VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELAGAVI – 590014

A 6th Semester Computer Graphics and Image Processing

Mini-Project Report on

"Real-Time Head Pose Estimation Using OpenCV"

Submitted in partial fulfillment of the requirements for the award of degree of

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND ENGINEERING

Submitted by:

PRAMOD J TOPANNAVAR
(2AG21CS069)

SIDDAPPA TOTAGER
(2AG22CS413)

Under the Guidance of

Prof. Priyanka Pujari

Assistant Professor, Dept. of CSE,

AITM, Belagavi.

ANGADI INSTITUTE OF TECHNOLOGY & MANAGEMENT

BELAGAVI-590009

2023-2024

ANGADI INSTITUTE OF TECHNOLOGY & MANAGEMENT

BELAGAVI -590009

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Certificate

This is to certify that the 6th Semester Mini-Project entitled **"Real-Time Head Pose Estimation Using OpenCV"** is work carried out by PRAMOD J TOPANNAVAR(2AG21CS069) and SIDDAPPA TOTAGER(2AG22CS413) in partial fulfillment of the requirements for the award of the degree of **Bachelor of Computer Science & Engineering under Visvesvaraya Technological University, Belagavi** during the year 2023-2024. It is certified that all the correction/suggestion indicated for internal assessment have been incorporated in the report. The Computer Graphics and Image Processing 6th Semester Mini-Project report has been approved as it satisfies the academic requirements in respect of the Mini-project work prescribed for the Bachelor of Engineering degree.

Signature of the Guide	**Signature of the HOD**	**Signature of the Principal**
Prof. Priyanka Pujari	**Dr. Dhanashree Kulkarni**	**Dr. Anand Deshpande**
Assistant Professor,	**Professor and Head,**	**Principal and Director,**
Dept. of CSE, AITM	**Dept. of CSE, AITM**	**AITM, Belagavi**

Name of the Examiner:	**Signature with date:**
1. ………………………	**……………………..**
2. ……………………..	**......…………………**

Vision of the Department
To impart quality education in computer science and engineering with emphasis on professional skills to meet the global challenges in IT paradigm.
Mission of the Department
M1: Impart knowledge in current and emerging computing technologies by adopting various pedagogical approaches. **M2:** Develop a conducive environment to inculcate and nurture analytical and communication skills along with societal and ethical responsibility in all professional endeavors. **M3:** Enable students to be successful globally by being effective problem solvers and lifelong learners.

Program Educational Objectives (PEOs)
Graduates will be able to: **PEO1:** Utilize the strong foundation in mathematics, programming, scientific and engineering fundamentals necessary to formulate, analyze and solve IT related engineering problems, and endeavor themselves for higher learning. **PEO2:** Demonstrate an ability to analyze the requirements and technical specifications of software to articulate novel engineering solutions for an efficient product design. **PEO3**: Adapt to emerging technologies, and work as teams on multidisciplinary projects meeting the requirements of Indian and multinational companies. **PEO4:** Pursue professional career adopting work values with a social concern to bridge the digital divide and develop effective communication skills and leadership qualities, and **PEO5:** Understand the efficacy of life-long learning, professional ethics and practices, so that they may emerge as global leaders.

DECLARATION

We **PRAMOD J TOPANNAVAR(2AG21CS069) and SIDDAPPA TOTAGER(2AG22CS413)** studying in the Sixth semester of Bachelor of Engineering in Computer Science and Engineering at Angadi Institute of Technology and Management, Belagavi, hereby declare that this Mini project work entitled **“Real-Time Head Pose Estimation Using OpenCV”** which is being submitted for the partial fulfillment for the award of the degree of Bachelor of Engineering in Computer Science and Engineering from Visvesvaraya Technological University, Belagavi is an authentic record of us carried out during the academic year 2023-2024 under the guidance of **Prof. Priyanka Pujari, Assistant Professor,** Department of Computer Science and Engineering, Angadi Institute of Technology and Management, Belagavi.

We further undertake that the matter embodied in the dissertation has not been submitted previously for the award of any degree or diploma by us to any other university or institution.

Place: Belagavi
Date:

PRAMOD J TOPANNAVAR
SIDDAPPA TOTAGER

ACKNOWLEDGEMENT

It is our proud privilege and duty to acknowledge the kind of help and guidance received from several people in preparation of this Project. It would not have been possible to prepare the project report in this form without their valuable help, cooperation and guidance.

First and foremost, we wish to record our sincere gratitude to the Management of this college and to our beloved Principal and Director, Angadi Institute of Technology & Management, Belagavi **Dr. Anand Deshpande** for his constant support and encouragement and for making available library and laboratory facilities.

Our sincere thanks to **Dr. Dhanashree Kulkarni**, Professor and Head, Department of Computer Science and Engineering, AITM for her valuable suggestions and guidance throughout the period of this report.

We express our sincere gratitude to our guide, **Prof. Priyanka Pujari**, Assistant Professor, Department of Computer Science and Engineering, AITM, Belagavi for guiding us in investigations for this project and in carrying out experimental work. Our numerous discussions with her were extremely helpful.

The Project on " **Real-Time Head Pose Estimation Using OpenCV**" was very helpful to us in giving the necessary background information and inspiration in choosing this topic for the project.

Last but not the least, we wish to thank our **parents** for financing our studies in this college as well as for constantly encouraging us to learn engineering. Their personal sacrifice in providing this opportunity to learn engineering is gratefully acknowledged.

INDEX

Chapter	Page no
Chapter 1	
Introduction	**1-2**
1.1 Introduction to Computer Graphics	1
1.2 Introduction to OpenCV	1
1.2.1 OpenCV for a Developer	2
Chapter 2	
Description of Topic and Input Keys	**3**
2.1 Program Definition	3
2.2 Description	3
Chapter 3	
Requirement Specifications	**4**
3.1 Hardware Requirements	4
3.2 Software Requirements	4
Chapter 4	
Advantages and Disadvantages	**5**
4.1 Advantages of OpenGL	5
4.2 Disadvantages of OpenGL	5

Chapter 5

OpenGL functions **6**

5.1 OpenCV Functions 6

Chapter 6

Implementation **7-10**

Chapter 7

Snapshots **11-12**

Conclusion **13**

References **14**

ABSTRACT

The project titled "Real-Time Head Pose Estimation Using OpenCV" aims to develop a system capable of detecting and estimating the head pose of individuals in real time. Utilizing OpenCV for video processing and Mediapipe for facial landmark detection, the system captures live video feed and accurately identifies the position and orientation of the user's head. The primary objective is to monitor the head's orientation along both horizontal and vertical axes, providing real-time feedback on whether the user is looking left, right, up, down, or straight ahead. This system can be potentially applied in various fields, including virtual reality, human-computer interaction, and proctoring systems, where understanding the user's head orientation is crucial. The project leverages advanced computer vision techniques and threading for efficient real-time performance, ensuring minimal latency and high accuracy in head pose estimation.

CHAPTER 1

INTRODUCTION

1.1 Introduction to Computer Graphics

Computer Graphics is concerned with all the aspects of producing pictures or images using a computer. The field began humbly almost fifty years ago with the display of few lines on a cathode ray tube (CRT), Now we can create images with computer that are indistinguishable from photographs of real objects.

The development of Computer Graphics has been driven by both the needs of the user community and by the advances in hardware and software. The applications of Computer Graphics are many and varied. However, we can divide them into 7 major areas -

- Display of Information.
- Design.
- Simulation and Animation.
- User Interfaces.
- Graphs and Charts, CAD, Data Visualizations
- Image Processing, Education and Training, Entertainment Etc.

1.2 Introduction to OpenCV

OpenCV (Open Source Computer Vision Library) is an open-source computer vision and machine learning software library. OpenCV provides a comprehensive set of tools and utilities for processing and analyzing visual data, making it a versatile environment for developing portable interactive 2D and 3D applications in various fields such as robotics, augmented reality, and image and video analysis.

OpenCV was initially developed by Intel in 1999 and later supported by Willow Garage and Itseez. It is widely used in academic research, commercial applications, and industrial settings due to its extensive functionalities and cross-platform nature. OpenCV is written in C++ and has bindings in Python, Java, and MATLAB/Octave, making it accessible for a wide range of programming environments.

The library contains over 2,500 optimized algorithms, including a comprehensive set of computer vision and machine learning tools. These algorithms cover a wide array of functionalities such as image processing, feature extraction, object detection, facial recognition, and camera calibration. OpenCV also supports various image formats and offers real-time performance capabilities, making it suitable for time-sensitive applications.

Functions in the main OpenCV library have names that begin with the prefix "cv" and are stored in a library usually referred to as "cv" or "cv2" in Python. The core components include modules for image and video I/O, GUI capabilities, and high-level algorithms for feature detection, object tracking, and machine learning.

In addition to the core library, OpenCV includes several utility modules like the OpenCV Contrib repository, which offers experimental and additional functionalities. For windowing and interaction, OpenCV provides a simple high-level GUI module, allowing for the development of interactive applications. OpenCV's versatility and widespread use make it a cornerstone in the field of computer vision and a valuable tool for both beginners and experts.

1.2.1 OpenCV for a Developer

- OpenCV is not a programming language, but a comprehensive library with pre-defined functionality.
- Provides functions to set or get or change the state variables. Provides functions for image and video processing, such as filtering, edge detection, and transformation.
- Offers tools for feature detection, recognition, and extraction from visual data.
- Allows developers to capture, manipulate, and display images and videos in various formats.
- Includes machine learning tools for training models and performing classifications and regressions.
- Platform Independent
- Open Source Computer Vision Library is a cross-language, cross-platform software library for computer vision and machine learning. The library provides a comprehensive set of algorithms and tools for processing and analyzing visual data, making it suitable for a wide range of applications, from simple image manipulations to advanced computer vision tasks.

CHAPTER 2

DESCRIPTION OF TOPIC

2.1 Real-Time Head Pose Estimation Using OpenCV

The project develops a real-time head pose estimation system using OpenCV and Mediapipe, capturing live video via webcam to accurately estimate head orientation. It provides real-time feedback and visualization, determining whether the user is looking left, right, up, down, or straight ahead. Applications include virtual reality, human-computer interaction, and proctored exams.

2.2 Description

This mini-project explores the real-time detection and estimation of head poses, a critical component in applications like virtual reality, facial recognition, and user interaction systems. The project utilizes OpenCV for image processing and Mediapipe for facial landmark detection to estimate the 3D orientation of a user's head. The system tracks specific facial landmarks, calculates the head's rotation and tilt angles, and provides visual feedback through an on-screen display. The project aims to simulate an engaging and interactive experience by demonstrating how head pose estimation can enhance user interfaces and applications. The system not only estimates the direction of the user's gaze but also supports the identification of potential cheating behavior in proctored exams by monitoring head movements.

CHAPTER 3

REQUIREMENTS SPECIFICATION

A Software requirement definition is an abstract description of the services which the system should provide, and the constraints under which the system must operate. It should only specify the external behavior of the system.

3.1 Hardware Requirements

1. **Processor:** 1.5 GHz or faster processor.
2. **Processor Speed:** Intel Core i3 or better, or AMD Ryzen 3 or better.
3. **RAM:** 4 GB internal RAM.
4. **Graphics Tools:** OpenCV-compatible graphics hardware.
5. **Operating System:** Windows 10 or later, or Ubuntu 18.04 LTS or later
6. **Camera:** A webcam with a resolution of at least 720p for optimal performance.
7. **Monitor Resolution**: A color monitor with a minimum resolution of 1280x720 for better visibility

3.2 Software Requirements

1. **Operating system** : Windows or Ubuntu
2. **Other Software** : Python (recommended), C++ (optional).
3. **Libraries and Frameworks** :
    - **OpenCV**: For image processing and computer vision tasks.
    - **Mediapipe**: For facial landmark detection and head pose estimation.
4. **IDE/Code Editor**: An integrated development environment (IDE) such as PyCharm, Visual Studio Code, or any preferred code editor.

CHAPTER 4

ADVANTAGES and DISADVATAGES of OPENCV

4.1 Advantages

1. OpenCV is compatible with multiple operating systems, including Windows, Linux, and macOS.
2. Optimized for real-time applications, making it suitable for video processing and live analysis.
3. Provides bindings for multiple programming languages, including Python, C++, and Java, enhancing its accessibility for developers.
4. Backed by a strong open-source community, providing extensive documentation, tutorials, and support.
5. Supports integration with hardware acceleration technologies such as CUDA and OpenCL for improved performance on compatible devices.
6. Modular design allows developers to use specific components or extend functionalities as needed.

4.2 Disadvantages

1. Can be complex to set up and integrate with other libraries or frameworks, especially for beginners..
2. While it provides many computer vision functions, it lacks some higher-level features and may require additional libraries or custom implementations for specific tasks.
3. Performance can vary depending on the hardware and the specific algorithms used; optimization may be necessary for high-performance applications

CHAPTER 5

OPENCV FUNCTIONS

3.1 OpenCV FUNCTION

- **cv2.VideoCapture()**

  Initializes the video capture object to access the webcam and opens the default camera.

- **cv2.cvtColor()**

  Converts images between different color spaces.

- **cv2.flip()**

  Flips an image horizontally or vertically.

- **cv2.imshow()**

  Waits for a key event for a specified amount of time.

- **cv2.destroyAllWindows()**

  Closes all OpenCV windows and Called at the end to clean up any OpenCV windows opened during the program.

- **cv2.projectPoints()**

  Projects 3D points onto a 2D plane and Used to project the 3D coordinates of the nose onto the 2D image plane.

- **cv2.solvePnP()**

  Solves the Perspective-n-Point problem and Estimates the pose of a 3D object based on its 2D projection.

- **cv2.Rodrigues()**

  Converts rotation vectors to rotation matrices and Used to obtain the rotation matrix from the rotation vector.

CHAPTER 6

IMPLEMENTATION

4.1 SOURCE CODE

```
import cv2
import mediapipe as mp
import numpy as np
import threading as th

x = 0
y = 0

X_AXIS_CHEAT = 0
Y_AXIS_CHEAT = 0

def pose():
    global VOLUME_NORM, x, y, X_AXIS_CHEAT, Y_AXIS_CHEAT
    mp_face_mesh = mp.solutions.face_mesh
    face_mesh=mp_face_mesh.FaceMesh(min_detection_confidence=0.5,
min_tracking_confidence=0.5)
    cap = cv2.VideoCapture(0)
    mp_drawing = mp.solutions.drawing_utils

    while cap.isOpened():
        success, image = cap.read()
        image = cv2.cvtColor(cv2.flip(image, 1), cv2.COLOR_BGR2RGB)
        image.flags.writeable = False
        results = face_mesh.process(image)
        image.flags.writeable = True
        image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
```

```
img_h, img_w, img_c = image.shape
face_3d = []
face_2d = []

face_ids = [33, 263, 1, 61, 291, 199]

if results.multi_face_landmarks:
    for face_landmarks in results.multi_face_landmarks:
        mp_drawing.draw_landmarks(
            image=image,
            landmark_list=face_landmarks,
            connections=mp_face_mesh.FACEMESH_CONTOURS,
            landmark_drawing_spec=None)
        for idx, lm in enumerate(face_landmarks.landmark):
            if idx in face_ids:
                if idx == 1:
                    nose_2d = (lm.x * img_w, lm.y * img_h)
                    nose_3d = (lm.x * img_w, lm.y * img_h, lm.z * 8000)

                x, y = int(lm.x * img_w), int(lm.y * img_h)
                face_2d.append([x, y])
                face_3d.append([x, y, lm.z])

        face_2d = np.array(face_2d, dtype=np.float64)
        face_3d = np.array(face_3d, dtype=np.float64)
        focal_length = 1 * img_w

        cam_matrix = np.array([[focal_length, 0, img_h / 2],
                               [0, focal_length, img_w / 2],
                               [0, 0, 1]])
        dist_matrix = np.zeros((4, 1), dtype=np.float64)
        success, rot_vec, trans_vec = cv2.solvePnP(face_3d, face_2d, cam_matrix,
dist_matrix)
        rmat, jac = cv2.Rodrigues(rot_vec)
        angles, mtxR, mtxQ, Qx, Qy, Qz = cv2.RQDecomp3x3(rmat)
        x = angles[0] * 360
        y = angles[1] * 360

        if y < -10:
            text = "Looking Left"
        elif y > 10:
            text = "Looking Right"
        elif x < -10:
            text = "Looking Down"
        else:
            text = "Forward"
        text = str(int(x)) + "::" + str(int(y)) + text

        # Y is left / right
        # X is up / down
        if y < -10 or y > 10:
            X_AXIS_CHEAT = 1
        else:
            X_AXIS_CHEAT = 0

        if x < -5:
            Y_AXIS_CHEAT = 1
        else:
            Y_AXIS_CHEAT = 0
        nose_3d_projection, jacobian = cv2.projectPoints(nose_3d, rot_vec, trans_vec,
cam_matrix, dist_matrix)
```

```
            p1 = (int(nose_2d[0]), int(nose_2d[1]))
            p2 = (int(nose_3d_projection[0][0][0]), int(nose_3d_projection[0][0][1]))

            cv2.line(image, p1, p2, (255, 0, 0), 2)

            # Add the text on the image
            cv2.putText(image, text,  (20,  20),  cv2.FONT_HERSHEY_SIMPLEX,  1,  (0,  0,
255), 2)

        cv2.imshow('Head Pose Estimation', image)

        if cv2.waitKey(5) & 0xFF == ord('q'):
            break

    cap.release()
    cv2.destroyAllWindows()

if__name__== "__main__":
    t1 = th.Thread(target=pose)

    t1.start()
    t1.join()
```

CHAPTER 7

SNAPSHOTS

7.1 SNAPSHOTS

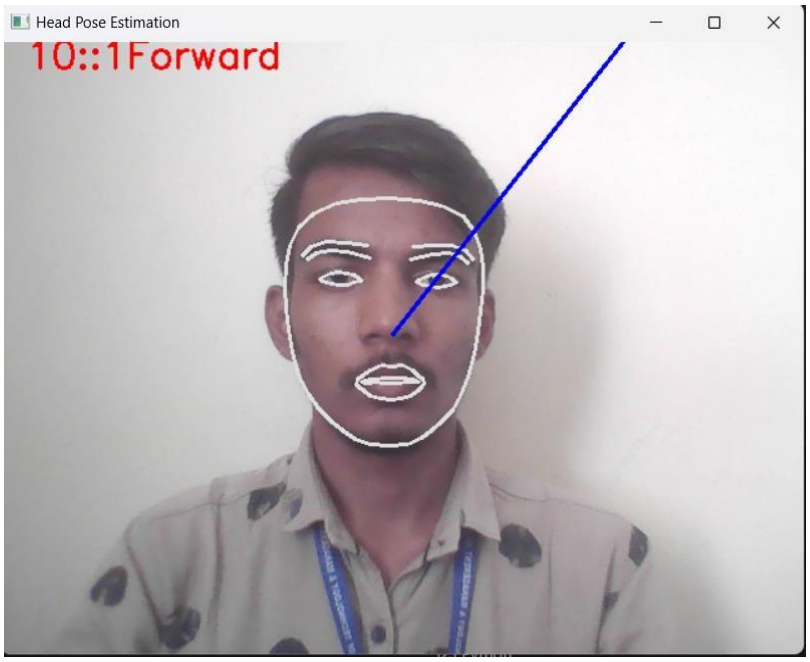

Fig 7.1a: Looking Forward

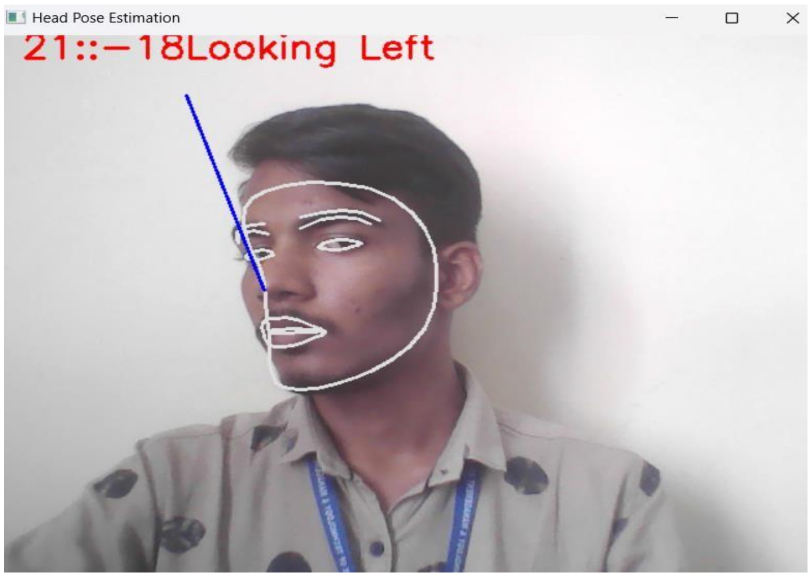

Fig 7.1b: Looking Left

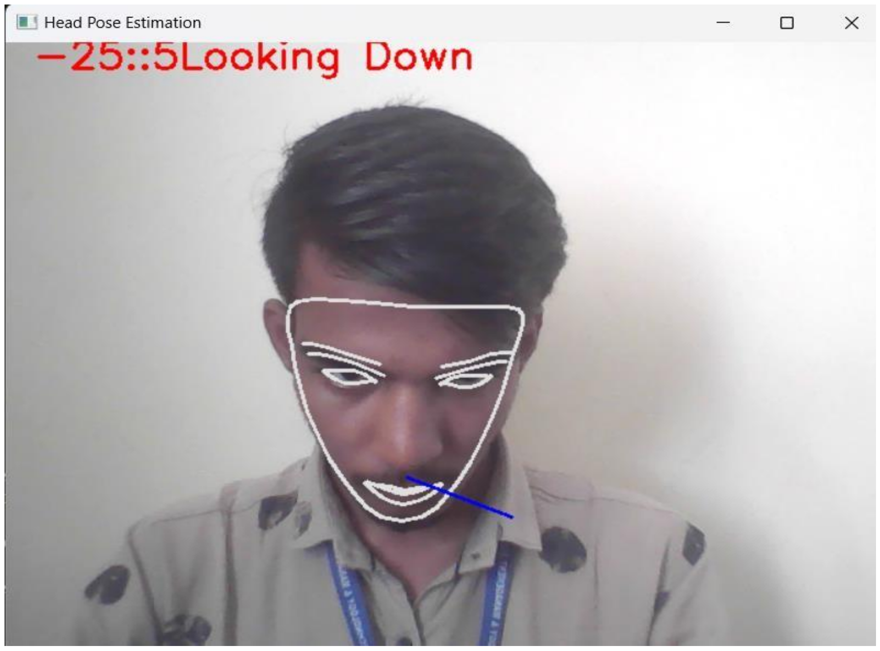

Fig 7.1c: Looking Down

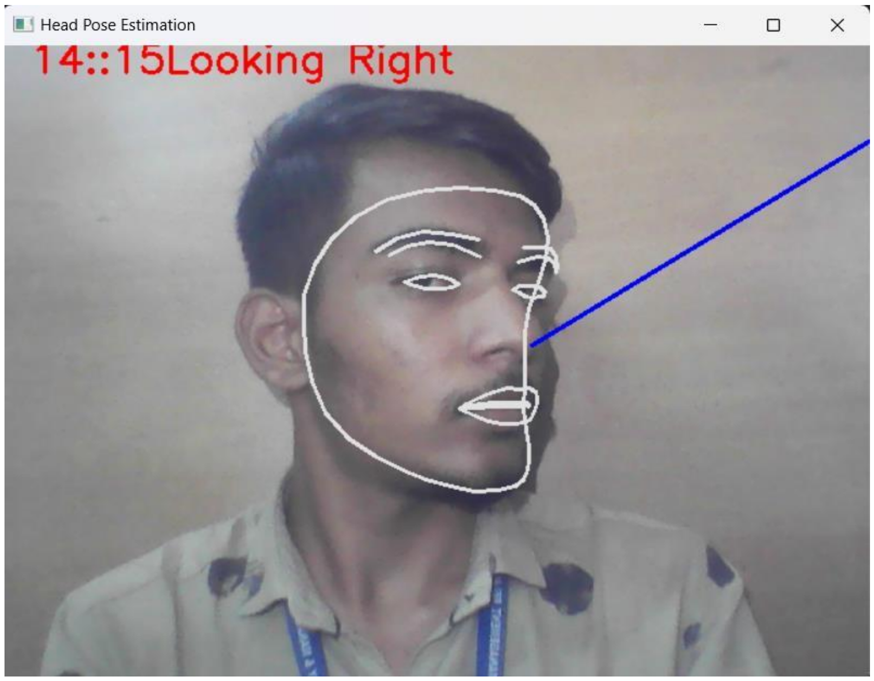

Fig 7.1d: Looking Right

CONCLUSION

The project successfully implements a real-time head pose estimation system using OpenCV and Mediapipe. By leveraging facial landmark detection and solving the Perspective-n-Point (PnP) problem, the system accurately determines head orientation, providing insights into user gaze direction. The application processes video input from a webcam, delivering immediate visual feedback with annotated head pose information on the video feed. The use of multi-threading ensures smooth performance, making the system responsive and efficient. Overall, the project effectively demonstrates the potential of computer vision techniques in real-time user interaction and opens avenues for further development in applications such as user experience and accessibility technologies.

REFERENCES

1. Gary Bradski and Adrian Kaehler: *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library*, O'Reilly Media, 2016.
2. David L. Poole, Alan K. Mackworth: *Artificial Intelligence: Foundations of Computational Agents*, Cambridge University Press, 2017.

WEBSITES

1. https://docs.opencv.org/
2. https://mediapipe.readthedocs.io/
3. https://nptel.ac.in/courses/108103174/
4. https://www.geeksforgeeks.org/opencv-python-tutorial/