Université Catholique de Louvain
Ecole Polytechnique de Louvain

LINGI1131 – Computer Language Concepts

Project
2015-2016

de Bellefroid Cédric - 3263-11-00
Somers Harold - 3789-13-00

May 13, 2016

1 Introduction

2 Round version

This first version was a round by round game. Each time the user press on a button, Pac-Man moves in the direction asked, if possible, and ghost moves randomly. Ghosts are not really moving randomly as they check if their previous movement is still available. This method prevent any ghost to stay in the same part of the map. Our implementation is threaded safe as only the MAP regulator is allowed to change the MAP variable.

2.1 Map Regulator

We developed a very clear structure. After drawing the map, the "MAP" procedure is launched in his own thread. The "MAP" procedure is the arbitrator of the game. It allows Pac-Man movements, make ghosts moving and determines points and Pac-Man's lives. It works as follows : the arbitrator is waiting for a movement from Pac-Man, determine if the movement is available, send on the ghost port a word to ask them new movements, compare Pac-Man's and ghost's move, calculate the new number of lives of Pac-Man and coins left or eaten by Pac-Man. The "MAP" procedure works as a Port-Object function. It keeps states in mind, the map and the number of Pac-Man alive, while waiting for Pac-Man moves. The all Map regulator was developed in order to extend the game for multiplayer. The MAP arbitrator is also the procedure that draws the new elements on the map (for example, black square when coins are eaten). It communicates with pacman by acknowledgements.

It is the MAP regulator that ends every thread when the game is finished.

2.2 Pac-Man thread

Every Pac-Man is launched in his own thread. When they receive a move on the commandPort, they send to the MAP arbitrator a message $(r()Ack)$ for their movement. The map arbitrator will then respond with the new position of Pac-Man in the Ack variable. Pac-Man is also a port object procedure. It keeps, the number of lives, the position, the number of coins eaten and the last movement in mind.

2.3 Ghost Thread

All ghosts are moving in the same thread. A list is representing actual state of every ghost (position, scared variable)... Ghost thread is waiting for instruction from the MAP regulator to choose an almost random movement and then send this movement back by acknowledgement on ports to the MAP regulator. It also wait for scared instruction if Pac-Man eats a power pellet.

For that purpose we added a special procedure in a thread for the scared mode. When Pac-Man eats a power pellet, the MAP regulator send a message to the scared thread with the time ghosts have to be scared. This thread manages to make ghost thread for a special time by sending them a message when scared mode begins and ends. When Pac-Man crosses a ghost, the MAP regulator determine if Pac-Man is the dead or the ghost eaten by looking at the scared variable present in each ghost state. If pacman eats a new power pellets while another scared mode is not finished, the new timer is added to the rest of the old one.

3 Continuous Version

Implementing the continuous version was quite short. Our continuous implementation is still declarative. In order to do that, we added two new threads that are communicating together. The first one is a Timer that sends every N millisecond a message to the continuousGame thread. The Continuous game thread is an intermediary between command received by the program and the command received by Pac-Man's thread. It is implemented as a POrtObject function with the last command

received in mind. When Continuous Game receives a message from timer, it sends to Pac-Man this last movement.

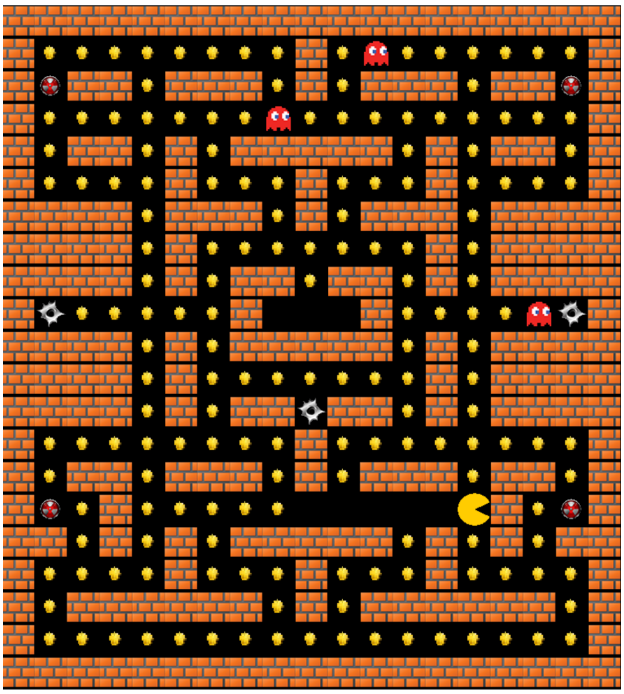

Figure 1

4 Special Features

All the code is implemented in order that new rules or feature can be added easily. It is the MAP regulator that commands all the rules of the game. We also made possible to play with two players but haven't.

5 improvement possibilities

When we thought about the code and the implementation, we kept in mind that we could possibly add more Pac-Man to play against some friends. So our architecture allows us to easily add this feature on our actual code. We could also add a graphical window to display the actual score and the remaining lives. We have implemented a continuous game, but the movements are still frame by frame. The game would be more playable if pac-man and the ghost were moving pixel by pixel. this improvement is not very difficult because we could synchronise the required time to travel 40 pixels with the timer waiting for a new move