3d Puzzle Game Using Godot Engine

1Sathiya Narayanan.S, 2Poovarasan.A, 3Prasanna.M, 4P.Abinaya
1student, 2student, 3student, 4assistant professor
Agni College of Technology

***Abstract* - To develop a simple 3D Video Game Application. Which has kinematics and statics present. A level with meshes, nodes that interact with one another with GDScript (Modified Python). The game operates with one player input that is recorded and processed real time through the script. For which the software in use is Godot Engine. A 3D/2D open source game development platform. We are creating a 3D puzzle video game with physics. We use Godot Engine because it is open source and light weight compared to other Video Game Engines. Since Godot is lightweight the processing power required is low. Hence our game can run in most of all machine. Combined with the use of GDScript. We use GDScript for two reasons it is a specially modified python script designed to script gaming and Godots lightweight platform makes GDScript scalable. As we can change the script on the go and expect immediate results. Hence the game would be optimized consistently. This removes the unnecessary need for later stage big patches and chunk updates to weigh in on the software.**

***keywords* - Game Development, Game Physics Engine, GDScript**

I. INTRODUCTION

The game operates with one player input that is recorded and processed real time game. Once the player reaches the destination within the given amount time the player will win the game otherwise he/she loses the game. This games works over the concept of nodes. And the base communication is set over these nodes. These nodes communicate through signals. There are many signals and timers. The base work of every node is to have a function with a call. And once the function is called the execution begins. These functions are directly connected to the nodes. And nodes are connected to one-another creating a spiral of network. Making the game functional.

The extra z coordinates make many of common technique that helped to make 2d game simple no longer. To aid in the transition, it is worth mentioning that the Godot uses similar APIs for 2D and 3D.

II. RELATED WORKS

Generally, with the quality of a game engine, the very aspect of game is directly affected. At present, worldwide, there are many game engines that are widely being used to develop games. Such as Unreal Engine 4, Cry Engine, Unity, Cocos2dX, KlayGE, WildMagic, OGRE, BigWorld, GameBryo etc.

In these game engines, Cocos2dX is one of the other game engine similar to Godot that supports multi-platform 2D mobile phone games engines, is opened using C++, based on OPENGLES. And based on Cocos2d-iphone, support iOS, the platform such as Android, Winodws. Unity Technologies allows player to create 3D video game, building visualization, real-time three-dimensional. The multi-platform comprehensive development of games instrument of the type interaction content such as animation, it is the game engine of a high integration.

It is an object of the invention to provide a kind of game engine system for being used to develop 3D game, the automotive engine system includes Basic module, engine modules, Rendering Plug module and engine expansion module and engine resource editing machine, wherein, the basic mould Block is by encapsulating the various common mathematical functions related to different platform, for realizing the game engine system to different development platforms Higher level of abstraction, and for engine upper layer module and engine user of service to be isolated with development platform, the basic module Including platform library, tool storage room, memory submodule, math library, serialization submodule and thread submodule; Engine modules are used to carry for the core component of interactive realtime graphic application program, rendering module, AI modules, physical module, UI modules, audio mould are included Block, Special Effect System module, resource management module, animation, script module, log pattern, card module, to realize sports ground Scape and role render and simulate and the management and extension of resource.

Disadvantages

A. Process resource consumption is high
B. Not portable
C. Too slow to operate or edit in a medium end machine
D. Does not have a native script

III. METHODOLOGY

Godot aims to offer a fully integrated game development environment. It allows developers to make a game from scratch, needing no other tools beyond those used for content creation (visual assets, music, etc.). The engine's architecture is made round the concept of a tree of "nodes". Nodes are organized inside "scenes", which are reusable, instanceable, inheritable, and nestable groups of nodes. All game resources, including scripts and graphical assets, are saved as part of the computer's file system (rather than in a database). This storage solution is intended to facilitate collaboration between game development teams using software version controlsystems.

The engine supports deployment to multiple platforms and allows specification of texture compression and resolution settings for each platform. Currently, supported platforms include Linux, macOS, Microsoft Windows, BSD, Android, iOS, Universal Windows Platform, HTML5, and WebAssembly.

Games using Godot are often created with a spread of programming languages including C++, C#, and the other language with GDNative bindings like Rust, Nim, and D.

Godot also has its own built-in scripting language, GDScript, a high-level, dynamically typed programming language very similar to Python. Unlike Python, GDScript is optimized for Godot's scene-based architecture and may specify strict typing of variables. Godot's developers have stated that a lot of alternative third-party scripting languages like Lua, Python, and Squirrel were tested before deciding that employing a custom language allowed for superior optimization and editor integration. The engine also supports visual coding via its own built-in visual programming language VisualScript, designed to be a visual equivalent to GDScript

Godot includes a script editor with auto indentation, syntax highlighting and code completion. It also features a debugger with the power to line breakpoints and program stepping.

Godot's graphics engine uses OpenGL ES 3.0 for all supported platforms; otherwise, OpenGL ES 2.0 is used. Future support for Vulkan is being developed. The engine supports normal mapping, specularity, dynamic shadows using shadow maps, baked and dynamic Global Illumination, and full-screen post-processing effects like bloom, DOF, HDR, and gamma correction. A simplified shader language, almost like GLSL, is additionally incorporated. Shaders can be used for materials and post-processing. Alternatively, they will be created by manipulating nodes during a visual editor.

We also use blender for creating objects. Blender is a free and open-source 3D computer graphics toolset. It is used for creating animated films, visual effects, art, 3D printed models, motion graphics, interactive 3D applications, virtual reality and computer games.

Blender is a modeling and animation software studio. It comes with a quite advanced and efficient game engine. Through the embedded Python interpreter, the game engine is wide open for implementing real-time interaction of external control software with the internal game engine logic. The game physics may be utilized whenever some mechanical part is under uncontrolled motion, like sliding or falling

It is the full support for the 3D pipeline such as simulation, modeling, animation and rigging. The main advantage over the blender is works equally well on Linux, Windows and Mac Operating system.

IV. WORKING

The base level is created using a Spatial node. Here we use a 3d spatial node. Then we add a surface upon which the game can take place. And that surface would have a mesh and collision shape.

For the main player character, we introduce a kinematic body and that is also set a mesh and collision shape. The collision shape from the area, that acts as the ground, and the collision shape of the character interact with each other. And due to which the character is placed on the ground and does not fall through the ground.

Now we have to script a code connecting the controls to the player. Parameters such as velocity, speed are set. With which the ball can now accelerate in the desired direction.

For the goal; we have set collecting a set number of coins within a specific time limit as the goal. And for which we have to add an area and a signal indicating that the ball has touched the coin. And adding a label to keep count of the coin. The timer is set once the level has come into play. Due to the easy-ness of the game we have decided not to reveal the timer as a label giving it some difficulty. When the player collects all the coin he either progresses to the next level which is called through a script or the game ends when he beats the final level. If he fails to beat the game within the time. The game over screen is presented. Where they can go back to the Main menu.

V. COMPARISON

Other engines which are used to design and develop games are much time consuming and very heavy on the system. Which can be a problem for low end systems to run it efficiently. The process times are very vivid that they are fast and require very little power.

Godot is advantageous compared to other engines such as Unity because you don’t need to install Godot Engine. You just have to download and run. And unlike other game engine Godot Engine is small, it comes sits at a mere 800mb of data space. And can be carried even in a flash drive. And the time taken to change anything in the project (Such as patches and bug fixes) can be done in real time. And the results can be checked in the output as is. This makes it time efficient as well. And Godot is a free engine. This makes it Time, Cost, and Space efficient.

VI. RESULTS AND DISCUSSION

Programming in a script that is much similar to python, but specifically created for video games have created a direct and straight forward approach to achieve the goal. The game starts at the screen welcoming the player and summarizing the rules of the game. The player then is taken to the game screen. This works as intended through the script. Then the game starts and player controls the character. When the player collects all the coin the game over-win screen is presented. Or when the time runs up the game over-loss screen is presented. All of this works within the frame work and with accordance to the script as intended.

VI. SUMMARY AND CONCLUSION

This study aims to design and develop a gaming application using Godot Engine. A real time 3D gaming is built with the high-level user experience and a good quality animation for the purpose of excitement during the gameplay is done. Multiple levels are placed so that the difficulty over the different level will change accordingly respectively. Through this cornerstone for a new concentration on Game Development, we have conveyed our views and ideas on promoting education via innovation. The functions that Godot supports autonomously are very abundant. All game developments are possible such as shader, physics engine, network, terrain manipulation, audio, video, and animation, and it is enabled so that it is possible to revise, meeting demand of user according to the need.

REFERENCES

[1] E. Schmück, R. Flemming, P. Schrater and P. Cardoso-Leite, "Principles underlying the design of a cognitive training game as a research framework," 2019 11th International Conference on Virtual Worlds and Games for Serious Applications (VS-Games), Vienna, Austria, 2019, pp. 1-2, doi: 10.1109/VS-Games.2019.8864551.

[2] S. Bachelder, R. Santhanam, M. Hayashi and M. Nakajima, "Engagement in Computer and Video Games," 2013 International Conference on Cyberworlds, Yokohama, Japan, 2013, pp. 371-371, doi: 10.1109/CW.2013.77.

[3] R. Volkovas, M. Fairbank, J. R. Woodward and S. Lucas, "Practical Game Design Tool: State Explorer," 2020 IEEE Conference on Games (CoG), Osaka, Japan, 2020, pp. 439-446, doi: 10.1109/CoG47356.2020.9231863.

[4] I. Paliokas, C. Arapidis and M. Mpimpitsos, "PlayLOGO 3D: A 3D Interactive Video Game for Early Programming Education: Let LOGO Be a Game," 2011 Third International Conference on Games and Virtual Worlds for Serious Applications, Athens, Greece, 2011, pp. 24-31, doi: 10.1109/VS-GAMES.2011.10.

[5] D. Perez-Liebana, J. Liu, A. Khalifa, R. D. Gaina, J. Togelius and S. M. Lucas, "General Video Game AI: A Multitrack Framework for Evaluating Agents, Games, and Content Generation Algorithms," in IEEE Transactions on Games, vol. 11, no. 3, pp. 195-214, Sept. 2019, doi: 10.1109/TG.2019.2901021.

[6] A. Khalifa, M. C. Green, D. Perez-Liebana and J. Togelius, "General video game rule generation," 2017 IEEE Conference on Computational Intelligence and Games (CIG), New York, NY, 2017, pp. 170-177, doi: 10.1109/CIG.2017.8080431.

[7] Michael Nitsche, "Examples of Spatial Structures in Game Spaces," in Video Game Spaces: Image, Play, and Structure in 3D Worlds , MIT Press, 2008, pp.171-189.

[8] C. Bae, E. K. Kim, J. Lee, K. Kim and J. Na, "Generation of an arbitrary shaped large maze by assembling mazes," 2015 IEEE Conference on Computational Intelligence and Games (CIG), Tainan, Taiwan, 2015, pp. 538-539, doi: 10.1109/CIG.2015.7317901.

[9] R. P. Prager, L. Troost, S. Brüggenjürgen, D. Melhart, G. Yannakakis and M. Preuss, "An Experiment on Game Facet Combination\," 2019 IEEE Conference on Games (CoG), London, UK, 2019, pp. 1-8, doi: 10.1109/CIG.2019.8848073.

[10] P. H. Kim and R. Crawfis, "The quest for the perfect perfect-maze," 2015 Computer Games: AI, Animation, Mobile, Multimedia, Educational and Serious Games (CGAMES), Louisville, KY, USA, 2015, pp. 65-72, doi: 10.1109/CGames.2015.7272964.