Office of Strategic National Alien Planning

Genetic World Simulator
(GWS)

Project Team:
Samuel Cagle
Junjie Xu
REDACTED*
REDACTED*

April 25, 2014

Executive Summary

The Office of Strategic National Alien Planning (OSNAP) has made a substantial investment to research alien ecological systems. The Genetic World Simulator (GWS) is an attempt to simulate an alien world with various parameters in order to discover the potential weaknesses, patterns, and systems in alien biology and ecology.

GWS simulates a flat-state world with various physical features (water level, oxygen level, heat level) and alien organisms (entities with various characteristics such as growth rate, mutation rate, offensive and defensive mechanisms) to reveal underlying patterns of alien ecology and to discover the potential growth of any alien organisms in a predetermined environment.

Project Description

Anti-Kinetic Weapon Defense System (AKWDS) is the control software solution to integrate Flying Object Sensor Array (FOSA) and Ballistic Defense System (BDS). In some respects, the project can be thought of much like an automated Missle Command, Artillery, or Worms game. FOSA displays the data in a machine readable format. AKWDS houses the AI that makes decisions from the data. BDS actions the AKWDS firing command. FOSA and BDS for this project are backed by a basic physics simulation. We also plan to have FOSA output go to a 3D visualization in realtime or as a post-hoc analysis.

Simulation is planned to have 1 second between clock ticks. AKWDS processing time will be handled as a constant amount of delay or by rounding up the wall clock time to the nearest second. We simulate 1 second of delay between FOSA and AKWDS. 30 seconds of delay is simulated between AKWDS and BDS (simulating the time to move the firing mechanism). 60 seconds must elapse between shots.

We will write the n-body simulator, which will produce the FOSA stream and consume the BDS stream. We will also write AKWDS, which will consume the FOSA stream and produce a BDS stream. The simulator and AKWDS will live in separate process spaces and will be connected via TCP or MPI or some other IPC mechanism. We also plan to provide a 3D visualization using some visualization technology which may be online or posthoc (OpenGL, pov-ray, JavaFX, blender, VisIt, ???).

Highlevel Architecture

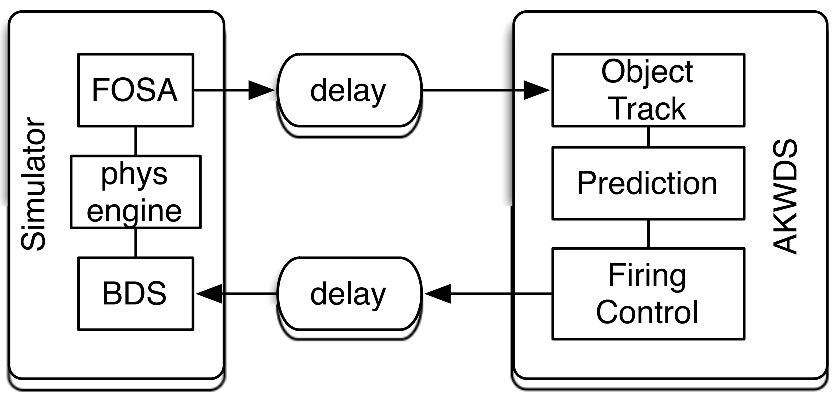

Figure 1: High Level Architecture Diagram

Figure 1 shows a block level diagram of the AKWDS software (without the visualization component). There are two main programs in the solution.

AKWDS This is the control software that OSNAP needs in support of its mission to plan for and execute plans related to real and imagined extraterrestrial encounters. In this case, a hostile invasion scenario.

Simulator The Simulator is responsible for emulating the world that a deployed AKWDS system would be deployed in. This software supports evaluation and tuning of AKWDS without enduring actual extraterrestrial invasion.

AKWDS and the Simulator will be connected over a network. The Simulator will be responsible for handling the effects of delay called out in the project description.

Simulator

The core of the simulator is a basic physics simulation. This N-body simulation will track projectiles and vehicles within the world and apply basic Newtonian physics. The simulator will expose the current location of each simulated object at regular intervals as a list of 3D cartesian coordinates and object radii. This stream will be sent using the FOSA format. The simulator will also take in a stream of BDS actions and add projectiles to the simulation as appropriate. The simulator also enforces some sanity rules regarding object motion and provides some basic flight paths for UFOs.

AKWDS

AKWDS will take the FOSA stream of object locations and attempt to formulate some tracking of the objects (object A with location L last time step is at location L' this time step). This will be used to help estimate the future location of the object so that targets can be lead correctly for firing. AKWDS will then select a set of object to fire at and set of cannons to fire from and send the needed information via the BDS stream.

Visualization

Visualization is not a core component of the project deliverable. However AKWDS debugging and demonstration will be very difficult without providing some animation of the system behavior. A visualizer, online or post-hoc, that works from the FOSA stream will be provided to help validate the software.

Parallel Plan

The AKWDS design includes several places to introduce and explore parallelism.

N-body The core N-body physics simulation seems like it will support up to one thread of execution per body. Efficiently handling interactions, such as collisions, can add some complexity to the solution.

AKWDS The main computation can be articulated as a processing pipeline with phases for identifying objects between samples, computing trajectories, and computing firing solutions.

Object Tracking At each sample, objects need to be identified and grouped. It seems like a non-trivial computation that could benefit from running in parallel and may be able to take advantage of stencils due to locality expectations between time steps.

Trajectories Trivially parallel. For N tracked objects, fit the data points to a function.

Firing Solutions Trivially parallel. For M targets (where M is the number of cannons to fire), compute the future state of the target and compute the required BDS heading, azmuth, and velocity to hit the target in the future.

We plan to look at system scaling by varying the number of initial objects in the simulation and the number of cannons available. We intend to look at wall clock time to complete a fixed number of simulation steps. We also intend to look at latency between the FOSA transmission time and BDS receive time to see if pipelining provides a significant benefit.

If we have extra time, we would like to explore using MPI to spread the simulator and AKWDS computations across nodes to increase the size of simulation that can be done in realtime.

Project Schedule

Week	Deliverable
5	Nail down physical elements to simulate, select 2D technology
6	Basic N-body simulation with mutations, and growth
7	Basic N-body simulation with predation and movement
8	2D output
9	Improving parallel performance and measurement
10	Poster length presentation including measurements