FPS GAME DEVELOPMENT USING ARTIFICIAL INTELLIGENCE, RAGDOLL PHYSICS & SPATIAL AUDIO FEATURE

[1]LAKSHYA BHATT, [2]VARUN SRIVASTAVA, [3]RAJ SAXENA, [4]SARTHAK PREENJA, [5]SNEHAL D CHAUDHARY

Department of Information Technology,
Bharati Vidyapeeth (Deemed to be University) College of Engineering, Pune, India.

***Abstract*: This paper basically describes about the employment of computer science in FPS games and the components that involve under it to make it a wholesome game. we chose this topic due to a personal motivation; we play computer games a lot most of the time they're either RTS or FPS games. One of the aims of our contemporary First-Person Shooter (FPS) design is to produce a fascinating experience to the player. Our game development design works to give a full Player immersion, then, is also alleged to be a primarily emotional and demonstrated, from an awareness of 'being in and part of' reality to 'being in and part of' virtual specified, within ideal case scenario, Virtuality becomes substituted for a reality. This paper actually examines the role of a sound in enabling such event in a real FPS games it should be achieved sonically through an attention on mimic instead of realism. The paper utilizes and develops the previous add which a conceptual framework for the look and analysis of run and gun. Using GAME AI to make game NPC learn a perfect reaction to an opponent (player). Ragdoll physics is being used in animation effect which is usually used as a replacement for static death animations in a video games for better enhanced experience. In this paper we are working on game which will fulfil most of the needs for the players which was not been delivered by old games (i.e. 2D pixels games, etc).**

***Keywords*: FPS creation, Ragdoll physics with Verlet Integration, artificial intelligence, sound sonification, 3D animation function.**

1. INTRODUCTION

From pixel gaming to a next gen evolutionary games our gaming world comes so for in terms of development. our gaming industry is no longer a zone for a certain group or consumer segment. With the arrival of mobile gaming and enhancements to hardware utilized in playing these games, gaming has become a viable sort of entertainment for players from all backgrounds and ages. This switch to mainstream has also meant an increase in revenues generated by the industry with about US $9.5 billion generated in the United States in 2007, 11.7 billion on 2008 and 25.1 billion in 2010. This paper generally based on shooting survival (first person shooter) game, where we use "Unreal game engine" to create a game. Unreal Engine is world's most open and advanced real-time 3D creation tool. Continuously evolving to serve not only its original purpose as a state of art game engine, today it gives creators across industries the freedom and control to deliver good quality content, interactive experiences, and immersive virtual environment. Taking our personal interest into consideration and therefore the scope of the paper for this particular course, we defined the research question to be: "What are currently used techniques of computer science in specific Real-Time Strategy and First-Person shooter game titles?" the employment of the phrase "specific game titles" is in fact a touch not understandable and with those arises question we work our game to make it better, also applied techniques to make it realistic and playable. To create such a thing, we use game engine and we are using UNREAL game engine which will have so many features and options for developer to help them give a shape to their game the way they wanted. In this project we are working on simple FPS (first person shooter) apocalyptic game where our playable character in game can be seen through player(user) perspective.
The Story element of our game follows as the main player is striving to survive and making sure that he didn't get attacked by zombie's club. As our player is the only one survivor left then the only option for him is to gear up and fight. Our game will provide many resources in game to fight zombies but make sure to use it wisely. Once zombie detects you, they will go after you until you hunt them, that's what our game is all about. Now to create such amazing game many building blocks is being used to make it complete content.

2. GAME DEVELOPMENT

To create a survival video game, we need to work on unreal engine. At first, we download and install unreal game engine in our system. Here we take UNREAL engine instead of others, since it comes so many features and options and it provides fluidity in any task when performed. Now to give a briefing about unreal engine we are mentioning some features of it for better understanding. So, for a start let's run the editor and it will ask the name of a project. Give a name and let's get started, after initialize game engine at first it will show you the default interface where you can see the level editor, it provides the core level creation functionality for Unreal Editor. This is where levels are created, viewed, and modified mainly by placing, transforming, and editing the properties of your object (Actor). In Editor, the scenes during which you create your game experience are generally referred to as Levels. You can consider a level as a 3D environment into which you place a series of objects and geometry to define the place your players will experience. There are various items used to work over, items may be world geometry, decorations in the form of Brushes, Static Meshes, lights, player starts, weapons, or vehicles. Items are added when is typically defined by the actual workflow employed by the level design team. the default interface consists of Tab bar and menu bar, toolbar, modes, content browser, viewport, world outliner, details. Now to make the game we first need the object or Actor(character) to establish there in workspace. But we will show what other things is being used which was earlier not been a part of an old games like how our game in unreal engine collaborates with our artificial intelligence, sound sonification and so on. We are mentioning all such things in this paper.

A) FPS CREATION

FPS also known as first person shooter/survival now a days gaining so much popularity and gamers found it fascinating so much that even in time of quarantine 2020, games which are built on basis of FPS earned millions in just a month. So, here we are established first person environment to give realistic setup features in our game to a player in our game engine. In a first-person shooter, the player takes part in an exceedingly 3-dimensional fight simulation from a first-person perspective. the target often is to make certain waypoints and achieve objectives set by the developer. During single player mode this involves eliminating enemies (NPCs) that try and make it harder for the player to attain an objective.
To create first person angle camera in a game which can be seen from player perspective, we need to put main camera which can be found in Modes tab after you implement your main camera you can found it in component tab and in content browser. We need to fit that camera right at the chest or at the place where you can see your game environment from player perspective then, put directional light to show everything clearly or to give sun effect there, that's up to developer to decide to use it or not.

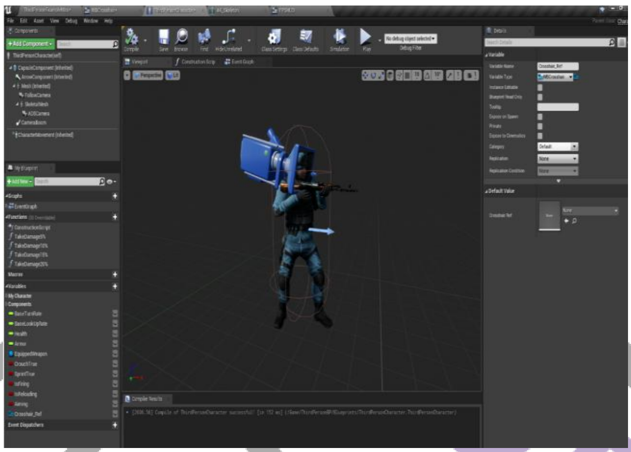

Fig 1. This figure shows the main camera is adjusted to give first person perspective look to a game.
First-person shooters often include NPCs that are controlled by means of a finite state machine which are made inside an event graph, an event graph is been part of every function and controls that are used for your NPC and main player in game, such features are available exclusively in unreal engine where instead of coding in C++ you will need to plot a state transition figure for you game objects and according to that state diagram we can control action and effects of our characters in game. To give different controls to your avatar such as How your player react to a certain event, what action to be take or different functions is been taken up by **BLUEPRINT ,** blueprint holds all function, variables, classes within it; mostly blueprint is a complete gameplay scripting system which define all objects or classes in the engine , it holds section of event graph and animation graph to work over characters action and skeleton mesh and animation sequence.

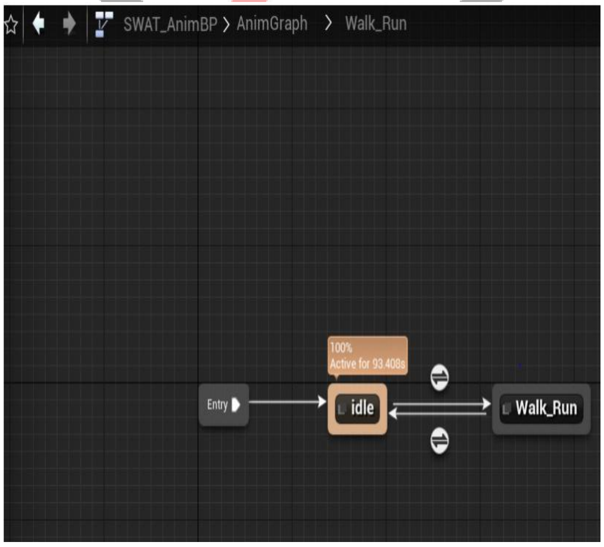

Fig 2. It shows the state of our character in game in an event graph.

Above fig. shows an event graph will show state of your main character in FPS game where it shows control of a player which is been controlled by user by creating state transition diagram in an event graph. We also show that how we look in main player look in his hands from first person perspective and how those hands work. Instead of code we draw state diagram to make function and controls for certain events blueprint will hold it all and result to machine is been shown in the form of C++ form code. Here in this paper we discuss about how fps game looks, how we make first person perspective game in our game engine and what main character react in FPS version of game. In an FPS form of game, you can see your surroundings like you are totally seeing with your eyes, you can interact with objects and tools in game when you run it like how you interact in real world. Fig 3. Will show you how you look from first person perspective through game.

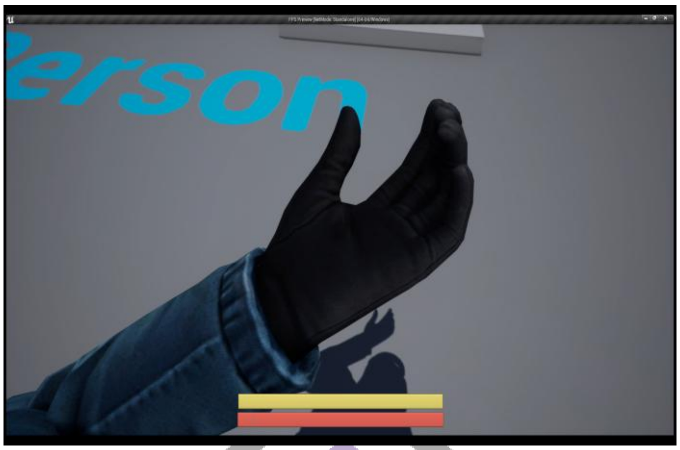

Fig 3. It shows how it looks from first person survival mode in a game.

B) **SOUND SONIFICATION IN GAME**

This game provide a definition of sonification because the use of a sound generator to rework non-audio data into sound so as to facilitate, or perhaps provide new, understanding of that data .For example muzzle effect sound of gun, damage done to your main player, etc those things can be explained because of sound only in a game. the aim of sonification is to watch and comprehend data which could otherwise, and in another form, be difficult to notice and understand. it's possible to feature further layers to the present particular conceptual onion or by suggesting that sound will always carry artefacts of the user space and equipment that's inhabited and utilized by the player so as to partake within the game which this adds another real space with the other real and illusory spaces already contained within the sound.

This paper will describe about how sound sonification is applied in game development. In level editor means your default interface where we are working, at first download the sound of all the portions where you need to produce sound in our game we need foot step, crouching gun muzzle effect sound so, for that we need to download that file and then create a new file in content browser and name that file as "audio" where you can import your all sound files easily.

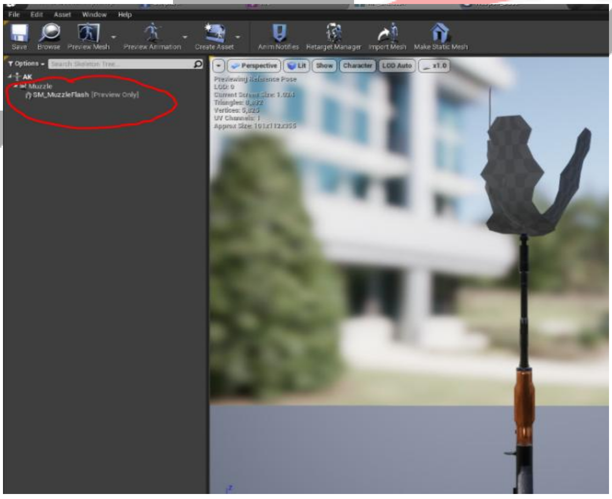

Fig 4. Shows audio file of muzzle is attached to AK47

Now create cue for each audio and then go to blueprint tab or you can check in "my blueprint" location at left side of interface where you can go to event graph of animation of that object in which you want to put a sound like walk action for character we will introduce footstep sound in that event graph and connect that audio in that state transition diagram. We are connecting audio file to that function or action where it is needed i.e. gunshot sound should be allocated with gun trigger animation only. Sometimes audio files are been supported with animation when we collect it from asset of unreal engine there, we do not need to add audio in event graph. There are setting to adjust timing of sound and alter the sound too so, that animation could fit with sound sonification properly. Sound sonification used a feature called Resonance Audio in game engine. This function of unreal engine will give us an option to configure and control spatialization, room effects, and occlusion settings in real time using the Resonance Audio plugin.

C) **3D ANIMATION FUNCTION**

Animation graph is being used in unreal engine to work over animation of game characters and objects. engine take care of animation and ragdoll physics both, if we need to differentiate between them then we can say that animation effect is puppet and ragdoll physics is giving life or realism to that puppet. There are some things to think about when creating game animations, and lots of differences and additional techniques required. In games you would like to make animations that reply to gameplay, storytelling, mixing both together and handling some other things. Most animations are short and as they're combined to every options so that they can create the character movement with attitude and conviction. In brief, the animations created help translate what the player would really like to try to within the world. Animation plays a major role in game development it takes care of bones structural movement in characters of game and how a particular object reacts to certain types of events. In our game project we first get our character rigid structure or other ragdolls from asset store or you can get from internet where animation parts are associated with that rigid bodies.

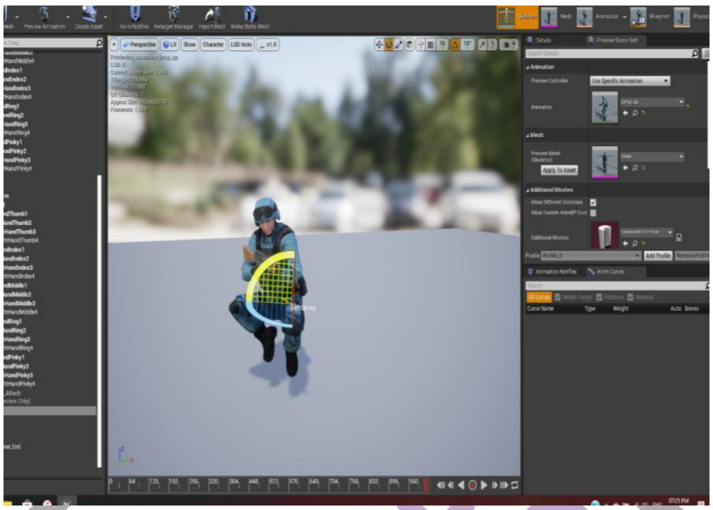

Fig 5. Shows hierarchical view of certain animation to adjust its skeletal features

When download those characters files, go to content browser and check those files name them by animation structure or it's up to you to decide the name. Then open that files you can check animation of those structures and easily you can attach your animation file with blueprint of animation graph of your character where you want to fit that animation that' what we did in our project also or you can work on blender to work on animation also. But it is not that easy as it seems because animation is not been provided with auto recognizing feature for how to react to certain event and act according to it, our action doesn't match the scenario. so, for that we will open that part of animation where we need to work on for example (check fig.5) like we need to make some adjustments in jumping action controls where we need to give adjustment to animation of hands and legs of ragdoll in such a way that the player jumps like a real human and for that go to that animation file and select "character" tab and from their select bones where we work on skeletal structure of a character's body. we adjust body structure of game body in such a way that our game and animation physics work parallel with our control of our games. We work on time count of animation, time lapse and so on.

D) **RAGDOLL PHYSICS WITH VERLET INTEGRATION**

Death in video game is usually a disappointment to many gamers usually because they lose some progress because of it. It means the tip of a combat round, the tip of grade and maybe the loss of minutes (or hours) of unsaved gameplay achievements effects all. But in games from an earlier era, death wasn't just boring — it absolutely was a graphical disappointment, too. the death in earlier pixel gaming and or in 3d games the death of a character in game is like, suddenly when enemy health goes to zero it just got vanished or enemy just turned into nothingness in a game which is generally as a gamer seems as a weird thing to out in a game. Since the first days, digital death has changed in amazing ways. Everyone knows that games have gotten violent, with untold gallons of blood and chipping bones being animated everyday across the world. But the realism that is needed of slumping, dead bodies has changed dramatically, too, thanks in large part to ragdoll physics. Ragdoll physics could be a category of procedural animation that displays human-like figures with more realistic motion. Sometimes the effect is strangely accurate. Other times the results are often overemphasized to the purpose of silliness, with arms and legs and structure flopping and twisting like, well, a ragdoll that consume alcohol. In today's physics function things look totally different—there are fighting games where the player controls one portion of the body of the fighter and therefore the rest follows along. In this paper we are describing what physics we applied to our characters in our game development. At first, when you download any asset file which have ragdoll body with it, they generally come with animation and ragdoll physics options too. So, check for all these files in content browser, go to your zombie file location where you can find zombie character. Now click on that file and create physics asset on your game. Here it will give you options like to care of Bone size of ragdoll and so on adjust those setting and after that go to your weapon blueprint and in that event graph delete the destroy target action because it is a reason why your enemy because of low health vanished so instead of that we use set simulate physics and now go back to aiming part of event graph of weapon where you find Simple AI (it gives AI to our character) where we can connect simple AI with target Mesh and that connects with set simulate physics. So, overall, what we are trying to do here is giving our gun instruction to not make target vanish but with applied physics it will push body backwards because of the collision between gun and ragdoll the impact will push body backward.

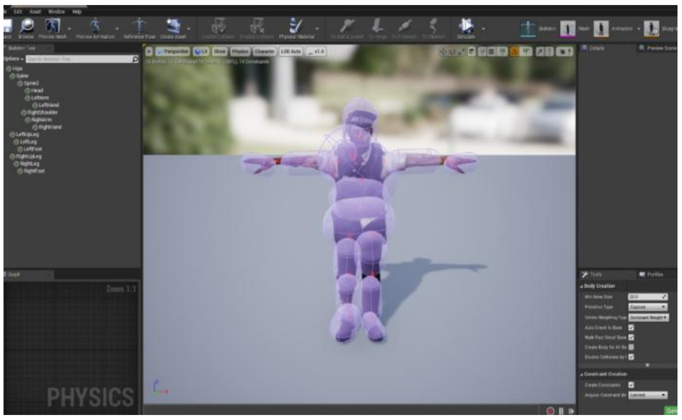

Fig 6. Working on ragdoll physics of a zombie character in game

Now the logic behind impact and speed attain of ragdoll is been learned from **VERLET INTEGRATION** which tells that the main idea behind the Verlet integration approach to ragdoll is that the positions of the joints are determined by point particles. The "bones" of the skeletons structure of ragdoll, represented by length constraints that enforce two neighboring particles, are an exact distance apart. Extra length constraints can be added as necessary to prevent unnatural poses, as well as self-intersection of the ragdoll and calculating a length constraint is the equivalent of resolving a sphere collision. We are putting small sphere in hip, joints and other part of body of ragdoll to effect of collision on our ragdoll. It generally works on following principles and according to it adjust the momentum gain by ragdoll, force added to the ragdoll because of collision. Those principle are-

x'=x+v.Δt x^'=2x-x^*+a.Δt

v'=v+a.Δt x^*=x

so, after some calculations the game engine will able to give character a suitable physics.

E) **ARTIFICIAL INTELLIGENCE IN GAME**

In today's modern era where video games will become most popular thing in entertainment section, they were continuing to evolving and there is no stopping it. In video games, computer science (AI) is employed to get responsive, adaptive or intelligent behavior primarily in non-player characters (NPCs) almost like human-like intelligence that gives ability to react and learn according to the update and progress made in the game. So, our FPS game also cover up a section of artificial intelligence. To apply AI on an enemy/NPC of our game we need to first create blueprint class of zombie, named that file as SimpleAI then, import skeletal mesh of zombie in game to work on AI of zombie character (which is our enemy in game) of our game. Now we need to work on animation of basic animation of zombie like walking, running and standing idle after that part we have to make sure to apply NAV MESH BOUNDS in our game which tells our NPC AI to where that character can move up to. Now we need to make enemy see our protagonist because with that pawn sense it will chase after your player until it drops your protagonist health to zero. It will create a field of vision for an enemy and with the help of detection mechanism in it when a player comes under his range NPC will recognize you and target you. We need to adjust pawn setting like site radius and peripheral vision angle to make pawn sense work and then fit it into your blueprint of your zombie. Fig 7 Will show an example of it.

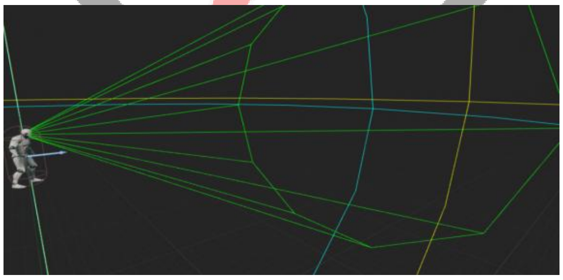

Fig 7 shows you pawn senses implemented in our game

Now we need to set AI in zombie's blueprint means under its event graph. enemy action under artificial intelligence where we need to make state transition in event graph for a zombie AI, we will divide it into 3 parts-

i. **To make ZOMBIE ROAM when he can't see your protagonist.**
ii. **To make ZOMBIE CHASE AND HIT you after he sees you.**
iii. **Zombie's hit will give 20% DAMAGE to a health of your player.**

While covering those section your event graph will have to make sure that that **AI move to** property is been intact because it plays a major role in given intelligence to your enemy in game, it holds many major options such as pawn, destination (which take input of zombie location) which it generally took from GetActor Location and take radius from GetRandomReachablePointInRadius where it tells that up to what distance enemy can move and all information is been pointed out to a Destination. **AI move to** take actions further according to a success or failure of its task.

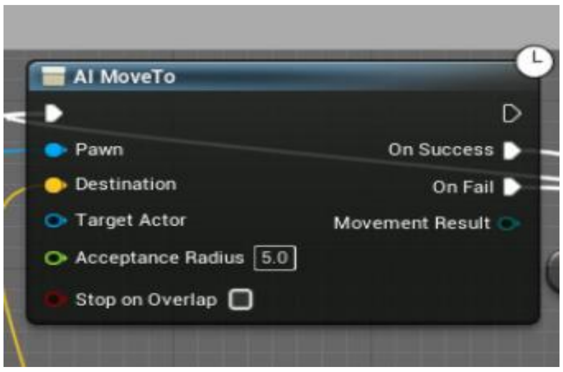

Fig 8. Shows "AI moves to" and properties in it

All those properties will help AI work properly in this game. So, all that part left is to set sphere collider to set at the hand of zombie structure through its SimpleAI file where you introduced your skeletal mesh (zombie) earlier. The sphere collider is introduced because of the iii. Part which we saw earlier, where we need to make zombie attack player and with every hit it should take 20% of the health from our protagonist for that we need sphere collider and to attach that collider to the left hand of your zombie structure we use Socket because of which the sphere collider work perfectly and within sync with the left hand of our zombie or we can say work in sync with animation of our zombie. Generally, sphere collider is being created with a function to damage a target. So, when zombie's hand moves to hit a target, collider on his hand touches the target which leads to damage in health.

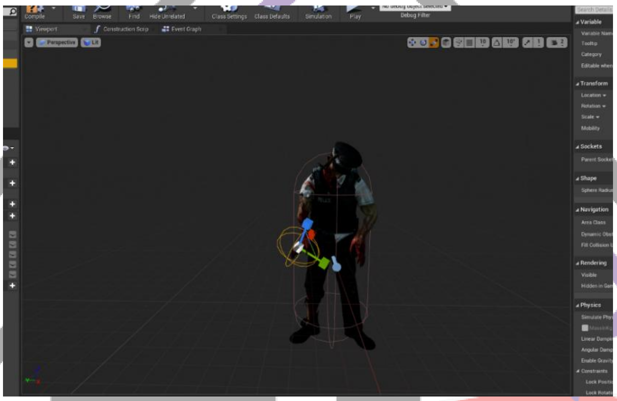

Fig 9. Shows sphere collider on right hand of our zombie.

After putting all those things together, when we run our game one can see perfect example of AI in our zombie's action while playing a game.

3. CONCLUSION

Here in this paper we are mentioning different options like artificial intelligence and ragdoll physics to give our game upper-hand in terms of advancement compared to old video games where those options are hard to find or even if they were used, the old game engine does not give freedom which comes along with those options. Pawn sense, certain death animation, collider adjustment they were not been found in old games. Sound plays a great role in shooting games; user can understand certain action with the help of sound used in our game which was not been found in old video games. 3D Animation effect is the one which generally get gamer attention, in this paper we are mentioning the different animation made in our game to the characters which will choose proper actions according to a certain event takes place in game. In today's world, the game is showing great advancement in terms of graphics, physics, etc. and game play perspective is also changing according to the player because of which we created FPS version of game where it gives gamer a proper experience of a virtual world of video game.

Now there is still so many upgradations can be done in FPS game because there is still some restraint that is suppressing potential of our game like in the upcoming time there is a concept called FULL DIVE IMMERSION which will truly defines FPS in their actual means. Physics in game animation is still not that smooth enough to make player not just to give him proper game experience but also to make them understand and feel that game concept. The use of AI in real-time task and first-person shooter games concerns the proposal of concepts that can be used to improve and further develop the use of artificial intelligence in computer games. there isn't a very high pace in improving the artificial intelligence in games is perhaps because the enemy or NPC characters in video game is not challenging enough compared to online multiplayer games. Either way, game AI is an area that is open for new developments.

REFERENCES

[1] Buro, M. (2003). Real-Time Strategy Gaines: a replacement AI Research Challenge. IJCAI'03 Proceedings of the 18th international joint conference on computing (pp. 1534-1535).
[2] Morgan Kauffman. Booth, M. (2009). The AI Systems of Left 4 Dead. computing and Interactive Digital Entertainment. Valve.
[3] Valve Software. 2004. Half-Life 2 [Computer program]. Electronic Arts.

[4] Wasik, B. 2006. “Grand Theft Education: Literacy within the Age of Video Games.” Harper's Magazine September, 31—39.

[5] Mark Oude Veldhuis, 2011.Artificial Intelligence techniques used in First-Person Shooter and Real-Time Strategy games [Human Media Interaction seminar 2010/2011: Designing Entertainment Interaction].

[6] Weber, B. G., Mawhorter, P., Mateas, M., & Jhala, A. (2010). Reactive planning idioms for multiple scale game AI. IEEE Conference on Computational Intelligence and Games, (pp. 115122).

[7] Pa.Megha,L.Nachammai,T.M.Senthil Ganesan. working on 3D GAME DEVELOPMENT USING UNITY GAME ENGINE. International Journal of Scientific & Engineering Research Volume 9, Issue 3, March-2018.