A Brief Discussion on Evolutionary Robotics

Arvid Laveno Ling

June 2, 2023

1 Context

The following discussion follows from an assigment contected to the course *TME-290, Autonomous Robots* in which the algorithm operating a robotic lawnmover is to be implemented. As such, the discussion will revolve around application of the concepts of Evolutionary Robotics (**ER**) to this problem.

The problem is formulated as; given a fixed known map (lawn), optimize a behaviour for cutting said lawn in the most efficient manner.

From this context, Evolutionary Robotics may yield improved results compared to a normal control strategy due to the underlying stochastic parameters of the problem. These being random events in the environment, such as rain and the growth rate of the grass (of course dependent of the rain). Rain appear causal with other phenomanon in nature, but is in general to complex to model deteministicly. Thus, we have good justifactions for the hypotheses that one can fit a regression to the environmental data in this application, i.e, grass height and rain intensity, etc. Which would mean that an Evolutionary Algorithm would be capable of infering good control decisions.

2 Implementation

For the particular implementation of an Evolutionary Algorithm for our problem, I will propose a neural based design. Again, this follows from the proposed correlation in rain data and grass growth, aswell as the underlying causality in the rain data itself.

Given some arbitrary network with a set depth and width we would set up our algorithm as a large population of permutations of initializations of said network. These variations would then be allowed to act in the simulated environment, provided in the assignment.

The outputs of the network would be the commands necessary to control the robot and are therefore relativly set. Similarly, we should design the input layer such that it can recieve all the availible data from the onboard sensors. However, we should also consider allowing the algorithm to know its current position. The map contains constrained zones where the robot can not move. Intuitivly, it seems that this information would be benificial for the performance of the algorithm. One should test both options.

2.1 Fitness Function

With the architecture covered, we move on to the evaluation of the performance of each individual in order to make a good selection for propagation. In our, case we have a main objective; *Cut the lawn as efficient as possible. That is, cut the most amount of units of grass per unit of time.* And an implied objective; *Never run out of battery.* A robot that ran out of energy mid-mission, would get an awful long term cutting-score. One proposition for fitness function could be:

$$f(t;x) = \frac{1}{d}\int_0^{t_d} dt x(t) \tag{1}$$

where $x(t)$ is the average of cut grass over time which been avaraged over the total distance traveled d. Some literature seems to refer to this class of formulation as standard [**NELSON2009345**], however my interpretation may not be. I would propose to use:

$$f(t;x,d) = \sum g(d) + x(t) \tag{2}$$

Here, the distance traveled is squished into some nice function that makes it comparable with the avarge cutting over time. In this case a low score would be best. The function g can vary depending on how we would like to emphasis the relative importance between each parameter. For instance, the function:

$$g(d) = d_0 \exp[-\gamma d] \tag{3}$$

with $0 < d_0, \gamma < 1$, would prioritize genetic combinations with long survival time, i.e, long distance traveled, in earlier generation. Meanwhile for long term performance, due to the tail of the exponential, improvement in cutting efficiency would be valued more when long distances are already possible.

If one were to find even greater urgency of achiving long distance performance a linear version of g could also be good. However, one should in this case ensure that the chosen function remains positive in the limit.

If the lawnmover still tends to run out of battery, one could also further modify the fitness function by for instance penalizing low avarage batteri levels during the run.

3 Common Pitfalls

Due to the importance of the fitness function, the most common issue with evolutionary training is that the function output plataues too early before good performance is achieved. This is why testing multiple functions are important.

Further more, the standard problem in machine learning remains, that is the genetic algorithm finding a optimal set of weights for its neurons that happens to be a local minima, not the actual global minima. A possible work around for this, specific to genetic algoritms, is to introduce a probability of neurons mutating between generations. Hopefully, this can cause the system to perturb from a local minima, if a global minima exists.