is close, in which case, we will slowdown or stop.

That is exactly the information that we use for our neural network. To keep it simple we have four positions *x,y*, obstacles and we need the velocity of the vehicle. To be able to do this, we need the position *(x,y)* of every object, the position *(x,y)* of the road the position *(x,y)* of the vehicle and we also need the position *(x,y)* of obstacles.

Because the neural network is a using a sigmoid function which is defined by:

$$v_j = \frac{e^{z_j}}{e^{z_j} + 1} = \frac{1}{1 + e^{-z_j}}, \qquad j = 1, m.$$

The input needs to be between 0.0 and 1.0.

2.2. The Output

The output needs to control the vehicle's speed and direction. That would be the acceleration, the brake and the steering wheel. So we need 3 outputs; one will be the acceleration/brake since the brake is just a negative acceleration, and the others will be the position.

The output from the neural network is also between 0.0 and 1.0 as the input. As it is illustrated in Figure 1, we give result in an entry (the initial positions of the reference route *(X,Y)*, the velocity *V*, back to donkey *D*, *F* traffic light, vehicle *O* which shows the state of overshoot) and three outputs (new positions *(X,Y)*, and the velocity *V*).

According to the calculations of the network (back propagation algorithm) the control action is applied to the input of the process that lets you know the state of the system to calculate the error which has the input of the network controller.

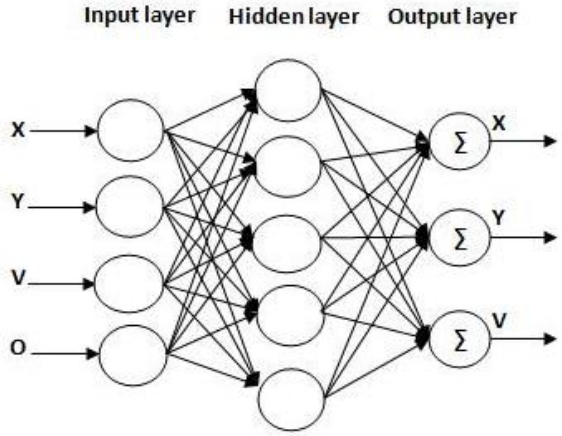

Figure 1. ANN Architecture

2.3. Training

As I mentioned earlier, we need to train the neural network to approximate the function we need it to accomplish. We need to create a set of input-output that would be the main reaction we want the neural network to have.

Choosing the right input-output to train a neural network is probably the trickiest part of all. I had to train the network with a set of input-output, see how it reacted in an environment, and modify the entries as needed. Depending on how we train the network, the vehicle can hesitate in some situations and get immobilized.

We make a table (Table 1) with different arrangements of obstacle relative to the vehicle, velocity and the desired reaction from the UV.

Table 1. ANN situations

Input Neurons Relative to obstacle and velocity			**Output Neurons Relative to velocity**	
obstacle	***Velocity***	***Antecedent***	***Acceleration***	***Consequent***
No obstacle	Full Acceleration		Full Acceleration	No Action
No obstacle	Low Acceleration		Accelerate	
Donkey	Full Acceleration	Distance donkey is D	Slow down	
Traffics lights	Full Acceleration	Color traffic lights is Green	Slow down	
Traffics lights	Full Acceleration	Color traffic lights is Red	Stop	
Vehicle	Full Acceleration	Our vehicle speed is greater than the vehicle on the road	Full Acceleration	Change of position and exceeding
Vehicle	Full Acceleration	Speed of your vehicle is smaller than that of vehicle in road	Full Acceleration	No Action

3. Simulatio Result and Discussion

To show the contribution of the control by ANN, simulation was approved on an unmanned vehicle. Figure 2 shows the path followed by the UV controlled by the proposed ANN including obstacles described in the above sections.

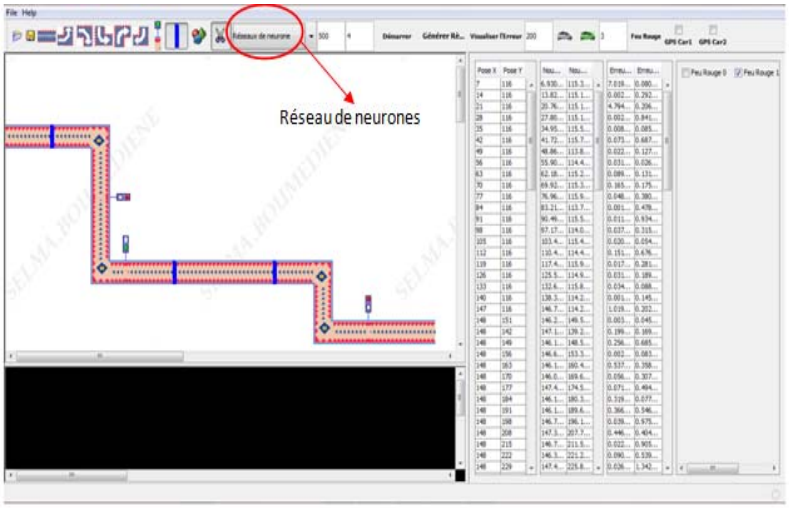

Figure 2. Learning about this sample reference path

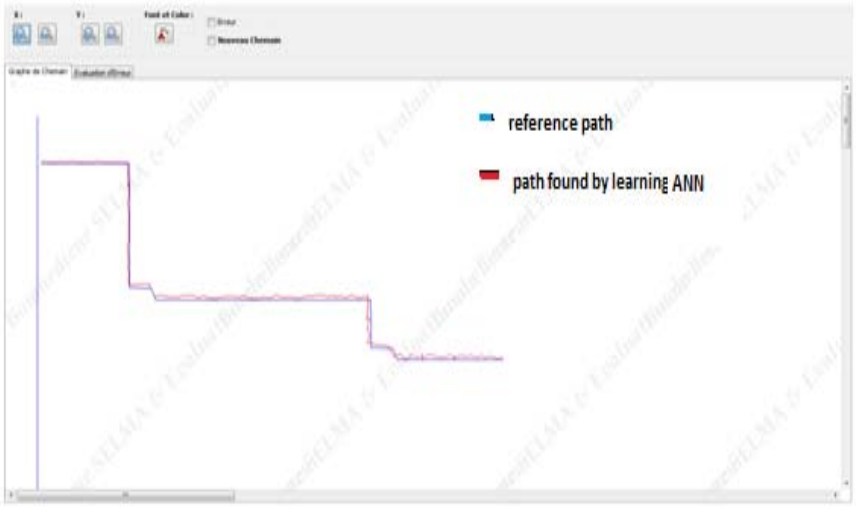

Figure 3. The path found by RNA

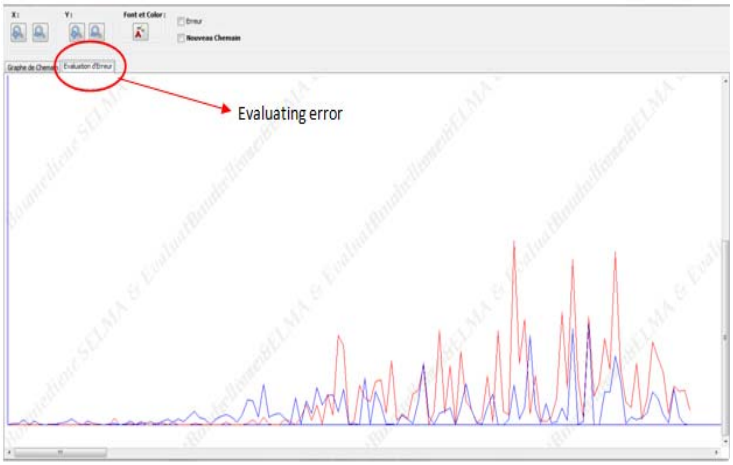

Figure 4. The error between the reference path and the path calculated

As we can see from Figure 2 and Figure 3 the path obtained from simulation setup is more close to the reference path which validates the proposed method. The function of the Speed Controller subsystem control is to achieve the desired speed that is to say the increase and decrease in speed on the road and especially in front of obstacles, so the accelerator control. Figure 7 shows the variation of the UV speed in function of obstacles come across the road.

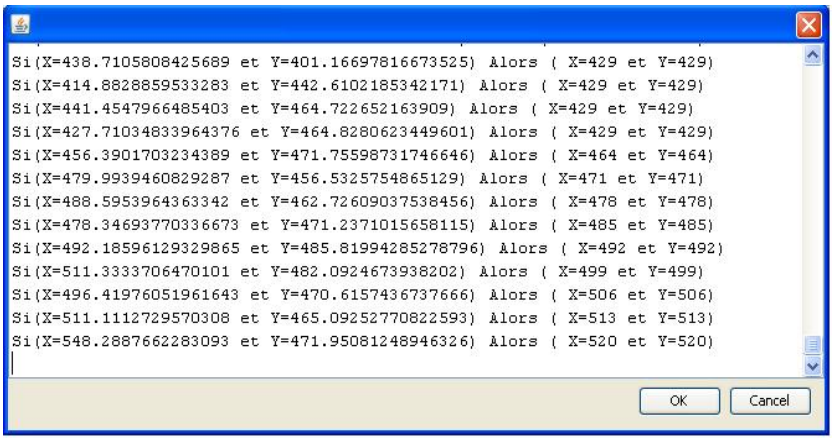

Figure 5. Generation of rules

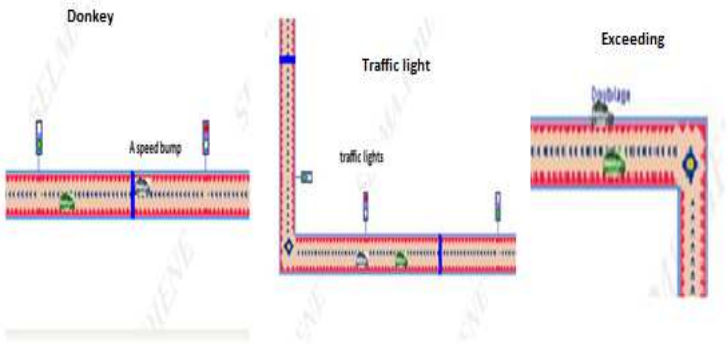

Figure 6. The obstacles Maneuver with UV

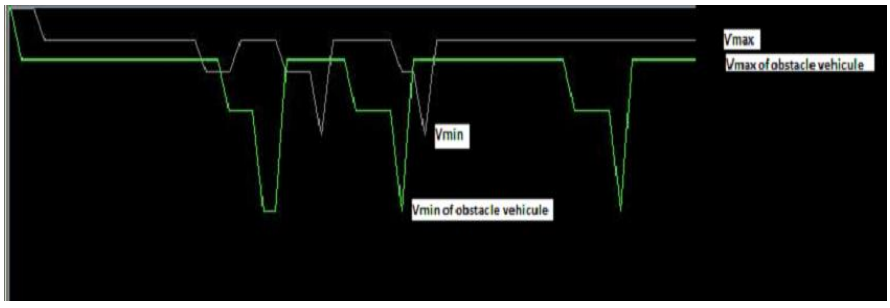

Figure 7. Vehicle speed variations at obstacles

From the above figure, it can be seen that the UV can indeed avoid obstacles and reach the targets. To verify the feasibility of proposed method Table 2 shows results of neural network controller.

Table 2. The values of statistical results

Datasets	Statistical results	
Rate accuracy (%)	Total classification Accuracy (%)	
Set X	97.63	
Set Y	98.86	98.24

Therefore, it can be concluded that the ANN controller have a good potential to effect fast response to obstacles and reduce errors.

4. Conclusion

Using a neural network with back propagation works well for our purposes, but there are some problems once implemented and tested. Some improvement or modification can be done to either make the program more reliable or to adapt it to other situations.

ANN architecture has demonstrated a good performance in modeling the trajectory of an Unmanned Vehicle. This pre processing feature allows ANN to converge faster and better. Through experimental tests, it is found that the proposed system is a powerful tool for controlling non linear dynamical systems.

References

[1] W Banks, G Hayward. Fuzzy logic in embedded microcomputers and control systems, A2-490 Dutton Drive Waterloo, Ontario, Canada: Byte Craft Limited. 2001.
[2] L Doitsidis, KP Valavanis, NC Tsourveloudis, M Kontitsis. *Framework for fuzzy logic based UAV navigation and control.* In Proceedings of the IEEE international conference on robotics and automation. New Orleans, LA. 2004.
[3] HB Verbruggen, HJ Zimmerman, R Babuska. Fuzzy algorithms for control. USA: Kluwer Academic Publishers. 1999.
[4] B Andrievsky, A Fradkov. *Combined adaptive autopilot for an UAV flight control.* In International conference on control applications Glasgow. 2002: 290–291.
[5] KJ Aström, B Wittenmark. Adaptive control. Lund Institute of Technology: Addison Wesley Publishing Company. 1989.
[6] CJ Schumacher, R Kumar. *Adaptive control of UAVs in close-coupled formation flight.* In American control conference. 2000: 849–853.
[7] Y Sundararajan, N Li, P Sratchandran. Neuro-controller design for nonlinear fighter aircraft maneuver using fully tuned RBF networks. *Automatica*. 2001: 1293–1301.
[8] O Cordon, F Gomide, F Herrera, L Magdalena. Ten years of genetic fuzzy systems current framework and new trends. *Fuzzy Sets and Systems.* 2004; 141: 5–31.
[9] W Ren, RW Beard. *CLF-based tracking control for UAV kinematic models with saturation constraints*. In 42nd IEEE conference on decision and control. 2003.
[10] B Selma, S Chouraqui. Trajectory estimation and control of vehicle using neuro-fuzzy technique. *International Journal of Advances in Engineering & Technology*. 2012; 3(2): 97-107.