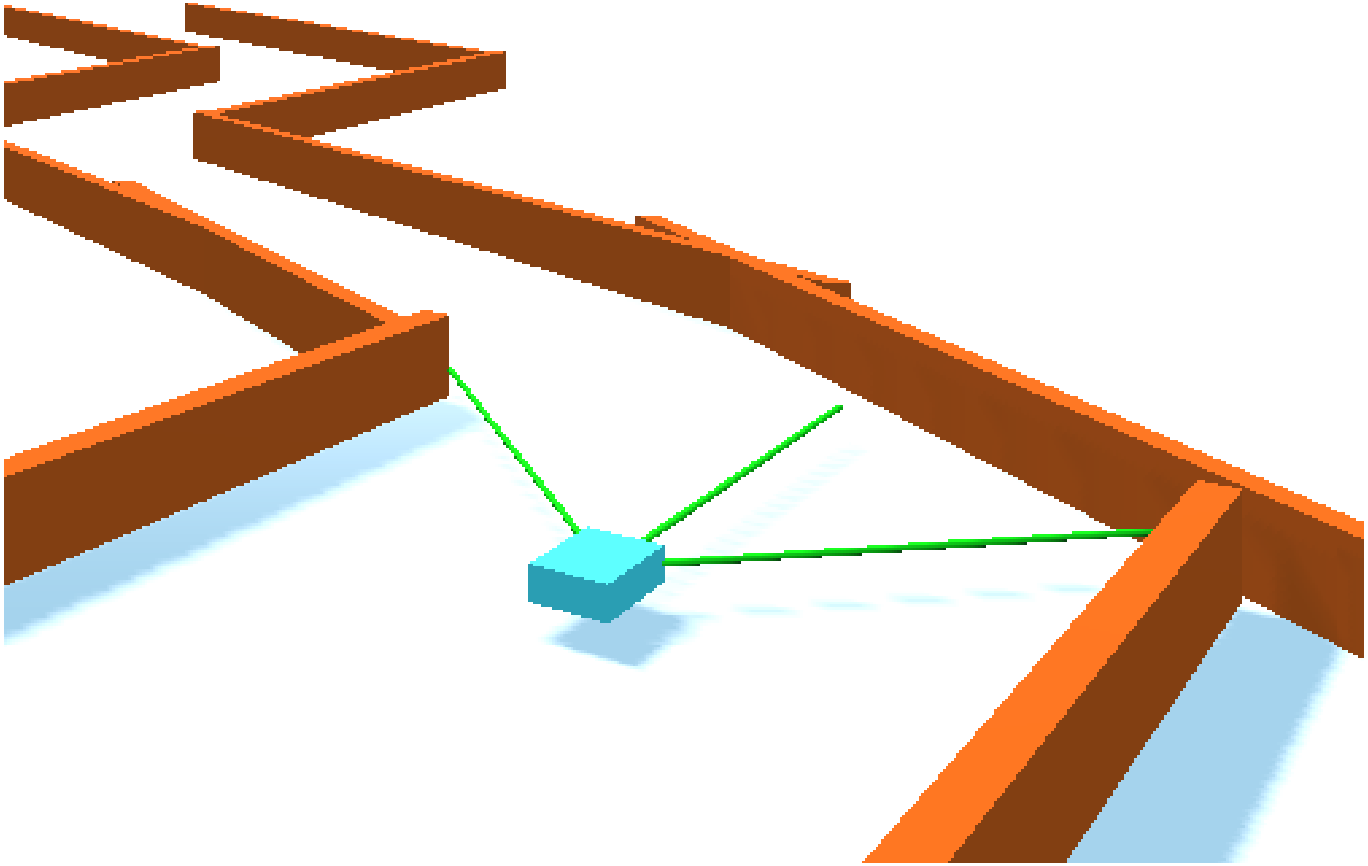
Self-Driving (Virtual) Car

Peter Sumners

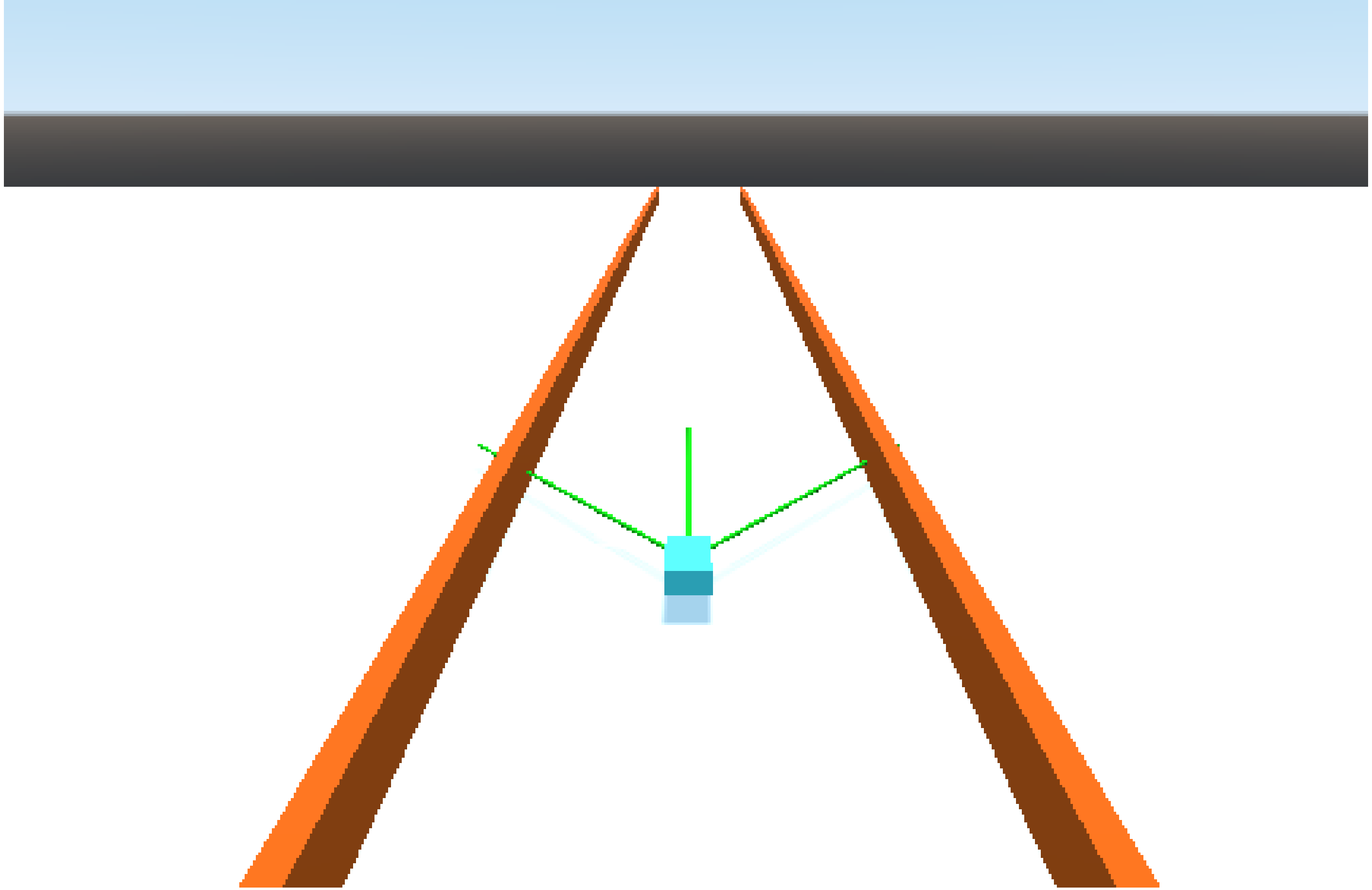
Executive Summary

- car: sensors and movement
- neural network: weights and stuff
- genetic algorithm: hello, darwin
- courses (2 of them!)
- why? because it’s cool

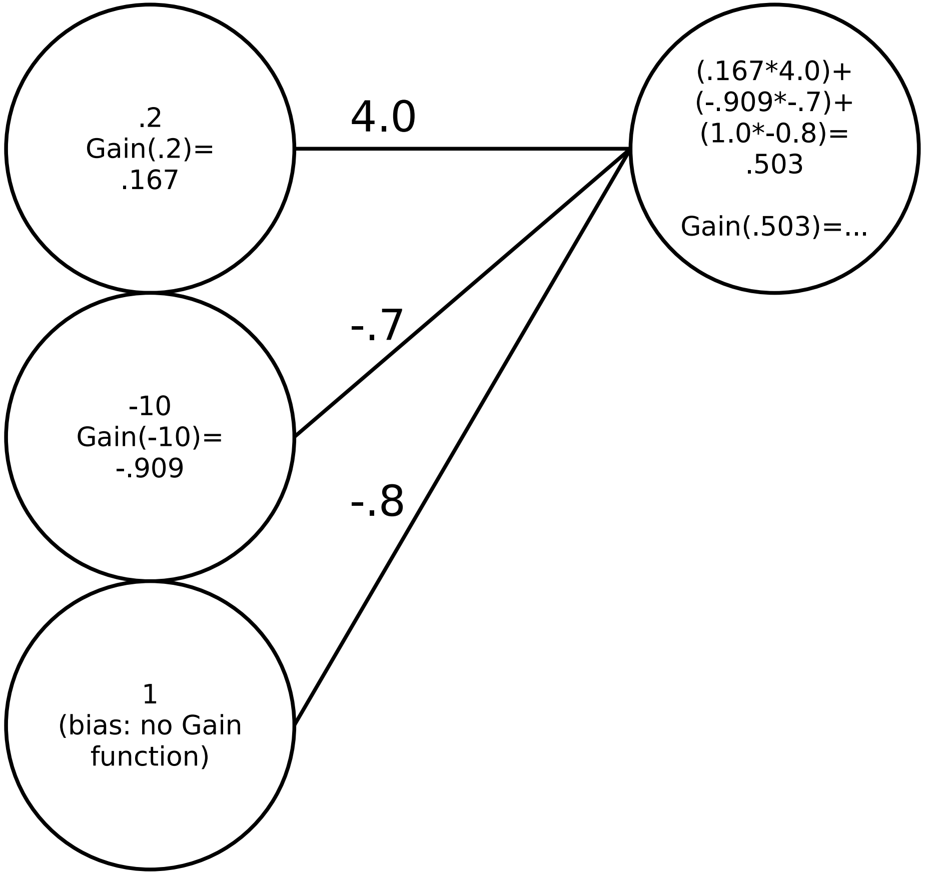
Car in maze



Car on straightway



Neural Network



.2
Gain(.2)=
.167

-10
Gain(-10)=
-.909

1
(bias: no Gain
function)

4.0

-.7

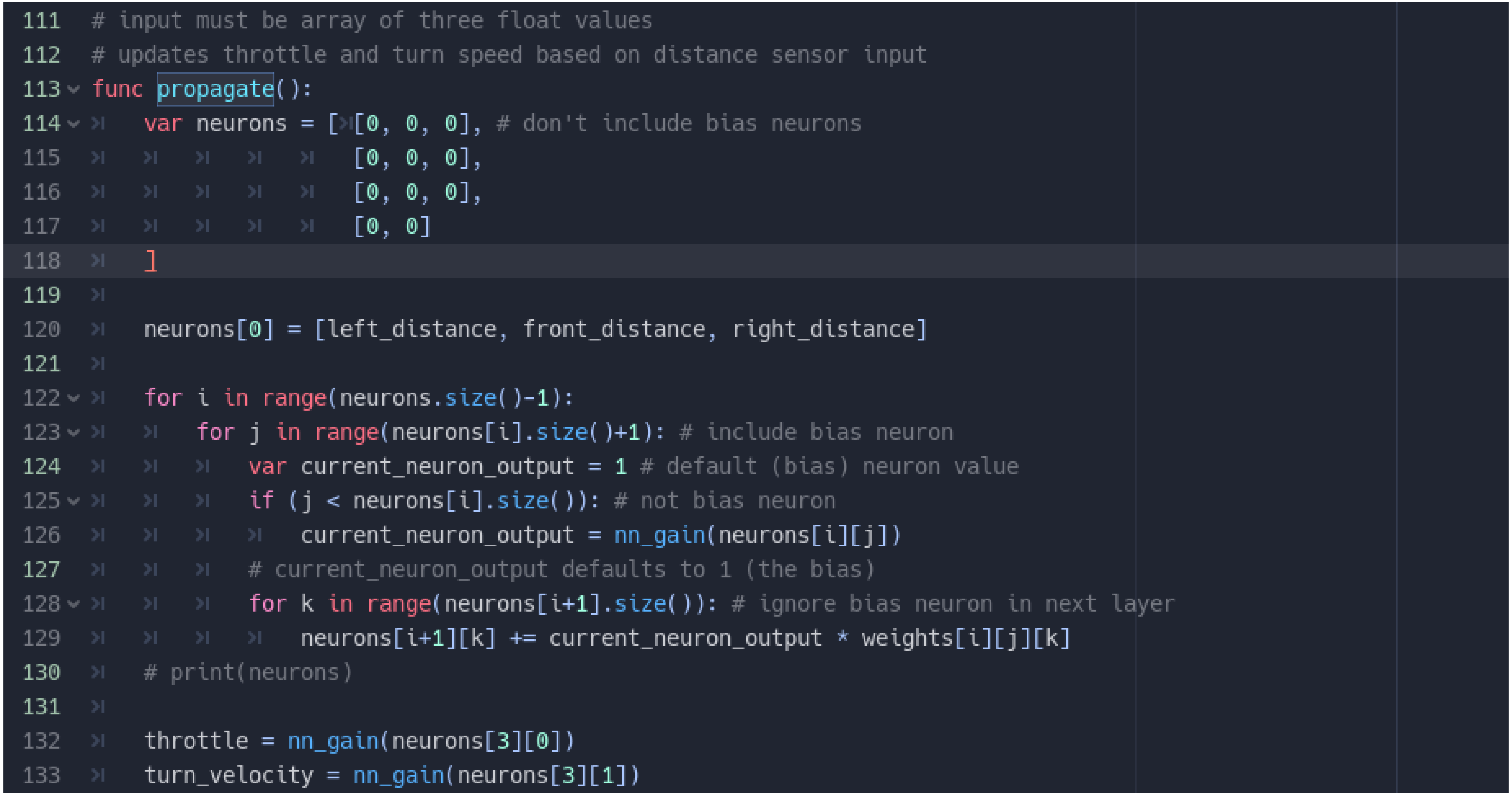
-.8

(.167*4.0)+
(-.909*-.7)+
(1.0*-0.8)=
.503

Gain(.503)=...

Gain(x) =
x / (1+|x|)

Neural Network

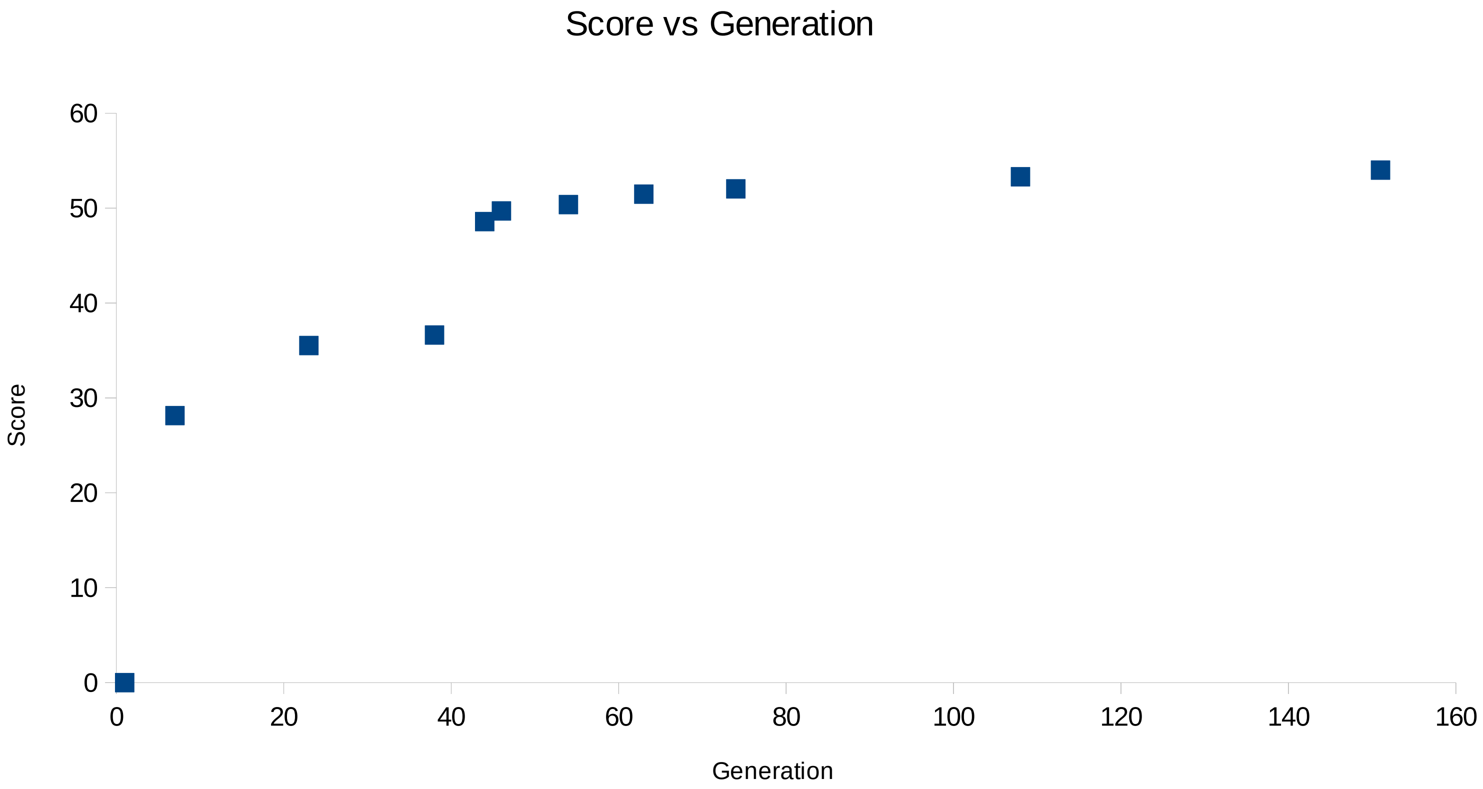


```
# input must be array of three float values
# updates throttle and turn speed based on distance sensor input
func propagate():
    var neurons = [ [0, 0, 0], # don't include bias neurons
                    [0, 0, 0],
                    [0, 0, 0],
                    [0, 0]
    ]

    neurons[0] = [left_distance, front_distance, right_distance]

    for i in range(neurons.size()-1):
        for j in range(neurons[i].size()+1): # include bias neuron
            var current_neuron_output = 1 # default (bias) neuron value
            if (j < neurons[i].size()): # not bias neuron
                current_neuron_output = nn_gain(neurons[i][j])
            # current_neuron_output defaults to 1 (the bias)
            for k in range(neurons[i+1].size()): # ignore bias neuron in next layer
                neurons[i+1][k] += current_neuron_output * weights[i][j][k]
    # print(neurons)

    throttle = nn_gain(neurons[3][0])
    turn_velocity = nn_gain(neurons[3][1])
```

Genetic Algorithm

- split time into “trials“ (10s) and “generations“ (4 trials)
- modify weights each trial (one weight per trial)
- take the best of the generation, and copy that weight set

Learning Performance (Maze)



Question time!