Swarm-PI System Overview

Stephen Majercik, Frank Mauceri, Grace Handler, Jack Truskowski

Updated 5/1/16

**I. Directory Structure**

**Max Patch Files:**

*ControlPanel\_XX-XX.maxpat* - This is the max file that the simulation and contains everything related to the simulation including the external jitter object that is used in the simulation. Instructions on running the simulation can be found in this file in Presentation Mode.

*FlockParamUI\_XX-XX.maxpat* - This is a max patch that is embedded in the *ControlPanel* patch. Contains the number fields for all the parameters of a flock and packages when a parameter is changed in the Control Panel, this file packages a message that will be sent to the external object.

ie) Changing the speed of the first flock to 5.0 would produce the message: 'speed 5.0 1', where 5.0 is the new speed for this flock, and 1 is the flock whose speed will be updated

**Code:**

*XX-XX\_Code* - This directory contains the code for the jitter external, *jit.boids3d.mxo*, which performs updates to the boids. Opening *jit.boids3d.xcodeproj* will allow you to make changes to the code which can then be compiled into a new external (For compilation instructions, refer to the User Manual in the *Help/* directory)

*Dependencies* - This directory contains various Jitter and Max header files that are required to compile the external. This directory should remain unchanged.

**Help Files:**

*Help ­-* Contains various instructions and help files that are useful for running and understanding the simulation and external object.

*README.md* - A markdown file for GitHub containing general information about anticipated future work.

**Other:**

*jit.boids3d.mxo* - The compiled jitter external object from the code in the *XX-XX\_Code* directory. This file is used in the Control Panel to perform updates on the boids.

*paramPreset.json* - A JSON object containing preset flock behaviors. Interacting with this file is not necessary, it is automatically loaded into the presets section of the Control Panel upon startup and presets can be saved within the Control Panel.

**II. Jitter External Data Structures**

This section contains info about the data structures used in the *jit.boids3d.mxo* object. This code can be found in the XCode project in the *XX-XX\_Code/* directory.

**Flock Parameters:**

The flock-specific parameters for the flocks are stored in arrays of size 6 (an index for each flock) in the jitter object struct.

Example Code:

typedef struct \_jit\_boids3d

{

double minspeed[MAX\_FLOCKS]

double maxspeed[MAX\_FLOCKS]

...

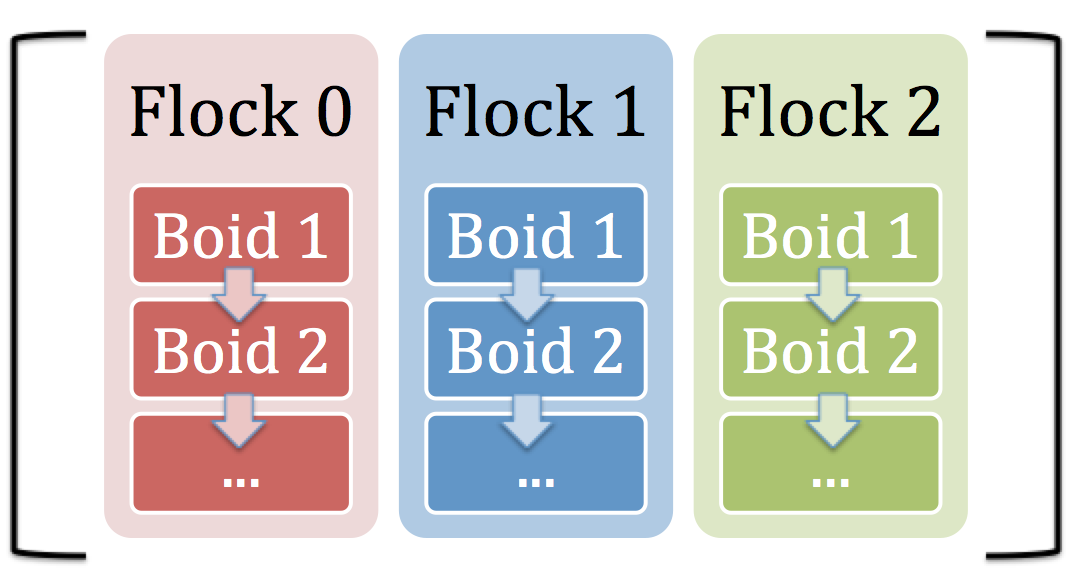
}

Note: the constant MAX\_FLOCKS is set to 6

**Boids:**

Boids are individual structs that are stored in a linked list for each flock. A linked list was used because this allows boids to be added and deleted quickly without the overhead of creating a new array of appropriate size and copying the boids over every time the count changes.

The linked lists for each flock are stored in an array of size 6 (an index for each flock). Refer to Fig. 1 for the organization of flocks/boids.



**flockPtr flockLL[MAX\_FLOCKS] =**

**Fig. 1: Organization of the boids linked lists**

The boids struct contains the following data about a boid:

- Flock ID (which flock this boid is a part of)

- Age (int, in timesteps)

- Old Position (double[3], xyz)

- New Position (double[3], xyz)

- Old Direction (double[3], xyz)

- New Direction (double[3], xyz)

- Speed (double)

- Neighbors (long[kMaxNeighbors])

- Neighbor Distance Squared (double[kMaxNeighbors])

- Pointer to the next boid in the flock

**Attractors:**

Attractors are also structs which are stored in a linked list (called attractorLL) in the \_jit\_boid3d object. The attractor struct contains the following:

- Attractor ID (int)

- Location of the attractor (double[3], xyz)

- Radius of the attractor (note: this is called attractorWeight)

- Pointer to the next attractor in the LL

**Neighborhood Lines:**

All of the neighborhood lines (a maximum of kMaxNeighborLines lines) are stored in the array:

NeighborLinePtr neighborhoodConnections[kMaxNeighborLines];

Each neighbor line in this array is a struct that contains the following data:

- Position of the first boid (float[3], xyz) - this is an endpoint of the line

- Position of the second boid (float[3], xyz) - this is the other endpoint

- Flock ID - neighborhood lines are only drawn between boids of same flock

**Other:**

- Birth location is stored in an array - double birthLoc[3]

- The center point of the current flock being updated is stored in an array - double tempCenterPt[3]

- The boundaries of the simulation are stored in an array - double flyrect[6]

**III. General Code Structure**

**1. Lines 0-375: Global variables, structs, and function definitions**

This section contains code for the above components. It also contains the initializer for the \_jit\_boids3d object which is called on startup and initializes all of its attribute methods

**2. Lines 375-725: Boids attribute methods**

This section contains all the attribute methods for the external. When a message is passed to the external to change a parameter, the cooresponding attribute method in this section is called, which then updates the global variable for this parameter stored in Section 1.

**3. Lines 725-990: Matrix output methods**

After every timestep, these methods are called to prepare the matricies that will be outputted in the 4 outlets of the external object back to the Max Patch.

jit\_boids3d\_matrix\_calc(): Prepares the matricies, populates matricies 2-4 with data, and outputs all of them to the max patch.

jit\_boids3d\_calculate\_ndim(): Called by the matrix\_calc method, populates the 1st matrix with data

For more information on the contents of each matrix going back to the Max Patch, refer to *Help/Swarm-PI\_Parameters.docx*

**4. Lines 990-1445: Boid update methods**

These lines contain all the functions that are used to update each boids direction and position every timestep.

FlightStep(): Calculates the new velocity and position of each boid. Calls the other methods in this section to get each component of the boids new velocity

**5. Lines 1445-end: Memory management methods**

Methods in this section are intializers for the various structs and objects in the external. Allocation and freeing of memory for all of these objects occurs in this section.