Airplane Detection using GP

By Collin Price

# Introduction

The goal of this experiment is to use a genetic program (GP) to evolve a spatial image analyzer to detect the location of airplanes. The typical image that will be used is a satellite photograph. The GP will colour all of the pixels it thinks belong to an airplane in red. The program will attempt to derive a classification rule for each pixel to determine if it is part of an airplane.

# Experiment Setup

**Parameters for Runs**

Random seed: current time

Population size: 1000

Number of generations: 100

Training set size: 1 Image

Testing set size: 1 Image

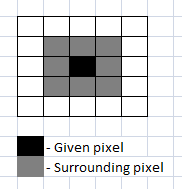
**GP Language**

Terminals

stdDev – standard deviation of nine surrounding pixels

9 pixels – is a 3x3 grid around a given pixel. See figure 1.

Figure 1:



Functions

‘-‘ – subtracts two given numbers

‘+’ – adds two given numbers

‘\*’ – multiplies two given numbers

‘/’ – protected division divides two given numbers. If second number of zero the result is zero

Max – returns the max number of two given numbers

Log – the natural logarithm of the absolute value of a given value

**Training Set**

The training image was created by taking a satellite photograph of some airplanes at an airport and masking each airplane with the colour red. See figure 2 for a sample image. The training phase of the GP was based on the number of pixels it detected as being an airplane compared to the red pixels on the masking image.

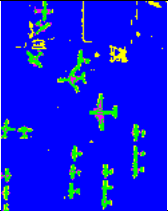
**Figure 2.**

1.  b) 
2. Is a black and white training image
3. Masked training image

**Testing Set**

In the testing set other satellite photographs are processed through the best Individual in the GP. Two images are generated from the test. The first image is a copy of the testing image except with the pixels it thinks is an airplane filled in with red pixels. The second image is a colour coded image that shows which pixels are correctly identified and which were incorrectly identified (Green = true positive, Blue = true negative, Yellow = false positive, Red = false negative). See figure 3 for an example.

**Figure 3.**

****

Sample output from the results of testing.

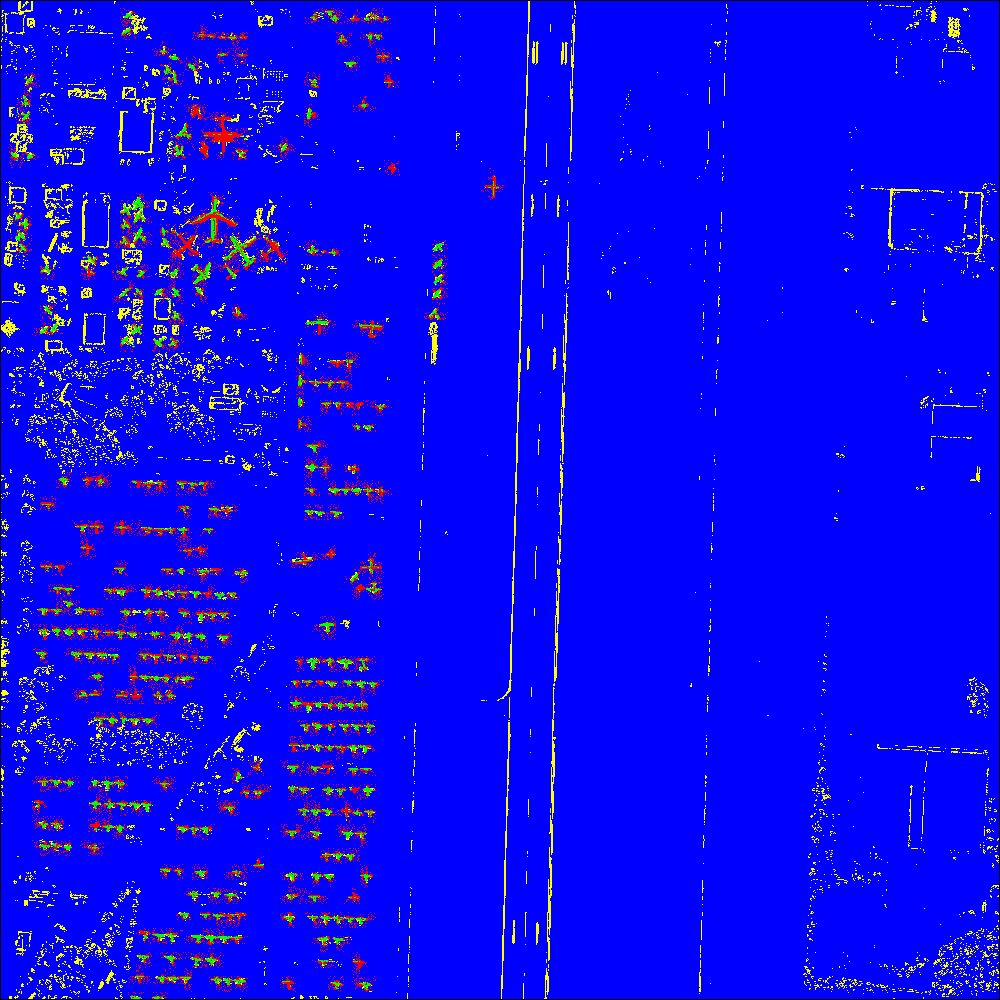
# Results

Best Run (Red pixels are when the GP thinks it’s an airplane)



Best Run:

Colour Coded (Green = true positive, Blue = true negative, Yellow = false positive, Red = false negative)



Resulting Program:

(- (- (- (+ (+ (\* (Log (e (\* (Log (/ (- p9

p8) (/ p1 (Log (e p4))))) p3))) (/ p3 p8))

(/ p2 p5)) (Max (/ (e (/ p3 p8)) p3) (+ (+

(\* (Log (e (\* (Log (- (/ (Max p6 p1) (/ p2

p5)) p4)) p9))) (/ p3 p8)) (\* (Log (/ p2

(Max (Max (/ (/ p6 p4) (/ p9 (Max p6 p9)))

(/ p8 p6)) p9))) (e (Log (Max (Log (e (\*

(Log (e p4)) (/ p3 p8)))) (Max p6 p1))))))

(Max (/ (e (/ p3 p8)) p3) (\* (Log (/ (- p9

p4) p6)) (\* (Max p6 p1) p1)))))) (\* p3 p9))

(+ (- p9 (+ (+ (\* (Log (e (\* (/ p2 p3) (+

p8 p5)))) (/ (/ p2 p5) p8)) (\* (Log (/ (-

(+ p8 p5) (/ (Max p6 p1) (/ p2 p5))) (Max

StdDev p9))) (e (Log (\* (Log (e p4)) (/ p9

(/ p2 p5))))))) (/ p8 (e (\* (/ p2 p3) (e

(- p5 (+ p3 p9)))))))) (- p9 (+ (+ (/ p8

p6) (\* (- (- (e (- StdDev (+ p3 (- p9 p4))))

(e (/ p6 p4))) (Log (/ (- p9 p4) p6))) (/

(e (/ (/ p8 p6) (Max p6 p2))) (- (Max (\*

(/ p3 p8) (/ p2 p3)) (\* (Log (/ p1 (Max (/

(- p9 p4) (/ p6 p4)) p9))) (Max (/ (/ p6

p4) (/ (e (/ p8 p8)) (Max p6 p9))) (/ p8

p6)))) (Max (Log (\* p3 p6)) (/ (\* p8 (/ (/

(/ (- p9 p4) (/ p6 p4)) (Max p6 p9)) (Max

(/ (Max p6 p1) (/ p2 p5)) p9))) (/ p9 (Log

(\* (Log (e p4)) (/ p6 p4)))))))))) (- (+

(/ p2 p5) p5) (\* p1 p9)))))) (+ (- (Log (e

p4)) (+ (+ (\* (Log (e (\* (Log (/ p9 (/ (+

p8 p5) (\* (/ p2 p3) (- p9 p4))))) p3))) (/

p3 p8)) (/ p2 p5)) (Max (/ (e (/ p3 p8))

p3) (- p9 p4)))) (- (Max p6 p9) p4)))

This graph is an average taken over 10 runs on the training set:

Training Image:

Number is correctly identified pixels (True Positives and False Positives). Total Pixels = 19468

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Run #1 | Run #2 | Run #3 | Run #4 | Run #5 | Run #6 | Run #7 | Run #8 | Run #9 | Run #10 |
| Hits | 17937 | 17889 | 17661 | 17902 | 17845 | 17703 | 17811 | 17922 | 17762 | 17839 |

Testing Image:

Number is correctly identified pixels (True Positives and False Positives). Total Pixels = 1000000

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Run #1 | Run #2 | Run #3 | Run #4 | Run #5 | Run #6 | Run #7 | Run #8 | Run #9 | Run #10 |
| Hits | 949474 | 938432 | 94592 | 94631 | 93521 | 947583 | 94936 | 94894 | 94925 | 94861 |

Another good resulting program:

(+ (Log (Log (\* (\* (/ (\* (Log (e p9)) (/

(/ (\* (Log p7) (e p5)) (e p8)) (\* (/ (\* (Max

(e (\* (Log p7) (- (Max p6 p2) p7))) p6) (/

(e p7) (+ (+ (Log p7) p2) (Max p6 p5))))

(Log (e p8))) (e p7)))) (Log (\* (\* (Max (e

(\* (Log p7) (- (Max p6 p5) p7))) (/ (e p7)

(Log p7))) (e p7)) (e p9)))) (/ (\* p5 (/

(/ (\* (Log p7) (e p5)) (Log (\* (\* (Max (e

(\* (Log p4) (- p9 p7))) (e p8)) (e p8)) (/

(e p7) (Log p7))))) (\* (/ (/ (e p7) (+ (+

(Log p7) p2) (Max p6 p5))) (+ (- (Max p6

(Max p6 p5)) p7) (Log p8))) (e p8)))) (Log

(\* p8 (e p9))))) (e p8)))) (Log (/ (Log (Log

(\* (\* (/ (e p7) (Log (\* (\* (Max (e (\* (Log

p4) (- (Max p6 p5) p9))) (e p8)) (e p8))

(e p8)))) (e p7)) (e (\* (/ p4 (Max p6 p5))

(+ (Log (e p7)) (/ (- (Max p6 p5) p9) p9)))))))

(e p8))))

# Discussion of Results

During the development of this program I found that the most difficult part was finding a suitable fitness function. At first the GP would optimize for non-airplane pixels so that during the testing phase it would return pixel as a non-airplane. This problem happened less often once standard deviation was added to the GP terminal set.

I found that the training and testing images for the GP has to be similar. If the GP was trained on pictures of large airplanes and tested on small airplanes it would have a hard time identifying pixels. In most cases it would act as an edge detector.

The type of airplanes the GP trained on were the only airplanes it could detect during the testing phase. If you look at the results from my testing you can see that darker airplanes were not detected as being airplanes. This could be because the GP was trained on lighter coloured airplanes.

According to the graphs the best individual had only increased a small amount over the 100 generations. I thought this was interesting so I output the best individual at the end of each generation to see if the results were improving. The early generations output terrible results. This showed that even the slightest improvement to the GP’s fitness would change the overall testing results dramatically.

# Conclusion

The results from my testing showed that airplane identification is possible through GP. The training of this GP took and long time and I think that if I had limited my tree depth I could have trained a lot faster. To improve the results of this GP I would have trained it on a wider variety of airplane shapes and colours. Another change would have been to make a large moving window. After comparing my results with other students I should have used a 5x5 grid or a 9x9 grid for a greater range of pixels. I suspect that the reason it mostly developed an edge detector was because of the small sample of 9 pixels.