Title:
A CAD Path and Motion Planning System for Flexible Agents using a Genetic Algorithm

Authors:
Miri Weiss Cohen, miri@braude.ac.il, Braude College of Engineering
Sergei Kovalchuk, sergeikov95@gmail.com, Braude College of Engineering

Keywords:
Path planning, Motion planning, Genetic Algorithm

DOI: 10.14733/cadconfP.2019.387-391

Introduction:

Path and motion planning for animated characters in 3D virtual environments plays an important role in a large variety of applications, ranging from creative virtual world to to design robotic path and motion planning[4]. For efficient and effective path and motion planning, planners must consider a number of factors, among them the nature of the environment itself and the specific agent characteristics. They must also seek to minimize costs as measured, in this work by energy consumed. Virtual environments vary widely and can range from static smooth environments to dynamic discontinuous constrained terrain. The problem is to find the least expensive path from point A (Start) to point B (Goal), while adapting it to the environmental constraints. Hence, planners must have the capability to adapt flexible path and motion to various terrain settings. Another consideration is the nature of the agents specifics. Agents differ in size, gait and step features, thus requiring different path and motion planning considerations. For example, an agent representing David would follow a different path and move differently than a agent representing Goliath, especially in the case of discontinuous terrain. Another aspect of path and motion planning is cost, where energy=cost. Different steps, even for the same character, have different defined costs and thus consume a different amount of energy.

The above constraints indicate that a composite solution is required for path and motion planning. Considering only one factor without the others would yield a solution that is not adaptive. More precisely, the resulting path would not meet the motion planning and minimal cost requirements and in some cases would not be realizable for the particular character.

Main Idea:

In this study, we propose a new approach that incorporates the three above factors: motion planning, the specific nature of each character, and flexibility of cost in defining each step. This parallel approach considers all three parameters simultaneously in the search procedure. The goal is implemented by means of a genetic algorithm (GA) that incorporates all the above factors while searching for the best solution. Because path and motion planning is an integral part of the animated procedure, it must be time efficient, while at the same time searching for the best solution. The proposed approach meets both these constraints.

Figure **??**:fig1 illustrates the problem. In the figure, the goal appears as an orange disk close to the viewer, and three possible paths and motions lead to the goal. Paths A and B reach the goal because the

characters' steps are sufficiently large to overcome the discontinuity in the terrain constraint, while Path C is unable to reach the goal. Although the characters moving along Path A and Path B both reach the goal, these paths may have different costs as the characters' steps are defined differently in terms of cost. For example, in both cases, the paths may have the same length but the characters' steps are combined differently, resulting in different costs. The proposed system is sufficiently flexible to handle a wide range of cases. It provides the user a huge range of capabilities and parameters for defining characters and 3D terrains.

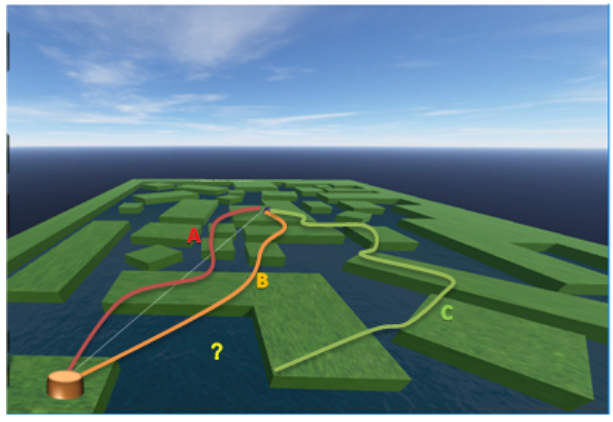


Fig. 1: Problem Definition

The motivation for using Genetic Algorithm (GA) is its advantage as a stochastic search algorithm and a problem-solving methodology [1]. It has advantages such as flexibility, adaptation, global search capability, and its suitability for parallel computation. GAs have been used to solve difficult problems with objective functions that are multi-model and multi-constrained. A wide range of genetic operators can be used to perform the crossover, mutations, and reinsertion of potential solutions, among others, to generate the offspring's population.

The first stage entailed encoding the solution. The goal in this encoding is to calculate the type of steps (allowing repetition) and angles necessary for the animated character to reach its destination. In order to solve this non-trivial NP-hard problem, the solutions were encoded into the chromosomes of the genetic algorithm using the value encoding method, where the values for the chromosome strings take the form of pairs (#Step, angle). A path is made up of steps, where each Step S_i has its own energy cost Si_{Energy}, calculated according the data values assigned to the motion of the step. The steps are flexible and the user can define them easily using the system's GUI. The Si_{Energy} costs are modeled by user definitions. The second stage was to determine the evaluation function. This required first calculating the general cost of the path (PathEnergy) by summing all the steps in the path:

$$PathEnergy = \sum_{s_i \in Path-Steps} Si_{Energy}$$

$$Si_{Energy} = \frac{Si_{Length}}{Si_{Length}} \cdot Sl_{Energy} - \epsilon$$

Each solution is evaluated by the `PathScore` of the calculated path. `PathScore` comprises the variables `PathEnergy` and `PathRemainingDistance`, each of which has a weight factor to determine how it influences fitness.

$$PathScore = \frac{1}{LenFactor \cdot (PathRemainigDistance + 1) + EnergyFactor \cdot PathEnergy}$$

where: `LenFactor` - defines the importance of the `PathRemainingDistance` parameter to the fitness. As `LenFactor` becomes larger, `PathRemainingDistance` becomes more significant in calculating the fitness. These leads to the selection of solutions that arrive closer to the target at the expense of path efficiency considerations. `EnergyFactor` - determines the significance of `PathEnergy`, that is, the importance of path constraints and optimality. These two factors help users control their preferred method of searching for the motion planning fit for their best goal, subjected to constraints.

Algorithm 1 - GA Path and Motion Planner

Input: initial position S_{pos}, goal position G_{pos} , list of possible movements $M_{1\cdots n}$.

Output: path Solution between S_{pos} and G_{pos} that consists of a sequence of movements from $M_{1\cdots n}$,and angles.

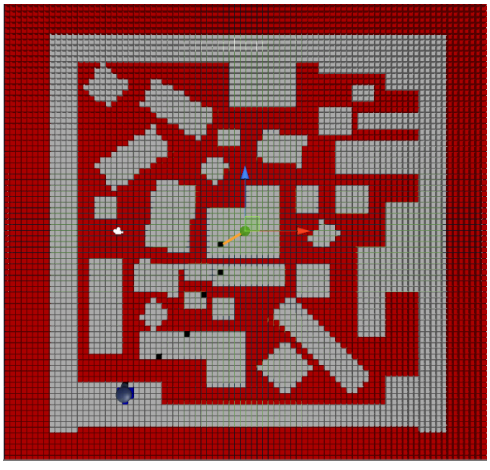
```
1: procedure GAPATH(S_pos, G_pos, M_1...n)
2:     Population ← GenRandPop(M_1...n)
3:     while pathFound == false do
4:         for all Path p in Population do          ▷ Execute sequence angles of path
5:             RunPath(p, S_pos)                    ▷ E_pos path coordinates
6:             p.RemainigDistance = Vector3.Distance(G_pos)
7:             for i = 1:pathLength do
8:                 p.PathEnergy += p.step[i].Energy
9:             end for                  ▷ CollisionPenatly calculated RunPath if collision in path
10:            p.PathEnergy += p.CollisionPenalty
11:            if p.remainingDistance < 1 & p.PathEnergy < Threshold &
    p.fitness < bestFitness then
12:                pathFound = true
13:                Solution = p
14:            end if
15:            Population = GenNewPop(Population,S_pos, G_pos)
16:        end for
17:        newSolution = RunGenerations(Population)
18:    end while
19: end procedure
```

Case Study Results - World-1:

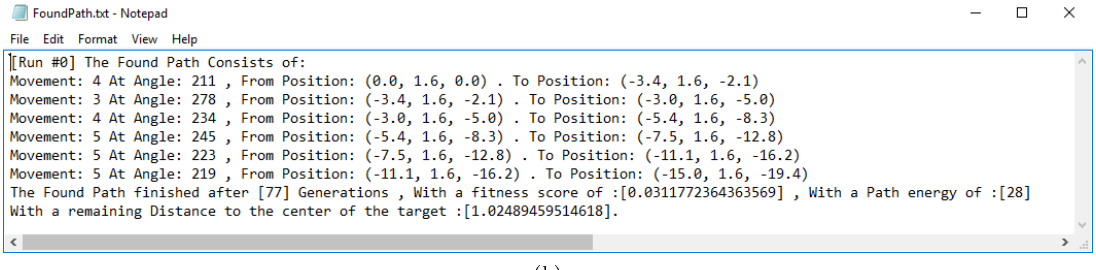
Simulations 1-3 represent results that differ in step cost, yielding different paths and representing different characteristic motions. Simulation 1: Figure 3a and Figure 3b depict the results of the first simulation. In this simulation, the user required a path constructed from larger steps. To this end, the user imposed the restriction of using only steps of 3,4,5 by stipulating that the values of energy cost in steps 1 and 2 be respectively high. The actual values the user defined for the five steps were 80, 90, 4.5, 6, and 7.5, respectively.

Figure 3a depicts a 2D pixelated world that clearly show the resulting path and motion planning. All the steps from Start to Goal were taken from the stipulated group of steps (3,4,5). Figure 3b shows the list of steps and their exact positions and angles. Because this world is defined as 2.5D, all heights

Fig. 2: Snapshot of the GUI where the designer can indicate parameters, for example in this virtual world-1.



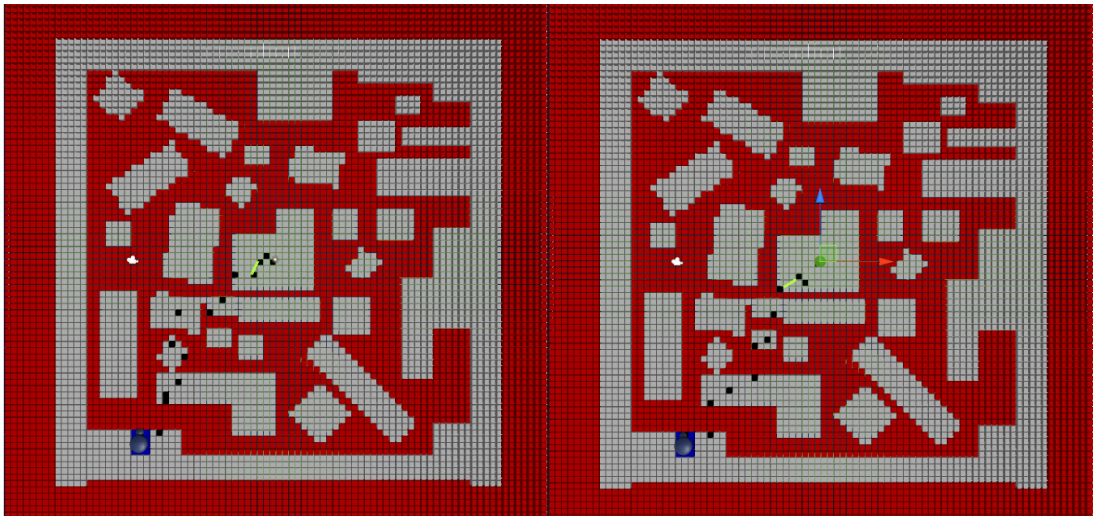
(a)



(b)

Fig. 3: World1 path and motion solution,with step restrictions of 3,4 and 5

are the same, while the user defined flexible Y coordinate as 1.6. The lower part of the window shows the calculated parameters: fitness score and path energy (cost). Based on these stipulations, the search strives to find a balance between number of generations (computing time) and cost of the path found.



(a) (b)

Fig. 4: Simulations 2,3, where(a) and (b) the path and motion planning results

Figure 4a, 4b depict the results of path and motion planning for cases in which the user imposed restrictions by giving preference to steps 1, 2 and 3. The two examples also assign different energy costs for these steps, resulting in different paths and motions. These two examples represent animations for different energy costs for the steps. The path indicated in Figure 4a result the cost of 37.6 units, and figure 4b, results 34.6.

Conclusions:

The approach uses the genetic algorithm method together with flexible parameters and variables adapted to different environments and characters. Motions are constrained by two parameters: 1) terrain, as tested on discontinuous cases, and 2) different kinds of steps, depicted by length, energy and cost. The user can adjust the value of each of these parameter values as required.

References:

[1] Eiben, A.-E.; Smith, J.-E.: Evolutionary Computing, In Introduction to Evolutionary Computing, 2015, Springer, Berlin, Heidelberg

[2] Mac, T.-T.; Copot, C.; Tran, D.-T.; De Keyser, R.: Heuristic approaches in robot path planning: A survey, Robotics and Autonomous Systems, 86, 2016, 13-28. https://doi.org/10.1016/j.robot.2016.08.001

[3] Norman, J.; Atlas C.; Roland, G.: Real-time path planning in heterogeneous environments, Computer Animation and Virtual Worlds, 24 (3), 2013, 285-295. https://doi.DOI:10.1002/cav.1511

[4] Yang, L.; Qi, J.; Song, D.; Xiao, J.; Han, J.; Xia, Y.: Survey of robot 3D path planning algorithms, Journal of Control Science and Engineering, 16, 2016, 5-24. https://doi.org/10.1155/2016/7426913