Reviewing the Evolutionary Approach for Obstacle Avoidance in a Mobile Robot

Gaurav Gupta (11907272/11272)

Department of Mechanical Engineering, IIT Kanpur

Abstract

In this paper, the evolution of a recurrent neural network in order to control an autonomous mobile robot is presented. Firstly, an environment containing obstacles is created, a robot embedded with IR sensors and control system is developed. The control architecture evolves in order to avoid the obstacles. The performance of the discussed method is evaluated in different environments and a wide range of system parameters.

Keywords: Autonomous Mobile Robots, System Modelling, Evolutionary Robotics

1. Introduction

The basic idea of evolutionary robotics is as follows - the control systems of the initial generations of a robot are encoded in artificial chromosomes and the population is put in an unknown environment. The performances of all such robots are defined by using a fitness score, the robots are allowed to evolve in accordance with these scores - this is essentially an attempt to replicate the biological evolution that gives rise to the survival of the fittest. This operation is repeated for many generations until an individual satisfying some predefined score is obtained or the population converges to a local optimum. The presented study is largely based on the work of Floreano, Nolfi[1] and Mondada [2]. The paper is organized into various sections. In the first section entitled 'System Modelling', the robot-environment development and interaction are discussed, thereafter the 'Control System and its Evolution' are presented. This is followed by 'Results and Discussion'. Lastly conlusions are drawn and some of the shortcomings of the method are discussed.

2. System Modelling

2.1. Robot: Kinematics

The robot under consideration is circular in shape embedded with 8 infrared sensors placed symmetrically ,it is a differentially wheeled robot. Motion of such a robot in the 2D plane can be conveniently represented in terms of speed **v** and direction θ. In terms of the angular velocity of the wheels ω_1, ω_2, this relationship can be expressed as follows -

$$v = \frac{(\omega_1 + \omega_2) * r_{wheel}}{2} \tag{1}$$

The change in the direction of the motion of the robot is given by dθ,

$$d\theta = \omega dt \tag{2}$$

Where,

$$\omega = \frac{(\omega_1 - \omega_2) * r_{wheel}}{W_{robot}} \tag{3}$$

In the above equations dt is the time for which the robot persists with the given $\omega_1, \omega_2, r_{wheel}$ is the radius of the robot wheel, W_{robot} is its wheel to wheel distance. (See *Appendix*)

2.2. Robot: Sensors

The robot consists of 8 infra-red sensors with a sensor range of 3 times the robot radius placed symmetrically along the robot circumference.
Sensor readings are generated as a result of the interaction with the obstacles within the specifie sensor range. These readings were understood to be directly proportional to the angle at which the IR rays strike the obstacle and inversely proportional to the between the sensor and obstacle boundary. Taking these two into account, the sensor reading s is modelled as follows -

$$s = \begin{cases} \frac{cos\theta}{d} & \text{if } d \leq s_{range}, -90 < \theta < 90 \\ 0, & \text{otherwise} \end{cases}$$

In the above formulation, d is the distance of the sensor from the obstacle, s_{range} is the sensor range and θ is depicted in fig.1

In the figure, A is the robot, B is the obstacle. The interaction of a single sensor with a single obstacle is shown. In case a sensor senses more than one obstacle, the largest reading is kept and others are not included.

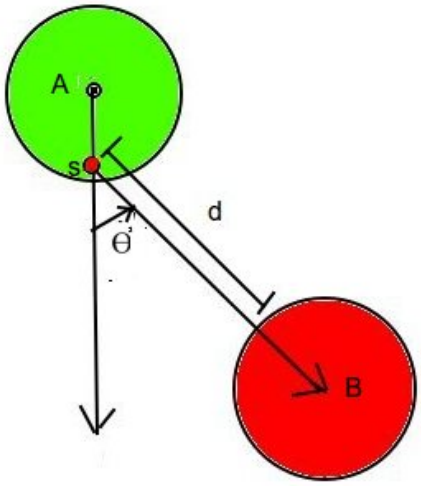


Figure 1: Interaction of sensor and obstacle

3. Control System and its Evolution

3.1. Control System Representation as a Neural Network

The control system of the robot was modelled as a neural network with an architecture as shown in fig.2. An artificial neural network is a collection of units connected by weighted links (or connection weights) used to transmit signals. The output of a unit y_i is a function ϕ of the sum of incoming signals x_j weighted by connection weight strengths w_{ij}.

$$y_i = \phi(\sum_{j}^{N} w_{ij} x_j) \tag{4}$$

Where,

$$\phi(x) = \frac{1}{1 + e^{-x}} \tag{5}$$

The control system used in the study consisted of one layer of synaptic weights from eight infrared proximity sensors to two motors unites. In adition an external bias unit was also attached to each output node, this was made to ensure that wheels have some motion even if the sensory input is zero. A set of recurrent connections was also implemented at the output unit as shown in the figure. The synaptic weight joining the j^{th} input with the i^{th} output is represented as w_{ij}. In addition, the bias weights are written as

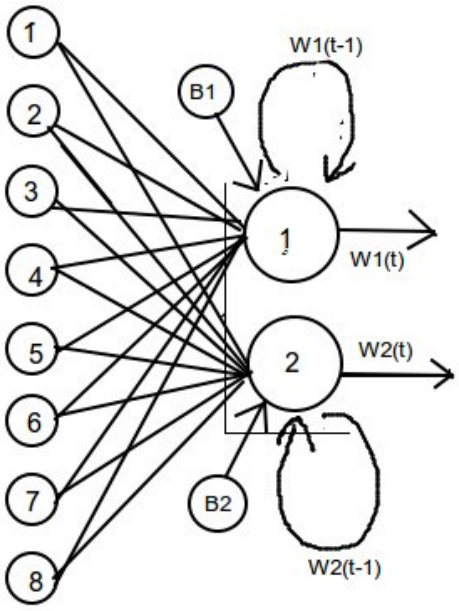


Figure 2: Control system of Robot

w_{b1} and w_{b2}. Finally the recurrent connection weights are depicted as w_{old1} and w_{old2}, these connections feed in the speed of the previous timestep in addition to the sensory and bias inputs. Each out unit was shifted in the range [-0.5,0.5] to account for both positive and negative rotation speeds.

3.2. Evolving the synaptic weights

Each syaptic weight was encoded into binary and stored in a substring of length of 10 bits giving it a range of [0, 1023]. While decoding, the value was divided by 100 and 5 was subtracted from it giving it an almost symmetric range of [-5,5.23]. Since, a total of 20 synaptic weights represent the control system of a single individual, total length of the chromosome was 200 bits. Each individual's performance was assesed for a total of $count = 1500$ sensory-motor loops, each loop lasting 0.3 seconds. If the individual bumped into an obstacle before 1500 loops a score of 0 was returned otherwise, the following score was reported (a modification of the following function will be discussed later)-

$$\phi = V(1 - \sqrt{\Delta v})(1 - i_{max}) \tag{6}$$

Where V is the sum of rotation speeds of the two wheels. This component is maximized by high rotation speed of wheels without regard to their direction of motion. Δv is the absolute value of the algebraic difference between the signed speed of the wheels. It is computed by adding 0.5 to each read value(that makes the range of each wheel speed in [0,1]), subtracting them together and taking the absolute value. Finally i_{max} is the normalized activation value of the infrared sensor with the highest activity and thus 1-i_{max} encourages obstacle avoidance.
The fitness function itself was analyzed and its effects on the robot motion were studied. The generations were evolved with single point crossover and mutation using *pyevolve* library in *Python*. The number of generations was also under consideration and required a seprate analysis.

4. Results and Discussions

A 2-D arena of size 500X500 was created using *Pygame*. The obstacles were placed around strategically to observe various patterns in robot behavior. For every generation of the evolving robot, it always began its motion at the center of the arena. The wheel velocities were assigned an initial value of zero. Robot always faced along the x-axis to begin with i.e. initial conditions were kept same for all iterations. Initial configurations was as depicted in figure 3. Further, the fitness scores(of the best individual in the generation) against the number of generations. For various parameters, the trend of the performance score remained very similar to the one shown in Fig.3.

4.1. Analysis : Number of Generations

The effect of number of generations on robot performance by evaluated. The fitness function used for this analysis was as follows -

$$\phi = V(1 - \Delta v^{1.5})(1 - {i_{max}}^2)|X_{max} - X_0| \tag{7}$$

In the above function, V plays a role similar to the one described in the previous section. $\Delta v^{1.5}$ was used to ensure that robot in not too penalised for rotation about its place. While using $\sqrt{\Delta v}$ it was observed that the robot was too hesitant to rotate as it imposed a strong penalty in the range [0,1]. Similarly, i_{max} was squared to lessen the effect of obstacle. This was done to ensure that robot does not hesitate in coming close to the obstacles. Finally, a 4^{th} term was added, in this function. $|X_{max} - X_o|$ was the maximum distance that the robot travelled from its starting position X_o at any point of

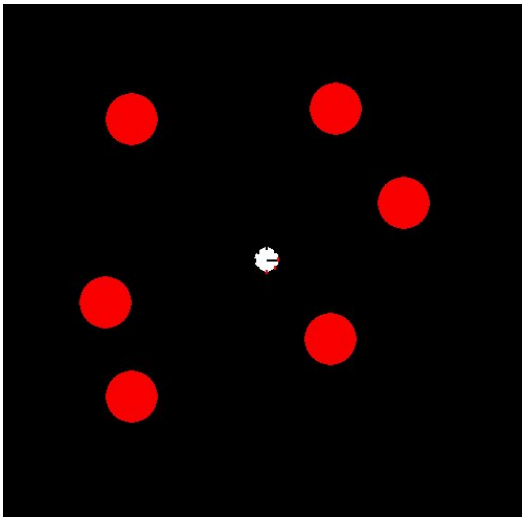


Figure 3: Initial Condition of the Robot

time during its life. This was done to get rid of the back and forth motion that the robot displayed in absence of the term. The table shows some of the parameters for which the study was conducted. 'Counts' is the number of sensory-motor loops for which the run was conducted. Fitness score represents the value of the fitness function of the best individual after the n_{th} generation. Here **n** is the number of generations for which the trial was conducted

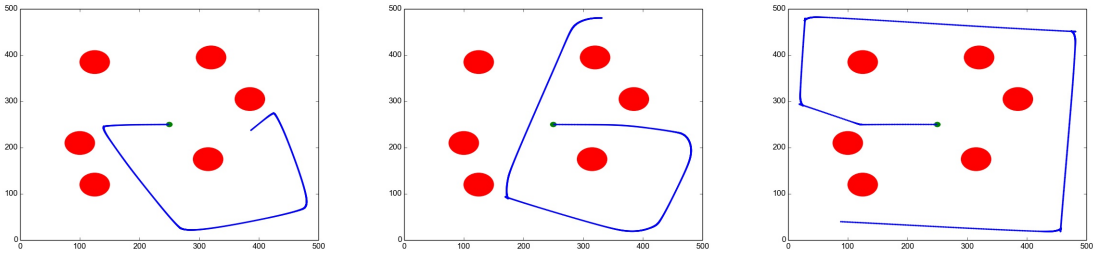
Table 1: Parameters and Observations

Fitness Function	$V(1-\Delta v^{1.5})(1-{i_{max}}^2)\lvert X_{max}-X_o\rvert$
Counts	1500
Generations	(5,15,45)
Fitness Score	(0.32, 0.35, 0.42)

The trajectory followed by the best individual after **n** generations is shown in the fig4.

4.2. Analysis: The Fitness Function

For this part of the study, the number of generations was kept constant at 20 and the count was set at 1500. The two fitness functions under consideration were described in sections 3.2 and 4.1 respectively. Fitness function



(a) After 5 generations (b) After 15 generations (c) After 45 generations

Figure 4: Trajectory of the best individual after 5,15,45 generations

introduced in section 3.2(fn^a) was proposed by [1], it was altered here(fn^b) for comparison. Fitness scores of the best individual after 20 generations and other parameters are available in the table.

Table 2: Parameters and Observations

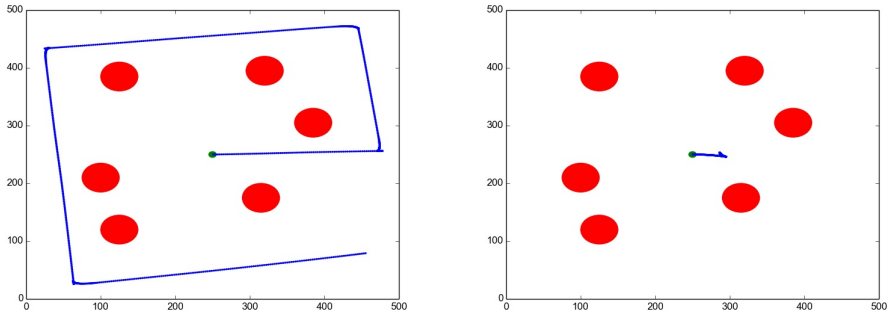
Fitness Function	$(V(1-\Delta v^{1.5})(1-{i_{max}}^2)\lvert X_{max}-X_o\rvert,\ V(1-\Delta v)(1-i_{max}))$
Counts	1500
Generations	20
Fitness Score	(0.45, 0.37)

The trajectory followed by the best individual after 20 generations for each of the two functions is shown below -

Clearly, the function described in section 3.2 fails to work here although mathematically, it achieved a fitness score of 0.37 which is comparable with the score obtained by fn^a. As shown in fig. 5b, the robot starts heading towards the right side, there it senses an obstacle but due to the heavy penalty imposed on it by the fitness function, it didn't move forward, nor did it try to rotate because rotation was also severely restricted by the function. On account of this observation, it is safe to suggest that fn^a gives better performance if all other parameters are kept same.

4.3. Analysis : Optimization Algorithm

For this part of the analysis, the optimization technique to be used for getting the best score from the fitness function was studied. First Genetic Algorithm was used for optmization purpose as described earlier. Overall



(a) Using fn^a (b) Using fn^b

Figure 5: Trajectory of the robot for fn^a, fn^b

the algorithm took 750 seconds for 50 generations. The fitness score of the best individual in each generation is shown in fig.6

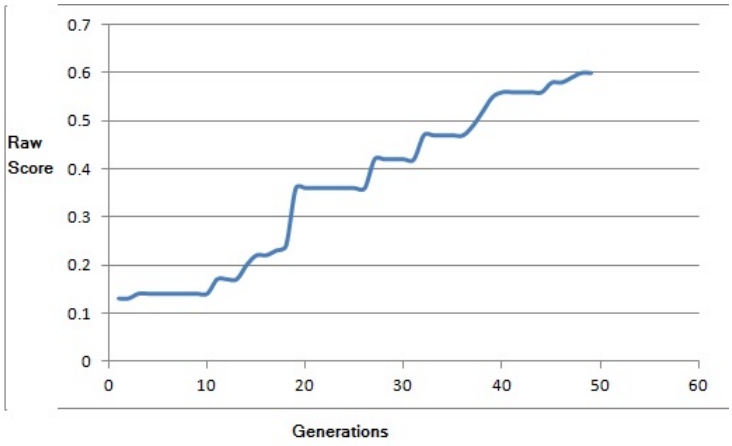


Figure 6: Fitness scores vs Generations

Among the more conventional techniques, Nelder-Mead method was chosen for its simplicity and the convenience of not determining complicated gradients, Jacobians. The algorithm was implemented using *scipy.optimize* library in python. However, the algorithm was aborted as it had a computation time of 60 minutes + for a tolerance of 0.001.

4.4. In an unknown environment

As a final part of the analysis, the robot population was developed in an environment with obstacle arrangement identical to those in the preceding subsections. The evolved weights of the best individual were then tested in an environment it was not accustomed to. The results were promising. The trajectory in this unknown environment is shown in fig.7.

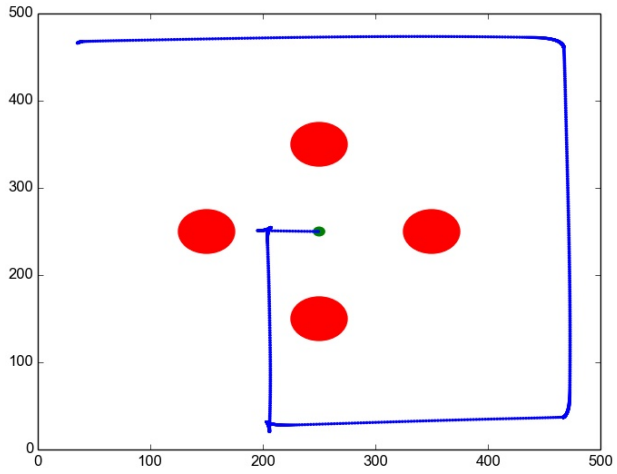


Figure 7: Behavior in Unknown Environment

5. Conclusion

Over the course of the project, an evolutionary approach for obstacle avoidance purposes was studied and implemented. The fitness function defined in the established literature was improved to suit the developed model. By varying the fitness function, the number of generations and various other parameters, some interesting behavior patterns were observed in the robot. Despite being novel and to an extent effective, the approach has some serious shortcomings. It requires a very rigorous modelling of the sensor environment. It is very difficult to generate accurate sensor readings in a simulation. Further, it is a very daunting task to implement the approach on physical robots. The complexity will only increase once the focus moves on to tasks more difficult than avoiding static obstacles.

6. References

1. Nolfi, Stefano, and Dario Floreano. "Evolutionary robotics." (2000).
2. Floreano, Dario, and Francesco Mondada. "Evolution of homing navigation in a real mobile robot." Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on 26.3 (1996): 396-407.
3. Golberg, David E. "Genetic algorithms in search, optimization, and machine learning." Addion wesley 1989 (1989).
4. Hornik, Kurt, Maxwell Stinchcombe, and Halbert White. "Multilayer feedforward networks are universal approximators." Neural networks 2.5 (1989): 359-366.

7. Appendix

Table 3: Values used thoughout the study

r_{robot}	12.5
$r_{Obstacle}$	25
r_{wheel}	20
$sensor_{range}$	75
dt	0.3