Real time online multiplayer gaming framework using Kinect Sensors

By,

Imtiaz Ahmed* - 14101023
Khondokar Sadman Shabab* - 13221028
MD. Shahadat Hossain* - 14101008
MD.Tashauf Akand* - 14101012
Shovon Majumder* - 14101019

Supervisor
Dr. Md. Ashraful Alam

Department of Computer Science and Engineering
BRAC University

Thesis report submitted to BRAC University in accordance with the requirements for the degree of Bachelor of Science in Computer Science and Engineering

Submitted in March 2018

BRAC UNIVERSITY
Inspiring Excellence

Department of CSE

Computer Science & Engineering

Thesis Type: □ **Pre-Thesis 1** □ **Pre-Thesis 2** □ **Final Thesis**

Semester: Spring / Summer / Fall **Year: 20** 18

Group Members Name:

SL No.	Name	ID
1	Imtiaz Ahmed*	14101023
2	MD. Shahadat Hossain*	14101008
3	MD.Tashauf Akand*	14101012
4	Shovon Majumder*	14101019
5	Khondokar Sadman Shabab*	13221028

Supervisor's Name: Dr. Md. Ashraful Alam

Co-Supervisor's Name (Optional):

Thesis Title:

Real Time Online Multiplayer Gaming Framework Using Kinect Sensors

Co- Supervisor's sign:

Supervisor's Sign:

66 Mohakhali Dhaka 1212 Bangladesh • Tel: +880 2 8824051 • Fax: +880 2 881 0383 •www.bracu.ac.bd

Declaration

We, hereby declare that this thesis is based on the results found by us. Materials of work found by other researchers are mentioned by reference. This thesis, neither in whole or in part, has been previously submitted for any degree.

Signature of the supervisor

Dr. Md. Ashraful Alam

Signature of Author

Imtiaz Ahmed

Signature of Author

Khondokar Sadman Shabab

Signature of author

MD.Shahadat Hossain

Signature of Author

MD.Tashauf Akand

Signature of Author

Shovon Majumder

Acknowledgement

Foremost, we would like to express my sincere gratitude to our advisor Dr. MD. Ashraful Alam for the continuous support of our Thesis, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped us in all the time of research and writing of this thesis. We could not have imagined having a better advisor and mentor for our Thesis study.

Last but not the least, we would like to thank our families and our parents for supporting us spiritually throughout our journey.

Finally, we thank BRAC University for giving us the opportunity to complete our thesis in the university.

Abstract

A real-time online multiplayer game framework is proposed and demonstrated. The proposed model is composed of multiple Microsoft Kinect sensors and Unity 3D game development tool. The motive is to make a real time multiplayer game that uses user's movement data as it's input through Kinect and build a gameplay around this concept. This type of project has not been attempted especially on windows platform and most Xbox multiplayer game only uses leaderboard as a way to connect multiple players. The game will be played solely by the interaction of players in real time even though they are apart. At first a skeleton of 25 bone joints has been made and make character models around it. And then detect user through Kinect and sync user's body with our model. This means now our character will move when user moves using the tracking data provided by Kinect sensor. A game environment for our game has been made and place user character in a specific spawn place. The process allows player characters to interact with each other within the virtual world. This whole process needs a server to perform and the server will have to manipulate data efficiently to allow player's a smooth gameplay without any lags.

INDEX

List of Figures

List of Tables

CHAPTER 1

1.Introduction

There have always been different kinds of games played by humans for their entertainment and recreational purposes. With the passage of time the form of game has changed in many ways. Humans used to play games and sports outside home before modern time. But, with increasing amount of people and decreasing amount of both time and space humans started playing games at home. Computer games has created a revolution in gaming world. People used to play normal video games. Then multiplayer co-op game came which is something like playing with friend in the same machine. For example, some of the earliest video games were two-player games, including early sports games (such as tennis_for two at 1958's and Pong at 1971), early shooter games such as *Spacewar!* (1962) [10]. Then the networking has added in the gaming world and made a huge revolution. People can now play games with friends and other people at the same time from different places. Ken Wasserman and Tim Stryker in an article named "BYTE" at 1980 identified three factors which make networked computer games appealing [11]. They are -

"1. multiple humans competing with each other instead of a computer.

2. incomplete information resulting in suspense and risk-taking.

3. real-time play requiring quick reaction" [11].

But now scientists are thinking about the physical health of the game addicted kids and teenagers. For this issue and to make another revolution in gaming world, Microsoft Kinect, Nintendo Wii, PlayStation Move was introduced.

1.1 Motivation

At the advent of twenty-first century, we have made numerous progression in digital entertainment in the form of video games, emerging E-sports titles signify an age where people can interact with people across long distance and play simultaneously. Ours is an attempt to further this form of entertainment by using a mix of virtual and augmented reality. We are using Microsoft Kinect to detect player body and use the movement of player in real time with a server. The game will use multiple player's data with player movement being the core part of gameplay. Kinect is a gesture detection device designed by Microsoft which is used for human gesture detection for gaming for Xbox 360. Although Kinect is actually designed for Xbox gaming, we are going to build a game for windows platform. Unity3D will be used for designing the game framework. The game will be a real time multiplayer so that multiple player can log on the same server in same time and play with each other in real time from different places. Our main challenges will include streamlining data given by Kinect, reducing ping, latency, input lag and efficient data handling on the server. Our goal is to normalize the complexity of VR and AR implementation of real-time online multiplayer game using Kinect and encourage this new form of entertainment.

1.2 Literature Review

The video game has undoubtedly greater impact for the sector of creation and presentation in any other entertainment or interactive media. Video games offer new ways to find primary source research, explore new paths of knowledge and questions and enhance the fields of digital humanities and virtual heritage [13].

From the observations of successful entertainment in gaming world, it has been hypothesized that implementing a multiplayer option may require a different approach from that of a
Single player option, in terms of game design. To find out the truth of this thought, an investigation of two educational games were designed and evaluated by the authors. a theoretical investigation from a game and learning perspective and an investigation of 23 case studies has been taken. From these investigations, it turned out that a “single-player approach” is data intensive, has formal rules, and uses direct transfer and individual learning. On the other hand, a “multiplayer approach” is less straightforward. From a game perspective, it can be characterized as process intensive and having social rules. When related to learning, however, it is better than single player as a person can learn from other person’s thinking and actions [12]. Outside the game’s virtual world, players can discuss in-game practices, rules and laws through online forums, fan sites, and wikis related to the game which proposes increases meaningful social interactions and collaborative work. Multiplayer gaming teaches well behaviors and also working together with other gamers to accomplish game-related tasks and to form teams is likely to lead to significant social ties within that gaming community [6]. And when the multiplayer game is built with Kinect sensors, then it would be more helpful, entertaining and thrilling as a player’s body movement is included in these games. It will also help a person to perform in some physical work which partially make them

healthy. And if some workout movement type game can be built, then it will surely make a person body and mind fresh and make them biologically fit.

1.3 Thesis Orientation

In Chapter 1, This project is introduced with some in depth history of the components and motivations for this project. in chapter 2, fundamentals and history of Kinect sensors, Unity game engine and online multiplayer games are discussed. In chapter 3, the proposed method of this project is stated in details with sections covering different aspects of project. In chapter 4, the development environment of this project i.e. system setup are stated and the overall outcomes of this project are discussed. In chapter 5, this paper is concluded while stating the limitations along with future expectations and possible expansions to this project.

The objective of this project is to create a real time multiplayer gaming framework for developers who wish to use such technology for their game.

CHAPTER 2

2.Fundamentals of Kinect sensors, Unity game engine and online multiplayer game

Kinect is a sensor which is developed by Microsoft mainly for Gaming in Xbox platform which can detect human gesture and send data to a device. Online Multiplayer game is a gaming environment where more than one person can play in the same game environment at the same time from different part of the world.

2.1 Kinect Sensors

Microsoft launched Kinect on June 14th,2010 for Xbox 360 video game console which is a motion sensing input device and comparatively one of the most advanced human motion sensing device in the market at present [14]. Depth image including Player Index, RGB image, Tilt, Microphone Array and Skeleton Tracking are the features supported by the Kinect services [15]. Additionally, main purpose of developing Kinect was to replace speech recognition with gesture recognition in game controlling methods for Xbox 360 [16]. Infrared projector, infrared camera, RGB camera and multi-array microphone are the main parts of Kinect to recognize gestures [14]. It makes a skeleton of a human body shape which is divided into five parts in three data sets. They are MSR action 3D dataset, Berkeley MHAD and HDMO5 [1].

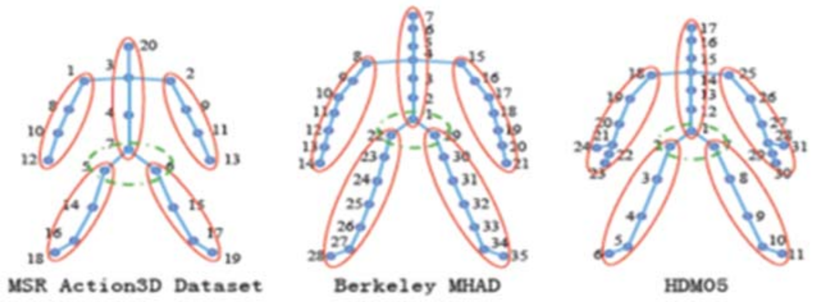


Fig.1. The human skeleton joints are divided into five parts in these three datasets.

2.2 Kinect SDK

Kinect SDK is provided by Microsoft and it provides application programming interfaces and device interfaces. The SDK decodes data from Kinect for developers to use in their work. In Kinect SDK version 2, we get some awesome and improved features that helps to detect body movement more efficient way and reduce noise from environment. This contains a wider horizontal and vertical field of view for depth and color than the previous versions. It also supports full HD color and microphone attached to the Kinect sensors are very improved which is nearly zero balanced. It has Lighting Independent Infrared in 30 fps, depth fidelity in 3X. It can detect 6 person's skeleton at the same time and for each player it supports 25 skeletal joints. It has thumb tracking, hand open or close gesture tracking and end of hand tracking. This feature has added a great impact in gaming. It is very helpful feature to give some control with gesture and make the game more interesting. For example, in shooting game we can shoot with two finger or thumb gesture, swap weapon with open hand gesture and reload with closed hand gesture. Again, in racing game we can make closed hand gesture as hand brake and two finger gesture as nitro boost and close hand gesture as brake. We also can use multiple gesture at the same time. Like two finger gesture in one hand and open finger gesture in another hand. This makes a game more interesting. Because for example in racing game we may use handbrake and nitro boost at the same time which is now possible for tracking the multiple gestures at the same time. Like this it makes a game more interesting and realistic.

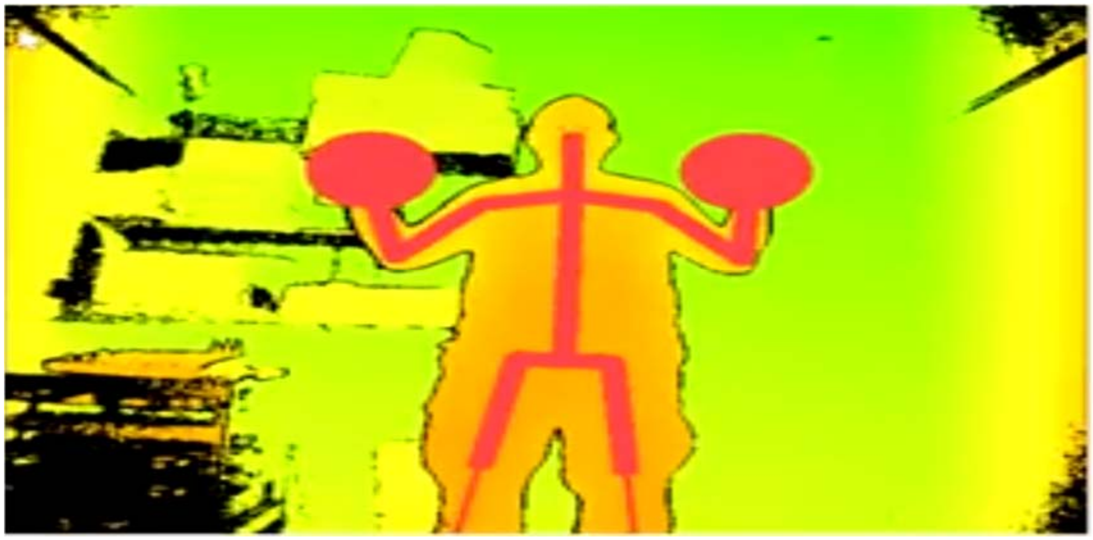


Fig.2. Kinect Close Hand Gesture.

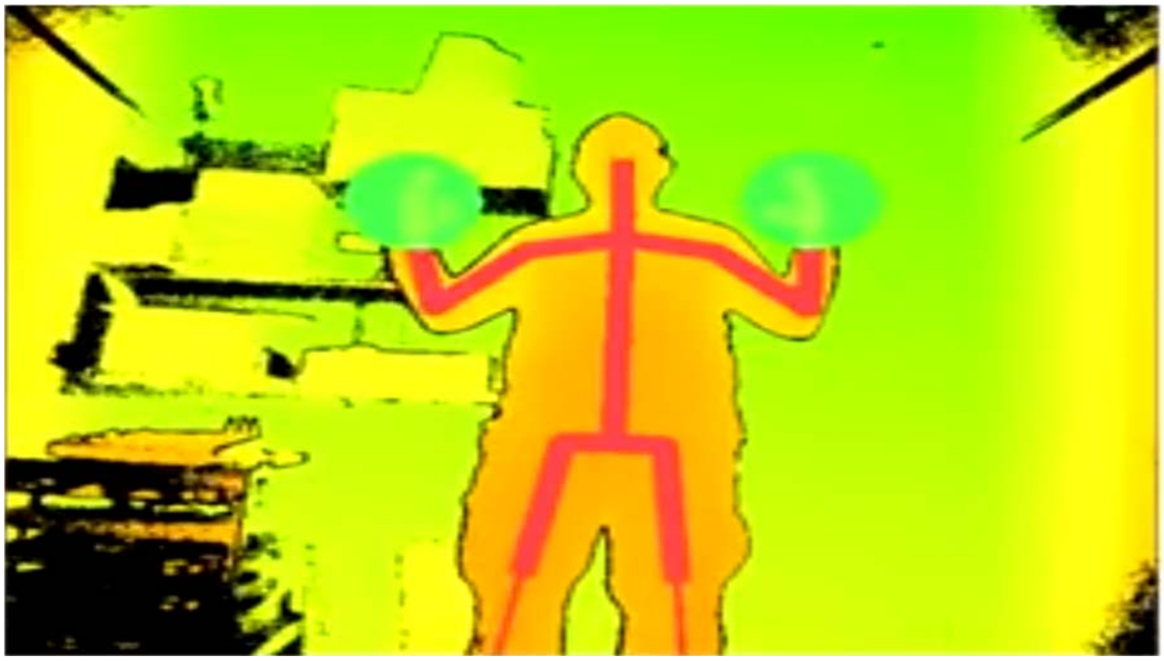


Fig.3. Kinect Open Hand Gesture.

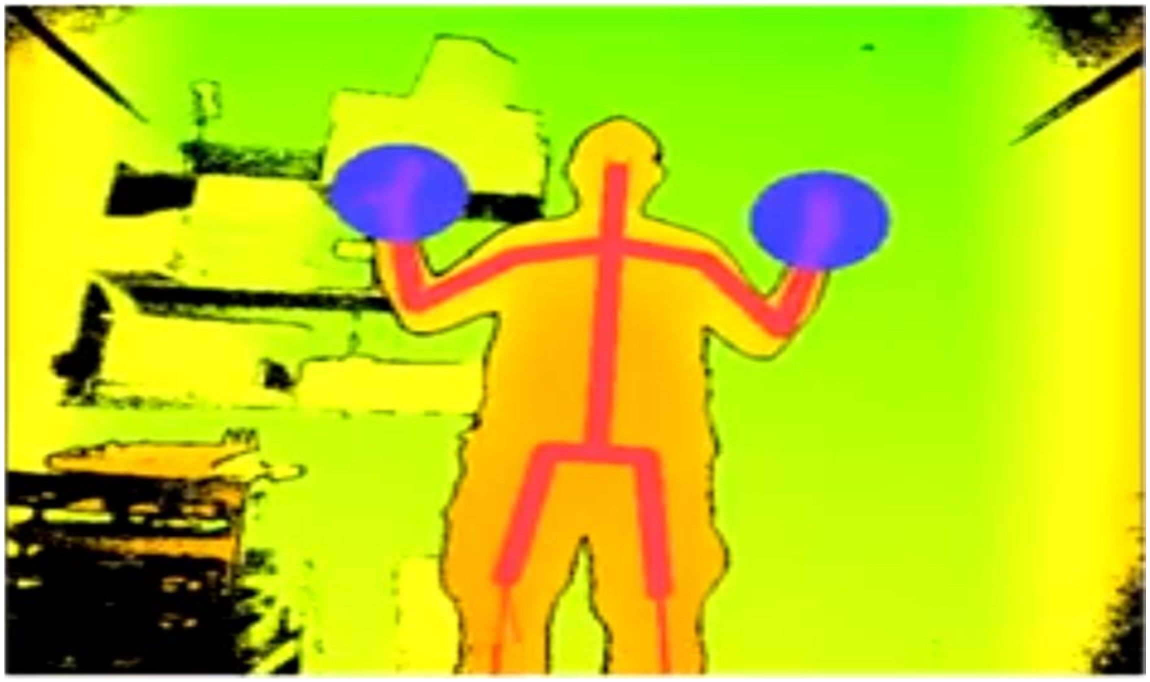


Fig.4. Kinect Two Finger or Thumb Detection Gesture.

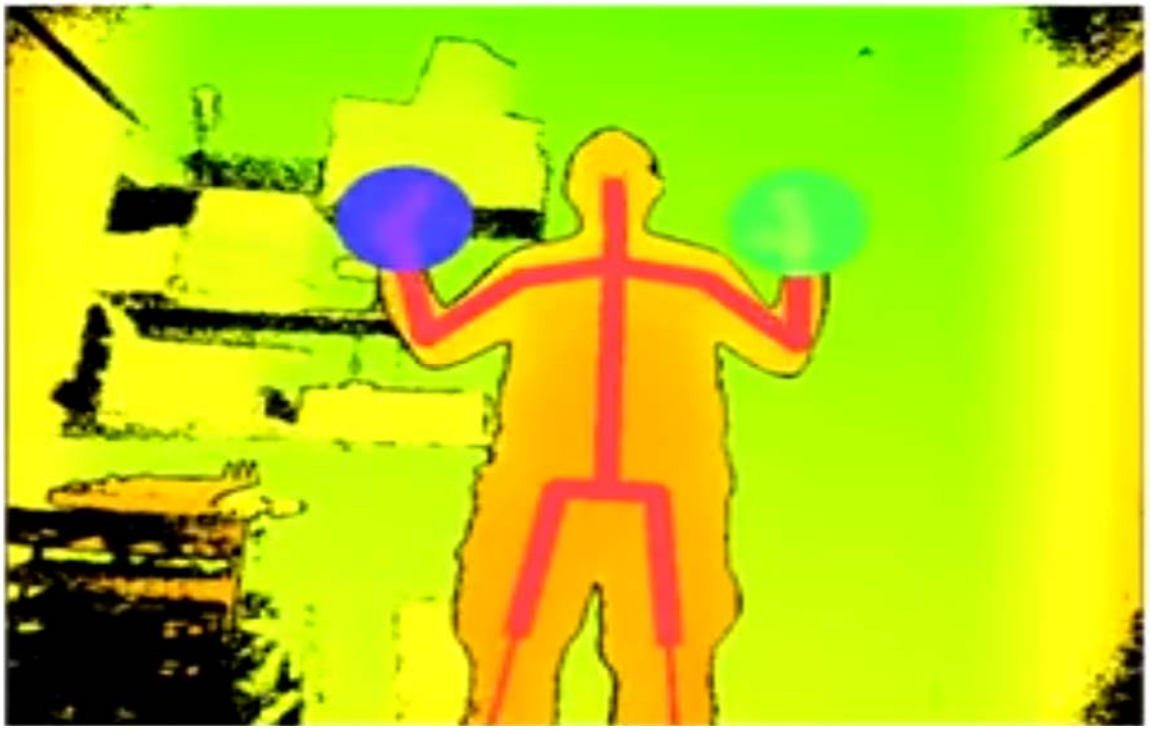


Fig.5. Kinect Multiple Gesture at the Same Time.

The range and the operation of the detection of a human gesture has improved which is 0.5 meter near and 4.5 meter far. And more importantly now multiple applications can utilize the sensors simultaneously [18].

Kinect SDK also support a gesture builder option through which we can record movement video and then cut clip from the video. These clips will help to specify the movement in a game when we will make it with any game engine. It also has an option through which developer can turn off the detection of any body part which will not be used in a game. Incase of a game like DX Ball, only lean left and right features are needed. So there is no need for hand or leg skeleton part. In such scenario, Developer can make them turn off to get more noise free appropriate data and faster data. As we are developing multiplayer game so the ability to turning off some unnecessary data is a huge help in handling server latency, ping, packet loss and other issues.

We have integrated this SDK to our UNITY3D platform through the library provided by the SDK. The skeleton detection will be operational in this method.

2.3 Unity Game Engine

Unity is a game engine which is one of the best choice for small studios, indie developers. It has both paid and free versions. The free version is quite handful for develop a game. It has a big community and for question answer and an effective asset store where many paid and free assets are available for using in a game. Through we can build 2D or #d game, single player or Multiplayer game and import the game in almost any platform in the world. Unity's commitment to cross platform delivery is praised and well known [2]. From Unity we can build game for Pc platform, IOS, Mac, Tizen, Xbox, Web, Facebook, Samsung TV and many more. Unity revealed its windows version in spring 2009 for pc-based game. Later in 2013, Unity took license for IOS, android, blackberry, windows mobile and give access to develop game in these platform to unity free users as well [2]. It supports C# JavaScript, Ruby and rails as language to make a game. At first it's graphics condition was not that much good. In Unity 5 and above it makes a huge improvement in 3D sector and graphics sector. In Unity 2017 and above version, it makes some plugin for making game cinematics and improves render and texture quality even better for a better graphics [3].

The Basics of a game engine to make a multiplayer game is to take data or send data to a server or database and then use API or Services to establish a connection. On the other side, a game will be developed by a game engine, in our case the engine is Unity and port it in various platforms. Then through the connection or networking people will play the game in those platforms. The connection or networking here will be working in both ways.

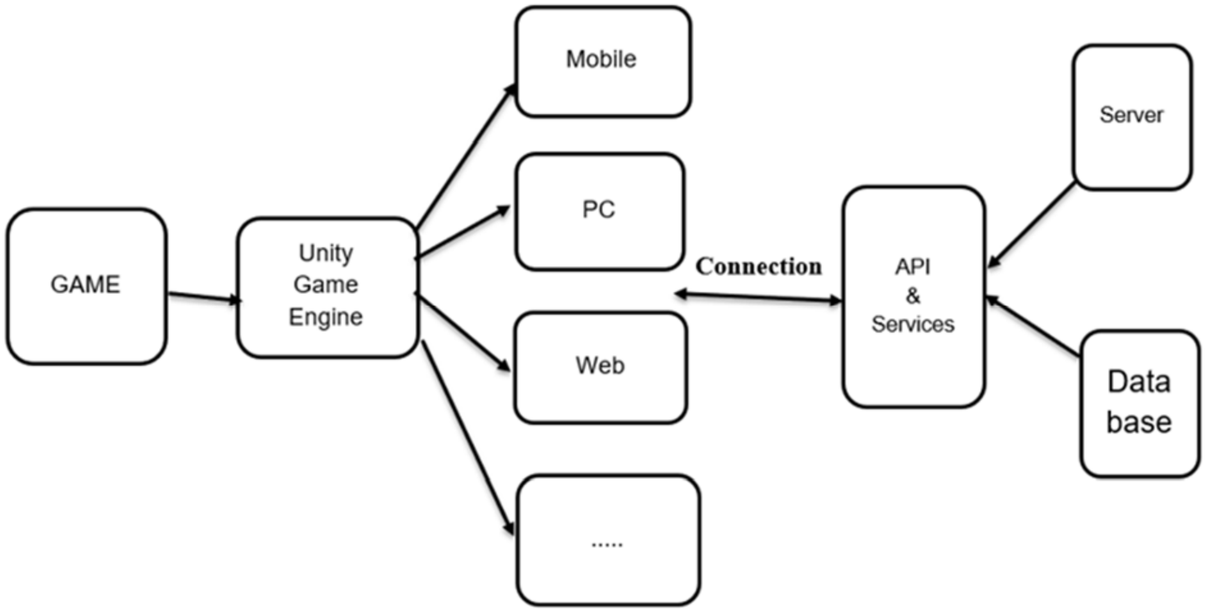


Fig.6. Multiplayer game architecture using Unity engine.

2.4 Online multiplayer game

Video Game is a recreational event for children from the very past time. But It is not just played by children. Nearly 58% of Americans play video games and the average gamer plays about 13 hours a week [4]. A big amount (68%) of video gamers are now adults (ESA, 2013) which in increasing day by day. Video games are often viewed as a source of distraction or worse, as a negative influence on people that play them. For instance, a large body of research has studied how game play violence can increase aggressive behavior, thoughts, and feelings in players [5]. As a player can interact with other people in the world, online multiplayer game has now become very popular throughout the whole world. It is making people globally social and also helps people in many different ways. A research said that gaming can be a social experience (Peña & Hancock, 2006) and suggests that social interaction in online game is a foundation of social capital (Steinkuehler & Williams, 2006). Many online game teaches teamwork, leadership, collaboration and cooperation (Raptr, 2014) [6].

There are mainly two kind of matchmaking system in multiplayer gaming environment. They are Host to client based system and client P2P system. In host to client system, a person creates a room where multiple client joins later. In client P2P system multiple game room exists and a player joins in the room. Then more player come through that player and more come from other player who joined later in the game room.

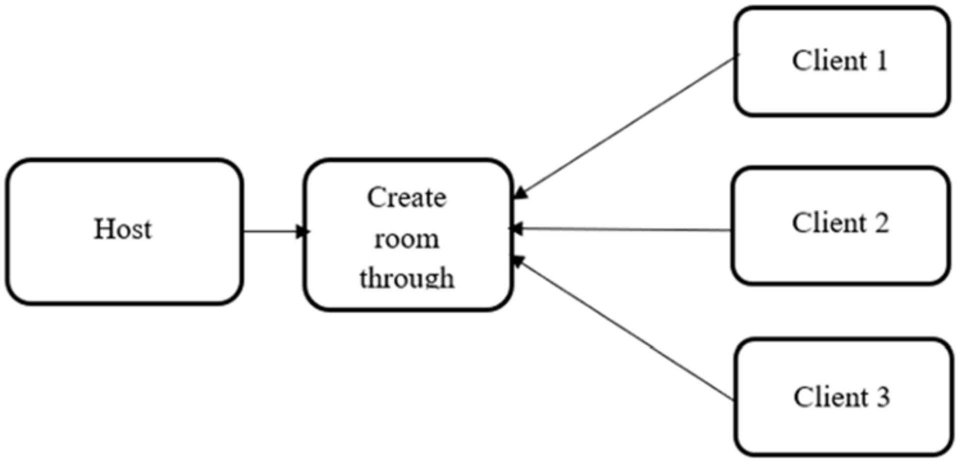


Fig.7. Online Multiplayer Diagram.

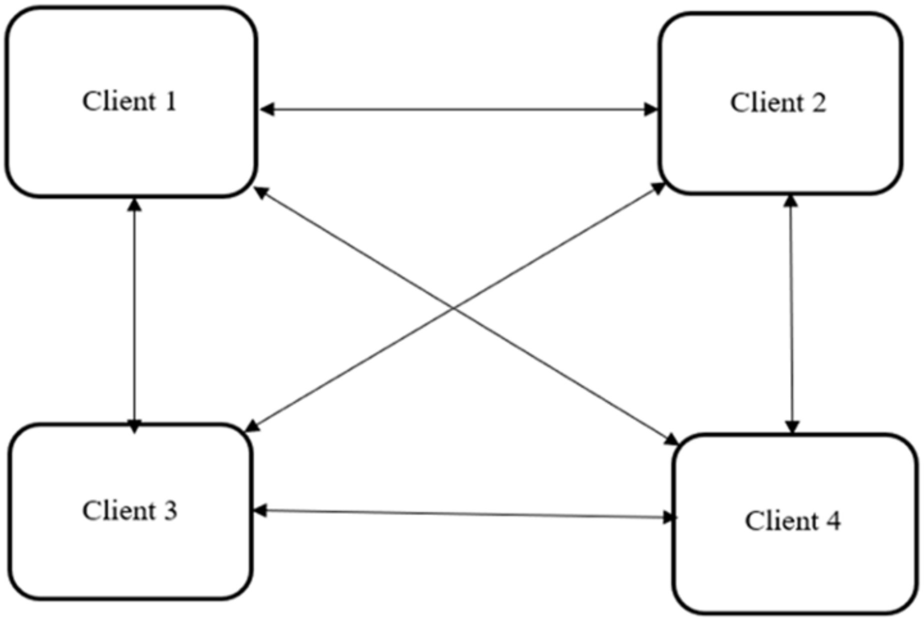


Fig.8. P2P Multiplayer Game.

CHAPTER 3

3. Proposed Method

We have built a multiplayer gaming framework using Kinect sensors and Unity game engine for Pc platform. Firstly, take the skeleton reading of a human from Kinect. Then import the Kinect reading to the game engine. After that rig the skeleton to a character so that the character will move the same way as the human player moves. Afterwards, connect it to network server and make it multiplayer so that user can play from far with one another through network. The framework will be a simple boxing type gaming framework where a player will get score when he hit an opponent's head and the score will update in network.

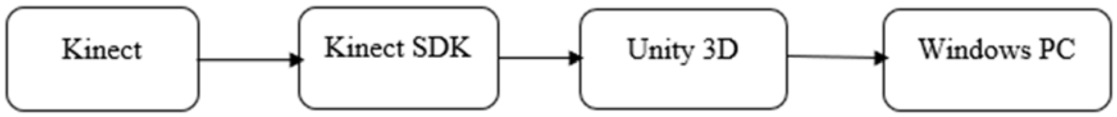


Fig.9. Basic demonstration of the project.

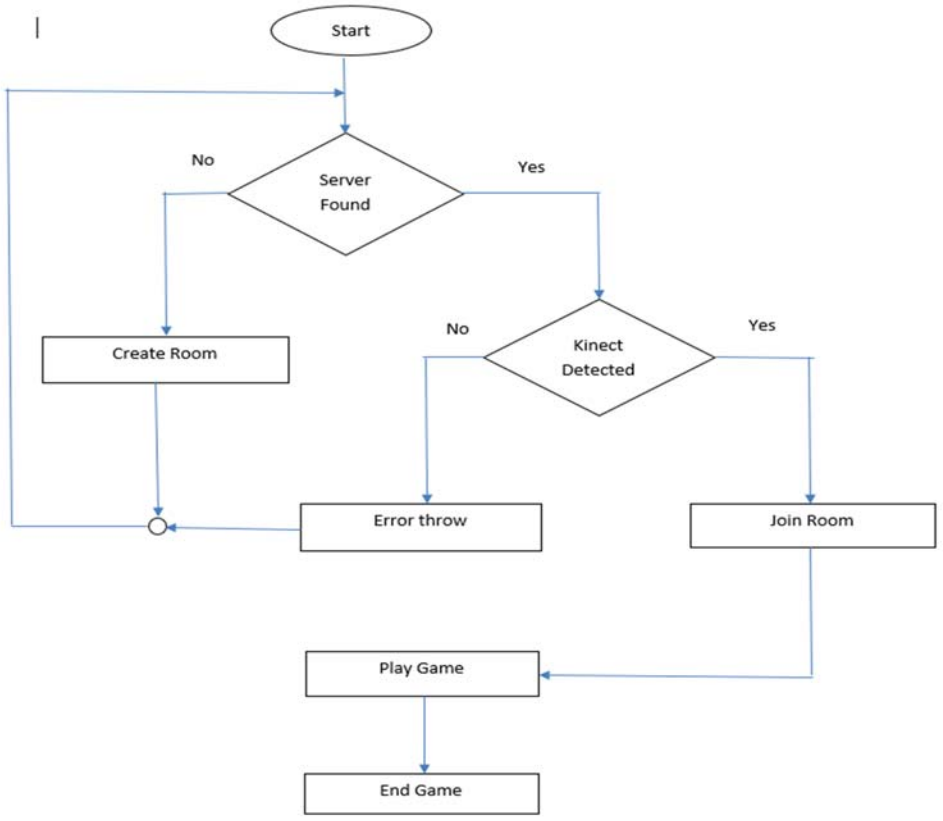


Fig.10. Project Flowchart.

3.1 Syncing skeleton with game character using Kinect SDK version 2 (v2)

The Kinect is used to take the real time data from user. Unity3D skeleton view is used and merged with Kinect sensor to read the gesture of the user. Then the task is to make the skeleton move like the user do in real time. But the Kinect has some fail detection if blind spot creates or speed of movement is too fast. So, considering these issues, try to minimize these problems and move the skeleton according to the user.

At the time when Kinect detects any human standing in front it creates a map of the whole body. The map is known as 'Skeleton'. After skeleton is generated we used every single of its position in the game character. Thus, the game character moves according to the real human player.

In Unity a library developed by Microsoft is provided for all types of Unity purposes. Kinect SDK v2 [15] supplies all the necessary libraries we need. By using “Windows. Kinect” library, we got all the body references of the human player. The following codes were used for assigning the skeleton data into a game character:

```
declaring bodySrcManager of GameObject type


storing body joints values in trackedJoint varriable of joint type


declaring bodyManager of BodySourceManager type


In the Start ()
if (bodySrcManager is null)
 do print("Assign Player")


else
 do bodyManager=bodyManager.GetComponent<BodySourceManager>() //detecting the whole
body of the user



In the Update ()


if (bodyManager is null)
  do return nothing


else
```

```
  do bodies= bodyManger.GetData()



foreach(var body in bodies)


  if(body.IsTracked)
    do pos=taking the body joints position value
```

Array named ‘bodies’ of ‘Body’ type stores all the skeleton data of a human. Which later on helps us to assign any body part to individual game Object [17].

3.2 UNET

After successfully taking user movement as input through Kinect, next step is to set up a networking environment on which, we would later integrate our previous method of taking user input through Kinect. We start by looking at a video on YouTube which demonstrates a simple way to implement online multiplayer environment with UNET [Quill18, youtube.com]. Despite the lag issues caused by the limitation of UNET online server, we managed to build a working set of player characters and their synchronized movement through network. This initial working player characters were implemented using simple spherical game objects of unity which had a script attached to them for movement to help test network server performance. Some of the initial lag issues originated from UNET framework’s proprietary NetworkTransform component which were solved by using unreliable data transmission on all channels on the proprietary NetworkManager component of the game environment.

3.3 Integration of skeleton and UNET

Finally, we replaced spherical player character with a Kinect detectable player character. This method turned out to not work because of the complete lack of support from Kinect library for

UNET and vice versa. So, in order to implement networking characteristics to Kinect detectable gameObject, we devised a roundabout way as stated in the process section.

3.4 Process

The process can be easily explained in the following steps:

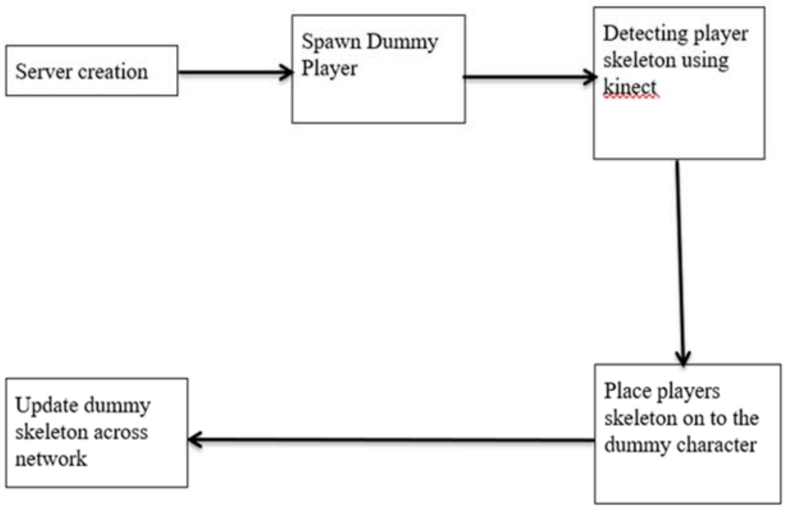

Server creation
Spawn Dummy Player
Detecting player skeleton using kinect
Place players skeleton on to the dummy character
Update dummy skeleton across network

Fig. 11. Project Steps.

1. Create server:

At first take an empty game object and put network Identity script on it to make the gameObject unique and specific as different player will be in the game. A spawnMethod script is added to the object to Instantiate the gameObject every time when a player joins in the game. Without network identity script unity cannot differentiate between different players. Instead of adding spawnMethod script in the player it was added to a normal game object and make a reference through a script to spawn our player. The reason behind that is to make the framework usable for every kind of game later on.. There is a problem if the object with network identity somehow gets destroyed. Then the id of the object through which it communicates with server will be destroyed as well. For example, in survival or shooting game, a player may die in some of the state of the game. So if the player with network identity is destroyed then the communication will also be destroyed. So we take a different gameObject and make it invisible so that it will not make any problem in real gameplay. The networkSpawn script build by UNET could not be used because it works for normal game object to spawn in a scene through network. But we are taking data through Kinect and here normal prefab generation doesn't work. Therefore, it was made manually and scripted separately.

2. Spawn dummy player skeleton:

In unity scripting start method contains the objects which is needed to be spawned in the game scene as soon as the game started. And the update method contains the things which will update in every frame of the game. So, update method needs to have all kinds of change scripted which we want to see in game like movement, action, death etc. But the problem here is writing anything in update method will affect every client connected in that game. So, the situation occur here is like, if I want to move my player leftward, then my all opponent player connected in the game will move leftward. So, the host player will control every player which is a very big issue. To solve that issue it is needed to specify to the server whose player is which one and the access of the individual players and other stuffs. In here use [Command] keyword is used. With this [Command] keyword we can make a method that gets called from the client and executed on the the server. So through this and with the help of unique network id we can differentiate a player control from another. These functions must begin with the prefix "Cmd" and cannot be static. [7]

3. Detect user's skeleton's movement using Kinect:

Kinect support 25 bones of a human to take data. Divide Kinect detectable player character in all 25 bones. We can make a character and then it can be implemented in it. But the character should have 25 bone parts to make the movement smooth. Otherwise it will look like as a body part has broken. Move the bone fragments rather than the whole character as a gameObject . At the start of the game, insure Kinect input and hide the initial player character's bone parts from the camera of the game scene.

At the initialization of networking, spawn a dummy character(bone joints) and replace it's transforms(x,y,z) and rotations with the original Kinect movable character's transforms(x,y,z) and rotations through scripts attached to each and every bone joints.

4. Place user's skeleton movements onto the dummy skeleton:

This effectively makes the bridge between Kinect library and UNET to perform synced movements of player characters over the network. Few things to note here are, firstly, there is no player character in literal sense, rather the accumulation of 25 bone parts that make the whole character. Secondly, the replacement method introduces a flickering issue that is a limitation of bandwidth of UNET. As of our understanding, unity relay services give roughly 4 kilobyte per second per client. [8] This amount is not enough to reliably propagate data as data are sent on each and every frame of the game which requires high bandwidth.

5. Update dummy skeleton ac:

For gaming issue UDP connection is used rather than TCP connection. Though TCP connection is reliable but if any packet is missing somehow then the game will be stuck until the packet is found. In the meantime other packets will come and cannot be accessed. So after a few time if the packet is not found then it will be in request timeout situation. And in case the packet is found then huge packet will be in a stack and mess things up. On the other hand if udp is used then if any packet is missing and not found and another packet comes in the meantime, then it will execute the new packet. It may cause some lag and packet loss but at least the game will run and not be in stuck situation. [9]

3.5. Environment

We designed a terrain or world for our game using blender and Maya. It will contain light effects, ground, various objects like trees, grasses, bushes and other objects. It is needed to handle shaders, effects, prefabs, particles, light balance, for making the environment perfect.

3.6 Character creation

To make an augmented reality based multiplayer game, game characters are needed. 3D game characters (i.e., 3D models) are created for the game. This whole system is constructed in unity which is a game engine but it can be used for 3D model rendering as well. There are different render engines such as V-Ray, Lumion and so on. They use different system algorithms for processing a 3D model for render. The game engines also have their own render system as well. In process of making a 3D game, we need to render the model on that specific game engine we are using for a better approach.

To make a 3D model compatible with Unity, we need to do it in 2 steps, these are

- Design/model
- Rigging

3.6.1 3D modeling

3-D modeling is the use of software to create a virtual three dimensional model of some physical object. This is a representation of an object in 3D.

For 3D modeling, there are several techniques that different software follows. such as

1. Box/ subdivision modeling
2. Edge/ Contour Modeling
3. NURBS/ Spline Modeling
4. Digital Sculpting
5. Procedural Modeling
6. Image-Based Modeling

From this list, we researched about each one of these techniques to know which one to follow and we found out that, we should go for Digital sculpting 3D modeling system to build a character in a more fast and efficient way. Digital sculpting system is now followed worldwide as a faster and accurate modeling.

For now, we've designed a 3D model of a warrior to test the whole system if it works, we can develop any character and object necessary in future.

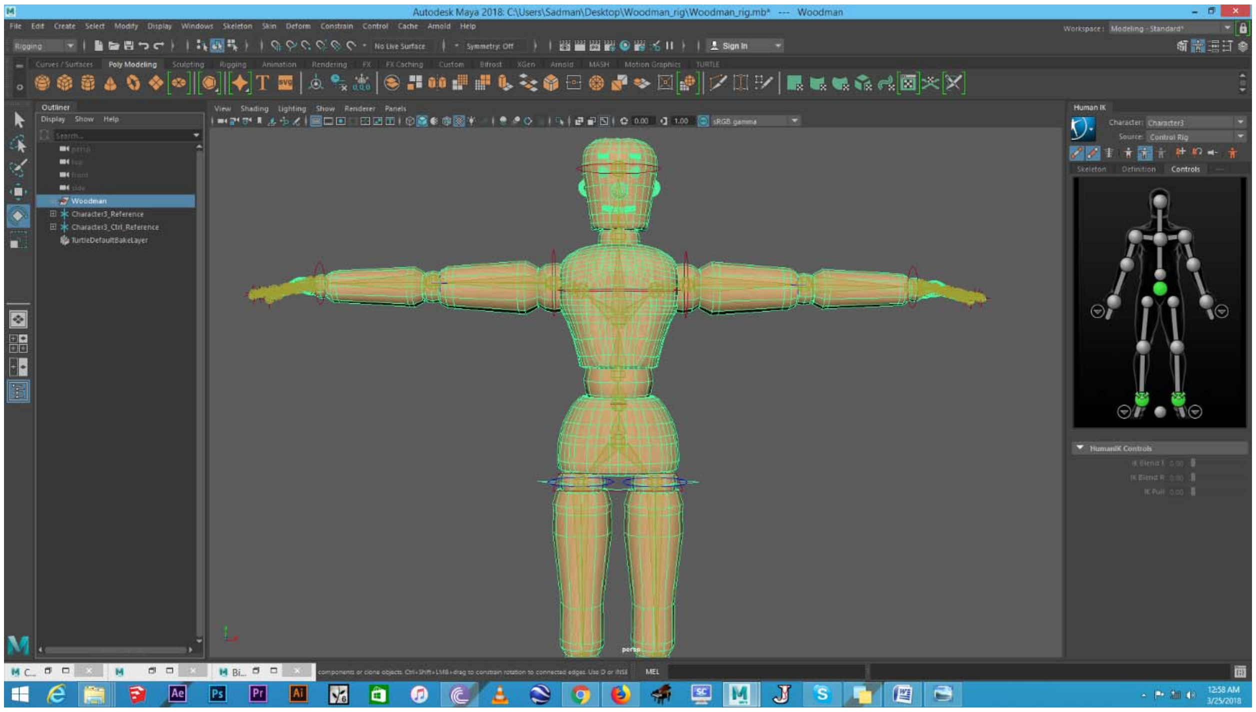


Fig. 12. Designing process 1.

3.6.2 Platform

Modeling a 3D character is not as easy as modeling lifeless objects such as tables or chairs. A human character has many curve shapes here and there. we used to use Sketchup for object building because it’s much more easier and time efficient to build it there. But in case of constructing a human body, we need to look after the curves too. Curves are much harder to handle in a linear approach.

In digital sculpting, meshes are created organically to mold and shape the model almost exactly like a sculptor would use rake brushes on a real chunk of clay. Digital sculpting has taken character and creature modeling to a new level, making the process faster, more efficient, and allowing artists to work with high-resolution meshes containing millions of polygons. Sculpted meshes are known for previously unthinkable levels of surface detail, and a natural (even spontaneous) aesthetic.

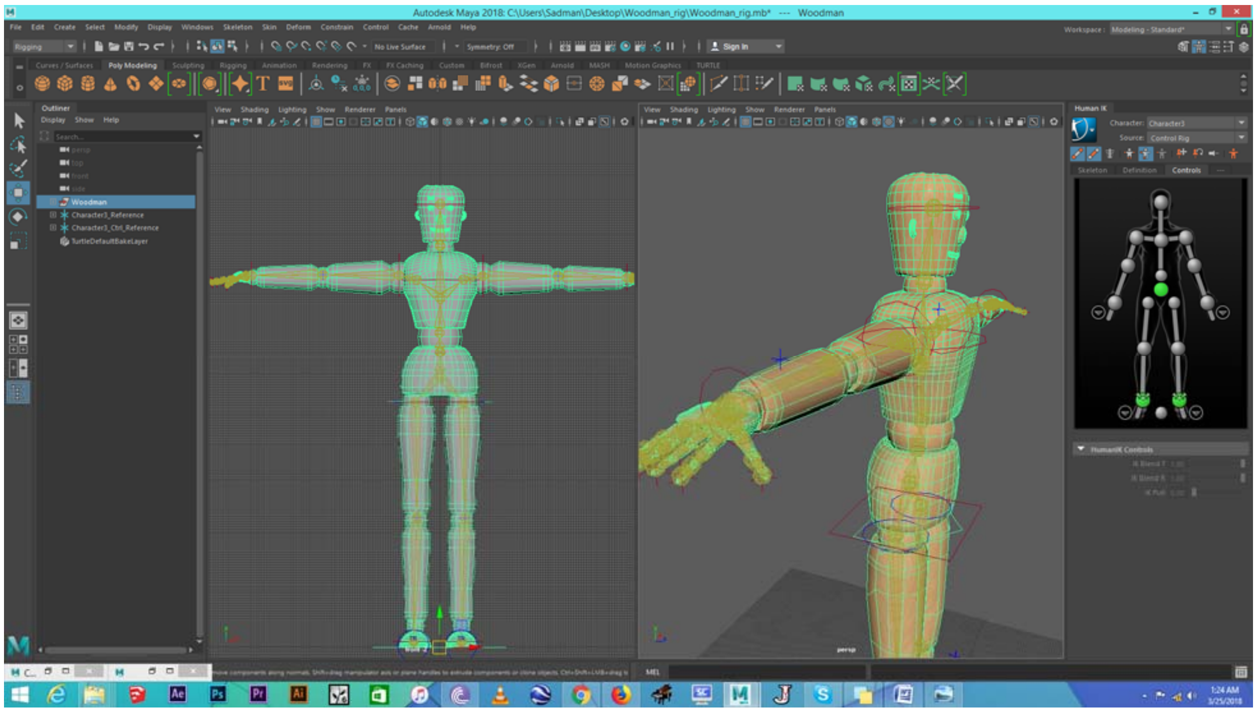


Fig.13. Designing process 2.

That is why we were searching for a better option for character modeling. We found a software which follows digital sculpting by Polygon based Modeling. This software we chose was AutoCAD Maya.

There are a variety of techniques you can use to create 3D polygonal models in Maya. Primitives are three-dimensional geometric shapes you can create in Maya. The primitive shapes available include spheres, cubes, cylinders, cones, planes, and many others. You can modify the attributes of basic primitives to make them more or less complex. You can also split, extrude, merge, or delete the various components on the primitive's polygon mesh using the various tools in the Modeling Toolkit in order to modify the primitive's shape. Many 3D modelers begin with polygon primitives as a starting point for their models. This technique is referred to as primitive-up modeling.

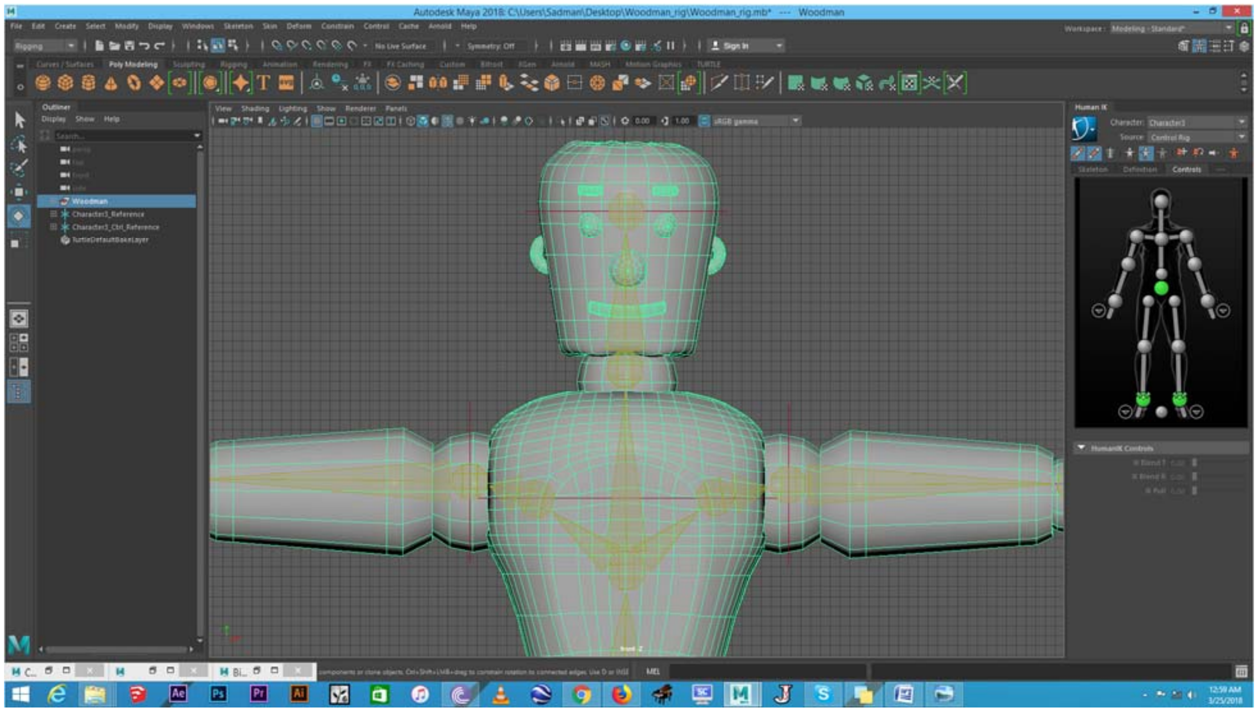

Autodesk Maya 2018: C:\Users\Sadman\Desktop\Woodman_rig\Woodman_rig.mb* --- Woodman
Outliner
Woodman
Character3_Reference
Character3_Ctrl_Reference
Human IK

Fig.14. Designing process 3.

Individual polygons can be created using the Create Polygon Tool or the Quad Draw Tool. Both these tools let you place individual vertices in the scene view to define the shape of individual polygon faces. You can then further split or extrude those faces to build out your polygon mesh. This polygon creation technique can be useful when you need to closely match a particular shape or outline. For example, if you needed to model a particular 3D logotype for an animated logo sequence or trace the outline of a 2D image using a bitmap image imported onto an image plane as a reference.

In AutoCAD Maya, we can work more précised by activating Vertex mode. In this mode, the main model is subdivided into a number of polygons. This number can be changed for a smoother or a rough model. The more subdivisions a model has, the smoother it gets. Every corner of the polygons is called Vertex by which you can move and change the shape of the model.

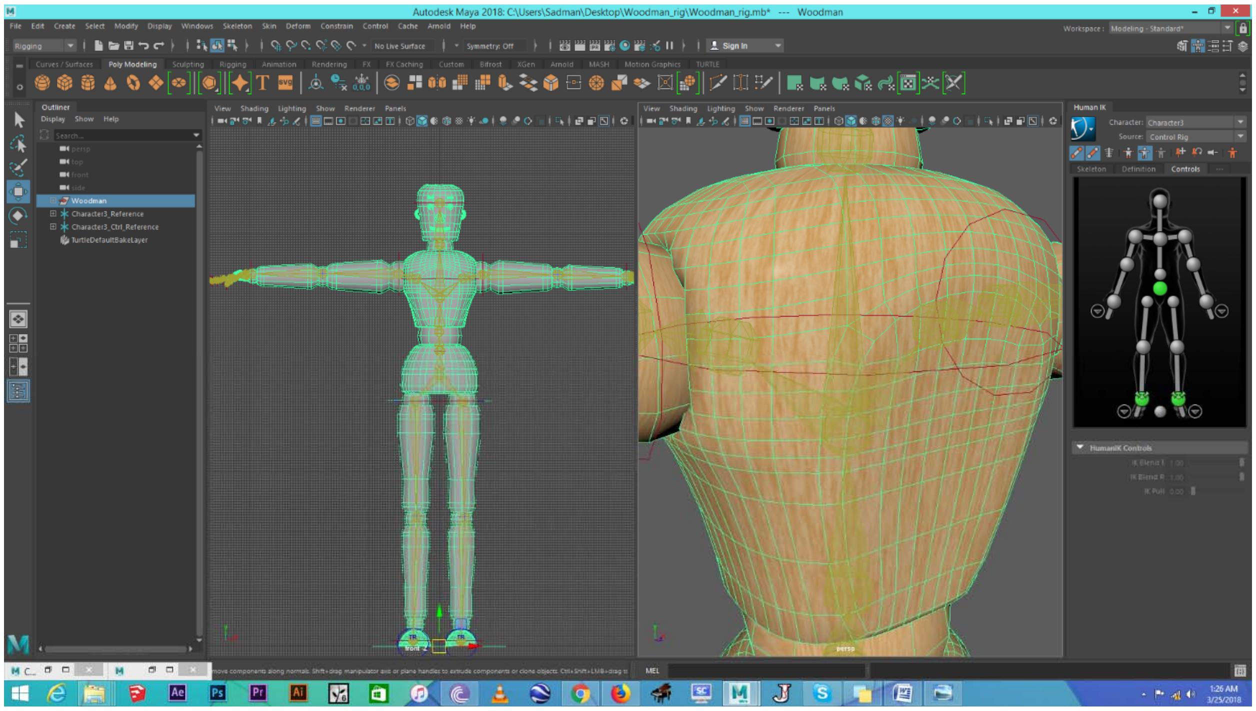


Fig.15. Designing process 4.

From the upper picture, the violate dots are called vertex. As you increase the sub divisions of the model, you can work for a smoother finish of the model which will be more accurate. This is the main plus point of using the polygon-based modeling system.

The best way to make a 3D model is to take reference pictures. we need three pictures as reference of a human body. The front view, the side view and the top view. we made a draft for the character first then used those as reference as the example given below.

3.6.3 Rigging

Once the character is modeled, you'll need to get it ready for animation. This process is called rigging. The goal of rigging is to add a skeleton and controls to your model so that an animator can manipulate and animate the character.

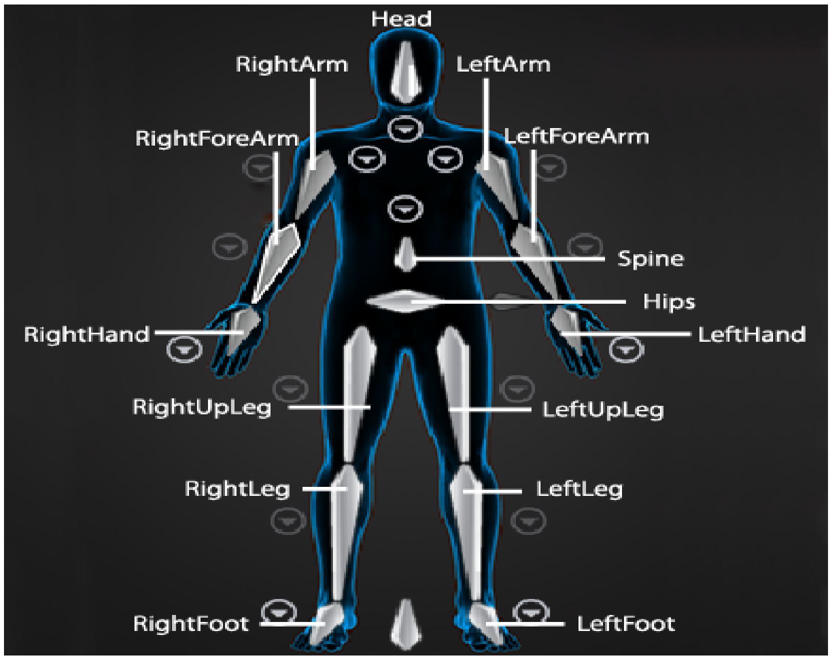


Fig.16. Rigging process 1.

Rigging is mainly the building process and synchronization of a skeleton underneath the body of the character. So, we need to know the bone design of a human being first before rigging. Its much better if we can find out a reference of the human bones, if we want it to be smooth. We need to build a skeleton of 20 bone joints at least. then we have the freedom to build rigs for fingers as well to have a smooth move.

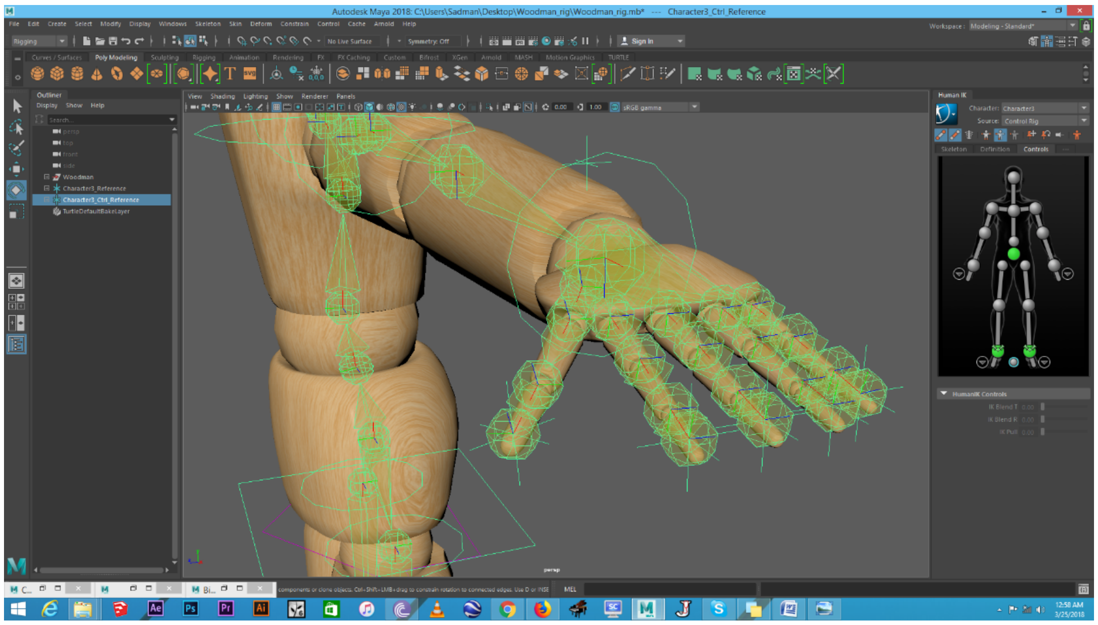
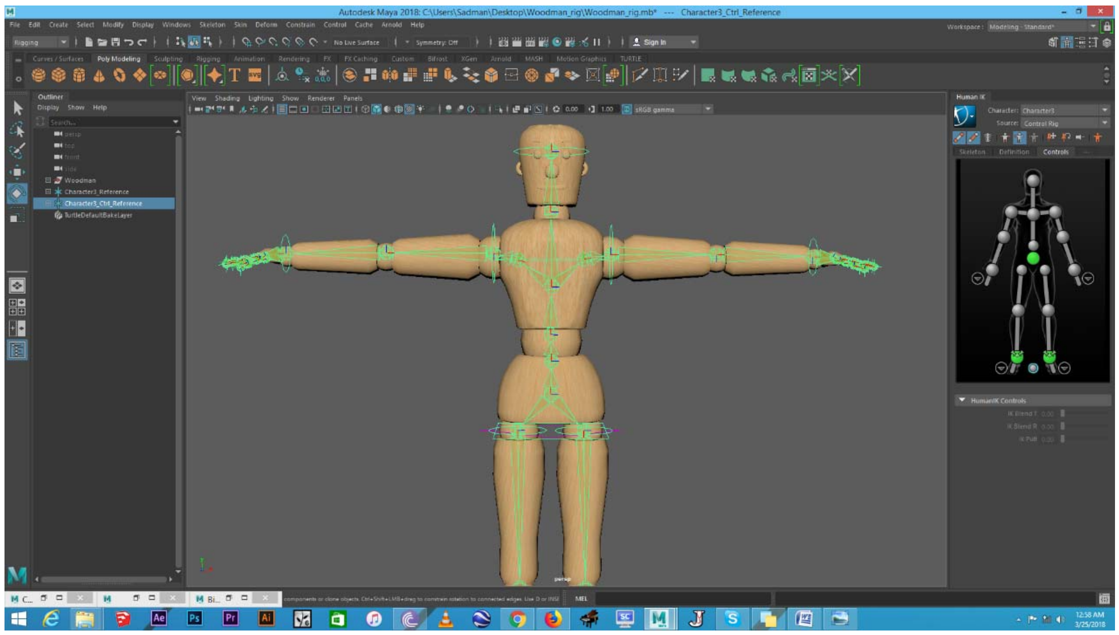

Autodesk Maya 2018: C:\Users\Sadman\Desktop\Woodman_rig\Woodman_rig.mb* --- Character3_Ctrl_Reference

Fig.17. Rigging process 2.

After placing all the bone joints in places, the skinning process skinning is the run time application of rigging. By skinning, the software basically determines the texture synchronization with the bones. There is a picture below, taken after the skeleton building and rigging process [20].


Autodesk Maya 2018: C:\Users\Sadman\Desktop\Woodman_rig\Woodman_rig.mb* --- Character3_Ctrl_Reference

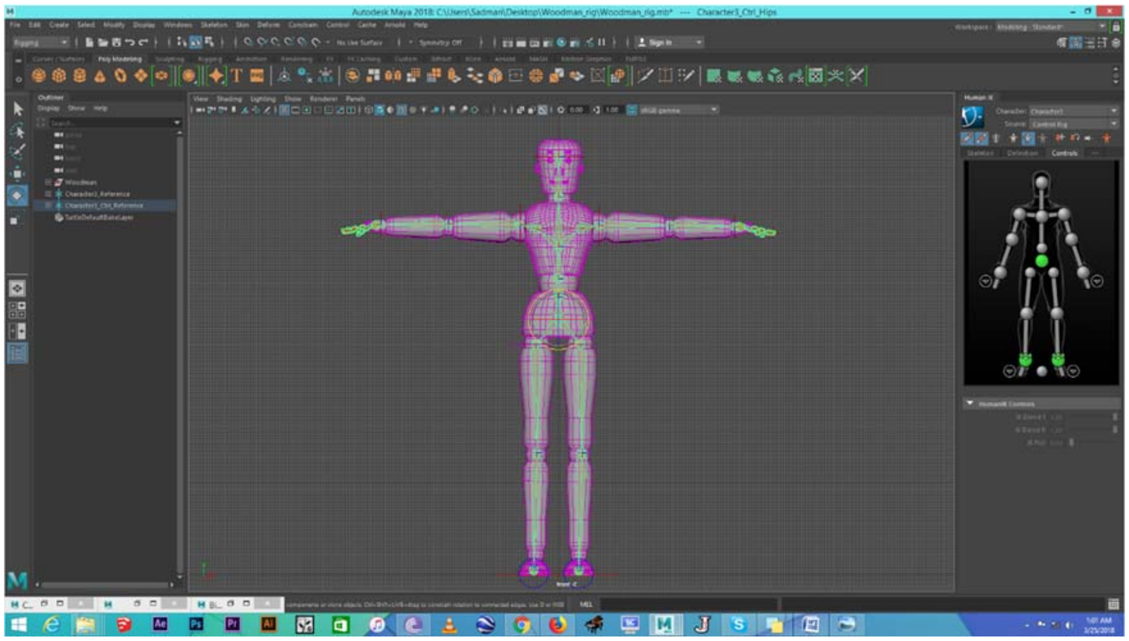

Autodesk Maya 2018: C:\Users\Sadman\Desktop\Woodman_rig\Woodman_rig.mb* --- Character3_Ctrl_Hips

Fig. 18. Rigging process 3.

3.6.4 Export

There are several exporting options in Maya. From all these options we chose FBX format export for this project. FBX is a format designed for interoperability between 3D content creation applications. There are several advantages to using FBX over a proprietary format like MB (Maya binary).
When Unity imports an FBX file, it can do so using its own FBX library. When Unity imports a proprietary file, it must do so using that software's API. This is slower and problematic. This is especially true with Maya, sometimes file imports fail due to the Maya API crashing many times. It might not even work at all in some cases [19].

The second advantage is that an FBX file only contains the information necessary to describe the model, materials, animations, etc. A proprietary file contains a bunch of information specific to the software, edit history and the like, that Unity doesn't need.

CHAPTER 4

4.Experimental Setup and Results

During the project a windows pc and Kinect were used. Online multiplayer game framework was developed using the following set up.

4.1 System Setup

In the experiment Intel core i7 was used as system CPU because unity needs multi-threading which works perfectly on this kind of CPU. For the graphics processing NVidia GeForce GTX 960m was capable enough to render the unity graphics engine. Since, Kinect SDK is not available for any other operating systems except windows, version 8.1 and 10 of windows were used. Configurations of the used equipment's are given below:

Table 1: CPU Specification.

CPU

Name	Intel core i7
Model	6700HQ
Clock Speed	2.6GHz
Core	4
Thread	8
Architecture	Skylake

Table 2: GPU Specification.

GPU

Name	NVIDIA GeForce GTX 960m
Core Clock	1096MHz
Cuda Core	640
Architecture	MAXWELL

Table 3: Software Specification.

Name	Type	Version	Architecture
Visual Studio 201**5**	Microsoft Development	2015	64 bit
Kinect SDK	Software development kit	2.0	64 bit
Unity 3D	Game engine	5.6	64 bit
Blender	3D Modeling		64 bit

4.2 Results

While building the character for Unity using Kinect, at first whole body part's references were taken using the library given by Kinect version 2 SDK(V2). After that those body references were implemented in default game object provided by Unity. It was checked simultaneously if Kinect was tracking human body perfectly or not. After the test was successful we replaced those default game objects with our rendered 3D model.

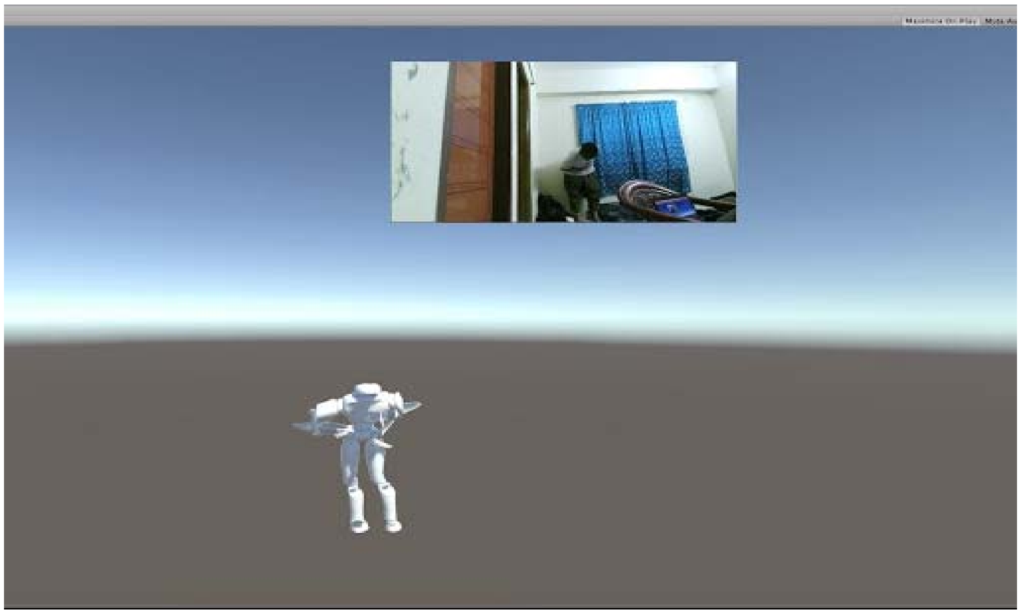


Fig.19. Rigged Character Movement Detection 1

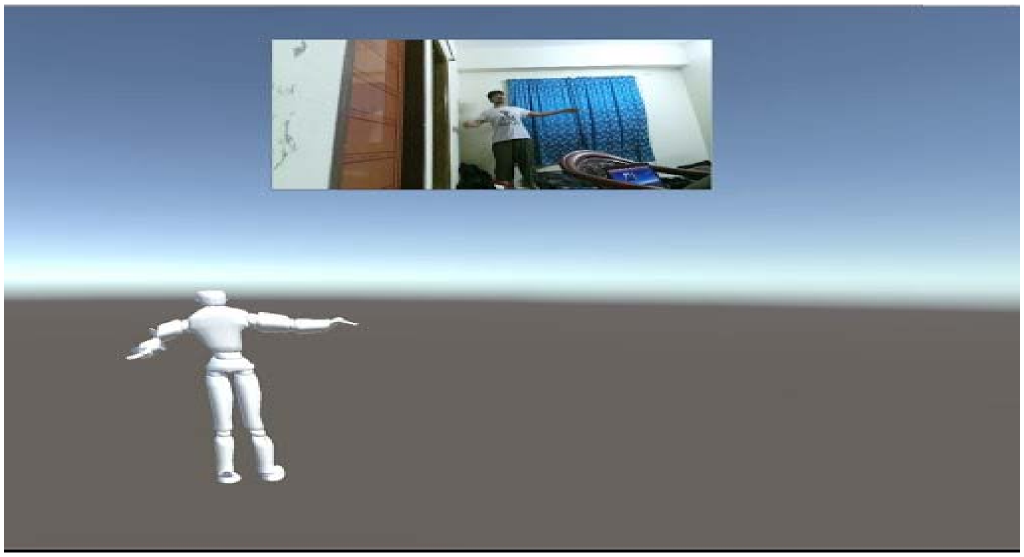


Fig.20. Rigged Character Movement Detection 2.

The results of this project were, moving a predesigned character using the input from Kinect, and update the movement data through network. One thing the proposed method failed to deliver in this project was the rigged body model we created. Then we code for some individual body parts and port them to the game scene. Below is a picture of the resultant game scene:

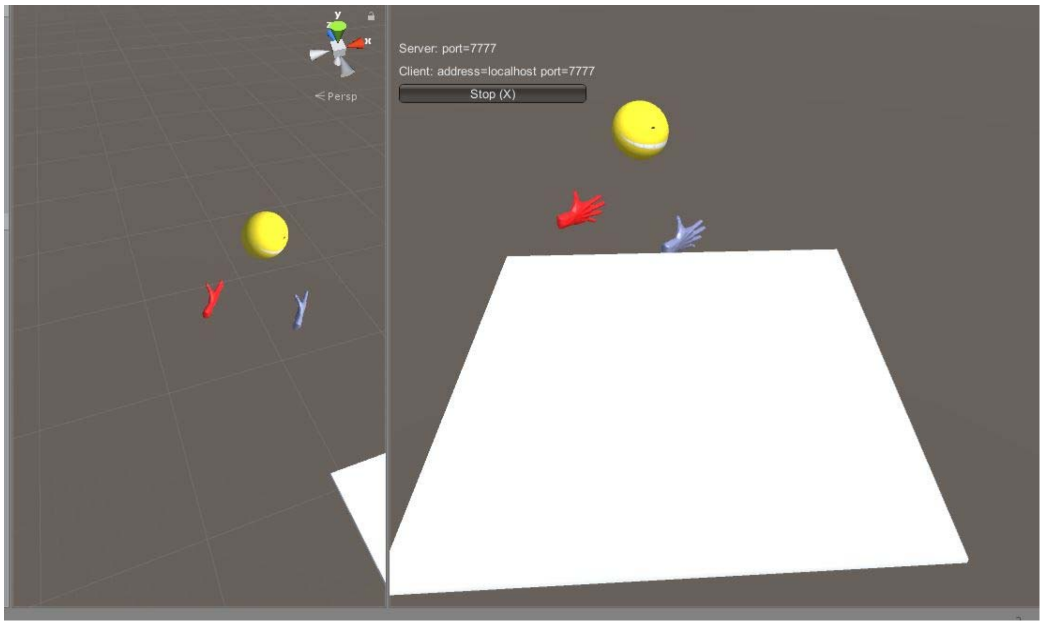


Fig.21. Obtained Result

After putting the terrain and some modification for our new build character body parts we get the final prototype figure for the project.

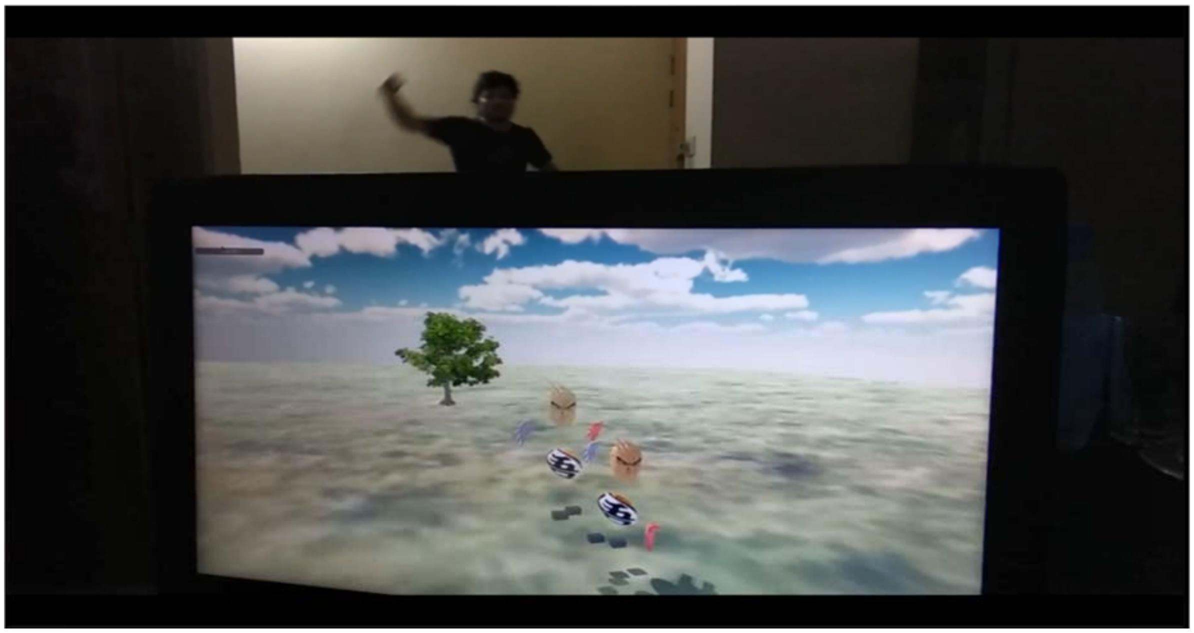


Fig.22. Final Framework Prototype.

As we get a big amount of flicker issue, we track the position of the player to ensure the player's spawning and positioning. The checking has done in both host pc and client pc and we get the same transform position value which indicates that out player is moving perfectly. But due to flicker issue we have some trouble to see the character in game scene. We took some data of different time which is shown in the table below –

Table 4: Data Set of Several Position in different time.

Transform Position	X axis	Y axis	Z axis
1.	1.2	10.2	-9.1
2.	0	0	4.1
3.	4	-4	16
4.	8.3	-4.3	17.2
5.	6.6	0.7	16.8

CHAPTER 5

5.Conclusion

After above mentioned findings, the end result was a framework that can be used to create kinect based real time multiplayer game. While our implementation contains character and environment design, note that anyone can use the basics of our findings to design their own character and environment as well as game logic. Though a developer may still find obstacles while implementing their own game logic in a Kinect based real time multiplayer game, this paper should give them necessary guidance throughout their work.

Future expectations and limitations:

In our project we have some limitations which we hopefully will overcome in future. Also we have some future expectations from our project that can be done in future as well.

Limitations:

A player needs to stay in a sufficient distance to track the skeleton perfectly. 3 to 4 meter is a good value in this case.
The room or environment should be spaced as much as possible so that less noise can affect. As we are using UNET free server so we are facing a flicker issue as the server response is not that good. There are also some partial lag exists.

Future expectations:

Kinect is using depth sensor and we are building a gaming framework for it. I Phone 10 is now also using depth sensor, and it is a matter of time that android will come with this. So developers can build game for mobile platform as unity supports cross platform development. They can also make a VR supported game with this. It is possible to use normal VR and a mobile app called Trinus VR to take data from pc and see it in VR double screen mode in mobile using cable only. In future we may lessen the flicker issue and port the game in a dedicated server.

References

[1] Yong Du, Wei Wang, Liang Wang, "Hierarchical Recurrent Neural Network for Skeleton Based Action Recognition," The Cvpr 2015 paper.

[2] Sue Blackman, 3D game Development in Unity 4, chapter 1 (Second edition)

[3] Unity version detail, www.Unity3D.com.

[4] NPD Group, Entertainment Software Association, 2013;

[5] Limperos et al., 2013; Anderson & Bushman, 2001, "Negative impact of gaming."

[6] Logan Molyneux, Krishnan Vasudevan, Homero Gil de Zúñiga, "Gaming Social Capital: Exploring Civic Value in Multiplayer Video Games," journal of computer mediated communication, (2011).

[7] Command Attribute detail,https://docs.unity3d.com/ScriptReference/Networking.CommandAttribute.html.

[8] Data send rate in unity https://unity3d.com/unity/features/multiplayer.

[9] Jouni Smed, Timo Kaukoranta, Harri Hakonen, "A Review on Networking and Multiplayer Computer Games.", Turku Centre for Computer Science, TUCS Technical Report No 454 April 2002.

[10] "Getting Connected". *Next Generation*. No. 19. Imagine Media. July 1996. p. 20. "There have been multiplayer electronic games since the dawn of computing. *Space War!* the first real video game, programmed by Steve Russell on the PDP-1, was an exclusive two-player game. So was Nolan Bushnell's pioneering coin-op *Pong"*.

[11] Wasserman, Ken; Stryker, Tim (December 1980). "Multi machine Games". *BYTE*. p. 24. Retrieved 18 October 2013

[12] Casper Harteveld and Geertje Bekebrede, "Learning in Single-Versus Multiplayer Games: The More the Merrier?", Simulation Gaming 2011 42: 43 originally published online 5 August 2010,

[13] Dawn Spring, "Gaming history: computer and video games as historical scholarship" American Public University, Charles Town, WV, USA, Published online: 06 Nov 2014.

[14] Y.H. Yang, W.Xu, H. Zhang, J.P.Zhang, "The Application of KINECT Motion Sensing Technology in Game-Oriented Study.", M.L. Xu, Zhejiang University, Hangzhou, China, Zhengzhou Universities, Zhengzhou, China, Published on: 20 March 2014

[15] Kinect sensor sdk detail, https://msdn.microsoft.com/en-us/library/hh438998.aspx

[16] Fakhteh Soltani, Fatemeh Eskandari, Shadan Golestan, "Developing a gesture-based game for deaf/mute people Using Microsoft Kinect.", Arak University Arak, Iran, Published on: 2012.

[17] Gesture control with Kinect and unity, https://channel9.msdn.com/Blogs/2p-start/Gesture-Control-with-Kinect-and-Unity-made-easy
[18] Kinect SDK features
https://docs.microsoft.com/en-us/previous-versions/windows/kinect
[19] Why should you use FBX format for imported meshes https://www.quora.com/Why-should-you-use-FBX-format-for-imported-meshes
[20] George Maestri, “Digital Character Animation 3”, Published Apr 12, 2006 by New Riders,