Lab 3

Creation of a pacman game

Authors: Bertocco Thomas-Killian - #13758322
Thanh Tu Luu - #13716589

Lecturer: Dr. Da Sheng Liu

CP5609 Intelligent agents and simulation | SP53 | 2019

29 January 2020

Modest Pacman (Java game)

The player control scheme of the game

- The player needs to run the Pacman.jar executable.
- The player controls Pac-Man through the maze with the arrows keys of the keyboard.
- If the player eats a power pellet, the ghosts will turn blue, and the player can eat the ghosts until they turn back to their normal color.
- The player must avoid contact with the ghosts if he is not under the power pellet’s effect.
- The player must eat every dots and power pellets to advance to the next stage.
- After the completion of the 3 stages, the player wins the game.

The unique scripting implementations

a) Ghost behaviours:

- GhostAmbushBehavior.java:
  The ghost looks at Pacman's position and orientation. It tries to predict the Pacman's future position and ambush him. To proceed, it just takes a Pacman position and adds 4 tiles in the direction of Pacman's orientation.

- GhostChaseBehavior.java:
  The simplest behavior. The ghost just moves to Pacman's position by implementing the FathFinder function which inputs the Pacman position and the position itself.

- GhostObliviousBehavior.java:
  The ghost who uses this behavior will first look at the distance between him and Pacman. If the distance is greater than 8 tiles, the ghost will move to Pacman's position. Else, it’ll move randomly on the map.

- GhostPinchBehavior.java:
  This behavior is a little bit complex. The ghost will look at the orientation of the chasing ghost (red ghost). It will draw a vector between the red ghost and its target position, and double the length of the vector. The end of the vector is the target of pinch behavior.

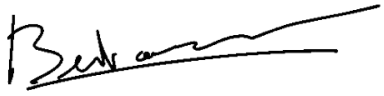
- GhostFleeBehavior.java:
  If Pacman eats a power pellet, every ghost will move to the opposite of their current target position.

b) PathFinder.java: This class has the role to gather all the positions of the elements in the game world and return the distance/position of the elements. It contains a list which stores the position of each element in the game world. Then it initializes the list with the position 0 on the map, and add the next elements of the list one by one.

Known issues / bugs

None.

Individual contribution form

Tasks	**Luu Thanh Tu**	**Thomas-Killian Bertocco**
Game World	70%	30%
Game Play & Mechanics	60%	40%
Control & UI	10%	90%
Aesthetics	60%	40%
Technical Complexity & Innovations	40%	60%
Documentation	50%	50%
Assets List	50%	50%
Signature		

Complete list of assets used in the project

ID	Asset file name	Asset type	Source	License
1	Simple pacman project	Java project	https://github.com/coolioasjulio/ArcadeMachine	MIT License

Versioning repository:

The project is hosted on Github:

https://github.com/Yunori/Lab3