Project Specification: Silverfish Simulation

Introduction

Simulation is related to another subject that I'm interested in, which is game programming. They're related because I find games that aren't completely predefined by a programmer more interesting. Basically what I love is typical games combined with some form of simulation.

Problem Statement

The problem is to simulate evolution. In this case we have Silverfish that a person is supposed to locate with a flashlight and try to kill with his foot. To make it successful I will have to rely on Natural Selection to evolve the Silverfish to avoid getting killed by the shoe. The flashlight and shoe should be controlled by an interface for the user. I also want to add the possibility to let some simply AI control the flashlight and shoe.

Approach

There are a few different techniques to accomplish evolution computationally [1]. I found simulated genes to be the most common one, after looking through some existing simulators; among them where the ones from the homepage referenced in the project description [2]. Neural Networks were the second most common technique I found in existing programs, which is something I've never done before, hence probably a bad idea to take on in only 8 weeks. It felt like a good choice to go with gene simulation because it'll work fine with the project description and it feels like something I can do in the time frame.

Choosing a language for this simulation felt like an easy task for me, as I'm in love with C++, and also have good enough knowledge in Windows and GDI[1] programming to handle the graphics and interface with ease. This feels like a good thing, as the focus should be on the simulation and not the user interface.

Simulation of genes is something I've never done before, and therefore I'll probably have to read a bit more about it. Up until now I've found that you have some specific characteristics, commonly among them are speed, lifespan, energy and reproduction. Each of the characteristics suffer a chance of mutation when a new creature is born, so that the children have a slightly different gene pool. At some point in the lifespan of the creatures they will spawn new children, and only the creatures who survive long enough can spawn children (hence natural selection).

---

[1] Graphics Device Interface, a Graphics API in Windows

I'll have a goal for a Basic implementation; today it's the following characteristics...

- Energy
- Speed
- Lifespan
- Reproduction
- Light Sensitivity - some limit - how close to the center of the light will they go?
- Avoid Light - some limit - If they're above some level of energy they avoid light completely.

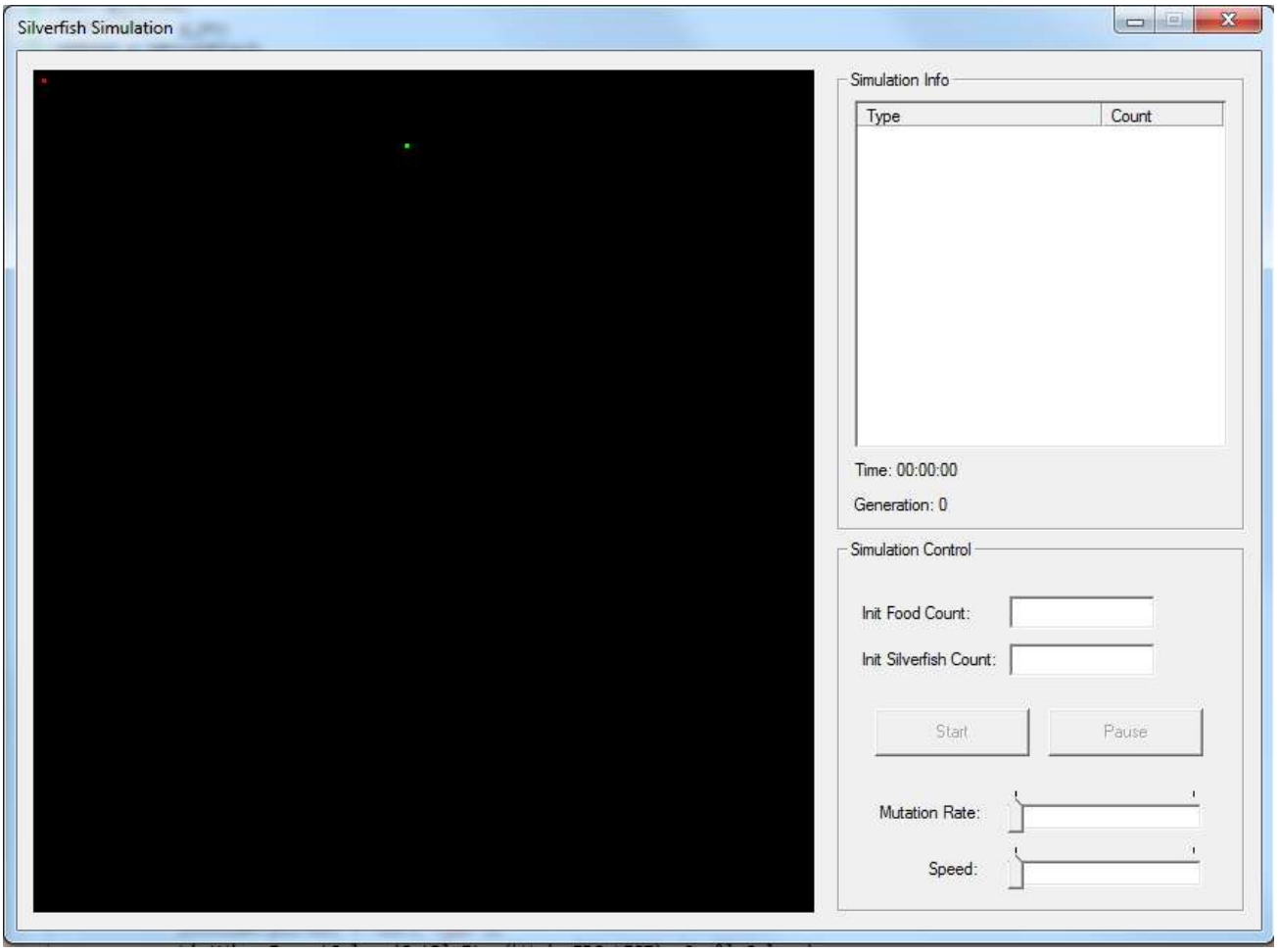
...along with:

- Food
    - Placement
        - A small amount everywhere
        - Lots where the light is
    - Energy
        - How much energy should each bit of food give?
- Outer boundaries
    - Rectangular
    - View? We can only see a small section of the entire simulation at a time
    - Entire? We can see the entire simulation in one window
- Interaction
    - Setting Initial Values
    - Controlling the Flashlight
    - Controlling the Shoe
    - Start/Pause/Resume
    - Speed Control
    - Mutation Control
    - Simulation Information - current values
    - Graphics

For the extended version:

- More characteristics - for example:
    - Preferred location
    - Danger assessment - when should they run away?
    - Danger algorithm - where should they run if the shoe is above them?
    - Viewing Radius - how far can the creature see?
    - Brains - How much information does each algorithm get?
    - Wait - wait for a duration - using less energy
- Eden - an area
    - Faster reproduction?
    - More food?
- AI
    - Flashlight
    - Shoe
- More controls

So far I've made a simple frame with (non-working) controls, based on an inspirational existing program - also coded in C++ [3]. This program was made with C++, Windows and GDI, and will most likely be revised later on in the project.



References

1. Evolutionary Computation; Wikipedia;
http://en.wikipedia.org/w/index.php?title=Evolutionary_computation&oldid=474090669

2. Simulators; Simulated Evolution; John R. Nash;
http://www.simulatedevolution.com/
http://www.simulatedevolution.com/developer-info/similar-programs

3. Inspirational Program; Youtube; scotchfaster;
http://www.youtube.com/watch?v=GuWJ1ZiwO2g

Time plan

I have roughly 8 weeks to accomplish this program along with a project essay.

Week	Task
1	Graphics and Interface
2	Research and Basic structure of the Simulation
3	Research and Basic structure of the Simulation or Basic Implementation
4	Half way meeting and Basic Implementation
5	Basic Implementation
6	Debugging and Basic/Extended Implementation
7	Start on Essay and Extended Implementation
8	Finish Essay

I should manage to finish a good graphics framework in a week. The framework should cope with smaller changes easily. Depending on the amount of research I need, I'll spend up to 2 weeks with research (probably won't need that much though). To setup a basic structure of the simulation I'm going to make a good hierarchy of classes to support all the parts I need. Once I have the framework, I'll code the different end components, and should be ready for some small demo by the half way meeting. I don't expect to be completely done with the Basic implementation by week 4, hence an extra week is needed. Thorough debugging is necessary before I continue with any extended versions, however during the last 2 weeks I'll mainly work with the essay and not extensions. I also have a few extra days after these 8 weeks, where I can polish the essay to perfection!