DAYANANDA SAGAR UNIVERSITY

Devarakaggalahalli, Harohalli, Kanakapura Road, Ramanagara Dt., Bengaluru – 562 112

Bachelor of Technology
in
COMPUTER SCIENCE AND ENGINEERING
(ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)

AI Mini Project

Automation of the Game “Hill climb Racing”

By

Chethan K Murthy [ENG22AM0009]
Neha Amin [ENG22AM0117]

Under the supervision of,

Dr. JAYAVRINDA VRINDAVANAM **Chairperson, CSE(AI&ML)**

Prof. PRADEEP KUMAR **Assistant Professor, CSE(AI &ML)**

Prof. MITHAGURU **Assistant Professor, CSE(AI &ML)**

Dr. MARY JASMINE **Assistant Professor, CSE(AI & ML)**

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING (AI&ML),
SCHOOL OF ENGINEERING
DAYANANDA SAGAR UNIVERSITY,BANGALORE
(2023-2024)

Dayananda Sagar University

School of Engineering

Department of Computer Science & Engineering

(Artificial Intelligence & Machine Learning)

Devarakaggalahalli, Harohalli, Kanakapura Road, Ramanagara Dt., Bengaluru – 562 112

CERTIFICATE

This is to certify that the AI Mini Project – I (22AMXXXX) work titled **"**Automation of the Game "Hill climb Racing" is carried out by **Chethan K Murthy** bearing **USN :ENG22AM0009,Neha Amin** bearing **USN:ENG22AM0117,** Bonafede students of Bachelor of Technology in Computer Science and Engineering (AI&ML) at the School of Engineering, Dayananda Sagar University, Bangalore in partial fulfillment for the award of degree in Bachelor of Technology in Computer Science and Engineering, during the year **2023-2024**.

Prof. Pradeep Kumar K
Assistant Professor
Dept. of CSE (AI&ML),
School of Engineering
Dayananda Sagar University
Date: 0 1 -01-2024

Dr. Jayavrinda Vrindavanam V
Chairman CSE (AI&ML)
School of Engineering
Dayananda Sagar University
Date: 01-01-2024

Prof. Mithaguru
Assistant Professor
Dept. of CSE(AI&ML),
School of Engineering
Dayananda Sagar University,
Date: 01-01-2024

Dr. Mary Jasmine
Assistant Professor
Dept of CSE(AI&ML)
School of Engineering,
Dayananda Sagar University,
Date:01-01-2024

DECLARATION

We, **Chethan K Murthy and Neha Amin** students of third semester B. Tech in **Computer Science and Engineering with speciation in Artificial intelligence and machine learning** , at School of Engineering, **Dayananda Sagar University**, hereby declare that the **AI Mini Project** titled **"**Automation of the Game "Hill climb Racing"**"** has been carried out by us and submitted to Prof. Pradeep Kumar during the academic year **2023-2024**.

Student 1: **Signature**

Name: Chethan K Murthy

USN: ENG22AM0009

Student 2: **Signature**

Name: Neha Amin

USN: ENG22AM0117

Place: Bangalore
Date: 01-01-2024

ACKNOWLEDGEMENT

It is a great pleasure for me to acknowledge the assistance and support of many individuals who have been responsible for the successful completion of this project work.

First, I take this opportunity to express my sincere gratitude to School of Engineering & Technology, Dayananda Sagar University for providing us with a great opportunity to pursue our Bachelor's degree in this institution.

I would like to thank ***Dr. Udaya Kumar Reddy K R, Dean, School of Engineering & Technology, Dayananda Sagar University*** *for his constant encouragement and expert advice.*

It is a matter of immense pleasure to express our sincere thanks to ***Dr. Jayvrinda Vrindavanam v, Department Chairman****,* ***Computer Science and Engineering(AI AND ML)****,* ***Dayananda Sagar University,*** *for providing right academic guidance that made the task possible.*

I would like to thank our guide ***Prof. Pradeep Kumar K, Prof. Mithaguru, Dr. Mary Jasmine Assistant Professors****,* ***Dept. of Computer Science and Engineering (AI AND ML)****,* ***Dayananda Sagar University****, for sparing his valuable time to extend help in every step of the project work, which paved the way for smooth progress and fruitful culmination of the project.*

I am also grateful to my family and friends who provided me with everyrequirement throughout the course.

I would like to thank one and all who directly or indirectly helped me in theProject work

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

LLM	Large Language Modelling
AI	Artificial Intelligence
NLP	Natural Language Processing
RL	Reinforcement Learning

ABSTRACT

This AI-driven project revolutionizes the gaming experience by introducing gesture-based controls to automate Hill Climb Racing. Leveraging cutting-edge technologies in computer vision and machine learning, the system enables players to control vehicle acceleration and braking through intuitive hand gestures, transcending traditional input methods. The implementation harnesses the power of OpenCV for real-time video processing and MediaPipe's hand tracking capabilities to accurately detect and interpret hand movements.

Through a sophisticated pipeline, the AI model identifies and tracks specific hand gestures, allowing users to seamlessly manipulate the in-game controls. Hand landmarks are detected, analyzed, and translated into game commands, empowering players to accelerate or brake by gesturing, creating a natural and immersive gaming interface.

This innovative integration of AI and gaming not only enriches user interaction but also showcases the potential for intuitive and accessible interfaces in gaming technology. The project highlights the fusion of computer vision, machine learning, and gaming to deliver a novel gaming experience, laying the foundation for future advancements in gesture-controlled gaming interfaces and emphasizing the boundless possibilities of AI-driven innovations in entertainment.

CHAPTER 1
INTRODUCTION

Introducing a groundbreaking synergy of artificial intelligence and gaming, this project redefines the conventional gameplay experience of Hill Climb Racing. By ingeniously integrating AI algorithms with gesture recognition technology, this innovation allows players to transcend traditional controls. Through the fusion of computer vision frameworks like OpenCV and hand tracking capabilities of MediaPipe, users gain the ability to control acceleration and braking in the game through natural hand movements.

This groundbreaking approach revolutionizes user interaction within gaming realms, offering a seamless and immersive experience by translating hand gestures into in-game actions. The project not only showcases the possibilities of AI-driven interfaces in gaming but also sets a precedent for intuitive, accessible, and engaging interactions between users and virtual environments. Emphasizing innovation at the intersection of AI and entertainment, this venture serves as a milestone in the evolution of gaming interfaces, paving the way for future advancements in gesture-controlled gaming experiences..

1.1 KEY ISSUES:

Key Issues:

1. Accuracy of Gesture Recognition:
Achieving precise and consistent recognition of hand gestures poses a challenge. Variability in lighting conditions, hand orientations, and occlusions can affect the accuracy of gesture detection, impacting the gameplay experience. Robust algorithms must handle these variations to ensure reliable control.

2. Real-Time Responsiveness:
Maintaining real-time responsiveness is crucial for seamless gameplay. Processing the video feed, detecting gestures, and translating them into game commands must occur swiftly to avoid latency. Balancing accuracy with speed in gesture recognition algorithms is essential for an immersive gaming experience.

3. User Interface Integration:
Seamlessly integrating gesture controls into the game's user interface requires careful design. Ensuring that the gesture-based input aligns with the game's mechanics and provides an intuitive experience is vital. User feedback and ergonomic considerations play a crucial role in optimizing this integration.

4. Adaptability and Robustness:
The system must be adaptable to diverse user behaviors and hand sizes while maintaining robustness across different environments. Ensuring that the AI model generalizes well and functions reliably for various users and settings is imperative for widespread adoption and user satisfaction. Calibration and adaptability are key for accommodating a broad user base.

1.2 OBJECTIVES:

1. Develop Accurate Gesture Recognition:

- The primary goal is to create a robust gesture recognition system capable of accurately identifying hand gestures in varying conditions. Utilizing advanced computer vision techniques, the system aims to precisely detect and interpret gestures for seamless gameplay, ensuring consistent and reliable control of acceleration and braking.

2. Ensure Real-Time Responsiveness:

- Focusing on optimizing algorithms and processing pipelines, the project aims to achieve real-time responsiveness between gesture detection and game control. Minimizing latency is crucial for an immersive gaming experience, requiring efficient methodologies for swift gesture analysis and translation into in-game actions without compromising accuracy.

3. Integrate Intuitive User Interface:

- The objective involves seamlessly integrating gesture controls into the game's user interface. Designing an intuitive interaction mechanism aligns hand gestures with game mechanics, enhancing user experience. User-centric interface design and ergonomic considerations play a pivotal role in ensuring that gesture-based controls feel natural and enhance gameplay.

4. Ensure Adaptability and Robustness:

- The project strives for adaptability across diverse user demographics and environmental conditions. The system aims to be robust, accommodating different hand sizes, orientations, and lighting scenarios. Ensuring the AI model's generalizability is crucial for widespread adoption, requiring calibration and adaptability for a broad user base.

1.3 Computer Vision

Computer vision refers to the field of artificial intelligence and computer science dedicated to enabling machines to interpret, understand, and analyze visual information from the real world, akin to human vision. It involves developing algorithms, techniques, and methodologies that allow machines, typically computers or robots, to gain high-level understanding and make decisions based on visual data captured through cameras or other imaging devices.

At its core, computer vision aims to replicate and expand upon human vision capabilities using computational models and mathematical algorithms. It encompasses a broad spectrum of tasks:

1. Image Processing: Involves manipulating and enhancing images to extract useful information or enhance their quality. This includes operations like filtering, segmentation, and noise reduction.

2. Object Detection and Recognition: Enables machines to identify and locate objects within images or videos. This involves algorithms capable of recognizing specific objects, faces, shapes, or patterns.

3. Scene Understanding: Involves understanding the context of an image or a series of images. It includes tasks like scene classification, semantic segmentation, and understanding relationships between objects in a scene.

4. Motion Analysis: Focuses on analyzing the movement of objects within videos or sequences of images. This includes tasks such as optical flow, object tracking, and activity recognition.

Computer vision finds applications across diverse industries and domains. In healthcare, it aids in medical imaging analysis and diagnostics. In autonomous vehicles, it enables object detection and navigation. In manufacturing, it assists in quality control and robotic automation. In entertainment, it powers augmented reality and facial recognition technologies.

1.4 Natural Language Processing (NLP):

Natural Language Processing is a pivotal domain in AI that empowers machines to comprehend, interpret, and generate human language. In your project, NLP techniques form the basis for tasks ranging from sentiment analysis and language translation to named entity recognition. Through the application of algorithms and linguistic principles, your system gains the ability to not only understand the structure of language but also extract meaningful insights from textual data. This domain facilitates seamless human-machine communication, making your AI project capable of engaging in sophisticated interactions, understanding context, and providing contextually relevant responses.

1.5 Reinforcement Learning:

Reinforcement Learning is a dynamic domain within AI where agents learn to make decisions by interacting with an environment and receiving feedback in the form of rewards. In your project, Reinforcement Learning could be employed for tasks that require continuous adaptation and improvement based on trial and error. This domain is particularly valuable in scenarios where explicit programming might be challenging, and the system needs to autonomously learn optimal strategies. Whether applied to game playing, robotics, or decision-making processes, Reinforcement Learning enhances your AI project's adaptability and capability to navigate complex, dynamic environments through learned experience.

CHAPTER 2 LITERATURE REVIEW

Article 1

Dynamic gesture recognition based on 2D convolutional neural network and feature fusion

Review

The document discusses the impacts and challenges of artificial intelligence (AI) translation tools on translation professionals. It highlights the increasing reliance on AI translation and its impact on the translation industry. The study aims to determine the advantages and disadvantages of AI translations compared to human translations. The literature review emphasizes the importance of developing high-quality translations amidst the rapid development of AI. It also highlights the subjective nature of existing translation standards and the need for an objective evaluation system.

The research design involves a comparative experiment with three types of articles: business texts, news reports, and literary works. The experiment uses evaluation criteria to compare human and AI translations and involves English professionals and non-professionals as study subjects. The findings of the experiment indicate that human translations outperformed AI translations in terms of accuracy, linguistic style, connotations of the source language, ideology, cultural context, and logical expression.

The experimental results reveal that AI translations excel in language style, especially in business English translations, but lag behind human translations in logical expression and fidelity to the original text. Respondents' feedback acknowledges the potential of AI translation but emphasizes the need for improvements in context analysis and logical language expression. They express a positive attitude towards AI translation tools and suggest collaboration between human translators and AI to ensure efficiency and accuracy.

translation industry.

Article 2

Human Pose Estimation Using MediaPipe Pose and Optimization Method Based on a Humanoid Model

Authors:Joon wok kim,Jin young chi

Link: https://www.mdpi.com/2076-3417/13/4/2700

Review

The paper addresses the challenge of monitoring seniors living independently by proposing a mobile robot equipped with a camera to capture poses and automatically analyze them for potential dangers. Humanoid joint angles estimation, crucial for recognizing poses, is typically solved using inverse kinematics (IK) for humanoid robots mimicking the human body structure. Traditional methods like geometric relationships and optimization algorithms prove challenging for various poses. Expensive motion capture systems provide precise 3D joint data, prompting the authors to propose a simpler method: photographing subjects with a 2D camera and using optimization algorithms with a 3D humanoid model.

The paper explores human pose estimation's widespread applications and classifies methods into 2D/3D coordinate estimation, single/multiple person-based, and single-image/video-based categories. It differentiates single-stage (direct 3D mapping) and two-stage methods (lifting from 2D to 3D) in deep learning-based pose estimation, highlighting their advantages and limitations. Despite technological advances, challenges like dataset scarcity, diverse poses, depth ambiguities, and computational demands persist in 3D pose estimation from 2D images.

The proposed system employs a two-stage approach: using MediaPipe Pose for 2D estimation and a fast optimization method, uDEAS, for joint angle estimation based on a 3D humanoid model. The paper introduces a 3D full-body humanoid model with added lumbar joints and a consistent joint rotation design. uDEAS optimizes joint angles to closely match the 2D humanoid model from 3D projections. This approach tackles data scarcity, pose variation, depth ambiguities, and computational complexity issues.

Experimental validations through simulations and real-world motion capture showcase the system's practicality for monitoring poses relevant to seniors. It achieves acceptable 3D pose estimation results suitable for mobile robot applications without requiring high-end GPU hardware. The contributions include the novel humanoid model, the uDEAS-based optimization approach, and addressing depth ambiguity challenges. The paper outlines the proposed system's structure, experimental results, and concludes by discussing future work in five comprehensive sections.

CHAPTER 3
PROBLEM DEFINITION

The project aims to develop a system that enables gesture-based control for game interaction, specifically targeting Hill Climb Racing. The primary challenge lies in creating a reliable and accurate mechanism that translates hand gestures into game commands for acceleration and braking. This involves real-time hand detection and tracking through computer vision techniques, interpreting various hand poses and movements to discern the user's intent. The system needs to accurately identify and differentiate between open and closed fingers to determine actions within the game. Additionally, ensuring seamless integration of this gesture recognition system with the game interface presents a technical hurdle. The solution demands robustness in recognizing hand gestures across diverse environmental conditions, lighting variations, and different hand orientations. Furthermore, optimizing this solution for real-time performance is crucial to deliver a responsive and enjoyable gaming experience. The project aims to overcome these challenges by leveraging OpenCV and MediaPipe libraries for hand detection and gesture recognition, integrating these functionalities into the game environment to provide users with an intuitive and immersive gaming interaction through hand gestures.

CHAPTER 4
METHODOLOGY

4.1 Code Implementation

In [ ]:
```
import cv2 as cv
import mediapipe as mp
from pynput.keyboard import Key, Controller

# Initialize Keyboard Controller
keyboard = Controller()

mp_draw = mp.solutions.drawing_utils # Function to Draw Landmarks over Hand
mp_hand = mp.solutions.hands # Hand Detection Function

fingerTipIds = [4, 8, 12, 16, 20]

# Capturing the Video from the Camera
video = cv.VideoCapture(0)

# Initializing the Hand Detection Function
hands = mp_hand.Hands(min_detection_confidence = 0.5, min_tracking_confidence = 0.5)

while True:
    success, image = video.read()

    # Converting the Image to RGB
    image = cv.cvtColor(image, cv.COLOR_BGR2RGB)

    # Processing the Image for Hand Detection
    image.flags.writeable = False
    results = hands.process(image)
    image.flags.writeable = True

    # Converting the Image back to BGR
    image = cv.cvtColor(image, cv.COLOR_RGB2BGR)

    # List to store the Landmark's Coordinates
    landmarks_list = []

    # If Landmarks Detected i.e., Hand Detected Sucessfully
    if results.multi_hand_landmarks:
        hand_landmarks = results.multi_hand_landmarks[-1]

        for index, lm in enumerate(hand_landmarks.landmark):
            h, w, c = image.shape # Height, Width, Channels
            cx, cy = int(lm.x*w), int(lm.y*h)
            landmarks_list.append([index, cx, cy])

        # Drawing the Landmarks for only One Hand
        # Landmarks will be drawn for the Hand which was Detected First
        mp_draw.draw_landmarks(image, hand_landmarks, mp_hand.HAND_CONNECTIONS)
```

```python
    # Stores 1 if finger is Open and 0 if finger is closed
    fingers_open = []

    if len(landmarks_list) != 0:
        for tipId in fingerTipIds:
            if tipId == 4: # That is the thumb
                if landmarks_list[tipId][1] > landmarks_list[tipId - 1][1]:
                    fingers_open.append(1)
                else:
                    fingers_open.append(0)
            else:
                if landmarks_list[tipId][2] < landmarks_list[tipId - 2][2]:
                    fingers_open.append(1)
                else:
                    fingers_open.append(0)

    # Counts the Number of Fingers Open
    count_fingers_open = fingers_open.count(1)

    # If Hand Detected
    if results.multi_hand_landmarks != None:
        # If All Fingers Closed --> Break
        if count_fingers_open == 0:
            cv.rectangle(image, (20, 300), (270, 425), (0, 255, 0), cv.FILLED)
            cv.putText(image, "BRAKE", (45, 375), cv.FONT_HERSHEY_SIMPLEX, 2, (255, 0, 0), 5)

            keyboard.press(Key.left)
            keyboard.release(Key.right)

        # If All Fingers Open --> Gas
        elif count_fingers_open == 5:
            cv.rectangle(image, (20, 300), (270, 425), (0, 255, 0), cv.FILLED)
            cv.putText(image, "GAS", (45, 375), cv.FONT_HERSHEY_SIMPLEX, 2, (255, 0, 0), 5)

            keyboard.press(Key.right)
            keyboard.release(Key.left)

    # If No Hand Detected
    else:
        keyboard.release(Key.right)
        keyboard.release(Key.left)

    # Show the Video
    cv.imshow("Frame", image)

    # Close the Video if "q" key is pressed
    if cv.waitKey(1) & 0xFF == ord('q'):
        break

video.release()
cv.destroyAllWindows()
```

This code employs computer vision and the MediaPipe library to detect and analyze hand gestures from a live video feed, specifically aimed at recognizing gestures for controlling a game or application. The primary steps involve:

1. Initializing Components: It sets up the keyboard controller, initializes the MediaPipe hands module for hand detection, and defines an array storing finger tip IDs.

2. Video Capture and Hand Detection: It captures video from the camera, processes each frame by converting it to RGB, detecting hand landmarks using MediaPipe, and converting the image back to BGR for visualization.

3. Landmark Detection and Gesture Recognition: It stores the coordinates of detected hand landmarks, draws them on the frame, and assesses finger positions to determine gestures (like open or closed fingers). Depending on the recognized gesture (e.g., fingers open or closed), it simulates keyboard inputs accordingly, mapping them to actions (e.g., pressing keys for gas or brake).

4. Display and Exit: The video feed with detected landmarks and recognized gestures is displayed. The code exits the loop and closes the video feed if the 'q' key is pressed.

Overall, this code integrates hand gesture recognition using MediaPipe, identifies specific gestures based on finger positions, and simulates keyboard inputs accordingly, making it suitable for applications where hand gestures control actions, such as gaming or other interactive experiences.

4.2 Algorithm

Initialization:

1. Import necessary libraries: OpenCV (`cv2`), MediaPipe (`mediapipe`), and `pynput` for keyboard control.
2. Set up keyboard controller using `Controller()` from `pynput`.
3. Define `mp_draw` and `mp_hand` modules for drawing hand landmarks and hand detection functions from MediaPipe.
4. Create an array `fingerTipIds` to store finger tip landmark IDs.

Video Capture and Hand Detection Loop:

5. Open the camera feed using `cv.VideoCapture()`.
6. Initialize the MediaPipe Hand module (`mp_hand.Hands()`), setting detection and tracking confidence thresholds.
7. Start an infinite loop for continuous video processing.
   - Read each frame from the video feed.
   - Convert the frame from BGR to RGB for hand detection processing.
   - Process the image using MediaPipe hands, obtaining hand landmarks.
   - Convert the image back to BGR for display purposes.
   - Store detected hand landmarks' coordinates in a list.
   - Draw the landmarks on the image for visualization.
   - Determine finger positions by analyzing landmark coordinates.
   - Count the number of open fingers.
   - If hand landmarks are detected:
     - Based on the finger count, simulate corresponding keyboard actions (e.g., gas or brake).
   - Display the frame with hand landmarks and recognized gestures.
   - Check for the 'q' key press to exit the loop.

Video Cleanup:

8. Release the video capture object (`video.release()`).
9. Close all OpenCV windows (`cv.destroyAllWindows()`).

End of Algorithm

CHAPTER 5
CONCLUSION

- In conclusion, the development of this project has successfully brought forth a versatile and intelligent chatbot capable of seamlessly addressing user queries and facilitating language translation. Through the integration of advanced natural language processing (NLP) algorithms and translation services, our chatbot has demonstrated its proficiency in understanding and responding to a diverse range of questions in a user-friendly manner.
- The incorporation of language translation functionality further extends the reach and usefulness of the chatbot, breaking down communication barriers and fostering cross-cultural interactions. Users can now effortlessly communicate in their preferred language, while the chatbot dynamically translates and responds, enhancing the overall accessibility and inclusivity of the system.
- As technology continues to evolve, this project serves as a testament to the power of artificial intelligence in enhancing user experiences and providing practical solutions. The chatbot's ability to adapt and learn from user interactions positions it as a valuable tool for individuals seeking information and communication across linguistic boundaries.
- Moving forward, there is potential for expansion and refinement, including the incorporation of additional languages, continuous improvement of translation accuracy, and the implementation of more sophisticated dialogue management techniques. The journey doesn't end here, and with ongoing advancements in AI and language processing, our chatbot is poised to evolve and remain at the forefront of intelligent conversational agents. Ultimately, this project represents a significant stride towards the creation of intelligent, user-centric solutions in the realm of natural language understanding and translation.

CHAPTER 6
REFERENCES

- https://developers.google.com/mediapipe

- https://www.researchgate.net/profile/Carlos-Soto-23/publication/236160250_Hand_gesture_recognition_with_depth_images_A_review/links/5550e98408ae956a5d25db4d/Hand-gesture-recognition-with-depth-images-A-review.pdf

- https://scholar.google.co.in/scholar_url?url=https://www.sciencedirect.com/science/article/pii/S0262885602001130&hl=en&sa=X&ei=dlmRZeSVIqGJ6rQP9_y8yAg&scisig=AFWwaeaZPm0M08u2RzeLwsos7qVz&oi=scholarr

- https://opencv.org/