Richard Cato
CSC 548

Final Project Proposal: Boids

Background:

Boids is a flocking simulation where each boid's movement is influenced by the boids around them. The basic rules that dictate the movement of boids based on neighboring boids are:

1. Separation - boids try to maintain a minimum distance
2. Alignment - boids try to move in the same direction
3. Cohesion - boids try to flock towards the center of the a cluster

This algorithm can often be found to simulate birds, particles, and other effects in film or games. My motivation behind picking this problem for parallel computing, is that simulating a large number of boids becomes computationally expensive due to the O(N^2) nature of the naive implementation where each boid is affected by every other boid. There are several different solutions to tackle this issue, and I'd like to compare a few possible solutions.

Parallel Optimizations

Below are the following parallel optimizations I would like to consider:

Computer Shader:

The calculations can be offloaded from the CPU to the GPU through a compute shader. The positions of the boids can be saved in a texture and sent to the GPU, allowing each fragment in the shader to run the boids simulation based off of all other boids.

Boid Clusters/Hierarchies:

A boids cluster would be a region of boids of specific densities. Boids within a single cluster would run the simulation based on the boids within their cluster, Then each cluster would be simulated against the average motion/position of other clusters. This method would use MPI to split the local simulation of each cluster amongst different nodes, then broadcast the average position and direction of their cluster to all other nodes.

Spatial Partitioning:

In this method, the space that the boids can occupy will be broken up into smaller regions. Each node in a compute cluster will be responsible for running the boids simulation for each boid within their region. The regions will overlap, allowing boids to potentially occupy several regions at once, causing the final result of the boid simulation to be reduced to a single node.

Sequential:

This is the classic sequential boids simulation on one core, and will be used to benchmark the other algorithms.

Milestones:

1. Sequential boids simulation
    a. Determine a number of boids that one core struggles to run in a reasonable amount of time.
2. Basic compute shader array summation
    a. Learn how compute shaders work
3. Compute Shader boids algorithm
    a. Compare results to sequential
4. Clusters algorithm
    a. Either start with pre-formed clusters of boids or determine a way to split boids into clusters at run time
    b. Determine how/if clusters should be combined if they collide
    c. Determine how/if clusters should split if the internal boids simulation grows too large
    d. Develop clustering algorithm
    e. Benchmark against sequential
5. Spatial Partitioning
    a. Determine spatial partitioning structure/algorithm
    b. Determine how nodes will reduce effects on shared boids to one node
    c. Benchmark against sequential.