CS570 - Artificial Intelligence Project 3a

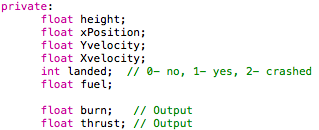
Chihsiang Wang

# Artifical neural Network

## By Chihsiang Wang

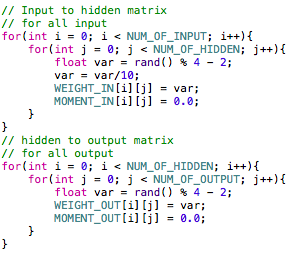
**Abstract**

The goal of this project is to make an ANN (Artificial Neural Network), which can help a moon lander to land successful after get enough training. There are seven inputs for the moon lander, after calculate with the hidden layer, it gives two output (burn and thrust) to control the lander. The ANN will use the training method to get the output and test if it can let the lander landing successful. And using back propagation algorithm to update the weight between each layers to find the error value, the error value will reduce and get closer to zero, by this method we can find the right output to land successful. The data declaration shows in the figure below:

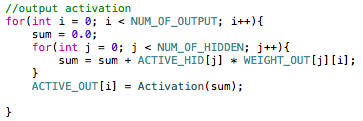


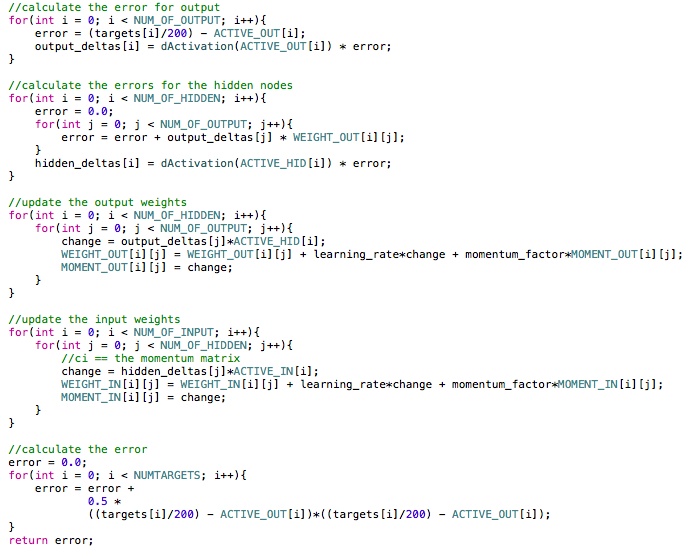
**Algorithm - ANN**

There have three layers in the ANN algorithm, input, hidden layer, and output layer. The main idea that how to get the output is that I give the random input and after calculate the weight and send it to the hidden layer. In the hidden layer I use 7 nodes to contain the value (because we have seven input).



With the activation function (using tanh() function) it will store the sum of the weights in each node, and send it to the output layer. Because of our weights are making randomly, so I use the back propagation method to get the value of error. By this method I can find how is the correct burn and thrust.

****

****

**Result**

I have tested with different acceleration, training times, and tried to let the lander lands for 5000 times, the outputs are list below:

**Acceleration 1:**

Macintosh HD:Users:chishaung:Desktop:螢幕快照 2013-04-07 下午10.02.45.png

Macintosh HD:Users:chishaung:Desktop:螢幕快照 2013-04-07 下午10.05.26.png

Macintosh HD:Users:chishaung:Desktop:螢幕快照 2013-04-07 下午10.08.00.png

**Acceleration 2:**

Macintosh HD:Users:chishaung:Desktop:螢幕快照 2013-04-07 下午10.02.27.png

Macintosh HD:Users:chishaung:Desktop:螢幕快照 2013-04-07 下午10.05.39.png

Macintosh HD:Users:chishaung:Desktop:螢幕快照 2013-04-07 下午10.08.37.png

**Acceleration 3:**

Macintosh HD:Users:chishaung:Desktop:螢幕快照 2013-04-07 下午10.02.03.png

Macintosh HD:Users:chishaung:Desktop:螢幕快照 2013-04-07 下午10.06.00.png

Macintosh HD:Users:chishaung:Desktop:螢幕快照 2013-04-07 下午10.09.13.png

**Discussion**

I have noticed that both acceleration and training times will both affect the success of landing. In the same training times, acceleration 1 has a better result than acceleration 3, I am not really sure what makes the difference, but I guess that the acceleration will affect the position-y to drop fast, in the other words, a higher acceleration will reduce the ANN’s analyzing time, so in this situation the ANN need more training time to realize what is the best output. And I am not surprised that in the same acceleration, more training times gives a better A.I. to make the right output.

I also notice that when the ANN is not full trained, it’s really not stable to give the answer, for example, I have tried to use the setting of acceleration 1 and trained the ANN for 5000 times, sometimes it give me about 80% chance to land successful, but sometimes it give me 20% of chance to land successful. I think that means before an ANN got full trained, what affect the result most is the “luck”, but after it got an enough trains, this situation will not occur much (but before it full trained, still happened sometimes).

**Conclusion**

ANN is a method to imitate how a human to think (input), and get the answer (output) after make a decision (hidden layer). As human, we can make a better decision after we got enough trains. But because of some outer effects, for example, in this project the wind will affect the x-position, and the acceleration will affect the y-position. When an ANN has lower outer effects (acceleration is smaller) and more training times, it gets more successful case to land safely. Because of the training algorithm is using the random value to create the weight, before the ANN has been whole trained, still have much risk to make the right decision, if we can’t avoid outer effects, the best way is to increase the training times to get the best result. But more training times means it will take more time space and memory space to solve it, though in this project it’s fast to train the algorithm, but when we apply ANN in some other more complex programs, it can be a problem to ask a A.I. to do a right decision in a short time.