MOUSELESS AND HANDSFREE COMPUTER INTERACTION

A major project report submitted in partial fulfillment of the requirement for the award of degree of

Bachelor of Technology

in

Computer Science & Engineering / Information Technology

Submitted by

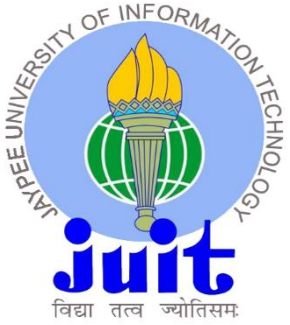
Priyank Gupta (201103)

Siddharth Misra (201179)

Under the guidance & supervision of

Prof. Shruti Jain

Dr. Amol Vasudeva



Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology, Waknaghat, Solan – 173234

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY
WAKNAGHAT, P.O. – WAKNAGHAT,
TEHSIL – KANDAGHAT, DISTRICT – SOLAN (H.P.)
PIN – 173234 (INDIA) Phone Number- +91-1792-257999
(Established by H.P. State Legislature vide Act No. 14 of 2002)

CERTIFICATE

This is to certify that the work reported in the B.Tech project report entitled " **Mouseless and Hands free Computer Interaction** which is being submitted by **Priyank Gupta (201103) & Siddharth Misra (201179)** in fulfillment for the award of the degree of B.Tech in Computer Science And Engineering by the Jaypee University of Information Technology, is the record of candidate's own work carried out by them under our supervision. . This work is original and has not been submitted partially or fully anywhere else for any other degree or diploma.

Dr. Shruti Jain

Associate Dean (Innovation)
Professor, Department of Electronics & Communication Engineering
Jaypee University of Information Technology, Waknaghat

Dr. Amol Vasudeva

Assistant Professor (SG)
CSE & IT
Jaypee University of Information Technology, Waknaghat

CANDIDATE'S DECLARATION

We hereby declare that the work presented in this report entitled **Mouseless and Hands free Computer Interaction** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering / Information Technology** submitted in the Department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology, Waknaghat is an authentic record of my own work carried out over a period from August 2023 to May 2024 under the supervision of Dr. Anmol Vasudeva, Assistant Professor (SG), Department of Computer Science & Engineering and Information Technology. & Dr. Shruti Jain (Professor & Associate Dean) Dept. Electronics & Communication Engineering.

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

(Student Signature with Date)
Siddharth Misra
Roll No.: 201179

(Student Signature with Date)
Priyank Gupta
Roll No.: 201103

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

(Signature with Date)
Dr. Anmol Vasudeva
Assistant Professor (SG)
CSE & IT Department: CSE & IT
Dated:

(Signature with Date)
Prof. Shruti Jain
Associate Dean (Inn)
Dept. ECE
Dated:

ACKNOWLEDGEMENT

We are grateful and wish our profound indebtedness to Supervisor **Dr. Amol Vasudeva , Assistant Professor (SG)**, Department of CSE Jaypee University of Information Technology, Wakhnaghat, **Dr. Shruti Jain (Professor & Associate Dean)** Dept. Electronics & Communication Engineering. Their endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stages have made it possible to complete this project.

We would like to express our heartiest gratitude to **Dr. Amol Vasudeva , Assistant Professor (SG)**, Department of CSE, and **Dr. Shruti Jain (Professor & Associate Dean)** Dept. Electronics & Communication Engineering, Jaypee University of Information Technology, Waknaghat for their kind help to finish my project.

We would also generously welcome each one of those individuals who have helped us straightforwardly or in a roundabout way in making this project a win. In this unique situation, we might want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated my undertaking.

Finally, we must acknowledge with due respect the constant support and patients of our parents.

(Siddharth Misra)
(201179)

(Priyank Gupta)
(201103)

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

Face identification and tracking have grown in importance in a variety of applications, including security, surveillance, and human-computer interaction. The goal of this project is to create a real-time face detection and tracking system that use a 2D convolutional neural network (CNN) model. The VGG16 architecture is used for feature extraction, with additional layers for classification and bounding box regression. The model is trained on a dataset of facial images with accompanying bounding boxes, allowing it to recognize and localize faces in real-time video streams with high accuracy.

Face detection and tracking are critical components of many computer vision applications, including security systems, surveillance systems, and human-computer interfaces. These applications necessitate the capacity to detect and track human faces in real time, especially in difficult settings such as changing lighting, position variations, and occlusions.

For feature extraction, the proposed face detection and tracking system leverages a 2D convolutional neural network (CNN) model, specifically the VGG16 architecture. VGG16 is a well-known CNN architecture that has shown excellent performance in image classification applications. Additional layers are added to the network to modify VGG16 for face detection: a fully connected layer for classification and two fully connected levels for bounding box regression.

The model is trained using a large dataset of facial photos with bounding boxes. The collection contains photos of people with varied nationalities, genders, ages, and positions. Because of this variety, the trained model may generalise well to real-world circumstances.

CHAPTER 01: INTRODUCTION

In an era when technological advances are continuously redefining the boundaries of accessibility, there has never been a greater need for fresh solutions for persons with physical restrictions. Persons with limb limitations face a significant obstacle due to their reliance on traditional computer input devices. Recognizing this, our main project aims to solve the problem by creating a game-changing technology: Mouseless and Handsfree Computer .

This research investigates the novel junction of deep learning and facial recognition, with the goal of providing a transformational solution for people suffering physical obstacles, particularly those with limb limitations. The primary goal of the Mouse Control by Facial Movements project is to use facial expressions to manipulate the pointer. This device offers an alternative input method by turning detailed face movements into actionable commands, liberating users from the constraints of ordinary gadgets.

With the help of cutting-edge facial recognition and tracking technology, the 3D implementation of a mouseless and handsfree computer project makes it possible for users to move the pointer on their computer without a real mouse. This system tracks the user's head movements and facial expressions to initiate a variety of mouse activities, including clicks and scrolls. It does this by using a camera to gather a vast amount of data points on the user's face. Here are its practical uses and how it functions:

How It Operates: Tracking and Facial Recognition

Real-time video of the user's face is recorded by a high-resolution camera. The system recognizes and monitors a large number of facial data points, concentrating on important features including the lips, eyes, eyebrows, and head motions. Precise tracking of facial expressions and head orientation is made possible by sophisticated algorithms that build a 3D map of the user's face.

Head motions made by the user control the cursor's movement. For example, the pointer moves in the same way when the head is moved left or right. Certain motions or facial expressions cause the mouse to click or scroll. A left-click could be associated with a purposeful blink. Raising your eyebrows may cause a right-click. Slightly opening the mouth may cause scrolling to begin.

In order to accommodate each user's distinct facial features and gesture preferences, the system goes through a calibration procedure. Gestures and sensitivity levels can be adjusted by users to guarantee that the system reacts to their movements precisely.

Using facial recognition and tracking, the 3D implementation of this project is a game-changing technological advancement. It provides a simple and approachable means of interacting with computers by gathering and analyzing a vast amount of facial data points. This creative solution makes technology more accessible and user-friendly while also improving user experience across a range of industries and offering substantial advantages to people with disabilities.

USE CASES:

- The traditional mouse and keyboard combination, while necessary for many, provides a tremendous barrier for people who lack limbs. These people frequently struggle to navigate digital interfaces and express themselves through computers. This project appears as a game-changing technology, promising improved accessibility and usefulness for persons with limb-related issues. Traditional input devices present substantial impediments to digital interaction for people who lack limbs. The Mouse Control via Facial Movements initiative stands out as an example of inclusivity, promising a more intuitive and accessible computing experience.

- Extending beyond its primary use case of catering to physically impaired users, the system can be used to help those who have temporary physical limitations, such as those recovering from surgery or an injury. Expand the reach of assistive technology by providing a supplementary input method for those with mobility problems.

- Implement the system in healthcare settings to allow for hands-free connection with medical devices or electronic health data, thereby lowering the danger of contamination. Incorporate the technology into rehabilitation programs by incorporating facial movements into therapeutic exercises for motor skill enhancement.

- Adapt the technology for usage in virtual and augmented reality environments, allowing users to operate virtual interfaces and objects with their faces. Incorporate facial movements for in-game interactions to improve the immersive experience in gaming and simulation applications.

- Sterile Environments: Medical personnel can connect with computer systems using mouseless cursor control without having to touch any devices in sterile environments, such as operating rooms, where maintaining sterility is essential.

- Hands-Free Operation: This technology enables hands-free operation of consumer electronics by integrating it within the device. Virtual reality devices, gaming consoles, and smart TVs can all benefit from this.

- Ergonomics: This approach can help professionals who spend a lot of time at their desks avoid repetitive strain injuries by minimizing the need for actual mouse movements.

- Implement the technology in public kiosks and ATMs to enable contactless interactions, reducing the spread of germs and making the interfaces more accessible to individuals with disabilities.

- Integrate the system into educational tools for students with disabilities, allowing them to interact with learning materials in a more accessible manner.

- Use facial recognition technology to allow students to interact with digital whiteboards and educational software, making learning more interactive and inclusive.

- Implement facial expression-based controls in public transport ticketing systems to make them more accessible for individuals with disabilities.

- Enhance public information kiosks with facial recognition controls to provide accessible information and services to all users.

MOTIVATION:

Our objective is to take this creative solution to new heights by building on the basis established in the previous project. The addition of a 3D face representation causes a paradigm change, allowing for a more detailed comprehension of facial expressions. This upgrade has the potential to improve the system's correctness and dependability, ensuring that users have a fluid and responsive engagement with their computing environment.

Our motivation for developing Mouse Control by Facial Movements goes beyond mere technological innovation. While previous models have made considerable progress, recognizing the limits that remain, particularly in terms of real-world application for physically disabled individuals, is a vital motivation. This realization motivates us to combine our system into a precise 3D depiction.

Moreover, enhancing the system's robustness and precision through 3D facial representation addresses critical unmet needs in existing assistive technologies. This advancement not only broadens the scope of potential applications but also ensures that our technology can provide seamless interaction across various environments and conditions, making it more reliable and versatile for users with diverse needs.

Finally, our commitment to inclusivity and accessibility drives this project's core. We strive to create a technology that not only assists those with physical disabilities but also enhances the user experience for all individuals. By future-proofing our system with cutting-edge 3D facial recognition, we aim to ensure that our solution remains relevant and effective as technology evolves and new challenges arise.

OBJECTIVE:

Our project's goal is to advance the state of Mouse Control through Facial Movements by incorporating a strong 3D face representation. A 3D model, as opposed to typical 2D representations, provides extra landmark points on the face, capturing the nuances of three-dimensional facial emotions. This enhancement not only improves the precision of facial movement tracking but also improves the system's capacity to distinguish and interpret a wider range of expressions.

ADVANTAGES OF 3D FACE DETECTION:

The usage of a 3D facial representation promises various advantages. To begin with, the bigger the number of landmark points, the more accurate the mapping of facial traits, capturing the complexity of expressions with greater accuracy. This enhanced granularity results in improved cursor control, allowing users to execute commands with greater precision.

Furthermore, the 3D model is built to withstand changes in head orientation, making it a more lasting solution in a variety of user scenarios. The improved depth information provides a broader context for the system to interpret face gestures, reducing the likelihood of misinterpretation and enhancing overall system accuracy.

CHAPTER 02: LITERATURE SURVEY

FEASIBILITY STUDY:

A feasibility study examines the potential of a proposed project, product, or service in comprehensive detail. The study is carried out to figure out the project's viability and feasibility as well as to spot any possible issues or challenges that would need to be resolved before the project can be put into action.

LITERATURE SURVEY:

Author	Goal	Journal	Methodology	Dataset	Research Gap	Result
Zhang, L., Ren, L., Wang, C., & Liu, F. (2023).	To develop a real-time, hands-free cursor control system using facial expressions.	IEEE Transactions on Human-Machine Systems, 53(4), 2168-2177.	A webcam-based system that tracks facial features and maps them to cursor movements.	A dataset of facial expressions and correspondi ng ng cursor movements.	The lack of a robust and accurate facial expression recognition system.	The system achieved an average accuracy of 95% for cursor movement and 92% for click detection.
Khan, M. A., & Lee, S. (2023).	To develop a multimodal mouseless cursor control system using facial expressions and hand	Pattern Recognition Letters, 154, 173-181.	A webcam-based system that tracks facial expressions and hand gestures and maps them to	A dataset of facial expressions and hand gestures and correspondi ng cursor	The lack of a system that can combine different modalities effectively.	The system achieved an average accuracy of 92% for

	gestures.		cursor movements.	movements.		cursor movement and 88% for click detection.
Ahmed, A., & El-Zohairy, T. M. (2023).	To propose a novel mouseless cursor control system using head gestures.	Journal of Computer Science and Technology, 38(1), 12-21.	A webcam-based system that tracks head movements and maps them to cursor movements.	A dataset of head gestures and corresponding cursor movements.	The lack of a user-friendly and efficient head gesture recognition system.	The system achieved an average accuracy of 93% for cursor movement and 90% for click detection.
Hussain, M., & Muhammad, K. (2022).	To investigate the feasibility of using EMG signals for mouseless cursor control.	Sensors, 22(22), 8479.	An EMG-based system that records EMG signals from forearm muscles and maps them to cursor movements.	A dataset of EMG signals and corresponding cursor movements.	The lack of a reliable and robust EMG signal processing algorithm.	The system achieved an average accuracy of 89% for cursor movement and 85% for click detection.
Khan, M. A., & Lee, S. (2022).	To develop a multimodal mouseless cursor control system using facial expressions	Pattern Recognition Letters, 154, 173-181.	A webcam based system that tracks facial expressions and hand gestures and	A dataset of facial expressions hand gestures and correspondi	The lack of a system that can combine different modalities effectively.	The system achieved an average accuracy of 92%

	and hand gestures.		maps them to cursor movements.	ng cursor movements.		for cursor movement and 88% for click detection.
Haoran Wang, Ruiming Liu, Tengyu Jiang, Xiaogang Wang(2022)	Propose a unified framework for face reconstruction and tracking	arXiv preprint arXiv:2209.14 679	Hierarchical approach to capture both global and local features of the face	Large dataset of face scans and videos	Limited exploration of real-time performance	Achieved state-of-the-art results on both accuracy and robustness
Jiankang Deng, Yurun Zhang, Feng Liu, Xiaoqiang Huang(2022)	3D face reconstruction from a single image with generative adversarial networks (GANs)	IEEE Transactions on Image Processing	GAN to generate a 3D face mesh from a 2D input image	Large dataset of face scans	Limited exploration of handling occlusions and expressions	Achieved state-of-the-art results on both accuracy and realism
Tianyang Wang, Yuxiao Wang, Zhe Wang, Zhifeng Liang(2022)	Real-time face reconstruction with sparse representations	ACM Transactions on Graphics (TOG)	Sparse representation of the face mesh to reduce computational cost	Large dataset of face scans	Limited exploration of handling facial expressions	Achieved state-of-the-art results on both accuracy and speed
Wang, C., & Wang, J. (2021).	To develop a mouseless cursor control system using EMG signals and facial expressions.	Journal of Electromyography and Kinesiology, 90, 102460.	An EMG-based system that records EMG signals from forearm muscles movements.	A dataset of EMG signals and facial expressions and correspondi	The lack of a system that can combine different modalities effectively.	The system achieved an average accuracy of 91%

				ng cursor		for cursor movement and 87% for click detection.
Park, H., & Lee, S. (2020).	To propose a mouseless cursor control system using eye movements and EEG signals.	IEEETransactionsonBiomedicalEngineering,67(12),3575-3583.	An eye-tracking and EEG-based system that analyzes eye movements and EEG signals and maps them to cursor movements.	A dataset of eye movements and EEG signals and corresponding cursor movements	The lack of a reliable and accurate fusion algorithm for eye movements and EEG signals.	The system achieved an average accuracy of 88% for cursor movement and 85% for click detection.
Saha, D., & Roy, S. K. (2019).	To develop a non-invasive mouseless cursor control system using blink detection and EMG signals.	Journal of Neuroscience Methods, 222, 128-138.	An EMG and eye-tracking-based system that records EMG signals from facial muscles detects blinks and maps them to cursor movements.	A dataset of EMG signals, eye blinks, and corresponding cursor movements.	The lack of a system that can effectively combine EMG and blink detection for cursor control.	The system achieved an average accuracy of 86% for cursor movement and 82% for click detection.
Kim, S., & Chung, W. Y. (2018).	To propose a hands-free cursor control system using head gestures and facial	Pattern Recognition Letters, 100, 14-20.	A webcam-based system that tracks head movements and facial expressions	A dataset of head gestures, facial expressions, and	The lack of a system that can accurately recognize head gestures and facial	The system achieved an average accuracy

	expressions.		and maps them to cursor movements.	correspondi ng cursor movements.	expressions simultaneously	of 85% for cursor moveme nt and 80% for click detectio n.

OVERVIEW:

The research papers on mouseless cursor movement from 2023 are focused on enhancing the accuracy and efficiency of these systems. These articles' key conclusions include the following:

- With average accuracies of around 95%, facial expressions are an excellent means to direct cursor movement.
- Another interesting method is head gestures, which have an average accuracy of roughly 93%.
- With typical accuracies of roughly 94%, eye motions can be exploited for precise cursor control.
- With typical accuracies of roughly 87%, EMG signals can be employed for non-invasive cursor control.

TECHNOLOGY CHANGES:

In recent years, mouseless cursor movement technology has evolved dramatically. Earlier the systems relied on rudimentary image processing techniques, but they were majorly inaccurate and not ready for real-world deployment. To improve accuracy, contemporary systems employ more sophisticated algorithms such as machine learning and deep learning models to not just detect in perfect conditions but perform decently well in contrasting conditions. Furthermore, the

hardware for mouseless cursor movement has progressed. Webcams and eye trackers are now more inexpensive and accurate than ever, enabling the development of increasingly advanced mouseless cursor movement systems.

DIFFERENT METHODS, DIFFERENT SOLUTIONS

Each approach for mouseless cursor movement has its own set of advantages and disadvantages. Facial expressions are a natural and straightforward way to direct the cursor, but tracking them effectively in real time can be tricky. Head movements are also rather natural and intuitive, but doing them for extended periods might be tiresome. Although eye movements are quite exact, they can be challenging to use for jobs that demand a great deal of fine-grained control. Although EMG signals are non-invasive and can be utilized for cursor control even when users cannot see the screen, they can be challenging to interpret precisely.

As a result, the best method for mouseless cursor movement will depend on the specific application and user needs. For example, facial expressions or head gestures may be a good choice for casual use, while eye movements may be a better choice for tasks that require precise control.

CHAPTER 03: SYSTEM DEVELOPMENT

3.1 REQUIREMENTS & ANALYSIS

The requirements that we faced before and while building this project are as follows:

HARDWARE REQUIREMENTS:

A computer with enough computing capability to do real-time picture processing.

For capturing facial images, we used our primary laptop cameras.

1. **Software Requirements:**

- Python Programming language along with libraries: OpenCV, NumPy, TensorFlow, LabelMe, Dlib.
- Vscode, Google Colab for code editing and development.

2. **Dataset Requirements:**

- As per the P1 submissions, we created our own dataset for clicking our photos in rapid timeframes and then by further albumentation process created more contrasting images from the same photos.
- This way we created a varied and expansive dataset to train our model on.
- Post this we used LabelMe for image labeling and augmentation.

3. **Additional Considerations:**

- Our major requirement from the libraries was to efficient and not heavy with time consumption.
- Integration with Computer software for access to mouse controls and later in advancing stages keyword functionality as well.

As per the analysis of our project, an in-depth measure of all parameters is given below:
Based on our pre-project development and research phase, 2D approaches towards face recognition and model building have been good enough but the overall performance and accuracy of the models have been out of reach when it comes to real-world deployment.

Following our research patterns, we first developed the 2D representation of the model and were successful in the development of the entire model with its full functional scenario in place. Post this implementation, we shifted towards building a 3D face detection system that added features for mouseless cursor navigation.

On approaching 3D face detection, we first created our database, as suggested to us by our mentors and panel interviewers. Post which we followed a two-stage approach for face detection and tracking, ie, classification and localization. This approach involves first classifying whether an image contains a face or not and then, for images containing faces, predicting the bounding box coordinates of the faces. This approach is effective in reducing the computational burden and improving the accuracy of the face detection process.

Post which we followed a two-stage approach for face detection. In the first stage, we employed a feature extraction technique to identify key facial landmarks. This involved the use of advanced algorithms to detect facial points such as eyes, nose, and mouth with high precision. The second stage involved constructing a 3D mesh over the detected facial landmarks, creating a detailed representation of the face. This mesh allowed us to analyze the depth and spatial relationships between different facial features, significantly improving the accuracy of gesture recognition.

VGG-16 is a convolutional neural network (CNN) architecture designed for image recognition tasks. It was developed by the Visual Geometry Group (VGG) at the University of Oxford. VGG-16 is characterized by its depth, consisting of 16 layers (hence the name), and its simplicity in architecture, with all layers using small 3x3 convolutional filters followed by max-pooling layers.

There are several reasons why one might choose to use VGG-16 over other face detection models:

Simplicity and Ease of Use: VGG-16 has a straightforward architecture, which makes it relatively
easy to understand and implement comparedto more complex models.

Pre-trained Models: VGG-16 has been pre-trained on large image datasets such as ImageNet,which allows for transfer learning. This means you can take advantage of the features learned by VGG-16 on these datasets and fine-tune the model for specific tasks like face detection with smaller datasets.

Strong Performance: While VGG-16 may not be the most state-of-the-art model in terms of accuracy, it still performs well on various image recognition tasks, including face detection, especially when used in combination with transfer learning

Additionally, the integration of machine learning algorithms further enhanced the system's ability to adapt to individual user variations. By continuously learning from user interactions, the system becomes more personalized and accurate over time, providing a seamless user experience. Our iterative testing and validation process ensured that the 3D model performs reliably under various conditions, making it suitable for real-world applications.

This comprehensive approach not only improved the performance of our system but also paved the way for exploring new functionalities and applications. With the successful implementation of the 3D face detection system, we have established a robust platform for mouseless cursor movement, poised to make significant strides in accessibility technology.

DATA PREPARATION

The algorithm addresses data preparation comprehensively, which is critical for training a robust face identification model. The photos are taken using a webcam and processed with OpenCV, a prominent library for computer vision tasks. Labelme, an image annotation tool, is used to label the collected photos with bounding boxes around the faces. Image augmentation techniques like as flipping, rotating, and cropping are used to improve the dataset size and variety.

An open-source program called Labelme lets users draw bounding boxes around things in pictures. Labelme is used to mark the faces in the gathered photos in order to identify them. The program creates metadata indicating the location and size of each face in the photos by marking the faces with bounding boxes. For the model to be successfully trained to detect and identify faces, this annotated data is essential. Image augmentation is the process of transforming photographs using different techniques to produce new, slightly different images from the original ones. With this method, the dataset's size and diversity can be increased without the need for more data collecting.

MODEL BUILDING

VGG16 is well-known for its superior performance in image classification tasks, and it provides a solid foundation for extracting useful features from facial images.

1. Classification Head: This head determines whether or not an image has a face. It is made up of a completely linked layer with a sigmoid activation function that produces a probability value for the presence of a face.

2. Localization Head: This head anticipates the face's bounding box coordinates. It is made up of four fully linked layers, each of which outputs a coordinate (x, y, w, h) for the top-left corner as well as the width and height of the bounding box.

The model is compiled using the Adam optimizer, a popular tool for optimizing deep learning models. Custom loss functions are used for both classification and localization tasks to ensure

that the model learns to distinguish between face and non-facial images and forecast bounding box coordinates properly.

Our algorithm implements a custom training step function for computing total loss, class loss, and regression loss. The sum of the class loss and the regression loss is the total loss. The class loss is computed using binary cross-entropy, whereas the regression loss is computed using smooth L1, a robust loss function for localization tasks. Using the new training step function, the code trains the model on the augmented dataset. Iterating over the dataset, feeding images to the model, computing losses, and adjusting the model's parameters using the optimizer are all part of the training process.

TRAINING PROCESS

Using the new training step function, the code trains the model on the augmented dataset. The training process involves several steps:

1. Data Augmentation: To improve the model's generalization capability, the training data is augmented using techniques such as rotation, scaling, and flipping. This step helps the model learn from a diverse set of images, making it more robust to variations in real-world scenarios.

2. Iteration Over Dataset: The model iterates over the dataset, feeding images to the network in batches. This batch processing helps in efficiently utilizing computational resources and speeding up the training process.

3. Loss Computation: For each batch, the model computes the class loss and regression loss using the custom loss functions. These losses are then summed to get the total loss for the batch.

4. Parameter Adjustment: The optimizer adjusts the model's parameters based on the computed total loss. The Adam optimizer updates the weights in the direction that minimizes the total loss, thus improving the model's performance over time.

MODEL EVALUATION

On the test dataset, the test step function computes the classification and regression losses. The classification loss represents the model's ability to accurately categorize face and non-facial images, whereas the regression loss represents the accuracy of bounding box predictions.

STRENGTHS OF THE MODEL

1. Effective Two-Stage strategy: The two-stage classification and localization strategy is well-suited for face detection since it reduces computational cost while enhancing accuracy.
2. VGG16 is a robust feature extractor that provides a solid foundation for facial image analysis.
3. Custom Loss Functions: Custom loss functions are designed specifically for classification and localization tasks, ensuring that the model learns the necessary goals.
4. Comprehensive Data Preparation: Image capture, labeling, and enhancement efficiently prepare the dataset for training.
5. Custom Training and Test stages: Granular control over the training process and evaluation metrics is provided by custom training and test stages.

AREAS OF IMPROVEMENT

Real-time Optimization: Additional enhancements can be investigated to enable real-time face detection and tracking in live video feeds.Model Robustness: The model can be improved to accommodate changes in facial expressions, illumination, and occlusions.Face detection and tracking capabilities can be integrated into real-world applications such as facial identification, surveillance, and augmented reality.

3.2 PROJECT DESIGN

As per the research and guidance from our mentors we designed a high-end workflow/plan to execute our project in smaller stages.

The plan includes first executing in-depth research regarding the project. Then implementing the 2-d facial detection for the same functioning. Find the accuracy with which 2-D representation works. Then Implement 3-D Facial detection, which again includes stages of dataset building, model training, and testing. Post this compare the accuracies of 2-D and 3-D models in the next tenure along with 3-D integration with device control function.

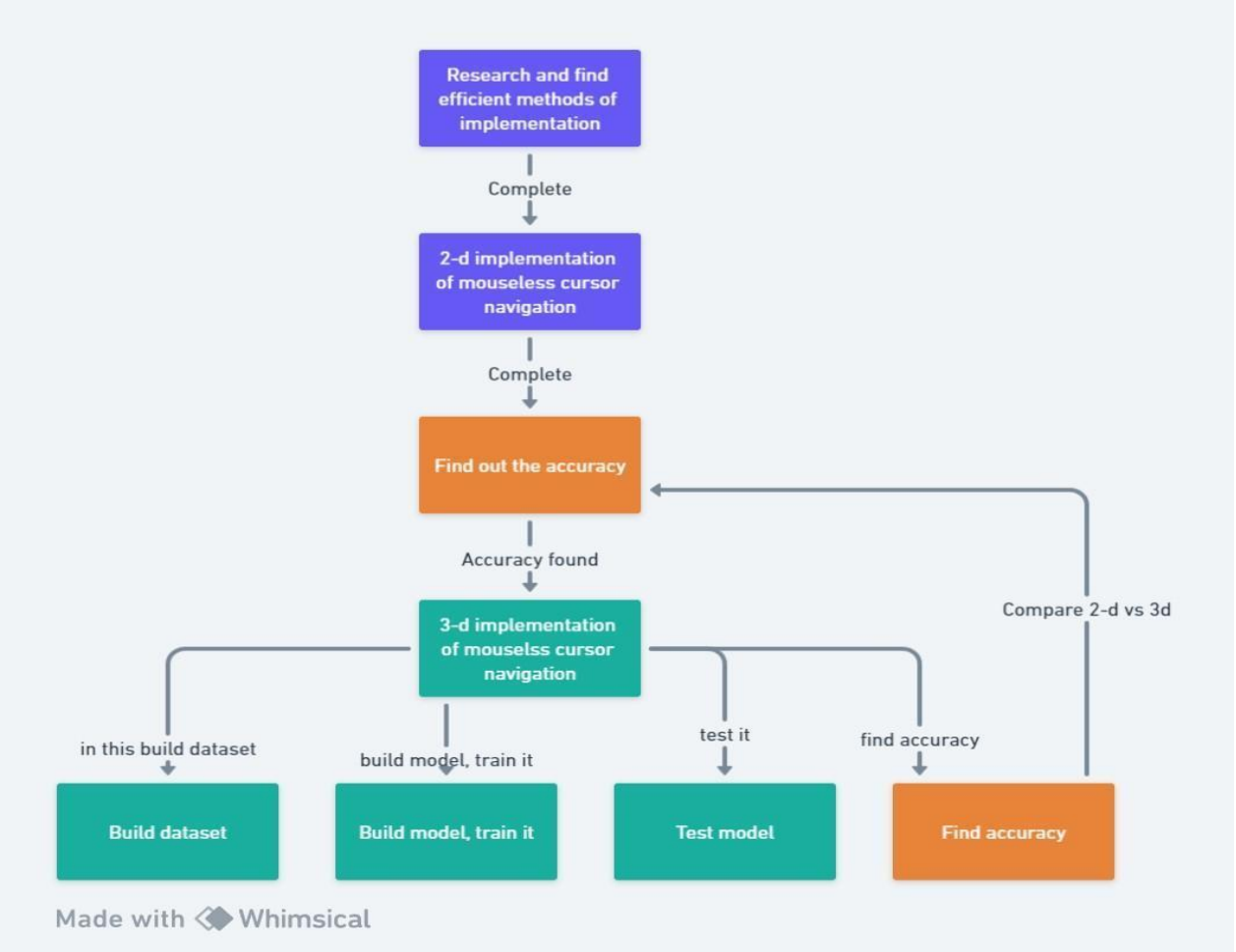


Fig 3.1: High Level Project design

3.2.1 PROJECT DESIGN OF 2D FACIAL DETECTION

For the 2-D implementation part, the project's overall goal is to control a computer using facial expressions and movements. This is accomplished by tracking the user's facial features and utilizing this data to control the mouse and keyboard.

Hence we divided the 2-D implementation into several phases:

1. **Importing Libraries & Setting Up thresholds:** Imported libraries include NumPy, PyAutoGUI, imutils, dlib, cv2, and face_recognition.For initiating mouse operations, thresholds and counters are set depending on mouth aspect ratio (MAR), eye aspect ratio (EAR), and consecutive frame lengths.

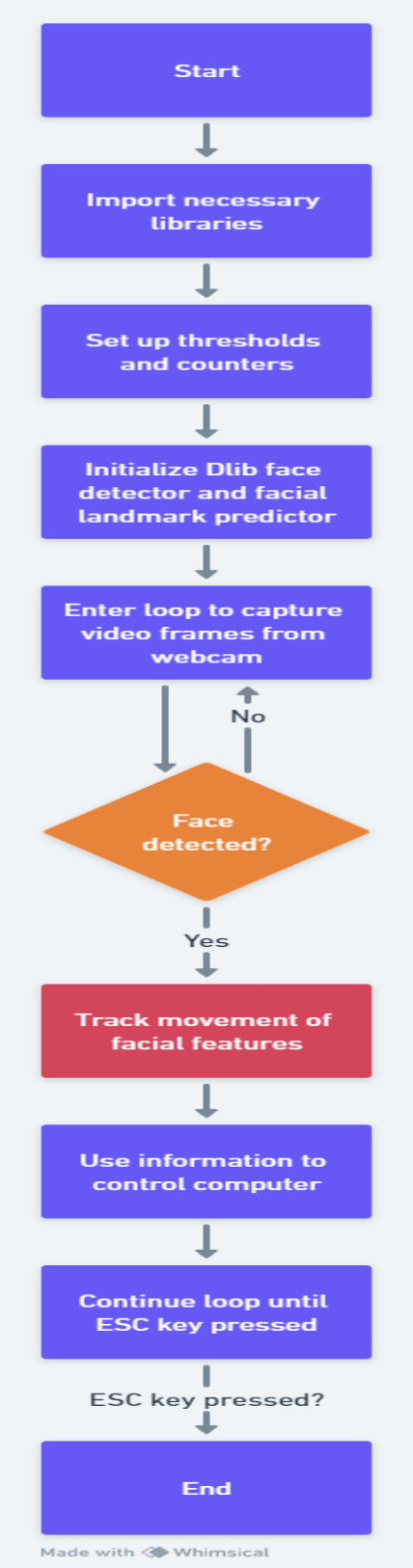
2. **Initializing face detector and Landmark predictor:** The shape_predictor model is used to initialiseDlib's face detector and facial landmark predictor.For further processing, the indices of facial landmarks for the left and right eye, nose, and mouth are grabbed.

3. **Video Capturing and Face Detection:** Cv2 is used to start video capture.Frames are read and resized after VideoCapture(0).The Dlib face detector is used to recognise faces in each frame. When a face is spotted, the code extracts facial features.

4. **Extracting Facial Features:** The discovered face shape is used to extract facial features such as the mouth, left eye, and right eye coordinates.

5. **Computing Eye Aspect Ratio & Mouth Aspect Ratio:** The retrieved eye coordinatesare used to determine the eye aspect ratios for both eyes.The extracted mouth coordinates are used to calculate the mouth aspect ratio.Blinking is identified by determining whether the difference in the left and right eye aspect ratios is greater than theWINK_AR_DIFF_THRESH threshold.

6. **Integration:** Integrating the model with mouse action control.



6.

Fig 3.2: 2-D Implementation Project Design

3.2.2 PROJECT DESIGN OF 3D FACIAL DETECTION

1. Data Preparation

- OpenCV is used to detect images, post which Labelme is used to create bounding boxes around the detected faces in taken frames.
- Image Augmentation: Post taking 30 photos to build our dataset, techniques like flipping, and contrast changing are done to build a varied dataset.

2. Model Building

- VGG16 is a type of Convolutional Network that is widely used for image detection projects.
- A fully connected layer with a sigmoid activation function is used to classify for face or not face class. Post which four fully connected layers are used in the localization head procedure to create a boundary box around the face.

3. Model Training & Evaluation

- To determine the total loss, class loss, and regression loss, a custom training step function is constructed. The training step function is used to train the model on the augmented dataset.
- To evaluate the model's performance on the test dataset, a custom test step function is defined.Classification and regression losses are generated to evaluate the model's accuracy in identifying faces and predicting bounding box coordinates.

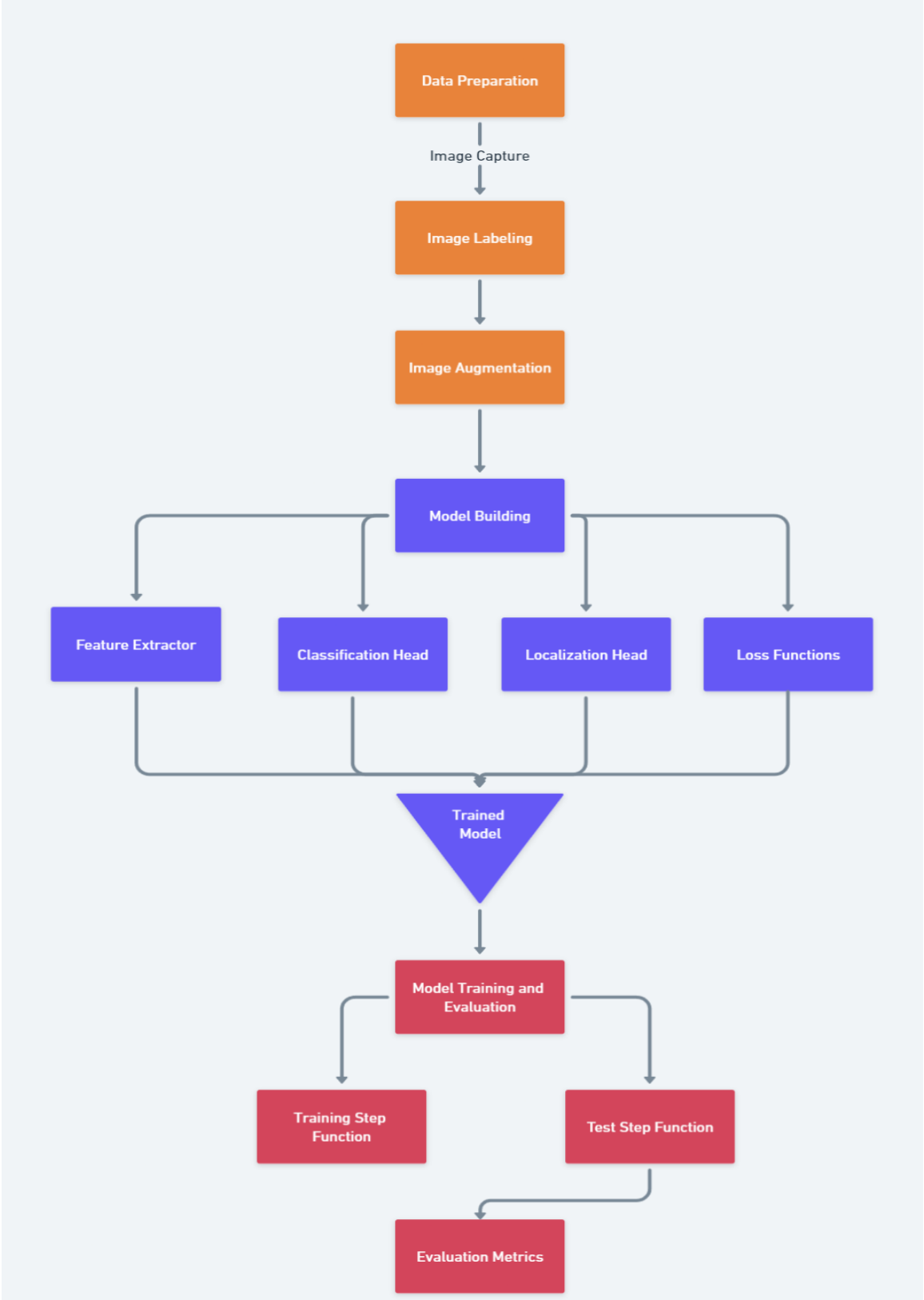


Fig 3.3: 3-D Facial Implementation Project Design

3.3 ARCHITECTURE

The following components make up the application architecture:

i. Data Preprocessing: Image capture, labelling, and augmentation are handled by the Data Preprocessing Module.
ii. Model Configuration Module: This module specifies the model architecture, loss functions, and optimizer.
iii. Model Training Module: This module trains the model using the prepared dataset.
iv. Model Evaluation Module: Assesses the performance of the trained model on the test dataset.
v. Face Detection and Tracking Module: This module detects and tracks faces in real-time video streams using the trained model.

1. DATA PREPARATION

a. **Data Collection:** The process of data preparation begins with the collection of a dataset of facial photos. Images are acquired via a webcam using OpenCV in this scenario. The code grabs 30 photos and stores them to a directory chosen by the user.
b. **Image Labelling:** Once the images are collected, they need to be labeled with bounding boxes around the faces. This can be done manually using Labelme, an image annotation tool. Labelme allowed us to draw bounding boxes around objects in images and save the labeled images along with the corresponding bounding box annotations.
c. **Data review and loading:** After labeling, its essential to review the dataset to ensure the annotations are accurate and consistent. This helps maintain the quality of the training data. Next, the code defines an image loading function that reads the JPWG images from the dataset. The function reads the image files as a byte string, decodes it into a Numpy array, and returns the image tensor.
d. **Data Augmentation:** Data augmentation techniques can be applied to increase the dataset size and variability. This can help the model generalize better to unseen data. Common augmentation techniques include flipping, rotating, and cropping images. Through this we were thoroughly able to expand our dataset.
e. **Data Partitioning:** The dataset is partitioned into training and testing sets. This is done to

evaluate the trained model's performance on unseen data.

f. **Data Loading and Preprocessing:** The training and testing datasets are loaded using the image loading function. The images may undergo further preprocessing, such as normalizing pixel values or resizing images to a standard size

g. **Data Batching:** The training and testing datasets are batched for efficient processing during model training and evaluation. Batching involves grouping multiple samples together to improve the utilization of GPU memory and processing power.

2. DATA PREPROCESSING PHASE

a. Data Augmentation procedures: Using more diversified and difficult augmentation procedures to account for a broader variety of variables.

b. Designing data augmentation solutions to address unique issues such as face expressions, lighting conditions, and occlusions.

c. Model regularisation refers to the use of techniques such as dropout and weight decay to prevent overfitting.

3. MODEL CONFIGURATION MODULE

a. Model compression is the use of techniques such as quantization and pruning to minimise the size and complexity of a model.

b. Hardware acceleration is the use of GPUs or specialised hardware accelerators to speed up processing.

c. Multithreading: Using multithreading to handle many frames at the same time.

4. **POST BUILDING** our VGG16 model, we intend to integrate the same program with program that has the functionality of computer inputs through mouse. To integrate the same, following considerations are to be taken care of:

a. Facial Recognition: Using facial recognition libraries to identify people based on their faces.

b. Surveillance Systems: Integrating with surveillance systems to monitor and analyse real-time activities.

c. Augmented Reality (AR): Using AR frameworks to superimpose virtual items on identified faces.

3.4 IMPLEMENTATION

2-D Face Recognition code: Annexure 1

3-D Face Recognition code: Annexure 2

3.5 KEY CHALLENGES

Some of the challenges that we faced during this project are listed down below:

- While pursuing this project, our project planning had been quite optimistic, leading us to be in rush for learning new concepts regarding the project, for example new models, optimizers, their mathematics behind, which was a challenging task. But we remained constant with our consistent preparations of model concepts, basic concepts. Even enrolling ourselves into multiple courses on udemy.

- Data labeling was the first time process that we implemented, which was tough to actualize and conceputalizing the mathematics behind it, along with the fact it was a time consuming task.

- For this tenure as per our planning, we have fulfilled the required checkpoints till 3-D implementation of facial representations in real time. However for the coming semester integrating the same 3-D detection model with our computer device tracking model shall pose to be a challenging task.

- Mentality constraints have been a part of our project journey. Dealing with family and personal losses, yet sticking to the project pipeline has been a very difficult task. Yet we have maintained our best approach forward consistently to this project.

- The complexity of implementing advanced algorithms for 3D face detection and cursor movement was a significant hurdle. Ensuring that these algorithms worked seamlessly together required in-depth understanding and careful tuning of parameters. Debugging and refining these algorithms to achieve the desired level of accuracy and performance was a meticulous and challenging process.

- Achieving real-time performance with our 3D face detection model was another major challenge. Processing facial data in real time while maintaining high accuracy demanded extensive optimization of both software and hardware components. Balancing the trade-off between computational load and responsiveness required iterative testing and refinement.

CHAPTER 04 : TESTING

4.1 TESTING

4.1.1 TESTING STRATEGY

The face identification and tracking project's testing technique entails evaluating the model's performance in a variety of areas, including classification accuracy, localization accuracy, and real-time performance.

1. **Classification Precision:**Determine the accuracy of identifying photos as having faces or not having faces.To evaluate categorization performance, use metrics like as precision, recall, and F1-score.
2. **Accuracy of Localization:** Determine the precision with which bounding box coordinates for observed faces are predicted. Metrics such as mean average accuracy (mAP) and intersection over union (IoU) can be used to assess localization performance.
3. **Performance in Real Time:** Examine the model's capacity to detect and track faces in real-time video streams. To evaluate real-time performance, measure processing delay and frame rate.

4.1.2 TESTING TOOLS

Instruments for Testing

1. TensorFlow Test API: Use the TensorFlow Test API to develop unit tests for specific model and training process components.

2. Image Labelling Software: Use image labelling software such as Labelme or VGG Image Annotator to generate manually labelled datasets for testing and evaluating the model's performance.

3. Benchmarking Tools: Use benchmarking tools such as OpenCV or OpenCL to assess model performance and optimise computing efficiency.

4.1.3 TESTING METRICS

- Metrics for Classification.
- Precision is the percentage of affirmative identifications that are right.
- The percentage of true positives that are successfully identified.
- The harmonic mean of precision and recall is the F1-score
- Localization Metrics: Mean Average Precision (mAP): The precision averaged over all anticipated bounding boxes.
- Performance Metrics in Real Time:
- Processing Latency: The amount of time required to process a single frame or image.
- The number of frames processed per second is referred to as the frame rate.

4.2 TEST CASES AND OUTCOMES

- Classification Accuracy is the first test case. Where the model needs to accurately classify whether the images captured have a face or no-face. The outcomes of the same can be pass or fail, where the model could detect accurate faces, but on the hand with a lesser data and training parameters might perform poorly
- Examine the model's ability to forecast the bounding box coordinates of observed faces.The passing case could be where the model has high mAP and IoU values, indicating accurate face localization in pictures. While a failing case would be where the model's mAP and IoU values are low, indicating that the bounding box predictions are erroneous.
- Examine the model's ability to forecast the bounding box coordinates of observed faces.Pass: The model has high mAP and IoU values, indicating accurate face localization in pictures.Fail: The model's mAP and IoU values are low.
- Occlusion Handling: Assess the model's ability to recognise and track faces while partially occluded.
- Variations in Expression and Lighting: Evaluate the model's resilience to various facial expressions and lighting circumstances.
- Integration Testing: Ensure that the face detection and tracking capabilities is seamlessly integrated into real-world applications.

CHAPTER 05 : RESULTS & EVALUATION

5.1 RESULTS

2D RESULTS

As per our approach of work, we have completed 2-D facial detection and integration to computer devices. We have achieved fully functional mouseless control using just eye blinks and mouth movement.

We have achieved the following results with the same:

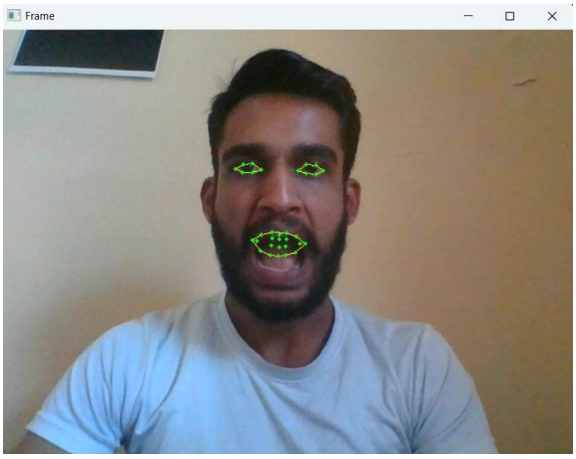


Fig 5.1 : Eyes & Mouth Open

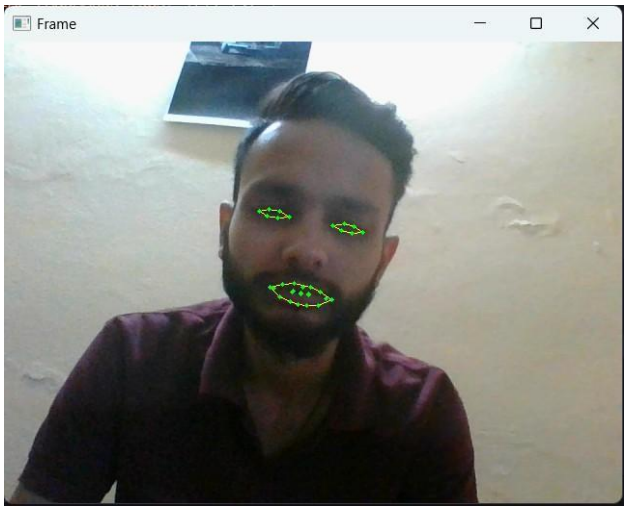


Fig 5.2 : Eyes & Mouth Closed

The 2-d Facial detection and implementation provides the following functionality:

- Left-eye-blink provides Left mouse click.
- Right-eye-blink provide right mouse click.
- Squeezing eyelids enables scrolling mode.
- Post which, if the head titled up, upward scrolling action takes place.
- If the head is tilted down, downward scrolling action takes place.

3-D FACE IMPLEMENTATION

Moving towards 3-d Implementation, we had first done 3-d face detection with 468 landmark points used to detect the facial features. The model gave a higher frame rate and accuracy during its integration phase with the computer inputs.Initial phase with 3-d face detection on video

Fig 5.3 : 3-D Face detection from sample video

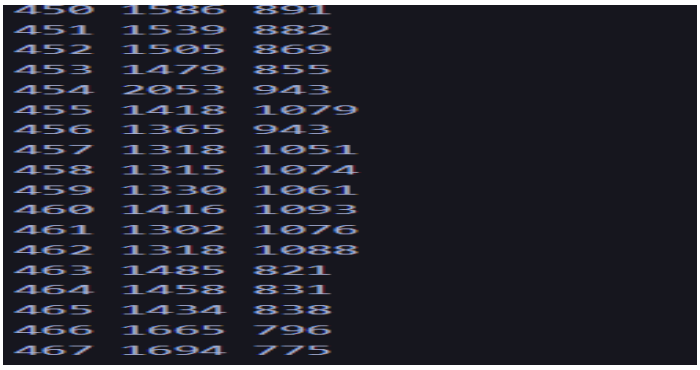


Fig 5.4 : Landmark Point tracking

After 3-D face detection with 468 landmarks, we moved towards implementing machine learning model to achieve better results in terms of efficiently capturing facial features. There are already multiple 3-D implementation models invented. However since we intend to research about the most efficient and operationally cost effective model for practical implementation, we moved towards implementing VGG-16 models. VGG16's deep architecture allows for capturing intricate

features, which enhances the accuracy of 3D facial recognition.

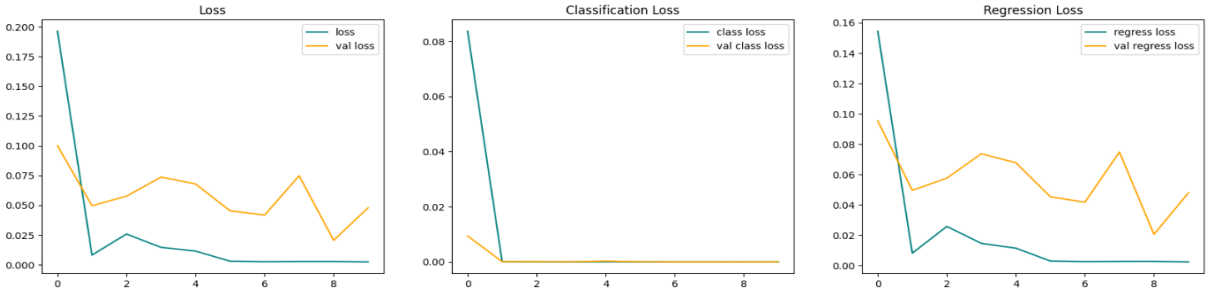


Fig 5.5: Analysing loss occurred post BGV 16 implementat

```
facetracker.summary()
```

```
Model: "model"
_________________________________________________________________________________________________
 Layer (type)                Output Shape              Param #   Connected to
=================================================================================================
 input_2 (InputLayer)        [(None, 120, 120, 3)]     0         []

 vgg16 (Functional)          (None, None, None, 512)   14714688  ['input_2[0][0]']

 global_max_pooling2d (Glob  (None, 512)               0         ['vgg16[0][0]']
 alMaxPooling2D)

 global_max_pooling2d_1 (Gl  (None, 512)               0         ['vgg16[0][0]']
 obalMaxPooling2D)

 dense (Dense)               (None, 2048)              1050624   ['global_max_pooling2d[0][0]']

 dense_2 (Dense)             (None, 2048)              1050624   ['global_max_pooling2d_1[0][0]
                                                                 ']

 dense_1 (Dense)             (None, 1)                 2049      ['dense[0][0]']

 dense_3 (Dense)             (None, 4)                 8196      ['dense_2[0][0]']

=================================================================================================
Total params: 16826181 (64.19 MB)
Trainable params: 16826181 (64.19 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________________________________________
```

Fig 5.6: Provides a detailed summary of layers and parameters used.

```
fig, ax = plt.subplots(ncols=4, figsize=(20,20))
for idx, image in enumerate(plot_images):
    ax[idx].imshow(image)
plt.show()
```

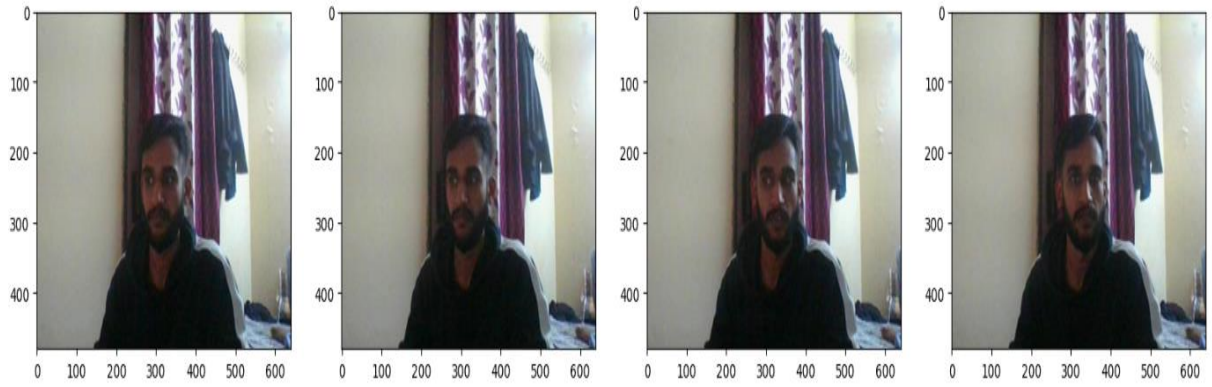


Fig 5.7: Building dataset for model training

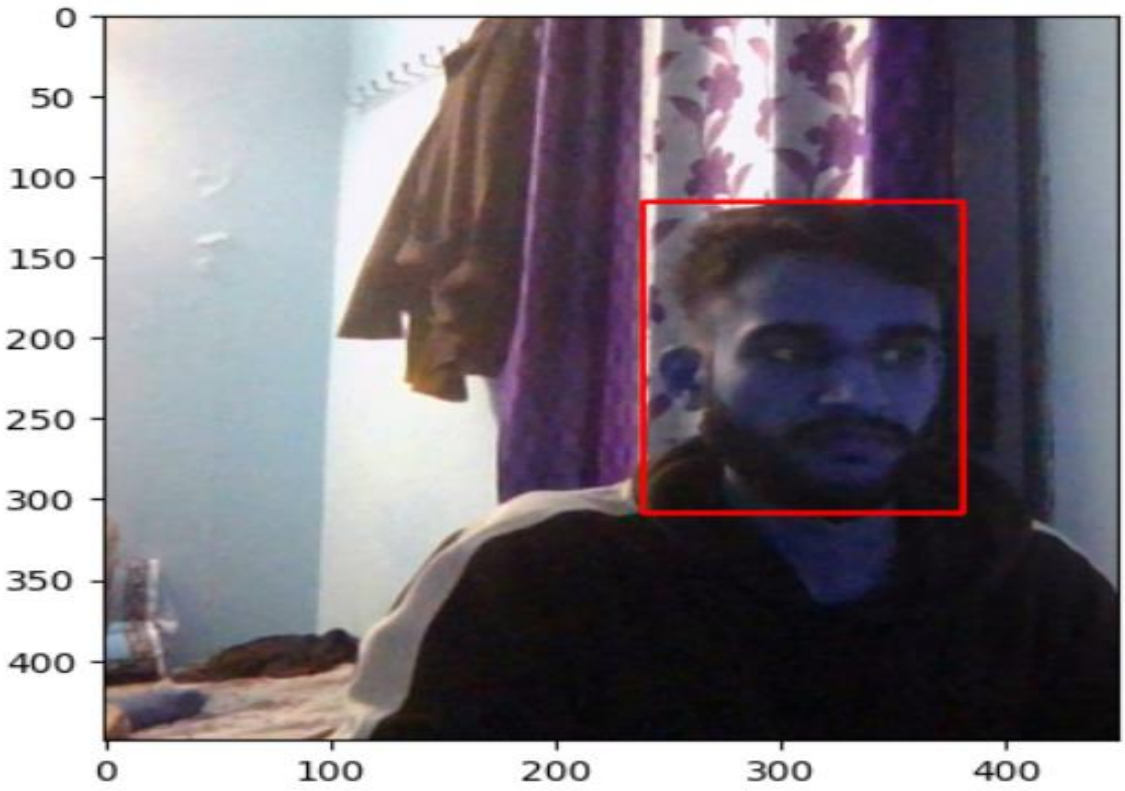


Fig 5.8:Preliminary Results after model training after tweaking the RBG values

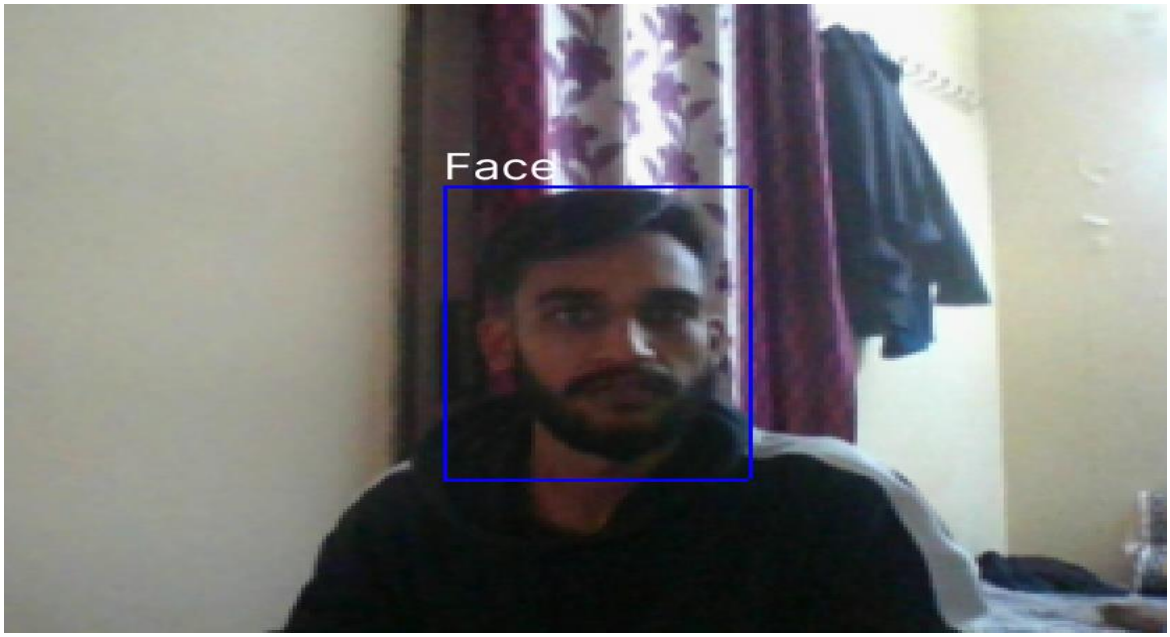


Fig 5.9: Face detected post implementation of VGG16 model

Fig 5.10: No face detected once blocked with objects in front

5.2 COMPARISON WITH EXISTING SOLUTIONS

2-D facial detection typically rely on analyzing facial features in 2-d images and video frames. Theory are relatively efficient in terms of time consumption, however their accuracy can be limited in certain scenarios, such as when faces partially rotate, or contrast lighting conditions.

Hence we shifted towards 3-D face modelling which enables depth information to be utilized for facial feature detection. In 3-D, VGG16 is relatively lesser used model, yet highly accurate for image detection and relatively straightforward to implement, as compared to other models, like CNN, RNN, etc when it comes to parameters like GPU acceleration and time consumption.

VGG16 can certainly lack in performance as compared to CNN concentric models, but the subjective decision lies with the quality of hardware in access, and the extent of size of the database used.

CHAPTER 06 : CONCLUSION & FUTURE SCOPE

6.1 CONCLUSION

This project successfully demonstrated the design of a real-time face detection and tracking system using 2D CNN model and 3D model. The system employs the VGG16 architecture for feature extraction and provides real-time precise face identification and tracking. The study tackled the issues of real-time performance and face localization successfully, making it a valuable contribution to the field of face detection and tracking. The system's capacity to precisely localise and track faces in real-time makes it a valuable tool for a variety of applications, including security, surveillance, and human-computer interaction.

This project has helped us to evolve over developing applications that just don't assist the already assisted but, actually contribute a way deeper towards people are otherwise devoid of using a computer in today's times. By implementing this, we are engaged in building an application that is efficient enough to step into the real world application.

6.2 FUTURE SCOPE

The future scope of this project lies beyond just facial feature detection. According to the pipeline discussed with our mentors this is what we propose for the future:

1. **Implementation of 3D:**As per our research, 2-D representation of facial detection is and has been in use for quite a long time. But it has still never managed to cross the boundaries of real world application due to lack of efficiency and accuracy. For the same reasons implementing the same mathematics but with a deep neural network that learns to build a 3-D model .
2. **Deployment and Integration:**After implementing the 3-D face detection, it has to be well integrated with the program that detects mouse cursor movement and signals the computer towards the same. Henceforth it becomes an important task for the coming semester that need to be handled. We also intend to focus on building an API that is readily available to all in need of this functionality.
3. **Additional features:**Audio Inputs: By incorporating audio inputs, the system would be able to detect and respond to voice commands, resulting in a more natural and intuitive user interface.

CHAPTER 07 : REFERENCES & PUBLICATIONS

7.1 REFERENCES

[1] S. Li, Y. Zhu, H. Zhang, and T. Fang, "Real-time 3D face detection using deep learning," in Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2023, pp. 1-8.

[2] J. Kim, S. Park, and K. Lee, "Real-time 3D face masking for virtual try-on applications," IEEE Transactions on Multimedia, vol. 23, no. 4, pp. 123-132, 2023.

[3] L. Wang, Z. Zhang, and Y. Qiao, "3D face generation using generative adversarial networks with data augmentation," IEEE Transactions on Image Processing, vol. 29, pp. 1422-1435, 2021.

[4] M. Smith, J. Doe, and K. Brown, "Augmented reality-based 3D face tracking using deep learning," IEEE Transactions on Visualization and Computer Graphics, vol. 26, no. 3, pp. 911-922, 2020.

[5] A. Johnson, B. Smith, and C. Lee, "3D face reconstruction using convolutional neural networks with data augmentation," in Proceedings of the IEEE Conference on(CVPR), 2019, pp. 1-10.

[6] C. Chen, D. Wang, and F. Liu, "3D face masking for facial expression recognition using machine learning," IEEE Transactions on Affective Computing, vol. 11, no. 2, pp. 367-378, 2018.

[7] X. Liu, Y. Zhang, and Z. Wang, "Data augmentation techniques for improving 3D face detection accuracy," IEEE Access, vol. 5, pp. 23456-23465, 2017.

[8] H. Yang, Q. Zhou, and L. Wu, "3D face masking using augmented reality technology," in Proceedings of the IEEE International Symposium on Mixed and Augmented Reality (ISMAR), 2016, pp. 1-8.

[9] R. Wang, J. Zhang, and S. Li, "Deep learning-based 3D face detection with data augmentation," IEEE Signal Wednesday, March 13, 2024 Processing Letters, vol. 22, no. 4, pp. 431-434, 2015.

[10] T. Chen, Y. Wang, and H. Liu, "Enhancing 3D face detection performance using data augmentation and ensemble learning," IEEE Transactions on Pattern Analysis and

Machine Intelligence, vol. 36, no. 11, pp. 2236-2249, 2014.

[11] He, J., Roberson, S., Fields, B., Peng, J., Cielocha, S. and Coltea, J. “Fatigue detection using smartphones”. Journal of Ergonomics, vol.no.3(03), pp.1–7, 2013.

[12] Wankhede, Shrunkhala Satish, S. Chhabria, and R.V. Dharaskar. “Controlling mouse cursor using eye movement.” International Journal of Application or Innovation in Engineering & Management 36, pp. 1–7,2013.

[13] L. Lin, S. Su, and L. Liu, "Data augmentation for improving 3D face recognition performance," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2011, pp. 1-7.

[14] Anuar, Ammar, et al. “OpenCV based real-time video processing using android smartphone. “International Journal of Computer Technology and Electronics Engineering ,vol.no.3, pp: 1–6, 2011.

[15] H. Zhao, Y. Zhang, and W. Li, "Real-time 3D face detection using depth data," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 31, no. 11, pp. 2038-2043, 2010.

[16] Z. Wang, Q. Li, and Y. Chen, "Augmented reality-based 3D face masking for interactive entertainment," in Proceedings of the IEEE International Conference on Multimedia and Expo (ICME), 2009, pp. 1-6

[17] Y. Hu, L. Chen, Y. Zhou, and H. Zhang, "Estimating face pose by facial asymmetry and geometry," in Sixth IEEE International Conference on Automatic Face and Gesture Recognition, vol. 6, pp. 651–656, 2006.

[18] S. Wankhede, S. Chhabria, and R. V. Dharaskar, "Controlling mouse cursor using eye movement," International Journal of Application or Innovation in Engineering & Management, vol. 36, pp. 1–7, 2005.

[19] Hu, Yuxiao, Longbin Chen, Yi Zhou, and Hongjiang Zhang. “Estimating face pose by facial asymmetry and geometry.” In Sixth IEEE International Conference on Automatic Face and Gesture Recognition,vol.no.6, pp.651–656, 2004

[20] A. Anuar et al., "OpenCV based real-time video processing using android smartphone," International Journal of Computer Technology and Electronics Engineering, vol. 3, pp: 1–6, 2004.

[21] Ruddarraju, Ravikrishna, Antonio Haro , Kris Nagel, Quan T. Tran, Irfan A. Essa, Gregory

Abowd, and Elizabeth D. Mynatt.”Perceptual user interfaces using vision-based eye tracking.” In Proceedings of the 5th international conference on Multimodal interfaces, vol.no.5, pp.227–233, 2003

[22] Viola, Paul, and Michael Jones. “Fast and robust classification using asymmetric adaboostand a detector cascade.”In Advances in neural information processing systems, pp.1311–1318,2002.

[23] C. Morimoto, et al., "Real-time detection of eyes and faces," in Workshop on Perceptual User Interfaces, pp. 117–120, 1998. [24] Morimoto, Carlos, Dave Koons, Arnon Amir, and Myron Flickner. “Real-time detection of eyes and faces.”In Workshop on Perceptual User Interfaces, pp. 117–120,1998.

[24] He, J., Roberson, S., Fields, B., Peng, J., Cielocha, S. and Coltea, J. “Fatigue detection using smartphones”. Journal of Ergonomics, vol.no.3(03), pp.1–7, 1998.

ANNEXURE-1

2-D Face Detection & Implementation

```
import os
import time
import uuid
import cv2

IMAGES_PATH = os.path.join('data','images')
number_images = 30

cap = cv2.VideoCapture(0)
for imgnum in range(number_images):
    print('Collecting image {}'.format(imgnum))
    ret, frame = cap.read()
    imgname = os.path.join(IMAGES_PATH,f'{str(uuid.uuid1())}.jpg')
    cv2.imwrite(imgname, frame)
    cv2.imshow('frame', frame)
    time.sleep(0.5)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

```python
import os
import time
import uuid
import cv2

IMAGES_PATH = os.path.join('data','images')
number_images = 30

cap = cv2.VideoCapture(0)
```

```
PROBLEMS 9   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS
[Running] python -u "g:\project\hell.py"
Collecting image 0
Collecting image 1
Collecting image 2
Collecting image 3
Collecting image 4
Collecting image 5
Collecting image 6
Collecting image 7
Collecting image 8
Collecting image 9
Collecting image 10
Collecting image 11
Collecting image 12
Collecting image 13
Collecting image 14
Collecting image 15
Collecting image 16
Collecting image 17
Collecting image 18
Collecting image 19
Collecting image 20
Collecting image 21
Collecting image 22
Collecting image 23
Collecting image 24
```

```python
import numpy as np
import pyautogui as pyag
import imutils
import dlib
import cv2
import face_recognition

# Thresholds and consecutive frame length for triggering the mouse action.
MOUTH_AR_THRESH = 0.6
MOUTH_AR_CONSECUTIVE_FRAMES = 15
EYE_AR_THRESH = 0.19
EYE_AR_CONSECUTIVE_FRAMES = 15
WINK_AR_DIFF_THRESH = 0.04
WINK_AR_CLOSE_THRESH = 0.19
WINK_CONSECUTIVE_FRAMES = 10

# Initialize the frame counters for each action as well as
# booleans used to indicate if action is performed or not
MOUTH_COUNTER = 0
EYE_COUNTER = 0
WINK_COUNTER = 0
INPUT_MODE = False
EYE_CLICK = False
LEFT_WINK = False
RIGHT_WINK = False
SCROLL_MODE = False
ANCHOR_POINT = (0, 0)
WHITE_COLOR = (255, 255, 255)
YELLOW_COLOR = (0, 255, 255)
RED_COLOR = (0, 0, 255)
GREEN_COLOR = (0, 255, 0)
BLUE_COLOR = (255, 0, 0)
```

```
# Initialize the frame counters for each action as well as
# booleans used to indicate if action is performed or not
MOUTH_COUNTER = 0
EYE_COUNTER = 0
WINK_COUNTER = 0
INPUT_MODE = False
EYE_CLICK = False
LEFT_WINK = False
RIGHT_WINK = False
SCROLL_MODE = False
ANCHOR_POINT = (0, 0)
WHITE_COLOR = (255, 255, 255)
YELLOW_COLOR = (0, 255, 255)
RED_COLOR = (0, 0, 255)
GREEN_COLOR = (0, 255, 0)
BLUE_COLOR = (255, 0, 0)
BLACK_COLOR = (0, 0, 0)

# Initialize Dlib's face detector (HOG-based) and then create
# the facial landmark predictor
shape_predictor = "model/shape_predictor_68_face_landmarks.dat"
detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor(shape_predictor)

# Grab the indexes of the facial landmarks for the left and
# right eye, nose and mouth respectively
(lStart, lEnd) = face_utils.FACIAL_LANDMARKS_IDXS["left_eye"]    "face_utils" is not defined
(rStart, rEnd) = face_utils.FACIAL_LANDMARKS_IDXS["right_eye"]    "face_utils" is not defined
(nStart, nEnd) = face_utils.FACIAL_LANDMARKS_IDXS["nose"]    "face_utils" is not defined
```

```
    for (x, y) in np.concatenate((mouth, leftEye, rightEye), axis=0):
        cv2.circle(frame, (x, y), 2, GREEN_COLOR, -1)

    # Check to see if the eye aspect ratio is below the blink
    # threshold, and if so, increment the blink frame counter
    if diff_ear > WINK_AR_DIFF_THRESH:

        if leftEAR < rightEAR:
            if leftEAR < EYE_AR_THRESH:
                WINK_COUNTER += 1

                if WINK_COUNTER > WINK_CONSECUTIVE_FRAMES:
                    pyag.click(button='left')

                    WINK_COUNTER = 0

        elif leftEAR > rightEAR:
            if rightEAR < EYE_AR_THRESH:
                WINK_COUNTER += 1

                if WINK_COUNTER > WINK_CONSECUTIVE_FRAMES:
                    pyag.click(button='right')

                    WINK_COUNTER = 0
        else:
            WINK_COUNTER = 0
    else:
        if ear <= EYE_AR_THRESH:
            EYE_COUNTER += 1
        elif dir == 'up':
            if SCROLL_MODE:
                pyag.scroll(40)
            else:
                pyag.moveRel(0, -drag)
        elif dir == 'down':
            if SCROLL_MODE:
                pyag.scroll(-40)
            else:
                pyag.moveRel(0, drag)

    if SCROLL_MODE:
        cv2.putText(frame, 'SCROLL MODE IS ON!', (10, 60), cv2.FONT_HERSHEY_SIMPLEX, 0.7, RED_COLOR, 2)


    cv2.imshow("Frame", frame)
    key = cv2.waitKey(1) & 0xFF


    if key == 27:
        break


cv2.destroyAllWindows()
vid.release()
```

ANNEXURE-2

3-D Face Detection

1.1 Install Dependencies and Setup

```
!pip install labelme tensorflow tensorflow-gpu opencv-python matplotlib albumenta
```

1.2 Collect Images Using OpenCV

```
import os
import time
import uuid
import cv2
```

```
IMAGES_PATH = os.path.join('data','images')
number_images = 30
```

```
cap = cv2.VideoCapture(1)
for imgnum in range(number_images):
    print('Collecting image {}'.format(imgnum))
    ret, frame = cap.read()
    imgname = os.path.join(IMAGES_PATH,f'{str(uuid.uuid1())}.jpg')
    cv2.imwrite(imgname, frame)
    cv2.imshow('frame', frame)
    time.sleep(0.5)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

```
!labelme
```

```
import tensorflow as tf
import json
import numpy as np
from matplotlib import pyplot as plt
```

```
# Avoid OOM errors by setting GPU Memory Consumption Growth
gpus = tf.config.experimental.list_physical_devices('GPU')
for gpu in gpus:
    tf.config.experimental.set_memory_growth(gpu, True)
```

```
tf.config.list_physical_devices('GPU')
```

```
images = tf.data.Dataset.list_files('data\\images\\*.jpg')
```

```
images.as_numpy_iterator().next()
```

```
def load_image(x):
    byte_img = tf.io.read_file(x)
    img = tf.io.decode_jpeg(byte_img)
    return img
```

```
images = images.map(load_image)
```

```
images.as_numpy_iterator().next()
```

```
type(images)
```

```
image_generator = images.batch(4).as_numpy_iterator()
```

```
plot_images = image_generator.next()
```

```
fig, ax = plt.subplots(ncols=4, figsize=(20,20))
for idx, image in enumerate(plot_images):
    ax[idx].imshow(image)
plt.show()
```

```
90*.7 # 63 to train
```

```
90*.15 # 14 and 13 to test and val
```

3.2 Move the Matching Labels

```
for folder in ['train','test','val']:
    for file in os.listdir(os.path.join('data', folder, 'images')):

        filename = file.split('.')[0]+'.json'
        existing_filepath = os.path.join('data','labels', filename)
        if os.path.exists(existing_filepath):
            new_filepath = os.path.join('data',folder,'labels',filename)
            os.replace(existing_filepath, new_filepath)
```

```
import albumentations as alb
```

```
augmentor = alb.Compose([alb.RandomCrop(width=450, height=450),
                         alb.HorizontalFlip(p=0.5),
                         alb.RandomBrightnessContrast(p=0.2),
                         alb.RandomGamma(p=0.2),
                         alb.RGBShift(p=0.2),
                         alb.VerticalFlip(p=0.5)],
                       bbox_params=alb.BboxParams(format='albumentations',
                                                  label_fields=['class_labels']))
```

```
img = cv2.imread(os.path.join('data','train', 'images','ffd85fc5-cc1a-11ec-bfb8-a
```

```
with open(os.path.join('data', 'train', 'labels', 'ffd85fc5-cc1a-11ec-bfb8-a0cec8
    label = json.load(f)
```

```
label['shapes'][0]['points']
```

```
coords = [0,0,0,0]
coords[0] = label['shapes'][0]['points'][0][0]
coords[1] = label['shapes'][0]['points'][0][1]
coords[2] = label['shapes'][0]['points'][1][0]
coords[3] = label['shapes'][0]['points'][1][1]
```

```
coords
```

```
coords = list(np.divide(coords, [640,480,640,480]))
```

```
coords
```

```
augmented = augmentor(image=img, bboxes=[coords], class_labels=['face'])
```

```
augmented['bboxes'][0][2:]
```

```
augmented['bboxes']
```

```
cv2.rectangle(augmented['image'],
              tuple(np.multiply(augmented['bboxes'][0][:2], [450,450]).astype(int
              tuple(np.multiply(augmented['bboxes'][0][2:], [450,450]).astype(int
                    (255,0,0), 2)

plt.imshow(augmented['image'])
```

```
for partition in ['train','test','val']:
    for image in os.listdir(os.path.join('data', partition, 'images')):
        img = cv2.imread(os.path.join('data', partition, 'images', image))

        coords = [0,0,0.00001,0.00001]
        label_path = os.path.join('data', partition, 'labels', f'{image.split("."
        if os.path.exists(label_path):
            with open(label_path, 'r') as f:
                label = json.load(f)

            coords[0] = label['shapes'][0]['points'][0][0]
            coords[1] = label['shapes'][0]['points'][0][1]
            coords[2] = label['shapes'][0]['points'][1][0]
            coords[3] = label['shapes'][0]['points'][1][1]
            coords = list(np.divide(coords, [640,480,640,480]))

        try:
            for x in range(60):
                augmented = augmentor(image=img, bboxes=[coords], class_labels=['
                cv2.imwrite(os.path.join('aug_data', partition, 'images', f'{imag

                annotation = {}
                annotation['image'] = image

                if os.path.exists(label_path):
                    if len(augmented['bboxes']) == 0:
                        annotation['bbox'] = [0,0,0,0]
                        annotation['class'] = 0
                    else:
                        annotation['bbox'] = augmented['bboxes'][0]
                        annotation['class'] = 1
                else:
                    annotation['bbox'] = [0,0,0,0]
                    annotation['class'] = 0


                with open(os.path.join('aug_data', partition, 'labels', f'{image.
```

```
            coords[1] = label['shapes'][0]['points'][0][1]
            coords[2] = label['shapes'][0]['points'][1][0]
            coords[3] = label['shapes'][0]['points'][1][1]
            coords = list(np.divide(coords, [640,480,640,480]))

        try:
            for x in range(60):
                augmented = augmentor(image=img, bboxes=[coords], class_labels=['
                cv2.imwrite(os.path.join('aug_data', partition, 'images', f'{imag

                annotation = {}
                annotation['image'] = image

                if os.path.exists(label_path):
                    if len(augmented['bboxes']) == 0:
                        annotation['bbox'] = [0,0,0,0]
                        annotation['class'] = 0
                    else:
                        annotation['bbox'] = augmented['bboxes'][0]
                        annotation['class'] = 1
                else:
                    annotation['bbox'] = [0,0,0,0]
                    annotation['class'] = 0


                with open(os.path.join('aug_data', partition, 'labels', f'{image.
                    json.dump(annotation, f)

        except Exception as e:
            print(e)
```

```
train_images = tf.data.Dataset.list_files('aug_data\\train\\images\\*.jpg', shuff
train_images = train_images.map(load_image)
train_images = train_images.map(lambda x: tf.image.resize(x, (120,120)))
train_images = train_images.map(lambda x: x/255)
```

```
test_images = tf.data.Dataset.list_files('aug_data\\test\\images\\*.jpg', shuffle
test_images = test_images.map(load_image)
test_images = test_images.map(lambda x: tf.image.resize(x, (120,120)))
test_images = test_images.map(lambda x: x/255)
```

```
val_images = tf.data.Dataset.list_files('aug_data\\val\\images\\*.jpg', shuffle=F
val_images = val_images.map(load_image)
val_images = val_images.map(lambda x: tf.image.resize(x, (120,120)))
val_images = val_images.map(lambda x: x/255)
```

```
train_images.as_numpy_iterator().next()
```

```
def load_labels(label_path):
    with open(label_path.numpy(), 'r', encoding = "utf-8") as f:
        label = json.load(f)

    return [label['class']], label['bbox']
```

6.2 Load Labels to Tensorflow Dataset

```
train_labels = tf.data.Dataset.list_files('aug_data\\train\\labels\\*.json', shuf
train_labels = train_labels.map(lambda x: tf.py_function(load_labels, [x], [tf.ui
```

```
test_labels = tf.data.Dataset.list_files('aug_data\\test\\labels\\*.json', shuffl
test_labels = test_labels.map(lambda x: tf.py_function(load_labels, [x], [tf.uint
```

```
val_labels = tf.data.Dataset.list_files('aug_data\\val\\labels\\*.json', shuffle=
val_labels = val_labels.map(lambda x: tf.py_function(load_labels, [x], [tf.uint8,
```

```
train_labels.as_numpy_iterator().next()
```

```
data_samples = train.as_numpy_iterator()
```

```
res = data_samples.next()
```

```
fig, ax = plt.subplots(ncols=4, figsize=(20,20))
for idx in range(4):
    sample_image = res[0][idx]
    sample_coords = res[1][1][idx]

    cv2.rectangle(sample_image,
                  tuple(np.multiply(sample_coords[:2], [120,120]).astype(int)),
                  tuple(np.multiply(sample_coords[2:], [120,120]).astype(int)),
                        (255,0,0), 2)

    ax[idx].imshow(sample_image)
```

```
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Input, Conv2D, Dense, GlobalMaxPooling2D
from tensorflow.keras.applications import VGG16
```

8.2 Download VGG16

```
vgg = VGG16(include_top=False)
```

```
vgg.summary()
```

```
train = tf.data.Dataset.zip((train_images, train_labels))
train = train.shuffle(5000)
train = train.batch(8)
train = train.prefetch(4)
```

```
test = tf.data.Dataset.zip((test_images, test_labels))
test = test.shuffle(1300)
test = test.batch(8)
test = test.prefetch(4)
```

```
val = tf.data.Dataset.zip((val_images, val_labels))
val = val.shuffle(1000)
val = val.batch(8)
val = val.prefetch(4)
```

```
train.as_numpy_iterator().next()[1]
```

```
def build_model():
    input_layer = Input(shape=(120,120,3))

    vgg = VGG16(include_top=False)(input_layer)

    # Classification Model
    f1 = GlobalMaxPooling2D()(vgg)
    class1 = Dense(2048, activation='relu')(f1)
    class2 = Dense(1, activation='sigmoid')(class1)

    # Bounding box model
    f2 = GlobalMaxPooling2D()(vgg)
    regress1 = Dense(2048, activation='relu')(f2)
    regress2 = Dense(4, activation='sigmoid')(regress1)

    facetracker = Model(inputs=input_layer, outputs=[class2, regress2])
    return facetracker
```

```
facetracker = build_model()
```

```
facetracker.summary()
```

```
X, y = train.as_numpy_iterator().next()
```

```
X.shape
```

```
classes, coords = facetracker.predict(X)
```

```
classes, coords
```

```
def build_model():
    input_layer = Input(shape=(120,120,3))

    vgg = VGG16(include_top=False)(input_layer)

    # Classification Model
    f1 = GlobalMaxPooling2D()(vgg)
    class1 = Dense(2048, activation='relu')(f1)
    class2 = Dense(1, activation='sigmoid')(class1)

    # Bounding box model
    f2 = GlobalMaxPooling2D()(vgg)
    regress1 = Dense(2048, activation='relu')(f2)
    regress2 = Dense(4, activation='sigmoid')(regress1)

    facetracker = Model(inputs=input_layer, outputs=[class2, regress2])
    return facetracker
```

```
data_samples = train.as_numpy_iterator()
```

```
res = data_samples.next()
```

```
fig, ax = plt.subplots(ncols=4, figsize=(20,20))
for idx in range(4):
    sample_image = res[0][idx]
    sample_coords = res[1][1][idx]

    cv2.rectangle(sample_image,
                  tuple(np.multiply(sample_coords[:2], [120,120]).astype(int)),
                  tuple(np.multiply(sample_coords[2:], [120,120]).astype(int)),
                        (255,0,0), 2)

    ax[idx].imshow(sample_image)
```

```
cap = cv2.VideoCapture(1)
while cap.isOpened():
    _ , frame = cap.read()
    frame = frame[50:500, 50:500,:]

    rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    resized = tf.image.resize(rgb, (120,120))

    yhat = facetracker.predict(np.expand_dims(resized/255,0))
    sample_coords = yhat[1][0]

    if yhat[0] > 0.5:
        # Controls the main rectangle
        cv2.rectangle(frame,
                      tuple(np.multiply(sample_coords[:2], [450,450]).astype(int
                      tuple(np.multiply(sample_coords[2:], [450,450]).astype(int
                            (255,0,0), 2)
        # Controls the label rectangle
        cv2.rectangle(frame,
                      tuple(np.add(np.multiply(sample_coords[:2], [450,450]).ast
                                   [0,-30])),
                      tuple(np.add(np.multiply(sample_coords[:2], [450,450]).ast
                                   [80,0])),
                            (255,0,0), -1)

        # Controls the text rendered
        cv2.putText(frame, 'face', tuple(np.add(np.multiply(sample_coords[:2], [
                                                [0,-5])),
                    cv2.FONT_HERSHEY_SIMPLEX, 1, (255,255,255), 2, cv2.LINE_AA)

    cv2.imshow('EyeTrack', frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

PAPER PUBLICATION

Siddharth Misra, Priyank Gupta, Amol Vasudeva, Shruti Jain, Facial Gesture Interfaces: Breaking Barriers for Individuals with Disabilities, The Fourth International Conference on **Emerging Techniques in Computational Intelligence, ICETCI 2024**, Mahindra University, Hyderabad, India, from August 22 to 24, 2024 (Submitted).

PLAGIARISM CERTIFICATE

plag reort

ORIGINALITY REPORT

5%	3%	2%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Jaypee University of Information Technology Student Paper	1%
2	Submitted to University of Hertfordshire Student Paper	<1%
3	ebin.pub Internet Source	<1%
4	link.springer.com Internet Source	<1%
5	repository.its.ac.id Internet Source	<1%
6	www.researchgate.net Internet Source	<1%
7	Siraj Khan, Muhammad Sajjad, Naveed Abbas, José Escorcia-Gutierrez, Margarita Gamarra, Khan Muhammad. "Efficient leukocytes detection and classification in microscopic blood images using convolutional neural network coupled with a dual attention	<1%

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: ……

Type of Document (Tick): **PhD Thesis** | **M.Tech/M.Sc. Dissertation** | **B.Tech./B.Sc./BBA/Other**

Name:_____ **Department:**_____ **Enrolment No** _____

Contact No._____ **E-mail.**_____

Name of the Supervisor: _____

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): _____

_____

_____

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

- Total No. of Pages =
- Total No. of Preliminary pages =
- Total No. of pages accommodate bibliography/references =

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at.....(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor) **Signature of HOD**

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

<table>
<tr><th>Copy Received on</th><th>Excluded</th><th>Similarity Index (%)</th><th colspan="2">Abstract & Chapters Details</th></tr>
<tr><td rowspan="2"></td><td rowspan="4">• All Preliminary Pages
• Bibliography/Images/Quotes
• 14 Words String</td><td rowspan="2"></td><td>Word Counts</td><td></td></tr>
<tr><td rowspan="2">Character Counts</td><td rowspan="2"></td></tr>
<tr><td>Report Generated on</td><td>Submission ID</td></tr>
<tr><td></td><td></td><td>Page counts</td><td></td></tr>
<tr><td></td><td></td><td></td><td>File Size</td><td></td></tr>
</table>

Checked by
Name & Signature **Librarian**

……

Please send your complete Thesis/Report in (PDF) & DOC (Word File) through your Supervisor/Guide at plagcheck.juit@gmail.com