Flock Box Manual

Version 1.3

QUICK START

Refer to any of the Demo scenes for an example of how to set up a FlockBox. If you're starting from scratch there are 3 primary components you'll need:

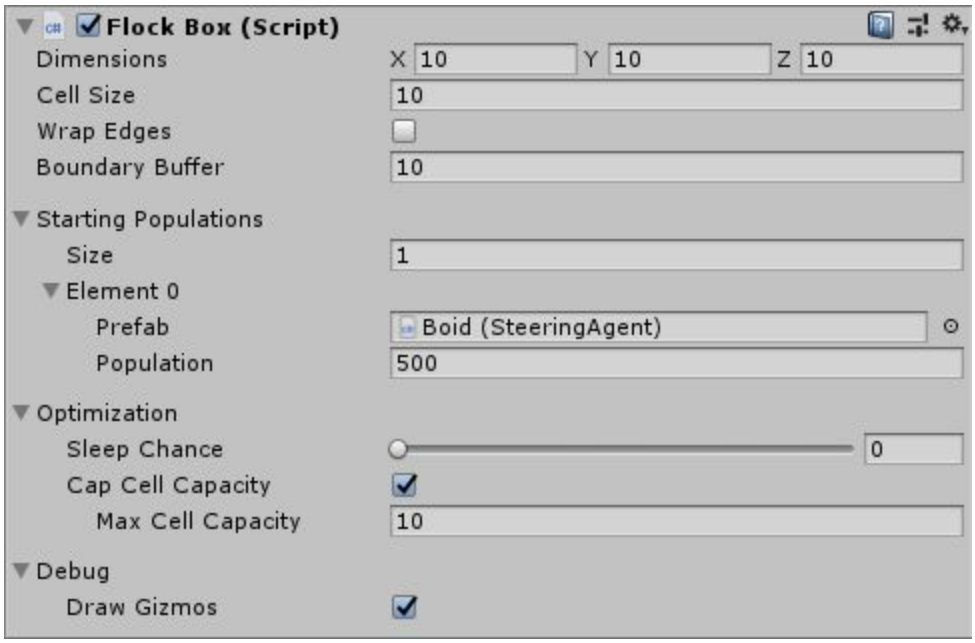
1. SteeringAgent prefab
    a. Add a SteeringAgent component to an empty GameObject and save it as a prefab.
2. BehaviorSettings asset

a. Use Create > Asset > BehaviorSettings to create a new BehaviorSettings asset in the project. Add any desired behaviors (Alignment + Cohesion + Separation is the classic Boid formula).
b. Drag and drop the settings onto the SteeringAgent prefab's activeSettings slot.

3. FlockBox component
   a. Add a FlockBox component to an empty GameObject in the scene.
   b. Create a new slot in Starting Populations and add the SteeringAgent prefab.

Press Play and watch them go!

FLOCK BOXES



Dimensions

Number of cells in each dimension. The world space extents of the FlockBox will be dimensions * cellSize. Note: you can set one dimension to 0 to create a 2D FlockBox.

Cell Size

FlockBox uses spatial hashing for optimization, meaning that the box is divided into cells that each keep track of the agents they contain for faster lookup. Cells that are too large or small may cause performance issues. The ideal cell size is usually going to be around the max perception distance of the most common agents.

Wrap Edges

Defines whether or not agents can travel through the boundary and wrap to the other side of the box. If false, agents will always be constrained to positions inside the box.

Boundary Buffer

Defines a “buffer zone” which agents will attempt to steer out of when wrapEdges is set to false. (see Containment Steering Behavior)

Starting Populations

Define populations of Agents that should be instantiated in Start(). Their position and velocity will be random. Note: you can also drag individual Agents into the Scene as children of the FlockBox and they will automatically be detected (see Avoidance Demo).

Sleep Chance

A percentage value that determines how often Steering Agents skip updating their velocity. A value of .5 means that roughly half the Steering Agents will not update on any given frame. This can be great for performance if the reduced fidelity isn’t a problem.

Cap Cell Capacity

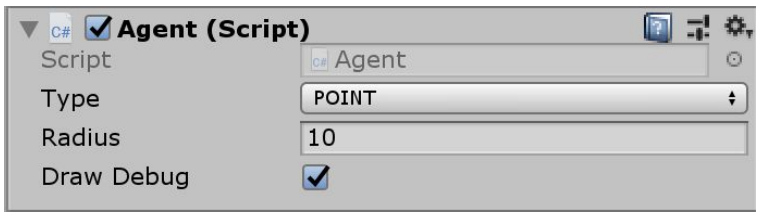
Limits the number of Agents that will be recorded in each cell. When Agents are evenly distributed, they can make use of spatial hashing to limit the number of neighbors they need to calculate with. If Steering Agents become bunched together, spatial hashing will no longer provide a benefit. In case of overcrowding, this cap will limit the total number of neighbors that each individual is aware of to prevent the framerate from getting dragged down.

Display Gizmos

Draw debug information in the Scene View. The full extent of the FlockBox and boundary buffer in world space are drawn as wireframe boxes. At runtime cells that have at least one occupant will be shaded in.

AGENTS

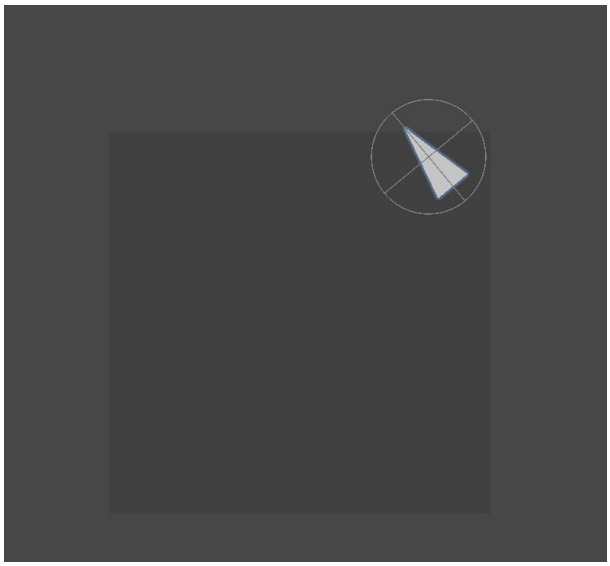
An Agent is any entity that is involved in the Flock Box simulation. A basic Agent will not move.



Type

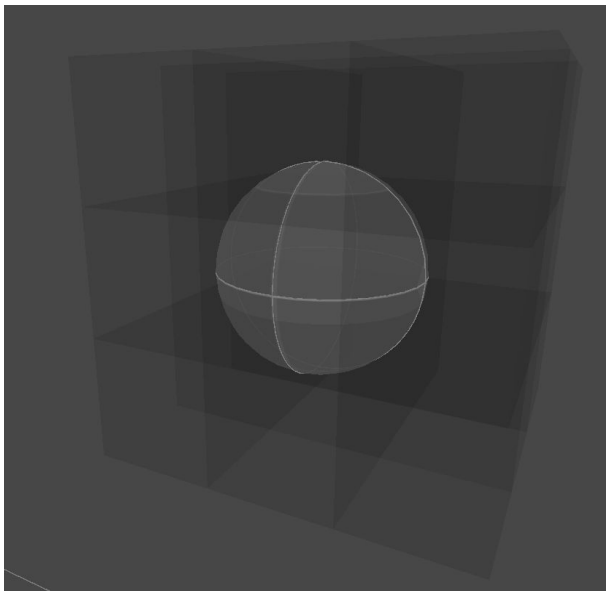
POINT and LINE behave similarly to SPHERE and CYLINDER, but can update much faster when moved. The trade-off is that they offer lower fidelity. In general, SPHERE and CYLINDER should only be used for large, immobile Agents, while POINT and LINE should be used for small, mobile Agents (like Boids).

POINT



Agent will only occupy a single cell based on their center point, even if their radius extends into neighboring cells.

SPHERE

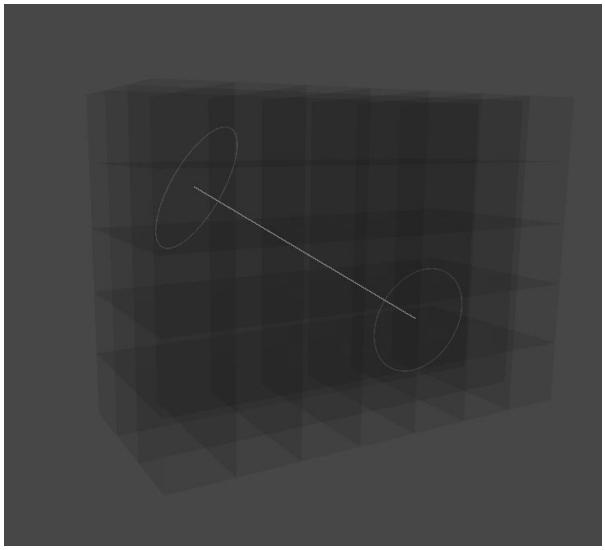


This Agent will occupy all cells within a bounding box containing a sphere.

LINE

This Agent will occupy all cells in a straight line. The cells are calculated using Bresenham's line algorithm.

CYLINDER



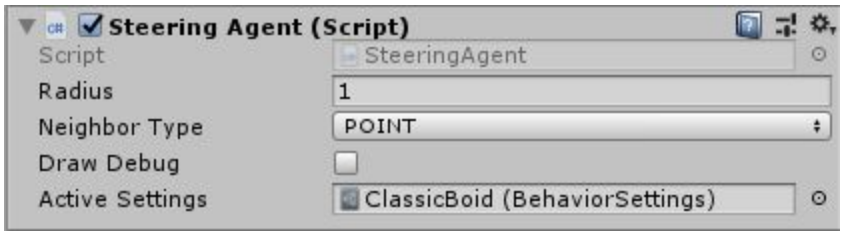
This Agent will occupy all cells within an axis-aligned bounding box containing a cylinder.

Draw Debug

Draw a Gizmo representing the Agent’s Shape

Steering Agents

Steering Agent is derived from Agent, but adds the ability to update its position and velocity based on its surroundings.

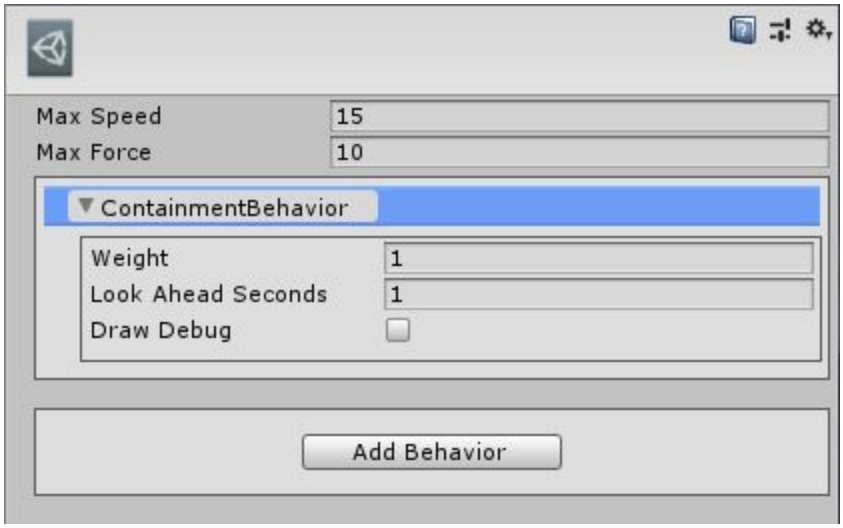


Active Settings

Reference to a Behavior Settings asset.

BEHAVIOR SETTINGS

Create a new BehaviorSettings asset with Create > Asset > BehaviorSettings.



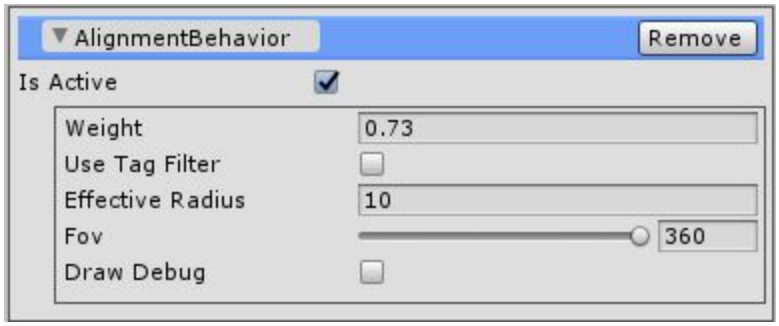
Max Speed

The top speed that a steering agent can reach.

Max Force

The maximum force that can be applied to the steering agent. Tune this value to control how fast the agent accelerates.

Steering Behaviors



Weight

Adjust this value to change the relative influence of each steering behavior.

Tag Filter

Uses Unity's built-in Tag system. A tag filter should be used if a behavior's steering calculation should only take certain agents into consideration and ignore all others. For example: if agents with an Avoidance behavior should only avoid obstacles and not each other, tag the obstacle as "Obstacle" and have the Avoidance behavior filter for that tag. (See Avoidance example). This may not be applicable for all Steering Behaviors (eg. Wander).

Draw Debug

If you want a visual representation of each steering behavior, enable Draw Debug and assign the behavior a color. Steering vectors will be drawn in the Scene View.

PACKAGED BEHAVIORS

Steering Behaviors have been implemented as described by Craig Reynolds [1]. Highlighted in yellow bellow are the classic component behaviors of a Boid.

Alignment

Steer towards the average forward direction of surrounding agents.

Separation

Steer away from nearby agents. Weighted by distance so that the closer other agents get, the greater the separation force becomes.

Cohesion

Steer towards the midpoint of surrounding agents.

Containment

Containment is added automatically to each BehaviorSettings, and is only used when inside a FlockBox with wrapEdges disabled. This behavior will steer the agent out of a buffer zone defined by FlockBox.boundaryBuffer.

Avoidance

Steer to avoid an upcoming obstacle. Will stop applying force when the agent's path will not intersect, or is tangential to the surface of the obstacle.

Flee

Steer away from the midpoint of other agents.

Seek

Steer towards a stationary target.

Pursuit

Steer towards an anticipated interception point with a moving target.

Arrival

Steer towards a stationary target, but reduce steering force linearly on the approach to come to a stop.

Wander

Steer in a random direction (calculated with Perlin noise).

Leader Follow

Similar to Arrival. Steer toward a point that is behind a target Leader. If multiple potential leaders are found, the closest one will be chosen.

WRITING CUSTOM BEHAVIORS

To create a custom behavior, write a class that inherits from SteeringBehavior (or RadialSteeringBehavior or ForecastSteeringBehavior) and override GetSteeringBehaviorVector(). If there is information that you need to calculate your steering vector that isn't currently accessible, consider adding it as a new field in the SurroundingsInfo struct and filling it in inside FlockBox.GetSurroundings().

Agent Instance Data

It's possible for agents to have custom instance data without extending the Agent class through Agent.SetAttribute() and Agent.GetAttribute(). Values are stored in a Dictionary<string, object>, so you need to cast them to the proper type when retrieving them. This is used in SeekBehavior to keep track of a particular seek target and prevent constant target switching, which leads to more natural behavior. Instance variables can also be added by extending SteeringAgent, but the Attribute system is useful when you want a Steering Behavior to be usable by any Type of SteeringAgent, without filling up SteeringAgent with new field that most agents will never need to use.

Adding Custom Behaviors to Behavior Settings

All non-abstract classes derived from of SteeringBehavior will appear in the ”Add Behavior” list automatically.

Editing Values in the Inspector

Any properties that you want to be able to edit in the inspector should either be public or marked with the [System.Serializable] attribute. Further editor customization is not supported because I’ve found that PropertyDrawers and polymorphism don’t work together when the objects are stored in a collection as their base class (SteeringBehavior array in this case).

CHANGE LOG

Version 1.1 11/2/2019

- Added CYLINDER shape
- Expanded support for LINE shape, including new fields exposed in Inspector
- Changed “NeighborType” to “Shape”

Version 1.2 11/9/2019

- Added LeaderFollow behavior
- Fixed bug preventing Pursuit behavior from functioning.

Version 1.3 12/12/2019

- Optimized memory allocation.
- Added Optimization option to cap the number of Agents recorded in each cell.

Contact

Send and comments or questions to:

marc@cloudfinegames.com

Works Cited

[1] Reynolds, Craig. (1999). Steering Behaviors For Autonomous Characters.