D.K.T.E. Society'sTextile and Engineering Institute, Ichalkaranji.

(An Autonomous Institute)

DEPARTMENT OF INFORMATION TECHNOLOGY
2020-2021

Project report on

"Gesture Controlled Gaming"

Under the guidance of

Prof. Mrs.D.M.Kulkarni

Group Members

1.	Mr. Shahid Rafik Mujawar	18UIT72013XX
2.	Mr. Anirudhha Anil Lokare	18UIT72007XX
3.	Mr. Aniket Manik Hiremath	17UIT12015XX
4.	Ms. Diksha Indrajit Patil	18UIT71010XX
5.	MS. Madhu Shribhagwan Toshniwal	17UEL11104IT
6.	Mr. Sayyad Alamgir Jahangir	16UIT12047XX

D.K.T.E. Society'sTextile and Engineering Institute, Ichalkaranji.

(An Autonomous Institute)

ACCREDITED WITH 'A+' GRADE BY NAAC

An ISO 9001: 2015 Certified

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that

1.	Mr. Shahid Rafik Mujawar	18UIT72013XX
2.	Mr. Anirudhha Anil Lokare	18UIT72007XX
3.	Mr. Aniket Manik Hiremath	17UIT12015XX
4.	Ms. Diksha Indrajit Patil	18UIT71010XX
5.	MS. Madhu Shribhagwan Toshniwal	17UEL11104IT
6.	Mr. Sayyad Alamgir Jahangir	16UIT12047XX

Have successfully completed the project work, entitled,

Gesture Controlled Gaming

in partial fulfillment for the award of degree of Bachelor of Technology in Information Technology. This is the record of their work carried out during the academic year 2020-2021.

Date: **Place:Ichalkaranji**

Prof. Mrs. D.M. Kulkarni	**Prof. P.V. Kadole**	**Prof. D. V. Kodavade**
[Project Guide]	**[Director/Principal]**	**[HeadofDepartment]**

DECLARATION

We are the undersigned students of Final YearB.Tech I.T. declare that, the field work report entitled **Gesture Controlled Gaming** written and submitted under the guidance of Prof. Mrs. D.M. Kulkarni is our original work. The empirical findings in this report are based on the data collected by us. The matter assimilated in this report is not reproduction from any readymade report.

Date:

Place: Ichalkaranji.

Name **Signature**

1. Mr. Shahid Rafik Mujawar
2. Mr. Anirudhha Anil Lokare
3. Mr. Aniket Manik Hiremath
4. Ms. Diksha Indrajit Patil
5. MS. Madhu Shribhagwan Toshniwal
6. Mr. Sayyad Alamgir Jahangir

INDEX

ABSTRACT

Recently, strong efforts have been made to bridge the gap between human and computer-based systems by making the interactions (which were via input devices like keyboards and mouse) as natural as possible through gesture controls. Gesture recognition is useful for processing information from humans which is not conveyed through speech or type.

The gaming controls we use in our day-to-day life using controls from keyboards or mobile screens for e.g., pressing a right arrow, left arrow to move the object can be easily done by using our own gestures to do the same task. It will be very easy for the users to play and will help them to improve their Physique.

1.Introduction

Why do we need gaming controls using gestures? The answer to this question is, in normal gaming we use the controls by pressing it, to provide ease to the user we developed controls using gestures which will be interesting as well. As there are a large number of users who love to play games so it is better to provide some different features for them. For+ e.g., In arcade, action and racing games, players usually control the game with the help of keyboards or joysticks. This may be hectic for some people, so using the gestures there will be a little physical exercise and it will be more interactive and affordable. There is a need to create a much more convenient way to communicate with computers while playing the games, that would feel more natural and interactive to human users. The traditional methods include the use of keyboards, mouse and other controller I/O devices to control games. These methods had some bottlenecks which include the complexity of the controller device and the generic unnaturalness of the interaction as a whole, and the series of wired connections which limit the range of operation of the computer by the user. This project aims to demonstrate an easier and efficient way of interacting with the computer.

a. Problem definition:

Operating game controls using hand gestures.

b. Aim and objectives of the Project:

- To recognize the coordinates of the body.
- To take action according to the gestures done by the user.
- To make games more interactive for users.
- Without using hardware controlling the game.

c. Scope and limitation of the Project:

▪ Scope

This project will be used by any person who enjoys playing games. As it does not require much resources. This works best on car racing games and it is more interactive than playing games using keyboard controls. You can actually feel that you are driving the car.

▪ Limitations

- This doesn't support on games with higher complexity, like PUBG, GTA 5, etc.
- It cannot be run without a webcam.
- On other hand, if the gesture is not done properly and if it is not recognized,

  a player can tend to miss any movement, or will be performing different movements.

d. Timeline of Project

For the life cycle process, we have used giant chart representation which shows the duration of requirement analysis, designing, coding and testing period spent on the project. So, in the start of July, we researched about the requirements that would be needed for the project and what resources must be required for better performance of the project, what language could be used, related projects, libraries that can ease the process, etc. We spent nearly 2 months analyzing all the requirements that would be needed for the project.

After analyzing requirements, we started to work on the designing phase in the month of September.

We created an architectural diagram, data flow diagram, sequence diagram and system flow diagram.

After designing we worked on implementation in the month of November. We required 3-4 months in implement the code and GUI for the application. And after the Coding phase, we did all the required testing in the duration of 1 month.

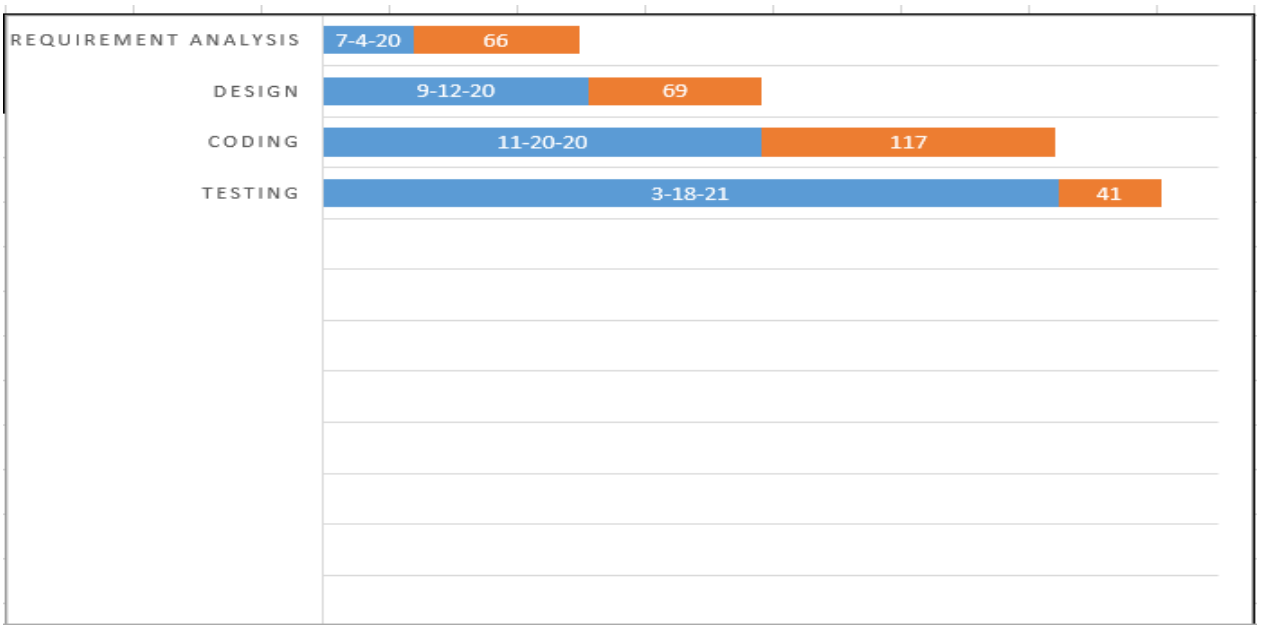


Fig 1.1 Timeline of project

Background study
And
Literature overview

2. Literature Overview

a. Literature Overview:

1) As we can see the existing gaming systems, In that use of a traditional keyboard, Mouse and console are unavoidable. According to today's technology, it feels more unnatural to play most of the games. Nowadays, players would like to be involved in the game. Physical control devices can’t fulfill that requirement.

2) Today, we are having some consoles from companies like Sony, Microsoft, Oculus, which uses virtual reality in their gaming systems. It gives us a more realistic experience of gaming. But they are expensive and only accessed using high-end machines. Most of the people cannot afford that.

b. Existing System:

Talking about the related technology, Sony has developed a PSVR kit which is PlayStation Virtual Reality kit. It consists of four to five gadgets used for controlling the game. This costs around 1,25,000 Rs, this is a huge amount and of course not affordable to the common gamer. Also, we have many driving controllers for racing games which consist of a steering wheel, pedals, and gear shifters. This too costs about 12000 Rs approximately. There are many controllers in the electronic market used for playing the games, though some have their own advantages, and disadvantages, but the common disadvantage is cost so we have hit the common disadvantage of existing systems and brought our project.

Requirement Analysis

3. Requirement Analysis

a. System Requirement

1. Software and Hardware requirements:

- **Hardware requirement:**

- External Webcam (if user uses computer)
- RAM – minimum 6 GB
- Processor – Intel Core i5 5th generation or higher.

- **Software requirements:**

- Operating System – Any OS that supports Python.
- Python 3.7 with eel and keyboard library.
- Browser which supports JavaScript.

b. Functional Requirements:

1) UI Requirements: -

- Software should have neat and simple design so that users will find it easy to handle it.
- There should be a popup for instructions.
- There should be two buttons to select control type, WASD controls or ARROW buttons.
- The camera popup window should be of size 300*300px.

c. Analysis

- If gestures are not done properly then the action will not happen according to the user.
- If the user sits very close to the webcam, then software will find it difficult to recognize the coordinates.
- If the background light fluctuates then it might cause trouble in recognizing coordinates

System Design

4. System Design

a. Architectural Design of system: -

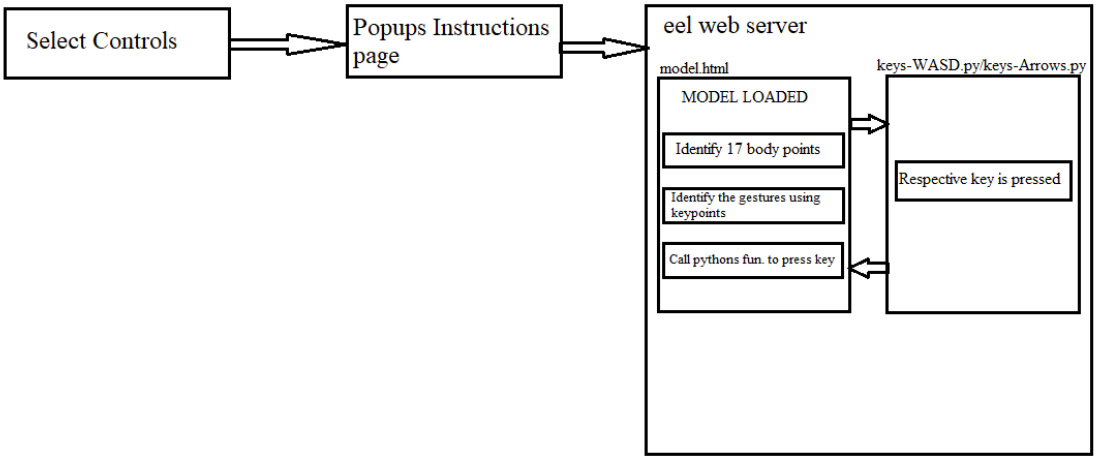


Fig 4.1 Architecture diagram of project

a. Data Flow Diagram

Level-0 Data Flow Diagram

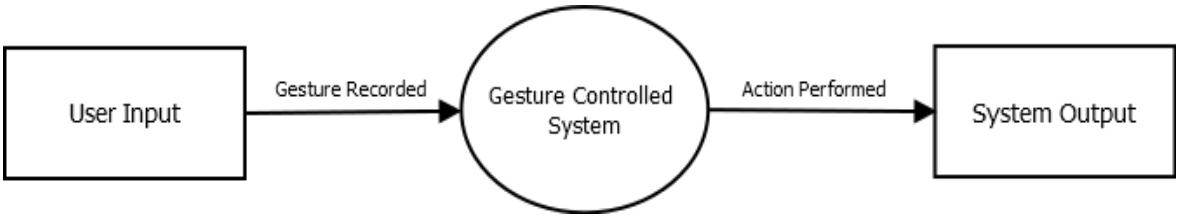


Fig. 4.2 DFD Level 0

Level-1(a) Data Flow Diagram

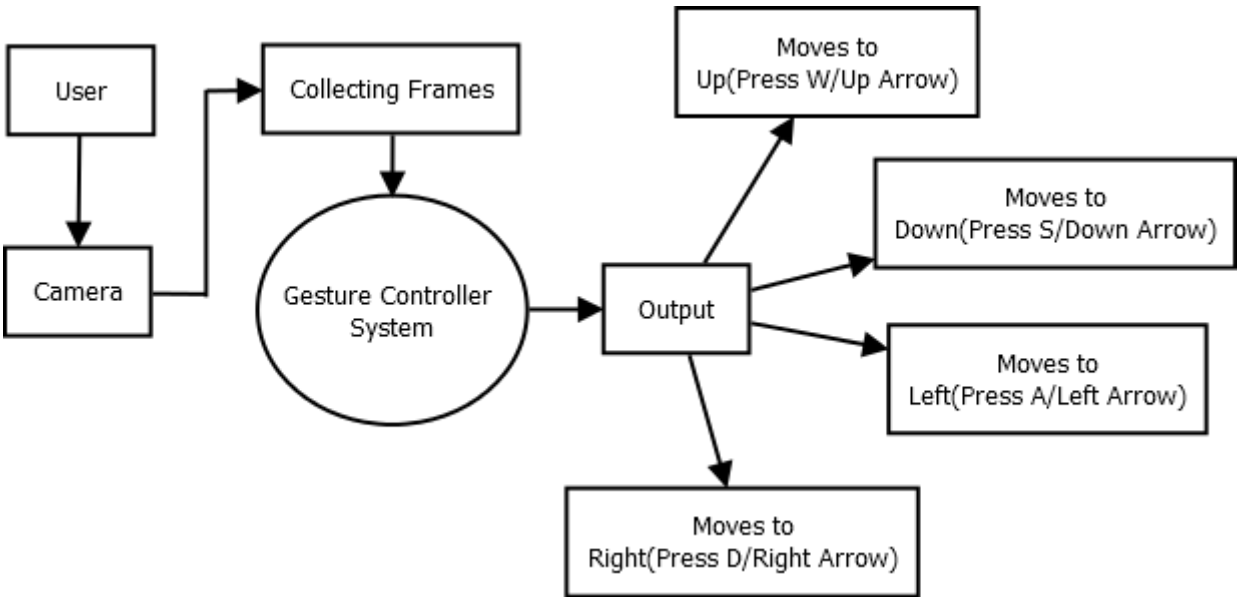


Fig 4.3 DFD Level 1

b. Sequence Diagram

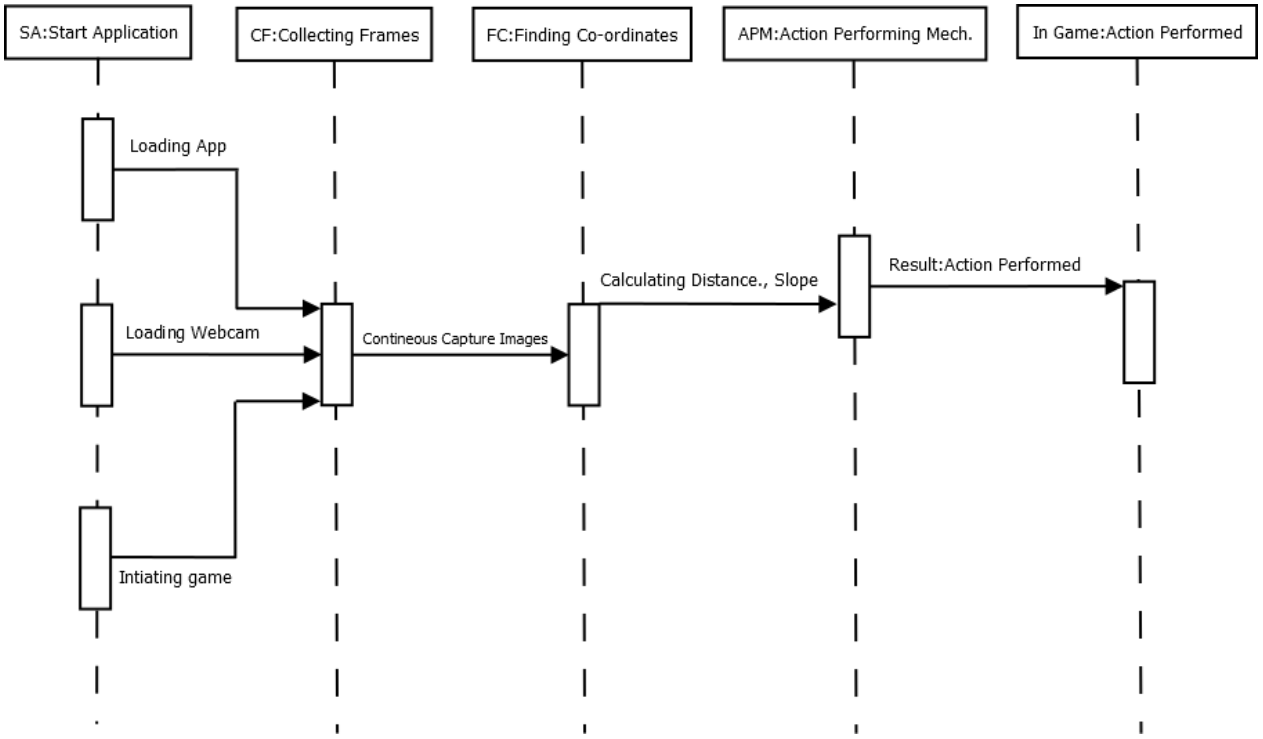


Fig 4.4 Sequence Diagram

c. System Flow Diagram

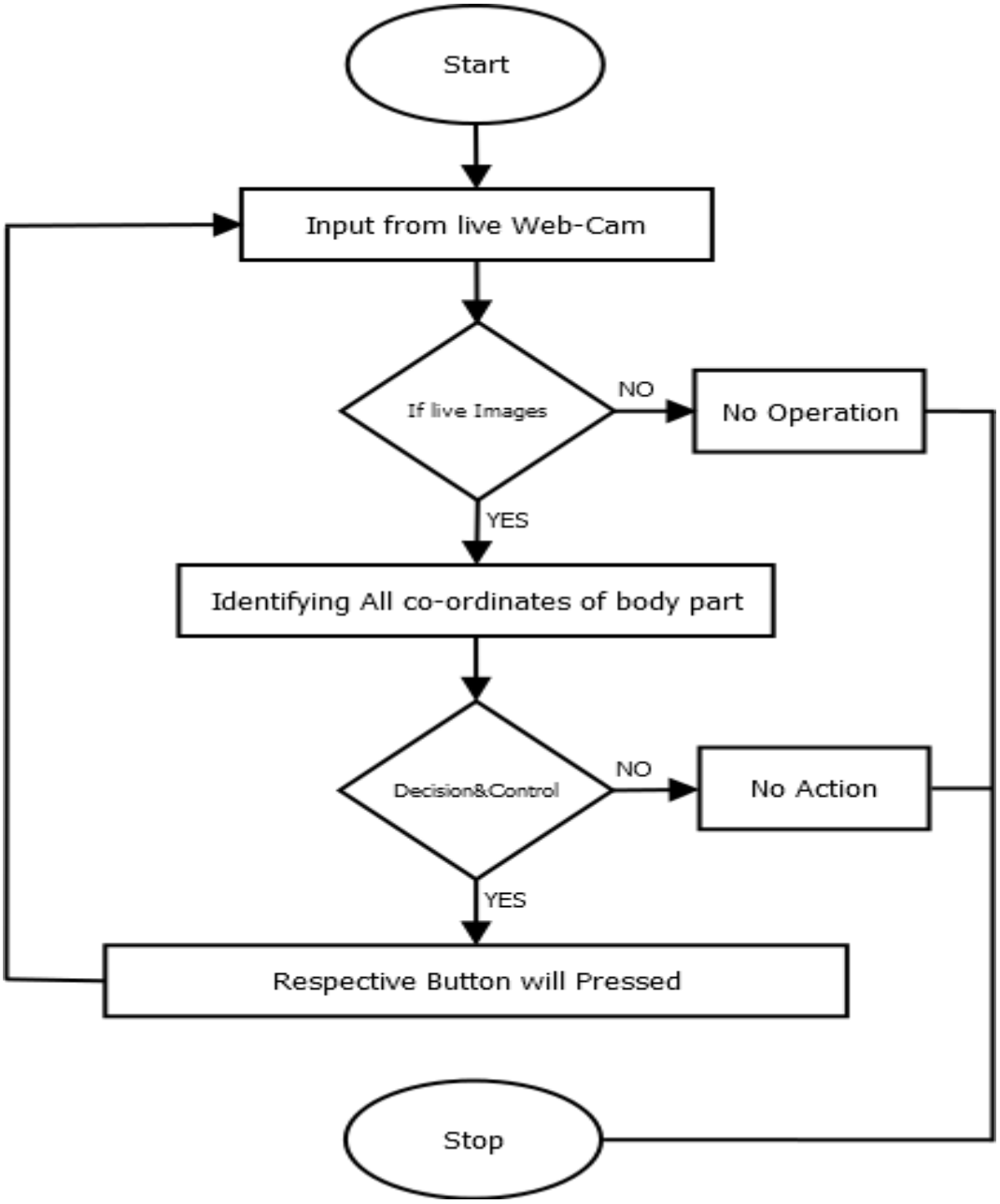


Fig 4.5 System Flow Diagram

d) Use Case Diagram:

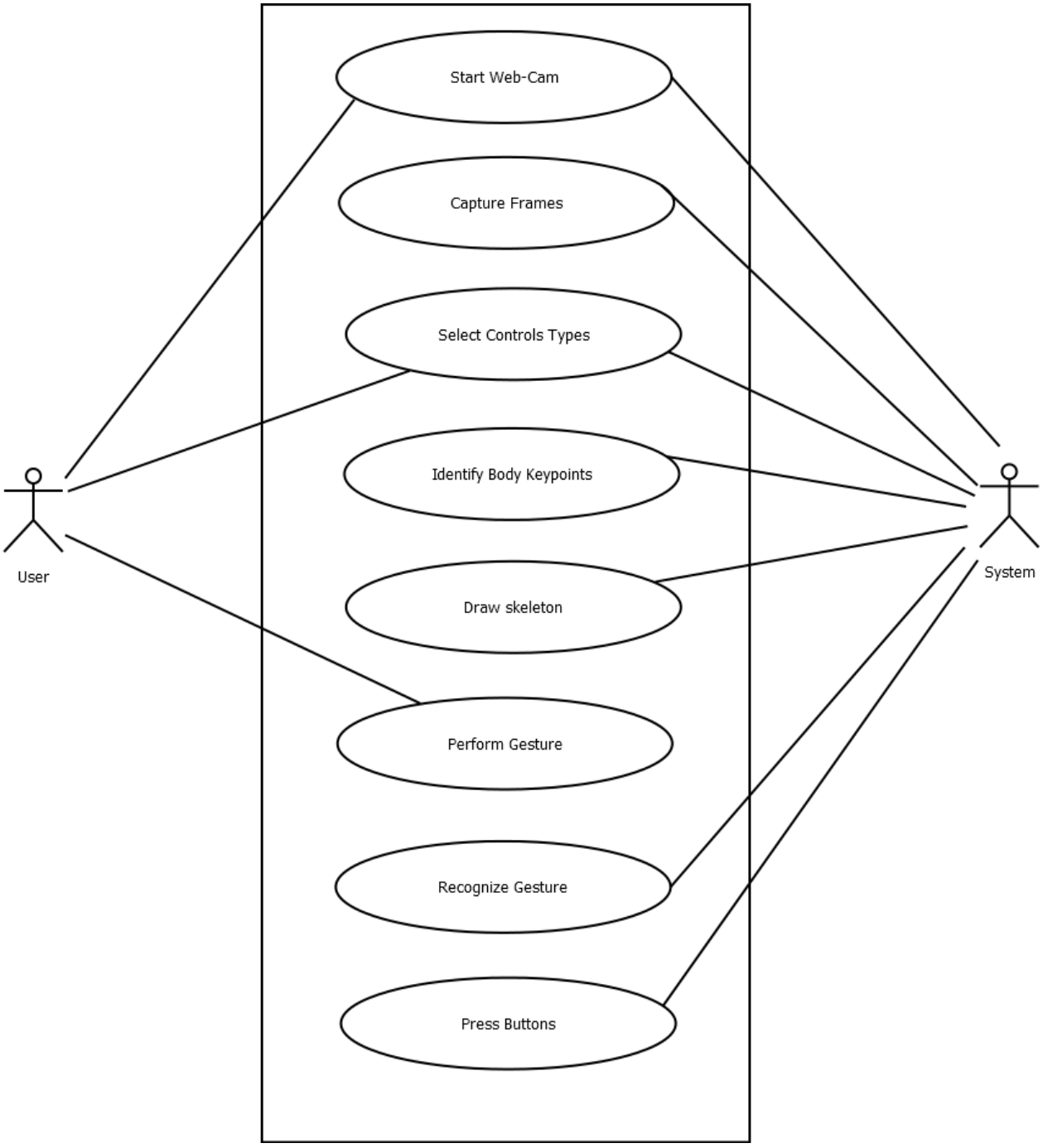


Fig 4.6 Use case diagram

Implementations

5. Implementation

a) Detailed Description of Methods

1. PoseNet() method:

We have used the PoseNet() method from ml5 in JavaScript. PoseNet is a machine learning model which is used for Real-time Human Pose Estimation. PoseNet will roughly calculate either one or multiple poses, meaning there is a version of the algorithm that may detect only 1 person in an image/video and therefore the other version that may detect multiple persons in an image/video. PoseNet returns an object of 17 body points. Each object is assigned a unique part like right eye, left eye, nose, right elbow, left elbow, etc. Each object contains x and y coordinates of a selected part and score of it.

It takes the input as a processed camera image and offers the output as information about the key points. The key points which are detected within the output are indexed by an element ID, with a confidence score in range 0.0 to 1.0. This score indicates the probability that at which particular location the key point exists.

PoseNet are mostly used to roughly calculate either one pose or multiple poses, which means there's a version of the algorithm that may detect just one person in an image/video and one version that may detect multiple persons in an image/video. Why are there two versions? The one person pose detector is quicker and simpler but requires just one subject present within the image (more thereon later). We cover the single-pose first because it's easier to follow.

At a top most level pose estimation happens in two phases:

- An input RGB image is provided to a convolutional neural network.
- Either one or multi pose decoding algorithm is used for decoding poses, pose confidence scores, key point positions, and key point confidence scores from the model outputs.
- Pose — at the top most level, PoseNet will return a pose object that contains an inventory of key points and an instance-level confidence score for every detected person.
- Pose confidence score — It determines the confidence within the estimation of a pose. Its range is between 0.0 to 1.0. It will hide the poses that don't seem to be deemed strong enough.
- Key point — a part of a person's pose that's calculated, like the nose, right ear, left knee, right foot, etc. It contains both an edge and a key point confidence score.

The various body joints detected by the PoseNet model are tabulated below:

Id	Part
0	nose
1	leftEye
2	rightEye
3	leftEar
4	rightEar
5	leftShoulder
6	rightShoulder
7	leftElbow
8	rightElbow
9	leftWrist
10	rightWrist
11	leftHip
12	rightHip
13	leftKnee
14	rightKnee
15	leftAnkle
16	rightAnkle

PoseNet currently detects 17 key points illustrated in the following diagram:

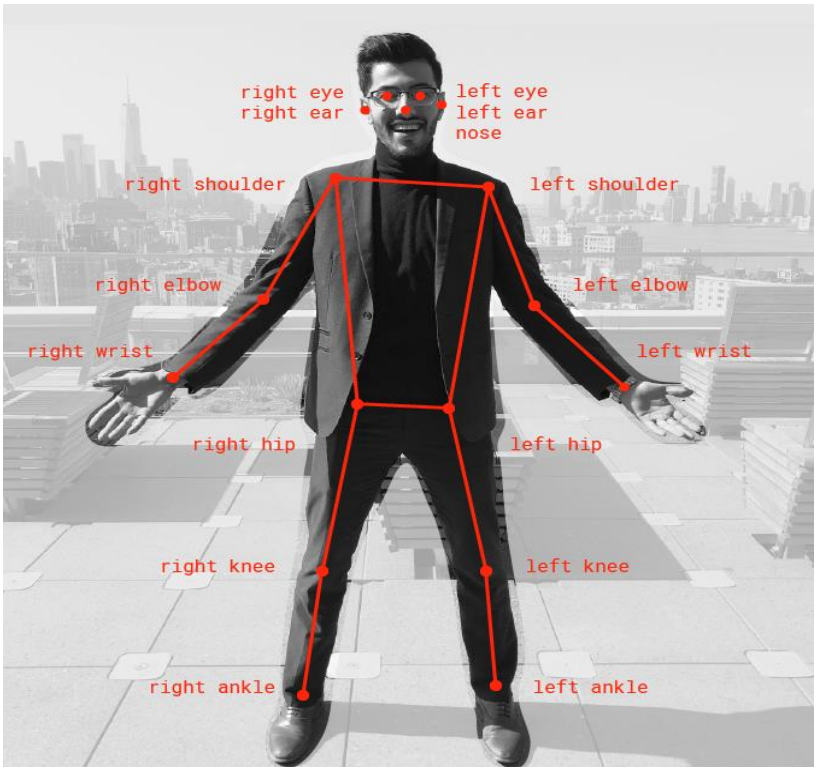


Fig 5.1 keypoints identified by model

Sample key point output of nose:

```
{ // pose #1
"score": 0.42985695206067,
"keypoints": [
{ // nose
"position": {
"x": 126.09371757507,
"y": 97.861720561981
},
"score": 0.99710708856583
},
```

b) Methodology

1. Game controls:

   There is main 5 controls in car racing game

- Forward
- Backward
- Left
- Right
- Handbrake

a) Forward Movement:

Steps:

1. We collect x and y coordinates of the right hand and left hand.

2. Then we find distance between them.

3. If distance is more than 110, car will go forward by pressing button W

Fig 5.b.1 Gesture to move forward

b) Backward Movement:

1. We collect x and y coordinates of the right hand and left hand.
2. Then we find distance between them
3. If distance is less, car will go backward by pressing button S

Fig 5.b.2 Gesture to move back

c) Right Movement:

1. We collect x and y points of the right hand and left hand.
2. Then we find slope between them
3. If slope is negative, car will go in right by pressing button D

Fig 5.b.3 Gesture to move right

d) Left Movement:

1. We collect x and y points of the right hand and left hand.

2. Then we find slope between them

3. If slope is positive, car will go in left by pressing button A

Fig 5.b.1 Gesture to move left

e) Handbrake:

1. We collect the y coordinates of the right hand.

2. Then we take y axis of both hands

3. If the hands are down more than 350. Then we apply handbrake by pressing button Space Bar

Fig 5.b.1 Gesture to apply handbrakes

Integration and Testing

6. Testing

Testing Performed: -

1. Test Cases for GUI :

MODULE NAME	Home Page						
PREPARED BY	Alamgir Sayyad						
EXECUTION DATE	18/03/21						
EXECUTED BY	Aniruddha Lokare						
SCENARIO ID	1	SCENARIO DESCRIPTION		Check Home Page working.			
S.NO	TEST CASE ID	TEST CASE DESCRIPTION	PRECONDITION	EXPECTED RESULT	POSTCONDITION	ACTUAL RESULT	STATUS
1	TC_HP_01	Checking all components from UI- WASD control Button	Application should be open	Keys-WASD.py to be executed	Redirect to index.html page	Keys-WASD.py to be executed	PASS
2	TC_HP_02	Checking all components from UI-ARROW control button	Application should be open	Keys-ARROW.py to be executed	Redirect to index.html page	Keys-WASD.py to be executed	PASS

2. Test Cases for HTML Files :

MODULE NAME	Index.html						
PREPARED BY	Alamgir Sayyad						
EXECUTION DATE	25/03/21						
EXECUTED BY	Shahid Mujawar						
SCENARIO ID	2	SCENARIO DESCRIPTION		Check ' Index.html ' UI page.			
S.NO	TEST CASE ID	TEST CASE DESCRIPTION	PRECONDITION	EXPECTED RESULT	POSTCONDITION	ACTUAL RESULT	STATUS
1	TC_IP_01	Checking html file components-popup instruction	Selected any of the control style 1)WASD Control 2)Arrow Control	HTML page with background image and pop-up instructions should be loaded.	Proceed button should be visible	HTML page with proper background image and pop-up instructions are loaded.	PASS
2	TC_IP_02	Checking html file components-proceed button	Popup instruction should be visible	Redirect to model.html page	Redirect to model.html page	Redirected to model.html page	PASS

3. Test Cases for Project Model :

MODULE NAME	Project Model						
PREPARED BY	Alamgir Sayyad						
EXECUTION DATE	30/03/21						
EXECUTED BY	Aniket Hiremath						
SCENARIO ID	3	SCENARIO DESCRIPTION		Verify project model working.			
S.NO	TEST CASE ID	TEST CASE DESCRIPTION	PRECONDITION	EXPECTED RESULT	POSTCONDITION	ACTUAL RESULT	STATUS
1	TS_M_01	Checking all js files are included or not.	Should be redirected from clicking proceed button in index.html	JS model file and dependent model files should be load.	Model should be ready	Included model dependent files and model should be loaded	PASS
2	TS_M_02	Checking working of the model.	Included model dependent files.	Web camera starts capturing frames.	Skeleton should be detected by the model	Web camera started capturing frames	PASS
3	TS_M_03	Detecting the 17 key bodypoints.	Web camera should be started capturing frames.	Model should detect and draw skeleton using all 17 keypoints from input image frames.	Score should be more than 20%	Model detected and draw skeleton using all 17 keypoints from input image frames.	PASS
4	TS_M_04	Detecting the 17 key bodypoints when no person is detected	Web camera should be started capturing frames.	No skeleton should be drawn.	Score should be more than 20%	No skeleton drawn	PASS

4. Test Cases for ‘WASD’ Controls :

MODULE NAME	Gesture - WASD						
PREPARED BY	Alamgir Sayyad						
EXECUTION DATE	06/04/21						
EXECUTED BY	Shahid Mujawar						
SCENARIO ID	4	SCENARIO DESCRIPTION		Verification of all hand gestures detection with 'WASD controls' and their respective actions.			
S.NO	TEST CASE ID	TEST CASE DESCRIPTION	PRECONDITION	EXPECTED RESULT	POSTCONDITION	ACTUAL RESULT	STATUS
1	TS_GC_W_01	Test inputed hand gesture for action of pressing W button	Hands should be in straight horizontal line and distance should be greather than 120px TEST:- dist=130.70	Press Keyboard Button 'W'	Action after pressing button W should be performed in game	Keyboard Button 'W' Pressed	PASS
2	TS_GC_W_02	Test inputed hand gesture for action of pressing W button	Hands should be in straight horizontal line and distance should be greather than 120px TEST:- dist=200.31px	Press Keyboard Button 'W'	Action after pressing button W should be performed in game	Keyboard Button 'W' Pressed	PASS
3	TS_GC_A_01	Test inputed hand gesture for action of pressing A button	Slope should be greater than 0.5 TEST: -slope=0.8	Press Keyboard Button 'A'	Action after pressing button A should be performed in game	Keyboard Button 'A' Pressed	PASS
4	TS_GC_A_02	Test inputed hand gesture for action of pressing A button	Slope should be greater than 0.5 TEST: -slope=1.3	Press Keyboard Button 'A'	Action after pressing button A should be performed in game	Keyboard Button 'A' Pressed	PASS
5	TS_GC_S_01	Test inputed hand gesture for action of pressing S button	Hands should be in straight vertical line and distance should be less than 110px TEST:- Slope=101	Press Keyboard Button 'S'	Action after pressing button S should be performed in game	Keyboard Button 'S' Pressed	PASS
6	TS_GC_S_02	Test inputed hand gesture for action of pressing S button	Hands should be in straight vertical line and distance should be less than 110px TEST:- Slope=95.3	Press Keyboard Button 'S'	Action after pressing button S should be performed in game	Keyboard Button 'S'	PASS
7	TS_GC_D_01	Test inputed hand gesture for action of pressing D button	Slope should be less than 0.5 TEST:- slope = -0.71px	Press Keyboard Button 'D'	Action after pressing button D should be performed in game	Keyboard Button 'D' Pressed	PASS
8	TS_GC_D_02	Test inputed hand gesture for action of pressing D button	Slope should be less than 0.5 TEST:- slope = -0.6px	Press Keyboard Button 'D'	Action after pressing button D should be performed in game	Keyboard Button 'D' Pressed	PASS
9	TS_GC_B_02	Test inputed hand gesture for action of pressing Space button	Y-axis of both right hand left hand should be more than(>) 350px	Press Keyboard Button 'Space'	Action after pressing button Space should be performed in game	Keyboard Button 'Space' Pressed	PASS

5.Test Cases for Arrow Control :

MODULE NAME	Gesture - Arrow						
PREPARED BY	Alamgir Sayyad						
EXECUTION DATE	10/04/21						
EXECUTED BY	Aniket Hiremath						
SCENARIO ID	5	SCENARIO DESCRIPTION	Verification of all hand gestures detection with 'Arrow controls' and their respective actions.				
S NO	TEST CASE ID	TEST CASE DESCRIPTION	PRECONDITION	EXPECTED RESULT	POSTCONDITION	ACTUAL RESULT	STATUS
1	TS_GC_UP_01	Test inputed hand gesture for action of pressing Arrow up button	Hands should be in straight horizontal line and distance should be greather than 120px TEST:- dist=130.70	Press Keyboard Button 'Arrow up'	Action after pressing button Arrow up should be performed in game	Keyboard Button 'Arrow up' Pressed	PASS
2	TS_GC_UP_02	Test inputed hand gesture for action of pressing Arrow up button	Hands should be in straight horizontal line and distance should be greather than 120px TEST:- dist=200.31px	Press Keyboard Button 'Arrow up'	Action after pressing button Arrow up should be performed in game	Keyboard Button 'Arrow up' Pressed	PASS
3	TS_GC_UP_03	Test inputed hand gesture for action of pressing Arrow down button	Hands should be in straight vertical line and distance should be less than 110px	Press Keyboard Button 'Arrow down'	Action after pressing button Arrow down should be performed in game	Keyboard Button 'Arrow down' Pressed	PASS
4	TS_GC_UP_04	Test inputed hand gesture for action of pressing Arrow down button	Hands should be in straight vertical line and distance should be less than 110px	Press Keyboard Button 'Arrow down'	Action after pressing button Arrow down should be performed in game	Keyboard Button 'Arrow down' Pressed	PASS
5	TS_GC_UP_05	Test inputed hand gesture for action of pressing Arrow left button	Slope should be greater than 0.5 TEST:-slope=1.0	Press Keyboard Button 'Arrow left'	Action after pressing button Arrow left should be performed in game	Keyboard Button 'Arrow left' Pressed	PASS
6	TS_GC_UP_06	Test inputed hand gesture for action of pressing Arrow left button	Slope should be greater than 0.5 TEST:-slope=0.8	Press Keyboard Button 'Arrow left'	Action after pressing button Arrow left should be performed in game	Keyboard Button 'Arrow left' Pressed	PASS
7	TS_GC_UP_07	Test inputed hand gesture for action of pressing Arrow right button	Slope should be less than -0.5 TEST:-slope=-0.8	Press Keyboard Button 'Arrow right'	Action after pressing button Arrow right should be performed in game	Keyboard Button 'Arrow right' Pressed	PASS
8	TS_GC_UP_08	Test inputed hand gesture for action of pressing Arrow right button	Slope should be less than 0.5 TEST:-slope=-0.9	Press Keyboard Button 'Arrow right'	Action after pressing button Arrow right should be performed in game	Keyboard Button 'Arrow right' Pressed	PASS
9	TS_GC_UP_09	Test inputed hand gesture for action of pressing space button	Y-axis of both right hand left hand should be more than(>) 350px	Press Keyboard Button 'Space'	Action after pressing button Space bar should be performed in game	Keyboard Button 'Space' Pressed	PASS

Performance Analysis

7. Performance analysis

1. Time: -

Methods	Time Required
Opening application	1 sec
Loading Model	2 sec
Gesture to action	0.002 sec

2. Accuracy: - For below scores, if the score is 0.99 means 99% model has detected that it is a nose.

Keypoints: Array (17)

0: {score: 0.9993923902511597, part: "nose", position: {…}}

1: {score: 0.999896764755249, part: "leftEye", position: {…}}

2: {score: 0.9998162984848022, part: "rightEye", position: {…}}

3: {score: 0.8612481355667114, part: "leftEar", position: {…}}

4: {score: 0.847524881362915, part: "rightEar", position: {…}}

5: {score: 0.9947163462638855, part: "leftShoulder", position: {…}}

6: {score: 0.982994556427002, part: "rightShoulder", position: {…}}

7: {score: 0.5718628168106079, part: "leftElbow", position: {…}}

8: {score: 0.8672078251838684, part: "rightElbow", position: {…}}

9: {score: 0.020063506439328194, part: "leftWrist", position: {…}}

10: {score: 0.05249423533678055, part: "rightWrist", position: {…}}

11: {score: 0.014359373599290848, part: "leftHip", position: {…}}

12: {score: 0.017596164718270302, part: "rightHip", position: {…}}

13: {score: 0.0014035503845661879, part: "leftKnee", position: {…}}

14: {score: 0.0033228739630430937, part: "rightKnee", position: {…}}

15: {score: 0.0008590858778916299, part: "leftAnkle", position: {…}}

16: {score: 0.0009565502987243235, part: "rightAnkle", position: {…}}

Applications

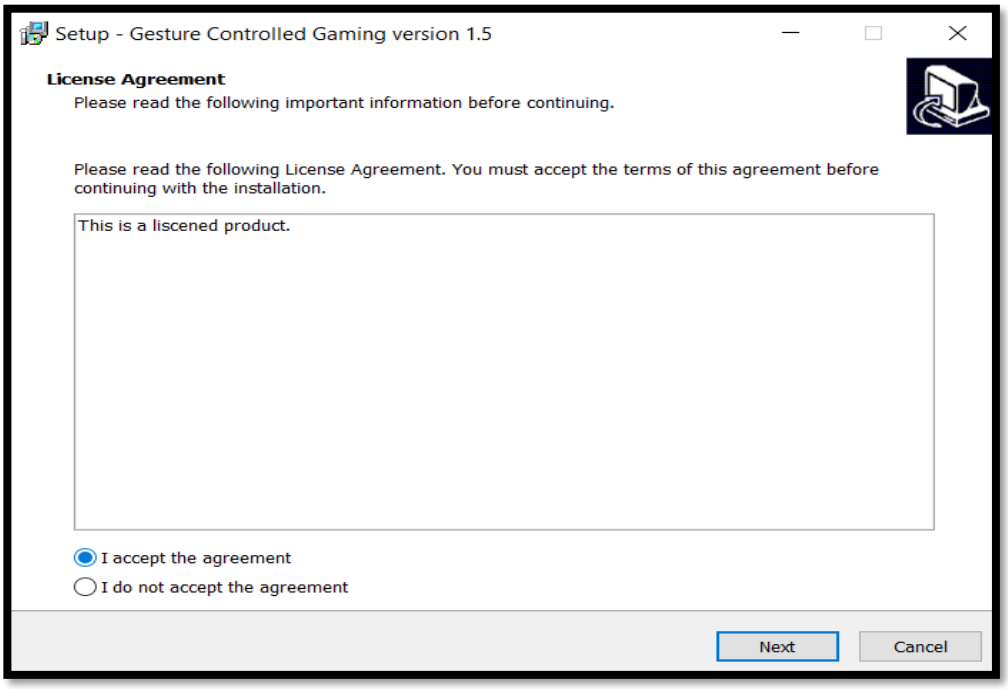
8. Applications

- Can be used on any type of racing games which can be the online browser game or installed game.
- Can be used to play other category games like Arcade, shooting, Action games.
- Apart from the gaming we can use this application to control music players or video players too.
- Also, it can be a solution in case of failure of mouse, we can control basic actions in OS with the gestures, like opening files, selecting files etc.

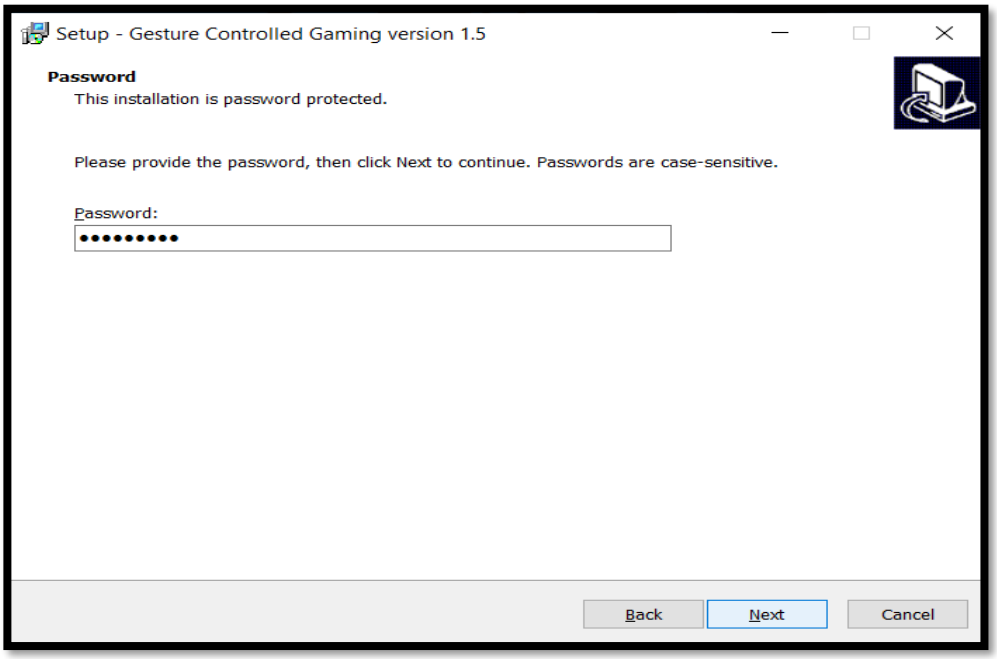
Installation Guide and User Manual

9. Installation Guide and User Manual

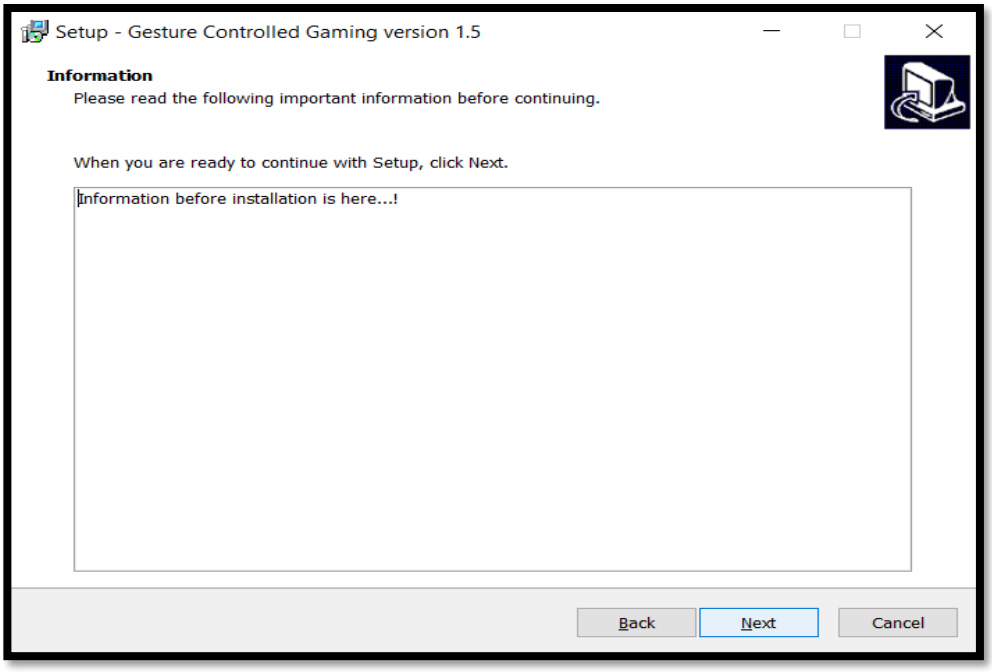
1) After downloading the setup of gesture-controlled gaming open the setup by double click you will redirect to this window. Click on ***I accept the agreement*** and then click on the ***Next.***



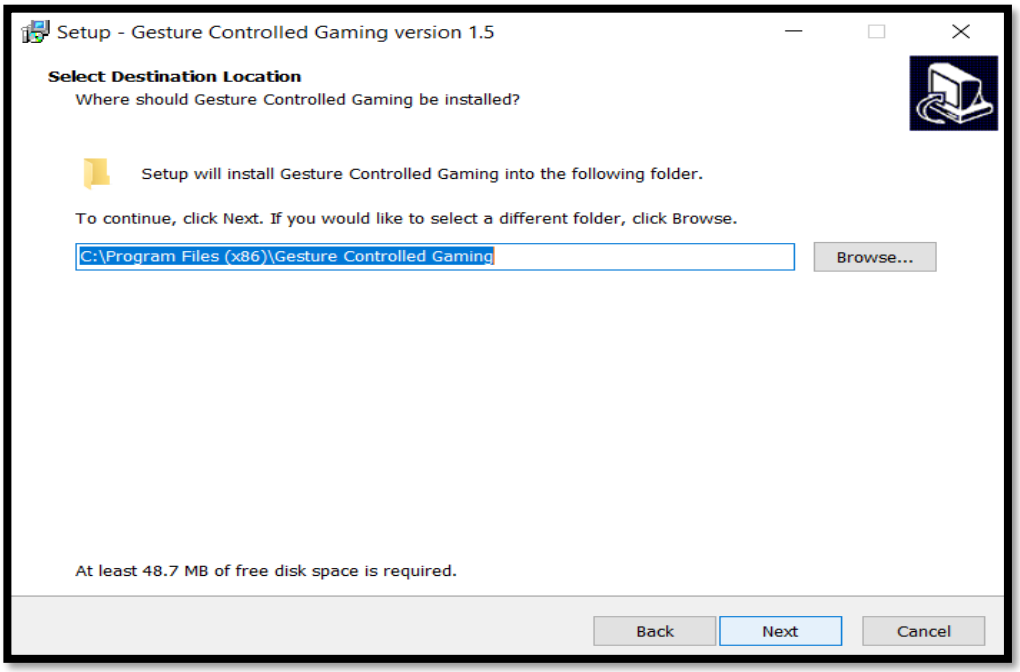
2) After clicking on Next, you will see the window in which you have to type the password by us. And then click on the ***Next***.



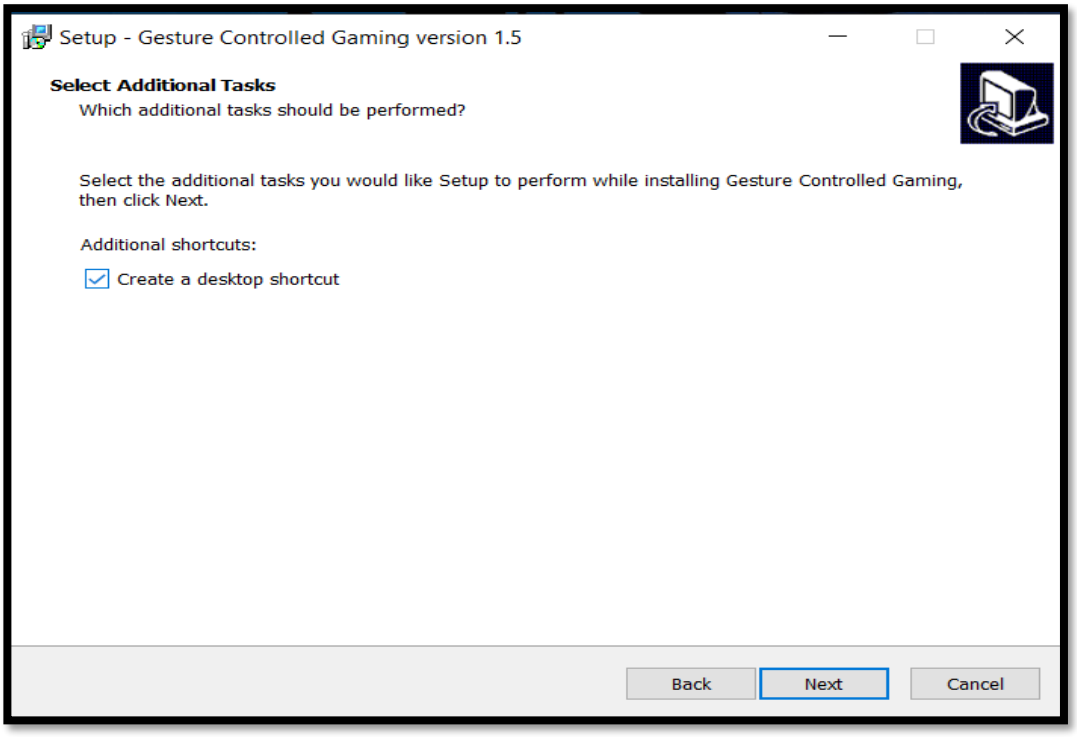
3) After you click on Next, u will see this window and you have to again click on ***Next.***



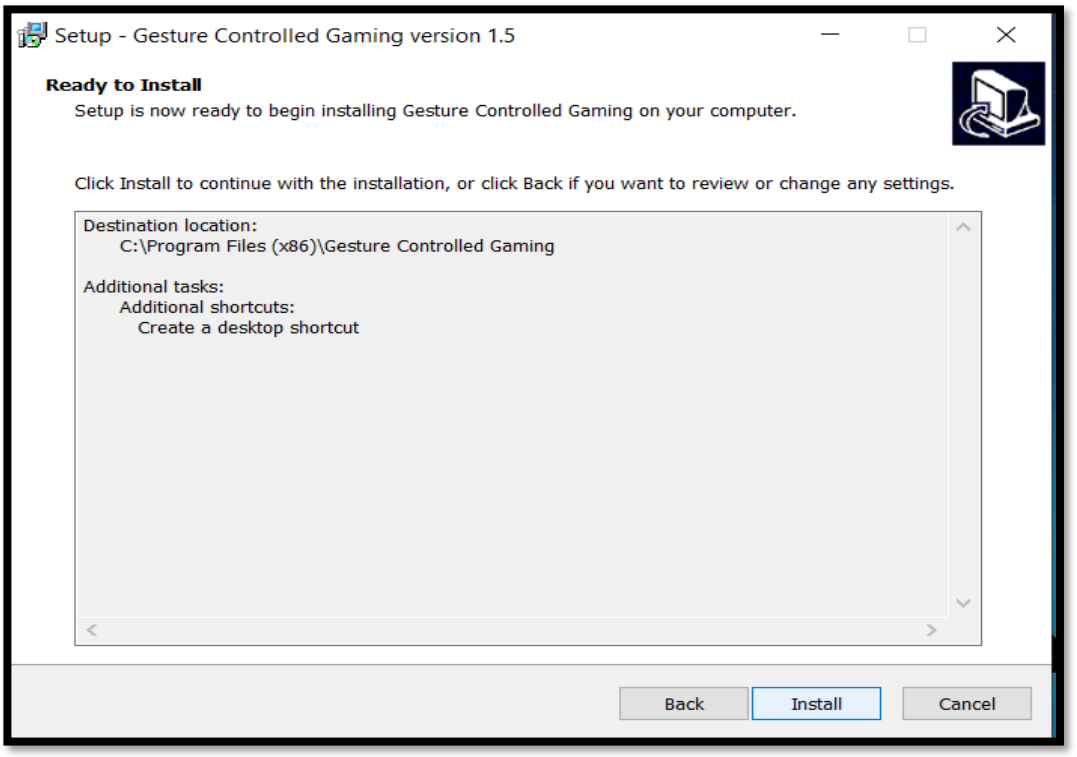
4) Now select the path where you want to install the setup and click on ***Next.***



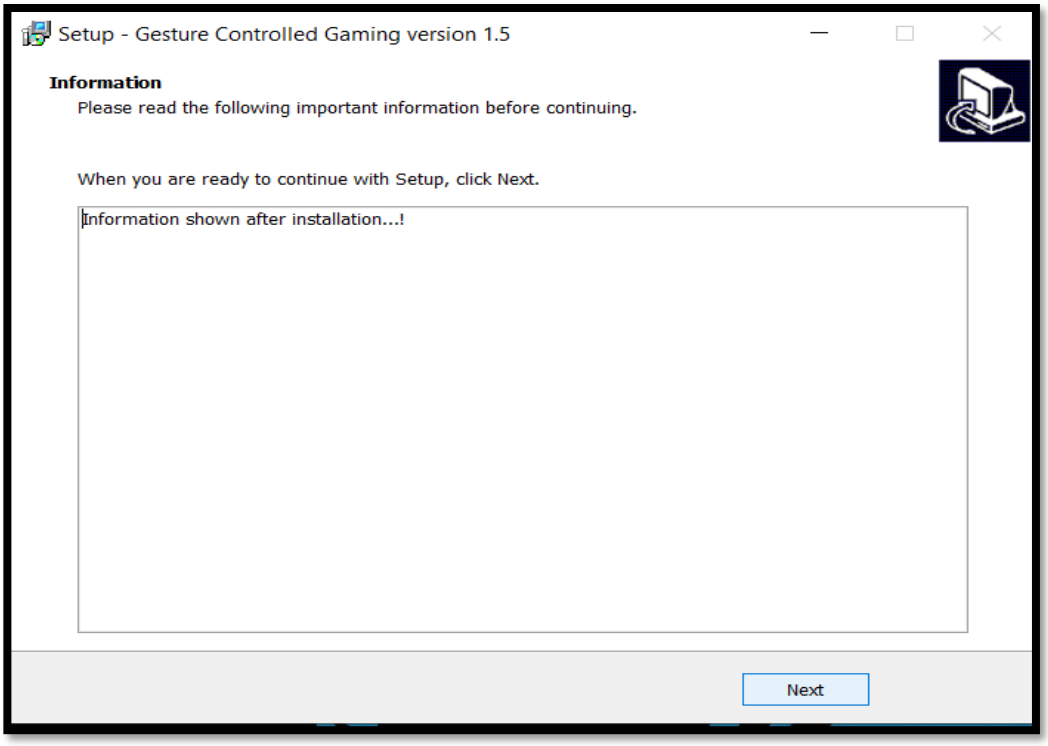
5) Now if you want to create a desktop shortcut then check the check box(recommended), and then click on the ***Next.***



6) After you click on Next you will redirect to the above window, here click on ***Install.***



7) Click on ***Next.***



8) And the last step is to click on ***Finish*** and installation is done.

Pre-requisites: -

1. Python should be installed, if not installed, download from below link:
https://www.python.org/downloads/

2. Install Keyboard library
Open command prompt and run the command below
pip install keyboard

3. Install eel library
Open command prompt and run the command below
pip install eel

Cost Estimation

10. Project Cost

a. Total Cost to run project:

Hardware	Cost
Computer/Laptop	Rs. 40000/-
Webcam if does not have Integrated Web camera	Rs. 1500/-
Total	Rs. 55000/-

ETHICS

11. Declaration of Ethics

As a Technology & Engineering Student, I believe it's unethical to,

1. Surf the web for private interest and non-class related purposes during classes.
2. Make a duplicate of software for private or commercial use.
3. Make a replica of software of an admirer.
4. Loan CDs of software to friends.
5. Download pirated software from the web.
6. Distribute pirated software from the web.
7. Buy software with a one user license and then install it on multiple Computers.
8. Share a pirated copy of software.
9. Install a pirated copy of software.

Reference

1. https://ml5js.org/reference/api-posenet/
2. https://www.fritz.ai/pose-estimation/
3. https://www.tensorflow.org/lite/examples/pose_estimation/overview
4. https://heartbeat.fritz.ai/human-pose-estimation-using-tensorflows-posenet-model-e5770f0a0a31
5. https://medium.com/roonyx/pose-estimation-and-matching-with-tensorflow-lite-posenet-model-ea2e9249abbd
6. https://sujalpaudel.medium.com/posenet-what-and-why-a08a3402cd5a