15-466/15-666 Computer Game Programming Fall 2015

Project 1. Due: 11:59PM, September 29th, 2015.

In this assignment, you will implement simple virtual agent movement behaviors and basic obstacle avoidance in Unity.

This assignment has **no starter code**. You will create a Unity project from scratch, populate it with objects, and script those objects yourself. You are free to use any assets you wish to in your project, as long as they are able to perform the tasks required by the assignment. If you use assets from the internet, do try to respect the licenses they come with, so you can showcase your assignment without worrying about licensing restrictions. Because there is no starter code, please make your code easy to read and well-annotated.

If you need assistance with setting up a Unity project please drop by office hours or email Pasan (pjulsaks@andrew.cmu.edu).

1. Movement Behaviors

In your project, you must include 10 characters that can move around according to certain movement behaviors. These characters can be a stock 3D character from Unity's libraries, such as the construction worker, a 3D model from the Unity asset store, or a custom model of your design. We encourage you to experiment with Unity's animations features, but remember that the important part is the movement behaviors. The 10 characters don't need to be unique models, you can use 10 instances of the same model. To complete this part of the assignment, you must create a script that implements the following behaviors. You may find it useful to create one script that can toggle between the different behaviors instead of making different scripts for each behavior. Keep in mind that part 2 of the assignment will ask you to add collision avoidance to your behaviors, so plan accordingly.

The Behaviors

- **Wander** The typical wander behavior. The agent wanders around without sudden velocity changes or other anomalies.
- **ReachGoal** This is an "arrive" behavior. The agent approaches a goal position and comes to a stop when it arrives. Remember that Unity can assign game objects to variable in the Inspector, so it may be useful to have a GameObject that can serve as a goal that can be easily modified in engine.
- **FlockingWander** All agents with this behavior active will wander in one group. If only one agent has this active, then that agent just wanders. You may declare one character to be the leader of the group that the other characters flock around. While grading we will look for a group of agents in a pack moving together and hope to see the velocity of the whole group change in a wandering fashion. Please make three separate Unity scenes, one for each behavior, and configure them so that the proper behavior is shown as soon as the scene is started. Document any configuration options for each scene to help us evaluate your behavior. Failure to split your project into three scenes will make your project more difficult to grade and we might miss certain things that were actually included in your project.

Extra Credit (optional)

- **VShapeFormationWander** This behavior should be something like FlockingWander but the groups of agents should form a V shape, like a flock of geese flying. If you are unfamiliar with this phenomenon, try putting “flock of geese” into an image search engine or contact the TA. It is okay to declare one agent as a “leader”. You may want to balance the formation so that cases where only one half of the V is active do not happen. Successful implementation of this behavior will give at most an extra 10%.
- **Flee** This should be opposite of “ReachGoal” behavior. However, after certain distance between the agent and the target, the agent should stop running away from the target and just wandering around. In this case the target should be wandering around or moving toward the agent with slower speed.
- **FleeFromGroup** Same as “Flee” behavior, but there should be multiple targets for each agent to run away from. Thus, you should clearly mark each targets (different model/color) for better visualization of the behavior.
- **CircleFormation** When this behavior is selected, the agents should form the shape, while minimizing the distance each agent has to move to form one. You can add more agents to the scene to make these formations more noticeable.
- **SquareFormation** Same as **CircleFormation**
- **TriangleFormation** Same as **CircleFormation**.
- **Additional features** You may add any additional interesting features as you see fit, such as animation blending, particle effects, or UI. However, implementation of these features will give you at most an extra 10%, so only work on this once you’ve finished the required behaviors. Please note that simply using a different model besides the default construction worker does not count as extra credit.

2. Obstacle Avoidance

On top of the behaviors themselves, you must also implement obstacle avoidance as part of your behaviors. Obstacle avoidance should take some level of priority over the movement behaviors. This means that you should not let characters walk through walls or fall off the stage. Your characters should also try to avoid colliding with each other while they are moving around the stage. Do NOT rely on the Unity physics engine for this. Collider and RigidBody components to prevent penetration of agents with obstacles are off-limits and will be disabled during grading. However, you may use ray-casting features to assist your calculations. If you use these features, include what features you used in your write-up. You may use whatever bounding object you like to simplify your collision calculations: boxes, cylinders, spheres are all ok. For this assignment, you are allowed to include an explicit list of obstacles to avoid in your movement behavior script.

3. Documentation

With your project, please hand in a document that explains the approach you took and how you implemented it for each behavior. Please keep this as concise as possible. Do not just give us pseudocode with no explanation and please use English. Also include details of how to properly run your project and any options and settings you've added in your project.

4. Video

In addition to your write-up, please include a short video of your characters performing each of the movement behaviors. This is a required part of your hand-in materials. The video will give you something to show off later, in demo reels for instance, and helps resolve confusion while grading. To capture video from Unity, you can use whatever software you want. For Windows, this includes programs such as FRAPS, Bandicam or CamStudio and for Mac this includes Quicktime screen capture. If you're having problems, email the TA or come to office hours.

5. Submission

Submit your entire Unity project by compressing it into a common archive format such as .zip or .rar. Redmine accounts on Max's lab server will be set up for each student and submission can be completed there. Look out for an email with more details.

6. Grading Criteria

Since there are different number of people in each group, the requirement will be a bit different. For 1-person team, in order to get full mark (100%) you need to get 100 points, for 2-person team 120 points and for 3-person team 140 points. To be specific:

1-person team's total score will be equal to the number of points earned

2-person team's total score will be equal to the number of points earned multiplied by 0.833 (=1/1.2)

3-person team's total score will be equal to the number of points earned multiplied by 0.712 (=1/1.4)

Proper implementation of the following Behaviors based on realism of motion (e.g., no instantaneous velocity switching, no oscillations, no collisions, no slowdowns due to computationally-intensive operations)

- Wander (20 pts)
- ReachGoal (20 pts)
- FlockingWander (20 pts)

Collision Avoidance also based on realism of motion

- Wander (10 pts)
- ReachGoal (10 pts)
- FlockingWander (10 pts)

Documentation provided (5 pts)

Video provided (5 pts).

Extra Credit:

- VShapeFormationWander (10 pts)
- Flee (10 pts)
- FleeFromGroup (10 pts)
- CircleFormation (10 pts)
- SquareFormation (10 pts)
- TriangleFormation (10 pts)
- Additional features (10 pts)