VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JNANA SANGAMA", BELAGAVI, KARNATAKA-590018

A Project Dissertation on

"SQUID GAME AND MUSIC SYNCHRONIZATION"

Submitted in partial fulfillment for the award of degree of Bachelor of Engineering in Computer Science & Engineering during the year 2022-2023

By

BHAVANA V R	**4MN19CS012**
KRUTHIK GOWDA H R	**4MN19CS027**
RAMYA B L	**4MN19CS042**
SHAMA R	**4MN19CS050**

Under the guidance of

Prof. Suhasini
Assistant Professor
Dept of CS&E,
MIT Thandavapura

2022-2023

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

MAHARAJA INSTITUTE OF TECHNOLOGY, THANDAVAPURA

JUST OFF NH 766, NANJANAGUDU TALUK, MYSORE DISTRICT - 571302

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

MAHARAJA INSTITUTE OF TECHNOLOGY THANDAVAPURA

MYSORE-571302

CERTIFICATE

This is to certified that the project work titled "**Squid Game and Music Synchronization"** has been successfully carried out by **BHAVANA VR [4MN19CS012], KRUTHIK GOWDA HR [4MN19CS027], RAMYA BL [4MN19CS042], SHAMA R [4MN19CS050]** bonafide students of **Maharaja Institute of Technology Thandavapura in partial fulfilment of requirements of Degree of Bachelor of Engineering in Computer Science & Engineering of Visvesvaraya Technological University, Belgaum** during the academic year 2022-23. It is certified that all corrections/suggestions indicated for the internal assessment have been incorporated in the report deposited in the Department library. The project report has been approved has it satisfies the academic requirements with respect to the project work prescribed for Bachelor of Engineering Degree.

Signature of guide	Signature of the HOD	Signature of the Principal
Prof. Suhasini	**Dr. Ranjit K N**	**Dr. Y. T. Krishne Gowda**
Assistant Professor	Associate Professor	Principal
Dept. of CS&E,	HoD of CS&E	MIT Thandavapura
MIT Thandavapura	MIT Thandavapura	

External Viva

Name of the Examiners **Signature with date**

1. _____

2. _____

ACKNOWLEDGEMENT

It gives me immense pleasure to bring my Project report entitled "**SQUID GAME AND MUSIC SYNCHRONIZATION**" in the Eighth Semester Engineering Course.

We are very thankful to **Dr. Y T Krishne Gowda, Principal, MITT** for having supported us in our academic endeavours.

We would like to extend our thanks to **Dr. H K Chethan, Professor and Mentor, CS&E, MIT, Thandavapura, Dr. Ranjit K N, Associate Professor and Head, Department of CS&E,** for providing us timely suggestions, encouragement and support to complete this work.

We would like to sincerely thank Our project guide **Suhasini, Assistant Prof, Dept. of Computer Science and Engineering** for providing relevant information, valuable guidance and encouragement to complete this report.

We would also like to thank all our teaching and non-teaching staff members of the Department for their support. We are always thankful to our parents for their valuable support and guidance in every step.

We express Our deepest gratitude and indebted thanks to MITT which has provided me an opportunity in fulfilling our most cherished desire of reaching the goal.

Bhavana V R **4MN19CS012**

Kruthik Gowda H R **4MN19CS027**

Ramya B L **4MN19CS042**

Shama R **4MN19CS050**

ABSTRACT

According to a recent info-trends study, mobile and camera device users will have taken more than 1.5 trillion images, a sharp increase from the data from 2016. These image data will be used in a variety of real-time applications, including visual video surveillance, object identification, object detection, and classification. Advanced computer vision algorithms that were an upgrade over traditional computer vision techniques were created to manage these enormous volumes of data automatically. One of the most crucial tasks is object detection, which can greatly enhance the functionality of a variety of computer vision-based applications, including object tracking, license plate detection, mask and social distance detection etc., deep learning neural networks is a powerful programming paradigm which learns multiple levels of representation and abstraction of data such as images, sound, and text. In this project we intend to create a recently trending movie “squid game” in which a game was played on movement of a player at a particular time based on the music played.

LIST OF CONTENTS

LIST OF FIGURES

TABLE OF CONTENTS

CHAPTER - 1

INTRODUCTION

1.1 Introduction

Saliency and scalable object detection are two recent examples of object detection methods based on computer vision systems. The conventional object detection procedure can be divided into region detection, feature extraction, and classification. These systems have a number of problems with variations in pose, changing lighting settings, and increased complexity of localization of an object within the given image.

In the world of computer vision, the world is displayed in 3D, but the input to humans and computers is two-dimensional. In addition, computer systems can only process binary bits of data, but some useful programs only work in two dimensions. To take full advantage of computer vision, you need 3D information, not just a collection of 2D views. These limited opportunities, 3D information is portrayed directly and is clearly not as good as people use. However, designing a robust, feature-rich extract for all types of objects is considered a tedious task due to the different number of lighting variations. To perform visual recognition, you need a classification model and library to further distinguish the categories of objects. Of the object.

The application of these advanced computer vision techniques along with various machine learning algorithms like NumPy, TensorFlow which provide support for advanced mathematical calculations along with vast repositories of libraries and packages available in python like Tkinter for front- end design and development can solve various real-world problems, these coupled with deep learning algorithms and accurate object detection using Yolo and various iterations for neural networks prove to be a boon in improving the accuracy of object detection and tracking and application of these into the real world.

In context to the current issue concerning expensive games with augmented reality and virtual reality because of their costly sensors and high-end processing power required for their application also not convenient for small-scale manufacturers to develop without sufficient research and development, that’s when the application of advanced computer vision comes into picture which has proven to be useful in various application like yoga pose estimation and various complex exercises like posture management thus success in this field provides a view to approaching other aspects of applicability.

An approach to gaming implementation of a television series which proved to be a motivation for this implementation of a game based solely just on the camera sensor and

advanced computer vision technologies like tensor flow for artificial intelligence and NumPy for calculations and Tkinter for front-end development, we intend to replicate the real world game played in the movie scene into a gaming scenario using the cheapest of resources as possible and proving that immersive games can also be cheap and not needing any complex consoles media-pipe is a python package built by google for building machine learning pipeline models for processing time-series of data like videos, audio ,etc. Its cross-platform Framework works in Desktops, Servers and android as well as iOS use case like Raspberry Pi media-pipe is powered by revolution product and services that we use daily. Unlike power-hungry machine learning Frameworks, media-pipe requires minimal resources.

Media-pipe opened up a whole new world of opportunity for researchers and developers following public release. media-pipe Toolkit comprises the Framework and the Solutions, The diagram shows how media pipe is organized with its features, multi-threading is a way of multitasking using threads, as the modern processors have the capability to run multiple threads at a time, its highly beneficial to run different tasks on different threads of a processor to reduce the delay and improve performance of human pose estimation from video plays a critical role in various applications such as quantifying physical exercises, sign language recognition, and full-body gesture control. For example, it can form the basis for yoga, dance, and fitness applications. It can also enable the overlay of digital content and information on top of the physical world in augmented reality.

Media-pipe Pose is a ML solution for high-fidelity body pose tracking, inferring 33 3D landmarks and background segmentation mask on the whole body from RGB video frames utilizing our Blaze Pose research that also powers the ML Kit Pose Detection API. Current state-of-the-art approaches rely primarily on powerful desktop environments for inference, whereas our method achieves real-time performance on most modern mobile phones, desktops/laptops, in python and even on the web.

1.2 Overview with Problem Identification

Many researchers have contributed innovative algorithms and approaches in the area of human action recognition system and have conducted experiments on individual data sets by considering accuracy and computation. In spite of their efforts, this field requires high accuracy with less computational complexity. The existing techniques are inadequate in accuracy due to assumptions regarding clothing style, view angle and environment. Hence, the main objective of this thesis is to develop an efficient multi-view based human action recognition system using shape features.

1. Current Augmented Reality and Virtual Reality games are too expensive as the need of costly sensors to recognize movements is very high.
2. Training of players, athletes and dancers' posture and response time currently needs an experienced human supervision.
3. Motion detection using cameras is not much used in game development due to its accuracy Issues also as it hinders profit margin of console developers.

1.3 Objective

To build a system that detects and isolates the movement of human subject and simultaneously comparing it with synchronization of music to determine state of the game.

1.4 Scope

The project is designed to detect the age of the person using the images captured andpredicts the pose and also the accuracy is done. The project sets out to use various techniques for a human pose estimation, analyzing the impacts of different techniques used with the music synchronization.

Scope Of Study

- To design and train a system which can isolate and segment the human object from the background noise.
- To check for movement detection by background subtraction of pixel instances of 2 frames.
- Check the music playing status and movement detection in real time.
- Check and disqualify of the person from the game if found to be moved.

1.5 Existing System

Many types of immersive games, many with respect to virtual reality and recently augmented reality applications has increased in the basic format, but they are quite expensive as they're need arises for expensive sensors and customized hardware(consoles) and software are needed. Previous use cases of computer vision had very complex process of object detecting and tracking which proved to be slower and more process intensive with so many processes needed for detection and its application and unable to compete with their console's counterpart.

1.6 Proposed System

Recent advancements in advanced computer vision and popularity of python as the go to programming language for artificial intelligence and machine learning applications led to many developers getting involved in making custom libraries and packages, also improvement in tensor flow allowed faster insight discovery from data. Due to this aspect, and the lack of intelligence in current generation gaming consoles meant that we could try to develop immersive games with just basic sensors and cameras. Furthermore, as success and acceptability were seen in AR in the use cases of posture and yoga training as well as trying on of glasses and visualizing a piece of furniture in the living environment. Thus, as a first approach we intend to replicate implementation of a real-world game (squid game-red light green light) with just the camera backed by artificial intelligence.

Advantages

- Real-time data extraction of the images done by users and estimating the postures and accuracy.
- Users will be motivated to use our application.

1.7 Applications

- Cost effective augmented reality/virtual reality games.
- Creating immersive gaming experience without using huge number of sensors and motion tracking.
- Advancement of opencv software using artificial intelligence and machine learning.

CHAPTER - 2

LITERATURE SURVEY

Author name & Year	Title	Methodology	Result
Swetha U Krishnan **Year-2022**	An analysis of the visual narrative of Squid Games in the South Korean context	Mild background tone and dim lighting, flute music playing	End player who completes the goal or distance will win the game
Lamiyah Khattar **Year-2021**	Analysis of Human Activity Recognition using Deep Learning	2-D CNN model, Manifold Learning	Acceleration change with the amplitude being just above +10.0 in both the cases
Murari Mandal **Year - 2020**	A Unified Deep Framework for Moving Object Recognition	CNN algorithm, temporal depth (TDR), Regression algorithm (linear and logistic regression)	The Motion is able to distinguish between static and moving objects in all scenarios
Ajmal Shahbaz **Year - 2019**	Convolutional Neural Network based Foreground Segmentation for Video Surveillance Systems	Two Algorithm - feature extractor part which is CNN and the sampling of the extracted feature using bilinear interpolation	Sigmoid function is employed to squash the probabilities into background and foreground
Dong Hye Ye **Year - 2018**	Deep Learning for Moving Object Detection and Tracking from a Single Camera in	Deep learning methods (CNN), Object detection and tracking algorithm using CNN	Improve detection accuracy and help in the detection of moving objects

	Unmanned Aerial Vehicles (UAVs)		
Mohammadreza Babaee **Year-2017**	A deep convolutional neural network for video sequence background subtraction	A novel algorithm for background model (image) generation; a novel CNN for background subtraction and post-processing of the networks output using median filter	Vectorized feature maps of the last convolutional layer in our CNN are fed into a MLP perform pixel wise classification
First A. Haidi Zhu **Year-2017**	Moving object detection with Deep CNNs	Moving object detection with Deep CNNs	Merge segmented regions
Satrughan Kumar **Year-2016**	Segmentation of Moving Objects using Background Subtraction Method in Complex Environments	Sharpening and Smoothing techniques Deep learning, GMM method	Extract the moving object and local motion
Jaima Gallego **Year-2014**	Foreground object segmentation for moving camera sequences based on foreground	Foreground object segmentation system on probabilistic models	Achieves the best scores in sequences where rigid objects

Table 2.1: Literature Survey

CHAPTER - 3

SYSTEM REQUIREMENT SPECIFICATIONS

3.1 Functional Requirements

Functional requirement defines a function of a software system or its component. A function is described as a set of inputs, behavior, and outputs. Functional requirements may be technical details, data manipulation, encryption and decryption of data, processing, and other specific functionality that define what a system is supposed to accomplish.

The various libraries used in the project are:

1. OpenCV
2. Tkinter
3. Media pipe
4. Numpy
5. Play sound

Open CV:

(Open-Source Computer Vision Library) is an open-source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc.

Tkinter:

The Tkinter package (“Tk interface”) is the standard Python interface to the Tcl/Tk GUI toolkit. Both Tk and Tkinter are available on most Unix platforms, including macOS, as well as on Windows systems. Running python -m tkinter from the command line should open a window demonstrating a simple Tk interface, letting you know that tkinter is properly installed on your system, and also showing what version of Tcl/Tk is installed, so you can read the Tcl/Tk documentation specific to that version.

Tkinter supports a range of Tcl/Tk versions, built either with or without thread support. The official Python binary release bundles Tcl/Tk 8.6 threaded. See the source code for the Tkinter module for more information about supported versions.

Media pipe in Pose Estimation:

Human pose estimation from video plays a critical role in various applications such as quantifying physical exercises, sign language recognition, and full-body gesture control. For example, it can form the basis for yoga, dance, and fitness applications. It can also enable the overlay of digital content and information on top of the physical world in augmented reality.

Media-pipe Pose is a ML solution for high-fidelity body pose tracking, inferring 33 3D landmarks and background segmentation mask on the whole body from RGB video frames utilizing our Blaze Pose research that also powers the ML Kit Pose Detection API. Current state-of-the-art approaches rely primarily on powerful desktop environments for inference, whereas our method achieves real-time performance on most modern mobile phones, desktops/laptops, in python and even on the web.

The solution utilizes a two-step detector-tracker ML pipeline, proven to be effective in our Media Pipe Hands and Media Pipe Face Mesh solutions. Using a detector, the pipeline first locates the person/pose region-of-interest (ROI) within the frame. The tracker subsequently predicts the pose landmarks and segmentation mask within the ROI using the ROI-cropped frame as input. Note that for video use cases the detector is invoked only as needed, i.e., for the very first frame and when the tracker could no longer identify body pose presence in the previous frame. For other frames the pipeline simply derives the ROI from the previous frame’s pose landmarks.

The pipeline is implemented as a media-pipe graph that uses a pose landmark subgraph from the pose landmark module and renders using a dedicated pose renderer

subgraph. The pose landmark subgraph internally uses a pose detection subgraph from the pose detection module.

Note: To visualize a graph, copy the graph and paste it into Media Pipe Visualizer. For more information on how to visualize its associated subgraphs, please see visualizer documentation.

NumPy:

Which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed. NumPy is a Python package. It stands for 'Numerical Python'. It is a library consisting of multidimensional array objects and a collection of routines for processing of array.

Play sound:

Module contains only a single function names playsound (), it requires only one argument the path to a file, it may be a local file or a link to a repository, it works with av or mp3 files, an optional second argument which is true by default can make it run asynchronously.

3.2 Non-Functional Requirements:

Non-Functional Requirements describe the aspect of the system that is not directly related toits functional behavior. Non-functional requirements define system properties and constraintsit arises through user needs, because of budget constraints or organizational policies, or due to external factors such as safety regulations, privacy registration and so on.

The different Non-functional requirements for our project are:

Easy to Operate

The system should be easy to operate and should be such that it can be developed within a short period and fit in the limited budget of the user.

Performance Requirements

Since the software is online, therefore much of the performance of the system depends on the **t**raffic that is present online and the speed of the Internet. We are trying to give an improved performance by setting cookies to the functions so that when the user submits something for the second time, the processing is done much quicker.

Usability Requirements

The Navigation for the various operations is arranged in an orderly fashion based on the requirements. The interface also must provide a soothing look to the eye of the user.

Portability Requirements

The system should be portable and should be able to switch any environment changes such asa change of database within a very short period.

Easy to Operate The system should be easy to operate and should be such that it can be developed within a short period and fit in the limited budget of the user.

- Secure access to confidential data.
- 24 X 7 availability
- The flexible service-based architecture will be highly desirable for future extension

3.3 Hardware Requirements

- System: Intel core i5
- RAM: 4 GB
- Camera: 8 MP
- Hard disk: 1TB memory space

3.4 Software Requirements

- Operating system: windows 10
- Coding Language: Python

3.5 Requirement Traceability Matrix

Serial No	Requirement ID	Requirement Brief	Requirement Description
1	RID-1	Validation for a camera check	To verify the start of the camera
2	RID-2	Validation for image capture	To verify that image is captured or not
3	RID-3	Validation for pose detection	No pose should be detected when user is not in frame
4	RID-4	Validation for Image	Identify movement of the human
5	RID-5	Validation of Music	To play music when the human is detected
6	RID-6	Embedding extraction	Identify key points extracting its embedding
7	RID-7	Pushing into a pretrained model	All structured embedding data will be pushed into a pretrained model
8	RID-8	Movement Detection	Identifying user is eliminated or winner

Table 3.1: Requirement Traceability Matrix

CHAPTER - 4

SYSTEM ANALYSIS AND DESIGN

4.1 System Analysis

The system design process builds up general framework building design. The programmingoutline includes speaking to the product framework works in a shape that may be changed into one or more projects. The prerequisite indicated by the end client must be put systematically. An outline is an inventive procedure; a great configuration is a way to the viable framework. The framework "Outline" is characterized as "The procedure of applyingdifferent systems and standards with the end goal of characterizing a procedure or a framework inadequate point of interest to allow its physical acknowledgment". Different configuration components are taken after to add to the framework. The configuration detail portrays the components of the framework, the segments or components of the framework, and their appearance to end clients.

4.2 System Architecture

The architectural configuration procedure is concerned with building up a fundamental basicsystem for a framework. It includes recognizing the real parts of the framework and the interchange between these segments. The beginning configuration procedure of recognizingthese subsystems and building up a structure for subsystem control and correspondence is called construction modeling outline and the yield of this outlined procedure is a portrayal of the product structural planning. The proposed architecture for this system is given below.It shows the way this system is designed and the brief working of the system.

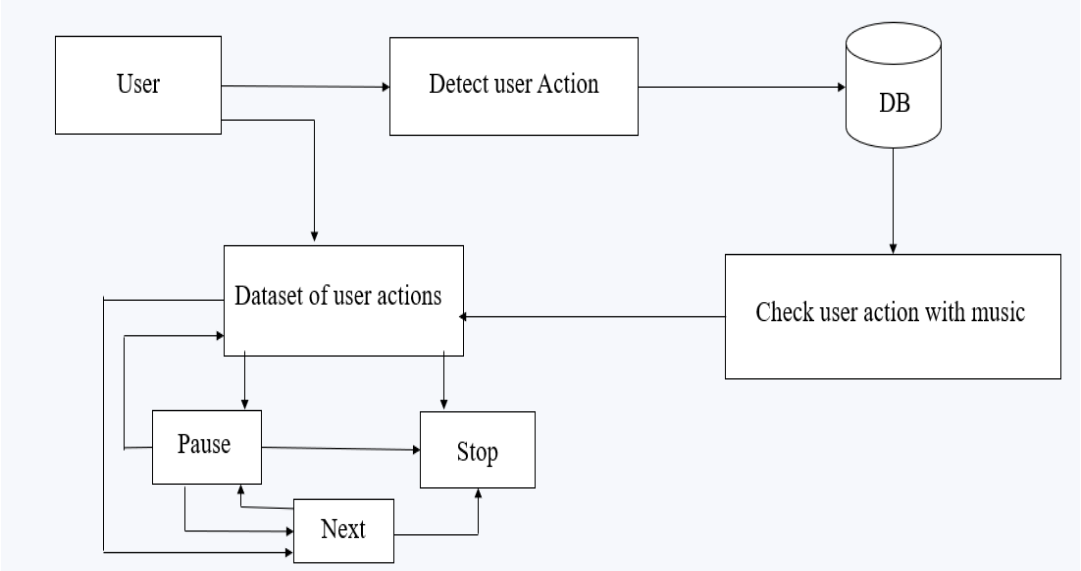


Fig 4.1 System Architecture

4.2 High-Level Design

High-Level Design (HLD) explains the architecture that would be used for developing a software product. The architecture diagram provides an overview of an entire system, identifying the main components that would be developed for their interfaces. The HLD uses possibly non-technical to mildly technical terms that should be understandable to the administrators of the system. In contrast, low-level design further exposes the logical detailed design of each of these elements for programmers.

4.3.1 Sequence Diagram

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagramthat shows how processes operate with one another and in what order. It is a construct of aMessage Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams. A sequence diagram also includes lifelines, which represent the lifespan of an object in the system. The lifeline is a vertical dashed line that extends from the top to the bottom of the diagram and is labeled with the name of the object it represents. Messages are shown as horizontal arrows that cross the lifelines of the objects they involve. Sequence also show the time and duration of each message exchange, which is indicated by vertical dotted lines and timing constraints. Timing constraints can be used to indicate the time delay.

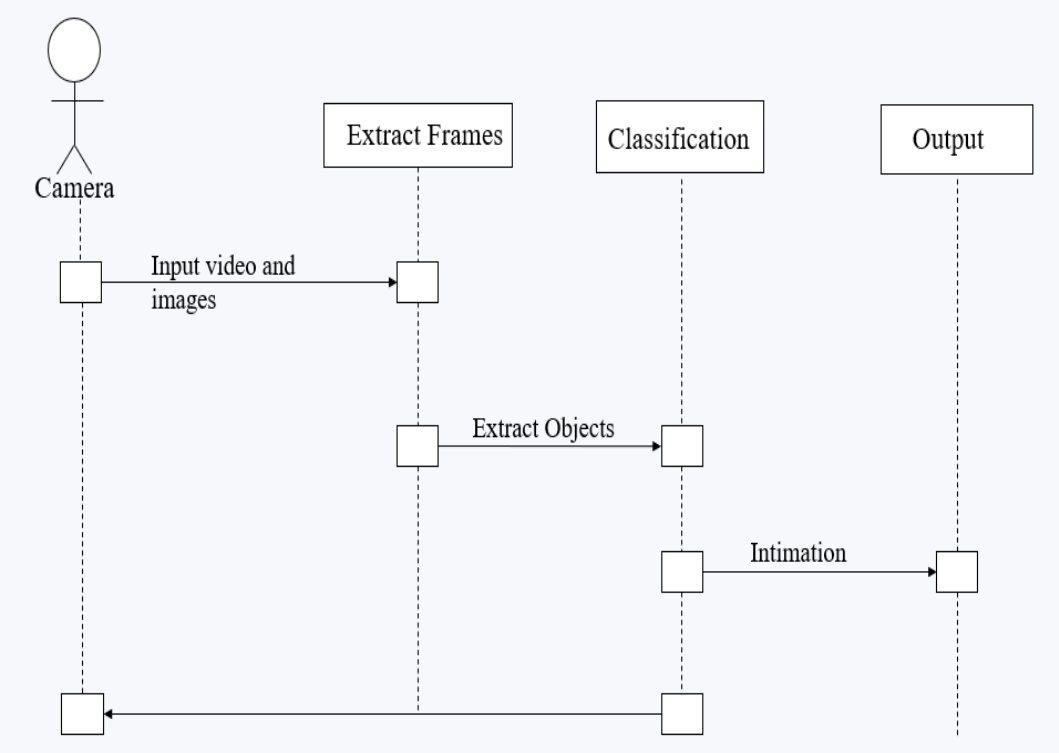


Fig 4.2 Sequence Diagram

4.3.2 Use Case Diagram

A use case diagram is a graphical representation of the interactions between a system and its users or external systems. It is a type of behavioral diagram in Unified Modeling Language (UML) and is used to define the scope of a system and its requirements. The primary elements of a use case diagram are actors and use cases. Actors are the users or external systems that interact with the system, while use cases represent the specific functionality or tasks that the system can perform. Use case diagrams also help to establish a common understanding of the system's requirements among stakeholders, including developers, project managers, and users. They can be used to communicate the system's functionality and requirements to all stakeholders, helping to ensure that everyone is on the same page. A typical use case diagram includes actors, use cases, and relationships. Actors are the users or external systems that interact with the system, while use cases represent the specific tasks or functionality the system can perform. Relationships between the actors and use cases can be of different types, including association, extend, and include. Association relationships show that an actor is associated with a particular use case. Extend

relationships show that a use case can be extended to include additional functionality. Include relationships show that a use case includes another use case as a subtask.

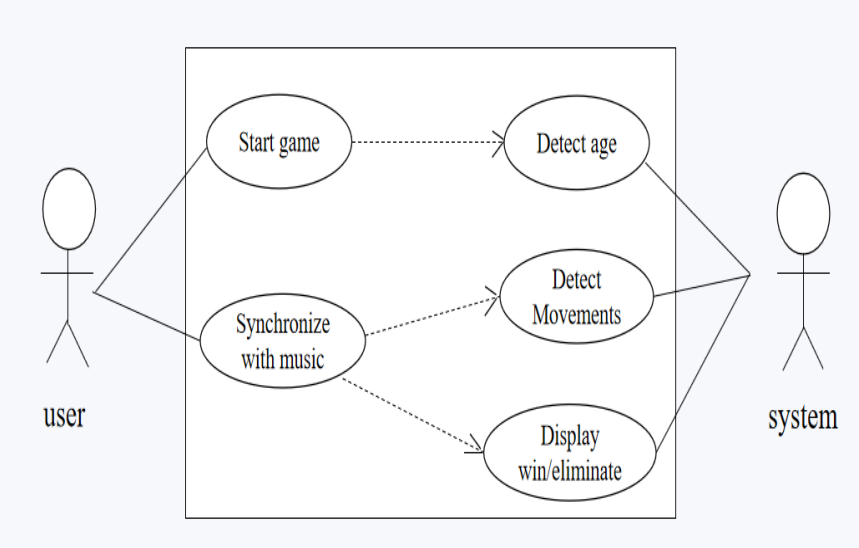


Fig 4.3 Use Case Diagram

4.4 Low-Level Design

Low-Level Design (LLD) is a component-level design process that follows a step by-step refinement process. This process can be used for designing data structures, required software architecture, source code, and ultimately, performance algorithms. Overall, the data organization may be defined during requirement analysis and then refined during data designwork. During the detailed phase, the logical and functional design is done and the design ofapplication structure is developed during the High-Level Design. Low-level diagrams are typically used in the development phase of a project to help engineers and developers understand the specific details of the system's components and their interactions. They are also used in testing and debugging to identify issues and optimize performance. Low-level diagrams can include various types of diagrams, including flowcharts, data flow diagrams, network diagrams, and component diagrams. Each type of diagram provides a specific perspective on the system's components and interactions, depending on the system's complexity and requirements.

4.4.1 Flowchart

A flowchart is a diagram that represents a set of instructions. Flowcharts normally use standard symbols to represent the different types of instructions. These symbols are used to construct the flowchart and show the step-by-step solution to the problem.

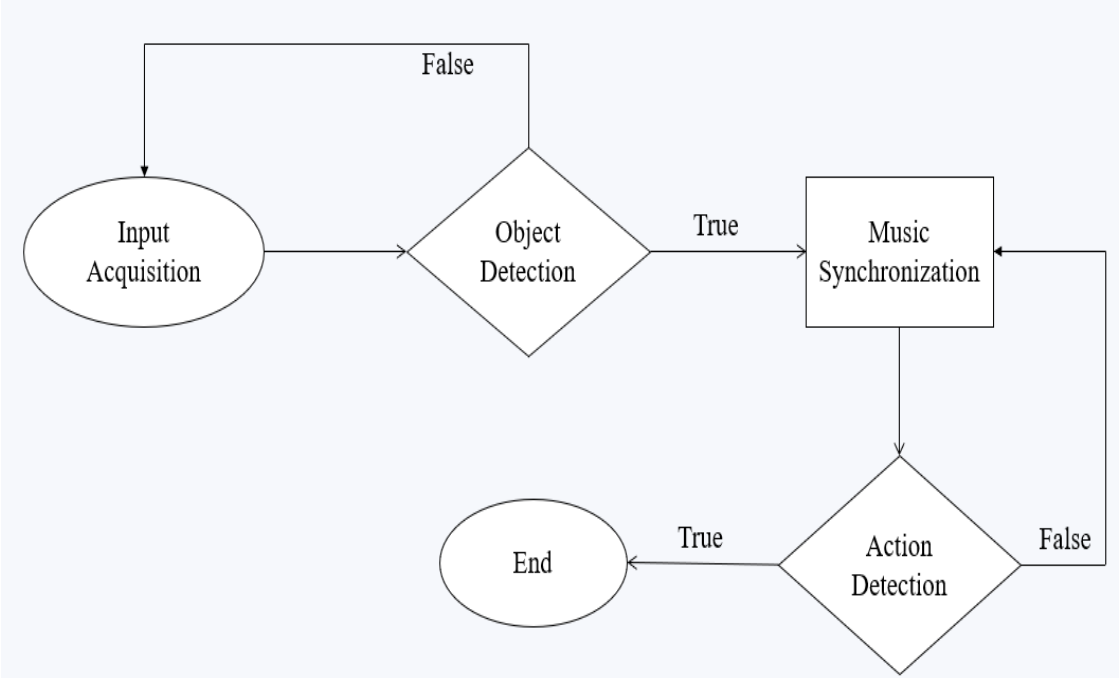


Fig 4.4: Flow Chart

CHAPTER-5
IMPLEMENTATION

5.1 Image Acquisition

Image Acquisition is the first step in any image processing system. The general aim of any image acquisition is to transform an optical image (real-world data) into an array of numerical data which could be later manipulated on a computer. Image acquisition is achieved by suitable cameras. We use different cameras for different applications. In this Project, the webcam is used for the image acquisition where the user face is captured.

5.2 Image Preprocessing

Image pre-processing is the term for operations on images at the lowest level of abstraction. These operations do not increase image information content but they decrease it if entropy is an information measure. The aim of pre-processing is an improvement of the image data that suppresses undesired distortions or enhances some image features relevant for further processing and analysis tasks.

There are 4 different types of Image Pre-Processing techniques and they are listed below.

1. Pixel brightness transformations/ Brightness corrections
2. Geometric Transformations
3. Image Filtering and Segmentation
4. Fourier transform and Image restauration

There are two types of Brightness transformations and they are below:

1. Brightness corrections
2. Gray scale transformation

Brightness transformations modify pixel brightness and the transformation depends on the properties of a pixel itself. In PBT, output pixel's value depends only on the corresponding input pixel value. Examples of such operators include brightness and contrast adjustments as well as color correction and transformations.

Contrast enhancement is an important area in image processing for both human and computer vision. It is widely used for medical image processing and as a pre- processing step in speech recognition, texture synthesis, and many other image/video processing applications.

Image segmentation is a commonly used technique in digital image processing and analysis to partition an image into multiple parts or regions, often based on the

characteristics of the pixels in the image. Image segmentation could involve separating foreground from background, or clustering regions of pixels based on similarities in color or shape.

5.3 Pose Detection

Pose estimation is a computer vision technique used to detect and track the position and orientation of human bodies or objects in images or videos. It involves analyzing the pixels in an image or video to identify key body points or features and using this information to estimate the pose or configuration of the object. There are different approaches to pose estimation, but one of the most common is the use of deep learning models such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). These models are trained on large datasets of annotated images or videos to learn the relationship between the input pixels and the output pose. In human pose estimation, the key body points or features that are typically detected include the head, neck, shoulders, elbows, wrists, hips, knees, and ankles. These points can be used to estimate the position and orientation of the body in 2D or 3D space.

Pose estimation has many practical applications, such as in sports and fitness tracking, motion capture for animation and gaming, surveillance and security, and robotics. For example, it can be used to track the movements of athletes during training or competitions, monitor the posture of workers to prevent injury, or guide the movements of robots in industrial settings. Pose estimation can also be combined with other computer vision techniques such as object detection and tracking to enable more complex applications such as action recognition, where the goal is to recognize specific actions or activities performed by humans or objects in the video. It is a computer vision technique that involves detecting and tracking the position and orientation of human bodies or objects in images or videos. It has many practical applications in areas such as sports, fitness, motion capture, surveillance, security, and robotics, and is often combined with other computer vision techniques to enable more complex applications.

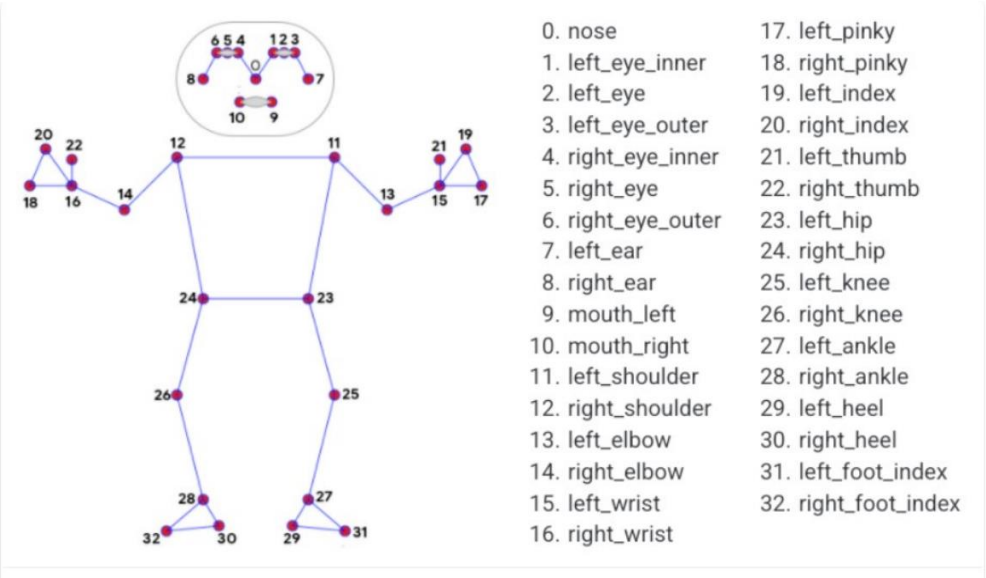
Fig 5.1: Pose Estimation

5.4 Skeleton Formation

Skeleton formation is a process in computer vision and graphics that involves detecting and modeling the underlying structure or skeleton of an object in an image or 3D scene. The skeleton can be thought of as a simplified representation of the object's shape or structure, capturing its key features and characteristics. Skeleton formation can be achieved through various techniques, including edge detection, region growing, and curve or surface fitting. These techniques aim to identify and extract the object's boundaries and features, such as corners, endpoints, and edges, and use them to generate a skeleton model. In 2D images, the skeleton can be represented as a set of connected line segments or curves that approximate the object's shape or structure. In 3D scenes, the skeleton can be represented as a set of connected surfaces or volumes that capture the object's shape and motion. Skeleton formation has many practical applications in computer vision and graphics, such as object recognition, tracking, and segmentation, shape analysis and classification, and animation and simulation. For example, it can be used to recognize and track human or animal movements in videos, segment and classify medical images of organs and tissues, or animate characters and creatures in games and movies.

In addition to its practical applications, skeleton formation is also of interest in the field of computer vision and graphics because it provides a way to study and understand the structure and organization of natural and artificial objects in a quantitative and systematic way. It enables researchers to analyze the shape and motion of objects, identify

common patterns and structures, and develop new algorithms and models for computer vision and graphics.



5.2: Landmark Points

5.5 Music Synchronization

Music synchronization can also be implemented in Python using various libraries and tools available for audio processing and analysis. One popular library for audio processing in Python is Playsound, which provides tools for loading, analysing, and manipulating audio signals. Playsound can be used to extract features from audio signals, such as tempo, rhythm, and harmony, which can then be used to synchronize music with visual media. Another popular library for audio analysis in Python is Essentia, which provides tools for extracting various music features, including beat detection, key and chord recognition, and mood analysis. These features can be used to synchronize music with visual media and create a more immersive and engaging experience for the audience.

Python also provides tools for working with MIDI files, which are commonly used in music production and synchronization. The Mido library provides tools for reading and writing MIDI files, as well as tools for manipulating MIDI events and messages. This can be useful for synchronizing music with visual media in real-time, such as in live performances or interactive installations.

5.6 Modules Used

5.6.1 OpenCV

OpenCV also provides tools for machine learning, such as support vector machines (SVMs) and decision trees, which can be used for tasks such as classification, regression, and clustering. These machine learning tools can be trained on large datasets of labelled images or videos to learn patterns and relationships between the input data and the output. In addition, OpenCV provides tools for image and video processing, such as filtering, thresholding, and morphological operations. These tools can be used to pre-process images and videos before applying computer vision algorithms, or to enhance the visual quality of the output.

OpenCV is widely used in various industries, such as robotics, autonomous vehicles, surveillance, and medical imaging. It is also used in academic research for computer vision and machine learning applications. OpenCV is a powerful and versatile library for computer vision and machine learning applications, providing a wide range of tools and functions for processing and analysing images and videos in real-time. Its cross-platform and multi-language support make it a popular choice for developers and researchers working in different environments.

Fig 5.3: OpenCV

5.6.2 Playsound

Playsound is a Python library that allows you to play sound files in your Python scripts. It is a lightweight, cross-platform library that works on Windows, macOS, and Linux. The library uses the default sound player of the operating system to play the audio files. This means that on Windows, it uses Windows Media Player, on macOS it uses QuickTime, and on Linux, it uses the Streamer framework.

Playsound is very easy to use. Once you install the library, you can import the playsound function and use it to play sound files in your Python scripts.

One of the advantages of playsound is its simplicity. It doesn't require any additional dependencies or complex configurations to work. This makes it a great option for developers who want to add simple sound playback functionality to their Python scripts.

5.6.3 MediaPipe

MediaPipe is an open-source, cross-platform framework for building multimodal machine learning applications. It was developed by Google and is maintained by the MediaPipe team. The framework provides a wide range of tools and components for developing applications that process audio, video, and other sensor data in real-time. Some of the key features of MediaPipe include:

1. Modular Architecture: MediaPipe is designed with a modular architecture, allowing developers to build complex applications by combining pre-built components or building custom components.
2. Cross-Platform Support: MediaPipe supports multiple platforms, including Android, iOS, Linux, and Windows.
3. Real-time Performance: MediaPipe is optimized for real-time performance, making it well-suited for applications that require low latency and high throughput.
4. Machine Learning: MediaPipe provides built-in support for machine learning frameworks, such as TensorFlow, allowing developers to easily integrate ML models into their applications.

One of the most popular components of MediaPipe is the Pose Estimation module, which provides real-time detection of human body poses using machine learning algorithms. This module can be used in a wide range of applications, such as fitness tracking, gesture recognition, and augmented reality.

MediaPipe also provides tools for video and audio processing, including object detection, face detection, and hand tracking. These tools can be used to build a wide range of applications, such as video analytics, virtual try-on, and live streaming.

Fig 5.4: Media Pipe

5.6.4 Tkinter

Tkinter is a standard Python library for creating graphical user interfaces (GUIs). It is a built-in module in Python, so no additional installation is required. Tkinter provides a set of tools for creating windows, buttons, labels, text boxes, and other GUI elements. Tkinter provides a wide range of widgets, such as buttons, labels, text boxes, and menus, that can be used to create windows, dialogs, and other user interface components. These widgets can be customized with various properties, such as colors, fonts, and sizes, to create visually appealing interfaces. Tkinter also provides event handling mechanisms that allow developers to define how the application responds to user actions, such as mouse clicks, key presses, and window resizes. This makes it possible to create interactive applications that can perform complex tasks in response to user input. One of the key advantages of Tkinter is that it is easy to learn and use, even for beginners. The library provides a set of widgets, such as buttons and labels, that can be added to a GUI with just a few lines of code. Tkinter also provides layout managers that help to position widgets in a window.

Fig 5.5: Tkinter

5.6.5 Multi-Threading:

Multithreading is a programming concept that involves running multiple threads of execution concurrently within a single program. A thread is a lightweight sub-process that can run concurrently with other threads, sharing the same memory and resources of the parent process. Multithreading is used to achieve concurrency in programs, which can improve the performance and responsiveness of the application. For example, in a video player application, the main thread can handle user input and display the user interface, while a separate thread can handle the video decoding and playback.

In Python, multithreading is supported by the threading module, which provides a way to create and manage threads. Python's Global Interpreter Lock (GIL) is a feature that ensures only one thread can execute Python bytecode at a time. This means that multithreading in Python does not provide true parallelism, as only one thread can execute

Python code at a time. However, multithreading can still provide benefits such as improved performance and responsiveness for I/O-bound applications.

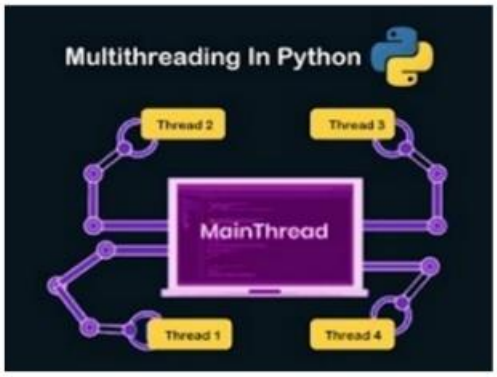


Fig 5.6: Multi-Threading

5.6.6 MoveNet

MoveNet Multipose is a computer vision model developed by Google that enables real-time and accurate detection and tracking of multiple human poses in images or videos. It is an extension of the MoveNet architecture, designed specifically to handle scenarios with multiple individuals present in the scene. It utilizes deep learning techniques, leveraging a lightweight convolutional neural network architecture. This design choice allows the model to be deployed on a variety of devices, including mobile phones, tablets, and embedded systems, while still maintaining real-time performance. The primary objective of MoveNet Multipose is to estimate the locations of key body keypoints for each individual in the image or video. These keypoints include the nose, shoulders, elbows, wrists, hips, knees, and ankles. By accurately determining the positions of these keypoints, the model can infer the poses and movements of each person. The MoveNet Multipose architecture consists of several stages. Firstly, a single-pose detector identifies and localizes each individual's bounding box within the frame. This initial detection provides a rough estimate of the person's location. Next, a multi-pose detector refines the detection by estimating the key point locations within each bounding box. This refinement step improves the accuracy and robustness of the pose estimation. To ensure consistency and smoothness across consecutive frames, MoveNet Multipose employs a keypoint association and tracking algorithm. This algorithm maintains the identity of individuals and tracks their poses over time, even in complex scenarios with occlusions or overlapping individuals.

MoveNet Multipose is trained on large-scale annotated datasets, allowing it to learn from a diverse range of human poses and variations. The model benefits from transfer learning techniques, which enable it to generalize well to different environments and datasets. The lightweight architecture and efficient optimizations of MoveNet Multipose make it an ideal solution for various real-time applications. It can be used in fitness tracking applications to analyse and monitor exercise movements, in gesture recognition systems for natural user interfaces, and in augmented reality and virtual reality experiences to enable realistic and interactive human interactions. It is a powerful and efficient computer vision model that enables real-time and accurate detection and tracking of multiple human poses. Its lightweight architecture, combined with its ability to handle occlusions and overlapping individuals, makes it a valuable tool for a wide range of applications in fields such as fitness, gaming, and human-computer interaction.

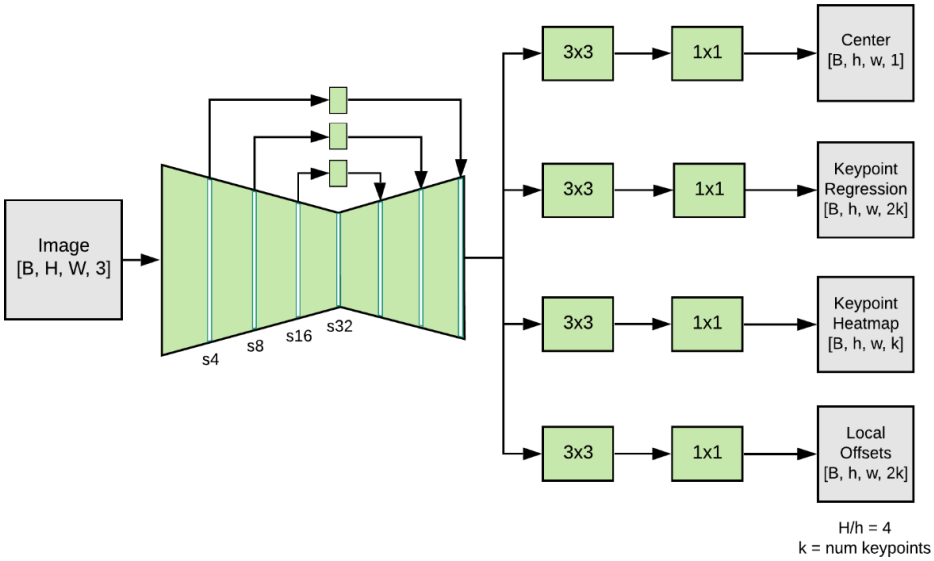


Fig 5.7: MoveNet Architecture

5.6.7 TensorFlow

TensorFlow is a popular deep learning framework developed by Google, while multipose refers to the task of detecting and tracking multiple human poses in images or videos. However, I can provide you with information on how TensorFlow can be used for multipose pose estimation. In TensorFlow, multipose pose estimation can be accomplished by utilizing various pre-existing models and techniques. One popular approach is to use a combination of object detection models and pose estimation models. Firstly, object detection models such as Faster R-CNN, SSD (Single Shot MultiBox Detector), or YOLO (You Only Look Once) can be employed to identify and localize individuals within an image or video frame. These models can provide bounding box predictions around each

person, which serve as input for subsequent pose estimation.For pose estimation, TensorFlow provides several options. One commonly used model is OpenPose, which is a deep learning-based approach that estimates body keypoints for each individual. OpenPose utilizes a multi-stage architecture, combining convolutional neural networks and graphical models to detect and associate keypoints.

Another popular option is the MoveNet model, which we discussed earlier. MoveNet is a lightweight model that can estimate poses for multiple individuals in real-time. It is compatible with TensorFlow and can be utilized for multipose pose estimation tasks. To utilize these models, TensorFlow offers a variety of APIs and tools. The TensorFlow Object Detection API allows you to train and deploy object detection models, while the TensorFlow Model Zoo provides pre-trained models that can be used out-of-the-box. Additionally, TensorFlow's high-level API, such as Keras, simplifies the process of building and training pose estimation models. Once you have the object detection and pose estimation models set up, you can run them sequentially on images or video frames to detect individuals and estimate their poses. By leveraging the power of TensorFlow's parallel computing capabilities, you can achieve real-time or near real-time performance.

It's important to note that the choice of models and techniques may vary based on your specific requirements and constraints. Additionally, there may be other custom or state-of-the-art models available outside the TensorFlow ecosystem that can also be used for multipose pose estimation. While there is no specific "TensorFlow Multipose" model, TensorFlow provides a rich ecosystem of tools, models, and APIs that can be utilized to develop multipose pose estimation systems. By combining object detection models with pose estimation models like OpenPose or MoveNet, TensorFlow enables accurate and real-time detection and tracking of multiple human poses in various applications.

Fig 5.8: TensorFlow

CHAPTER - 6

TESTING

6.1 Design of Test Case

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies, and/or a finished product It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail unacceptably. There are various types of tests. Each test type addresses a specific testing requirement. Test cases are a crucial component of software testing, as they are used to ensure that a software application or system is working as intended. A test case is a set of steps or actions that are executed to verify the functionality, performance, and reliability of an application or system.

6.2 Types of Testing

6.2.1 Unit Testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at the component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results. The benefits of unit testing include improved code quality, reduced software defects, faster debugging and troubleshooting, and greater confidence in the reliability of the software. By isolating and testing individual components of a system, developers can identify and fix defects more quickly and efficiently, reducing the overall time and cost of software development.

6.2.2 Integration Testing

Integration tests are designed to test integrated software components to determine if they run as one program. Testing is event-driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfied, as shown by successful unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components. The benefits of integration testing include improved software quality, reduced software defects, faster debugging and troubleshooting, and greater confidence in the reliability of the software. By testing the interactions between different software components, developers can ensure that the software works as intended, and that all modules and services function correctly and efficiently. This testing can be complex and time-consuming, and requires careful planning and coordination to ensure that all components are tested thoroughly and effectively. It should be complemented by other testing methodologies such as unit testing, system testing, and acceptance testing to provide a comprehensive and effective testing strategy.

6.2.3 Functional Testing

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items: Valid Input: identified classes of valid input must be accepted. Invalid Input: identified classes of invalid input must be rejected. Functions: identified functions must be exercised. Output: identified classes of application outputs must be exercised. Systems/Procedures: interfacing systems or procedures must be invoked. The organization and preparation of functional tests are focused on requirements, key functions, or special test cases. In addition, systematic coverage identifies business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined. The benefits of functional testing include improved software quality, reduced software defects, faster debugging and troubleshooting, and greater confidence in the reliability of the software. By testing the software's functionality against its requirements, developers can identify and fix defects and issues before the software is released to the end-user. However, functional testing may not cover all aspects of the software, such as non-functional requirements or compatibility with different hardware and software configurations.

Therefore, it should be complemented by other testing methodologies such as non-functional testing and compatibility testing to provide a comprehensive and effective testing strategy.

6.2.4 System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points. The benefits of system testing include improved software quality, reduced software defects, faster debugging and troubleshooting, and greater confidence in the reliability of the software. By testing the entire system as a whole, developers can identify and fix defects and issues that may have been missed during integration testing or other testing phases. However, system testing can be complex and time-consuming, and requires careful planning and execution to ensure that all aspects of the software are tested thoroughly and effectively. It should be complemented by other testing methodologies such as unit testing, integration testing, and acceptance testing to provide a comprehensive and effective testing strategy.

6.2.5 White Box Testing

White Box Testing is a testing in which the software tester knows the inner workings, structure, and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black-box level. The benefits of white box testing include improved code quality, reduced software defects, faster debugging and troubleshooting, and greater confidence in the reliability of the software. By testing the internal structure and implementation of the software, developers can identify and fix defects and issues at an early stage, before they can propagate to other parts of the system and become more difficult and expensive to fix. Therefore, it should be complemented by other testing methodologies such as black box testing and grey box testing to provide a comprehensive and effective testing strategy.

6.2.6 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure, or language of the module being tested. Black box tests, like most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document.

It is a testing in which the software under test is treated, as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works. The benefits of black box testing include improved software quality, reduced software defects, faster debugging and troubleshooting, and greater confidence in the reliability of the software. By testing the software from the perspective of an end-user, testers can identify and fix defects and issues that may impact the end-user experience. However, black box testing may not cover all aspects of the software, such as internal logic and control structures, and may not identify defects that are related to the software's internal workings. Therefore, it should be complemented by other testing methodologies such as white box testing and grey box testing to provide a comprehensive and effective testing strategy.

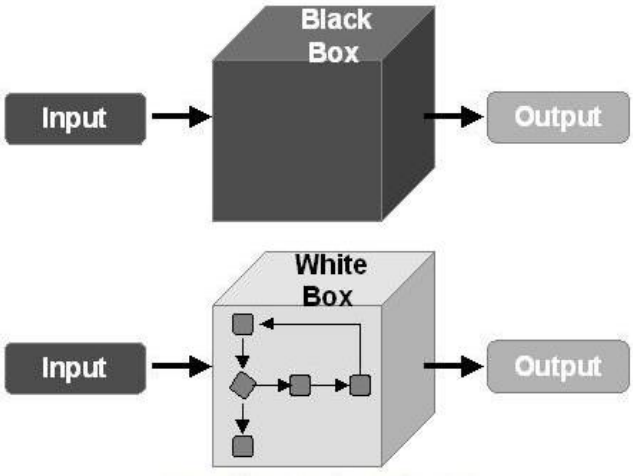


Fig 6.1: Black box and White box Testing

6.3 Test Cases

Sl No.	Req_Id	Test Case Reference	Input	Output	Test Result	Remarks
1	UTC-1	TC-1	Camera stream	Should capture input image	Pass	Image captured success
2	UTC-2	TC-2	User not in frame	Skeleton was not drawn	Pass	Image not captured
3	UTC-3	TC-3	Video of the human	Object like human will be detected	Pass	Object detected
4	ITC-1	TC-4	Video stream	Detect objects	Pass	Capture image and automated object classification
5	ITC-2	TC-5	Human action like walking	Action recognition	Pass	Functioned properly
6	ITC-3	TC-6	Video stream	Action recognition with music	Pass	Music synchronizat -ion
7	ITC-4	TC-7	Image	Unable to calculate position according to dataset	Fail	Game results are not concluded
8	STC-1	TC-8	Execute program in different OS	Performed better in various OS	Pass	Better performance of OS

Table 6.1 Test cases

CHAPTER - 7

RESULT DISCUSSIONS AND SNAPSHOTS

7.1 Result Analysis

This is the result of what we have found the best choice for optimizing the storage space and upload bandwidth over the cloud is source-based de-duplication. The process of distributed deduplication helps in achieving the process of reliability, confidentiality, and security. Also, a good deduplication ratio can be achieved by modifying the deduplication algorithm further. Our project front end was developed in Tkinter with play sound module which plays instructions 10 seconds and directly linked to the backend. When the criteria for execution of the program i.e. complete visibility of all the 33 landmarks visibility is needed to start the game. When green light is signaled the player is free to move When the red light is signaled object detection using media pipe landmark begins and if movement is detected player is eliminated. When the movement is detected via matrix generated in the landmarks and after NumPy results crosses a certain threshold, which is considered as the decision making of the game.

7.2 Snapshots

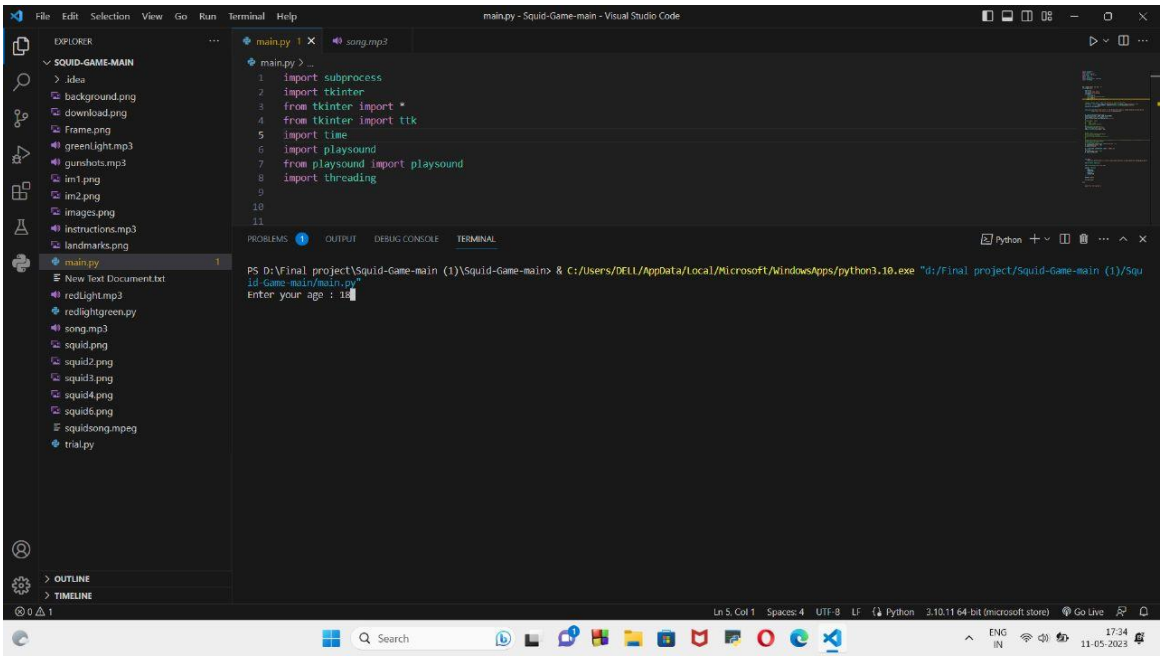


Fig 7.1: Entering the Age

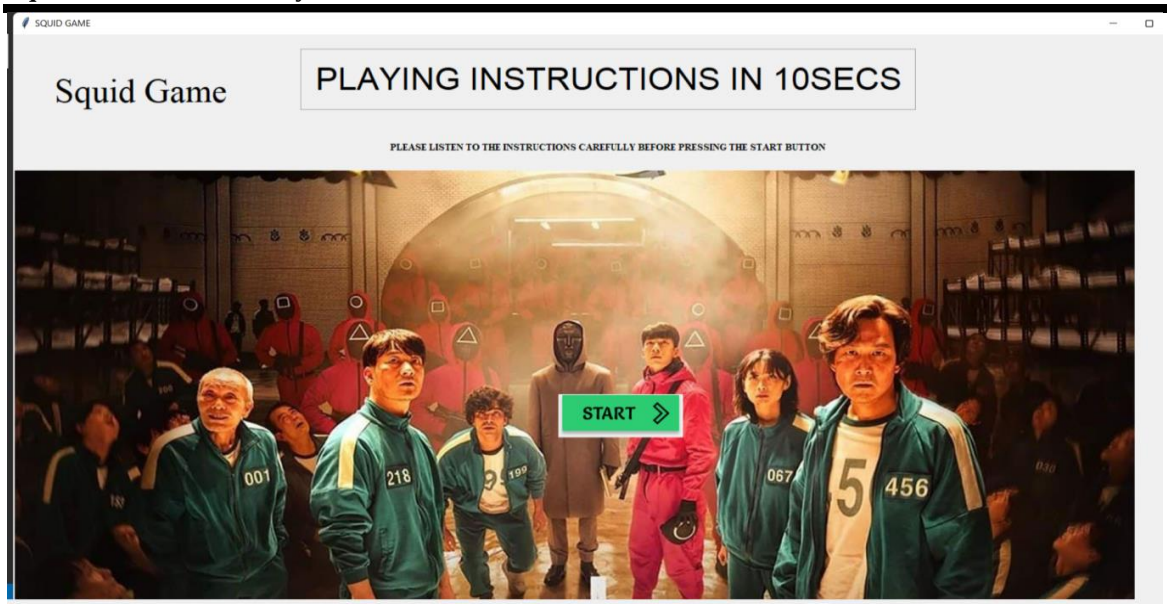


Fig 7.2: Front End of Squid Game

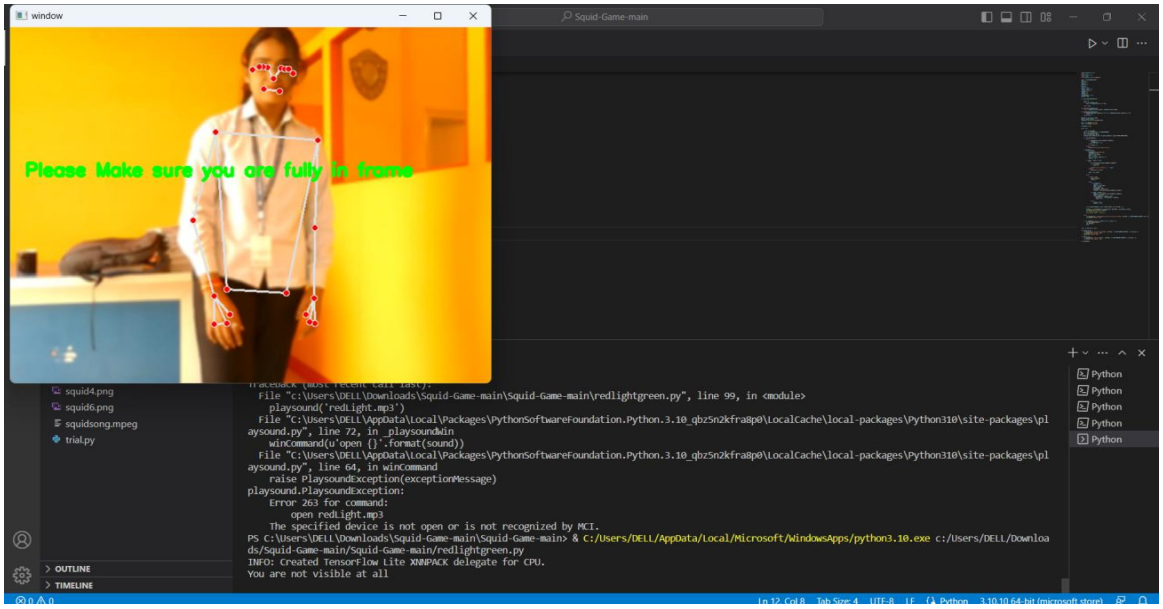


Fig 7.3: Criteria not meeting Visibility

Fig 7.4: Green Light Signaled and Game begins

Fig 7.5: Red Light Signaled and movement detection begins

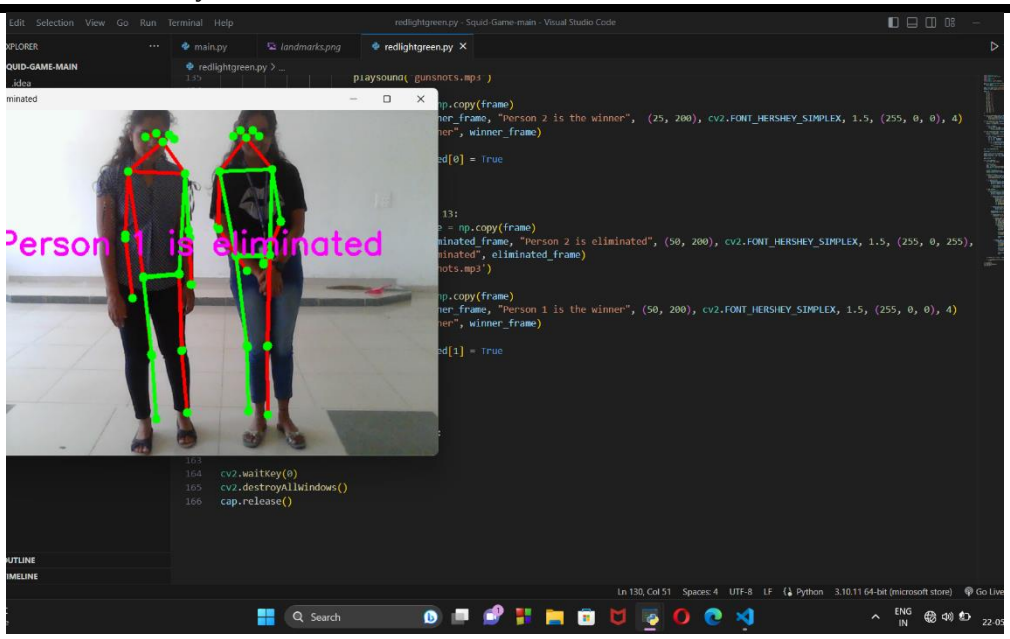


Fig 7.6: Result showing elimination of person 1

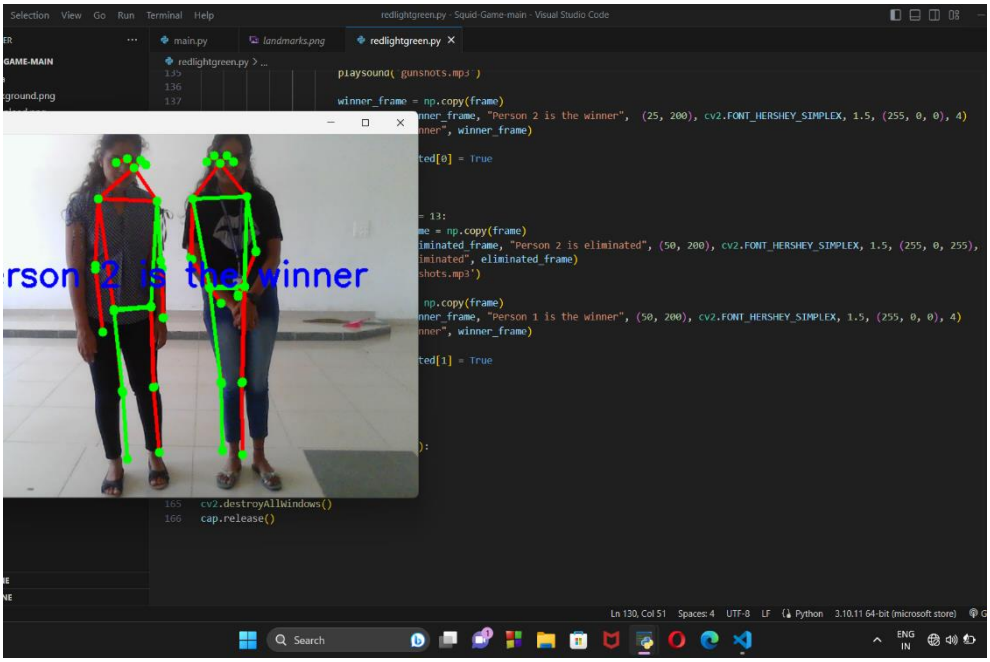


Fig 7.7: Result showing person 2 as the winner

CONCLUSION AND FUTURE ENHANCEMENT

Conclusion

When compared to the conventional immersive gaming technologies and the hardware and processing capabilities needed for its effective functioning, we have proposed implementation of artificial intelligence and machine learning into the immersive gaming technology, which was possible due to the previous success observed in implementation of media pipe in exercise posture as well as yoga pose detection which facilitated its application in the field of immersive gaming, With the help of media pipe and ai and ml implementation using tensor flow the capabilities of the system needed for an immersive gaming experience is considerably reduced.

Future Enhancement

For the future enhancement, we shift our research to multi-view human action estimation. The problem is more complex because of the multiple views and inference in 3D space and thus requires more training data for learning a model. For those reasons, we consider generative models or a combination between generative and discriminative models. We can implement this in Malls or any gamming center for real-time playing of game.

BIBLIOGRAPHY

[1] Venkata Ramana, Lakshmi Prasanna "Human activity recognition using opencv", International journal of creative research thoughts (IJCRT), (2021).

[2] Lamiyah Khattar, Garima Aggarwal "Analysis of human activity recognition using deep learning." 2021 11th International Conference on Cloud Computing, Data Science & Engineering.

[3] Long Cheng,Yani Guan "Recognition of human activities using machine learning methods with wearable sensors" IEEE Members Research and Development Departmentin 2017.

[4] Ran He, Zhenan Sun "Adversarial cross spectral face completion for NIR-VIS face recognition." IEEE paper received on January 2019.

[5] Jing Wang, Yu cheng "Walk and learn: facial attribute representation learning",2016 IEEE Conference on Computer Vision and Pattern Recognition.

[6] Yongjing lin,Huosheng xie "Face gender recognition based on face recognition feature vectors", International Conference on Information Systems and Computer Aided Education (ICISCAE),(2020).

[7] Mohanad Babiker, Muhamed Zaharadeen "Automated Daily Human Activity Recognition for Video Surveillance Using Neural Network." International Conference on Smart Instrumentation, Measurement and Applications (ICSIMA) 28- 30 November 2017.

[8] Neha Sana Ghosh, Anupam Ghosh "Detection of Human Activity by Widget." 2020 8th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) June 4-5,2020.

[9] AN YANG, WANG KAN "Segmentation and Recognition of Basic and Transitional Activities for Continuous Physical Human Activity" IEEE paper on 2016.