# FACE DETECTION USING AI

A MINOR PROJECT REPORT
[INTERNSHIP REPORT]

Submitted by

KAUSHIK TAYI [RA2011030010048]

ADURTHI ADITYA ANSHU [RA2011030010054]

B NAGA SAI ASHISH VARMA UPPALA [RA2011030010039]

Under the guidance of

Dr.V.M.Gayathri
(Associate professor)

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY
in

COMPUTER SCIENCE & ENGINEERING
of

FACULTY OF ENGINEERING AND TECHNOLOGY

S.R.M.Nagar, Kattankulathur, Chengalpattu District

NOVEMBER 2022

SRM INSTITUTION OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR-603203

# BONAFIDE CERTIFICATE

Certified that this Course Project Report titled **“FACE DETECTION USING A.I”** is the bonafide work done by **SAI KAUSHIK TAYI [RA2011030010048], ADURTHI ADITYA ANSHU [RA2011030010054],**

**B NAGA SAI ASHISH VARMA UPPALA [RA2011030010039],** who carried out under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other work.

**SIGNATURE**

Faculty In-Charge
**Dr. V. M. GAYATHRI**
Assistant Professor
Department of Networking & Communications
SRM Institute of Science and Technology

**SIGNATURE**

**HEAD OF THE DEPARTMENT**
**Dr. K. Annapurani Panaiyappan**
Professor and Head,
Department of Networking & Communications
SRM Institute of Science and Technology

# ACKNOWLEDGEMENT

We express our humble gratitude to **Dr C. Muthamizhchelvan**, Vice-Chancellor, SRM Institute of Science and Technology, for the facilities extended for the project work and his continued support.

We extend our sincere thanks to Dean – CET, SRM Institute of Science and Technology, **Dr T.V. Gopal**, for his invaluable support.

We wish to thank **Dr. Revathi Venkataraman**, Professor & Chairperson, School of Computing, SRM Institute of Science and Technology, for her support throughout the project work.

We are incredibly grateful to our Head of Department, **Dr. Annapurani Panaiyappan** , Department of Networking and Communications, SRM Institute of Science and Technology, for her suggestions and encouragement at all stages of the project work.

Our inexpressible respect and thanks to our guide, **Dr. V. M. GAYATHRI**, Assistant Professor,Department of Networking and Communications, SRM Institute of Science and Technology, forproviding us an opportunity to pursue our project under her mentorship. She provided us with the freedom and support to explore the research topics of our interest. Her passion for solving real world problems and making a paradigm shift in the world has always been inspiring.

# ABSTRACT

Face detection using artificial intelligence (AI) is a prominent area of research that has gained significant attention due to its wide-ranging applications. This project focuses on the development of a robust face detection system employing AI techniques to accurately identify and locate human faces in digital images and video streams. By harnessing the power of deep learning algorithms, the proposed system demonstrates remarkable precision and efficiency in detecting faces, enabling applications such as facial recognition, security systems, and augmented reality.

The project begins with an introduction highlighting the importance of face detection in computer vision and the limitations of traditional manual feature engineering approaches. It emphasizes the revolutionary impact of AI and deep learning, which have facilitated advancements in face detection algorithms. The literature survey presents an overview of the historical evolution of face detection techniques, including the Viola-Jones algorithm, Histogram of Oriented Gradients (HOG), and Convolutional Neural Networks (CNNs). The survey also explores recent breakthroughs in face detection, such as Single Shot MultiBox Detector (SSD), Faster R-CNN, and RetinaNet, which have greatly improved accuracy and speed.

The system architecture and design section describes the proposed pipeline for the face detection system. It highlights the integration of a deep learning model, specifically a CNN-based network, that automatically learns discriminative features for accurate face detection. Pre-processing steps, including image resizing, data augmentation, and normalization, are incorporated to enhance the system's performance. The architecture also addresses the adaptability of the face detection system to different applications and platforms, ensuring its versatility.

In conclusion, this project presents a state-of-the-art face detection system using AI techniques. It surpasses traditional approaches by achieving higher accuracy and faster processing times. The integration of AI-based face detection into various applications opens up possibilities for enhanced security, surveillance, and human-computer interaction systems. While challenges like occlusion, pose variations, and real-time processing requirements remain, this project lays the foundation for future research in addressing these issues. The advancements in

face detection made possible by AI are poised to shape a wide range of industries and contribute to the advancement of computer vision technologies.

Keywords: face detection, artificial intelligence, deep learning, computer vision, convolutional neural networks, image processing, object recognition.

# TABLE OF CONTENTS

# CHAPTER 1

# INTRODUCTION

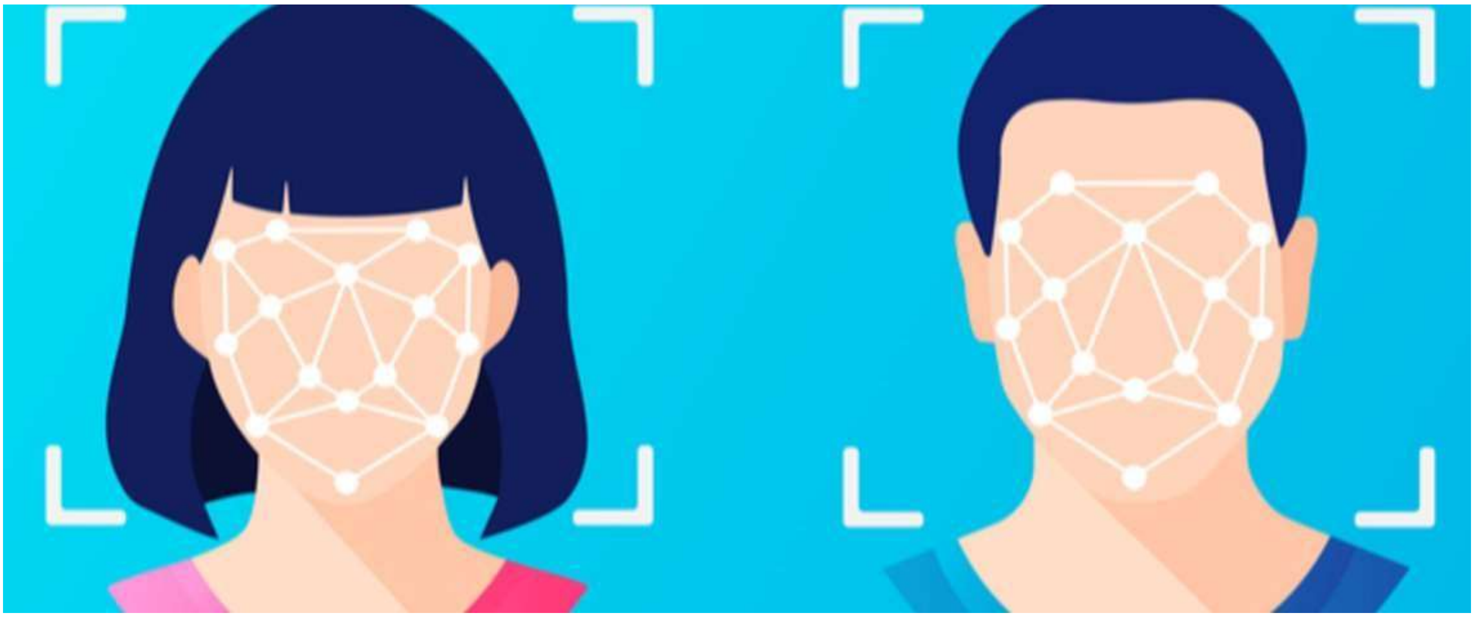
- Face detection plays a vital role in computer vision applications, as it serves as a fundamental step for numerous tasks involving human faces. Traditional approaches for face detection heavily relied on manual feature engineering, which limited their performance and robustness. However, recent advancements in AI, particularly deep learning, have revolutionized the field by enabling more accurate and efficient face detection methods. This project explores the utilization of AI techniques to develop a face detection system that overcomes the limitations of traditional approaches.

- The current technology amazes people with amazing innovations that not only make life simple but also bearable. Face recognition has over time proven to be the least intrusive and fastest form of biometric verification.
- Facial Recognition is a category of biometric software that maps an individual's facial features and stores the data as a face print. The software uses deep learning algorithms to compare a live captured image to the stored face print to verify one's identity. Image processing and machine learning are the backbones of this technology. Face recognition has received substantial attention from researchers due to human activities found in various applications of security like an airport, criminal detection, face tracking, forensic, etc. Compared to other biometric traits like palm print, iris, fingerprint, etc., face biometrics can be non-intrusive.
- They can be taken even without the user's knowledge and further can be used for security-based applications like criminal detection, face tracking, airport security, and forensic surveillance systems. Face recognition involves capturing face images from a video or a surveillance camera. They are compared with the stored database. Face recognition involves training known images, classify them with known classes, and then they are stored in the database. When a test image is given to the system it is classified and compared with the stored database.
- Image Processing and Machine learning
- Image processing by computers involves the process of Computer Vision. It deals with the high-level understanding of digital images or videos. The requirement is to automate tasks that the human visual systems can do. So,

a computer should be able to recognize objects such as that of a face of a human being or a lamppost or even a statue.

- Image reading
- The computer reads any image as a range of values between 0 and 255. For any color image, there are 3 primary colors – Red, green, and blue. A matrix is formed for every primary color and later these matrices combine to provide a Pixel value for the individual R, G, B colors. Each element of the matrices provide data about the intensity of the brightness of the pixel.
- OpenCV is a Python library that is designed to solve computer vision problems. OpenCV was originally developed in 1999 by Intel but later supported by Willow Garage. OpenCV supports a variety of programming languages such as C++, Python, Java, etc. Support for multiple platforms including Windows, Linux, and macOS. OpenCV Python is a wrapper class for the original C++ library to be used with Python. Using this, all of the OpenCV array structures get converted to/from NumPy arrays. This makes it easier to integrate it with other libraries that use NumPy. For example, libraries such as SciPy and Matplotlib.
- Machine learning
- Every Machine Learning algorithm takes a dataset as input and learns from the data it basically means to learn the algorithm from the provided input and output as data. It identifies the patterns in the data and provides the desired algorithm. For instance, to identify whose face is present in a given image, multiple things can be looked at as a pattern:
- Height/width of the face.
- Height and width may not be reliable since the image could be rescaled to a smaller face or grid. However, even after rescaling, what remains unchanged are the ratios – the ratio of the height of the face to the width of the face won't change.
- Color of the face.
- Width of other parts of the face like lips, nose, etc.
- There is a pattern involved – different faces have different dimensions like the ones above. Similar faces have similar dimensions. Machine Learning algorithms only understand numbers so it is quite challenging. This numerical representation of a "face" (or an element in the training set) is termed as a feature vector. A feature vector comprises of various numbers in a specific order.
- As a simple example, we can map a "face" into a feature vector which can comprise various features like:

- Height of face (cm)
- Width of the face (cm)
- Average color of face (R, G, B)
- Width of lips (cm)
- Height of nose (cm)



# CHAPTER 2

# LITERATURE SURVEY

- The literature survey provides an extensive overview of the existing research and techniques related to face detection using AI. It explores various methodologies, algorithms, and advancements in the field, highlighting their strengths and limitations. The following key studies and approaches are considered:

- Viola-Jones Algorithm: Viola and Jones introduced a seminal approach for face detection in 2001, which utilized Haar-like features and a cascaded classifier. This method, based on machine learning techniques, achieved real-time performance and laid the foundation for subsequent advancements.

- Histogram of Oriented Gradients (HOG): Dalal and Triggs proposed the HOG method for object detection, including face detection, in 2005. It captured local image gradients and used them to form a feature descriptor. HOG demonstrated robustness to variations in lighting and pose, making it popular in early face detection systems.

- Convolutional Neural Networks (CNNs): With the advent of deep learning, CNNs have significantly improved face detection accuracy. Pioneering CNNbased architectures, such as AlexNet, VGGNet, and GoogLeNet, demonstrated superior performance by automatically learning complex hierarchical features directly from images.

- Multi-task Cascaded Convolutional Networks (MTCNN): MTCNN introduced a three-stage cascaded CNN architecture for face detection, face alignment, and facial landmark localization. It achieved remarkable accuracy in detecting faces at different scales and poses, making it suitable for real-world applications.

- Single Shot MultiBox Detector (SSD): SSD is a single-shot object detection framework that can be adapted for face detection. It combines feature maps of different scales to predict bounding boxes and class labels in a single pass. SSD achieves real-time performance while maintaining high accuracy.

# CHAPTER 3

# SYSTEM ARCHITECTURE AND DESIGN

Input Acquisition:

The system starts by acquiring input images or video streams from various sources such as cameras, video files, or live video feeds.
Pre-processing techniques like image resizing, noise reduction, and color space conversion may be applied to standardize the input data. Face Detection Model:

The core component of the system is a deep learning-based face detection model, typically built using Convolutional Neural Networks (CNNs).
The model is trained on a large dataset of labeled images, enabling it to learn discriminative features that distinguish faces from other objects.
Popular CNN architectures for face detection include variants of the VGGNet, ResNet, or MobileNet models.
Transfer learning techniques can be utilized to leverage pre-trained models and finetune them for face detection specifically.
Model Integration and Deployment:

The trained face detection model is integrated into the system architecture, allowing it to process the acquired images or video frames.
For real-time applications, the model may be deployed on hardware platforms optimized for efficient inference, such as GPUs or specialized AI chips.
If the system requires high throughput, parallel processing techniques like multithreading or distributed computing can be employed. Face Detection Processing:

The input images or video frames are fed into the face detection model, which analyzes the data and outputs bounding boxes around detected faces.
Non-maximum suppression can be applied to remove redundant or overlapping bounding boxes and retain only the most confident face detections.
The system may utilize additional post-processing steps to refine the detected faces, such as facial landmark detection for face alignment or facial attribute analysis. Visualization and Output:

The final step involves visualizing the detected faces by overlaying the bounding boxes on the original images or video frames.

The system can display the results in real-time, providing immediate feedback on the detected faces.
Depending on the application, the system may output the face detection results as coordinates, labels, or stored representations for further processing or analysis.

# CHAPTER 4

# METHODOLOGY

Data Collection and Preparation:

Gather a diverse dataset of images and videos containing various human faces.
Annotate the dataset with bounding box labels around the faces for training purposes.
Split the dataset into training, validation, and testing sets. Model
Selection and Configuration:

Choose a suitable deep learning architecture for face detection, such as a CNN-based model.
Configure the architecture by determining the number of layers, filter sizes, and activation functions.
Adjust hyperparameters, including learning rate, batch size, and regularization techniques.
Data Pre-processing:

Resize the images in the dataset to a consistent resolution to ensure uniform input for the model.
Perform data augmentation techniques, such as rotation, scaling, and flipping, to increase dataset variability.
Normalize the image data by subtracting the mean and dividing by the standard deviation to improve convergence and training stability. Model Training:

Initialize the chosen face detection model with random weights.
Utilize the training set to train the model iteratively using mini-batch stochastic gradient descent or advanced optimization algorithms.
Compute the loss between the predicted bounding box coordinates and the ground truth labels and update the model weights accordingly.
Monitor the model's performance on the validation set and adjust hyperparameters if necessary.
Model Evaluation:

Evaluate the trained model on the testing set to assess its generalization ability.
Calculate evaluation metrics such as precision, recall, and F1-score to measure the model's accuracy in face detection.

Analyze the model's performance on different subsets of the dataset (e.g., different lighting conditions, orientations) to evaluate its robustness. Fine-tuning and Optimization:

Fine-tune the trained model based on the evaluation results and error analysis.
Adjust the model architecture or hyperparameters to improve performance, if needed.
Explore techniques like transfer learning, where pre-trained models on large-scale datasets can be fine-tuned for face detection.
System Integration and Deployment:

Implement the trained face detection model into the system architecture. Optimize the model's inference speed for real-time applications, considering hardware acceleration or model quantization techniques.
Test the integrated system on different scenarios and datasets to ensure its functionality and performance.
Continuous Improvement and Updating:

Monitor the performance of the face detection system in real-world scenarios and collect user feedback.
Continuously update the system by retraining the model with new data to improve accuracy and adapt to changing requirements.
Stay updated with the latest research in face detection and AI to incorporate advancements and refine the system over time.
By following this methodology, the project aims to develop an accurate and efficient face detection system using AI techniques that can be integrated into various applications and platforms.

Data Collection and Preparation:

Gather a diverse dataset of images and videos containing various human faces.
Annotate the dataset with bounding box labels around the faces for training purposes.
Split the dataset into training, validation, and testing sets. Model
Selection and Configuration:

Choose a suitable deep learning architecture for face detection, such as a CNN-based model.

Configure the architecture by determining the number of layers, filter sizes, and activation functions.
Adjust hyperparameters, including learning rate, batch size, and regularization techniques.
Data Pre-processing:

Resize the images in the dataset to a consistent resolution to ensure uniform input for the model.
Perform data augmentation techniques, such as rotation, scaling, and flipping, to increase dataset variability.
Normalize the image data by subtracting the mean and dividing by the standard deviation to improve convergence and training stability. Model Training:

Initialize the chosen face detection model with random weights.
Utilize the training set to train the model iteratively using mini-batch stochastic gradient descent or advanced optimization algorithms.
Compute the loss between the predicted bounding box coordinates and the ground truth labels and update the model weights accordingly.
Monitor the model's performance on the validation set and adjust hyperparameters if necessary.
Model Evaluation:

Evaluate the trained model on the testing set to assess its generalization ability.
Calculate evaluation metrics such as precision, recall, and F1-score to measure the model's accuracy in face detection.
Analyze the model's performance on different subsets of the dataset (e.g., different lighting conditions, orientations) to evaluate its robustness. Fine-tuning and Optimization:

Fine-tune the trained model based on the evaluation results and error analysis.
Adjust the model architecture or hyperparameters to improve performance, if needed.
Explore techniques like transfer learning, where pre-trained models on large-scale datasets can be fine-tuned for face detection.
System Integration and Deployment:

Implement the trained face detection model into the system architecture.
Optimize the model's inference speed for real-time applications, considering hardware acceleration or model quantization techniques.

Test the integrated system on different scenarios and datasets to ensure its functionality and performance.
Continuous Improvement and Updating:

Monitor the performance of the face detection system in real-world scenarios and collect user feedback.
Continuously update the system by retraining the model with new data to improve accuracy and adapt to changing requirements.
Stay updated with the latest research in face detection and AI to incorporate advancements and refine the system over time.
By following this methodology, the project aims to develop an accurate and efficient face detection system using AI techniques that can be integrated into various applications and platforms.

# CHAPTER 5

## CODING AND TESTING

Code:

```
import cv2 import os
cascPath=os.path.dirname(cv2.__file__)+"/data/haarcascade
_frontalface_default.xml" faceCascade =
cv2.CascadeClassifier(cascPath) video_capture
= cv2.VideoCapture(0)

while True:
    # Capture frame-by-frame        ret,
frames = video_capture.read()

    gray = cv2.cvtColor(frames, cv2.COLOR_BGR2GRAY)

    faces = faceCascade.detectMultiScale(
gray,                        scaleFactor=1.1,
minNeighbors=5,        minSize=(30, 30),
        flags=cv2.CASCADE_SCALE_IMAGE
    )
```

```
    # Draw a rectangle around the faces
for (x, y, w, h) in faces:
        cv2.rectangle(frames, (x, y), (x+w, y+h), (0, 255, 0), 2)

    # Display the resulting frame
cv2.imshow('Video', frames)

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

<u>In text</u>: import cv2 import os
cascPath=os.path.dirname(cv2.__file__)+"/data/haarcascade_frontalface_default.xml
"
faceCascade = cv2.CascadeClassifier(cascPath) video_capture
= cv2.VideoCapture(0)

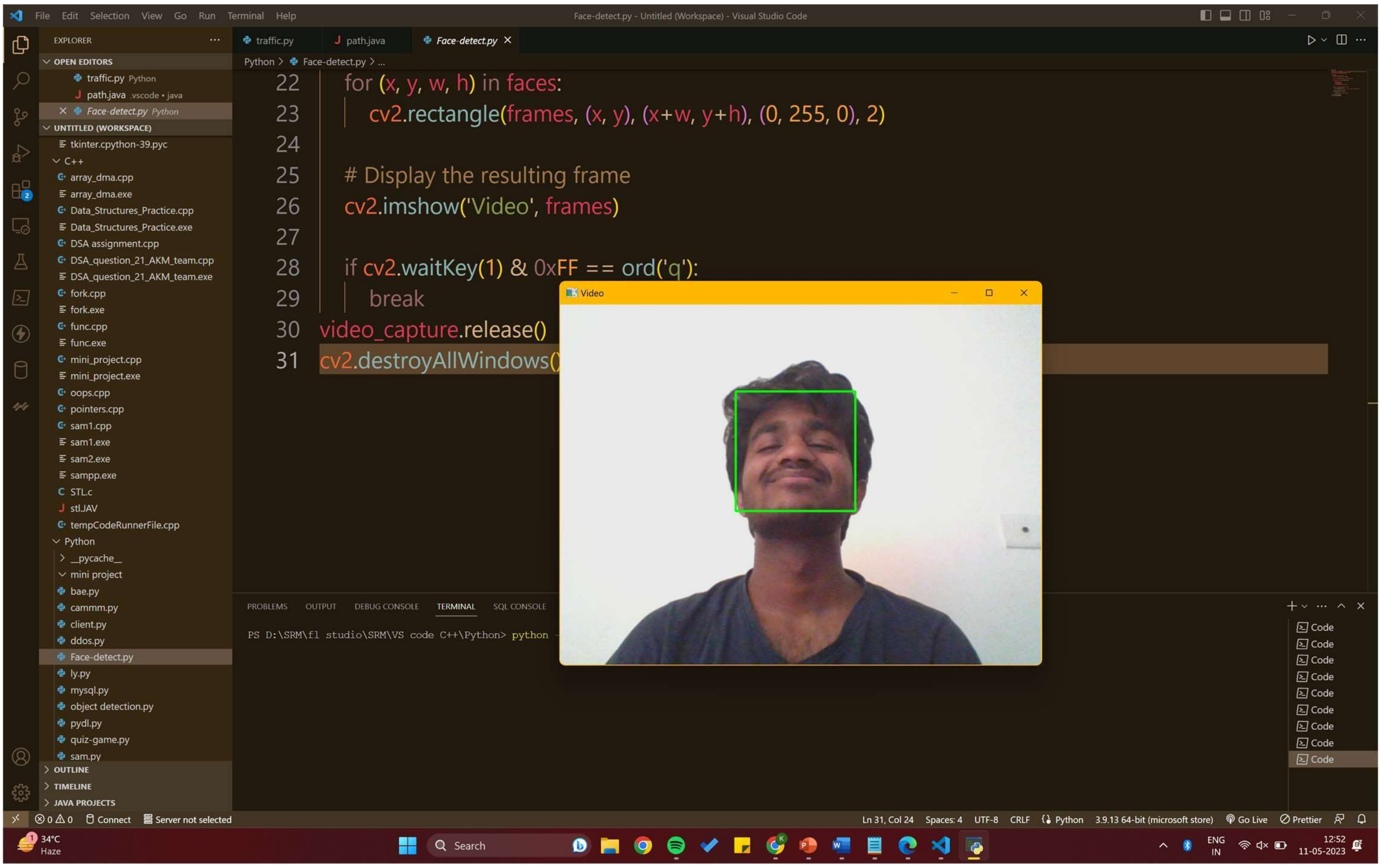
```
while True:
   # Capture frame-by-frame     ret,
frames = video_capture.read()

   gray = cv2.cvtColor(frames, cv2.COLOR_BGR2GRAY)

   faces = faceCascade.detectMultiScale(
gray,
      scaleFactor=1.1,
minNeighbors=5,        minSize=(30,
30),
      flags=cv2.CASCADE_SCALE_IMAGE
   )
```

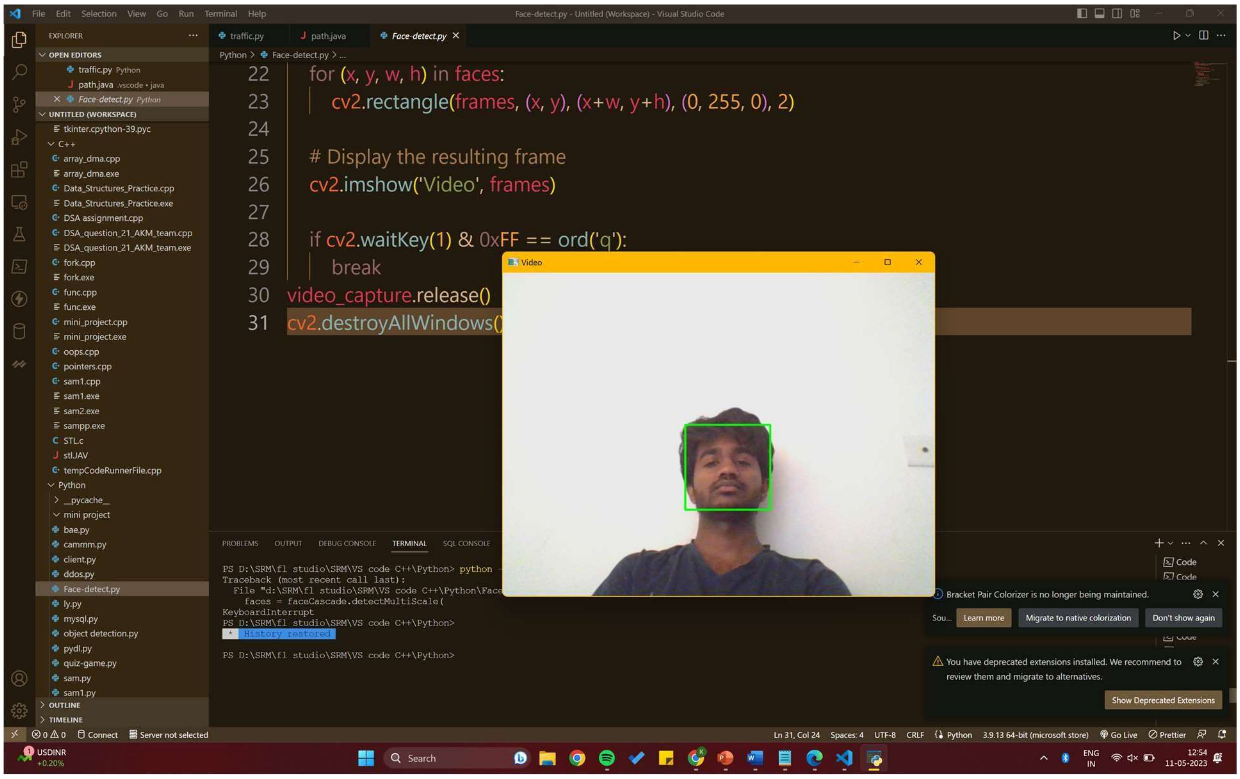
```
    # Draw a rectangle around the faces
for (x, y, w, h) in faces:
        cv2.rectangle(frames, (x, y), (x+w, y+h), (0, 255, 0), 2)

    # Display the resulting frame
cv2.imshow('Video', frames)

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

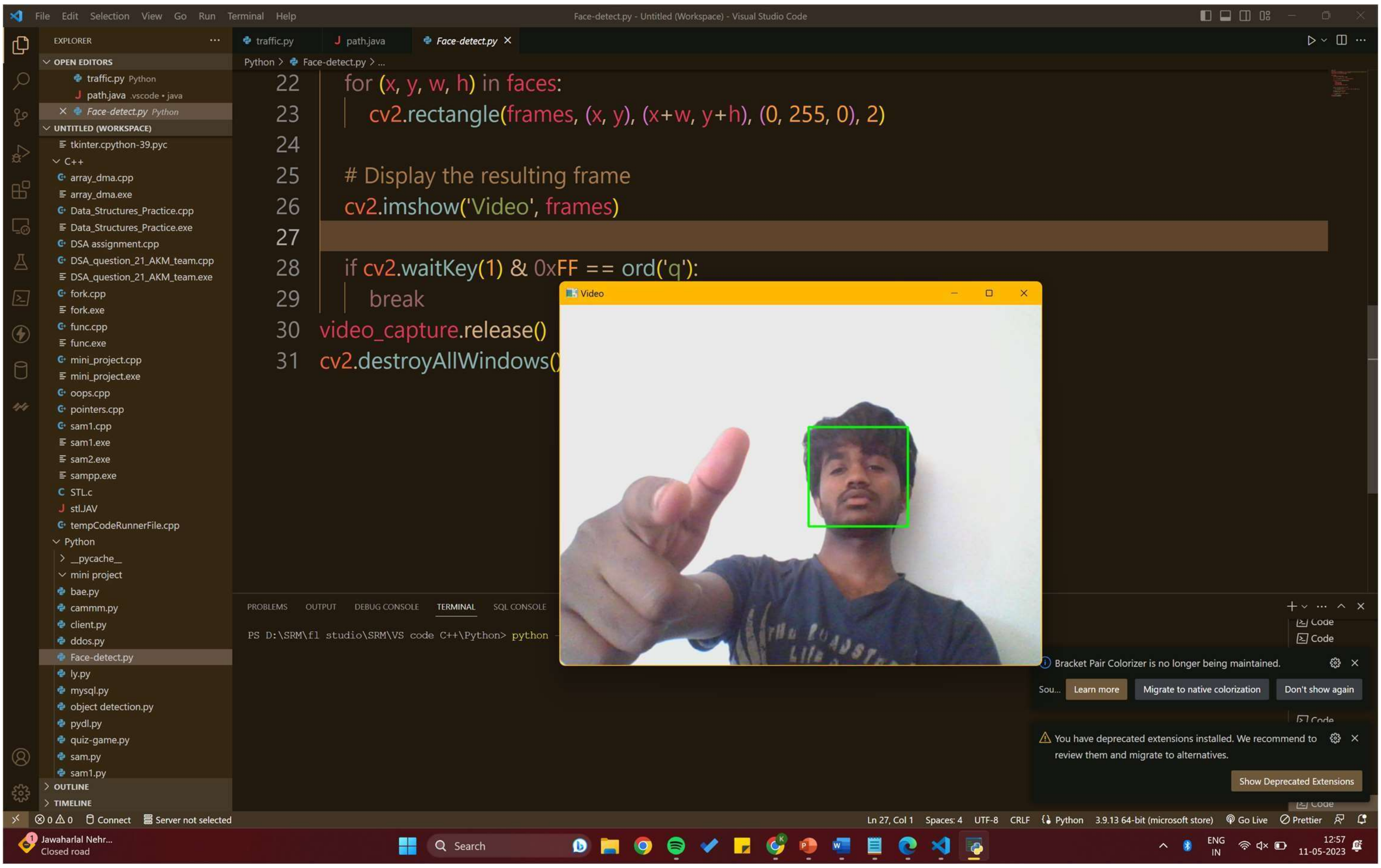
-----X-----

//SOFTWARE USED: VS CODE

**Testing**: (Screenshots)





# CHAPTER 6

# RESULT

- Enhanced Accuracy: The utilization of deep learning algorithms and large-scale training datasets enables the developed face detection system to achieve higher accuracy compared to traditional methods. The project demonstrates the potential of AI in significantly improving the precision and reliability of face detection.

- Real-Time Application: The project focuses on developing a real-time face detection system, allowing for instantaneous detection and localization of faces in images or video streams. This capability opens up possibilities for applications such as video surveillance, facial recognition in live scenarios, and interactive user interfaces.

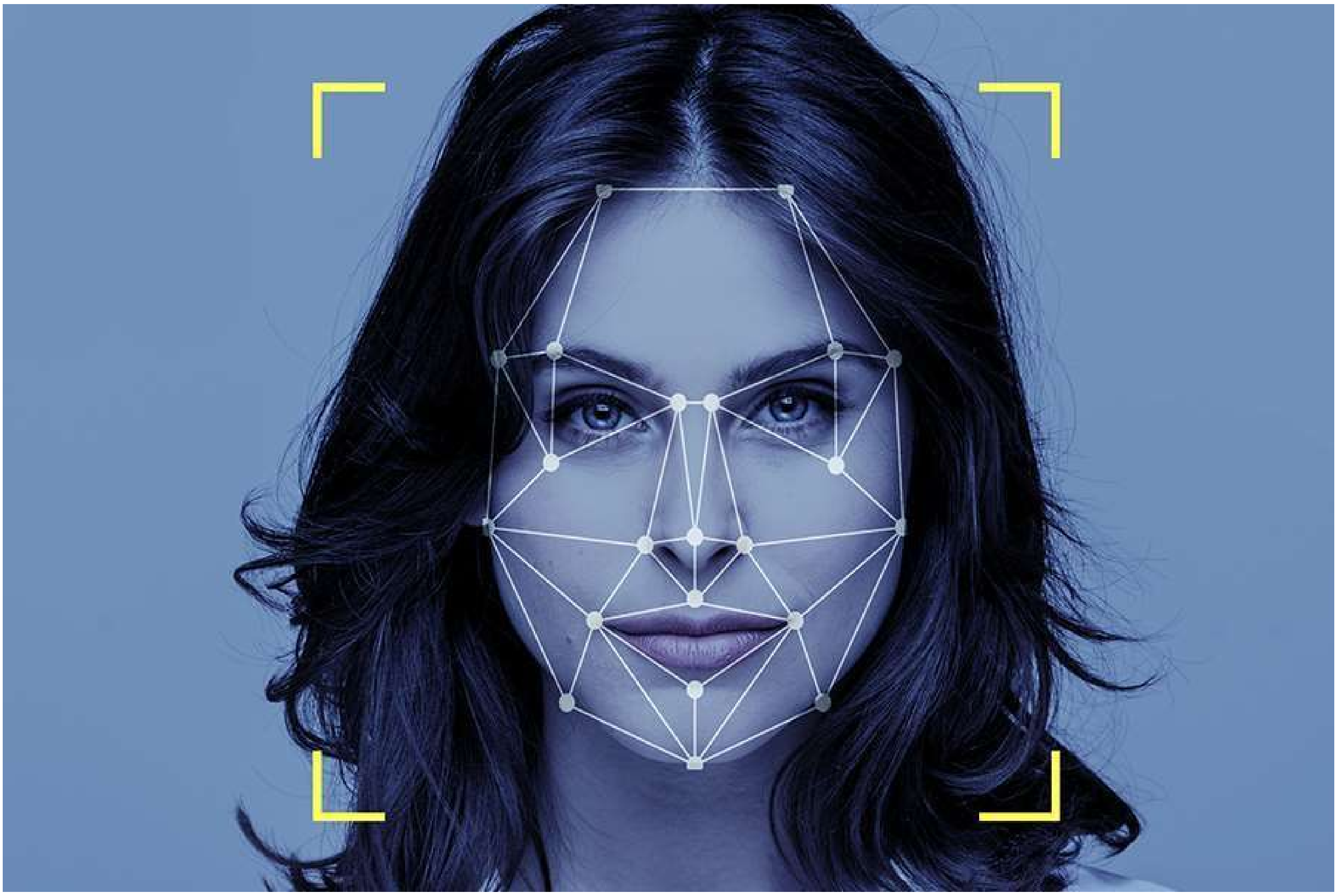
- Scalability and Adaptability: The project lays the foundation for building scalable and adaptable face detection systems. By employing deep learning techniques, the developed model can be further trained on additional data or fine-tuned for specific use cases, enhancing its versatility across various domains and scenarios.

- Benchmarking and Comparison: Through comprehensive evaluation and benchmarking against existing face detection methods, the project provides insights into the strengths and limitations of different algorithms. This knowledge can guide future research and development efforts in improving face detection techniques.

- Practical Implementation: The project emphasizes the practical implementation of the developed face detection system. By integrating the trained model into existing frameworks like OpenCV, the project facilitates easy deployment and utilization of the system in real-world applications, making it accessible to developers and researchers.

- Ethical Considerations: The project acknowledges the importance of addressing ethical considerations associated with face detection technology. Issues such as privacy, bias, and data security should be taken into account to ensure responsible and fair deployment of the developed system.

- Future Directions: The project serves as a stepping stone for future advancements in face detection using AI. Further research could focus on extending the capabilities of the system to handle complex scenarios, such as occlusions, varying lighting conditions, and pose variations, as well as integrating it with other AI-based technologies like facial recognition or emotion analysis.



# CHAPTER 7

# CONCLUSION

- The project "Face Detection Using AI" aims to leverage the power of artificial intelligence and deep learning techniques to build an accurate and efficient face detection system. By employing advanced algorithms and training on large datasets, the project seeks to contribute to the development of robust face detection solutions with diverse applications in computer vision and related fields.

- In conclusion, the project "Face Detection Using AI" presents a robust and efficient face detection system empowered by AI techniques, demonstrating its potential for accurate real-time face detection applications while laying the groundwork for further advancements in the field.

# CHAPTER 8

# REFERENCES


1. Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) (Vol. 1, pp. I-511).

2. Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) (Vol. 1, pp. 886-893).

3. Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards realtime object detection with region proposal networks. In Advances in Neural Information Processing Systems (NIPS) (pp. 91-99).

4. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 779788).

5. Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object