CS324 - Computer Graphics: Pac-Man Documentation - Alex Macpherson

Game Features

In remaining faithful with the original game[1], the solution (Fig.1) incorporates a number of features, including scoring, a persistent high score, player lives - affording two retries before the game is over, and a third if the player reaches 10,000 points - and game levels of increasing difficulty. The game can also be paused to show a help-screen by pressing the ESC key (Fig.3).

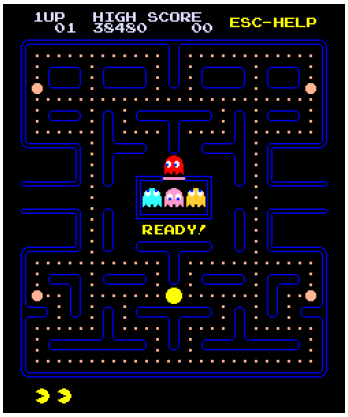

Figure 1: *Solution remains faithful to original Pac-Man where possible*

Original sprites have been up-scaled for better quality at higher resolutions, and are animated by changing the texture over time (Fig.2). Brief pauses in play to display score when eating a fruit/ghost afford a more genuine feel.

Ghosts are implemented with their original AI[2], each representing a unique trait - for example the red ghost, *Blinky*, chases Pac-Man directly, where the pink ghost, *Pinky*, attempts to ambush the player by targeting ahead of Pac-Man. Ghost behaviour is characterised into a number of modes, from their movement within the spawn pen to hunting Pac-Man in waves - switching between their unique **chase** AI and a **scatter** mode, dispersing each ghost towards a different corner of the map. Wave AI is determined by time, with **chase** waves increasing- and **scatter** waves decreasing in length as the game goes on, enacting greater difficulty with each level.

Figure 2: *Pac-Man's death animation sequence, comprising of 11 distinct sprites*

On eating a big pill, ghosts become **frightened**, moving slowly and erratically as Pac-Man is able to consume them for a score bonus. A multiplier is added for eating numerous ghosts before the timer runs out and they reenter the wave-determined AI.

Pac-Man and ghosts can use portals on either side of the map to quickly travel to the opposite side. Remaining true to the original game, ghosts travel at half speed in the portal tunnels. Equally, there are a few junctions at which ghosts cannot elect to travel upwards, affording players a quick getaway.

Bonus fruits spawn randomly in the lower-third of the map and can be eaten for a score bonus before they disappear. Fruits in later levels are worth more points, their value reflecting the level's difficulty.

Design Overview

The GLUT idle function forms the basis for a game loop, using time-calculated sleeps to restrict iterations to *30fps*. A clear separation is made between game logic and graphics, with all logic being determined in the game loop prior to calling a redraw, which draws a fully-calculated game state.

The solution heavily relies on texture-binding using *LibPNG* within *OpenGL*, with a custom method drawing sprites of a given texture, size and angle to give the game an authentic feel.

Object-oriented programming enables encapsulation of common code and simplicity of character and game manipulation for both Pac-Man and Ghosts. On top of this, a number of *ENUM* types are defined to improve code readability across the solution.

The map is divided into tiles, each of a different type - *Walls, Pills, etc.*. Though the map is drawn as a sprite, a 2D array represents the type of each tile, with characters storing and

tracking their coordinates to determine position within the map and world, both logically and when drawn, as well as any collisions. A custom method translates map coordinates to world coordinates for ease of drawing. Push Matrices are used throughout to ensure correct rendering.

User Guide

Compiling the Solution

Custom Makefiles for both Windows and Linux have been provided in the submission, which reference all required sources and dependencies (ie. LibPNG, enabling PNG images as textures).

The solution is compiled on Linux using:

```
$ make -f Makefile.linux pacman
```

If compiling more than once, it may be easier to setup a symbolic link:

```
$ ln -fs Makefile.linux Makefile
```

The solution can then be compiled using:

```
$ make pacman -B
```

Note: Optional *-B* flag forces a recompile. Compiling on Windows requires a UNIX-like environment - ie. MinGW. Using this, compilation commands are nearly identical; replace *Makefile.linux* with *Makefile.windows*.

Running the Game

Once compiled, the game can be run with:

```
$ ./pacman
```

This will create a *highscore.txt* file on starting if one does not exist already.

How to Play

Pac-Man is a deceptively simple game:

1. Eat pills and fruits while avoiding ghosts to score points
2. Pac-Man loses a life if caught by ghost
3. Advance to next level by eating all pills
4. Eat big pills to scare ghosts - eat while scared for bonus points
5. Eat several ghosts in a row for temporary score multiplier
6. Highest score is saved on game over

Figure 3: *Help-screen is displayed when game is paused*

Player controls:

- Move Pac-Man using arrow keys
- ESC key pauses the game
- ESC key from pause screen to quit
- Any key from pause or game over screen to resume/restart game

Though simple in concept, the game is far more challenging to play than it may appear - good luck and have fun!

References

[1] J. Pittman, "The pac-man dossier." http://www.gamasutra.com/view/feature/3938/the_pacman_dossier.php?print=1. [Accessed: 2016-12-23].

[2] C. Birch, "Understanding pac-man ghost behavior." http://gameinternals.com/post/2072558330/understanding-pac-man-ghost-behavior, 2010. [Accessed: 2016-12-28].

Word Count: 750 excl. headings & references. Refer to comments for more detail.