Concept of Distributed Computing Using Multiplayer Game

PROJECT REPORT

Submitted for the course: **Parallel and Distributed Computing (CSE 4001)**

SLOT – D1

By

Team Members:

RAVI VOLETI - 15BCE0082
RONIT CHAUDHURI - 15BCE0054 (slot:D2)
ANSHUL HEDAU - 15BCE2079
SOURAV KUMAR DASH – 15BCE0320

Submitted to:

Prof. **Manoov R**

B.Tech
. in
Computer Science and Engineering
School of Computing Science & Engineering (SCOPE)

Acknowledgement

The outcome of this project required a lot of guidance and assistance from many people and we are extremely fortunate to have got this all along the completion of this project work.

We express our sincere gratitude to Prof. **Manoov R** (School of Computing Science & Engineering) Associate Professor, VIT University, Vellore, for giving us an opportunity to do the project work on **Concept of Distributed Computing Using Multiplayer Game**. We are extremely grateful to him for his constant support and guidance.

We are thankful to and fortunate enough to get constant encouragement, support and guidance from all our friends and teachers, which helped us in successfully completing our project work.

RAVI VOLETI
[15BCE0082]

RONIT CHAUDHURI
[15BCE0054]

ANSHUL HEDAU
[15BCE2079]

SOURAV KUMAR DASH
[15BCE0320]

Aim:

The aim of the project is to enhance the concept of distributed computing using multiplayer game. We run the multiplayer game in different screens parallel and use client server connection using router to connect the different players.

Objective:

The main objectives of the project are as follows:

- Designing a simple game using the software unity, having a suitable map, and dynamically moving players and enemies.
- The simple features of the game include players being able to move in all possible directions, players being able to hit the enemies, and suitable decrease in lifeline on being hit.
- The game can parallely run in many screens on a single machine and also players can connect to each other using distributed system features and player on a single screen.
- Basically the game uses the shared memory where every instance of the game is stored and there is an internal memory for each system in the distributed environment.
- The game has a dual feature such that the player can both play against the enemy as well as the system.
- The player can move in various directions and the game doesn't limit the number of distributed system.

Software Requirements

The following software are required for implementing the project successfully:

- Unity Software – The latest version for game developers.
- Windows
- Mac OS –X
- C++ compiler
- GitHub repository

Hardware Requirements

The following hardware is required for implementing the project:

- Router with appropriate bandwidth for connecting machines
- Desktop with suitable processor of at least 1Mhz
- More than one machine to show the connection of players successfully.

Introduction

Description about distributed architecture in multiplayer game

MULTIP LAYER GAMING NETWORKING: - In the beginning plot were electronic networked equal to peer, with each calculator which central info with each other in a fully connected mesh regional anatomy. In today 's date still it's used in RTS plot and interestingly for some intellect, but perhaps because it's the first way in which hoi polloi think games are networked. But the Holocene epoch era is of action mechanism games so peer to peer networking arrangement has been obsolete so we need different node and holder's. There is a problem in compositor's case of doom that is was only designed for networking over LAN only. Here comes the aspect of lagging where we need to moves or shoot we need to wait for inputs. So, in 1996 Saint John and his squad did when he released quake using client/server instead of peer to peer.

DISTRIBUTED COMPUTING: - A distributed computer organisation consists of multiple software organisation ingredient that are on multiple calculator, but campaign as a one system. The computers that are in a distributed system can be physically finis together and connected by a local network, or they can be geographically distant and connected by a wide surface area network. A distributed system can consist of any number of possible configurations, such as C.P.U., personal computers, workstations, minicomputers, and so on. The goal of distributed calculation is to brand such a network work as a single computer. Distributed computing system can campaign on computer hardware that is provided by many vendors, and can use a variety of standards-based software components. Such system is independent of the underlying software. They can run on various operating scheme, and can use various communications protocols. Some hardware might use UNIX or Linux as the operating system, while other hardware might use Windows operating organisation. For intermachine

communications, this hardware can use SNA or Transmission control protocol /IP on Ethernet or Token Ring.

Arranged diversions are quickly developing from little 4-8 man, one-time play recreations to substantial scale amusements including a great many members and tenacious diversion universes. Be that as it may, as most Internet applications, current organized amusements are unified. Players send control messages to a focal server and the server sends (significant) state updates to all other dynamic players. This plan experiences the notable strength and adaptability issues of single server outlines. For instance, complex amusement play and AI calculation keep even very much provisioned servers from supporting more than a few several players for first individual shooter (FPS) diversions. Further, customer server diversion outlines regularly constrain players to depend on foundation gave by the amusement makers. These frameworks are here and there not all around provisioned or extensive; subsequently, they either give poor execution or keep clients from playing their diversion long after their buy. An appropriated configuration can conceivably address the above weaknesses.

Be that as it may, architecting disseminated applications is troublesome. The most essential test in conveying an application is in apportioning the application's state (e.g., the amusement world state) and execution (e.g., the rationale to reproduce player and diversion AI activities) among the taking an interest hub. Appropriating an organized diversion is made considerably more troublesome by the execution requests of the constant amusement play. What's more, since the amusement play of an individual player makes an interpretation of to updates to the mutual condition of the diversion application, there is significantly more compose activity and compose sharing than most disseminated applications. Luckily, there are two principal properties of such diversions that we can exploit in tending to these difficulties. In the first place, diversions endure feeble consistency in the application state. For instance, even current customer server executions limit intuitive reaction time by introducing a pitifully predictable perspective of the amusement world to players. Second, amusement play is typically represented by a strict arrangement of guidelines that make the peruses/composes of the mutual state profoundly unsurprising. For instance, most peruses and composes of a player happen upon objects which are physically near the player.

The test, at that point, is to touch base at an adaptable and proficient state and rationale apportioning that empowers sensibly steady amusement play without acquiring much dormancy. This paper introduces the outline, execution and assessment of Colyseus, a novel appropriated engineering for intelligent multiplayer recreations intended to accomplish the above objectives. Our plan depends on two key compositional choices: First, appropriated questions in Colyseus take after a solitary

duplicate consistency display – i.e., all keeps in touch with a protest are serialized through precisely one hub in the framework. This permits low-idleness peruses and composes at the cost of powerless consistency. This mirrors the consistency model of a customer server engineering, but on a for each protest premise. Besides, Colyseus uses region and consistency in the development examples of players to theoretically pre-get objects required for driving amusement rationale calculation. Proficiently finding items to pre-get is accomplished utilizing a versatile circulated go inquiry query benefit.

So this amusement fundamentally utilizes the appropriated engineering to actualize the multiplayer include on various screens. Here we split the screens and play the diversion on a similar guide, so this uses a mutual memory having appropriated design. Appropriate customer server correspondence between frameworks has been accomplished with the utilization of the switch.

Implementation –

Network Manager –

For the network to recognize and handle players and spawning we use a prefab called network manager which has a network manager script attached to it.

Figure 1 : Statistics when Game starts

Figure 2 : Statistics when 1 Player Joins the game

Figure 3 : Statistics when 2 Player Joins the game

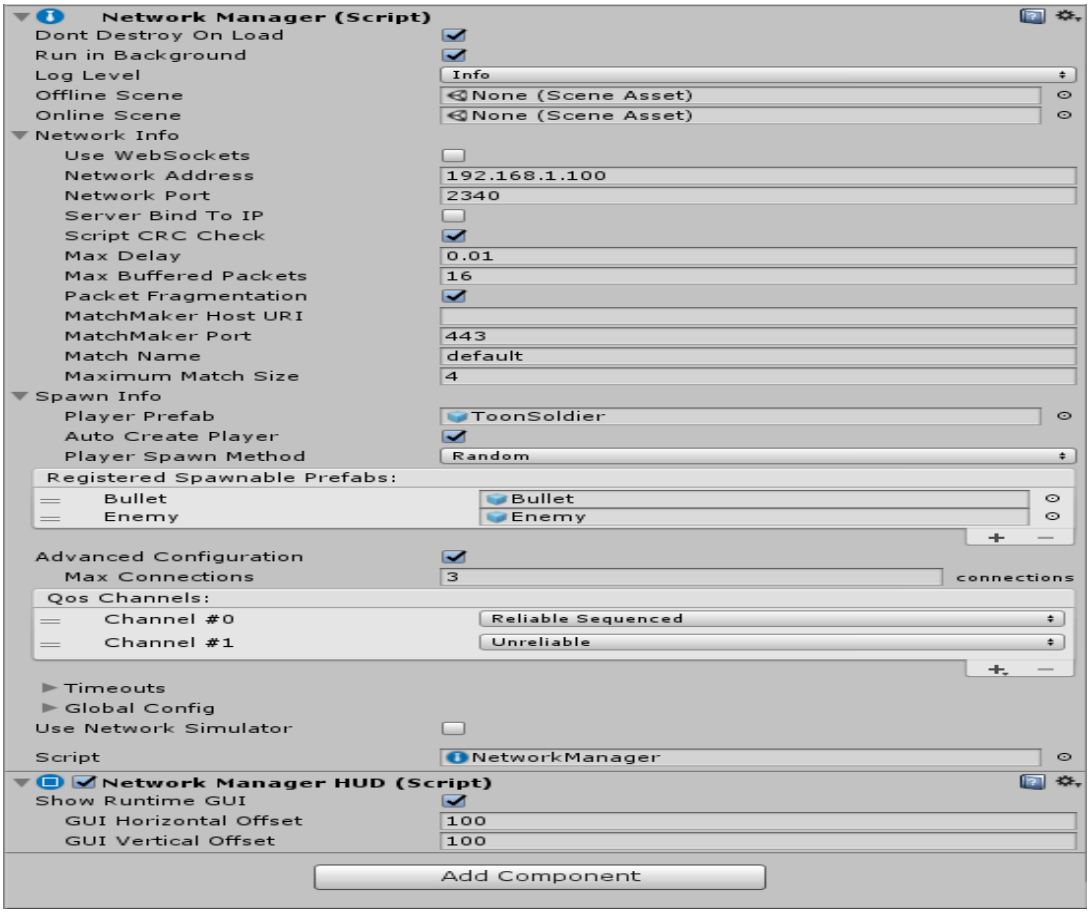

Figure 4 : Network manager (Script)

Outputs –

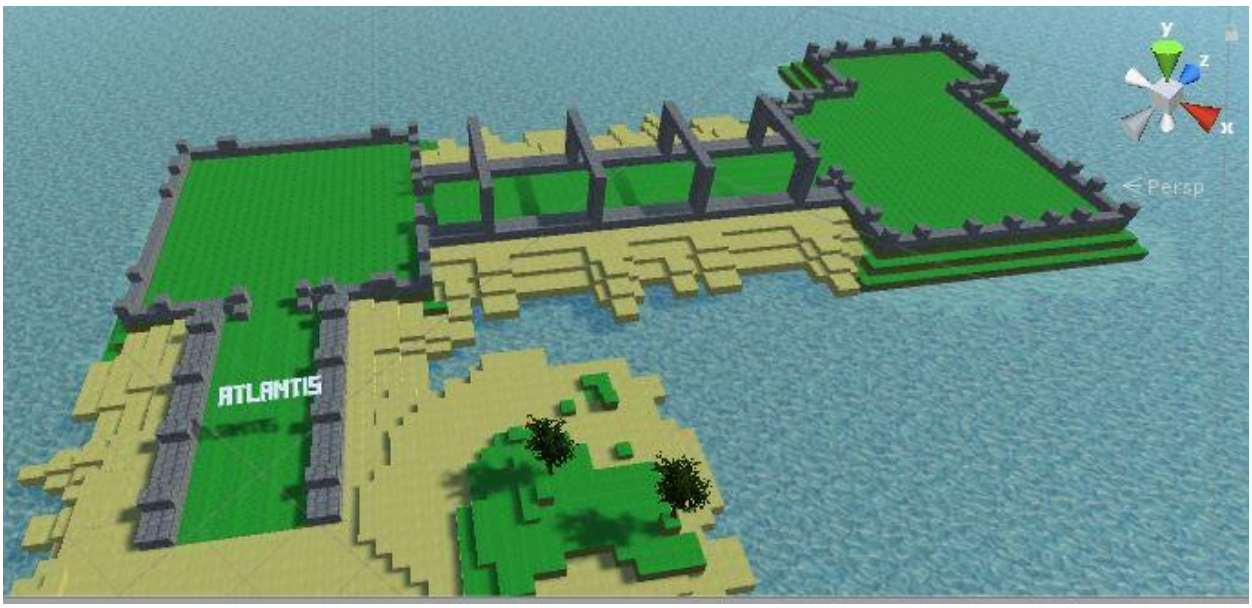

Figure 5 : Map of the Game

Figure 6 : Starting Scene of Game

Figure 7 : Fighting Sequence of Game

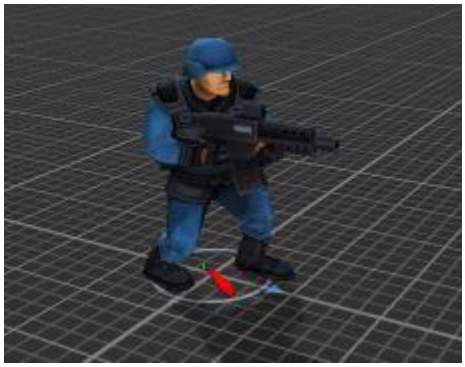

Figure 8 : Image of Hero

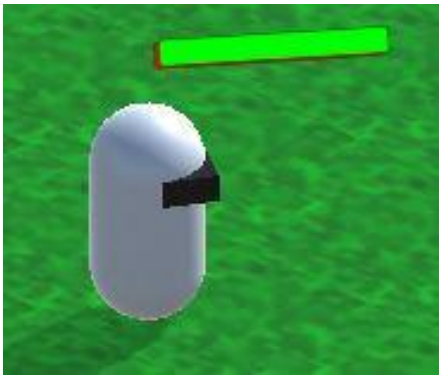

Figure 9: Image of Enemy

Link of the project

https://github.com/xelese/PDC-Project

Download link

https://drive.google.com/drive/folders/0B1ly-qt-vJNAY2F1MC1OTTlvZlE?usp=sharing

Conclusion

The project is almost complete and proper display of screenshots and video for demonstration of the game will be uploaded for the final review. We can conclude from the project that parallel computing can also be applied in the domain of gaming and entertainment, and like

all other fields here also it helps to improve the performance of the system on the basis of time complexity and space complexity. Here actually the concept of distributed computing has been used to connect different machines through local area network, so that players can play against each other.

Future Enhancements

For the future enhancement we are thinking of increasing the features of the game, like for now we can add features like shooting more enemies and making dynamic and moving enemies, and for now we can only connect 3 machines in a router, and we will increase the number from 3 to unlimited, so that players can play over a wide range, and now there is also one more problem like we can't play over a certain distance, and we will make the provision of playing over a wide area network.

References

We have referred to the following websites for our project which are follows:

1. http://www.sbgames.org/sbgames2011/proceedings/sbgames/papers/comp/full/10-92115_2.pdf
2. https://web.cs.wpi.edu/~cs4513/d14/papers/net-aspects-02.pdf
3. http://ieeexplore.ieee.org/document/6363221/
4. https://unity3d.com/learn/tutorials
5. https://www.lynda.com/Unity-training-tutorials/1242-0.html
6. https://www.techopedia.com/definition/7/distributed-computing-system
7. http://wla.berkeley.edu/~cs61a/fa11/lectures/communication.html