A method to verify a path planning by a back-propagation artificial neural network

Aldo-Francisco Contreras-González, José-Isidro Hernández-Vega, C. Hernández-Santos, and Dolores-Gabriela Palomares-Gorham

Instituto Tecnológico de Nuevo León, División de Estudios de Posgrado e Investigación, Av. Eloy Cavazos #2001 Col. Tolteca, Guadalupe, Nuevo León, México.
aldocontrego@gmail.com, jose.isidro.hernandez@itnl.edu.mx,
WWW home page: http://posgrado2.itnl.edu.mx

Abstract. This paper proposes a method to reduce the error on the position given by the global position system (GPS), using a back-propagation artificial neural network (ANN) focused on a known linear path employed as target, which is used on the creation of path planning for an Unmanded Aereal Vehicle (UAV) tested on real time. This document shows the step by step development of the equations used and the algorithm for trigonometric functions and then to a lineal path made by a path planning code. The implementation of this project was made on Python for real time flight in which the GPS data are acquired and graphed to analyse results.

Keywords: Path planning, Unmanned Aerial Vehicle , Artificial Neural Network, Global Position System

1 Introduction

Programming control of autonomous paths for UAVs, nowadays, has been difficult for the inaccuracy of the received signal from the global position system (GPS); [6] the signal received is inconsistent because of diverse factors such as ionospheric and atmospherics delays, satellite clock and receiver errors, multipath effects, between others. The destination place is affected on a range of 5 to 10m from the desired point because the physical location does not match with the device signal. When the device performs an autonomous path it is physically following a lineal trajectory, but it is sending an intermittent signal on the space showing a variation on several points on the space which are not real.

Jha, Chattopadhyay, [4] proposes a framework for the direct navigation to objectives in a reactive form without Global Positioning facilities on dynamic and unknown environments. A network of mobile sensors is used to localise interest regions to the path planning of an autonomous mobile robot. The algorithm developed has been used to spread local decisions for target detection on a mobile

sensor network, thus, an assumption location map (belief system) for the target detected by the network is achieved.

Nieuwenhuisen, M [5] mentions that the good performance on aerial vehicles on partially known environments to world level require coherent plans based on incomplete models and fast reactions to obstacles unknown for the real time path planning on free collisions paths.

An artificial neural network is used to reduce the error given by the global position system [2] and give more accuracy to the point to get on the path [8] as we mentioned, the kind of path is point to point and the algorithm for path planning has to be accurate and precise to arrive to the point.

During the tests that were carried out to feed the artificial neural network and establish knowledge, a linear path followed by a thread was conducted; the data acquired by the GPS of the UAV is stored in real time On this project the GPS device is used [10] without the inertial measurement unit (IMU).

2 ANN math used

The mathematical method developed here, proposed by Aguado [1], consists on estimating the network represented on a simplified form on the Fig.1. To illustrate this let us assume that we have a set of data pairs, $P = 1, 2, ..N$ for the neuron $S_\jmath$ of the layer, this is defined as the functions corresponding to the sample excitation p data.

$$S_j^p = \sum_{i=1}^{n} W_{ij}^p x_i^p + W_{n+1,j}^p \tag{1}$$

We used the back-propagation artificial neural network shown on Fig. 1; to use this, we must make a standard data input and output in the range 0-1. To declare "bias" as random numbers, the input given to the network are values different to zero. A sigmoid function is used as activation function, then the output of the neuron $\jmath$ of the hidden layer is expressed as:

$$h_j^p = f(S_j^p) = \frac{1}{1 + exp(-S_j^p)} \tag{2}$$

Excitation of the neuron k of the output layer is calculated analogously by the expression:

$$r_k^p = \sum_{j=1}^{l} v_{jk}^p h_j^p + v_{l+1,k}^p \tag{3}$$

and finally, the output of the neuron k of the output layer is expressed as:

$$O_k^p = f(r_k^p) = \frac{1}{1 + exp(-r_j^p)} \tag{4}$$

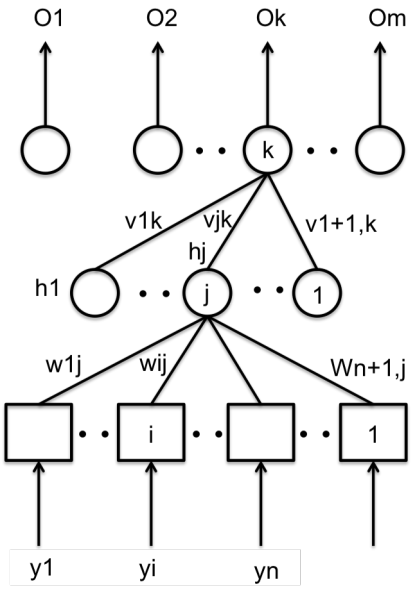

Fig. 1. Back-propagation artificial neural network

We now define the output error k, for sample p as:

$$e_k^p = y_k^p - O_k^p \tag{5}$$

The criterion to be minimized in the sample $\jmath$ is defined by:

$$E_p = \frac{1}{2}\sum_{k=1}^{m}(e_k^p)^2 = \frac{1}{2}\sum_{k=1}^{m}(y_k^p - O_k^p)^2 \tag{6}$$

The training process of the neural network consisted in presenting the inputs sequentially, calculate the outputs on the network, the error and the citerion E, and to apply the procedure to minimize the function error; the procedure is to always move in the direction of the negative gradient of the function with respect to the coefficients w and v, this is called "steepest descent", [9].

3 ANN Algorithm

The mathematical method was implemented with MATLAB R2014a. In this tool the equations are written and adapted to the language to estimate any given function, for ν number of samples on the network, for ξ number of periods, for η learn coefficient between 0 and 0.9 and for any neuron number $\jmath$. This is represented on the algorithm shown below:
Algorithm 1:

1. Compute number of samples
2. Compute function
3. Convert sigmoid function data
4. Compute periods, learn coefficient and neuron number
5. While: ep $\leq$ Number of periods
6. $\rightarrow$ For $i = 1$: neuron number
7. $\rightarrow \rightarrow$ $h_j^p = f(S_j^p) = \frac{1}{1+exp(-S_j^p)}$
8. $\rightarrow$ For $i = 1$:neuron number
9. $\rightarrow \rightarrow$ $r_k^p = \sum_{j=1}^{l} v_{jk}^p h_j^p + v_{l+1,k}^p$
10. $\rightarrow$ $Error = (Target - O_k^p = f(r_k^p) = \frac{1}{1+exp(-r_j^p)})$
11. $\rightarrow$ For $i = 1$:neuron number
12. $\rightarrow \rightarrow$ using $v_{jk}^p = v_{jk}^{p-1} + \eta \delta_k^p h_j^p$
13. $\rightarrow$ For $i = 1$:neuron number
14. $\rightarrow \rightarrow$ using $w_{ik}^p = w_{ij}^{p-1} + \eta \Delta_j^p x_i^p$
15. $\rightarrow$ For $i = 1$:neuron number
16. $\rightarrow \rightarrow$ using $h_j^p = f(S_j^p) = \frac{1}{1+exp(-S_j^p)}$
17. $\rightarrow$ For $i = 1$:neuron number
18. $\rightarrow \rightarrow$ using $r_k^p = \sum_{j=1}^{l} v_{jk}^p h_j^p + v_{l+1,k}^p$
19. $\rightarrow$ using $O_k^p = f(r_k^p) = \frac{1}{1+exp(-r_j^p)}$
20. $\rightarrow$ using $e_k^p = y_k^p - O_k^p$
21. $\rightarrow$ $ep = ep + 1$
22. $mse = mean((Nsal - Target).^2)$
23. Plot

You can find this MATLAB code on: `https://drive.google.com/open?id=0B6xtNWuLHIRMT2U0c0gtbG1MNDg`

4 Function graphic

For any trigonometric or lineal function with one variable, we can use the last algorithm, in which the number of tests corresponds to the amplitude of the graph or the number of tests to be estimated in the ANN. Trigonometric functions such as sine and tangent were tested in order to verify the behaviour and the mean squared error (mse).

To match what we have done at the time, for path planning using global positioning data it is necessary to know the starting point and end point of the desired path. Once obtained these data, we can create our lineal function full of points to match with the quantity of inputs as Target, with the input data obtained from the global position system. After knowing our target input data and our data target we can create our artificial neural network.

5 Path planning verification

The path planning works like a multithread algorithm. The first thread named "data acquisition" is responsible for sending a command to update the UAV and to perform the most frequent calculations updating these data with global variables.

The second thread named "acquisition and writing sample", performs verification of existence of strong GPS signal and checks the level of battery charge, and once all data is correct, it starts the storage of the sampling information such as latitude, longitude and height above ground. The third thread named "Commands sender" is the UAV control; it generates a virtual barrier between the current point and the next point, checks that the angle is physically relative to the angle generated against the GPS point and it sends the UAV motion commands to perform the route.

In tests to feed the neural network and establish knowledge, took just by walking in a straight line followed by a thread ensuring that is kept as straight as possible, while advancing, the data acquired by the GPS device are saved, on this project a only the GPS devise is used [10], without use the inertial measurement unit (IMU).

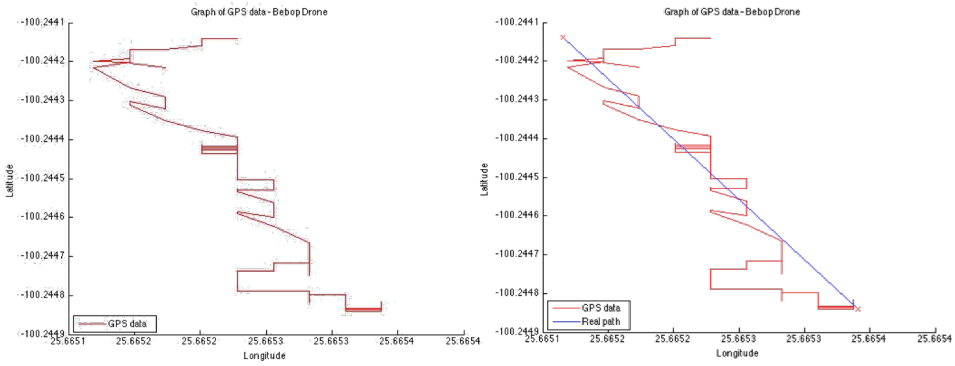

Fig. 2. GPS data without ANN and starting point to ending point trajectory

The file was plotted, as shown in Fig.2 left, contains 736 points of latitude and 736 in longitude, using this data to feed the network as a target, they are generated for each point of the actual travel, as sown on the Fig.2 right.

6 Results

For testing was used the UAV from Parrot company the Bebop 1, the programming language was Python, to establish communication and commands sending was used the library of Martin[7] which allows to connect and move the drone under commands, this library was modified in some parts to have a better control. The weather conditions were non controlled, under a wind of 20 knot and a temperature around the 23 - 28 ^oC.

Fig. 3. Updating ANN with GPS data to the path planning

After feeding the ANN we can see the behaviour (Fig.4) taken by the estimation of the back-propagation artificial neural network who is very similar to the established straight as target with an $mse = 7.651903681967e - 05$.

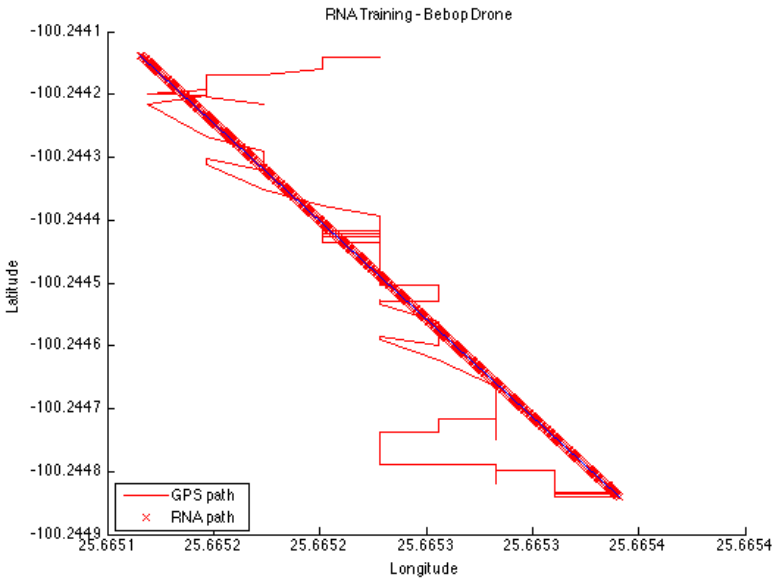

Fig. 4. GPS path without filter and using the ANN

Once the ANN was trained, we took the synaptic weights and implemented them in the path planning algorithm modifying the original data given by the GPS, as shown in Fig. 3. To test physically this algorithm three flights were tested as shown in Fig. 5, with the same GPS points to reach for all(Table 1) and with uncontrolled weather conditions using a flight algorithm of path planning by GPS; we can compare the improvement in accuracy with respect to the data given by the GPS (left side without feedback of the ANN and right side with feedback). The sample data is acquired by generating a file in real time by saving the current global position of the UAV every second elapsed to be interpreted and plotted later.

Table 1. GPS data for path planning

Latitude	Longitude
25.665330	-100.244560
25.665399	-100.244327
25.665293	-100.244223

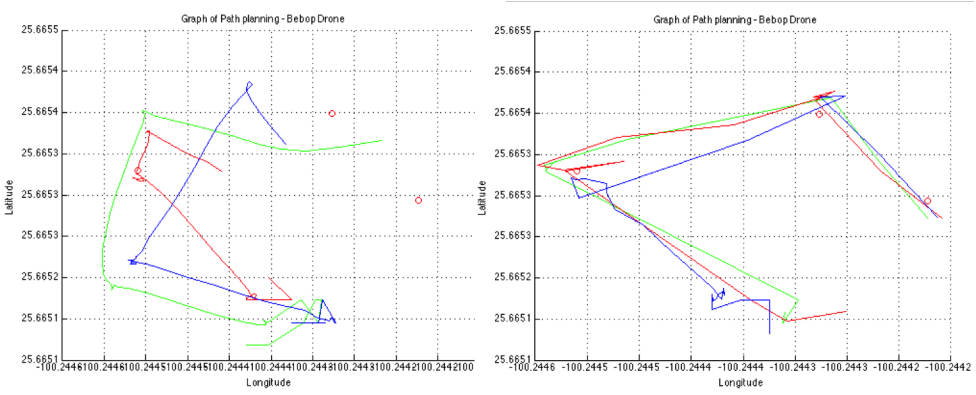

Fig. 5. Three flights comparison

The same algorithm for path planning was used in all tests. In Fig. 5 red circles represent points target of the path planning algorithm. On the left side we have an error in circumference of about 10m (32.8ft); on the contrary, on the right side (which uses the estimate of the ANN), an error to the circumference of 1.5m (4.6ft) was obtained. During the flight, this was measured by placing flags on the testing ground in the points shown in Table 1.

7 Conclusions and future work

In this document we studied a method of reducing the error given by the GPS using path planning algorithm for UAVs. The ANN back-propagation algorithm predicts the location of the UAV giving a start point and an end point to generate a straight line that breaks down into the number of data provided by the GPS. As the UAV moves, the algorithm predicts its location with respect to the learned error and the point at which it encounters respect to the line, thus giving a smooth and accurate point to point travel. It is noteworthy that the tests were taken on a field without climate controlled conditions and without obstacles in the path of UAVs. A sampling time of one second is considered, as the calculated processing time for this algorithm (in a conventional system) is greater than 0.5 and less than 0.8 seconds.

Our future task is to compare this method and implement a Kalman filter [3] in order to compare the performance and accuracy of the path planning code.

References

1. A. Aguado Behar.: "Temas de identificacion y control adaptado" Instituto de Ciberntica Matemtica y Fsica (ICIMAF), La Habana Cuba, 257–268 (2000)
2. El-Sheimy, Naser, Kai-Wei Chiang, and Aboelmagd Noureldin.: "The utilization of artificial neural networks for multisensor system integration in navigation and positioning instruments." Instrumentation and Measurement, IEEE Transactions on 55.5, 1606–1615 (2006)
3. Guerrero, Juan S. Guerrero, et al.: "Instrumentation of an Array of Ultrasonic Sensors and Data Processing for Unmanned Aerial Vehicle (UAV) for Teaching the Application of the Kalman Filter." Procedia Computer Science 75 : 375–380 (2015)
4. Jha, Devesh K., et al.: Path planning in GPS-denied environments via collective intelligence of distributed sensor networks. International Journal of Control, 1–16 (2015)
5. Niewenhuisen, Matthias; Behnke, Sven.: Layered mission and path planning for MAV navigation with partial environment knowledge. En Intelligent Autonomous Systems 13. Springer International Publishing, 307–319 (2016)
6. Ryu, Ji Hyoung; Gankhuyag, Ganduulga; Ching, Kil To.: Navigation system heading and position accuracy improvement through GPS and INS data fusion. Journal of Sensors, 501, 7942963 (2016)
7. robotika/katarina (online) GitHub, https://github.com/robotika/katarina
8. Sharaf, Rashad, and Aboelmagd Noureldin.: "Sensor integration for satellite-based vehicular navigation using neural networks." Neural Networks, IEEE Transactions, 18.2, 589–594 (2007)
9. Teunissen, P. J. G.: Least-squares estimation of the integer GPS ambiguities. En Invited lecture, section IV theory and methodology, IAG general meeting, Beijing, China. (1993).
10. Zhao, Handong; LI, Zhipeng.: Ultra-tight GPS/IMU Integration based Long-Range Rocket Projectile Navigation. Defence Science Journal, 66.1, 64–70 (2016)