An artificial life approach for the animation of cognitive characters

Lavish Gulati Vakul Gupta Chirag Gupta Kartik Gupta

CS461 - Computer Graphics

November 10, 2020

Abstract

The project addresses the problem of cognitive character animation. Finite state machines are used for the behavioral control of characters. Our approach rests on the idea that the cognitive character arises from the evolutionary computation embedded in the artificial life simulation, which in our case is implemented by the finite state machine. This project to build virtual worlds is aimed at the graphic simulation of an arena, where a small mobile robot can perform requested tasks while behaving according to their own motivation and reasoning. Each robot is an intelligent agent that perceives the virtual environment through a simulated vision system and reacts by moving away from or approaching the object it sees.

Contents

1 Introduction

1.1 Artificial Life

Artificial life is a term originally intended to mean the simulation of macroscopic behavioral aspects of living beings using microscopically simple components [1]. However, the term spanned to designate a wide variety of simulated living creatures, including virtual characters whose behavior emerges from hierarchical and functionally specialized complex structures like an animal's body [2]. Both kinds of simulations are very interesting from the computational point of view: they offer very attractive means to computationally model complex behavior, a subject that has gained a special relevance under both the theoretical and the applied standpoints. **Artificial life** worlds are computational simulations like virtual places where animated characters interact with the environment and with other virtual beings of the same or distinct categories. Different levels of sophistication can be found in these virtual creatures, from unicellular life with minimalist models to complex animals with detailed bio-mechanical models.

1.2 Cognitive Animation

Artificial intelligence techniques are mixed with evolutionary computation and computer graphics to generate a computer **animated character** capable of acting in response to a storyboard or screenplay and displaying naturalism when performing other tasks not explicitly specified. The animated character displays behavior that emerges from the dynamics of a complex system (for instance, systems with learning, reasoning, ontology, cognitive processes, etc.) In the history of computer animation, characters had been first animated using geometrical and physical rules and constraints. New tendencies point to behavioral animation, where characters have some degree of autonomy to decide their actions.

1.3 Finite State Machine

A **Finite State Machine**, or FSM, is a computation model that can be used to simulate sequential logic, or, in other words, to represent and control execution flow. Only one single state of this machine can be active at the same time. It means the machine has to transition from one state to another in to perform different actions. A Finite State Machine is any device storing the state of something at a given time. The state will change based on inputs, providing the resulting output for the implemented changes. It can be used to model problems in many fields, including mathematics, artificial intelligence, games or linguistics.

1.4 Aim of the project

Following the track to **cognitive animation**, the project exploits the capability of several methodologies and techniques that employs adaptive algorithms and evolutionary programming. We first build an artificial environment, the **ARENA** for animated virtual creatures. Our goal is to obtain virtual creatures capable of performing specific tasks in their environment by exploring certain strategies and adapting those whenever necessary. The ARENA robot - called **WOXBOT** has a vision system consisting of a simulated camera and an image processing algorithm that classify the visual patterns to provide input to a motor system controlled by a deterministic finite state machine (FSM). This FSM is an automaton obtained from an optimization procedure implemented with a **genetic algorithm**, and applied through generations of robots.

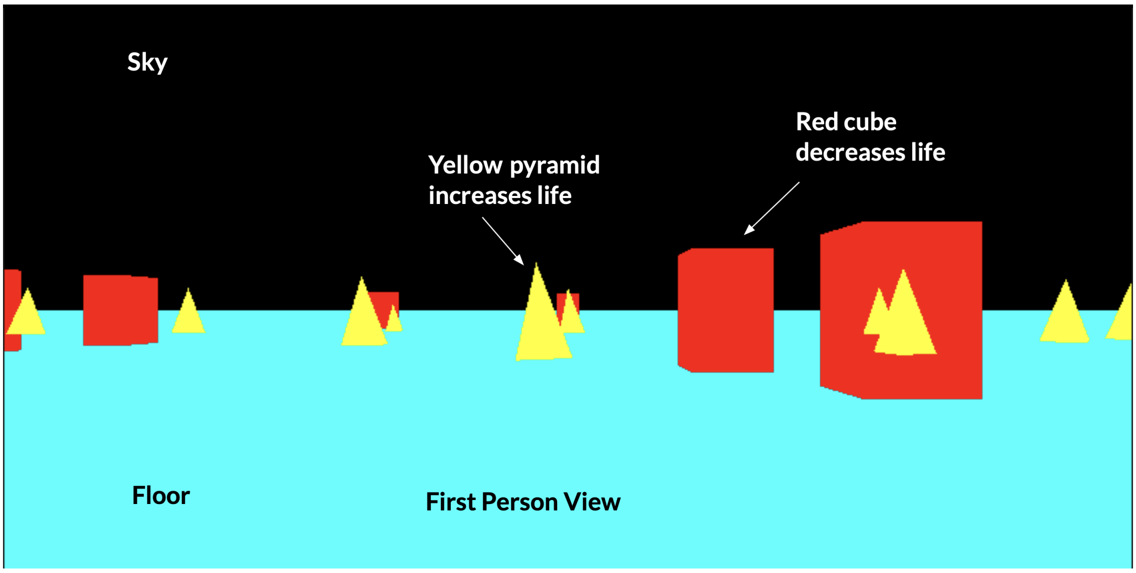
1.5 Intelligent Agents

Intelligent agents may be understood as computational entities that behave with autonomy in order to manipulate the information associated with its knowledge. Our robots also exhibit "life" features that allow their classification as intelligent agents. In order to be autonomous and to present intelligent behavior these agents need to be able to sense the world, by getting information about the environment and creating their own representation of this world. Furthermore, they should be able to analyze this information, to use it to increase their knowledge and to reason about actions to be taken. Finally, they must be able to express these decisions through actions such as movement, communication, etc.

2 Implementation

2.1 Sensing and Perception (Pattern Recognition)

One of the tasks that WOXBOT has to perform in this first project is to be aware of nutrients (yellow pyramids) and hurting entities (red cubes) that are present in ARENA. The visual information is gathered from the 3D surrounding scene by projecting it to a view port with 3 color channels, namely R (red), G (green) and B (blue). To differentiate and spatially locate these entities, we have designed an algorithm that takes input the visual feed taken from the robot and outputs the location of target colour in the view according to the intensities of the color in various regions.



2.2 Action and behavior: learning and reasoning

The WOXBOT character is an intelligent agent with a simulated visual sensor to pick images of the environment from its point of observation. The image processor inside the agent analyze these images classifying the visual patterns. These outputs are tokens fed into the agent control system, which is an FSM. Based on the above codes, the FSM chooses an action from its repertory. It is set initially with a random structure that is improved based on evolutionary computation concepts.

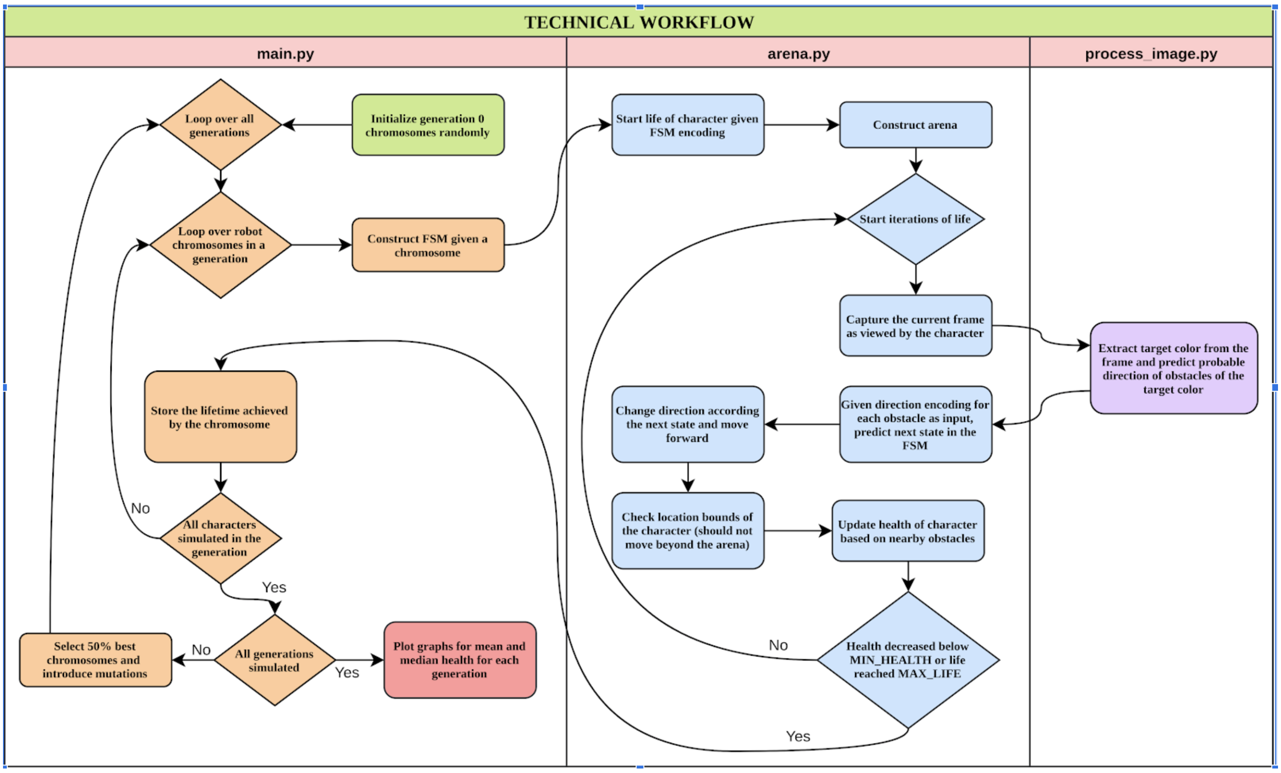
2.3 Agent Chromosome

The FSM is represented by a string of bits coding its states, inputs and actions. This string is named the WOXBOT chromosome. For each FSM state there is a chromosome section. All these sections have the same structure: for each of the 16 possible inputs, there is an entry on the chromosome state section, composed by code of the action taken upon the given input.
Chromosome example: **10**10**10**01**10**11**11**10**01**01**10**00**11**11**01**10

2.4 Evolutionary Process: Survival of the fittest and mutations

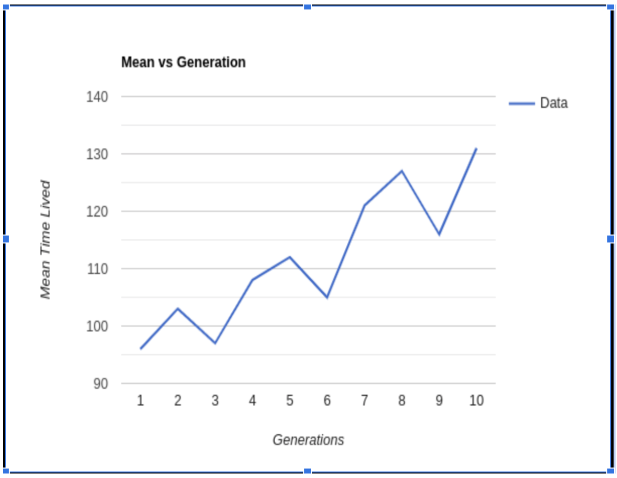
At the start of the evolutionary process, a population of WOXBOTS is generated by sampling chromosomes at random from a uniform distribution. This population constitutes the first generation. Each member of the current generation is put into the ARENA, and it is assigned a performance value proportional to the duration of its life. After each generation of robots in the ARENA, the fittest ones are selected to be the parents of the next generation and their chromosomes are grouped in pairs. The best chromosomes (Top 50% chromosomes with the highest life) are selected which undergo mutations. This constitutes part of the next generation. This procedure follows, giving thus more generations, until there is a stabilization of the fitness value of the population.

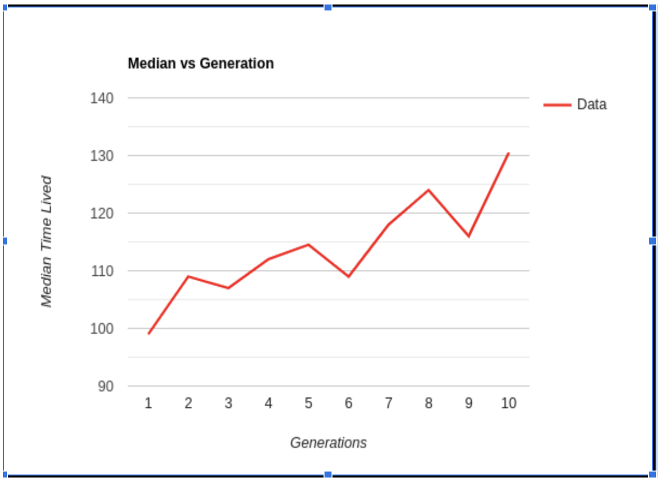
2.5 Technical Workflow



2.6 Results

These are the results corresponding to the case when there are only two types of obstacles: Yellow Pyramid and Red Cube. Here we can notice than both mean and median generally increases over the generations and becomes better till we reach the last(10th) generation.





3 Challenges

3.1 Not encountering obstacles

During the early stage of the project, we noticed that some robots were not encountering obstacles and roaming in the free area of the arena which lead to health neither decreasing nor increasing and hence life becoming equal to MAXLIFE. In real life, we constantly need to feed onto resources to survive and thrive. If we do not feed onto resources, health will decrease constantly. To overcome this problem, we constantly decrease health of the robot by a fixed amount at each iteration of the life so that the robot rarely reaches it's MAXLIFE.

3.2 Respawning obstacles

The issue we were facing is that once a robot encountered a yellow pyramid(which increases the health), some of the robots started roaming around that in order to increase their life and thus not exploring other areas of the arena. In real life, resources don't remain fixed at same point in space and might vary according to time. The solution to this problem is that we constantly re spawn the obstacles in the arena whenever that obstacle is encountered by the robot.

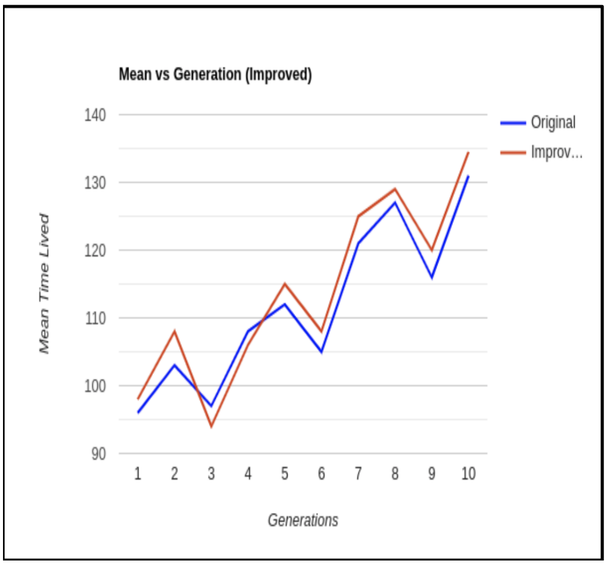
3.3 Other challenges

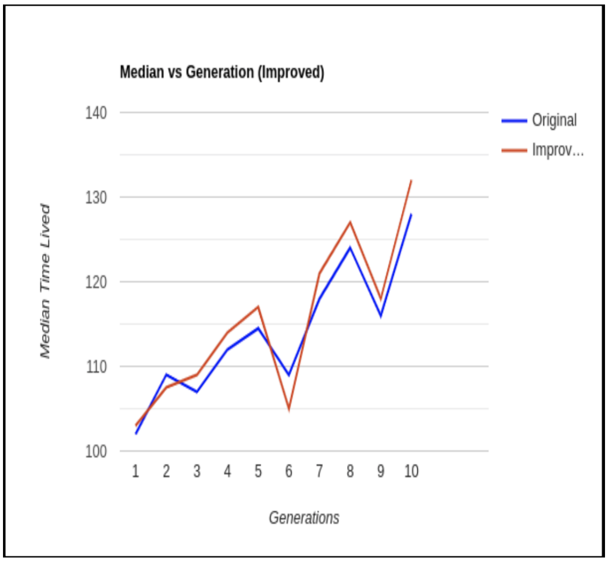
- As we are capturing the view of robot at each iteration, this leads to heavy computation and hence is the bottleneck of our program.
- Training time for the whole algorithm was around 3-4 hours. SOLUTION: Multiprocessing can be used to optimize training time.

4 Improvements

4.1 Expanding action set of the character

In the original implementation, the woxbot could only perform 4 actions based on the input (Straight, Turn Left, Turn Right and Backwards) In our improved implementation, we've added two more actions that can be performed by the woxbot i.e Strafe Left and Strafe Right. With this, we are getting better results in terms of average life lived by the woxbot generations.

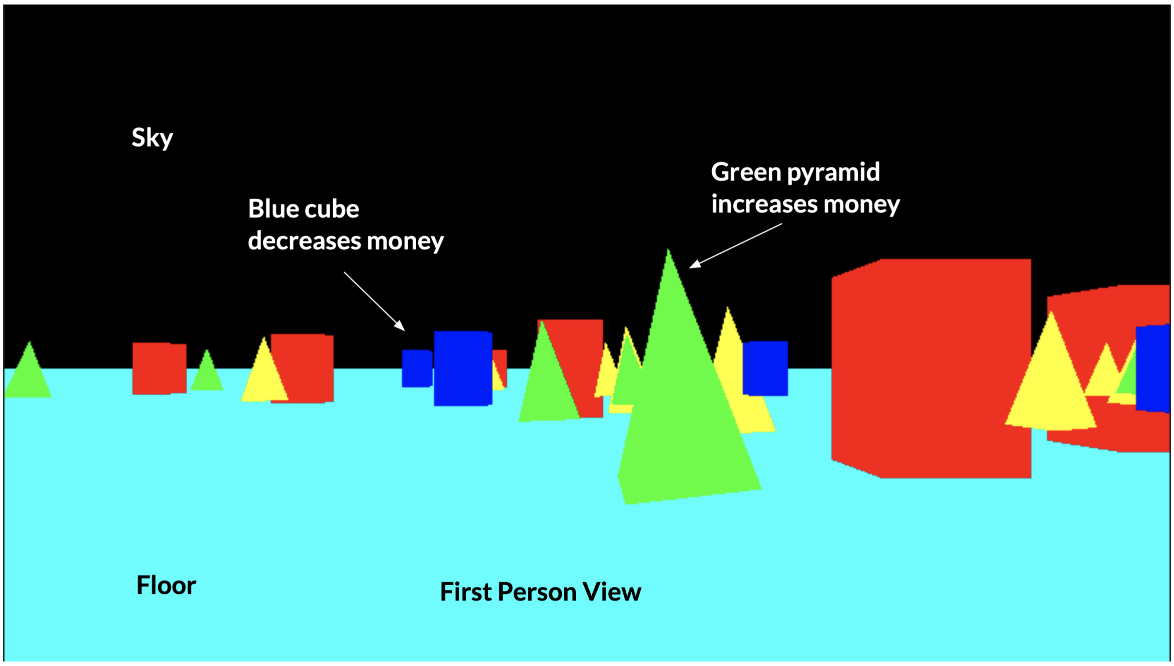




4.2 Introducing more performance factors

In the original implementation, the performance evaluation was done only on the basis of number of iterations performed (Life of WoxBot) In our improved implementation, we introduce a new parameter "MONEY" and calculates the performance on different functions of health and money throughout its lifetime. So the WoxBot will now have to look for a balanced amount between MONEY and HEALTH to extend its viability. We have used the following performance measures:

- Max Health achieved
- Max Money achieved
- Linear combination of Health and Money
- MaxHealth * MaxMoney



4.3 Applications

This scenario can be useful for many purposes: for behavior modeling, as a laboratory of learning algorithms, for research on societies of virtual characters, in the study of collective dynamics of populations, etc. As an example of application, a given research project can be targeting visual recognition methods, so it will be using WOXBOT as a subject employing the methods under test, and will take ARENA as a laboratory for evaluation experiments. Another research project instead, can target autonomous robot traveling strategies. In this case, the ARENA floor will be the field of traveling, and WOXBOTS will be simulating the autonomous vehicles. Yet the WOXBOT could be reshaped to have the aspect of a given car or truck model and the ARENA could be configured to resemble a street, in a project intended to analyze vehicle traffic conditions.

5 References

[1] Bonabeau EW, Therulaz G. Why do we need artificial life? In: Langton CG, editor. Artificial life an overview. Cambridge, MA: MIT Press, 1995.

[2] Terzopoulos D. (org.). Artificial life for graphics in animation, multimedia and virtual reality. In: ACM/SIGGRAPH 98 Course Notes 22, August 1998.

[3] Adami C. Introduction to artificial life. Berlin: Springer, 1998.

[4] Bentley PJ, editor. Evolutionary design by computers. Loss Altos, CA: Morgan Kaufmann, 1999.

[5] Fogel DB. Evolutionary computationFtoward a new philosophy of machine intelligence. New York: IEEE Press, 1995.

[6] Phillips C, Badler NI. Jack: a toolkit for manipulating particulated figures. In: ACM/SIGGRAPH Symposium on User Interface Software, Banff, Canada, October 1988.

[7] Badler NI, et al. Simulating humans: computer graphics, animation and control. Oxford: Oxford University Press, 1992.