depth of the object, using this feature we

try to create

**Chapter No 1 Introduction**

**1.1 History**

We believe that virtual game development is an area in which we will enjoy working, but there is also a rapidly growing market in the gaming sector, despite the current economic downturn. If we look at the market share figures for gaming in the United States, we can see that the market share was worth USD$10.3 billion in 2004 and USD$65 billion in 2011.

Some game development companies are based in Ireland, but they had the opportunity to meet with many more and hear about their current projects. Redwind, Havock, and Popcap are examples of such businesses. On November 4, 2010, the Kinect was released. The SDK's beta version was released on June 16, 2011, and the official version was released on February 1, 2012. Kinect holds the Guinness World Record for "Fastest Selling Consumer Electronic Device," with 8 million units sold in the first 60 days of its release.

We could see the potential of Kinect when it first arrived on the market, and now, a year later, it has begun to fully exploit the technology's capabilities. Other technologies, such as the Wii or PlayStation Move, which both use motion technology, were considered. However, these technologies did not have as much functionality as our proposed method, which works similarly to the Kinect and allows you to track multiple elements without the use of a hand-held controller. Skeletal tracking, depth precision, voice control, and a standard camera are examples. It will have a significant impact on PCs without requiring the purchase of Kinect hardware.

|  |  |
| --- | --- |
|  | 1 |

Virtual Gaming Through Human Gesture Detection Using Kinect Introduction

**1.2 Necessity**

In today's world, people play motion sensing games on high-end, expensive consoles. These motion sensing games are made by companies like Microsoft, Sony, Nintendo, and Wii, and they are only compatible with consoles made by the same companies. Furthermore, these games are more costly than standard PC games.

There are numerous advantages to PC gaming over console gaming. Consoles cannot be upgraded to high graphics or processing power, whereas PCs can. The number of PC gamers worldwide is also significantly higher than that of console gamers. However, these PC games are played with a keyboard, mouse, and joystick. So, for this project, we concentrated solely on delivering a motion gaming experience on a PC.

Through this attempt people who want to play PC games as they play motions sensing games, they don’t need to buy costly consoles and Kinect hardware. All they need is to buy a standard camera and we will provide them a gesture set designed for specific input control.

**1.3 Objectives**

Objectives behind completion of this project are the following:

1. To improve gaming experience of PC gaming by enabling people to operate PC games through gestures.
2. To provide user friendly and easy to use gesture set for playing PC games.
3. To attract gamers toward PC gaming as there are lots of advantages of PC gaming over Console gaming.
4. To promote controller free gaming as this user will not need have games controller like keyboard, mouse or joysticks.
5. To offer experience of motion gaming for PC games without need of expensive gaming console and additional game controllers.
6. To deliver motion sensing gaming experience at very cheap cost as people will not need to buy consoles if they want to play normal PC games instead of expensive console games.
7. To investigate and research gesture technology using standard camera.
8. To investigate and research current uses of gesture recognition technology in current games.
9. To develop understanding of Python programming.

Virtual Gaming Through Human Gesture Detection Using Kinect Introduction

**1.4 Theme**

Following the above market trend, this project aims to improve the gaming experience by introducing a new way of working from an application standpoint, as there are no similar applications for Windows-based personal computers. The PC gamer will receive a gesture set tailored to the specific PC game. The main goal of the app, which sets it apart from other similar gaming apps, is the integration of motion detection and gaming application control, which eliminates the need to hold or even touch a keyboard, mouse, or controller device to play the game.

Although the application models will have fixed types of moves and gestures, the actual gaming environment will be able to mimic your movements, similar to motion sensing games. However, you will gain experience with motion sensing games.

This application may include a gesture set for a specific PC game. As a result, the user will only need a regular camera and not a gaming console. PC gaming has a number of advantages over console gaming. There are millions of PC games available on the market, and all of them can be played using gestures with this application.

**1.5 Problem Definition**

To implement cost effective system that read data from camera, detects human and accordingly perform action in order to

1. Play games through gestures (without using gaming consoles).
2. To control system functions through gestures.

|  |  |
| --- | --- |
|  | 3 |

Virtual Gaming Through Human Gesture Detection Using Kinect Literature Survey

**Chapter No.2 Literature survey**

**2.1 Introduction**

Human detection is typically done in images captured with visible light cameras. Due to variations in pose, clothing, lighting conditions, and background complexity, detecting humans in images or videos is a difficult task. In recent years, there has been a lot of research into human detection, and various methods have been proposed. The majority of the research relies on images captured with visible-light cameras, which is a natural way of doing things similar to what human eyes do. Some methods use statistical training based on local features, such as gradient-based features like HOG and EOH, and others use image extraction techniques like scale-invariant feature transform (SIFT) and others.

Because objects may not have consistent colour and texture but must occupy an integrated region in space, depth information is an important cue when humans recognise them. In recent decades, there has been research into using range images for object recognition and modelling. Range images have several advantages over 2D intensity images: they are resistant to colour and illumination changes. Range images are also simple representations of 3D data. However, due to lasers, earlier range sensors were expensive and difficult to use in human environments. Microsoft has now released the Kinect, which is both inexpensive and simple to use. It also lacks the drawbacks of lasers, allowing it to be used in a human environment and facilitating research in human detection, tracking and activity analysis.

They presented a novel model-based method for human detection from depth images in the paper to which we referred. This method detects people's poses in indoor environments using standard camera data. It uses a pose detection process that includes a 2D edge detector that uses both the depth image's edge information. As a result, we've decided to use this method to detect human gestures and send commands to the back-end application, Games.

|  |  |
| --- | --- |
|  | 4 |

Virtual Gaming Through Human Gesture Detection Using Kinect Literature Survey

**2.2 Existing System-**

Today peoples are using high end gaming consoles and Kinect hardware for playing motion sensing games. The existing systems include popular gaming consoles like WII, Nintendo, PS1, PS2, PS3, and XBOX 360. Systems like WII or Play station Move which both use motion technologies. However, these technologies did not have as much functionality as our proposed method, which works similarly to the Kinect and allows you to track multiple elements without the use of a hand-held controller. These can include skeletal tracking, depth precision, voice control and a standard camera. These systems are based on the light sensing technology and require joysticks, light emitting devices etc.



Figure 2.1 Existing Devices of Play station

|  |  |
| --- | --- |
|  | 5 |

Virtual Gaming Through Human Gesture Detection Using Kinect Literature Survey

**2.3 Limitations of existing system: -**

* Console is required for playing games.
* The existing system can only be used for playing games.
* Light emitting sensors or joysticks are required in existing systems such as ps3 Nintendo, WII.
* Most of the games are compatible only with particular gaming console manufacturers.

**These motion sensing games are not compatible with PCs or laptops.**

|  |  |
| --- | --- |
|  | 6 |

Virtual Gaming Through Human Gesture Detection Using Kinect SRS

**Chapter 3. Software Requirements Specification**

****

**3.1 Introduction**

Kinect is used with a console device for applications. We are attempting to solve the problem of skeletal tracking of a human body without using the Microsoft Kinect sensor because it is expensive, so we are attempting to optimise hardware by removing console devices and Kinect hardware.

**3.1.1 Product Scope**

Because of a number of complications such as variations in pose, lighting conditions, and the complexity of the background in the tracking environment, image processing-based human recognition remains a difficult task. This project presents a method for tracking human gestures using image processing techniques and pose data from a standard camera.

**3.1.2 User Classes and Characteristics**

The users are most likely advanced gamers, children and people having hobby as gaming.

**3.1.3 Operating Environment**

* Windows 7 or 8 Onwards
* Intel x86/x64 architecture.
* The system is built on top of the Windows operating system. Windows 7 or 8 is required for the running python.

|  |  |
| --- | --- |
|  | 7 |

Virtual Gaming Through Human Gesture Detection Using Kinect SRS

**3.1.4** **Design and Implementation Constraints**

The system performs its function using Python libraries like OpenCV. As far as the customer is concerned, all that is required on his or her machine is a standard camera and the Python framework.

**Software Requirements**

* Windows 7/8 onwards
* Anaconda
* Python Programming

**Hardware Requirements**

* Pentium Processor IV or Higher
* Min 10 GB HDD
* RAM: 512MB or higher
* 2.66 GHz or faster processor
* Standard Camera.

**3.1.5 Assumptions and Dependencies**

* User has a large enough workspace area to accommodate the usage of a camera field of view.
* User has a computer system running Windows Operating System
* The user has a powerful enough computer system that can handle the additional processing associated with OpenCV and frame media pipe introducing pose detection into the system.

**3.2 System Features**

1. System eliminates the console and allows us to play motion sensor games.
2. System detects human motion by using pose detection with maximum accuracy.
3. System allows us to play existing computer games using motion sensing technique.

|  |  |
| --- | --- |
|  | 8 |

Virtual Gaming Through Human Gesture Detection Using Kinect SRS

1. System provides a user-friendly interface.
2. System provides us ability to operate Computer systems functionality through gestures.

**3.3 External Interface Requirements**

**3.3.1 Communications Interfaces**

* USB is used to communicate between Camera and PC
* Keyboard and Mouse is used to emulate control.

**3.4 Non-functional Requirement**

**3.4.1 Performance Requirements**

* The user should stand between 2 and 8 feet from camera.
* The system should run in Real-time
  + Should be able to track only 1 person in the Workspace.
  + Should be able to scale up to detect human body pose in real time.
  + Should be able to run on commercially available computing platforms
  + Should be able to emulate input from mapped posed detected.

**3.4.2** **Safety Requirements**

* The workspace should have no obstacles or obstructions in the way of the user.
* Elongated continuous usage of the system may be harmful, so it is highly recommended that two hours continued use sessions are followed by a fifteen-minute break with the system powered off.
* A very small percentage of people may experience a seizure when exposed to certain visual images, including flashing lights or patterns that may appear in video games. Even people who have no history of seizures or epilepsy may have an undiagnosed condition that can cause these “photosensitive epileptic seizures” while observing the displays.

**3.4.3 Software Quality Attributes**

* **Robustness** - The system needs to be robust enough to generate a realistic virtual experience. The system should also be able to dynamically detect motions and body pose in real-time to generate accurate environment. The system needs to properly send system input based on detected pose.
* **Reliability** - The system should able to continuously run for a long duration of time (multiple hours) and not suffer from system slowdowns or crashes caused from memory leaks and zombie process.
* **Portability** - The software should be able to run on any Microsoft windows-based platform. To set up and tear down the entire system, displays need to be set in place. Camera need to be placed on the displays, Camera need to be connected to the computer, and the program need to be executed. Ideal set up/tear down time should be approximately two minutes.
* **Ease of use** - Someone with little or no technical experience in the operations of electronics should be able to setup and use this system by following a simple set of instructions.
* **Ease of Learning** - The learning curve for this software should be short since the software should perform the corresponding tasks based on natural human motions.

|  |  |
| --- | --- |
|  | 12 |

Virtual Gaming Through Human Gesture Detection Using Kinect System Design

**Chapter No. 4 Proposed Method**

****

**4.1 System Architecture**

**4.1.1 Proposed Architecture**

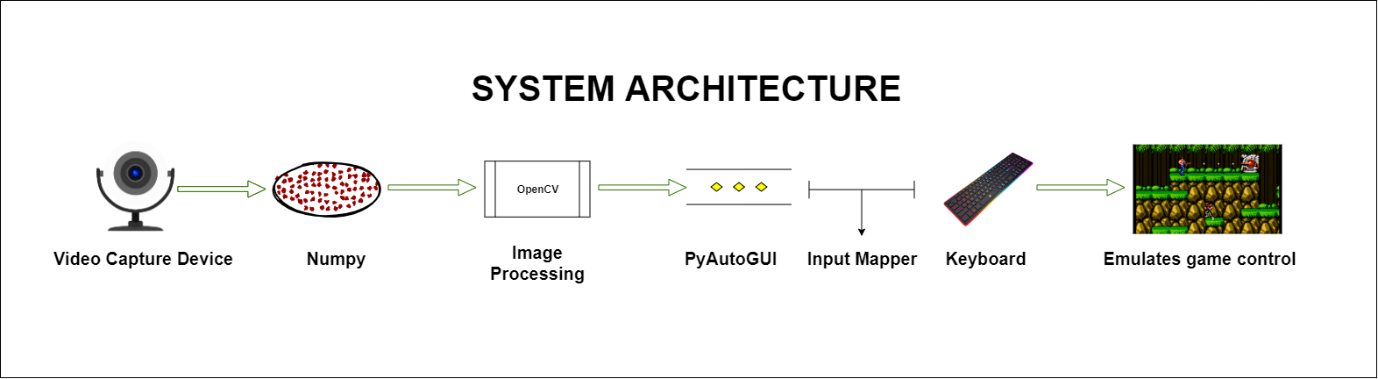
****

Fig 4.1 Architecture

**4.2 Components of Architecture**

**4.2.1 Video Capture Device:**

Video capture device used here is a standard webcam which either comes inbuilt or has to be connected externally via a USB cable. A webcam is a video camera that sends or streams an image or video to or through a computer network, such as the Internet, in real time. Small cameras that sit on a desk, attach to a user's monitor, or are built into the hardware are known as webcams. Webcams can be used during a video chat session between two or more people, with live audio and video conversations. Users can use webcam software to record or stream video over the Internet. Because video streaming over the Internet consumes a lot of bandwidth, compressed formats are commonly used. Because higher resolutions would be reduced during transmission, a webcam's maximum resolution is also lower than most handheld video cameras. Webcams are less expensive than most video cameras due to their lower resolution, but the effect is sufficient for video chat sessions. The term "webcam" (a clipped compound) can also refer to a video camera that is connected to the Internet indefinitely rather than for a specific session, and that provides a view for anyone who visits its web page over the Internet. Some of them are expensive, rugged professional video cameras, such as those used as online traffic cameras.

A webcam is a computer-connected camera. It can take still photos or video, and with the help of software, it can stream video over the Internet in real time. A webcam, such as the Logitech Webcam C270, is shown in the fig 4.2.1.

Most webcams today are either built into the top edge of a laptop's display or connected to the computer's USB or FireWire port.



Fig 4.2.1 – Logitech c270 HD webcam

The XCoffee, also known as the Trojan Room coffee pot, was the first widely publicised webcam. With the help of Quentin Stafford-Fraser and Paul Jardetzky, the camera went live in 1991. In November 1993, Daniel Gordon and Martyn Johnson assisted in connecting the camera to the Internet.

People didn't waste time going down to get coffee because the XCoffee webcam monitored a coffee pot outside the Trojan Room in the University of Cambridge. The website had over 150,000 people online watching the coffee pot after it was mentioned in the news. On August 22, 2001, the webcam was turned off. The image depicts how the XCoffee appeared on the internet.

**4.2.2 NumPy:**

NumPy is the most important Python package for scientific computing. It's a Python library that includes a multidimensional array object, derived objects (such as masked arrays and matrices), and a variety of routines for performing fast array operations, such as mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation, and more.

The Nd array object is at the heart of the NumPy package. Many operations are performed in compiled code for performance, and this encapsulates n-dimensional arrays of homogeneous data types. The following are some key distinctions between NumPy arrays and standard Python sequences:

* Unlike Python lists, NumPy arrays have a fixed size when they are created (which can grow dynamically). When you change the size of a Nd array, it creates a new one and deletes the old one.
* The elements of a NumPy array must all be of the same data type and thus have the same memory footprint. The exception is that arrays of (Python, including NumPy) objects can be used, allowing for arrays of various sizes.
* NumPy arrays facilitate advanced mathematical and other types of operations on large numbers of data. Typically, such operations are executed more efficiently and with less code than is possible using Python’s built-in sequences.
* NumPy arrays are used by a growing number of scientific and mathematical Python-based packages; while they typically support Python-sequence input, they convert it to NumPy arrays before processing, and they frequently output NumPy arrays. To put it another way, knowing how to use Python's built-in sequence types isn't enough to efficiently use much (if not all) of today's scientific/mathematical Python-based software; you also need to know how to use NumPy arrays.

It is the most important Python package for scientific computing. It is an open-source programme. It has a number of features, including the following:

* A powerful N-dimensional array object
* Sophisticated (broadcasting) functions
* Tools for integrating C/C++ and Fortran code
* Useful linear algebra, Fourier transform, and random number capabilities

NumPy can be used as a multi-dimensional container of generic data in addition to its obvious scientific applications. NumPy allows arbitrary data types to be defined, allowing NumPy to integrate with a wide range of databases seamlessly and quickly.

NumPy is a widely used general-purpose library that serves as the foundation for many other computation libraries, including SciPy, scikit-learn, TensorFlow, matplotlib, and OpenCV. Having a basic understanding of NumPy makes it easier to work with other higher-level libraries.

Virtual Gaming Through Human Gesture Detection Using Kinect System Design

**4.2.3 OpenCV**

OpenCV (Open-Source Computer Vision Library) is an open-source library that contains hundreds of computer vision algorithms. Computer vision is a method for comprehending images and videos, as well as manipulating and retrieving information from them. Artificial Intelligence relies on or is primarily based on computer vision. Self-driving cars, robotics, and photo correction apps all use computer vision to some extent.

OpenCV is a large open-source library for computer vision, machine learning, and image processing, and it now plays an important role in real-time operations, which are critical in today's systems. It can be used to identify objects, faces, and even human handwriting in images and videos. Python can process the OpenCV array structure for analysis when it is combined with other libraries like NumPy. We use vector space and perform mathematical operations on these features to identify image patterns and their various features.

The first version of OpenCV was 1.0. OpenCV is free for both academic and commercial use because it is released under a BSD licence. It supports Windows, Linux, Mac OS, iOS, and Android and has C++, C, Python, and Java interfaces. The main focus of OpenCV when it was created was real-time applications for computational efficiency. To take advantage of multi-core processing, everything is written in optimised C/C++.

Applications of OpenCV: There are lots of applications which are solved using OpenCV, some of them are listed below

* face recognition
* Automated inspection and surveillance
* number of people – count (foot traffic in a mall, etc)
* Vehicle counting on highways along with their speeds
* Interactive art installations
* Anomaly (defect) detection in the manufacturing process (the odd defective products)
* Street view image stitching
* Video/image search and retrieval
* Robot and driver-less car navigation and control
* object recognition
* Medical image analysis
* Movies – 3D structure from motion
* TV Channels advertisement recognition

Image processing is a technique for performing operations on images in order to improve them or extract useful information from them. If we talk about the basic definition of image processing then “Image processing is the analysis and manipulation of a digitized image, especially in order to improve its quality”.

The goal of image processing is to assist a computer in comprehending an image's content. OpenCV is a programming library primarily used for image processing. It serves as the de facto API for computer vision applications. Image processing applications can help us solve a variety of real-time problems.

Image processing is a type of signal processing in which the input is an image, such as a photograph or video frame, and the output is an image or a set of image-related characteristics. OpenCV is a programming library primarily used for image processing. The open source Berkely Software Distribution licence makes it freely available. Intel began the project as a research project. OpenCV includes a number of tools for dealing with computer vision issues. It includes low-level image processing functions as well as high-level face detection, feature matching, and tracking algorithms.

We are humans, and we can easily discern that the image is of a person who is me. However, if we ask the computer, "Is this my photo?" The computer is unable to respond because it is unable to solve the problem on its own.

Any image is read by the computer as a range of values between 0 and 255. There are three primary channels in any colour image: red, green, and blue.

OpenCV's main user interface is written in C++. Python, Java, and MATLAB now have complete user interfaces. Wrappers for other languages like C#, Perl, and Ruby have been created. Since 2010, a CUDA-based GPU interface has been in development. OpenCV has received support from Intel, as well as Willow Garage, a privately funded new robotic research institute. OpenCV runs on a variety of operating systems, including Windows, Android, Blackberry, OpenBSD, iOS, and Linux. The development of new OpenCV modules to support robotic perception is currently underway.

**4.2.4 PyAutoGUI**

PyAutoGUI allows your Python scripts to automate interactions with other applications by controlling the mouse and keyboard. The API is intended to be straightforward. PyAutoGUI runs on Python 2 and 3 and works on Windows, macOS, and Linux.

PyAutoGUI has several features:

* Moving the mouse and clicking in the windows of other applications.
* Sending keystrokes to applications (for example, to fill out forms).
* Take screenshots, and given an image (for example, of a button or checkbox), and find it on the screen.
* Locate an application’s window, and move, resize, maximize, minimize, or close it (Windows-only, currently).
* Display alert and message boxes.

**4.2.5 Standard Input Device**

A keyboard and mouse are used for the majority of computer interactions. The mouse allows the user to position the cursor, draw, and execute programme functions by clicking mouse buttons, while the keyboard allows the user to type letters and numbers. Many other actions can be performed on the keyboard using function keys, cursor keys, control keys, or keyboard macros in addition to basic typing. However, the mouse duplicates many of these actions. Many people nowadays prefer to use a mouse instead of keyboard keys to operate their computers. It's important to align the centre of your body with the centre of the keyboard part you use the most, which is usually either the alphabetic area or the numeric keypad, for good posture.



Fig 4.2.2 Keyboard & Mouse

**4.2.6 Emulator**

A video game console emulator allows a computing device to emulate the hardware of a video game console and play its games on the emulating platform. Emulators frequently include extra features that go beyond the limitations of the original hardware, such as broader controller compatibility timescale control, increased performance, improved clarity, easier access to memory modifications (like Game Shark), one-click cheat codes, and unlocking of gameplay features. Emulators are also useful in the creation of new games for older, discontinued, or rare consoles, as well as in the development of homebrew demos.

A ROM file (a copy of game cartridge data) or an ISO image (a copy of optical media) is typically used to supply the emulator with the game's code and data. These files are created by either specialised tools for game cartridges or regular optical drives reading the data. [1] Despite the increasing length of time between the discontinuation of the original system and product, most games retain their copyright, forcing regular consumers and emulation enthusiasts to resort to downloading games for free from various websites rather than purchasing and ripping the contents legally (although for optical media, this is not uncommon for legitimate owners). As an alternative, specialized adapters such as the Ret rode allow emulators to directly access the data on game cartridges without needing to copy it into a ROM image first.

Emulators can be designed in one of three ways: purely software, which is the most common, such as MAME using arcade ROM images; purely hardware, such as the ColecoVision's adapter to accept Atari VCS cartridges; or hybrid solutions, such as hardware bridgeboards for various Amiga computers that could run IBM PC-compatible software.

To avoid any potential conflicts with non-public intellectual property, emulators are typically created by reverse engineering hardware information. Some hardware specifications may be made public for developers to use in their emulation efforts, but there are often layers of information that remain as trade secrets, such as encryption details. Operating code stored in the BIOS of hardware can be disassembled and analysed in a clean room setting, with one person disassembling and another documenting the code's function separately. An emulation on the target hardware can be built once enough information about how the hardware interprets the game software has been gathered. To avoid contaminating the clean room nature of their project, emulation developers typically avoid any information that may come from untraceable sources. In 2020, for example, a large trove of information about Nintendo's consoles was leaked, and teams working on Nintendo console emulators like the Dolphin emulator for GameCube and Wii stated that they were staying away from the leaked information to avoid tarnishing their project.

Following the creation of an emulator, a copy of the game software must be obtained, a step that may have legal ramifications. This usually necessitates the user copying the contents of the ROM cartridge to computer files or images that the emulator can read, a process known as "dumping" the contents of the ROM. Other proprietary formats, such as PlayStation CD games, use a similar concept. While it is not required for emulation of the first arcade or home console, most emulators also require a BIOS dump, which varies depending on distribution region and hardware revisions. Emulators can sometimes be used to apply ROM patches, which update the ROM or BIOS dump to fix incompatibilities with newer platforms or change aspects of the game. The emulator then emulates the hardware using the BIOS dump, while the game software is replicated using the ROM dump (with any patches).

Although the primary purpose of emulation is to make older video-games execute on newer systems, there are several advantages inherent in the extra flexibility of software emulation that were not possible on the original systems.

* ROM hacking and modification
* Enhanced technical features
* Bypassing regional lockouts
* Cheating and widescreen functionality

Here is the list of some popular video game emulators

* FCEUX
* NESticle
* Nestopia
* AdriPSX
* bleem!
* bleemcast!
* Connectix Virtual Game Station
* ePSXe
* PCSX-Reloaded

Virtual Gaming Through Human Gesture Detection Using Kinect System Design

**4.3 Algorithm and Flow Chart**

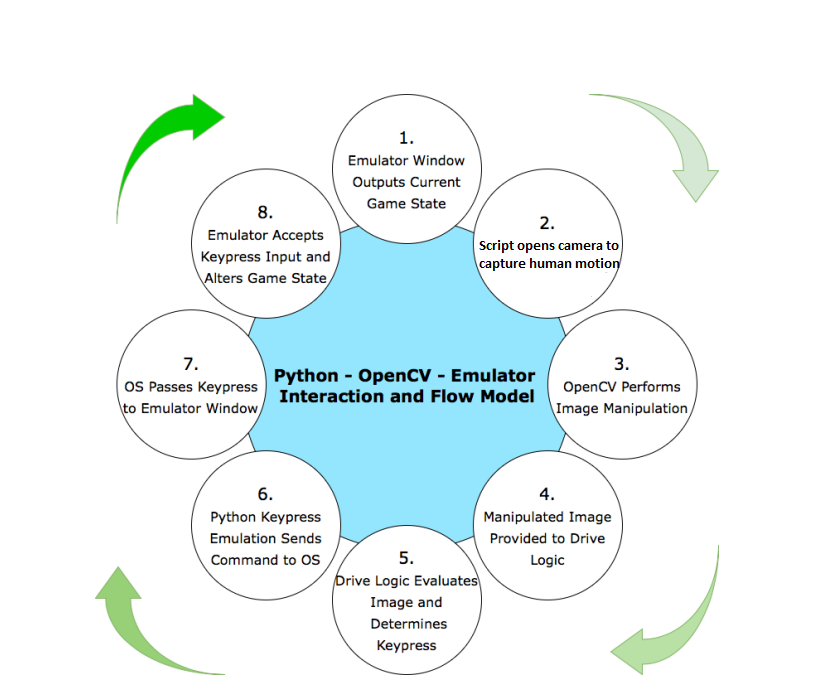
**4.3.1 Flow Chart of Project**

Figure 4.6 Flow Chart of Project

1. **Initialize Emulator**
   1. Initialize the emulator.
   2. Run the game from Game ROM.
2. **Executing the program**
   1. Opens camera to capture human motion
3. **Image Manipulation**
   1. Computer vision library OpenCV performs image Manipulation.
   2. It captures 20-30 frames per second.
4. **Track Skeleton**
   1. Skeleton frame ready method is used.
   2. As the user is continuously moving in front of camera, his movements are tracked.
5. **Pose Detection**
   1. Media pipe logic for processes the image for gesture detection.
6. **Gesture Detection**
   1. Once the gesture is detected the pyAutoGUI module comes in action.
7. **Send Key press Event**
   1. When a gesture is detected and threshold is matched, a particular gesture event is invoked.

**4.3.2 MediaPipe Pose**

In applications like quantifying physical exercises, sign language recognition, and full-body gesture control, human pose estimation from video is critical. It can be used in yoga, dance, and fitness applications, for example. In augmented reality, it can also enable the overlay of digital content and information on top of the physical world.

MediaPipe Pose is a machine learning solution for high-fidelity body pose tracking that uses our Blaze Pose research to infer 33 3D landmarks and a background segmentation mask on the whole body from RGB video frames. For inference, most current state-of-the-art approaches rely on powerful desktop environments, whereas our method achieves real-time performance on most modern mobile phones, desktops/laptops, in Python, and even on the web.

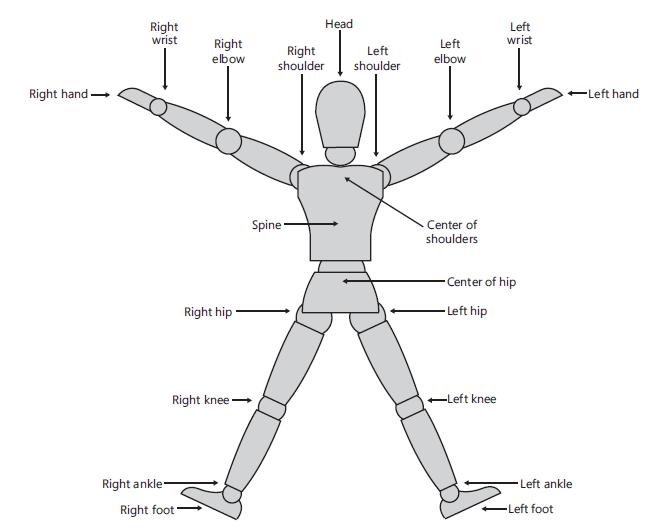


Figure 4.7 Name of each point in the human body skeleton

Each skeleton joint is measured in a three-dimensional (X, Y, Z) plane. The X and Y coordinates specify the location of the joint in the plane, and the player facing the Kinect sensor is in the Z direction.

When a joint is represented with X, Y, and Z coordinates in a three-dimensional plane, the X and Y coordinates actually indicate the joint location in the plane, and Z indicates how far the joint is from the sensor. If the joints move from the right-hand side to the left-hand side or vice versa, the X axis of the joint will change. Similarly, for moving joints in the upwards or downwards direction, the value of the Y axis will change. Changes in the Z axis will reflect if the joints move forward or backwards from the sensor. Calculations for the basic gestures can be done by either of the following:

* Calculating the distance between different joints
* Comparing the joints' positions and the deviation between the joints' positions

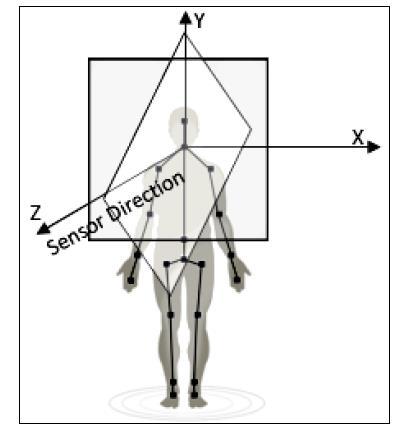


Figure 4.8 Representation of X, Y, Z co-ordinates in 3- dimensional

**4.3.3 Calculating the distance between two joints**

Skeleton data representation is three dimensional; however, before looking into the 3D coordinate plane, let's first consider the points in a 2D coordinate plane with only X

and Y axis and see how to calculate the distance between two points. In general mathematics, to calculate the distance between two points, we need to make use of the Pythagorean Theorem. The theorem states that: For a right-angled triangle, the square of the hypotenuse is equal to the sum of the squares of the other two sides*.* Refer to the following diagram, which shows how the Pythagorean theorem can be applied to calculate the distance between two joints:

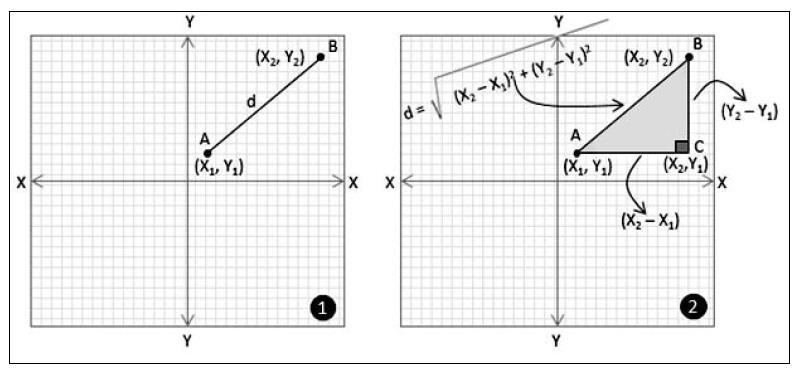
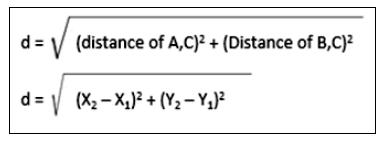


Figure 4.9 Calculating the distance between two points to join

Consider that you have a point A (X1, Y1) and a point B (X2,Y2) in a two-dimensional coordinate plane. You want to calculate the distance (let's call it "d") between point A and point B (refer to the image marked as 1). To calculate the distance using the Pythagoras theorem we have to first draw a parallel line to the X axis from point A and another line from point B, which is parallel to the Y axis. Consider both the lines meeting at point C (X2, Y1). As we know, the X and Y axes are perpendicular to each other; the triangle formed by the points A, B, and C is a right-angled one. Now, the value of "d", the distance between points A and B, will be the hypotenuse of the right-angled triangle that was formed by the points A, B, and C*.* The distance between A and B, now can be calculated using following formula:



**4.3.4 Pose Landmark Model**

The landmark model in MediaPipe Pose predicts the location of 33 pose landmarks

MediaPipe Pose can predict a two-class segmentation mask for a full-body segmentation mask (human or background).

Let's break down the code by first understanding the function's functionality in a nutshell: This function will detect the pose of a person who is closely associated with the image and draw all of the landmark points in the pose that is detected in the image.

* **image:** This argument will demand the image in which the person is there to be detected.
* **pose:** The pose function which we created for images and video one of those will be required according to the requirements
* **draw:** This argument will have the Boolean value when the value is True which means the function should draw the landmarks points otherwise not.
* **display:** This argument will also expect the Boolean value and if it has the True value then it will show both the input image and the resultant image otherwise not.



Figure 4.10 Pose Landmark Model

**Output:**

* **output image:** It will return the input image with the landmarks points which were detected in the person’s pose.
* **results**: It shows the output on the input image i.e., it is responsible for the visibility of the output on the image.

Virtual Gaming Through Human Gesture Detection Using Kinect Technical Specification

**Chapter No 5. Technical Specification**

****

**5.1 Advantages**

Following are the advantages of our system:

* Optimization of hardware:
* Reduce hardware cost
* User friendly
* Portable
* Elimination of high-priced consoles for playing motion sensed games.
* Cheaper than current existing systems.
* Can be used for system control also.
* Affordable for normal users.

**5.2 Disadvantages**

Following are the disadvantages of our system:

* It can only track up to 1 player, so no 2 or more players can actively participate in a game.

Following are the disadvantages of standard camera:

* Application can only detect pose only if there is good lighting.
* Minimum length of 1m should be maintained between the player and the camera.

Virtual Gaming Through Human Gesture Detection Using Kinect Technical Specification

**5.3 Applications**

Following are the applications of our system:

* To play motion sensed games on a regular PC/Laptop by using only standard camera.
* Physical handicapped person can system control through gestures.
* Systems can be controlled remotely using our system.

|  |  |
| --- | --- |
| N.D.M.V.Ps KBTCOE Nashik | 28 |

Virtual Gaming Through Human Gesture Detection Using Kinect Software Implementation

**Chapter No. 7 Coding Implementation**

****

**7.1 Importing the Required Libraries**

Alongside with the idea of creating the application, the need to design some basic 3D graphics regarding the environment, was born. Given the fact that XNA provides an automated set of tools to start with, the implementation of the graphics could be quite easy and as for the basic visualization of a 3D component-model, only two things were necessary; a camera and a 3D model.

**7.1.1 Camera**

The camera will be a simple class, used as the perspective point of view for the models of the application, including a location. It could be imagined as an eye placed within a specific location, having a particular direction of view (look at) and also has a predefined distance of its near and far plane (how far the eye can see and which is the minimum distance that is available from the eye to vision). It is also essential that during the development, a single instance of the camera should be passed across all the models. Otherwise each model will have a different perspective view and the application will end up to be visually messed up.

**7.2 Kinect Integration**

Following this chapter, an in-depth analysis will be made considering the integration of the Kinect into the game, given the fact that this is the major concept of this particular effort. The sensor comes with a well demonstrated SDK as already mentioned. This specific SDK incorporates a precise anatomic skeleton tracking system which also is the key factor to the given application. This skeleton tracking system is capable to recognize up to twenty different parts of the human body and also can track up to two different people. As soon as the system tracks a person, an event is raised providing skeleton some special data for the captured body. For each part of the body that has been tracked, a relative to space position (x, y, and z) is determined along with a unique id.

|  |  |
| --- | --- |
| N.D.M.V.Ps KBTCOE Nashik | 31 |

Virtual Gaming Through Human Gesture Detection Using Kinect Software Testing

**Chapter No. 8 Software testing**

****

**8.1 Introduction**

**What is testing?**

Testing is the process of evaluating a system or its component(s) with the intent to find that whether it satisfies the specified requirements or not. This activity results in the actual, expected and difference between their results. In simple words testing is executing a system in order to identify any gaps, errors or missing requirements in contrary to the actual desire or requirements.

Software testing is a process of verifying and validating that a software application or program

1. Meets the business and technical requirements that guided its design and development, and
2. Works as expected.

**Goals of Software Testing**

Before going for testing the Application, it is important to determine the goals or aim for which testing is intended to perform.

* The foremost and basic intention behind Software testing is to make sure that the application under test is free of bugs and errors and in case it is not, it should be identified and sent for rectification.
* Software testing plays a vital role to improve the quality of the product/software by reporting bugs at different phases of development depending on the methodology followed for the project.
* Software testing brings with it a source of reliability that is important for any business to get hold of the market.
* System’s Stability is one another checkpoint that Software Testing aims at since a stable application always gives rise to more customers and hence increases the revenue of the organization.

|  |  |
| --- | --- |
| N.D.M.V.Ps KBTCOE Nashik | 32 |

Virtual Gaming Through Human Gesture Detection Using Kinect Software Testing

* Software testing ensures whether the application is ready to deliver to the client or not, so it aims to find the critical bugs early in the testing phase.

**Major types of testing**

**White-Box testing**

**White-box testing** (also known as **clear box testing**, **glass box testing**, **and transparent box testing** and **structural testing**) tests internal structures or workings of a program, as opposed to the functionality exposed to the end-user. In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs.

Techniques used in our project’s white-box testing include:

* API testing (application programming interface) – testing of the application using public and private APIs.
* Testing GUI of
  1. Kinect SDK
  2. Kinect API
* Code Coverage– creating tests to satisfy some criteria of code coverage. All statements in the program to be executed at least once.

**Black-box testing**

**Black-box testing** treats the software as a "black box", examining functionality without any knowledge of internal implementation. The tester is only aware of what the software is supposed to do, not how it does it.

|  |  |
| --- | --- |
| N.D.M.V.Ps KBTCOE Nashik | 33 |

Virtual Gaming Through Human Gesture Detection Using Kinect Software Testing

**8.2 Test Cases**

In this section we are elaborating the testing methods implemented by us in this project.

* Unit Testing
* Integration Testing
* Functional Testing
* System Testing
* Stress Testing
* Performance Testing
* Usability Testing
* Acceptance Testing
* Regression Testing

**1. Unit Testing**

In unit testing we focus on individual components used in the project and make sure that these components work as expected. In this section we have done the following things:

1. Testing the Kinect hardware and Kinect SDK.
2. Checking whether the Kinect camera is working properly or not.
3. Checking whether correct values are passed via API for human tracking and detection.

**2.** **Integration Testing**

In this section we have combined the modules in the project and check whether we get desired output. We have also recorded the test values that the Kinect Camera returns when the human is closest to the camera

|  |  |  |
| --- | --- | --- |
| Position of Human | Closest | Farthest |
|  |  |  |
| Values returned | 1.20000 | 7.5000 |
|  |  |  |

Table No. 8.1 Values returned by Kinect camera

|  |  |
| --- | --- |
| N.D.M.V.Ps KBTCOE Nashik | 34 |

Virtual Gaming Through Human Gesture Detection Using Kinect Software Testing

**3. System Testing**

With system testing we aim to test our system in various environments, since our system is meant to run on Windows platform, we ran the tests on Windows OS only. According to our conclusions our system runs on following versions of Windows.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Operating | System | Test Result | |  |  | Requirements | |  |
| Version |  |  |  |  |  |  |  |  |
|  |  |  |  | |  |  |  |  |
| Windows 8 |  | System | works | | as | .NET | Framework | 4+, |
|  |  | intended |  |  |  | Kinect Drivers | |  |
|  |  |  |  | |  |  |  |  |
| Windows 7 |  | System | works | | as | .NET | Framework | 4+, |
|  |  | intended |  |  |  | Kinect Drivers | |  |
|  |  |  |  | | |  |  |  |
| Windows XP |  | System | experiences | | | - |  |  |
|  |  | time glitches | | as | XP |  |  |  |
|  |  | doesn’t | fully | support | |  |  |  |
|  |  | Kinect drivers. | |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

Table 8.2 System Testing for various operating systems

**4. Stress testing**

In this section we evaluate how system behaves under unfavourable conditions. Here we try outing different combinations of actions and interpreting our results.

Only 2 Humans can be detected by the Kinect at a time, if some extra human comes in range of the Kinect camera, the camera loses its initialization. We have planned to eliminate this in the next release.

**5. Usability Testing**

Usability testing is performed to the perspective of the client, to evaluate how the GUI is user-friendly; this is part of black-box testing. Here in this project, as a part of usability testing, we have found the GUI to be user friendly. We have asked for input from neutral

|  |  |
| --- | --- |
|  | 35 |

Virtual Gaming Through Human Gesture Detection Using Kinect Software Testing

persons related to the GUI and found that the GUI is self-explanatory and the user finds it easy to understand functionality of the project from GUI.

**6. Functional Testing**

Functional testing is the testing to ensure that the specified functionality required in the system requirements works. It falls under the class of black box testing.

* Check for different functions involved in the code as parsing, encrypting data, secret sharing, decrypting, and lastly reuniting the fragments in order to achieve the retrieval of original data back.

**7. Performance Testing**

Performance testing is the testing to assess the speed and effectiveness of the system and to make sure it is generating results within a specified time as in performance requirements. It falls under the class of black box testing.

* Providing several no. of inputs in order to check for its correctness, efficiency, and what much time it takes to produce the result i.e., retrieval of original data.

1. **Acceptance Testing**

Acceptance testing is often done by the customer to ensure that the delivered product meets the requirements and works as the customer expected. It falls under the class of black box testing.

**9.** **Regression Testing**

Regression testing is the testing after modification of a system, component, or a group of related units to ensure that the modification is working correctly and is not damaging or imposing other modules to produce unexpected results. It falls under the class of black box testing.

* After once the system is used, its effectiveness will be studied…and if there exist any modification to be done, to increase the system efficiency, in that case algorithms will be again studied.

|  |  |
| --- | --- |
| N.D.M.V.Ps KBTCOE Nashik | 36 |

Virtual Gaming Through Human Gesture Detection Using Kinect Software Testing

**8.3 snap shots of the test cases and Test plan**

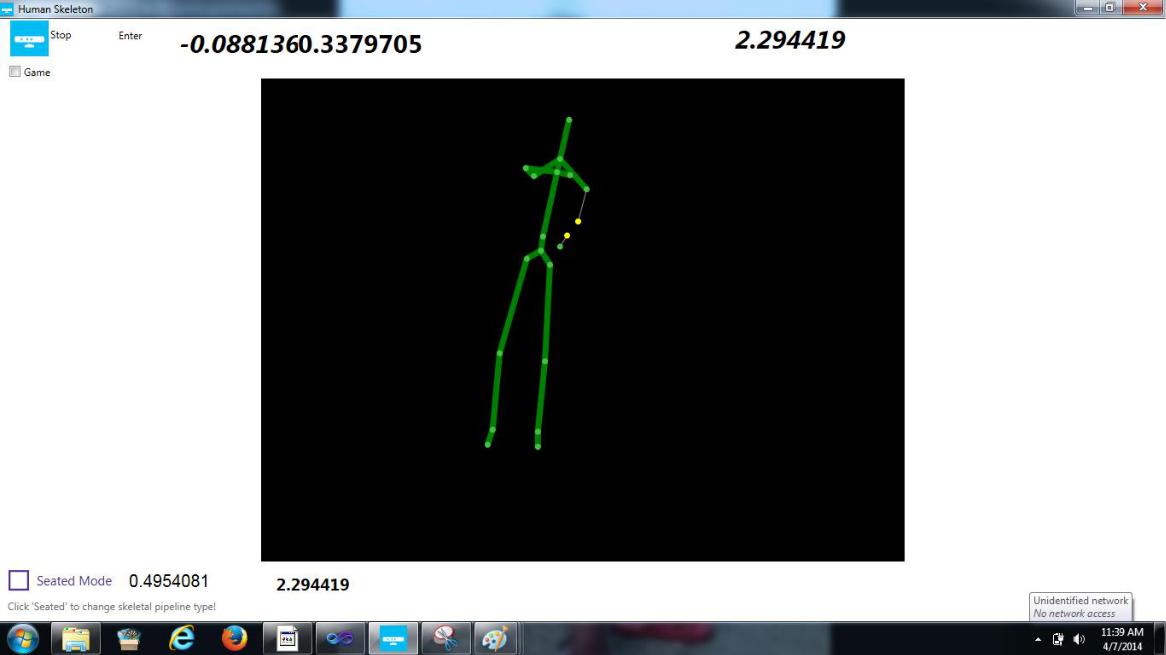
****

Figure 8.1 For computing the threshold value for setting the particular action

|  |  |
| --- | --- |
| N.D.M.V.Ps KBTCOE Nashik | 37 |

Virtual Gaming Through Human Gesture Detection Using Kinect Results

**Chapter No 9. Results (Snap shots of the results)**

****

**Let’s see what we have achieved in this project till now:**

1. At the moment there is no system developed for playing existing PC games via Kinect Camera on Windows platform, with our software we have achieved this. Hence, we are bringing motion sensed gaming on Windows platform via Kinect.
2. In our system we are able to play games which rely on following Keys: Up, Down, Left, Right, Space, Z, Shift, Enter

* Key Up

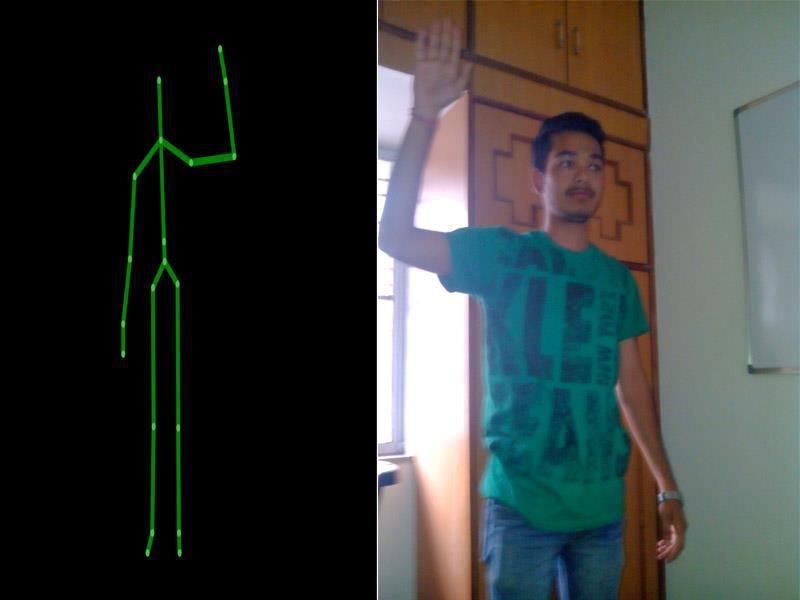


Figure 9.1 Key up action by hand gesture

|  |  |
| --- | --- |
| N.D.M.V.Ps KBTCOE Nashik | 38 |

Virtual Gaming Through Human Gesture Detection Using Kinect Results

* Key Left

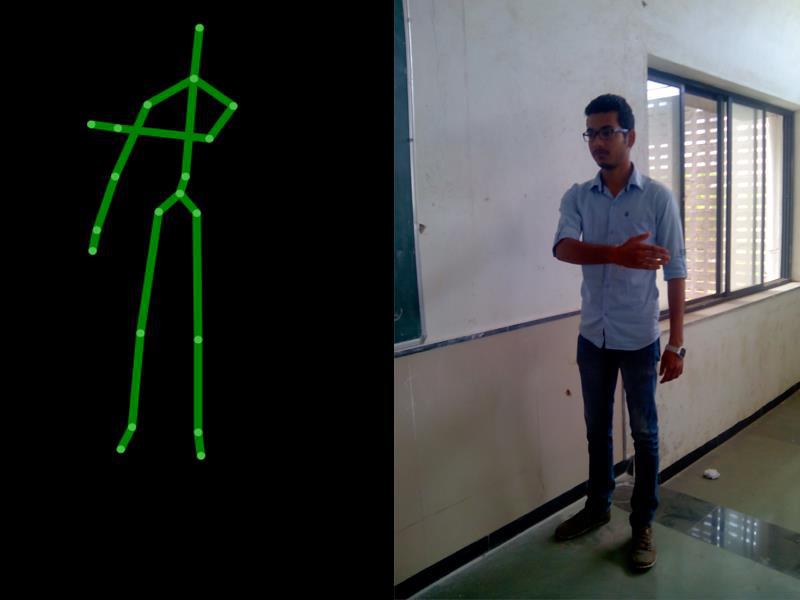


Figure 9.2 Key Left action by hand gesture

|  |  |
| --- | --- |
| N.D.M.V.Ps KBTCOE Nashik | 39 |

Virtual Gaming Through Human Gesture Detection Using Kinect Results

* Key Right

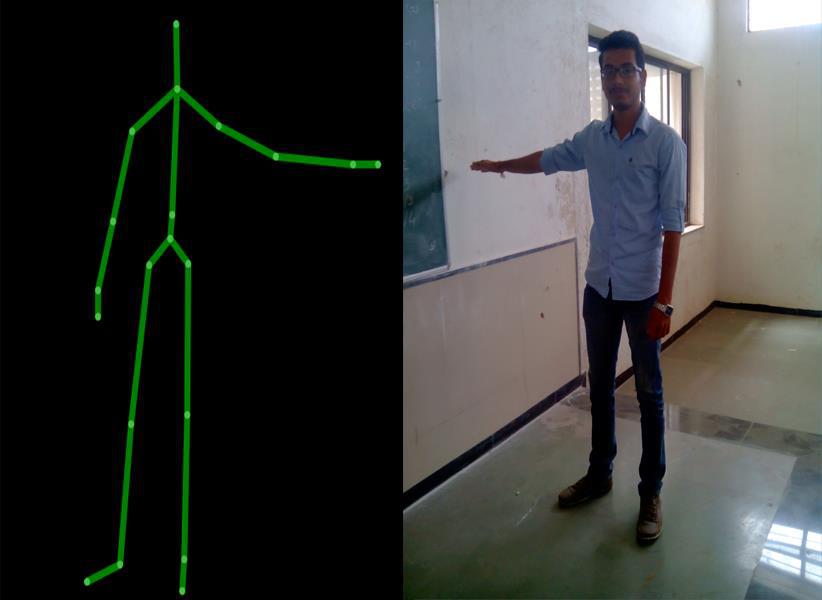


Figure 9.3 Key Right action by hand gesture

|  |  |
| --- | --- |
| N.D.M.V.Ps KBTCOE Nashik | 40 |

Virtual Gaming Through Human Gesture Detection Using Kinect Results

* Key kick

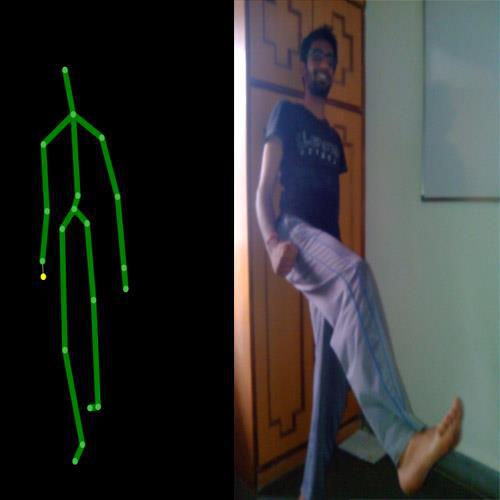


Figure 9.4 Key kick action by hand gesture

|  |  |
| --- | --- |
| N.D.M.V.Ps KBTCOE Nashik | 41 |

Virtual Gaming Through Human Gesture Detection Using Kinect Results

* Steer Left

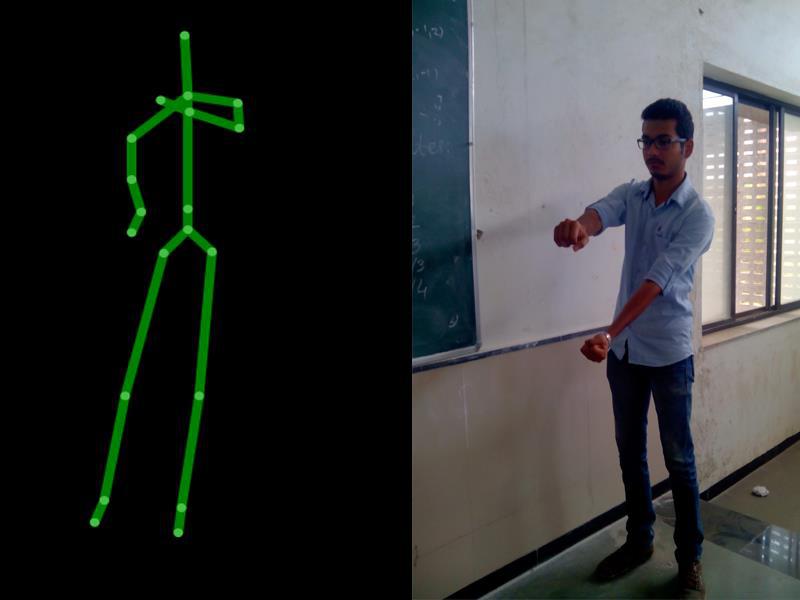


Figure 9.5 Steer Left action by hand gesture

1. We have successfully detected human gestures/actions and when these gestures are matched key press event is invoked.
2. On Commercial point of view we can release gesture sets for games along with documentation for users who have Kinect device and wish to enhance their current gaming experience on computers.

|  |  |
| --- | --- |
| N.D.M.V.Ps KBTCOE Nashik | 42 |

Virtual Gaming Through Human Gesture Detection Using Kinect Deployment & Maintenance

**Chapter No. 10 Deployment and Maintenance**

****

**10.1 Installation and un-installation**

**Kinect Developer toolkit**

That file is a setup executable file which will include all needed components including Kinect for Windows 1.8 Runtime and Drivers (The drivers are not updated for 1.8 in Device Manager), Microsoft Speech runtime components, C++ runtime components, and also includes the .NET 4.0 install executable which will download and install .NET 4.0 if neither .NET 4.0 nor .NET 4.5 is already present on the computer. Developers should include the Redistributable Code with their applications as part of a chain install with their own application setup.

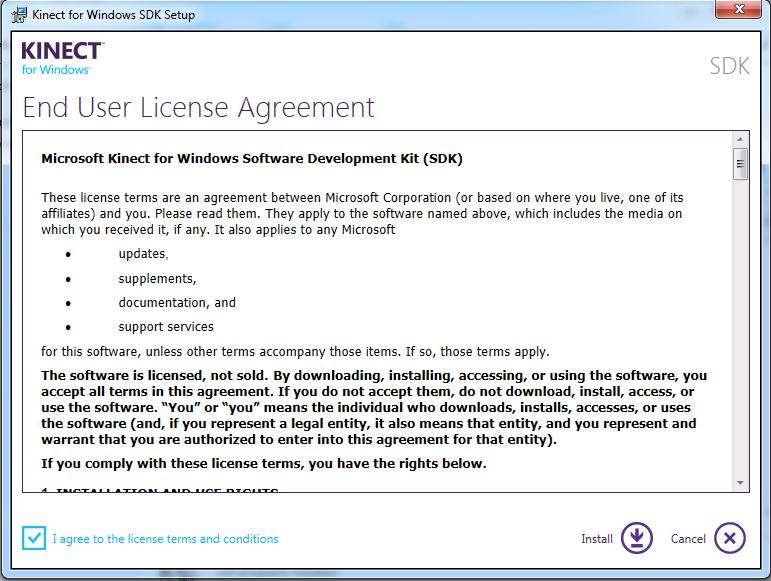


Figure 10.1 Installation of Microsoft for windows software development kit

|  |  |
| --- | --- |
| N.D.M.V.Ps KBTCOE Nashik | 43 |

Virtual Gaming Through Human Gesture Detection Using Kinect Deployment & Maintenance

**10.2 User Help**

**1) ADEQUATE SPACE FOR PLAYING**

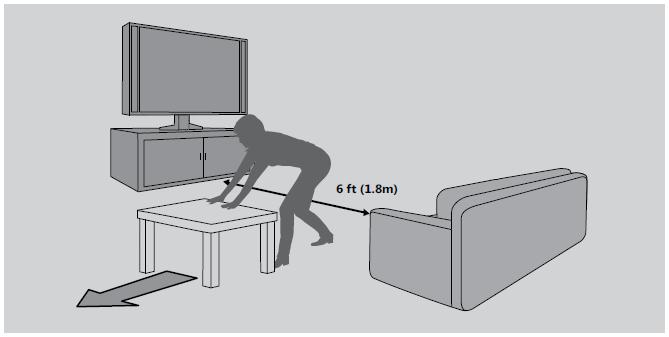
****

Figure 10.2 Free Space for playing the game

The Kinect sensor needs to be able to see you, and you need room to move. The sensor can see you when you play approximately 6 feet (2 meters) from the sensor Play space will vary based on your sensor placement and other factors

**Before playing:**

* Look in all directions (right, left, forward, backward, down, and up) for things you might hit or trip over.
* Make sure your play space is far enough away from windows, walls, stairs, etc.
* Make sure there is nothing you might trip on—toys, furniture, or loose rugs, for example. Also, be aware of children and pets in the area. If necessary, move objects or people out of the play space.
* Don’t forget to look up. Be aware of light fixtures, fans, and other objects overhead when assessing the play space.

**While playing:**

* Stay far enough away from the television to avoid contact.

|  |  |
| --- | --- |
| N.D.M.V.Ps KBTCOE Nashik | 44 |

Virtual Gaming Through Human Gesture Detection Using Kinect Deployment & Maintenance

* + Keep enough distance from other players, bystanders, and pets. This distance may vary between games, so take account of how you are playing when determining how far away you need to be.
  + Stay alert for objects or people you might hit or trip on. People and objects can move into the area during game play, so always be alert to your surroundings.
  + Make sure you always have good footing while playing:
  + Play on a level floor with enough traction for game activities.
  + Make sure you have appropriate footwear for gaming (no high heels, flip flops, etc.) or are barefoot, if appropriate.

1. **Set Up Your Sensor**

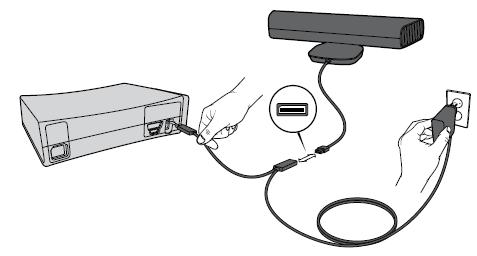
****

Figure 10.3 Hardware connection with Kinect camera

**To connect to your original Xbox 360 console:**

* Unplug any accessories from the back USB port on your console.
* Plug the sensor into the USB/power cable.
* Plug the USB/power cable into your console’s back USB port.
* Plug the AC adapter end of the USB/ power cable into a wall outlet. Use only the USB/power cable that is shipped with the product or is given to you by an authorized repair center. If you have an original Xbox 360 console with no hard drive, you should also attach a storage device with at least 256 MB free space. You can use an Xbox 360 Hard Drive, Xbox 360 Memory Unit, or a USB flash drive. If you’re using an Xbox 360 Wireless Networking Adapter that’s already connected

|  |  |
| --- | --- |
| N.D.M.V.Ps KBTCOE Nashik | 45 |

Virtual Gaming Through Human Gesture Detection Using Kinect Deployment & Maintenance

to the back USB port, you’ll need to disconnect its USB cable and reconnect it to a front USB port using the WiFi extension cable, provided.

**3) Electrical Safety**

As with many other electrical devices, failure to take the following precautions can result in serious injury or death from electric shock or fire or damage to the sensor. If you use AC power, select an appropriate power source:

* + The sensor’s power input is 12V DC @ 1.1A. Use only the AC adapter on the

USB/power cable that came with your sensor or that you received from an authorized repair center.

* + Confirm that your electrical outlet provides the type of power indicated on the

USB/power cable, in terms of voltage (V) and frequency (Hz). If you aren’t sure of the type of power supplied to your home, consult a qualified electrician.

* + Do not use non-standard power sources, such as generators or inverters, even if the voltage and frequency appear acceptable. Use only AC power provided by a standard wall outlet.

1. **Install the Sensor Software on Your Console**

Your console needs a system update before you can use it with your Kinect sensor.

**To update your console:**

* Turn on your console and insert the supplied disc. The update will install automatically. If it doesn’t start automatically, select the disc tray from the dashboard (as if you were playing a game from a disc).
* When the installation confirmation message appears, remove the disc and begin setting up your sensor.

|  |  |
| --- | --- |
| N.D.M.V.Ps KBTCOE Nashik | 46 |

Virtual Gaming Through Human Gesture Detection Using Kinect Conclusion and Future Scope

**Chapter No. 11 Conclusion and Future Scope**

****

**Conclusion**

Hence in this project we are eliminating high end console systems and replacing it with our regular PC/Laptop’s integrated with Kinect also we are controlling system without actually touching keyboard or mouse.

**Future Scope**

The major improvement that needs to be done in the project is increasing the frame--‐processing rate. Though the number of pyramid levels increase, the processing should be improved and the detection should happen in as low as 0.4 frame/second. This will be achieved by implementing some calculations in GPU. Further, attention needs to be mode. Efforts will be taken to implement the retained mode. Also the consistency of the human detection has to be improved, especially in far plane environment. Performing as many numbers of tests as possible in different locations and thereby achieving the relationship between the detection and the threshold values, the consistency can be improved. Focused in the OpenGL drawing routine because the frames are drawn in the immediate

Finally some sound effects could be included or even some background music to make the game more interest.

|  |  |
| --- | --- |
| N.D.M.V.Ps KBTCOE Nashik | 47 |

Virtual Gaming Through Human Gesture Detection Using Kinect Project Assignment

**Chapter No. 12References**

****

**12.1 References**

1. [Chia-Chih Chen](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=p_Authors:.QT.Chia-Chih%20Chen.QT.&searchWithin=p_Author_Ids:37656952900&newsearch=true) ; [Aggarwal, J.K..](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=p_Authors:.QT.Aggarwal,%20J.K..QT.&searchWithin=p_Author_Ids:37267346400&newsearch=true): Human Detection Using Depth Information by Kinect. CVPR, 1 (2011) 886-893.
2. T. Darrell, G. Gordon, J. Woodfill, and M. Harville, "Integrated Person Tracking using Stereo, Color, and Pattern Detection," Proceedings IEEE Conference on Computer Vision and Pattern Recognition, Santa Barbara
3. [http://www.KinectProgrammingGuide.com](http://www.kinectprogrammingguide.com/)/human\_trackingusing\_hardware/
4. http://www.codeplex.com/kinect/gesture\_detection/
5. N. Dalal and B. Trigges.: Histograms of oriented gradients for human detection. CVPR, 1 (2005) 886-893.
6. N. Dalal, B. Triggs, C. Schmid.: Human detection using oriented histograms of flow and appearance, in: European Conference on Computer Vision, Graz, Austria, May 7–13, 2006
7. T. Darrell, G. Gordon, J. Woodfill, and M. Harville, "Integrated Person Tracking using Stereo, Color, and Pattern Detection," Proceedings IEEE Conference on Computer Vision and Pattern Recognition, Santa Barbara, June 1998
8. D. Demirdjian, T. Darrell: 3-D Articulated Pose Tracking for Untethered Diectic Reference. ICMI 2002: 267-272
9. V. Ganapathi, C. Plagemann, D. Koller, S. Thrun. Real time motion capture using a single time-of-flight camera. Proceedings of CVPR 2010. pp.755~762
10. S. Ikemura, H. Fujiyoshi.: Real-Time Human Detection using Relational Depth Similarity Features. ACCV 2010, Lecture Notes in Computer Science, 2011, Volume 6495/2011, 25-38

|  |  |
| --- | --- |
| N.D.M.V.Ps KBTCOE Nashik | 48 |

Virtual Gaming Through Human Gesture Detection Using Kinect Project Assignment

1. HP. Jain and A. Subramanian. Real-time upper-body human pose estimation using a depth camera. In HP Technical Reports, HPL-2010-190, 2010
2. K. Levi and Y. Weiss. Learning object detection from a small number of examples: the importance of good features. CVPR 2(2004) 53-60
3. DG. Lowe.: Object Recognition from Local Scale-Invariant Features. Proceedings of the International Conference on Computer Vision. 2 (1999). pp.?1150–1157

**12.2 Glossary**

1. **Kinect:** A 3D Camera developed By Microsoft for depth tracking.
2. **Stereo Camera:** It is time of camera with 2 or more lenses with separate image sensor for each lens, hence it gets ability to capture 3D image.
3. **TOF camera:** Time of flight camera is range imaging camera system that determines time of flight of light signal from camera to the object.
4. **2D Intensity Images**: Image is represented using intensity of RGB colours in a 2D plane.
5. **ER Diagram:** Entity-relationship diagrams don't show single entities or single instances of relations. Rather, they show entity sets and relationship sets.
6. **UML Diagrams**: UML stands for Unified Modeling Language. There are different types of UML diagrams. Each UML diagram represents the system to the customer and the developer from a different perspective and different levels of abstractions.
7. **Database Administrator**: A DBA can be defined as a person who access to the database of an organization and has the authority to update the database.
8. **Functional Dependencies:** A functional dependency is defined as a relationship of attributes in a relational database. Thus we can say that FD is nothing but a relationship which shows the dependency of particular attribute on specific attribute or a group of attributes.

|  |  |
| --- | --- |
| N.D.M.V.Ps KBTCOE Nashik | 49 |

Virtual Gaming Through Human Gesture Detection Using Kinect Project Assignment

1. **Dataflow Diagrams:** They are the diagrams which show the actual flow of data in a system. There are different levels of DFDs. The starting level indicates the most basic representation of data flow. As the level of DFDs increase, it gives more and more detail view of the system.
2. **System Architecture:** It is a gist of the entire system. It consists of the representation of the components of the system and how these components are interconnected.
3. **Use Case Diagram:** It represents the use case view of a system. A use case is nothing but a functionality of a system.
4. **Sequence Diagram:** It deals with the sequence of messages flowing from one object to another.
5. **Collaboration Diagram:** It represents the structural organization of a system.
6. **State Chart Diagram:** It is used to represent the event driven state change of the system.
7. **Activity Diagram:** It describes the flow control in a system.
8. **Package Diagram:** It shows how model elements are organized into packages and also the dependencies among them.
9. **Deployment Diagram:** It is a set of nodes and their relationship.

**12.3 Acronyms:**

1. **DB:** Database.
2. **UML:** Unified Modelling Language.
3. **ERD:** Entity Relationship Diagram.
4. **FD:** Functional Dependency.
5. **OS** – Operating System.