UNIVERSITY OF
KWAZULU-NATAL

INYUVESI
YAKWAZULU-NATALI

COLLEGE OF AGRICULTURE, ENGINEERING & SCIENCE:

SCHOOL OF ELECTRICAL, ELECTRONIC AND COMPUTER ENGINEERING

CM3 Project

ENEL3CC

17 May 2015

Yasar Mahomed Abbas	Dineshri Pillay	Kemintha Naidoo	Thegan Moodley	Lushen Rajaruthnam
213532608	213505249	213504804	213507271	213510010
Electronic Eng.	Electronic Eng.	Electronic Eng.	Electronic Eng.	Electronic Eng.

Abstract

Groups of five students were required to produce a 2D game using the C++ programming language in Visual Studio 2013. Students familiarized themselves with Allegro 5 in order to add the graphical and sound element to the game. The aim of this project was for students to display their ability to apply OOP (Object Oriented Programming).

This was a collaborative project and all contributions made by each person were sent to a repository on Github. The knowledge of C++ taught in the Computer Methods 3 course along with separate research, were used in order to see the success of this project.

A game called, Luigi: The Other Plumber, was chosen; which was based on the knowledge, graphics, and sound of the well-known, classic game; Mario Brothers. This game instead focuses on the brother which was forgotten, as he completes his journey fighting enemies through different stages.

Contents

I. Introduction

A small room sits at the very top of a smaller house; in it a young boy plays with sticks imagined to be toys. He looks out of the tiny window to the outside world and sees his brother on the driveway, he is yelling at his girlfriend again. The makeup runs from her eyes and onto her tear stained pink dress. The brother's red overall straps smack his chest with anger as they continue their fight as the young boy watches from above.

The brother drags the girl back into the house as she sobs trying to gain composure and the yelling now becomes muffled by the floor boards. The young boy sits on for they are now underneath him. The Brother than takes his frustration out on their pet ape like he always does, not so much a loved pet than a stress reliever in his mind. The brother cannot stand the fact of being alone so he keeps the girl with him to fight away his inner fears.

All the lights remain on throughout the night because the young boy is afraid of ghosts in the dark. The room at the very top of the house is in darkness, the young boy sleeps in the dark, and he craves company of any kind, alive or not. No one goes to the top room; they have forgotten about the young boy, the child takes comfort in his toys and his imagination, a safe place.

The next day the boy is woken by loud bangs and breaking of furniture. He looks to the window to see the ape running away with the brother's girlfriend. The brother then storms out of the door and yells in anger and pain, not for the loss of the girl, but for the fear of being alone. The young boy can feel the brother's pain. He has watched from above, the ill treating of a beautiful girl he became fond off and now she is gone. He knows the brother cannot face the treacherous path that lay ahead to find the girl, the fear of the bad creatures that will emerge on the journey.

The neglected young boy fears the same but in the dark he has learned to accept the fear and deal with it; something the elder brother never learned. The child puts on his green jumpsuit and blue overalls and leaves the house in the night, his brother is unaware.

The morning comes and the elder brother goes up to the top room to find an empty bed. His mind in a bad liquor soaked state; comes to some dangerous conclusions. Thinking that his younger brother was planning to rob him of all that he loves, the elder brother puts on his olive green jumpsuit and red overalls. He sets out on a mission bent on revenge against all who has wronged him.

A few years pass and none of the brothers have completed their quests, both now in adulthood and still continue their boyish missions. The young boy, now a man, has come a long way and his mission to restore balance in the world is proving more difficult but he pushes forward in the places he finds and the enemies that the ape now known as King Koopa has created. The young man now equipped with the knowledge of plumbing just like his older brother, continues forward on his quest, finding new challenges each day and overcoming them. He can't help think of his distant brother who is still at the house he left behind. Or is he?

The elder brother sits eating a red polka-dotted mushroom and his twisted mind gets tangled even more around the idea of revenge against his younger brother. The now old man Whispers to himself over and over again with pain and anger in every instance, "I will find you Luigi..."

This is the story through the eyes of the younger brother, the underdog, the brave one, the OTHER PLUMBER.

1. Software Development Methodology

An agile methodology was chosen as it allowed all members to collaborate in a dynamic way. A design process was followed, which began with outlining the game and the basic code that would be needed. This was divided amongst the group, who each worked on developing their allocated area.

Bug testing was done individually at first, in an isolated environment, for different portions of code. This allowed the member responsible for a particular piece of code to better detect the reasons for bugs in that code later on, when different pieces of code were integrated.

Each member followed a process of continuous iteration, with the code being improved for efficiency and simplicity.

The agile methodology proved to be successful as when working in a large team, it is easier to follow a dynamic approach of solving problems when they occur and implementing new ideas when they are found, rather than planning the whole process before beginning development.

2. Software Architecture

The code for the game was split between two files; a header file in which structures are defined, and a source file. The source file followed by the header file is outlined below.

2.1. Basic Declarations

Header files are first included. These consist of the header file in which structures are defined as well as the various headers for Allegro add-ons.

Defined next are the keys used in gameplay, a structure to hold properties of the current player and several variables related to general gameplay such as resolution, player position and size of obstacles.

Declarations for numerous methods that are used later then follow.

Lastly are initialisations for the audio samples to be used and for the images of the various game characters and obstacles. The images needed for the background, pause screen and start screen are also initialised.

2.2. Main Game Loop

2.2.1. Variables

Before the main loop, several gameplay variables are initialised such that they are “empty” and can be used in starting a new instance of a game being played. Allegro objects are used to create a display, peripherals are added as events, and add-ons are initialised. The audio samples and images initialised earlier are defined.

2.2.2. Events

The main game loop begins with actions being provided for keyboard events (a key being pressed or released), a “close display” event (which exits the main game loop), and a timer event (which occurs if a key is not pressed and the display is not closed). The actions triggered by keyboard events are seen on the next page in Tables 1 & 2.

2.2.2.1. Keyboard Events

Key	If game is being played	If pause screen is on
Up	If Luigi is on the ground, he jumps (with sound). If already in the air, he remains so.	The pointer moves up the list of options.
Down	No action.	The pointer moves down the list of options.
Right	D is set true	No action.
Left	A is set true	No action.
Space	Space is set true & a projectile is shot.	No action.

Table 1: Result of pressing a key

Key	If game is being played	If pause screen is on
Escape	The game is paused and goes to the pause screen, pause music is played.	No action.
Up	If Luigi is in the air, he starts to fall.	No action.
Down	No action.	No action.
Right	D is set false.	No action.
Left	A is set false	No action.
Space	Action showed in Figure 1 below.	The option currently pointed at is selected: a new game is started, a demo plays, or the main game loop is exited.

Table 2: Result of releasing a key

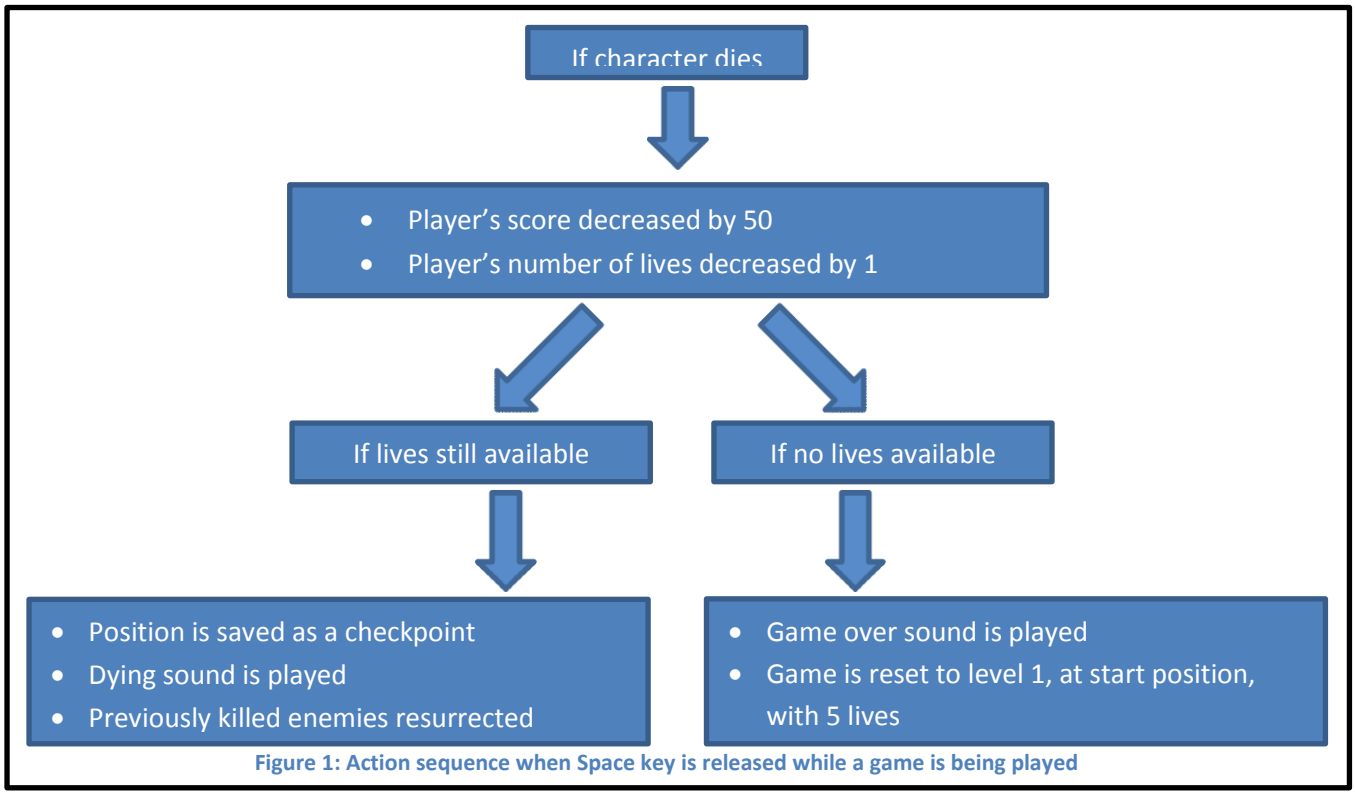

Figure 1: Action sequence when Space key is released while a game is being played

2.2.2.2. Timer Event

The timer event causes the following essential gameplay actions. If the distance covered by Luigi has increased, the score increases. The “jumping” and “moving character” sequences, seen in Figures 2 and below respectively, are also activated by the timer event. The “moving enemy” sequence moves the enemy character between two previously set boundaries.

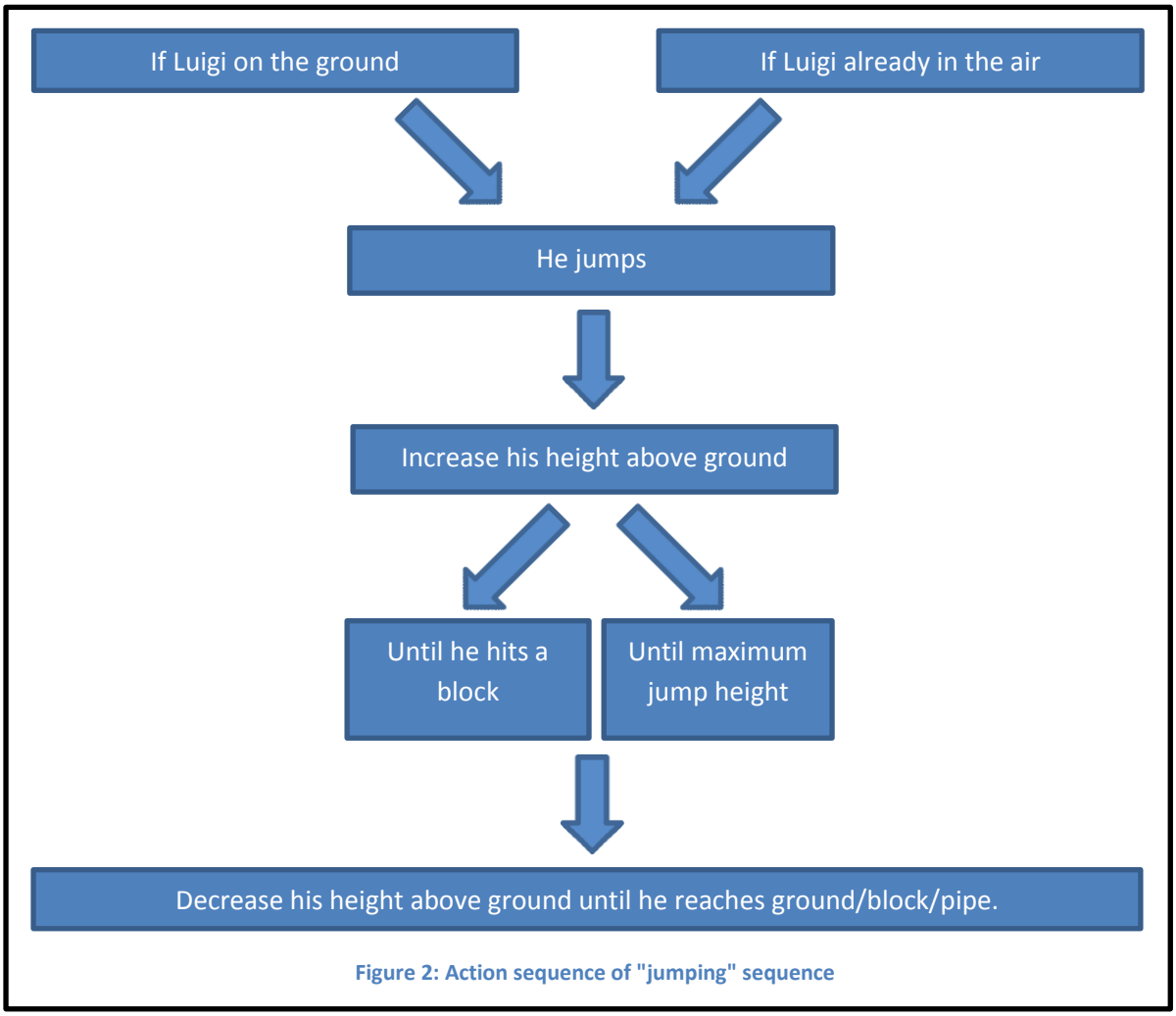

Figure 2: Action sequence of "jumping" sequence

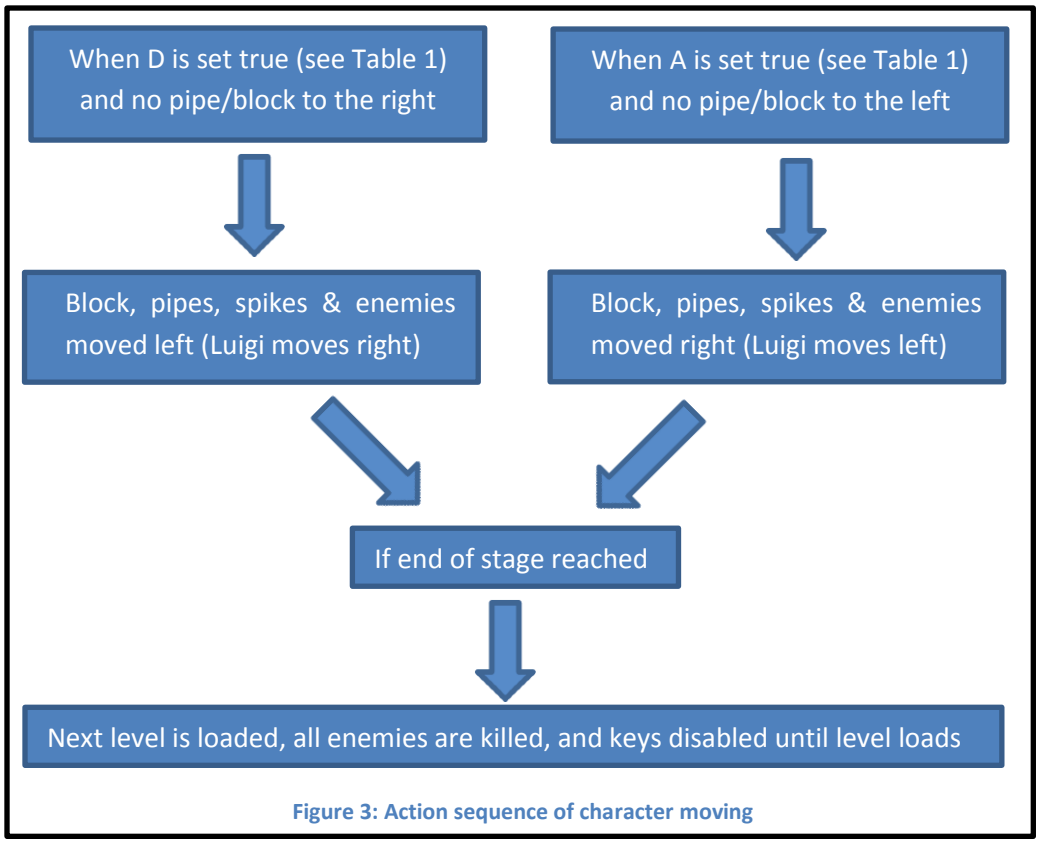

Figure 3: Action sequence of character moving

2.2.3. Text & Images

These main game action sequences are followed by text and image enhancements of the game using Allegro add-ons. The objects used in the game loop (spikes, blocks, pipes, enemies, projectiles) are created (drawn with Allegro), depending on the level. Text is added to display game information (i.e. “LIVES”, “SCORE”, and “POSITION”). This is also done for the pause screen: audio to be played and text and text and images to be displayed are added.

The main game loop is then exited, and objects of type Allegro that were created are destroyed.

2.3. Gameplay Implementation Methods

Brief descriptions of all the methods that are used in implementing the game are given below:

2.3.1. Creating a level

- initblock: For each level, the positions & other properties of the blocks are added.
- initPipe: For each level, the positions & other properties of the pipes are added.
- initSpike: For each level, the positions & other properties of the spikes are added.
- CreateEnemies: For each level, the positions of the enemies & other properties are added.

2.3.2. Blocks

- create: The block is drawn with an allegro command.
- block_limit: Checks to see player has exceeded the bounds of the block.
- box_right: Ensures player does not pass through block.
- box_left: Ensures player does not pass through block.
- noblock: Ensures player does not pass through block.

2.3.3. Pipes

- createPI: Pipes are drawn with Allegro command.
- checkpipe: ensures that player does not exceed the bounds of the pipe.
- pipe_right: Ensures player does not pass through pipe.
- pipe_left: Ensures player does not pass through pipe.

2.3.4. Spikes

- createSpike: Pipes are drawn with Allegro command.
- spikeCollide: The player dies if walks into/onto spike.

2.3.5. Projectiles

- initproj: Projectiles are initialised to a speed.
- drawproj: Projectiles are drawn with Allegro command.
- shoot: A shoot sound is played, and if enemy is hit, is killed.
- update: Allows projectile to move.
- HittingPipes: Plays a bumping sound if hits pipe.
- HittingBlocks: Plays a bumping sound if hits block.

2.3.6. Enemies

- setEnemies: Bounds of enemy's movement is set.
- DrawEnemies: Enemies are drawn with Allegro command if alive.
- StartEnemies: Enemies are added.
- Collision: Player score increases and enemy dies if hit by projectile.
- CollideEnemy: Player dies if collides with enemy, dying sound is played.
- ISEEDEADPEOPLE: All the enemies are resurrected.
- Apocalypse: All the enemies are killed.

2.4. Game Control Methods

- GenerateBG_G: The game's background is drawn.
- GenerateSTART: Either the pause or start screen is drawn depending on what is required.

2.3. Header File

This file consists of structure definitions for blocks, pipes, spikes, enemies and projectiles to save their various properties.

3. Team Contributions

Each member of the group was assigned to a specific task for the making of the game.

Yasar Abbas Mahomed: the basic movement of the character and assigning keys for each movement.

Dineshri Pillay: the creation of enemies such as the spikes, Goomba(mushroom) and Koopa(turtle).

Kemintha Naidoo: the death sequences and image addons for the entire game.

Thegan Moodley: designed the outline and structure of each of the three stages.

Lushen Rajaruthnam: the sound and graphic effect addons for the entire game.

Although each member had different tasks, the whole group was involved in putting the entire game together to ensure it ran smoothly

4. Game Play Instructions

The following keys are used for game play:

A – Left

D – Right

W – Up/Jump

S – Down

Space – Shoot

The basic strategy of this game is to get through to the end of the stage while trying to avoid getting killed by enemies (by shooting at and killing them) and obstacles such as spikes. The pipes and bricks can be used to navigate over obstacles by jumping onto them.

5. Conclusion

Students were enlightened about the skills and techniques needed to create a successful game. This project not only taught individuals about the art of software development but also taught students about team building.

The goals were: to work on software development in a team environment, to use a substantial library which is not a part of the language, to use version control in a collaborative environment, use the ability to apply a top down approach to the design of software and to show the use of good software engineering practices.

During the process of the development of the game, all goals were met and at the end, the game was a complete success.

6. Consulted Resources

1. http://alleg.sourceforge.net/docs.html
2. www.codeitive.com
3. https://wiki.allegro.cc
4. www.gillius.org/allegtut/
5. www.loomsoft.net/resources/alltut/alltut_index.htm
6. http://www.anothergames.com/book/allegro?page=1&title=allegro-framework-1
7. http://fixbyproximity.com/2d-game-development-course/

7. Relevant Github Links

Main repository: https://github.com/Lushen2015/CM3-PROJECT-2015

Collaborators:

https://github.com/Lushen2015

https://github.com/TheSlumo

https://github.com/dineshri

https://github.com/Thegan

https://github.com/kemintha