3rd International Conference on Research Developments in Applied Science, Engineering & Management
The Indian Council of Social Science Research (ICSSR)
Panjab University Campus, Chandigarh (India)
AEM–2018
Conference World
19th August 2018 www.conferenceworld.in ISBN : 978-93-87793-43-9

Implementation of Roll It Ball Game in Computer Graphics by Using C++ IDE

Lee Boon Xuan[1], Neekita Sewnundun[2],Wong Woon Suen[3]

BSc (Hons) in CS, Asia Pacific University, Kuala Lumpur, Malaysia

ABSTRACT

Gaming development is the interesting role in computer graphics which covers process of analyze, design and implement the game based on various rules and follows some conditions.Here this paper focusing on displaying the ability to implement a few core game development concepts which are the pick-up, dynamic movement, ray-casting, timer, dynamic texture, head-up display and graphical user interface. However this game will be feasible as each of them serves as function and player point of view this game will be taking less amount of time. In future this game will be developing in client server technologies.

Keywords:Analysis, Dynamic Movement, Rendering, Implementation, Testing

1. INTRODUCTION

This project is about developing a simple computer game by using Unreal Engine as the main tool. C++ project is chosen over blueprint-project. It should be reflecting the concepts of dynamic texture, ray casting, rendering and interaction between objects in the game. The game should also be devised in such a way that it has the ability to adapt or make rational selection between possible development approaches, algorithms, methodologies, tools, techniques and solutions. The concepts of the game contribute to making the game feasible as each of them serves as a function to the game which otherwise would make any game non-executable.

1.1 PICK-UP

Pick up is the ability to pick up an item in the game. This concept is displayed when the character moves over to touch the box of the item, thus destroying the item. Pick up is typically used for acquiring item in game. Items are to be set on the path of the character so when the player moves the character, it picks up the items.

1.2 DYNAMIC MOVEMENT

Dynamic movement is the ability to constantly move or oscillate back and forth in a pre-directed path. Dynamic movement can be used in getting player’s attention on a certain item or as an obstacle to the player. Dynamic movement is the run-time behaviour of the game-player system whereby he/she can as well control the motion of the character or simply objects in the game having designated motions assigned.

1.3 RAY-CAST

Ray-casting is a technique in which a beam no matter visible or invisible is projected from an initial coordinate to a final coordinate while tracing every single point along its trajectory. Ray Casting is known as a rendering algorithm which maps the start point to which it is assigned to the destination of the final point and can also be used to destroy objects when hit [1].

1.4 TIMER

Timer is the ability to enforce limitation on certain actions so that they cannot be used over and over again. Timer is typically known as cooldown of an action. The usage of timer is very broad as it dictates what actions can be performed and what actions cannot be performed in a given time. A warning timer might show the player how much time is left as to a showcase timer will only show how much time the player took to arrive to the destination or to finish the game.

1.5 DYNAMIC TEXTURE

Dynamic texture is the ability to alter the intensity of the texture of an object in game. The varying intensity of texture given the distance of player to the object gives off a realistic feeling of a world. This concept can also be used to trigger certain events in a game and the parameters of the objects' dynamic textures can be controlled with respect to the game makers' needs and requirements [2].

1.6 HEAD-UP DISPLAY (HUD)

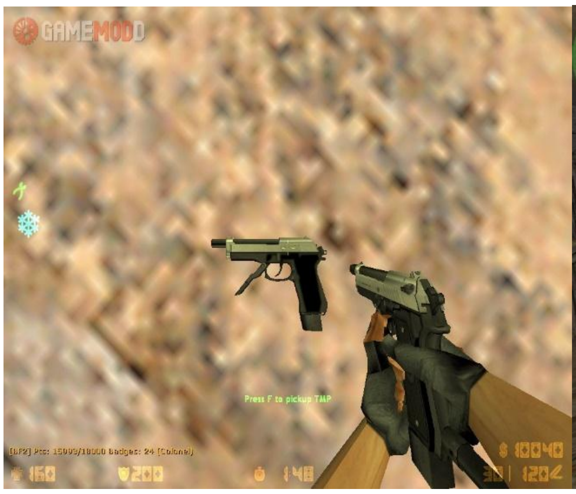
In gaming, head-up display (HUD) is a method to transparently display information of the game to the player without the player switching to any other tabs or windows. HUD is always being displayed from the start of a game until the end of a game. It is also the main thing that attracts the player's attention while playing and also serves as a plus to the interface. Popular examples of HUD include health, mana points, score and time remaining [3].

1.7 GRAPHIC USER INTERFACE (GUI)

Graphic user interface (GUI) is an interface displayed on screen to provide interaction with player. GUI in a simple game most commonly provides options for start game, end game and pause game which allows the player to directly interact with the game be it to pause it, resume, quit or simply restart the game. The GUI helps players to better navigate through what the Game has to offer.

2. LITERATURE REVIEW

The video game that the team researched on was Counter Strike 1.6. This game was one of the series of Counter Strike. It is the most popular online team shooter in history of first person shooter. It was originally developed and published by Minh "Gooseman" Le and Jess Cliffe in the year of 1999. Then they were hired by the Valve Corporation and the intellectual property of Counter Strike belongs to Valve Corporation. By studying Counter Strike 1.6, the team came to realise that it had exactly implemented some of the core concepts of a video game discussed in the above. Counter Strike 1.6 achieved "pick up" through walking over a weapon or bomb which was lying on the ground. When the player overlapped with the item on ground, given that the player had a free slot for that type of item, the item would be destroyed and then spawned to the player.Next, Counter Strike 1.6 had timer implemented. This could be seen from the rate of fire itself. Certain guns had higher rate of fire and vice versa. This rate of fire could be achieved by manipulating the timer so that the player could only fire the next shot after a fixed amount of time from the previous shot. The Magnum Sniper Rifle had the most noticeable timer set up since it required the longest cool down time after a shot. In addition, a manipulatable timer was given to every round. Timer was also applied after planting a bomb at a bombsite, shown in "Fig.1, Fig.2, Fig.3, and Fig.4".



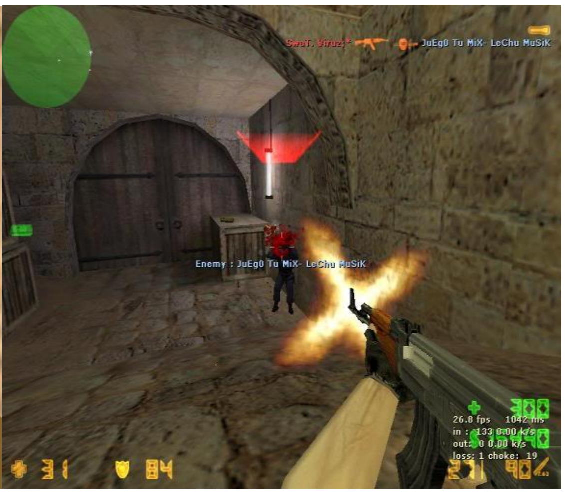


Figure.1 Pick-up weapon [4] Figure.2 Fire with ray-cast [5]

HUD could be seen almost any time when playing the game. Every player typically had a mini map on the top left corner, a small bomb icon on the left if the player was carrying bomb, a decisive health display at the bottom left, an armour value, time remaining, bullets remaining, and amount of money possessed. These important HUDs could be seen in the figure below

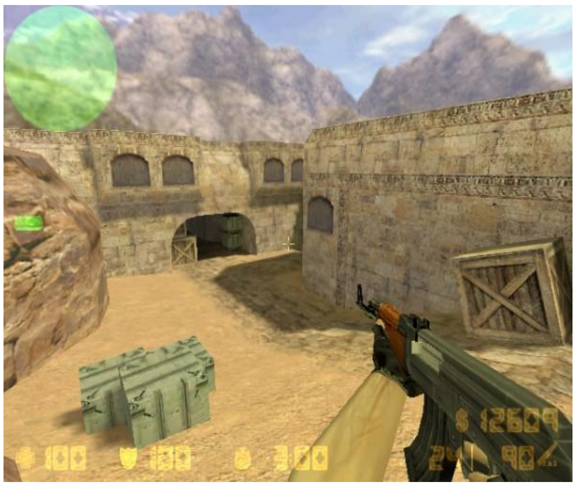


Figure.3 In game HUD [6] Figure.4 CS 1.6 Main Menu [7]

Unfortunately, Counter Strike 1.6 was an old game that came out when computer hardware was very limited to poor graphic card. Therefore, dynamic movement and dynamic texture could hardly or not have been applied to the game since these features would require better hardware in order to be processed. The screen had to be refreshed at a higher rate. Hence, the team concluded that all these concepts that they were going to use in the projects have been around for quite some time. These concepts as displayed by Counter Strike 1.6 prove to be very useful as well as applicable to a lot of fields depending on the creativity of the developer on how to apply these concepts into their project.

3. DESIGN AND METHODS

3.1 WINNING OR LOOSING CONDITIONS

There are a few criteria must be fulfilled by the player to win the game.
Winning criteria:

- Player reaches the destination point.
- Player has collected all coins of the stage.
- Time for the stage has not ran out.

In the following, in case any of these conditions is met, the player loses immediately.

Losing conditions:

- Player's life reaches zero.
- Time for the stage has ran out.

How player will die?

- Player falls off from the map.
- Player is being hit by laser.
- Player runs into a hazardous obstacle.

3.1 TESTING

Software Testing is the process of identifying errors or any missing requirements within a system to crosscheck whether after the construction the functional and non-functional requirements of the system are met. [8]Game Testing as for this project, is much similar to software testing differing only to actually playing the game as well to find bugs in the project execution so that these bugs can be overcome before any actual deployment is done. Testing has been one of the major steps taken by developers which adds to the project's value. Testing can also be regarded as proving the efficiency of the game. The testing comprises of steps to find bugs and defects within the project that could be either inside the codes or related to the interface itself with the sole goal of increasing the quality of the project and to make sure that the game actually responds to the requirements set and the functions of the project are fully executable. Testing allows a game to either be launched or guides the developers to debug and correct the defective parts of the output. [9]For this project, the team decided to carry out both Black Box Testing and White Box Testing as it was necessary to comply with all the functions and make sure that all the codes respond to the functions assigned. Also, Black Box Testing would make sure that the interface was well configured and comprised of no unnecessary buttons which might not be of any use. The team believed in the tests consolidating the whole project and understood the importance that the outputs would ensure that any bugs be debugged before the actual submission, the project's functionalities and non-functionalities be fully and freely executable. It was important to get through both tests to make sure all requirements of the assignment were fulfilled such as if the ray casting works, the coins rotate, the ball rolls, the character picks up the coins along the path, the timer within any part for any particular functions works and also if the interface is user friendly as the game was designed for audience of all groups. Therefore, the two types of testing would ensure that the project was fully consolidated in and outside the box, shown in "Fig.5".

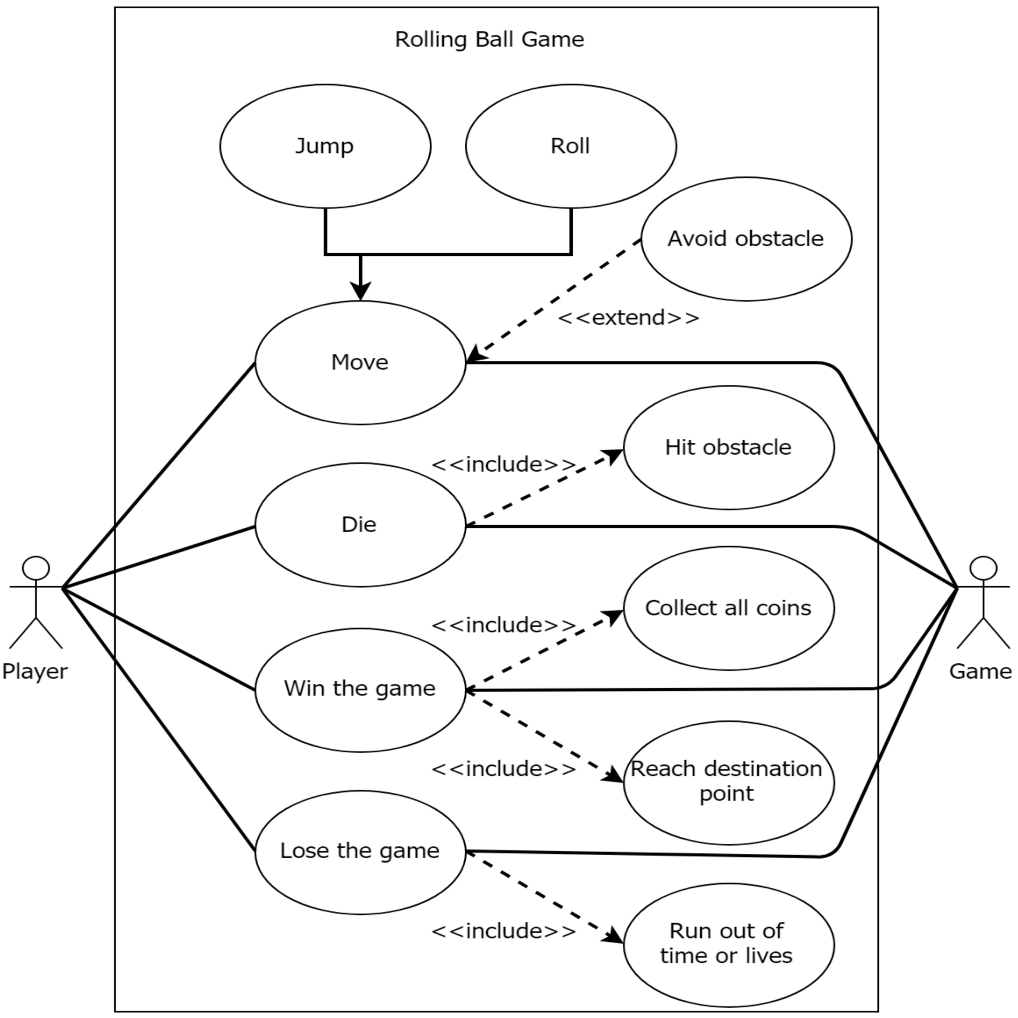


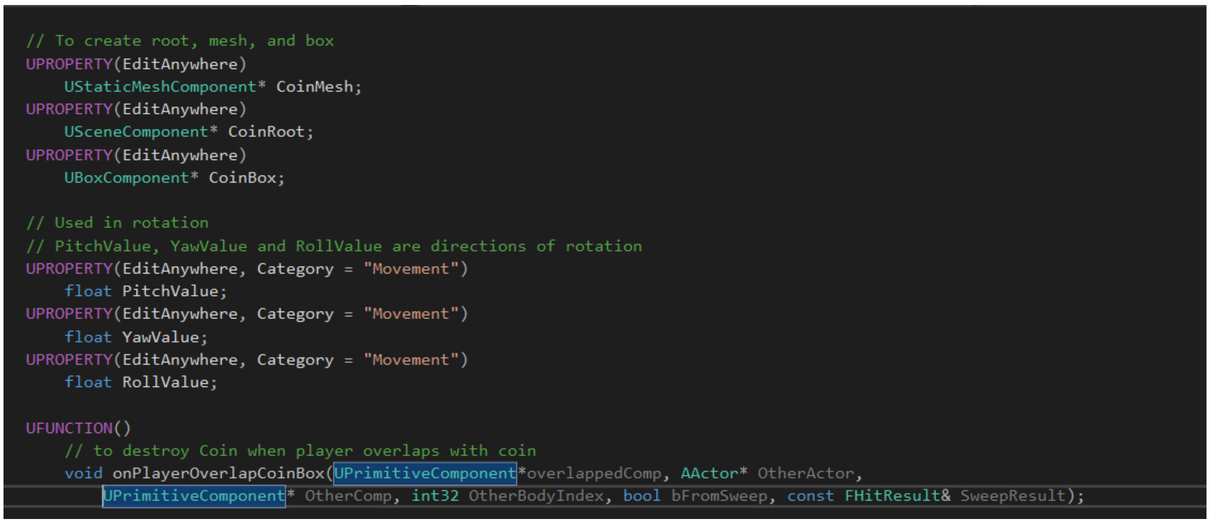
Figure.5 Use Case Diagram of the game [10]

4. IMPLEMENTATION

In this section, in depth technical explanation as well as detailed implementation of these concepts into the game is to be discussed

4.1 PICK UP

Pick Up is a concept that allows the player to pick up an object on the level. In this game, Pick Up concept is implemented in Coin actor to allow player to pick up coin in the level in order to progress in the game. To implement the concept, a Root, Mesh, and Box must be created. A Mesh is used to give the coin actor a texture, and Box is used to detect if the player collides with the coin. Then, a player overlap function will be used to indicate that player gets the coin by removing the coin and increasing the player coin counter, shown in “Fig.6”.



```
// To create root, mesh, and box
UPROPERTY(EditAnywhere)
    UStaticMeshComponent* CoinMesh;
UPROPERTY(EditAnywhere)
    USceneComponent* CoinRoot;
UPROPERTY(EditAnywhere)
    UBoxComponent* CoinBox;

// Used in rotation
// PitchValue, YawValue and RollValue are directions of rotation
UPROPERTY(EditAnywhere, Category = "Movement")
    float PitchValue;
UPROPERTY(EditAnywhere, Category = "Movement")
    float YawValue;
UPROPERTY(EditAnywhere, Category = "Movement")
    float RollValue;

UFUNCTION()
    // to destroy Coin when player overlaps with coin
    void onPlayerOverlapCoinBox(UPrimitiveComponent*overlappedComp, AActor* OtherActor,
        UPrimitiveComponent* OtherComp, int32 OtherBodyIndex, bool bFromSweep, const FHitResult& SweepResult);
```

Figure 6.Coin.h [11]

In coin header file “Fig..7” the variable CoinMesh, CoinRoot, and CoinBox are created to store the Mesh component, Coin component, and Box component. Then, a function called onPlayerOverlapCoinBox is created to define what to do if player enters the box.

4.2 SCREENSHOTS OF ACTUAL GAMEPLAY

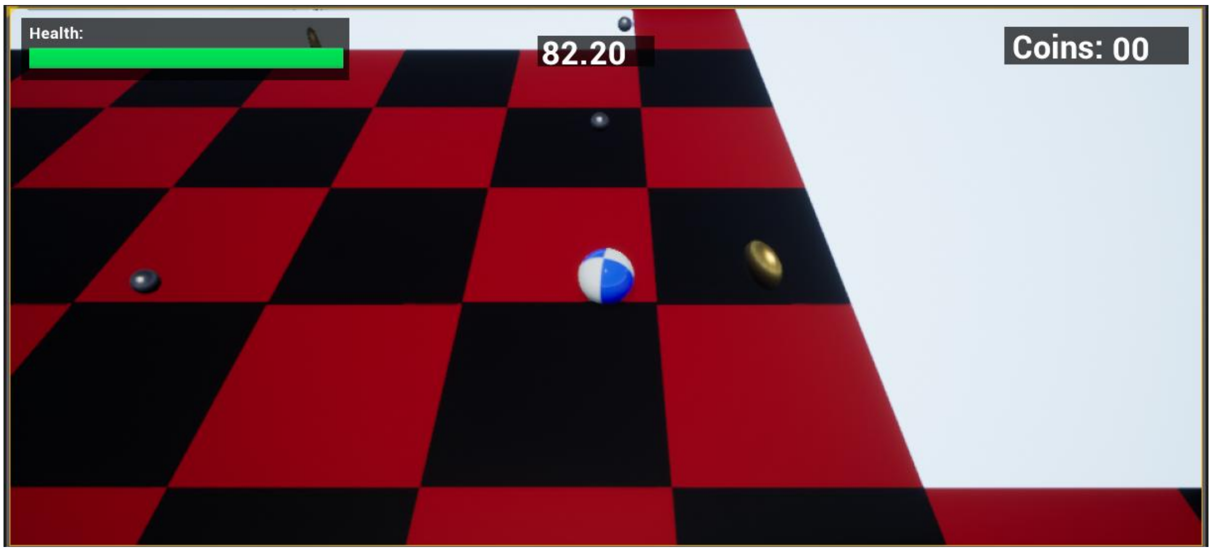


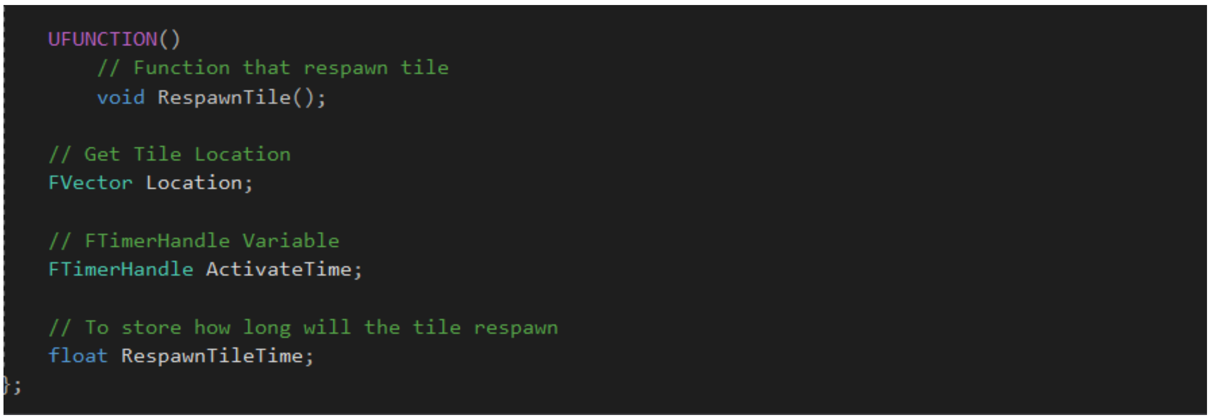
Figure 7. Approaching Coin [12]

4.3 DYNAMIC MOVEMENT

The concept of Dynamic Movement is an object that keeps moving during the game. In this game, Dynamic Movement is implemented in coin to make the coin look not as dull as the coin will be moving up and down while rotating itself. The concept is also implemented to make the laser rotate to stop the player from progressing.

4.4 TIMER

Timer is used to activate a function after a certain amount of time. In this game, the Timer concept is used in Timer Laser to activate the laser and deactivate the laser. It is also used in Disappearing Tile to respawn the tile that has been destroyed, shown in “Fig..8”.



```
    UFUNCTION()
        // Function that respawn tile
        void RespawnTile();

    // Get Tile Location
    FVector Location;

    // FTimerHandle Variable
    FTimerHandle ActivateTime;

    // To store how long will the tile respawn
    float RespawnTileTime;
};
```

Figure 8. DisappearingTile.h [13]

4.5 DYNAMIC TEXTURE

The concept Dynamic Texture is that an object changing its texture when a certain condition is met. In this game, the concept is used to make a tile change its texture and then disappear when the texture completely changed so that player is forced not to stand too long on the tile, shown in “Fig.9”.

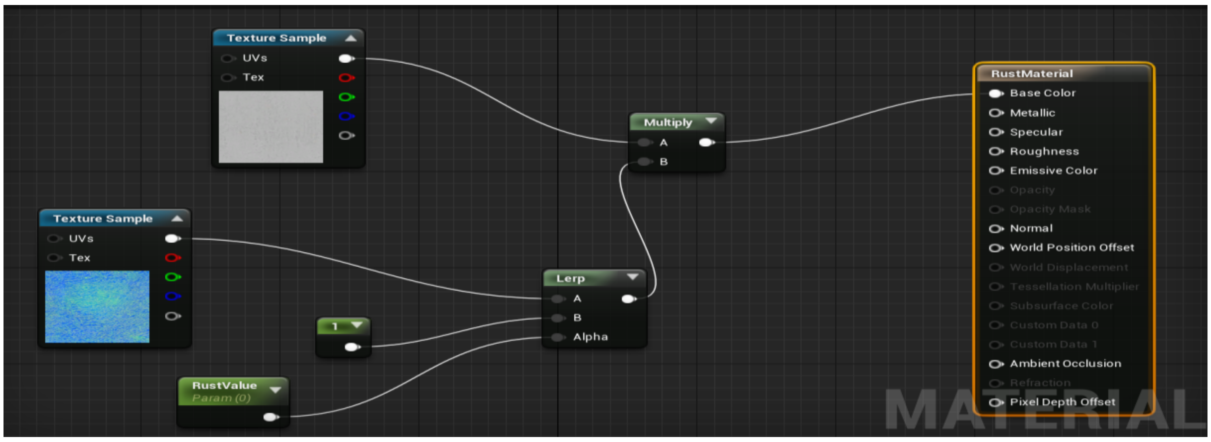


Figure 9. RustMaterial Blueprint [14]

4.6 GRAPHIC USER INTERFACE (GUI)

Graphic User Interface (GUI) is a user interface with graphic that allows player to interact with the game. In this game, GUI is used to display and allow player interaction on Main Menu, Winning Screen, and Losing Screen, shown in "Fig.10".

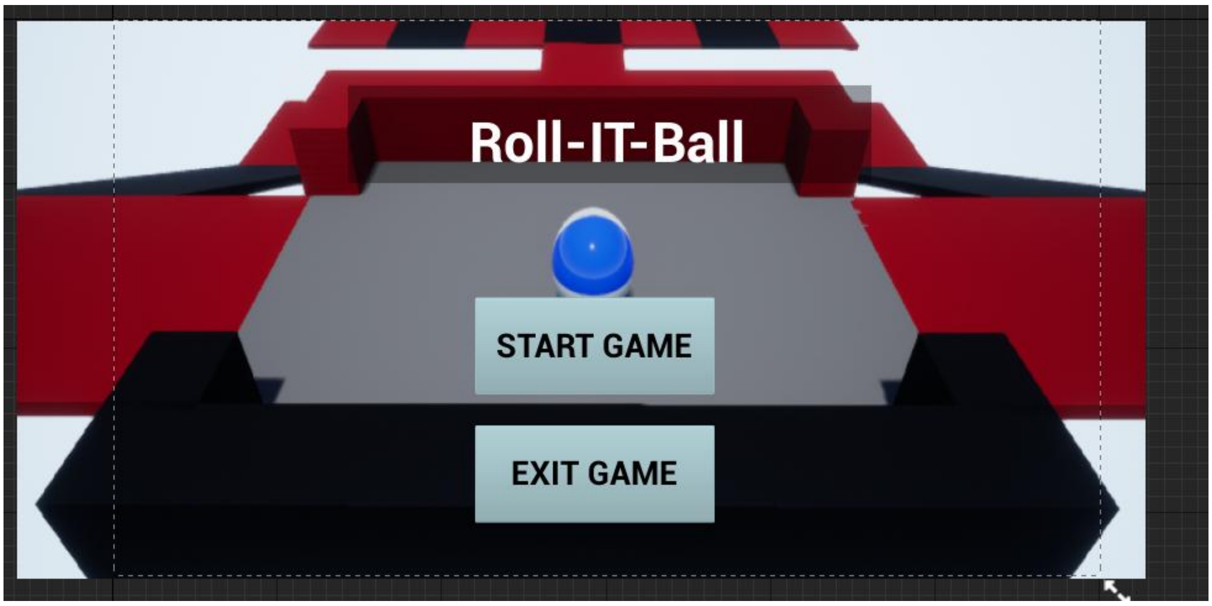


Figure 10.Main Menu Widget [15]

5. ADDITIONAL FEATURES

Throughout the game development the team has decided to implement a few extra features, so that the game become not-so plain-looking.

5.1 FAN

Fan is a feature that blows player who steps into an invisible box. To make a space blowing wind, the idea to the solution is borrowed from Dynamic Texture. When player enters a box, a flag is set and when player leaves the box, the flag is unset. Then an action is triggered when the flag is set.

5.2 SPINNING

It is the ability of rotating an actor. Dynamic Movement, since it is actually a modification of Dynamic Movement. Spinning can prove useful in drawing attention as well as used as an obstacle like what the team has done in Coin and RotatingLaser.

6. CONCLUSION

In general, this project was a success. Every concept required was beautifully implemented in a meaningful manner. This achievement was brought forth from the arduous effort and the rhythmic teamwork of the group. No pain no gain. The team covered each other's strength and weakness. All the blood, sweat and tears were paid off fabulously.A feature that allows player select difficulty of stage. In hard mode, clearing conditions and obstacles in the stage will be made much more difficult. For example, tiles will disappear faster, laser will spin faster or fire at a higher rate, more coins to be picked up, spawning of monster, fewer number of lives and last but not least, time limit for the round will be reduced.A server that allows multiplayer in the game, so that many players can play together or compete against each other. This concept can give player the opportunity to whether to face the solo challenge or team up and play with friends or just random players all around the globe.

7. ACKNOWLEDGMENT

The authors would like to share gratitude to Mr Umapathy Eaganathan, Lecturer in Computing, Asia Pacific University, Malaysia for the constant support and motivation which helped us to participate in this International Conference and also for journal publication.

REFERENCES

[1] Woop, S., Schmittler, J. and Slusallek, P. (2005). RPU. *ACM Transactions on Graphics*, 24(3), p.434.

[2] Doretto, G., Chiuso, A., Ying, N. and Soatto, S. (2003). *Dynamic textures*. [online] Vision.jhu.edu. Available at: http://www.vision.jhu.edu/reading_group/dorettoCSW03ijcv.pdf [Accessed 12 May 2018].

[3] Wilson, G. (2006). *Off With Their HUDs!: Rethinking the Heads-Up Display in Console Game Design*. [online] Gamasutra.com. Available at: https://www.gamasutra.com/view/feature/130948/off_with_their_huds_rethinking_.php [Accessed 12 May 2018].

[4] Bullit, Vince, Gooseman (2017). *Beretta M935 by FrenchyNeo*. [online] GAMEMODD - Mods for Games. Available at: https://www.gamemodd.com/cs/skinsweapons/tmp/717-beretta-m935-by-frenchyneo.html [Accessed 10 May 2018].

[5] yolasite (n.d.). *Counter Strike Official Game*. [online] Available at: http://counterstrikeofficialgame.yolasite.com/ [Accessed 10 May 2018].

[6] LinuxGSM. (n.d.). *csserver: Counter-Strike 1.6 – LinuxGSM*. [online] Available at: https://linuxgsm.com/lgsm/csserver/ [Accessed 10 May 2018].

[7] Yrrep, L. (2017). *HLDS Counter Strike 1.6 [Linux]*. [online] Instructables.com. Available at: http://www.instructables.com/id/HLDS-Counter-Strike-16-Linux/ [Accessed 10 May 2018].

[8] The Economic Times. (2018). *Software Testing*. [ONLINE] Available at: https://economictimes.indiatimes.com/definition/software-testing. [Accessed 10 May 2018]

[9] Rido Ramadan, R.R, (2015). Development of game testing method for measuring game quality. *In Data and Software Engineering*. Bandung, Indonesia, 26-27 Nov.2014. Indonesia: IEEE. 1-10.

[10] Wong, S. (2018). Use Case Diagram of the game. [PNG image]

[11] Lee, X. (2018). Coin.h. [PNG image]

[12] Lee, X. (2018). Coin.cpp. [PNG image]

[13] Lee, X. (2018). Approaching Coin. [PNG image]

[14] Lee, X. (2018). Dynamic Texture. [PNG image]

[15] Lee, X. (2018). MainMenu Widget. [PNG image]