RESEARCH ARTICLE OPEN ACCESS

Game Development using AI

Eynas Balkhair

Date of Submission: 11-12-2024 Date of acceptance: 23-12-2024

The project is a game that involves AI the game idea is similar to the idea of Pac man game that includes a player and some enemies. The idea is about a user that moves around or straight to reach a specific goal and the enemy(CPU) lies as obstacles in which the CPU also moves and tries to touch the user that are playing the CPU might track the users movement in order to move and stop the user. For the player to win a certain level he has to finish all the fruits (apples) that exist at that current level without losing all his life. A player can only lose a life by getting in contact with the enemy. The following game will be implemented using Game maker studio software.

I. Background Information

Introduction:

Introduction of Game Development using AI

Creating a game using AI is a process in which the computer needs to act smartly according to the situation or by adapting to the user’s motion, the way in which the system thinks or the logic process behind the action itself is not important as the action is (Schwab, 2009). To create a game in which users will be interested to play AI should not be intensely used in earlier levels as if the system is so smart in which the player can never win will make the player lose interest in playing the game(Jamieson, 2015). Games using Ai is concerned with the behavior of other players and its more ‘artificial’ than ‘intelligent’ . In an AI game the following titles are used and studied which are Decision-making , perceptions, FSM(including the states)and prediction (Kehoe, 2009).

About the game

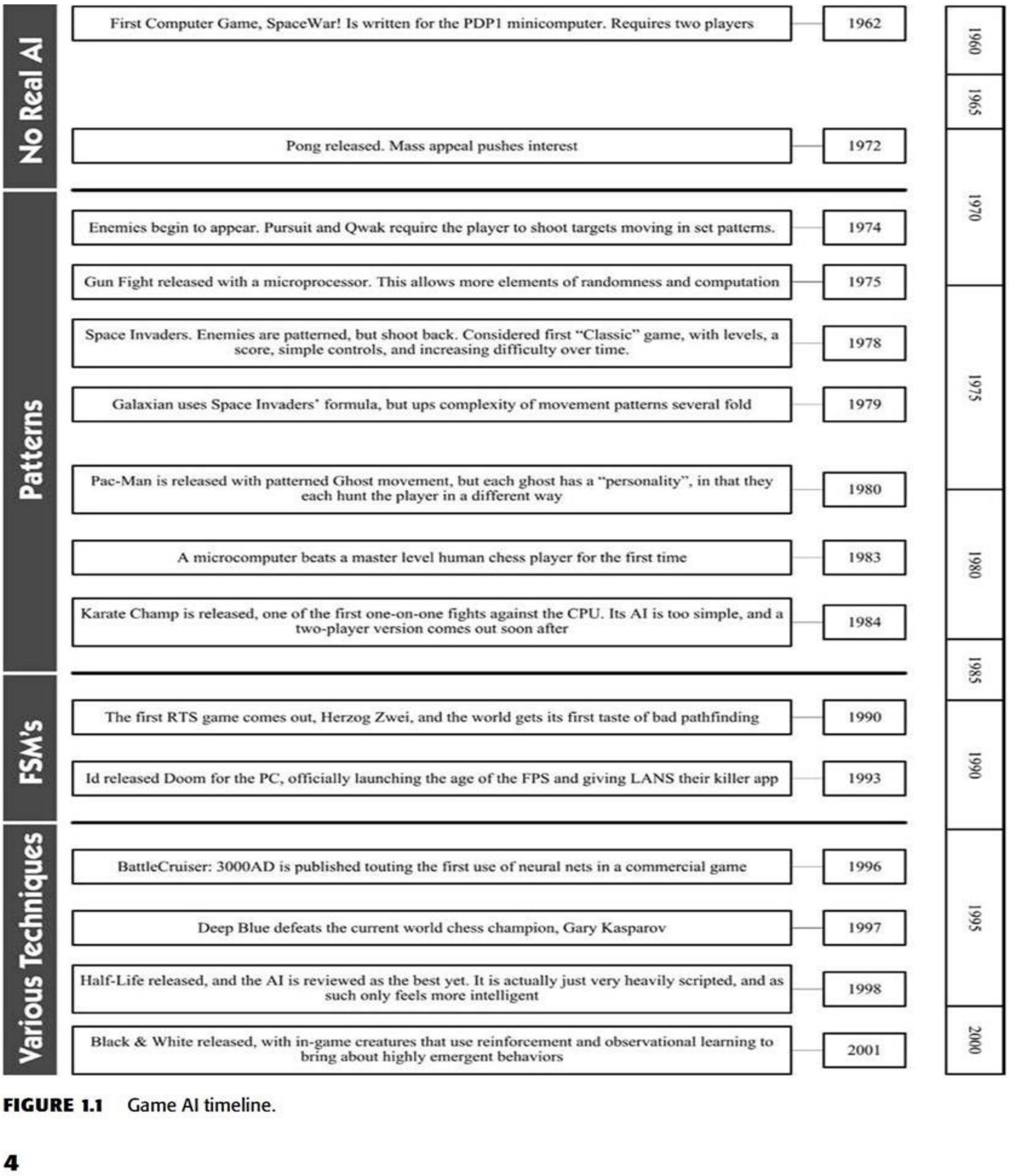
The game is like Pac man in which there is a user collecting stuff, there are barriers that lies in front of the user and by tracking his motion or by viewing him the barriers tries to attack the user and prevent him from completing his mission. Also the enemies (CPU player) can destroy one of the lives of the player if it came in contact with him. The game makes people either they playing or will play this game aware about what will happen and how to avoid the obstacles to win the game

Importance of Levels

My opinion for this topic that it is useful because users become aware of the problem related to obstacle avoidance and starts knowing how to solve this problem. For beginner users who are not familiar enough of this game will develop skills of recognizing the obstacles and preventing interactions with it. For example if I created this game so I will have a percentage of the users just losing in the game after that they will know how fight the other user and win the game. The beginner users in the game learned from advanced users in this game and these advanced users learned from previous trials of the game they played and they was losing on it most of the time.

History:

In the beginning of video games there were no AI included. The first game ever was created in 1958 by William Higinbotham which was called " tennis for two " it was simply a human vs human game which has nothing to do with AI .Same as well gone with the second Video game which was the first game to run in a computer was space wear by Steve Russell and it didn't include AI as well. Atari was the first game development company who invented an arcade game called "computer space" in the 1970's and after that game designers started to include AI within the video games. After human vs computers video games people started to believe that computers are actually thinking. After AI the way that computer acted made it impossible for human being to script the behavior of non-human objects. The computer's behavior was based on simple calculations for how the computer should behave which made the human players have the feeling that they're actually playing vs human being. In "pac man " the ghosts where chasing the user in a different way as if they have different personality. This made the users being ill used that there are 4 to 5 enmities chasing them not separate soft copies of the same computer. By the late 1980's computers entered to home so console games started to develop. Due to modern processor's high ability games became more complex at that time. Even computer designers developed high-quality graphical performance. Real time strategy was firstly developed at the late 1980's as a new genre. This genre meant to players high competency and entertainment. AI was challenging and high demanding requirements which made RTS frames more outstanding and more demanded and it was one of the reasons for the development of this new field. Games with AI have passed a long way ever since the 1950's .At the moment we are still witnessing an AI evolution of AI games and its even joining into the game development process.



First Computer Game, SpaceWar! Is written for the PDP1 minicomputer. Requires two players — 1962

Pong released. Mass appeal pushes interest — 1972

Enemies begin to appear. Pursuit and Qwak require the player to shoot targets moving in set patterns. — 1974

Gun Fight released with a microprocessor. This allows more elements of randomness and computation — 1975

Space Invaders. Enemies are patterned, but shoot back. Considered first "Classic" game, with levels, a score, simple controls, and increasing difficulty over time. — 1978

Galaxian uses Space Invaders' formula, but ups complexity of movement patterns several fold — 1979

Pac-Man is released with patterned Ghost movement, but each ghost has a "personality", in that they each hunt the player in a different way — 1980

A microcomputer beats a master level human chess player for the first time — 1983

Karate Champ is released, one of the first one-on-one fights against the CPU. Its AI is too simple, and a two-player version comes out soon after — 1984

The first RTS game comes out, Herzog Zwei, and the world gets its first taste of bad pathfinding — 1990

Id released Doom for the PC, officially launching the age of the FPS and giving LANS their killer app — 1993

BattleCruiser: 3000AD is published touting the first use of neural nets in a commercial game — 1996

Deep Blue defeats the current world chess champion, Gary Kasparov — 1997

Half-Life released, and the AI is reviewed as the best yet. It is actually just very heavily scripted, and as such only feels more intelligent — 1998

Black & White released, with in-game creatures that use reinforcement and observational learning to bring about highly emergent behaviors — 2001

FIGURE 1.1 Game AI timeline.

4

Fig 0: Game AI timeline

Schwab, 2009. Retrieved from: https://mehmetakifsonmez.files.wordpress.com/2013/12/ai-game-engine-programming.pdf

Motivation:

One of the motives of the project is to develop an understanding of AI by using it in the implementation of the game. Another motive could be the development of some skill regarding game development and the creation of a better understanding for AI concepts. In addition the use of game maker studio would be learned in this project by developing the game using this software.

II. Methodology

Introduction:

In order to let the reader understand our system architecture step by step, we will tell exactly what we are going to do in the steps to create a game about the user(s) will play against the CPU. We will present in the game how the user can win and avoid the bad situation by following some strategies.

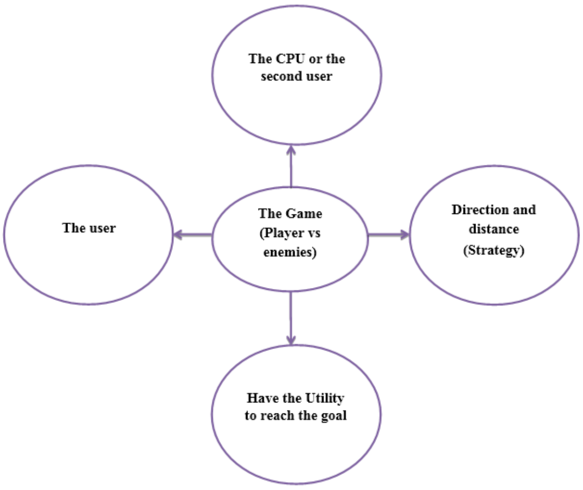

The CPU or the second user
The user
The Game (Player vs enemies)
Direction and distance (Strategy)
Have the Utility to reach the goal

Fig 1: Summary of what the game includes

In the human VS computer we are trying to make the user fight the computer using a specific program called: game maker. The game maker provides flexible game development environment in addition to artificial intelligence utilities.

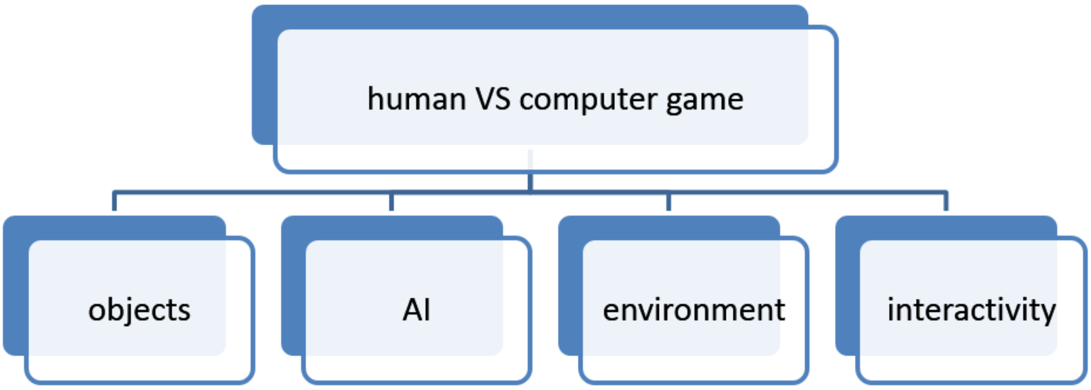


Figure 2: what makes human VS computer game

Human VS computer game has different kind of components in our design there will be objects , environment , interactivity and Artificial Intelligence as shown in figure 1.

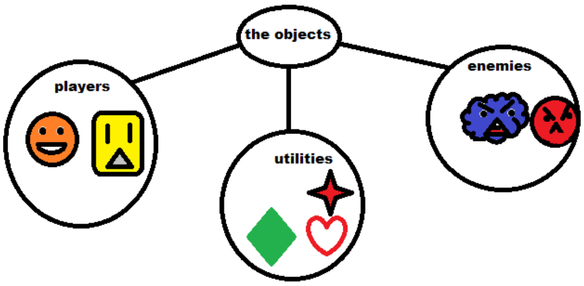


Figure 3: different objects and primary design of their characteristics.

The first step in our game is crating the objects that are to be used in the game. There are three types of objects : player(which the user will interact with using input devices) , enemies (with the CPU will take their control of) , the utility objects (which the user should collect before the enemy reached to him). The user should do two things : collect the utility objects , avoid the enemies . There are variety types of utility the user should collect. Each one should has specific price and distance and time . The standard utilities are the less beneficial but they are easy to get and nearer to the user , the more the benefit of the utility object the less timing to appear and the farther distance from the user. The last thing in this stage of design is the enemies the enemies are suppose to be objects that are managed and controlled by the CPU. The enemies are set to be chasing the player objects trying to touch them. As long as the enemy touches the player before it completes catching all its utilities objects the enemy wins. Otherwise the enemy looses the game and the player will move to the new stage.

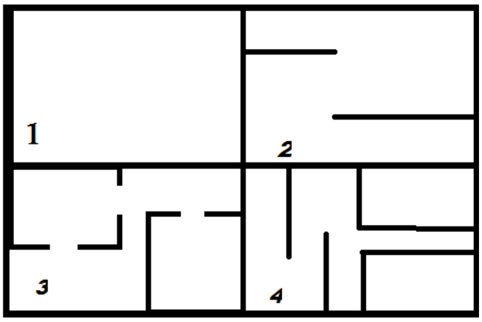


Figure 4 : primary puzzle design it shows that level 1 is less complex puzzle than level 2 but level 4 is the most complex.

Our next stage is the environment. The environment in our game differs through the stages. However were are not designing more than three levels in our game the different stages the more complex puzzles as shown in figure 3. The environment in the first stage should be plain environment where no blocks or puzzles available. In the second phase there should be some blocks in addition to another enemy in order to make the game more interesting and difficult. The last stage (if available) the environment should have more blocks –more complex puzzle – and there should be time limits . If the time ends the player looses which rise difficulty of the game . The environment should contain some moving objects and sounds to make it lovable and not boring . The moving objects should be transparent which means that they have not influence on the game environment(neither the players nor the enemies) .The environment should be well designed because the environment is one of the factors that makes the game more demandable and attractive. Attractiveness is how interesting your design and how it will be rated in the market. Different levels of attractiveness are the major reason to successfulness or fail. In our game and since we have limited resources we will try to make our game as attractive as possible. The will be materialized in association with how the game looks like , how is the scenario of the game and how is the difficulty per stage.

Game process general explanation

The object to play (player) with is created and the enemies are created on the grid, the objects are drawn with a certain x and y coordinates. The enemy is enabled to move randomly (means according to a certain direction) to approach the player. Walls are inserted between the player and the enemies, the enemy's moves randomly at the beginning to approach the player. A path finding algorithm is added to develop Ai for the enemy to try to approach the player in accordance to the algorithm not randomly. When a certain path is determined for the instance to move through the instance would operate well in the absence of many obstacle or when obstacles are not generated throughout the game, when the path is not specifically followed but a start point and an finish point is determined and a path is generated accordingly this might be a better solution. The game is deterministic and perfect information is given, the environment at the beginning is well known and any action will certainly cause a certain result.

Every event that occurs by the player causes an action by the CPU object if an event and action was set for every action the player takes. For instance if the player moves down the CPU in reaction moves up or if the player moves left the cpu moves right, forming actions for events is a good procedure but it could be read by the player as the game proceeds. In a more complex formulation of the game in which there are more walls, obstacles this procedure might deem ineffective in which actions that are in response to the events are so simple with no intelligence used by the system. Although in the path finding algorithm the instance finds its path to the user by reacting to the user's position or movements that causes his presence in the new position. The transition function in which the instances have depends on the events caused by the player as the player moves from a state to another.

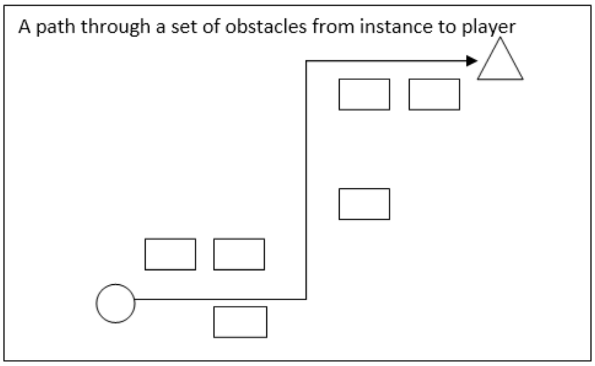


Fig 5: Path checking

The following tree shows the set of possible actions taken by the instance from a certain position x and y to a certain non-moving player (The grid is shown with the square instance and a circle player).

Game Tree (showing player’s movement and enemy)

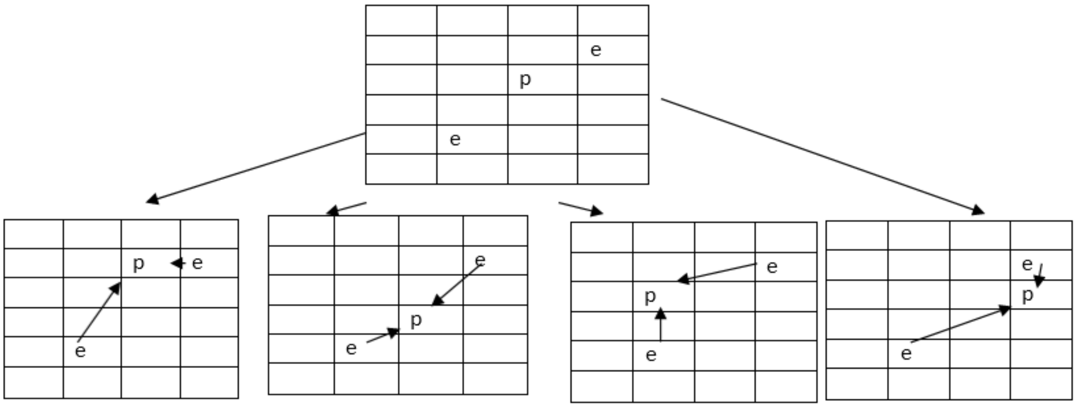


Fig 6. Showing player’s movement and enemy movement according players position the game truncates when enemy approaches player

Implementation

What took the most of the time is learning phase. The implementation took us some effort because we needed to learn while implementing the game.

The first thing we did is we created the spirits needed for objects we modified them and animated the enemy. Secondly we had to make the player object move our movements are the basic ones right , left , up and down. The player changes it face (flips) as long as it changes its move. Then there is the dot objects and the wall objects and the lives objects.

Rooms:

3 rooms were used:

The first is for the menus, to allow the user to choose whether to play or exit the game at the beginning.

The second room is the first level of the game which contains one instance of enemy 1.

The third room is the second level of the game which contains 2 instances of enemy 2.

Scripts

Some scripts were drawn others were loaded. The scripts center were used

Objects

Liveobjects how many chances left for the player in a single game before losing.

Wall objects: solid objects that construct the puzzle.

Dot (apple):objects raise the score of the player to make him close to win

Background

A background image is created and a spirit is created using an image of grass.

Enemy 1:

The enemy in the first room uses a move toward built in function to approach the player.

Enemy2

Enemy was drawn in the spirit as three images movement through images was done to show movement of mouth.

Code used for path finding:

```
grid=mp_grid_create(0,0,room_width/32,room_height/32,32,32);
path=path_add();
mp_grid_add_instances(path,object_wall,0);
mp_grid_path(grid,path,x,y,object_player.x,object_player.y,1);
path_start(path,2,"",0);
```

The first line is for the grid creation 0, 0 are initial left and top positions the second two values are the total number of cells in the grid as the height of the whole room is divided by the height of a single cell and the same applies for the width. 32 and 32 are the cell width and height (the snap values of the room).

The second line is for simply creating a path and assigning this path to the variable path

The third line is used to make the walls when the player approaches them it makes path finding not allowed. The grid is used in this function and precise is set to false.

The fourth line creates a path using the grid from the starting position (start state) of the enemy using the x, y the Object is currently in to end point (goal state) which is the x and y of the player, the last value is set to 1 or true to allow a diagonal path or movement.

The fifth line uses the path, sets the path speed to 2, no end action, and the path is set to relative.

Displaying Messages

A message is displayed to the user for the following purposes:

To inform him if he won or failed

To ask him if he wants to restart the game, restart the current level, or exit the game.

Object Code from game maker:

Information about object: object_player

```
Sprite: spr_pacman
Solid: true
Visible: true
Depth: 0
Persistent: false
Parent:
Children:
Mask:
No Physics Object
Create Event:
set the number of lives to 3
execute code:

score = 0;
win =0;

Alarm Event for alarm 0:
execute code:

game_end();
Step Event:
execute code:

if score =220
{
   win =1;
   alarm[0]= 60;
}
Collision Event with object object_wall:
start moving in directions 000010000 with speed set to 0
Collision Event with object object_enemie:
set the number of lives relative to -1
jump to the start position
Collision Event with object object5:
set the score relative to +10
Collision Event with object e2:
set the number of lives relative to -1
jump to the start position
Keyboard Event for <no key> Key:
start moving in directions 000010000 with speed set to 0
Keyboard Event for <Left> Key:
start moving in directions 000100000 with speed set to 4
set the sprite to sprite8 with subimage 0 and speed 1
Keyboard Event for <Up> Key:
start moving in directions 000000010 with speed set to 4
set the sprite to sprite6 with subimage 0 and speed 1
Keyboard Event for <Right> Key:
start moving in directions 000001000 with speed set to 4
set the sprite to spr_pacman with subimage 0 and speed 1
Keyboard Event for <Down> Key:
start moving in directions 010000000 with speed set to 4
set the sprite to sprite7 with subimage 0 and speed 1
Other Event: Outside Room:
wrap in both directions when an instance moves outside the room
Other Event: No More Lives:
display message: you FAIL
if the player does say yes to the question: Would you like to restart this level?
    Restart the current room
else
    end the game
Draw Event:
draw the lives at (32,32) with sprite sprite10
Draw the instance
if win is equal to 1
    display message: you WIN
    if next room exists
        if the player does say yes to the question: Would you like to continue to next level?
            Go to next room
        else
            end the game
    else
        if the player does say yes to the question: Would you like to restart the game?
            restart the game
        else
            end the game
```

____

Information about object: object_wall

```
Sprite: wf
Solid: true
Visible: true
Depth: 0
Persistent: false
Parent:
Children:
Mask:
No Physics Object
```

____

Information about object: object_enemie

```
Sprite: sprite_enemie
Solid: true
Visible: true
Depth: 0
Persistent: false
Parent:
Children:
Mask:
No Physics Object
```

Create Event:
jump to the start position
Step Event:
start moving in the direction of position (object_player.x,object_player.y) with speed 1
Collision Event with object object_wall:
bounce precisely against solid objects

____

Information about object: object5
Sprite: sprite9
Solid: false
Visible: true
Depth: 0
Persistent: false
Parent:
Children:
Mask:
No Physics Object
Collision Event with object object_player:
destroy the instance

____

Information about object: e2
Sprite: sprite11
Solid: false
Visible: true
Depth: 0
Persistent: false
Parent:
Children:
Mask:
No Physics Object
Step Event:
execute code:

```
grid=mp_grid_create(0,0,room_width/32,room_height/32,32,32);
path=path_add();
mp_grid_add_instances(path,object_wall,0);
mp_grid_path(grid,path,x,y,object_player.x,object_player.y,1);
path_start(path,2,"",0);
```

Collision Event with object object_wall:
bounce not precisely against solid objects

____

Information about object: play
Sprite: sprite13
Solid: false
Visible: true
Depth: 0
Persistent: false
Parent:
Children:
Mask:
No Physics Object
Mouse Event for Left Button:
Go to room rm_Game

____

Information about object: object6
Sprite: sprite14
Solid: false
Visible: true
Depth: 0
Persistent: false
Parent:
Children:
Mask:
No Physics Object
Mouse Event for Left Button:
end the game

YouTube video link that shows implementation and testing of the program
https://youtu.be/i1d0LqfYniY

III. Discussion and conclusion

In conclusion, a game similar to Pac man was created using two levels in which the game idea is about a player that tries to collect all the fruit an enemy that tries to reach the player and decrease the number of his lives till he loses. The enemy acts as a minimiser and the player acts as a maximizer in the following game. In addition, the environment in which the game works is deterministic.The game involves AI as it includes a path checking algorithm in its second level. The creation of the following game was significant in learning basic game development using game maker and the ability to use very simple AI in the game.

References:

[1]. Game Maker Tutorials - Artificial Intelligence, Pathfinding [HD]. (2012) Retrieved from:

[2]. https://www.youtube.com/watch?v=g7SwdmecICU

[3]. Jamieson, D. (2015, 3). Making AI fun: when good English is good enoughRetrieved Lecture Notes Online Web Site:http://gamedevelopment.tutsplus.com/articles/making-ai-fun-when-good-enough-is-good-enough--cms-23460.

[4]. Kehoe, D. (2009). Designing Artificial Intelligence for Games (Part 1) Retrieved Lecture Notes Online Web Site:https://software.intel.com/en-us/articles/designing-artificial-intelligence-for-games-part-1.

[5]. Mark, F. (2010,5). Artificial intelligence (video game). Retrieved Lecture Notes Online Web Site:https://en.wikipedia.org/wiki/Artificial_intelligence_(video_games.

[6]. Mark, A. (2014,3).[GMS-S] Using Mp_grids To Create AI. Retrieved Lecture Notes Online Web Site:

[7]. http://help.yoyogames.com/entries/50594316--GMS-S-Using-Mp-grids-To-Create-AI.

[8]. Rohde, M. (2014). GameMaker: Studio For Dummies. John Wiley & Sons.

[9]. Schwab, B. (2009). AI game English programming [PDF document]. Retrieved Lecture Book Online Web Site:https://mehmetakifsonmez.files.wordpress.com/2013/12/ai-game-engine-programming.pdf.

[10]. Yoyo, G. (2014). Game maker studio. Retrieved Lecture Notes Online Web Site:http://docs.yoyogames.com/source/dadiospice/002_reference/movement%20and%20collisions/motion%20planning/mp_grid_path.html.

[11]. Yoyo, G. (2014). Game maker studio(video game). Retrieved Lecture Notes Online Web Site:http://docs.yoyogames.com/source/dadiospice/002_reference/movement%20and%20collisions/motion.%20planning/mp_grid_add_instances.html