PROJECT BASED EXPERIENTIAL LEARNING PROGRAM

(NALAIYA THIRAN)

A GESTURE-BASED TOOL FOR STERILE BROWSING OF RADIOLOGY IMAGES

A PROJECT REPORT

Submitted by

VAISHNAVI PARVATHY N (111719106167)

SWETHA E (111719106155)

SOWMIYA M (111719106148)

SINDHUJA C (111719106147)

TEAM ID : PNT2022TMID16219

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

R.M.K.ENGINEERING COLLEGE

(An Autonomous Institution)

R.S.M. Nagar, Kavaraipettai-601 206

NOVEMBER 2022

CONTENTS:

A Gesture-Based Tool for Sterile Browsing of Radiology Images

1.INTRODUCTION:

1.1 OVERVIEW:

In this project we use gestures to browse images obtained during radiology. Gestures refer to non verbal forms of communication made using hands.

Humans are able to recognize body and sign language easily. This is possible due to the combination of vision and synaptic interactions that were formed along brain development . In order to replicate this skill in computers, some problems need to be solved: how to separate objects of interest in images and which image capture technology and classification techniques are more appropriate, among others.

In this project Gesture based Desktop automation, First the model is trained pre-trained on the images of different hand gestures, such as showing numbers with fingers as 1,2,3,4. This model uses the integrated webcam to capture the video frame. The image of the gesture captured in the video frame is compared with the Pre-trained model and the gesture is identified. If the gesture predicts is 0 - then images is converted into rectangle, 1 - image is Resized into (200,200), 2 - image is rotated by -45°, 3 - image is blurred, 4 - image is Resized into (400,400), 5 - image is converted into grayscale etc.

1. Defining our classification categories
2. Collect training images
3. Train the model
4. Test our model

1.2 PURPOSE:

The main purpose of our project is to create an application in which humans can interact with a system without any keyboard or mouse to obtain a sterile based interaction only by using Camera, So we develop an application which only needs a user’s hand sign to give an input through the camera to the system.

2.LITERATURE SURVEY:

1. **Computer Vision based Hand Gesture Recognition**

2.1.1. EXISTING PROBLEMS:

Sensor based recognition collects the gesture data by using one or more different types of sensors. These sensors are attached to the hand which record to get the position of the hand and then collected data is analysed for gesture recognition. Data gloves are an example of sensor based gesture recognition other sensors used were Wii controller, EMG sensors, accelerometer sensors, etc. Sensor based recognition has certain limitations. First of all it requires a proper hardware setup which is very expensive. Secondly, it hinders the natural movement of the hand. So to overcome the limitation of sensor based recognition, vision based techniques came into existence.

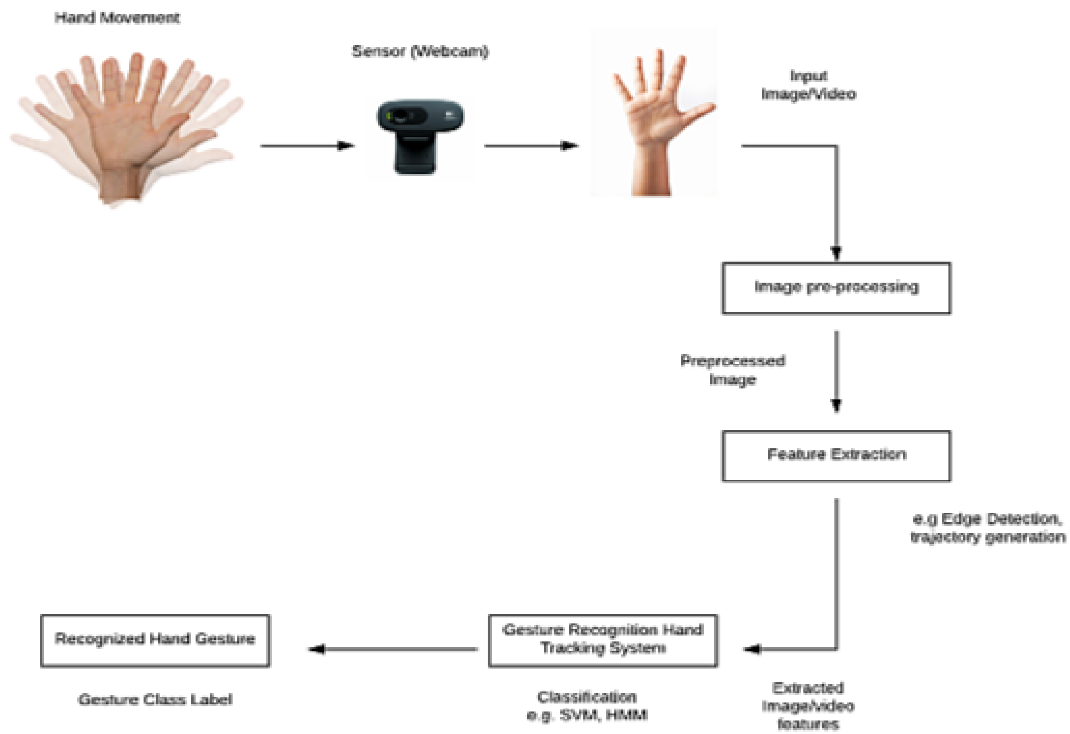
2.1.2. REFERENCES:

https://www.researchgate.net/publication/335803643_Computer

_Vision_based_Hand_Gesture_Recognition_A_Survey

2.1.3. PROBLEM STATEMENT DEFINITION:

Vision based techniques make use of cameras to capture the image for hand gestures. Vision based recognition makes use of many image processing algorithms to get hand posture information and movement of hand. This approach recognizes gestures from shapes, orientations, contours, and colour or motions features of a hand.Colored markers are an example of vision based recognition. But the vision based recognition also has some limitations that it is affected by illumination changes and cluttered backgrounds.



2. <u>Hand Gestures Recognition Using Radar Sensors for Human-Computer-Interaction</u>

2.2.1. EXISTING PROBLEMS:

Computing technology has become embedded in every aspect of our daily lives and man–machine interaction is becoming inevitable. It is widely believed that computer and display technology will keep on progressing further. A gateway which allows humans to communicate with machines and computers is known as the human–computer interface (HCI) . Keyboard and mouse and touch-screen sensors are the traditional HCI approaches. However, these approaches are becoming a bottleneck for developing user-friendly interfaces.

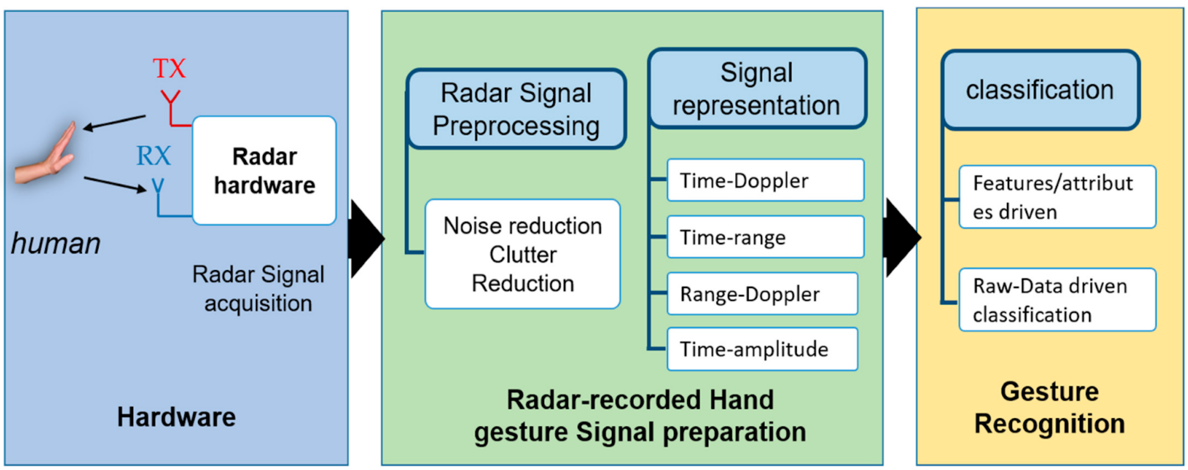
2.2.2. REFERENCE LINK:

https://www.mdpi.com/2072-4292/13/3/527/htm

2.2.3. PROBLEM STATEMENT DEFINITION:

A work presented by Ahmed and Cho (2020) demonstrated a performance comparison of different Deep Convolutional Neural Network (CNN)-based deep-learning algorithms using multiple radars. In addition to that, hand gesture recognition through radar technology also found application in the operating room to assist medical staff in processing and manipulating medical images.

Based on the transmitted signal, short-range radar sensors used for HGR can broadly be categorised as pulsed radar and continuous-wave (CW) radar. This categorization has been adopted previously in several radar related review articles for applications other than HGR. Pulsed radar, such as Ultra-Wideband Impulse-Radar (UWB-IR), transmits short duration pulses, whereas continuous-wave radar, such as Frequency Modulated Continuous Wave (FMCW) radar, transmits and receives a continuous wave. Both these radars are widely used for HGR purposes.



3.Gesture-controlled image system positioning for minimally invasive interventions

2.3.1. EXISTING PROBLEMS:

The number of percutaneous interventions is increasing due to the demographic change. Patients are older and more people get cancer. Especially older and multi-morbid patients often cannot be cured with invasive surgery, i.e. by resection of the lesion. With the help of imaging devices, such as CT scanners, the radiologist brings energy applicators into the tumour region with high precision percutaneously. However, most CT systems were developed as diagnostic devices and thus lack assistance and interaction concepts essential for the interventional workflow. Especially for flat panel CT's to acquire images to visualise the instrument and risk structure locations, the radiologist must position the X-Ray tube and the detector to the dedicated angle for the radiography. Observations revealed that the current workflow is not optimal, due to sterility issues that arise with the use of haptic buttons or touchscreens covered with a sterile drape . A lot of research effort has been done in interaction with visualisation and controlling of medical image viewers or registration of images.

2.3.2. REFERENCE:

Hatscher, B., Mewes, A., Pannicke, E., Kägebein, U., Wacker, F.,Hansen, C., Hensen, B. (2020). Touchless scanner control to support MRI-guided interventions. *International journal of computer assisted radiology and surgery*, *15*(3), 545–553. https://doi.org/10.1007/s11548-019-02058-1

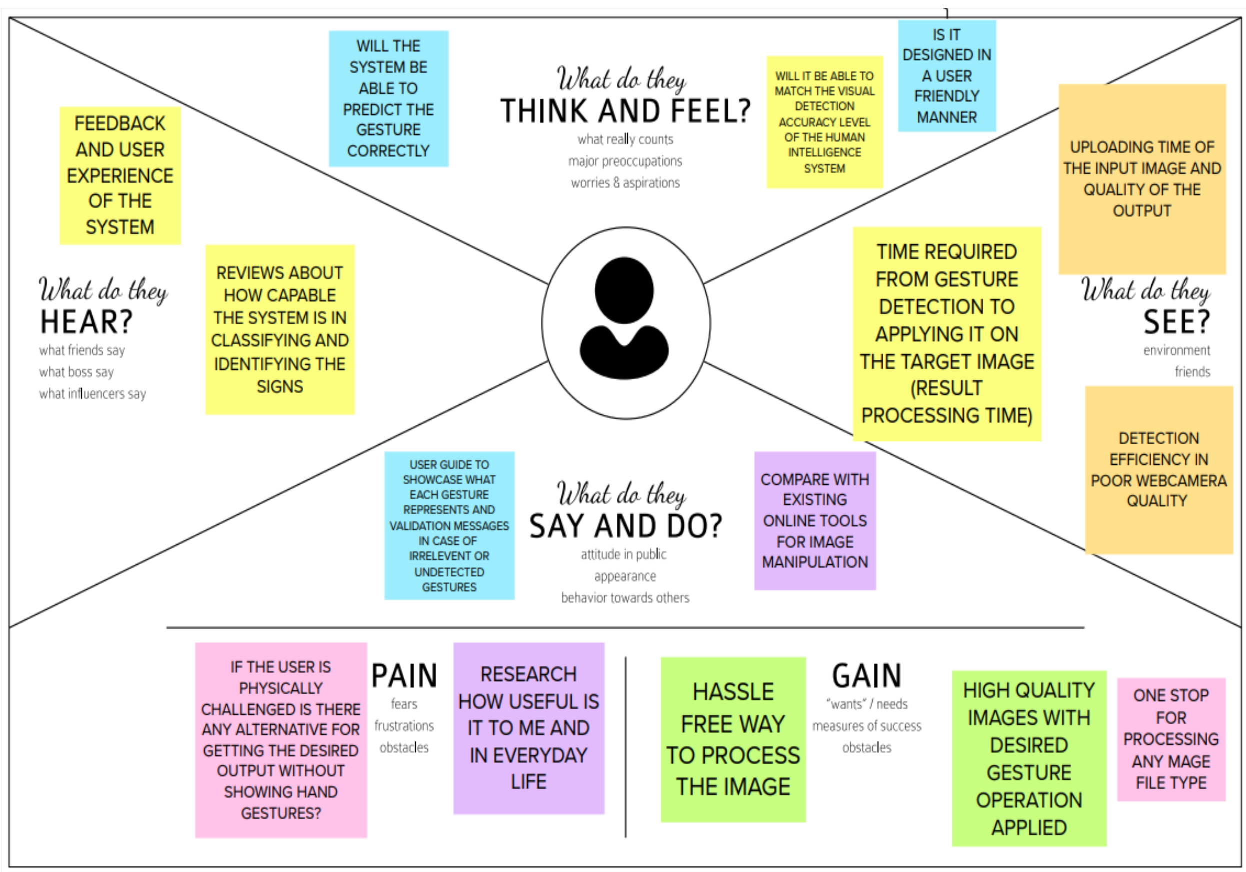
2.3.3.PROBLEM STATEMENT DEFINITION:

In this work a gesture interaction concept that translates touchless hand gestures into commands for special-purpose radiography imaging and a graphical user interface (GUI) for visualisation purposes is presented. With the help of this system, the interventionalist is capable of positioning the X-Ray tube and detector to a dedicated angle without leaving the sterile area and without breaking sterility.

This should improve the workflow of image-guided interventions and, consequently, save time and costs and improve the outcome of treated patients. The proposed interaction concepts were evaluated in a user study with 10 participants regarding error rate, task completion time, and usability.

3.IDEATION AND PROPOSED SOLUTION:

3.1.Empathy Map Canvas:



3.2 Ideation:

IDEA 1:

In this project we use gestures to browse images obtained during radiology. Gestures refer to non verbal forms of communication made using hands. A major challenge involved in this process is to provide doctors with efficient, intuitive, accurate and safe means of interaction without affecting the quality of their work. Keyboards and pointing devices, such as a mouse, are today's principal method of human—computer interaction. However, the use of computer keyboards and mice by doctors and nurses in intensive care units (ICUs) is a common method for spreading infections. Humans can recognize body and sign language easily. This is possible due to the combination of vision and synaptic interactions that were formed along brain development.

- User interacts with the UI (User Interface) to upload the image as input.
- Depending on the different gesture inputs different operations are applied to the input image.
- Once the model analyses the gesture, the prediction with operation

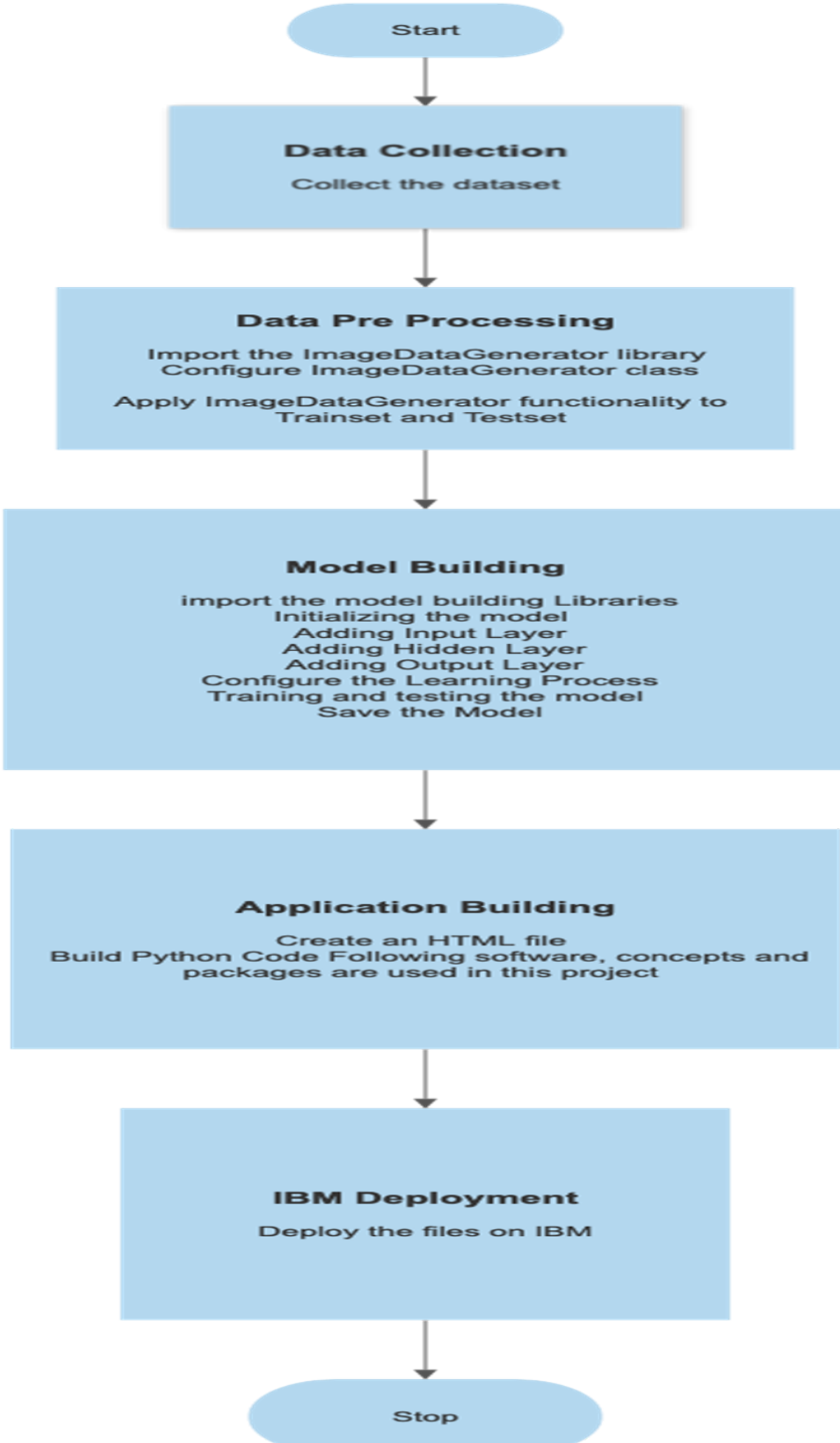
applied on the image is showcased on the UI.

To accomplish this, we have to complete all the activities and tasks listed below:

- Data Collection.
  - Collect the dataset or Create the dataset
- Data Pre processing
  - Import the ImageDataGenerator library
  - Configure ImageDataGenerator class
  - Apply ImageDataGenerator functionality to Train Set and Test Set
- Model Building
  - Import the model building Libraries
  - Initialising the model
  - Adding Input Layer
  - Adding Hidden Layer
  - Adding Output Layer
  - Configure the Learning Process
  - Training and testing the model
  - Save the Model
- Application Building
  - Create an HTML file

○ Build Python Code Following software, concepts and packages are used in this project

- Anaconda navigator
- Python packages:
    - open anaconda prompt as administrator
    - Type “pip install TensorFlow” (make sure you are working on python 64 bit)
    - Type “pip install opencv-python”
    - Type “pip install flask”


Start
Data Collection
Collect the dataset
Data Pre Processing
Import the ImageDataGenerator library
Configure ImageDataGenerator class
Apply ImageDataGenerator functionality to Trainset and Testset
Model Building
import the model building Libraries
Initializing the model
Adding Input Layer
Adding Hidden Layer
Adding Output Layer
Configure the Learning Process
Training and testing the model
Save the Model
Application Building
Create an HTML file
Build Python Code Following software, concepts and packages are used in this project
IBM Deployment
Deploy the files on IBM
Stop

IDEA 2:

Hand-Gesture Based HCI Design

In recent years, computing technology has become embedded in every aspect of our daily lives and man–machine interaction is becoming inevitable. A gateway which allows humans to communicate with machines and computers is known as the human–computer interface (HCI) . Keyboard and mouse and touch-screen sensors are the traditional HCI approaches. However, these approaches are becoming a bottleneck for developing user-friendly interfaces . Contrary to this, human gestures can be a more natural way of providing an interface between humans and computers. In this approach we are using the short-range radars which have the ability to detect micro-movements with high precision and accuracy .

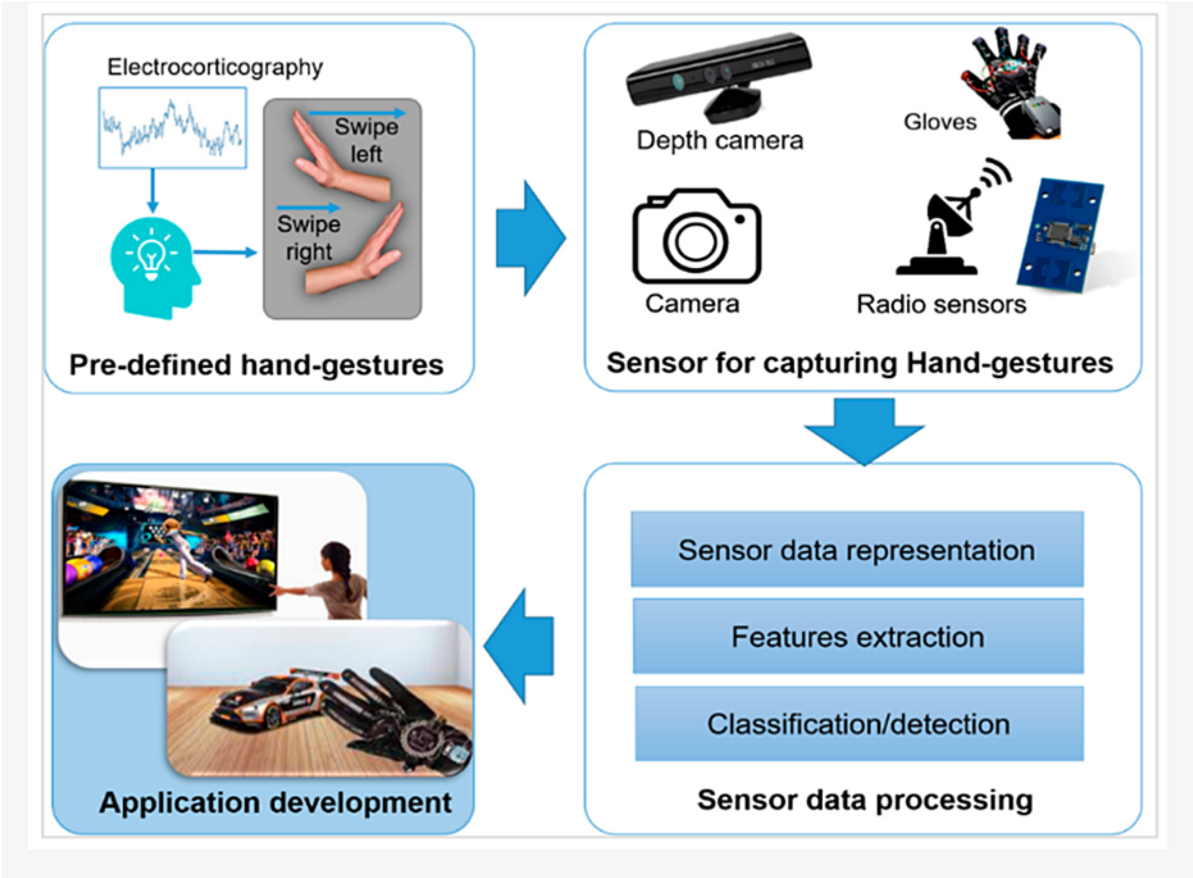
First, a neural spike is produced in the brain, which generates a signal that results in a voluntary motion of the hand. The brain signal corresponding to hand movement can be seen through electrocorticography . To detect the hand movements, several sensors exist, such as camera, depth-camera, and radio sensors. The signal at the output of these sensors is analysed using suitable algorithmic techniques to detect a predefined hand gesture. After successfully recognizing the desired hand-movements, these gesture-based systems can be used to classify the images based on radiology.

A wide range of sensors are available for acquiring signals against the performed hand gesture, and a radar sensor is used in our approach. Traditionally, optical sensors (camera), and wearable sensors (gloves) are widely used. These sensors can be classified as wearable and non-wearable. It can be seen that both types of technology possess their own strengths and weaknesses and can be selected according to the requirements of the application in consideration. Both the radar and the cameras provide a wireless interface for gesture recognition. Radar sensors have several benefits over camera-based recognition systems . Radar is not affected by lightning conditions and there are no related privacy issues. Users often do not feel comfortable being watched by a camera.

After data acquisition, the next step is processing the data and recognizing hand gestures. This includes data representation, useful features extraction, and classification. The classification can be performed by using signal-processing approaches,traditional machine learning or deep learning approaches.

Predefined hand gesture is performed by users in radar cross-sectional area.This signal is passed through a clutter reduction filter and the signal is represented in any of the mentioned signal representation schema. Finally,recognition is performed.

- Hand-gesture movement acquisition, where one of the available radar technologies is chosen;
- Pre-processing the received signal, which involves pre-filtering followed by a data formatting which depends on step 3. For example, the 1D, 2D, and 3D deep Convolutional Neural Network (DCNN) will, respectively, require data to be in a 1D, 2D or 3D shape;
- The final step of hand-gesture classification produces the various classification of the image. If the gesture prediction is 1 then images are blurred;2, image is resized;3,image is rotated etc.



Electrocorticography
Swipe left
Swipe right
Pre-defined hand-gestures
Depth camera
Gloves
Camera
Radio sensors
Sensor for capturing Hand-gestures
Sensor data representation
Features extraction
Classification/detection
Sensor data processing
Application development

IDEA-3:

"Hand Gesture Recognition Using Camera "

"Hand Gesture Recognition Using Camera " is based on the concept of Image processing. In recent years there is a lot of research on gesture recognition using kinect sensors on using HD cameras but camera and kinect sensors are more costly. This idea focuses on to reduce cost and improve robustness of the proposed system using simple web cameras .

The sterile gesture interface consists of a Canon VC-C4 camera, whose pan/tilt/zoom can be initially set using an infrared (IR) remote. This camera is placed just over a large flat screen monitor . Additionally, an Intel Pentium IV, (600MHz, OS: Windows XP) with a Matrox Standard II video-capturing device is used.

A two layer architecture is used: In the lower level "Gestix" provides tracking and recognition functions, while at the higher level a graphical user interface called "Gibson" manages imaging visualisation.

The Tracking Algorithm:

After a short calibration process, where a probability colour model of the doctor's hand is built, images of the surgeon's hand gesturing are acquired by video-camera and each image is back-projected using a colour model. The hand is then tracked by an algorithm which segments it from the background using the colour model back-projection and motion cues. This is followed by black/white thresholding, and a sequence of opening and closing morphological operations resulting in a set of components (blobs) in the image. The location of the hand is represented by the 2D coordinates of the centroid of the biggest blob in the current image.

“Gibson” Image Browser:

The “Gibson” image browser is a 3D visualisation medical tool that enables examination of images, such as: MRIs, CT scans and X-rays. The images are arranged over a multiple layer 3D cylinder. The image of interest is found through rotating the cylinder in the four cardinal directions. To interface the gesture recognition routines with the “Gibson” system, information such as the centroid of the hand, its size, and orientation are used to enable screen operations in the “Gibson” graphical user interface.

Hand Tracking and Operation Modes:

Gesture operations are initiated by a calibration mode in which a skin colour model of the user's hand or glove, under local lighting, is constructed. In a browse mode, superimposed over the image of the camera's scene is a rectangular frame called the “neutral area.” Movements of the hand across its boundary constitute directional browser commands. When a doctor/surgeon wishes to browse the image database, the hand is moved rapidly out of the “neutral area” toward any of four directions, and then back again. When such a movement is detected, the displayed image is moved off the screen and replaced by a neighbouring image. To evoke a zoom mode, the open palm of the hand is rotated within the “neutral area” clockwise/counterclockwise (zoom-in/zoom-out). To avoid the tracking of unintentional gestures, the user may enter a “sleep mode” by dropping the hand.

To re-arouse the system the user waves the hand in front of the camera. The selection of these gestures was designed to be intuitive, expressing the “natural” feeling of the user. For example, the left/right/up/down gestures evoke the actions used to turn pages in a book left/right, or flip notepad pages up/down. The rotation gesture (zoom-in/zoom-out commands) reminds one of a radio knob to increase or decrease volume. Dropping the hand (stop-tracking command) is associated to the idea of ‘stop-playing', while the waving gesture (“wake-up” command) is associated with ‘greeting a new person’.

3.2.BRAINSTORMING:

3.3.PROPOSED SOLUTION:

Proposed Solution:

S.No.	Parameter	Description
1.	Problem Statement	To design an ML model to identify and classify the hand gestures.
2.	Idea / Solution description	To develop a CNN based classifier model, which would be trained on our training data.
3.	Novelty / Uniqueness	We train a CNN based model to recognize the hand gesture. The training data include images that capture the hand gestures of 1,2,3,4,5 and 0. The image is resized without much loss of information and used for training a CNN based model. We use Python Flask to provide an interactive platform for our model.

4.	Social Impact / Customer Satisfaction	This project would help the doctors in operation theatres where physical contact between persons should be avoided in order to be sterilised and also prevent from any infections.
5.	Business Model (Revenue Model)	It can be sold as an open-source service to all the hospitals as a non-profitable work.
6.	Scalability of the Solution	The model could also be extended to other real world classifying problems like cancer detection from X-ray, COVID detection using X-ray images, mask detection, face detection etc..

1. CUSTOMER SEGMENT:

- Those who want to surf the internet for both Specially Abled person and common person.
- People want to communicate with “Hand signs”.
- Common People want to control “IOT devices through hand signs”.
- Who wants to control the “Screens with Hand Gestures”.

2. JOBS-TO-BE-DONE:

- Giving the necessary information for a particular thing to the customer.
- To make the computer to understand hand signs

3. TRIGGERS TR :

- They want to advance their lifestyle
- Easy to use (user friendly interface)
- Can be used in many sectors

4. EMOTIONS:

Before:

Confident – They feel confident about the software without knowing the basic English alphabetic hand signs.

After:

It makes them to use this software with full efficient

5.AVAILABLE SOLUTIONS:

- To have the Proper and regular Internet connection to stay connected to surf.
- By searching in books, e-books, online websites etc...
- To know the English alphabet hand signs to surf

6.CUSTOMER CONSTRAINTS:

- Anxiety - customers begin to get anxious when they still have no idea about what they have found.
- Available devices – Customers are required to have the necessary devices.
- Network Connection – It is necessary to have a network connection.

7.BEHAVIOUR:

- For Specially Abled people it makes it too difficult to communicate with computers.
- To advance life with hand gestures by controlling the cars, IOTs, screens etc…

8.CHANNELS OF BEHAVIOUR:

<u>8.1 ONLINE</u>

Online websites Social media platforms

<u>8.2 OFFLINE</u>

Customer throw words

9.PROBLEM ROOT CAUSE:

- Unaware of the object. New to the environment. Technology to capture the images.
- Technique of classification must be appropriate.
- Computers cannot understand the human hand signs.

10.OUR SOLUTION:

The following approaches are used to built an efficient “Web Application” that recognize the hand gestures:

IBM Watson Assistant - To build the web application as a user interface.

Flask - Web Application Framework. Convolutional Neural Network (CNN) - To find patterns in images to recognize objects, faces, and scenes.

4.REQUIREMENT ANALYSIS:

4.1. FUNCTIONAL REQUIREMENT:

The following are the required functional requirements.

FR No.	Functional Requirement(EPIC)	Sub Requirement (Story / Sub-Task)
FR-1	Accessing dataset	Datasets are collected by data preprocessing method then followed by data visualisation.
FR-2	Classification of dataset	Dataset includes data exploration to train the model accordingly.
FR-3	Splitting and train the data	In this phase, we split the dataset into training and test dataset and then train the models using CNN. We have used CNN (Convolution Neural Network) to train the model using various images of different hand gestures. Made a web application using flask where the user can upload the image which he wants to browse through.

FR-4	Test the model	In this phase, we tested the accuracy, precision and sensitivity of the models with the test dataset that was formed in the previous phase and the most accurate model is figured out.

Accessing data sets, classification of data sets, splitting and accessing the data set and finally testing the model are the important function requirements.

- Under ACCESSING THE DATA SET datasets are collected by data preprocessing method then followed by data visualisation.
- In CLASSIFICATION OF DATA SET the dataset includes data exploration to train the model accordingly.
- In SPLITTING AND ACCESSING THE DATA SET phase, we split the dataset into training and test dataset and then train the models using CNN. We have used CNN (Convolution Neural Network) to train the model using various images of different hand gestures. Made a web application using flask where the user can upload the image which he wants to browse through.
- Finally in TEST THE MODEL we tested the accuracy, precision and sensitivity of the models with the test dataset that was formed in the previous phase and the most accurate model is figured out.

4.2. NON FUNCTIONAL REQUIREMENTS:

Some of the notable non functional requirements include

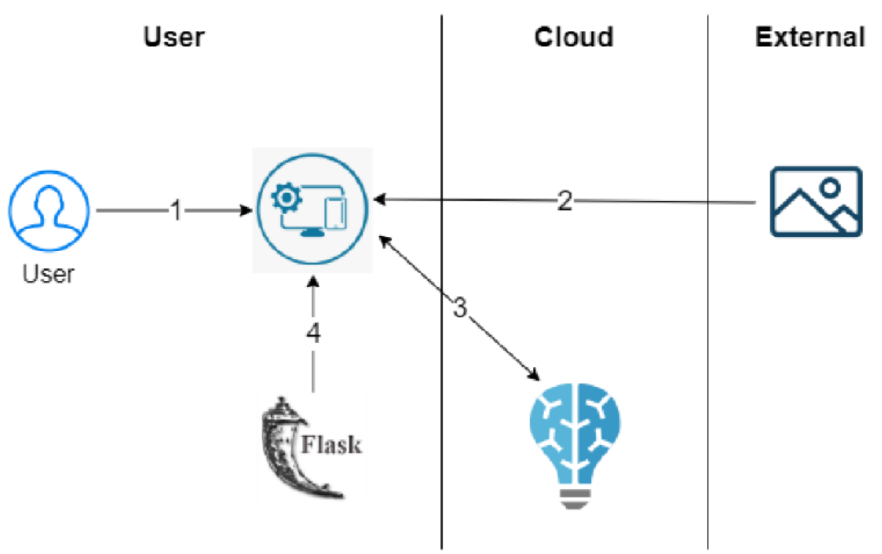
- Usability
- Performance
- Reliability
- Security

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The use of doctor-computer interaction devices in the operation room (OR) requires new modalities that support medical imaging manipulation while allowing doctors' hands to remain sterile, supporting their focus of attention, and providing fast response times.
NFR-2	**Security**	This interface prevented the surgeon's focus shift and change of location while achieving a rapid intuitive reaction and easy interaction.

NFR-3	**Reliability**	A vision-based hand gesture capture and recognition system that interprets in real-time the user's gestures for navigation and manipulation of images.
NFR-4	**Performance**	Data from two usability tests provide insights and implications regarding human-computer interaction based on nonverbal conversational modalities.

5. PROJECT DESIGN

5.1 Data Flow Diagrams:

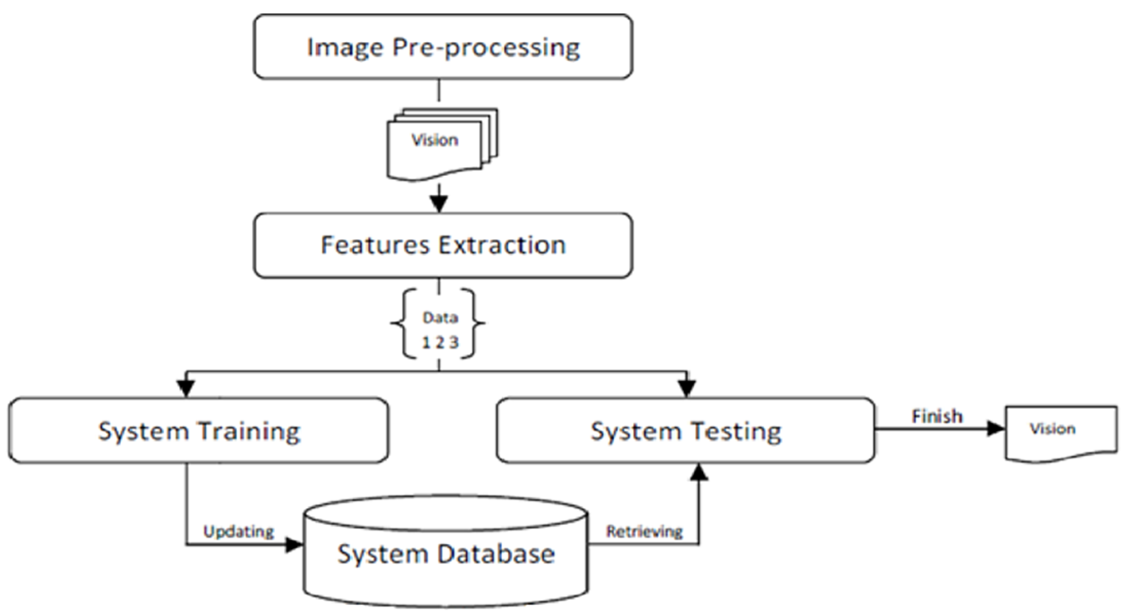


5.2 Solution Architecture:

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behaviour, and other aspects of the software to project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed, and delivered.

Solution Architecture Diagram:



Technical Architecture:

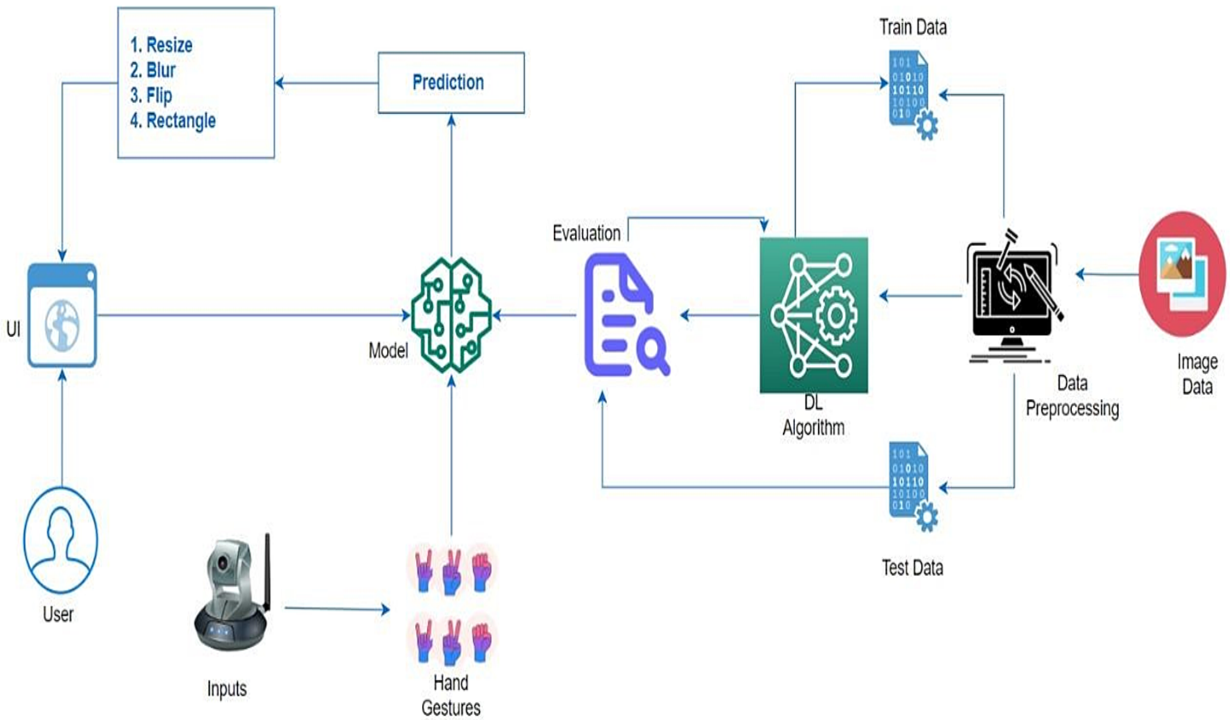


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	Web UI	HTML, CSS, JavaScript.
2.	Application Logic-1 Image Pre-processing	Input image is pre-processed with the help of library files	Python, TensorFlow
3.	Application Logic-2 Building Model	Building CNN model to recognize the gesture.	Python, Keras
4.	Application Logic-3 Creation of app	App is built to obtain gesture as input and to provide as output.	HTML, CSS, JavaScript
5.	Dataset	Hand gesture data set.	From IBM
6.	Cloud Database	User input image is stored in cloud.	IBM Cloud
7.	File Storage	File storage contains dataset and source code.	Server and Local Filesystem
8.	Machine Learning Model	CNN Model was used to recognize the preprocessed image by image capturing or by video segmenting.	CNN Model by Python, Keras

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	For development of code, package manager, for building model	Visual Studio Code, Conda, TensorFlow
2.	Resilient	Gestures can be captured in different environments (variable brightness and distance).	OpenCV, TensorFlow
3.	Availability	Deploy on highly available server	IBM Cloud
4.	Performance	CNN model is used to predict the input gesture in a shorter span of time.	TensorFlow, Keras
5.	Diverse Dataset	Data augmentation to generate more data from limited set of images.	Keras

5.3 User Stories

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority
Customer (Web user)	Launch Web App deployed in cloud	USN-1	As a user, I can launch the web app where I can upload the images for recognition	I can upload the images for classification	High
	Prediction	USN-2	As a user, I can get the predicted results from the model deployed in the cloud	I can resize the radiology image, blur the image, flip based on the hand gesture	High
	Deployment of web app in the cloud	USN-3	As a user, I need the web app to be accessible all over the world	I can access the web app deployed in the IBM cloud	Medium

	Deployment of AI model in the cloud	USN-4	As a user, I need the AI model to be accessible all over the world	I can access the model deployed in the IBM cloud	Medium
	Model Building	USN-5	As a user, I need an AI model which could classify or recognize the hand gestures	I can get the prediction from the AI model	Medium
	User Interface Building	USN-6	As a user, I need a web app for human computer interaction	I get User Interface for interaction with the model	Medium

6.PROJECT PLANNING AND SCHEDULING:

6.1 SPRINT PLANNING AND ESTIMATION:

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	10	6 Days	24 Oct 2022	29 Oct 2022	10	29 Oct 2022
Sprint-2	10	6 Days	31 Oct 2022	05 Nov 2022	10	05 Nov 2022
Sprint-3	10	6 Days	07 Nov 2022	12 Nov 2022	10	12 Nov 2022
Sprint-4	10	6 Days	14 Nov 2022	19 Nov 2022	10	19 Nov 2022

6.2 SPRINT DELIVERY SCHEDULE:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection	USN-1	Download the Dataset	10	High	Sindhuja C
Sprint-1		USN-2	Image Pre-processing(Import the library, Image preprocessing, Configure ImageDataGenerator, Apply image generator functionality)	10	High	Sindhuja C
Sprint-2	Model Building	USN-3	Import the Model Building Libraries and initialise the model	10	High	Sowmiya M
Sprint-2		USN-4	Adding CNN layers and Dense layers	10	High	Sowmiya M

Sprint-2		USN-5	Configure the learning process	10	High	Swetha E
Sprint-2		USN-6	Train test and save the model	10	High	Swetha E
Sprint-3	Website Building	USN-7	Create the HTML pages	10	High	Vaishnavi Parvathy N
Sprint-3		USN-8	Build Python code	10	High	Vaishnavi Parvathy N
Sprint-3		USN-9	Run the application	10	High	Vaishnavi Parvathy N
Sprint-4	Train The Model on IBM	USN-11	Register for IBM Cloud	10	High	Swetha E
Sprint-4		USN-12	Train the Model and Test the Model and its Overall Performance	10	High	Sowmiya M Sindhuja C

Velocity:

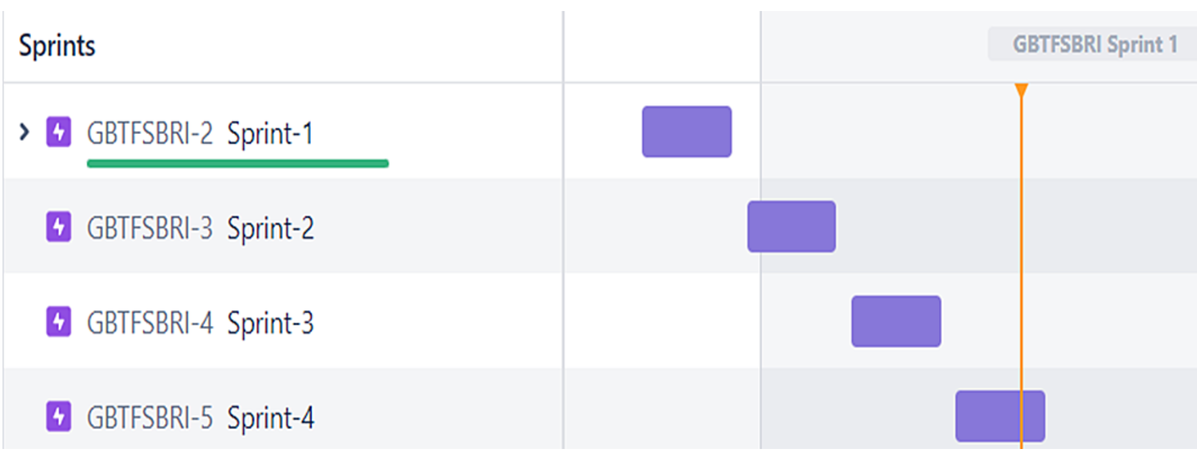
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV)

periteration unit

(story points per day)

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

6.3 REPORTS FROM JIRA:



7. CODING AND SOLUTIONING:

7.1 MODEL BUILDING:

Training Model:

```
# In[1]:
pip install tensorflow
# In[1]:
import numpy as np
import tensorflow
from tensorflow.keras.models import Sequential
from tensorflow.keras import layers
from tensorflow.keras.layers import Dense,Flatten
from tensorflow.keras.layers import Conv2D,MaxPooling2D
from keras.preprocessing.image import ImageDataGenerator
# In[2]:
train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_r
ange=0.2,horizontal_flip=True)
test_datagen=ImageDataGenerator(rescale=1./255)
# In[3]:
x_train = train_datagen.flow_from_directory(r'dataset/train',
                                            target_size=(64, 64),
                                            batch_size=5,
                                            color_mode='grayscale',
class_mode='categorical')
x_test = test_datagen.flow_from_directory(r'dataset/test',
                                          target_size=(64, 64),
                                          batch_size=5,
```

```
                                                   color_mode='grayscale',
                                                   class_mode='categorical')
# In[4]:
print(x_train.class_indices)
# In[5]:
model=Sequential()
# In[6]:
model.add(Conv2D(32,(3,3),input_shape=(64,64,1),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Conv2D(32,(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
# In[7]:
model.add(Flatten())


# In[8]:
model.add(Dense(units=128,activation='relu'))
model.add(Dense(units=6,activation='softmax'))
# In[9]:
model.summary()
# In[10]:
model.compile(optimizer='adam',loss='categorical_crossentropy',metrics
=['accuracy'])
# In[11]:
model.fit_generator(x_train,
                    steps_per_epoch = len(x_train) ,
                    epochs = 20,
                    validation_data = x_test,
                    validation_steps = len(x_test) )
# In[12]:
model.save('gesture.h5')
model_json=model.to_json()
with open("model-bw.json","w") as json_file:json_file.write(model_json)
```

Testing Model:

```
# In[1]:
from tensorflow.keras.models import load_model
import keras
import numpy as np
from IPython.display import Image
model=load_model("gesture.h5")
# In[2]:
img=keras.utils.load_img("dataset/test/1/1.jpg",grayscale=True,target_size=(64,64))
img
# In[3]:
x=keras.utils.img_to_array(img)
x=np.expand_dims(x,axis=0)
pred=model.predict(x)
pred=pred.astype(int).tolist()
pred=pred[0]
pred
# In[4]:
index=['0','1','2','3','4','5']
for i in range(len(pred)):
    if(pred[i]==1):
        print(index[i])
# In[5]:
for i in range(0,6):
    for j in range(0,5):
img=keras.utils.load_img("dataset/test/"+str(i)+"/"+str(j)+".jpg",grayscale=True
                                 ,target_size=(64,64))
        display(Image(filename="dataset/test/"+str(i)+"/"+str(j)+".jpg") )
        x=keras.utils.img_to_array(img)
        x=np.expand_dims(x,axis=0)
        pred=model.predict(x)
        pred=pred.astype(int).tolist()
        pred=pred[0]
        for k in range(len(pred)):
            if(pred[k]==1):
                print(index[k])
```

7.2 APPLICATION BUILDING:

HOME.HTML:

```
<html>
<script>
</script>
<style>
.header {    position: relative;
                    top:0;
                    margin:0px;
                    z-index: 1;
                    left: 0px;
                    right: 0px;
                    position: fixed;
                    background-color:rgb(10, 83, 109) ;
                    color: rgb(181, 228, 236);
                    box-shadow: 0px 8px 4px rgb(10, 102, 109);
                    overflow: hidden;
                    padding-left:20px;
                    font-family: 'Times New Roman';
                    font-size: 2vw;
                    width: 100%;
                    height:8%;
                    text-align: center;
             }
             .topnav {
  overflow: hidden;
  background-color: #84d3d3;
}

.topnav-right a {
  float: left;
  color: rgba(0, 0, 0, 0.519);
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
  font-size: 18px;
```

```
}
.topnav-right a.active {
  background-color: #07201e;
  color: rgb(238, 226, 234);
}
.topnav-right a:hover {
  background-color: rgb(181, 228, 236);
  color: rgb(6, 27, 36);
}

.topnav-right {
  float: right;
  padding-right:100px;
}

body {

  background-image: url("https://img.freepik.com/free-vector/watercolor-
background_87374-69.jpg?size=626&ext=jpg&ga=GA1.2.1848574980.1668617761") ;
  background-position: center;
  background-repeat: no-repeat;
  background-size: cover;

   }
   .button {
  background-color: #091425;
  border: none;
  color: rgb(181, 228, 236);
  padding: 15px 32px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 12px;
  border-radius: 16px;
}
.button:hover {
  box-shadow: 0 12px 16px 0 rgba(0,0,0,0.24), 0 17px 50px 0 rgba(0,0,0,0.19);
}
form {border: 3px solid #f1f1f1; margin-left:400px;margin-right:400px;}

input[type=text], input[type=password] {
  width: 100%;
```

```
  padding: 12px 20px;
  display: inline-block;
  margin-bottom:18px;
  border: 1px solid #ccc;
  box-sizing: border-box;
}

button {
  background-color: #091425;
  color: rgb(181, 228, 236);
  padding: 14px 20px;
  margin-bottom:10px;
  border: none;
  cursor: pointer;
  width: 17%;
  border-radius:4px;
  font-family:Montserrat;
}

button:hover {
  opacity: 0.8;
}

.cancelbtn {
  width: auto;
  padding: 10px 18px;
  background-color: rgb(181, 228, 236);
}

.imgcontainer {
  text-align: center;
  margin: 24px 0 12px 0;
}

img.avatar {
  width: 30%;
  border-radius: 50%;
}

.container {
  padding: 16px;
}
```

```
span.psw {
  float: right;
  padding-top: 16px;
}

/* Change styles for span and cancel button on extra small screens */
@media screen and (max-width: 300px) {
  span.psw {
     display: block;
     float: none;
  }
  .cancelbtn {
     width: 100%;
  }
}

.home{
	margin:80px;


  width: 84%;
  height: 500px;
  padding-top:10px;
  padding-left: 30px;

}
.login{
	margin:80px;
	box-sizing: content-box;
  width: 84%;
  height: 420px;
  padding: 30px;
  border: 10px solid rgb(13, 53, 68);
}
.left,.right{
 box-sizing: content-box;
 height: 400px;
 margin:20px;
 border: 10px solid rgb(13, 53, 68);
}

.mySlides {display: none;}
```

```css
img {vertical-align: middle;}

/* Slideshow container */
.slideshow-container {
  max-width: 1000px;
  position: relative;
  margin: auto;
}

/* Caption text */
.text {
  color: #9ac0c0;
  font-size: 15px;
  padding: 8px 12px;
  position: absolute;
  bottom: 8px;
  width: 100%;
  text-align: center;
}
/* The dots/bullets/indicators */
.dot {
  height: 15px;
  width: 15px;
  margin: 0 2px;
  background-color: #bbb;
  border-radius: 50%;
  display: inline-block;
  transition: background-color 0.6s ease;
}

.active {
  color: rgb(145, 216, 221);
}

/* Fading animation */
.fade {
  -webkit-animation-name: fade;
  -webkit-animation-duration: 1.5s;
  animation-name: fade;
  animation-duration: 1.5s;
}
```

```
@-webkit-keyframes fade {
  from {opacity: .4}
  to {opacity: 1}
}

@keyframes fade {
  from {opacity: .4}
  to {opacity: 1}
}

/* On smaller screens, decrease text size */
@media only screen and (max-width: 300px) {
  .text {font-size: 11px}
}
```

```
@import url("https://fonts.googleapis.com/css?family=Luckiest+Guy");
/* BODY */
body {

  position: absolute;
  top: 0;
  left: 0;
  right: 0;
  bottom: 0;
  width: 100%;
  height: 100%;
  overflow: hidden;
  font-family: "Arial", cursive;
  -webkit-font-smoothing: antialiased;
}

::selection {
  background: transparent;
}
/* CLOUDS */
body:before {
  content: "";
  position: absolute;
  top: 0;
```

```
  left: 0;
  right: 0;
  width: 0;
  height: 0;
  margin: auto;
  border-radius: 100%;
  background: transparent;
  display: block;
  box-shadow: 0 0 150px 100px rgba(255, 255, 255, 0.6),
    200px 0 200px 150px rgba(255, 255, 255, 0.6),
    -250px 0 300px 150px rgba(255, 255, 255, 0.6),
    550px 0 300px 200px rgba(255, 255, 255, 0.6),
    -550px 0 300px 200px rgba(255, 255, 255, 0.6);
}
/* JUMP */
h1 {
  cursor: default;
  position: absolute;
  top: 0;
  left: 0;
  right: 0;
  bottom: 0;
  width: 100%;
  height: 100px;
  margin: 70px;
  display: block;
  text-align: center;
}

h1 span {
  position: relative;
  top: 5px;
  display: inline-block;
  font-size: 25px;
  color: #061a1f;
  text-shadow: 0 1px 0 rgb(151, 201, 197), 0 2px 0 rgb(151, 201, 197), 0 3px 0
rgb(151, 201, 197), 0 4px 0 rgb(151, 201, 197),
    0 5px 0 rgb(151, 201, 197), 0 6px 0 transparent, 0 7px 0 transparent, 0 8px 0
transparent,
    0 9px 0 transparent, 0 10px 10px rgba(58, 159, 167, 0.4);
}
```

```
h1 span:nth-child(2) {
  -webkit-animation-delay: 0.1s;
  animation-delay:0.1s;
}

h1 span:nth-child(3) {
  -webkit-animation-delay: 0.2s;
  animation-delay: 0.2s;
}

h1 span:nth-child(4) {
  -webkit-animation-delay: 0.3s;
  animation-delay: 0.3s;
}

h1 span:nth-child(5) {
  -webkit-animation-delay: 0.4s;
  animation-delay: 0.4s;
}

h1 span:nth-child(6) {
  -webkit-animation-delay: 0.5s;
  animation-delay: 0.5s;

}

h1 span:nth-child(7) {
  -webkit-animation-delay: 0.6s;
  animation-delay: 0.6s;
}

h1 span:nth-child(8) {
  -webkit-animation-delay: 0.2s;
  animation-delay: 0.2s;
}

h1 span:nth-child(9) {
  -webkit-animation-delay: 0.3s;
  animation-delay: 0.3s;
}
```

```css
h1 span:nth-child(10) {
  -webkit-animation-delay: 0.4s;
  animation-delay: 0.4s;
}

h1 span:nth-child(11) {
  -webkit-animation-delay: 0.5s;
  animation-delay: 0.5s;
}

h1 span:nth-child(12) {
  -webkit-animation-delay: 0.6s;
  animation-delay: 0.6s;
}

h1 span:nth-child(13) {
  -webkit-animation-delay: 0.7s;
  animation-delay: 0.7s;
}

h1 span:nth-child(14) {
  -webkit-animation-delay: 0.8s;
  animation-delay: 0.8s;
}

/* ANIMATION */
@-webkit-keyframes bounce {
  100% {
    top: -20px;
    text-shadow: 0 1px 0 #ccc, 0 2px 0 #ccc, 0 3px 0 #ccc, 0 4px 0 #ccc,
      0 5px 0 #ccc, 0 6px 0 #ccc, 0 7px 0 #ccc, 0 8px 0 #ccc, 0 9px 0 #ccc,
      0 50px 25px rgba(0, 0, 0, 0.2);
  }
}
@keyframes bounce {
  100% {
    top: -20px;
    text-shadow: 0 1px 0 #ccc, 0 2px 0 #ccc, 0 3px 0 #ccc, 0 4px 0 #ccc,
      0 5px 0 #ccc, 0 6px 0 #ccc, 0 7px 0 #ccc, 0 8px 0 #ccc, 0 9px 0 #ccc,
      0 50px 25px rgba(0, 0, 0, 0.2);
  }
}
```

```
</style>

<body >
<h1 style="color: rgb(254, 250, 250);">
  <table style="width:100%">
    <tr>
      <th><img
src="https://media.tenor.com/images/30169e4a670daf12443df7d2dd140176/tenor.gif"
alt="NOT AVAILABLE" width="300" height="300"></th>

      <th></th>
    </tr>
  </table>
    <br>
    <span>HAND   GESTURE   RECOGNITION</span>
  </br>
    <span>OF</span>
  </br>
    <span>RADIOLOGY   IMAGES</span>
  </br>
    <span>THROUGH</span>
  </br>
    <span>STERILE   BROWSING</span>

</h1>
<div class="header">
<div style="width:50%;float:left;font-size:2vw;text-align:left;color:#c4dfd7; padding-
top:1%;padding-left:5%;"><b>HAND GESTURE RECOGNITION</b></div>
  <div class="topnav-right"style="padding-top:0.5%;color:white">
    <a  class="active" href="{{ url_for('home')}}"><u>Home</u></a>
    <a  class="active" href="{{ url_for('intro')}}">Introduction</a>
    <a  class="active" href="{{ url_for('image1')}}">Launch</a>
  </div>
</div>

</body>

</html>
```

INTRO.HTML:

```
<html>
<script>



</script>



<style>
.header {    position: relative;
                    top:0;
                    margin:0px;
                    z-index: 1;
                    left: 0px;
                    right: 0px;
                    position: fixed;
                    background-color:rgb(10, 83, 109) ;
                    color: rgb(181, 228, 236);
                    box-shadow: 0px 8px 4px rgb(10, 102, 109);
                    overflow: hidden;
                    padding-left:20px;
                    font-family: 'Josefin Sans';
                    font-size: 2vw;
                    width: 100%;
                    height:8%;
                    text-align: center;
             }
             .topnav {
  overflow: hidden;
  background-color: #FCAD98;
}

.topnav-right a {
  float: left;
  color: black;
  text-align: center;
```

```
  padding: 14px 16px;
  text-decoration: none;
  font-size: 18px;
}

.topnav-right a.active {
  background-color: #07201e;
  color: rgb(238, 226, 234);
}

.topnav-right a:hover {
  background-color: rgb(181, 228, 236);
  color: rgb(6, 27, 36);
}

.topnav-right {
  float: right;
  padding-right:100px;
}

body {



    background: url("https://img.freepik.com/free-vector/watercolor-background_87374-
69.jpg?size=626&ext=jpg&ga=GA1.2.1848574980.1668617761") ;
    background-position: center;
    background-repeat: no-repeat;
    background-size: cover;

  }
  .button {
  background-color: #091425;
  border: none;
  color: white;
  padding: 15px 32px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 12px;
  border-radius: 16px;
}
.button:hover {
```

```
  box-shadow: 0 12px 16px 0 rgba(0,0,0,0.24), 0 17px 50px 0 rgba(0,0,0,0.19);
}
form {border: 3px solid #f1f1f1; margin-left:400px;margin-right:400px;}

input[type=text], input[type=password] {
  width: 100%;
  padding: 12px 20px;
  display: inline-block;
  margin-bottom:18px;
  border: 1px solid #ccc;
  box-sizing: border-box;
}

button {
  background-color: #091425;
  color: white;
  padding: 14px 20px;
  margin-bottom:10px;
  border: none;
  cursor: pointer;
  width: 17%;
  border-radius:4px;
  font-family:Montserrat;
}

button:hover {
  opacity: 0.8;
}

.cancelbtn {
  width: auto;
  padding: 10px 18px;
  background-color: #f44336;
}

.imgcontainer {
  text-align: center;
  margin: 24px 0 12px 0;
}

img.avatar {
  width: 30%;
```

```
  border-radius: 50%;
}

.container {
  padding: 16px;
}

span.psw {
  float: right;
  padding-top: 16px;
}

/* Change styles for span and cancel button on extra small screens */
@media screen and (max-width: 300px) {
  span.psw {
     display: block;
     float: none;
  }
  .cancelbtn {
     width: 100%;
  }
}

.home{
	margin:80px;

  width: 84%;
  height: 500px;
  padding-top:10px;
  padding-left: 30px;

}
.login{
	margin:80px;
	box-sizing: content-box;
  width: 84%;
  height: 420px;
  padding: 30px;
  border: 10px solid blue;
}
.left,.right{
 box-sizing: content-box;
```

```
 height: 400px;
 margin:20px;
 border: 10px solid blue;
}

.mySlides {display: none;}
img {vertical-align: middle;}

/* Slideshow container */
.slideshow-container {
  max-width: 1000px;
  position: relative;
  margin: auto;
}

/* Caption text */
.text {
  color: #f2f2f2;
  font-size: 15px;
  padding: 8px 12px;
  position: absolute;
  bottom: 8px;
  width: 100%;
  text-align: center;
}
/* The dots/bullets/indicators */
.dot {
  height: 15px;
  width: 15px;
  margin: 0 2px;
  background-color: #bbb;
  border-radius: 50%;
  display: inline-block;
  transition: background-color 0.6s ease;
}

.active {
  background-color: #FCAD98;
}

/* Fading animation */
.fade {
```

```
  -webkit-animation-name: fade;
  -webkit-animation-duration: 1.5s;
  animation-name: fade;
  animation-duration: 1.5s;
}

@-webkit-keyframes fade {
  from {opacity: .4}
  to {opacity: 1}
}

@keyframes fade {
  from {opacity: .4}
  to {opacity: 1}
}

/* On smaller screens, decrease text size */
@media only screen and (max-width: 300px) {
  .text {font-size: 11px}
}
```

```
@import url("https://fonts.googleapis.com/css?family=Montserrat&display=swap");

* {
  padding: 0;
  margin: 0;
}

body {
  height: 100vh;
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
}

h1 {
  font-family: "Montserrat Medium";
  max-width: 90ch;
```

```
  text-align: center;
  transform: scale(0.94);
  animation: scale 3s forwards cubic-bezier(0.5, 1, 0.89, 1);
}
@keyframes scale {
  100% {
    transform: scale(1);
  }
}

span {
  display: inline-block;
  opacity: 0;
  filter: blur(4px);
}

span:nth-child(1) {
  animation: fade-in 1s 0.1s forwards cubic-bezier(0.11, 0, 0.5, 0);
}

span:nth-child(2) {
  animation: fade-in 0.8s 0.2s forwards cubic-bezier(0.11, 0, 0.5, 0);
}

span:nth-child(3) {
  animation: fade-in 0.8s 0.3s forwards cubic-bezier(0.11, 0, 0.5, 0);
}

span:nth-child(4) {
  animation: fade-in 0.8s 0.4s forwards cubic-bezier(0.11, 0, 0.5, 0);
}

span:nth-child(5) {
  animation: fade-in 0.8s 0.5s forwards cubic-bezier(0.11, 0, 0.5, 0);
}

span:nth-child(6) {
  animation: fade-in 0.8s 0.6s forwards cubic-bezier(0.11, 0, 0.5, 0);
}

span:nth-child(7) {
  animation: fade-in 0.8s 0.7s forwards cubic-bezier(0.11, 0, 0.5, 0);
```

```
}

span:nth-child(8) {
  animation: fade-in 0.8s 0.8s forwards cubic-bezier(0.11, 0, 0.5, 0);
}

span:nth-child(9) {
  animation: fade-in 0.8s 0.9s forwards cubic-bezier(0.11, 0, 0.5, 0);
}

span:nth-child(10) {
  animation: fade-in 0.8s 1s forwards cubic-bezier(0.11, 0, 0.5, 0);
}

span:nth-child(11) {
  animation: fade-in 0.8s 1.1s forwards cubic-bezier(0.11, 0, 0.5, 0);
}

span:nth-child(12) {
  animation: fade-in 0.8s 1.2s forwards cubic-bezier(0.11, 0, 0.5, 0);
}

span:nth-child(13) {
  animation: fade-in 0.8s 1.3s forwards cubic-bezier(0.11, 0, 0.5, 0);
}

span:nth-child(14) {
  animation: fade-in 0.8s 1.4s forwards cubic-bezier(0.11, 0, 0.5, 0);
}

span:nth-child(15) {
  animation: fade-in 0.8s 1.5s forwards cubic-bezier(0.11, 0, 0.5, 0);
}

span:nth-child(16) {
  animation: fade-in 0.8s 1.6s forwards cubic-bezier(0.11, 0, 0.5, 0);
}

span:nth-child(17) {
  animation: fade-in 0.8s 1.7s forwards cubic-bezier(0.11, 0, 0.5, 0);
}
```

```
span:nth-child(18) {
  animation: fade-in 0.8s 1.8s forwards cubic-bezier(0.11, 0, 0.5, 0);
}
span:nth-child(19) {
  animation: fade-in 0.8s 1.9s forwards cubic-bezier(0.11, 0, 0.5, 0);
}
span:nth-child(20) {
  animation: fade-in 0.8s 2.0s forwards cubic-bezier(0.11, 0, 0.5, 0);
}
span:nth-child(21) {
  animation: fade-in 0.8s 2.1s forwards cubic-bezier(0.11, 0, 0.5, 0);
}
span:nth-child(22) {
  animation: fade-in 0.8s 2.2s forwards cubic-bezier(0.11, 0, 0.5, 0);
}
span:nth-child(23) {
  animation: fade-in 0.8s 2.3s forwards cubic-bezier(0.11, 0, 0.5, 0);
}span:nth-child(24) {
  animation: fade-in 0.8s 2.4s forwards cubic-bezier(0.11, 0, 0.5, 0);
}span:nth-child(25) {
  animation: fade-in 0.8s 2.5s forwards cubic-bezier(0.11, 0, 0.5, 0);
}span:nth-child(26) {
  animation: fade-in 0.8s 2.6s forwards cubic-bezier(0.11, 0, 0.5, 0);
}span:nth-child(27) {
  animation: fade-in 0.8s 2.7s forwards cubic-bezier(0.11, 0, 0.5, 0);
}span:nth-child(28) {
  animation: fade-in 0.8s 2.8s forwards cubic-bezier(0.11, 0, 0.5, 0);
}
@keyframes fade-in {
  100% {
    opacity: 1;
    filter: blur(0);
  }
}

</style>

<body>

  <h1>INTRODUCTION</h1>
<h1>
```

```
  <span> Hand Gesture </span> <span> recognition system </span> <span> provides us
</span > <span> an innovative,</span> <span>natural,</span> <span>  user friendly
</span> <span> way of interaction </span > <span> with the computer</span>
  <span> which is more </span> <span> familiar to the </span> <span>human beings.
</span > <span> In our project, </span> <span> the hand region </span> <span> is
extracted from </span> <span> the background  </span > <span> by using</span>
<span> Region of intrest. </span> <span> Then, </span> <span> we will be </span >
<span>predicting the labels </span> <span> based on the </span> <span> CNN trained
model weights  </span> <span> of hand gestures </span > <span> using that predicted
labels</span>
<span> we apply if conditions </span> <span> to control some of the actions </span>
<span>like </span > <span>reshaping , blur, flip of the given image.</span>

</h1>
<!--Brian Tracy-->

<div class="header">
<div style="width:50%;float:left;font-size:2vw;text-align:left;color:#c1e2d9; padding-
top:1%;padding-left:5%;">Hand Gesture System</div>
  <div class="topnav-right"style="padding-top:0.5%;">

    <a class="active" href="{{ url_for('home')}}">Home</a>
    <a class="active" href="{{ url_for('intro')}}"><u>Introduction</u></a>
    <a class="active" href="{{ url_for('image1')}}">Launch</a>
  </div>
</div>
</body>

</html>
```

LAUNCH.HTML:

```
<html lang="en">

  <head>
  <meta charset="utf-8">
      <meta http-equiv="X-UA-Compatible" content="IE=edge">
      <meta name="viewport" content="width=device-width, initial-scale=0.6">
      <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>

      <link href="https://fonts.googleapis.com/icon?family=Material+Icons"
rel="stylesheet">
      <meta charset="UTF-8">
      <title>Predict</title>
      <link href="https://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.min.css"
rel="stylesheet">

      <script
src="https://cdn.bootcss.com/popper.js/1.12.9/umd/popper.min.js"></script>
      <script src="https://cdn.bootcss.com/jquery/3.3.1/jquery.min.js"></script>
      <script
src="https://cdn.bootcss.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
      <link href="{{ url_for('static', filename='css/main.css') }}" rel="stylesheet">
  <style>
  .bar
  {
  margin: 0px;
  padding:20px;
  background-color:black;
  opacity:0.6;
  color:black;
  font-family:'Roboto',sans-serif;
  font-style: italic;
  border-radius:20px;
  font-size:25px;
  }
  a
  {
  color:black;
```

```
float:right;
text-decoration:none;
font-style:normal;
padding-right:20px;
}
a:hover{
background-color:black;
color:black;
font-size:30px;
padding-left:10px;
}

div1{
  text-align: center;
  width: 650spx;
  height: 800px;
  padding: 190px;
  margin: 10px;
  position: absolute;
}



body
{
    background-image: url("https://img3.goodfon.com/wallpaper/nbig/a/af/ruki-znaki-steny.jpg");
    background-size: cover;
}

.header {  position: relative;
      top:0;
      margin:0px;
      z-index: 1;
      left: 0px;
      right: 0px;
      position: fixed;
      background-color:rgb(10, 83, 109) ;
      color: rgb(181, 228, 236);
      box-shadow: 0px 8px 4px rgb(10, 102, 109);
      overflow: hidden;
      padding-left:20px;
```

```
      font-family: 'Josefin Sans';
      font-size: 2vw;
      width: 100%;
      height:8%;
      text-align: center;
    }
    .topnav {
  overflow: hidden;
  background-color: #054b69;
}

.topnav-right a {
  float: left;
  color: black;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
  font-size: 18px;
}

.topnav-right a.active {
  background-color: #07201e;
  color: rgb(238, 226, 234);
}

.topnav-right a:hover {
  background-color: rgb(181, 228, 236);
  color: rgb(6, 27, 36);
}



.topnav-right {
  float: right;
  padding-right:100px;
}


.button {
background-color: #16315b;
border: none;
color: rgb(151, 177, 234);
padding: 15px 32px;
```

```
text-align: center;
text-decoration: none;
display: inline-block;
font-size: 12px;
border-radius: 16px;
}
.button:hover {
  box-shadow: 0 12px 16px 0 rgba(0,0,0,0.24), 0 17px 50px 0 rgba(0,0,0,0.19);
}
form {border: 2px solid black; margin-left:400px;margin-right:400px;}

input[type=text], input[type=password] {
  width: 100%;
  padding: 12px 20px;
  display: inline-block;
  margin-bottom:18px;
  border: 1px solid #ccc;
  box-sizing: border-box;
}

button {
  background-color: #091425;
  color: black;
  padding: 14px 20px;
  margin-bottom:10px;
  border: none;
  cursor: pointer;
  width: 17%;
  border-radius:4px;
  font-family:Montserrat;
}

button:hover {
  opacity: 0.8;
}

.cancelbtn {
  width: auto;
  padding: 10px 18px;
  background-color: #f44336;
}

.imgcontainer {
```

```
  text-align: center;
  margin: 24px 0 12px 0;
}

img.avatar {
  width: 30%;
  border-radius: 50%;
}

.container {
  padding: 16px;
}

span.psw {
  float: right;
  padding-top: 16px;
}

/* Change styles for span and cancel button on extra small screens */
@media screen and (max-width: 300px) {
  span.psw {
     display: block;
     float: none;
  }
  .cancelbtn {
     width: 100%;
  }
}

.home{
  margin:80px;

  width: 84%;
  height: 500px;
  padding-top:10px;
  padding-left: 30px;

}
.login{
  margin:80px;
  box-sizing: content-box;
  width: 84%;
  height: 420px;
```

```
  padding: 30px;
  border: 10px solid rgb(12, 91, 94);
}
.left,.right{
 box-sizing: content-box;
 height: 400px;
 margin:20px;
 border: 10px solid rgb(12, 91, 94);
}

.mySlides {display: none;}
img {vertical-align: middle;}

/* Slideshow container */
.slideshow-container {
  max-width: 1000px;
  position: relative;
  margin: auto;
}

/* Caption text */
.text {
  color: #f2f2f2;
  font-size: 15px;
  padding: 8px 12px;
  position: absolute;
  bottom: 8px;
  width: 100%;
  text-align: center;
}
/* The dots/bullets/indicators */
.dot {
  height: 15px;
  width: 15px;
  margin: 0 2px;
  background-color: #bbb;
  border-radius: 50%;
  display: inline-block;
  transition: background-color 0.6s ease;
}

.active {
  background-color: #266781;
```

```
}

/* Fading animation */
.fade {
  -webkit-animation-name: fade;
  -webkit-animation-duration: 1.5s;
  animation-name: fade;
  animation-duration: 1.5s;
}

@-webkit-keyframes fade {
  from {opacity: .4}
  to {opacity: 1}
}

@keyframes fade {
  from {opacity: .4}
  to {opacity: 1}
}

/* On smaller screens, decrease text size */
@media only screen and (max-width: 300px) {
  .text {font-size: 11px}
}

</style>
</head>

<body>
<div class="header">
<div style="width:50%;float:left;font-size:2vw;text-align:left;color:#c1e2d9;
padding-top:1%;padding-left:5%;">Hand Gesture System</div>
  <div class="topnav-right"style="padding-top:0.5%;">

    <a class="active" href="{{ url_for('home')}}">Home</a>
    <a class="active" href="{{ url_for('intro')}}">Introduction</a>
    <a class="active" href="{{ url_for('image1')}}"><u>Launch</u></a>
  </div>
</div>
<br>

  <div1 ><h1><font color="Black" size="6" font-family="Roboto">Hand Gesture
Recognition</h1><br>
```

```
    <p><i><font color="Black" size="4" fonr-family="sans-serif"></i>Provide an image
by selecting the "Choose" button below</p>
    <br>
          <div>
            <h4>Upload Image Here</h4>
        <form action = "http://localhost:5000/" id="upload-file" method="post"
enctype="multipart/form-data">
          <label for="imageUpload" class="upload-label">
            Choose...
          </label>
          <input type="file" name="image" id="imageUpload" accept=".png, .jpg,
.jpeg,.pdf">
        </form>
              <center>
        <div class="image-section" style="display:none;">
          <div class="img-preview">
            <div id="imagePreview">
            </div>
          </div>
          <div>
            <button type="button" class="btn btn-info btn-lg " id="btn-
predict">Predict!</button>
          </div>
        </div>
        <div class="loader" style="display:none;"></div>
              </center>
      </div>
      </div1>

          <footer>
      <script src="{{ url_for('static', filename='js/main.js') }}"
type="text/javascript"></script>
  </footer>

  </html>
```

App.py

```
from flask import Flask,render_template,request
# Flask-It is our framework which we are going to use to run/serve our application.
# request-for accessing file which was uploaded by the user on our application.
import operator
import cv2 # opencv library
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import numpy as np
from tensorflow.python.keras.models import load_model#to load our trained model
import os
from werkzeug.utils import secure_filename
app = Flask(__name__) # initializing a flask app
# Loading the model
model=load_model('ges.h5')
print("Loaded model from disk")
@app.route('/')# route to display the home page
def home():
    return render_template('home.html')



@app.route('/intro') # routes to the intro page
def intro():
    return render_template('intro.html')#rendering the intro page

@app.route('/image1',methods=['GET','POST'])# routes to the index html
def image1():
    return render_template('launch.html')#


@app.route('/predict',methods=['GET', 'POST'])# route to show the predictions in a web
UI
def launch():
    if request.method == 'POST':
        print("inside image")
        f = request.files['image']

        basepath = os.path.dirname(__file__)
        file_path = os.path.join(basepath, 'uploads', secure_filename(f.filename))
```

```
        f.save(file_path)
        print(file_path)
        cap = cv2.VideoCapture(0)
        while True:
            _, frame = cap.read()
            img = cv2.flip(frame, 1)


            # Got this from collect-data.py
            # Coordinates of the ROI
            x1 = int(0.5*frame.shape[1])
            y1 = 10
            x2 = frame.shape[1]-10
            y2 = int(0.5*frame.shape[1])
            # Drawing the ROI
            # The increment/decrement by 1 is to compensate for the bounding box
            cv2.rectangle(frame, (x1-1, y1-1), (x2+1, y2+1), (255,0,0) ,1)
            # Extracting the ROI
            roi = frame[y1:y2, x1:x2]

            # Resizing the ROI so it can be fed to the model for prediction
            roi = cv2.resize(roi, (64, 64))
            roi = cv2.cvtColor(roi, cv2.COLOR_BGR2GRAY)
            _, test_image = cv2.threshold(roi, 120, 255, cv2.THRESH_BINARY)
            cv2.imshow("test", test_image)
            # Batch of 1
            result = model.predict(test_image.reshape(1, 64, 64, 1))
            print(result)
            prediction = {'ZERO': result[0][0],
                          'ONE': result[0][1],
                          'TWO': result[0][2],
                          'THREE': result[0][3],
                          'FOUR': result[0][4],
                          'FIVE': result[0][5]}
            # Sorting based on top prediction
            prediction = sorted(prediction.items(), key=operator.itemgetter(1),
reverse=True)

            # Displaying the predictions
            cv2.putText(frame, prediction[0][0], (10, 120), cv2.FONT_HERSHEY_PLAIN, 1,
(0,255,255), 1)
            cv2.imshow("Frame", frame)
```

```
#loading an image
image1=cv2.imread(file_path)
if prediction[0][0]=='ONE':

    resized = cv2.resize(image1, (200, 200))
    cv2.imshow("Fixed Resizing", resized)
    key=cv2.waitKey(3000)

    if (key & 0xFF) == ord("1"):
        cv2.destroyWindow("Fixed Resizing")

elif prediction[0][0]=='ZERO':

    cv2.rectangle(image1, (480, 170), (650, 420), (0, 0, 255), 2)
    cv2.imshow("Rectangle", image1)
    cv2.waitKey(0)
    key=cv2.waitKey(3000)
    if (key & 0xFF) == ord("0"):
        cv2.destroyWindow("Rectangle")

elif prediction[0][0]=='TWO':
    (h, w, d) = image1.shape
    center = (w // 2, h // 2)
    M = cv2.getRotationMatrix2D(center, -45, 1.0)
    rotated = cv2.warpAffine(image1, M, (w, h))
    cv2.imshow("OpenCV Rotation", rotated)
    key=cv2.waitKey(3000)
    if (key & 0xFF) == ord("2"):
        cv2.destroyWindow("OpenCV Rotation")

elif prediction[0][0]=='THREE':
    blurred = cv2.GaussianBlur(image1, (21, 21), 0)
    cv2.imshow("Blurred", blurred)
    key=cv2.waitKey(3000)
    if (key & 0xFF) == ord("3"):
        cv2.destroyWindow("Blurred")

elif prediction[0][0]=='FOUR':

    resized = cv2.resize(image1, (400, 400))
    cv2.imshow("Fixed Resizing", resized)
    key=cv2.waitKey(3000)
    if (key & 0xFF) == ord("4"):
```

```
                cv2.destroyWindow("Fixed Resizing")

            elif prediction[0][0]=='FIVE':
                '''(h, w, d) = image1.shape
                center = (w // 2, h // 2)
                M = cv2.getRotationMatrix2D(center, 45, 1.0)
                rotated = cv2.warpAffine(image1, M, (w, h))'''
                gray = cv2.cvtColor(image1, cv2.COLOR_RGB2GRAY)
                cv2.imshow("OpenCV Gray Scale", gray)
                key=cv2.waitKey(3000)
                if (key & 0xFF) == ord("5"):
                    cv2.destroyWindow("OpenCV Gray Scale")

            else:
                continue




            interrupt = cv2.waitKey(10)
            if interrupt & 0xFF == 27: # esc key
                break


        cap.release()
        cv2.destroyAllWindows()

    return render_template("home.html")

if __name__ == '__main__':
   app.debug=True
   app.run(host='0.0.0.0', port=81)
```

7.3 MODEL ON IBM :

Training IBM

```
# In[15]:

pip install tensorflow

# In[16]:

import numpy as np

import tensorflow

from tensorflow.keras.models import Sequential

from tensorflow.keras import layers

from tensorflow.keras.layers import Dense,Flatten

from tensorflow.keras.layers import Conv2D,MaxPooling2D

from keras.preprocessing.image import ImageDataGenerator

# In[17]:

train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizon
tal_flip=True)

test_datagen=ImageDataGenerator(rescale=1./255)

# In[18]:

import os, types

import pandas as pd

from botocore.client import Config

import ibm_boto3

def __iter__(self): return 0
```

```
# @hidden_cell

# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.

# You might want to remove those credentials before you share the notebook.

cos_client = ibm_boto3.client(service_name='s3',

    ibm_api_key_id='RkZXMuxnKDp0xraOh75IChu9AG356Y7jp8BoVrGxwMEL',

    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",

    config=Config(signature_version='oauth'),

    endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

bucket = 'imageclassification-donotdelete-pr-y3vcdpcbkrumz5'

object_key = 'dataset.zip'

streaming_body_2 = cos_client.get_object(Bucket=bucket, Key=object_key)['Body']

# Your data file was loaded into a botocore.response.StreamingBody object.

# Please read the documentation of ibm_boto3 and pandas to learn more about the possibilities to load the data.

# ibm_boto3 documentation: https://ibm.github.io/ibm-cos-sdk-python/

# pandas documentation: http://pandas.pydata.org/

# In[19]:

from io import BytesIO

import zipfile

unzip = zipfile.ZipFile(BytesIO(streaming_body_2.read()),'r')

file_paths=unzip.namelist()

for path in file_paths:

    unzip.extract(path)
```

```
# In[20]:

x_train = train_datagen.flow_from_directory(r'dataset/train',

                                            target_size=(64, 64),

                                            batch_size=5,

                                            color_mode='grayscale',

                                            class_mode='categorical')

x_test = test_datagen.flow_from_directory(r'dataset/test',

                                          target_size=(64, 64),

                                          batch_size=5,

                                          color_mode='grayscale',

                                          class_mode='categorical')

# In[21]:

print(x_train.class_indices)

# In[22]:

model=Sequential()

# In[23]:

model.add(Conv2D(32,(3,3),input_shape=(64,64,1),activation='relu'))

model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Conv2D(32,(3,3),activation='relu'))

model.add(MaxPooling2D(pool_size=(2,2)))

# In[24]:

model.add(Flatten())

# In[25]:
```

```
model.add(Dense(units=128,activation='relu'))

model.add(Dense(units=6,activation='softmax'))

# In[26]:

model.summary()

# In[27]:

model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])

# In[28]:

model.fit_generator(x_train,

                    steps_per_epoch = 119 ,

                    epochs = 20,

                    validation_data = x_test,

                    validation_steps = 6 )

# In[29]:

model.save('gesture.h5')

# In[45]:

get_ipython().system('tar -zcvf image-classification-model_new.tgz gesture.h5')

# In[46]:

ls -1

# In[47]:

get_ipython().system('pip install watson-machine-learning-client --upgrade')

# In[80]:

from ibm_watson_machine_learning import APIClient

wml_credentials={
```

```
    "url":"https://us-south.ml.cloud.ibm.com",

    "apikey":"ARfcmDqbH5jx8YhSBmTBtKJQH820oh39jr17z0ueJKFB"

}

client=APIClient(wml_credentials)

# In[81]:

client=APIClient(wml_credentials)

# In[82]:

def guid_from_space_name(client,space_name):

    space=client.spaces.get_details()

    return(next(item for item in space['resources'] if
item['entity']['name']==space_name)['metadata']['id'])

# In[83]:

space_uid=guid_from_space_name(client,'imageclassification')

print("Space UID = " +space_uid)

# In[84]:

client.set.default_space(space_uid)

# In[85]:

client.software_specifications.list()

# In[88]:

software_spec_uid=client.software_specifications.get_uid_by_name("tensorflow_rt22.1-
py3.9")

software_spec_uid

# In[89]:

model_details=client.repository.store_model(model='image-classification-
```

```
model_new.tgz',meta_props={

    client.repository.ModelMetaNames.NAME:"CNN",

    client.repository.ModelMetaNames.TYPE:'tensorflow_2.7',

    client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_spec_uid

})

model_id=client.repository.get_model_uid(model_details)

# In[90]:

print(model_id)

# In[91]:

client.repository.download(model_id,'gestureModel.tar.gz')
```

Testing model IBM:

```
# In[1]:

from tensorflow.keras.models import load_model

import keras

import numpy as np

from IPython.display import Image

model=load_model("gesture.h5")

# In[2]:

img=keras.utils.load_img("dataset/test/1/1.jpg",grayscale=True,target_size=(64,64))

img

# In[3]:

x=keras.utils.img_to_array(img)

x=np.expand_dims(x,axis=0)
```

```
pred=model.predict(x)

pred=pred.astype(int).tolist()

pred=pred[0]

pred

# In[4]:

index=['0','1','2','3','4','5']

for i in range(len(pred)):

    if(pred[i]==1):

        print(index[i])

# In[5]:

for i in range(0,6):

    for j in range(0,5):
img=keras.utils.load_img("dataset/test/"+str(i)+"/"+str(j)+".jpg",grayscale=True

                                 ,target_size=(64,64))

        display(Image(filename="dataset/test/"+str(i)+"/"+str(j)+".jpg") )



        x=keras.utils.img_to_array(img)

        x=np.expand_dims(x,axis=0)

        pred=model.predict(x)

        pred=pred.astype(int).tolist()

        pred=pred[0]

        for k in range(len(pred)):

            if(pred[k]==1):

               print(index[k])
```

Download Model in your local system:

```
# In[1]:

from ibm_watson_machine_learning import APIClient

wml_credentials={

    "url":"https://us-south.ml.cloud.ibm.com",

    "apikey":"ARfcmDqbH5jx8YhSBmTBtKJQH820oh39jr17z0ueJKFB"

}

client=APIClient(wml_credentials)

# In[2]:

client=APIClient(wml_credentials)

# In[3]:

def guid_from_space_name(client,space_name):

    space=client.spaces.get_details()

    return(next(item for item in space['resources'] if
item['entity']['name']==space_name)['metadata']['id'])

# In[4]:

space_uid=guid_from_space_name(client,'imageclassification')

print("Space UID = " +space_uid)

# In[5]:

client.set.default_space(space_uid)

# In[6]:

client.repository.download("c507de1d-99bd-4597-a76b-
9cd42ebcc101",'gestureModel.tar.gz')
```

8.TESTING:

8.1 TEST CASE :

S.No	Cases	Pass/Fail
1.	Input image data	Passed
2.	Upload data	Passed
3.	Model integration	Passed
4.	Prediction	Passed
5.	Nutrition values	Passed

8.2 USER ACCEPTANCE TESTING:

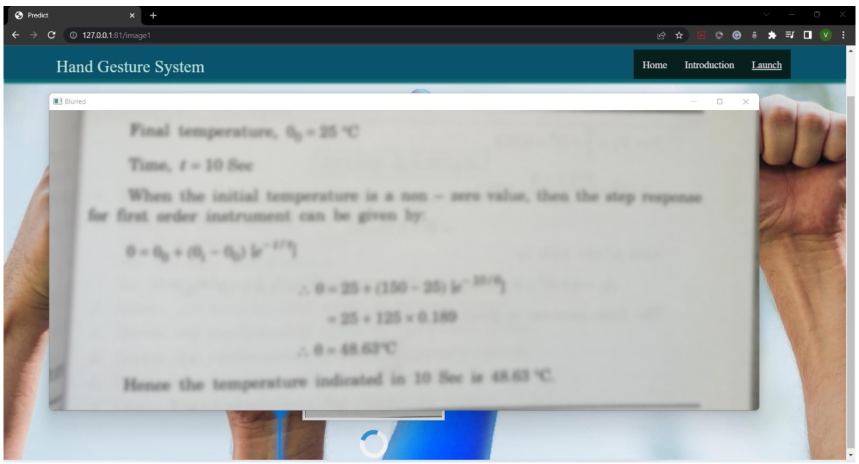
S.No	Cases	Yes/ No
1.	Keyword driven	Yes
2.	Responds in manually drafted rules	Yes
3.	Conversational Paradigm	Yes
4.	Learns from real interactions	No
5.	Training via historical data	No
6.	Has decision-making skills	No

9.RESULT:

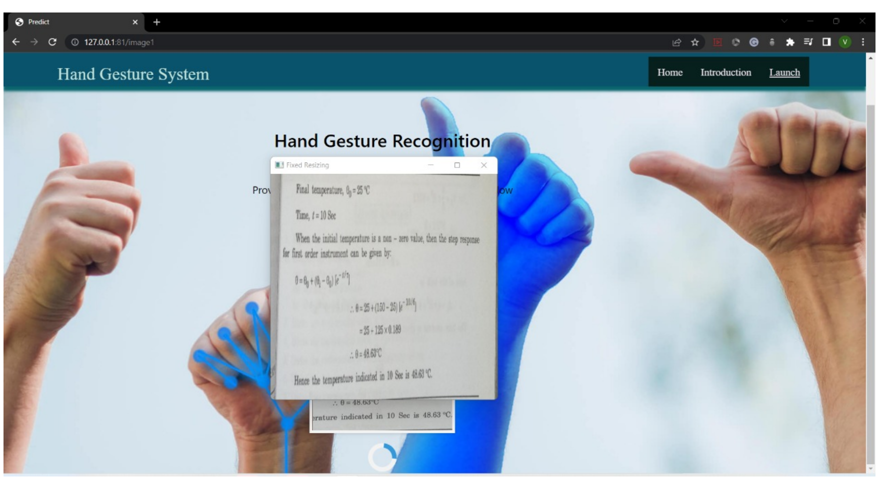
9.1 PROJECT METRICS:

Through this project we found that we can maintain the sterility of an operation theatre, etc by using hand based gesture tools to browse the images obtained.

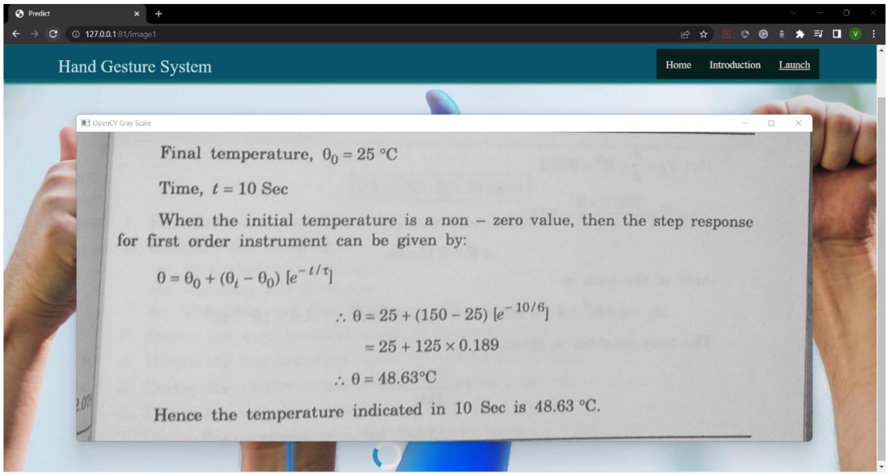
Hand gesture 3 - Blurred image



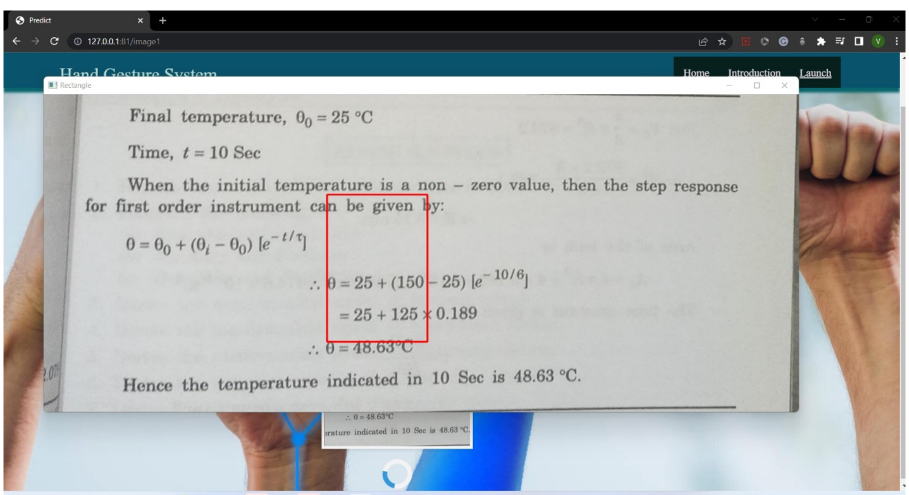
Hand gesture 4 - Resized image



Hand gesture 5 - OpenCV image



Hand gesture 0 - Drawing Rectangle over an image



10. ADVANTAGES & DISADVANTAGES:

Advantages :

1.As mentioned in Problem Statement, The Images can be viewed, Zoom, operated Without using Keyboard or Mouse for achieving Sterile Browsing.

2.Not only Radiology Images, But also Other Image Files can be uploaded and Viewed.

3.Can be easily implemented on a low powered processor.

4. Works in Both Linux and Windows.

Disadvantages :

1.Light position should be correctly positioned in order to improve accuracy.

2.Sometimes it will stop automatically after a certain period of time on running. In order is Continue, there should be some Input in the command prompt (Terminal, PS).

11. CONCLUSION:

So the main objective of our Project is to create an application in which the user uses their hands to give input to the system to browse Images. This is performed in order to achieve Sterile Browsing. Hence The application is built in Industrial method by following Agile methodology to understand the basic concepts of Agile methodology, and programming concepts of Python, Tensorflow . And also train and implement the model on IBM Cloud etc.

12. FUTURE SCOPE:

1. In future, certain disadvantages can be removed, and achieve full Sterile Browsing.

2.Might be implemented not only for sterile browsing but also for other applications.

3.Can be implemented on AR/VR 3D Tracking Technologies.

13.APPENDIX:

GITHUB LINK:

https://github.com/IBM-EPBL/IBM-Project-30597-1660150112

VIDEO DEMONSTRATION LINK:

https://drive.google.com/file/d/1P-
I6C2um3saVB18zAo6xbLsbAHq30H0I/view?usp=share_link