COMPUTER PROGRAMMING

Genetic algorithms for autonomous robot navigation

Submitted to:

Sir Shujat Ali

Submitted by:

Syed Zain-ul-Aabideen(2021_MC_86)

Content Table

Introduction

The GA is an evolutionary meta-heuristic that belongs to the broad family of evolutionary algorithms in informatics and computational mathematics. These algorithms are widely employed to provide high-quality solutions for optimization and search problems by concentrating on bio-inspired operators such as selection, convergence, or mutations. This algorithm is classified as an evolutionary algorithm. The evolutionary algorithms are used to tackle issues that do not have a well-defined solution. This method is utilised to solve optimization issues (such as scheduling and shortest path), as well as in modelling and simulation where a randomization function is employed. GA is a solution to the problem of optimising that is evolved towards better alternatives by populating the candidate (known as people, animals, or genotypes). Every candidate's solution contains a set of traits (the genes or phenotype) that may be developed and altered; solutions are commonly represented in binary digits as strings of 0s and 1s, however another codec is permitted. Evolution typically begins with a population of randomised individuals and is an iterative process, with the population considered as a means of generation for each reproduction. The fitness of everyone in the population is measured for each generation. The fitness, on the other hand, is generally the value of the goal feature being addressed. When a suitably fit person is picked at random from the existing population, the gene is updated to generate a new generation cycle for everyone (re-combined and perhaps altered at random). Over the following generation of the process, a fresh generation of candidate tactics would be used. Every subsequent generation is more suited to the population's conditions. An individual's ideal fitness score might be discovered.

Operators

In GA, operators are utilised. When the initial generation is formed, the GA develops it using the operators described below.

Selection Operator:

Individuals with higher fitness scores are preferred in this type of operator, allowing them to pass on their gene to the individual's subsequent generation.

Crossover Operator:

This indicates a generation of breeding between people. Using selection and crossover operators, choose two individuals at random. After being selected, genes are transferred at the crossover sites, creating a completely new progeny or person.

Mutation Operator

To utilise this method, insert random genes into kids to preserve genetic heterogeneity and avoid excessive divergences.

Genetic algorithm

Genetic algorithm It is a machine learning paradigm that develops behaviour patterns based on the representation of evolution mechanisms. It is accomplished by generating a population of entities identifiable by genetics on the interior of the machine. People in the population will go through a mutation period. It should be mentioned that development will not be aided in any way. As a result, there is no evidence to support the claim that the goal of evolution is to produce humans. As a result, the mechanisms of existence appear to be dispersed among many Persons contending for services in the World. The steps below are used to obtain fitness using GA.

1. Consider populations p at random.

2. Determine the population's fitness.

3. Repeat steps 4-7 until convergence is reached.

4. Pick any parent from the population at random.

5. Through the crossover process, creates a new population.

6. In order to accomplish mutation, insert random genes into a new population.

7. Determine the fitness of freshly produced populations.

Genetic programming

The major use of GA is genetic programming. A GA result is a quantity, but a genetic programming result is a virtual machine operation. This is essentially the start of the algorithms that run automatically. However, many experts believe that genetic programming and GAs are completely separate entities based on their characteristics. Furthermore, because it is genetically inspired, genetic programming is fundamentally separate from other methodologies to artificial intelligence, machine learning, neural networks, evolutionary structures, deep learning, or computational reasoning. ultimately Genetic Programming assists machines in finding answers without the need for programmers to determine the approach, i.e. what can be done. That achieves the goal of intelligent coding by genetically changing a population of automated systems based on Natural selection, Evolutionary theory, or biologically inspired activities. Genetic programming is well suited to a wide range of challenges.

Follow the steps below to begin adopting GA:

1. Create a GA Configuration.

2. Recognize genes and chromosomes.

3. Use the Fitness Method.

4. Create a terminating state.

5. Population evolution.

GA's Role in Emerging Areas

1.In Engineering Education

2. the Internet of Things

3.Intelligent Routing in MANET of Smart Devices

4.Smart Traffic Signal System

The application that we were supposed to develop for our project is as follows:

Description of the functions

void rand_gen():

This function is generating the random population for genetic coding.

void path_planning():

This function Is playing a crucial role in genetic of autonomous robot as it stores the path length along with number of steps taken and infeasible steps.

void Bubble sorting():

This function allows us to do the parent selection.

void mutation_fn ():

This function is made for mutation.

void cross_over_fn():

This function is doing cross over, this all helps us to achieve a better and healthy solution.

void Display_Sol():

This function allows us to display our robot path on screen.

void maxval():

Again, this function is only comparing the values stored in array to achieve the highest and the lowest value stored for each generation.

void Fitness ():

After having all the parameters required to calculate the fitness of the path generated, we are putting all the parameters in the fitness formulas for the robot navigation.

int check_sol():

This function is comparing the fitness of all the generations to get the fittest of the fittest path. It will stop as soon as it gets the fitness value up to 300.

References

https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3