Professor Please

15-466 P3: Intelligence Documentation

Anna Etzel (aetzel), Ivan Wang (icw), Jun Huo (jhuo)

Overview

We chose to implement Planning-based Decision Making to create smarter AI's for students in our game. Specifically, we want students to plan their movement via communication with each other. Together as a group, students will search for the professor and message each other about his last known location.

Testing

This project has 2 relevant scenes, *Title.unity* and *Planning.unity*. The Professor character is player-controlled by the arrow keys and can jump gaps with the Spacebar. To test, run the Demo/P3Demo.exe file (opening multiple instances for multiplayer), or load the *Title* scene in the editor.

Included in the Demo/ folder is a video simultaneously demonstrating multiplayer and intelligence.

Planning-based Decision Making

Each student in *Planning.unity* has a *Communication.cs* behavior script. They are assigned to a group (identified as an integer) and act according to a specific state, via a finite state automaton.

State 1: Wander

In the WANDER state, neither the student nor his/her groupmates know where the professor's location is. They simply walk around or pause to idle about their homework questions. If the student sees a professor or receives a ping from a friend, they transition to a different state.

State 2: Follow

In the FOLLOW state, the student has seen the professor and is following them around with questions. When he/she first sees the professor, the student will ping all the friends in their group who are wandering around, updating them with the last known location of the professor.

State 3: Seeking

In the SEEKING state, the student has received a ping about the professor's last known location from his/her friends. They will pathfind towards that location until they either see the professor (and switch to FOLLOW), or arrive. Upon arrival, they will switch back to WANDER and alert their groupmates that the professor is no longer there.

The demo contains 3 groups of students. When any one student sees a professor, the others in the group will be messaged and they will all swarm towards the professor's last known location.

Multiplayer

Using Unity's NetworkManager, we set up the scene to allow for multiple clients to connect to a server. Different players can control multiple professors who must run around avoiding students' questions.

Each player is set up as a Professor prefab, controlled in each client. Their movement is enabled only for the owner client (referenced via NetworkBehaviour.isLocalPlayer). This is synced to the server and broadcast to other clients. Students, on the other hand, run movement behaviors only on the server, and propagate their transforms and animations to the other clients.

We modified student behavior such that they now target the nearest professor, iterating through all GameObjects which contain the "Player" tag. Questions now spawn only on the screen of the player who is targeted, rather than all professors' screens. In addition, the game ends when the first professor loses, rather than all of them.

Chat

Using a GUILayout text box, players can now send messages in multiplayer. When there are at least 2 players online, the chat box appears (see *Chat.cs*). Typed messages are sent to the server and broadcast to all players. When received, the message will appear as a speech bubble (as if the other professor is asking them a question).

Additional Features (UI)

In addition to the features above, we implemented the following:

Title screen
An extra scene, *Title.unity*, opens into the game when the player presses a key. The networking code is set up to create a server or connect a client in this scene.

Speech bubbles
We added speech bubbles for the students' questions when they approach the professor. Each student chooses a random question from a predefined array of choices, and the bubble pops up at a random location on the screen. When there are more than some constant number of bubbles blocking the screen, the player loses.