Hand Gesture Recognition Application

Ms. Jiaqi Zhao
Khoury College of Computer Science
Northeastern University
San Jose, United States
zhao.jiaqi2@northeastern.edu

Ms. Kexuan Chen
Khoury College of Computer Science
Northeastern University
San Jose, United States
chen.kexua@northeastern.edu

Ms. Zhimin Liang
Khoury College of Computer Science
Northeastern University
Vancouver, Canada
liang.zhimi@northeastern.edu

***Abstract*—In today's digital age, presentations play a pivotal role in various domains, necessitating more intuitive and engaging interaction methods. This project proposes a novel Dynamic Hand Gesture Recognition System Application tailored to revolutionize presentation delivery by enabling gesture-controlled navigation and interaction with PowerPoint slides. Leveraging convolutional neural networks (CNN) trained on the LeapGesture dataset and the MediaPipe framework for real-time hand tracking, the system interprets specific hand gestures captured by a webcam to execute commands such as navigating slides and controlling pointers. Despite challenges in model accuracy, integration with MediaPipe enhances real-time performance, showcasing promising results for practical application. By offering a more natural and immersive interaction experience, this system aims to enhance presenter-audience engagement and facilitate dynamic presentations.**

***Index Terms*—Gesture recognition, Computer vision, Human-computer interaction,convolutional neural networks (CNN)**

I. Introduction

Gesture recognition technology has gained significant attention in recent years due to its wide range of applications across various fields, including entertainment, automation, healthcare, and academia. By interpreting human gestures as commands, these systems offer a more natural and seamless way of interacting with computers, reducing the need for physical input devices and enhancing user productivity.

The demand for intuitive and immersive user interfaces has never been greater. Traditional methods of interacting with digital systems, such as keyboards and mice, often impose limitations on user engagement and mobility. As a response to these challenges, there has been a burgeoning interest in developing innovative systems that leverage hand gestures for interaction.

Our project, the Dynamic Hand Gesture Recognition System Application, seeks to address this need by introducing a novel approach to interaction with digital systems. By harnessing the power of computer vision techniques and machine learning algorithms, our system aims to revolutionize the way users interact with various applications, particularly in the context of presentations.

Key concepts underlying this project include gesture recognition, computer vision, and human-computer interaction. Gesture recognition involves identifying and interpreting specific hand movements as commands, while computer vision enables the system to analyze and understand these gestures through image processing techniques. Human-computer interaction principles guide the design of intuitive and user-friendly controls that facilitate seamless interaction between the presenter and the presentation software.

By implementing a range of specific gestures, such as navigating to the previous or next slide, moving the pointer, and drawing/highlighting, our system aims to provide presenters with greater flexibility and freedom during presentations. Moreover, the application of gesture-based controls extends beyond traditional input methods, offering a more accessible and engaging experience for users with mobility impairments or those who prefer non-traditional interaction methods.

Through the development of our Dynamic Hand Gesture Recognition System Application, we envision a future where digital interaction is more natural, intuitive, and inclusive. This introduction sets the stage for our project, outlining its objectives, significance, and potential impact in the realm of user interface design and interaction.

In the subsequent sections, we will discuss the related papers and delve deeper into the methodologies employed, experiments conducted, and results obtained, ultimately culminating in a comprehensive analysis and discussion of our findings.

II. Related Work

A. Controlling Media Player with Hand Gestures using Convolutional Neural Network [1]

This paper presents a method for controlling media players using hand gestures detected by a convolutional neural network (CNN). The authors develop a system that enables users to control media playback, volume adjustment, and other functions using hand gestures captured by a webcam. Similar to our project, this paper demonstrates the application of computer vision and deep learning techniques to interpret hand gestures for controlling media players. While our project focuses on PowerPoint presentation control, this paper provides valuable insights into the implementation of gesture-based control systems using CNNs.

B. Hand Gesture Controlled Video Player Application [2]

In this paper, the authors introduce a hand gesture-controlled video player application that enables users to control media playback using hand gestures captured by a webcam. The system recognizes predefined hand gestures and translates

them into commands to play, pause, rewind, or fast forward videos. This paper is directly relevant to our project as it demonstrates the feasibility and effectiveness of using hand gestures for controlling media playback. While our project focuses on controlling PowerPoint presentations, both projects share similarities in utilizing hand gestures and computer vision techniques for interaction.

C. Visual Gesture Recognition for Text Writing in Air [3]

This paper presents a novel approach to visual gesture recognition, specifically focusing on recognizing hand gestures for text writing in the air. The authors propose a system that utilizes computer vision techniques to detect hand movements and interpret them as characters or words. The system allows users to write in the air using hand gestures, which are then translated into text input. While our project focuses on controlling PowerPoint presentations rather than text writing, both projects share a common foundation in visual gesture recognition and leverage computer vision techniques to interpret hand gestures in real-time.

III. Proposed Methodology

A. CNN model

Approach: We use the LeapGesture dataset as source to build and train a network to recognize hand gestures. This dataset contains 20,000 frames of hand gestures, which includes 10 types of hand gestures. We train and test the model and try to increase the accuracy rate for the model.

Implementation: We use python package pyTorch building and training a convolutional network to recognize different types of hand gestures. Then, we test the model using a video stream and implement it in presentation control.

Drawbacks: There are some drawbacks for our trained model. The image processing is complicated, it cannot significantly eliminate the background noise while processing the gesture recognition. Also, the accuracy of trained model is unsatisfied. There are only two types of hand gestures can be recognized stably. Therefore, we introduce MediaPipe library to increase the application performance and functionality.

B. Hand Detection and Landmark Localization

Approach: We employ the MediaPipe library along with OpenCV to detect and localize hand landmarks in real-time video frames. MediaPipe provides pre-trained models for hand detection and landmark localization, which we utilize to identify the presence of hands in the video stream and localize key landmarks such as fingertips and palm centers.

Implementation: The HandDetector class encapsulates the functionality for hand detection and landmark localization. We initialize the MediaPipe hands object with parameters such as the maximum number of hands, detection confidence threshold, and tracking confidence threshold. The findHands method processes each frame to detect hands and visualize landmarks using MediaPipe's drawing utilities. The findPosition method extracts the coordinates of specific landmarks for further analysis.

C. Gesture Recognition

Approach: We define a set of specific hand gestures corresponding to presentation navigation and interaction commands. These gestures include navigating to the previous or next slide, moving the pointer, drawing/highlighting, and deleting drawings. We analyze the positions of key landmarks, such as fingertips and palm centers, to classify gestures based on predefined criteria.

Implementation: The HandDetector class includes methods for recognizing gestures, such as fingersUp to determine finger positions and getCenterIndex to calculate the center of the hand. By analyzing the relative positions of landmarks and finger configurations, we classify gestures and trigger corresponding actions, such as slide navigation or drawing operations.

IV. Application of the Methodology

A. CNN Model Training

LeapGesture dataset consists of 10 types of hand gestures. For data processing, we load the data, resize all the images to 128*128 and split the data into training sets and testing sets.

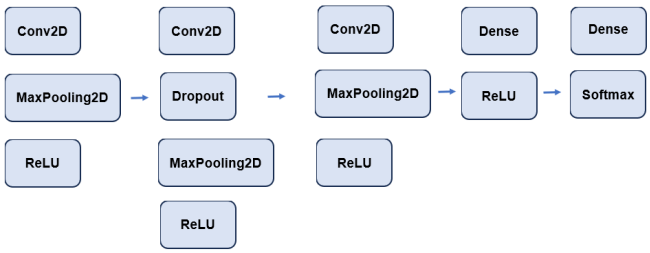


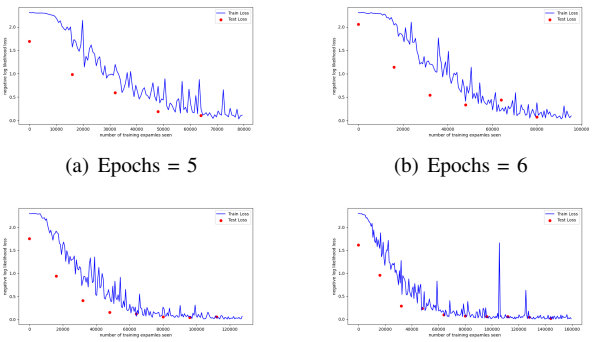
Fig. 1. Proposed CNN Model

Following is the proposed CNN model:

- A convolution layer with 32 5*5 filters
- A max pooling layer with a 2*2 window and a ReLU function applied.
- A convolution layer with 64 5*5 filters
- A dropout layer with a 0.5 dropout rate
- A max pooling layer with a 2*2 window and a ReLU function applied.
- A convolution layer with 128 5*5 filters
- A max pooling layer with a 2*2 window and a ReLU function applied.
- A flattening operation followed by a fully connected Linear layer with 128 nodes and a ReLU function on the output
- A final fully connected Linear layer with 10 nodes and the `log_softmax` function applied to the output.

After conducting several experiments, we found that the model would achieve stable accuracy after 7 epochs. Following are a visual representations of our findings.

We also use some sample image to test the trained model. However, the accuracy is not satisfied. You can see a sample



(a) Epochs = 5 (b) Epochs = 6

(c) Epochs = 8 (d) Epochs = 10

Fig. 2. Comparison of Epochs

testing result in Fig. 3. The accuracy rate of 10 hand gestures is only 30%.

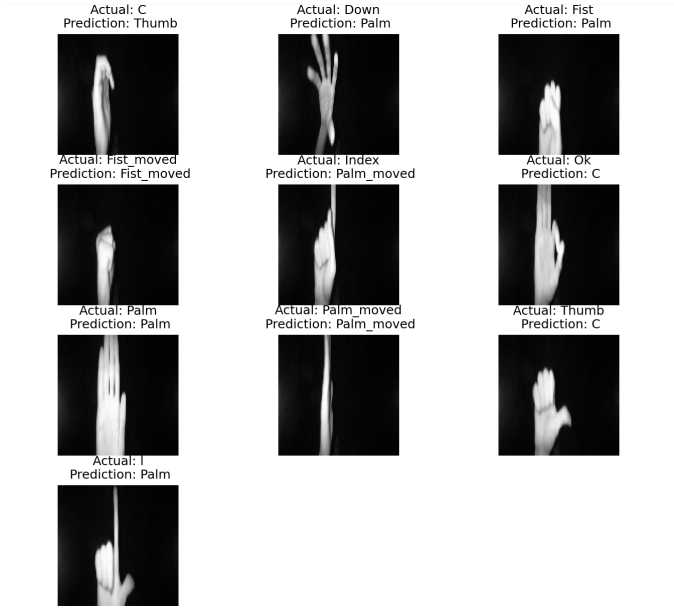


Fig. 3. Sample Result for trained model

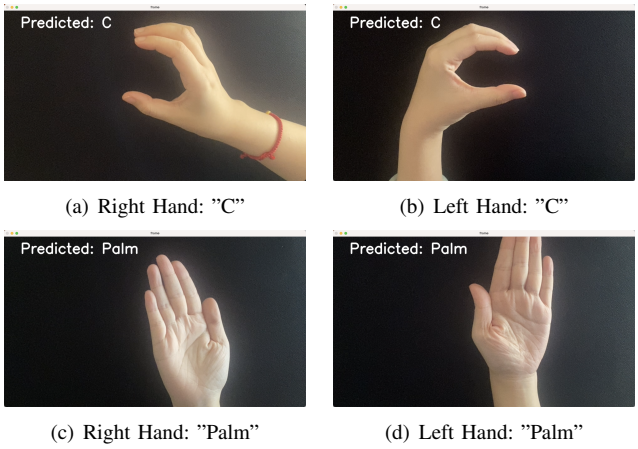
When we try to implement this model to video stream. Only two types of hand gestures can be stably recognized, ”C” and ”Palm” (See Fig. 4). Therefore, we implement these two types of gesture in the presentation controlling, which enables user to move the slides backward and forward.

B. MediaPipe and HandDetector

We use MediaPipe, a Google open-source package, for hand gesture recognition. It provides 21 landmarks on the hand (See Fig. 5).

Based on this, we developed a HandDetector class, which has the following methods:

- **findHands**: This function takes an image frame as input, converts it to RGB (since OpenCV uses BGR by default but MediaPipe requires RGB), and processes the frame to detect hands. If hands are detected, it draws landmarks and connections on the hands within the frame. (See Fig. 6)
- **findPosition**: This method fetches the positional landmarks of a specific hand (default is the first hand, handNo=0) detected in the frame. It calculates the pixel coordinates of each landmark in the image and stores them in lmList. The list of landmarks for the specified hand is returned, with each entry containing the landmark ID and its coordinates.
- **fingersUp**: This method determines which fingers are extended for the first hand detected in the frame. For each of the four fingers except the thumb, the method examines the y-coordinate of the fingertip landmark. For the thumb, it examines its x-coordinate. It returns a list where each entry corresponds to a finger (0 for down, 1 for up).
- **getCenterIndex**: Calculates and returns the center point of all detected landmarks in the last processed frame. This is useful for determining the central point of the hand based on the average positions of all landmarks. If no landmarks are detected, it returns (0, 0).



(a) Right Hand: ”C” (b) Left Hand: ”C”

(c) Right Hand: ”Palm” (d) Left Hand: ”Palm”

Fig. 4. Hand Gestures Recognition using CNN

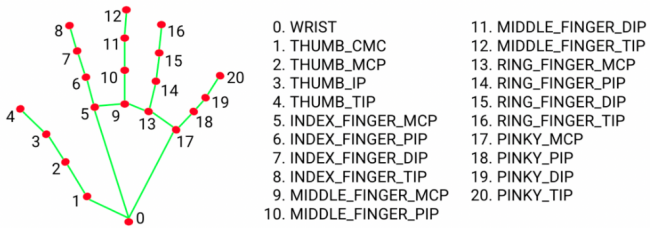


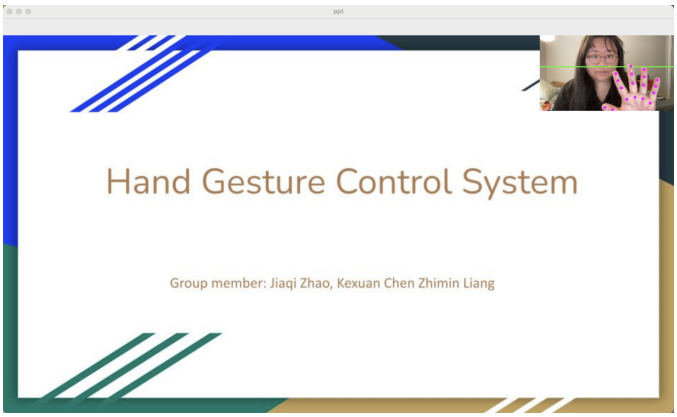
Fig. 5. MediaPipe Hand Landmark Model

After detecting the hand, we use the fingersUp method to determine the posture.

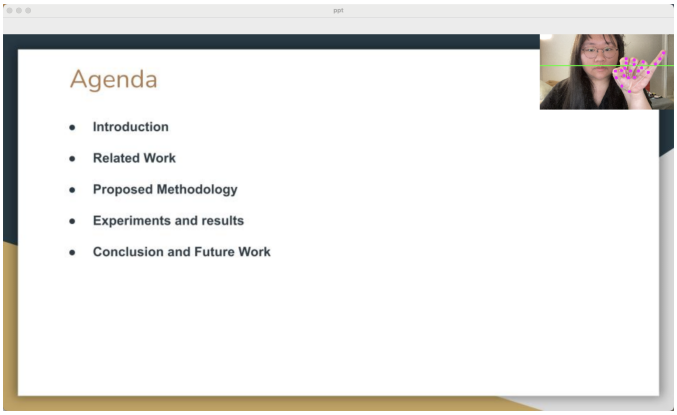
Gesture 1: If only the little finger is raised, display the next slide. (If it is already the last slide, no action is taken.) (See Fig. 6)

Gesture 2: If only the thumb is raised, display the previous slide. (If it is already the first slide, no action is taken.) (See

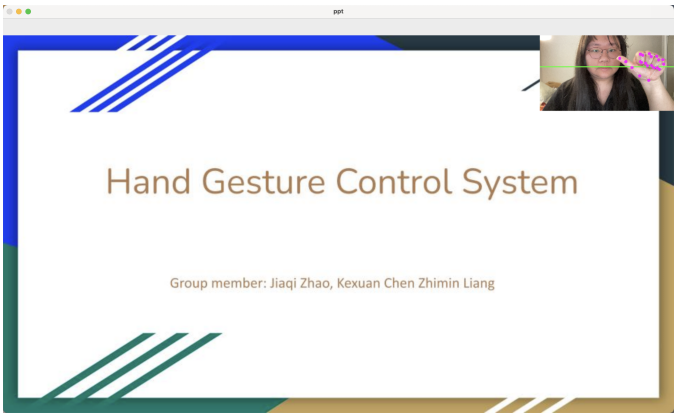
Fig. 6)



(a) Hand Detection



(b) Gesture 1: Show Next Slide



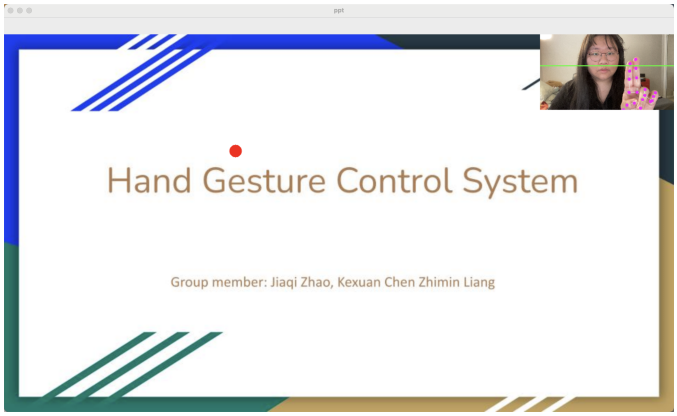
(c) Gesture 2: Show Previous Slide

Fig. 6. Hand Gestures Recognition using MediaPipe

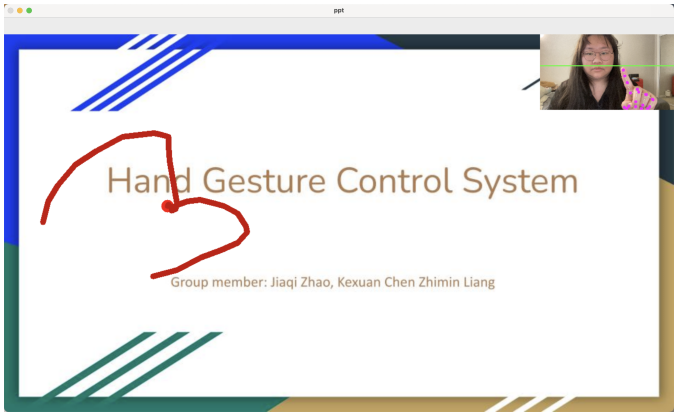
Gesture 3: If the index finger and middle finger are raised, a red dot appears at the corresponding position on the slide, used to highlight. (See Fig. 7)

Gesture 4: If only the index finger is raised, track the path of the fingertip, allowing the user to draw on the slide. (See Fig. 7)

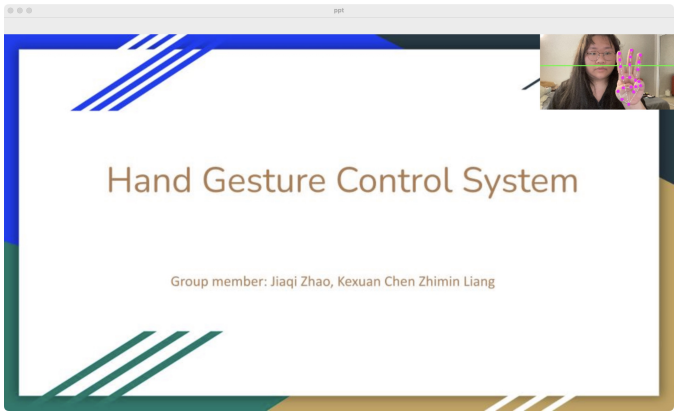
Gesture 5: If the index finger, middle finger, and ring finger are all raised, undo the last drawn pattern. (See Fig. 7)



(a) Gesture 3: Highlight



(b) Gesture 4: Draw Lines



(c) Gesture 5: Clear Drawings

Fig. 7. Hand Gestures Recognition using MediaPipe Cont'd

Note that for the first and second actions, we have set additional triggering conditions. Only when the fingertip's height exceeds the threshold height we have set will the action be detected and the slide switched. Additionally, we have set a 30-frame interval between the two actions to prevent the slides from switching too quickly.

For Gesture 3 and Gesture 4, we utilize the numpy package to adjust the coordinates of the fingertips. This adjustment ensures that movements made in specific areas of the screen—specifically the right half for the x-coordinate and

the central vertical band for the y-coordinate—are expanded to cover the entire screen dimensions. This transformation facilitates more precise and easier interaction with graphical elements on the interface, enhancing the overall drawing and user experience.

V. Conclusion and Future Work

In this study, we developed a hand gesture recognition system using a combination of a convolutional neural network (CNN) and MediaPipe technologies aimed at enhancing presentation control capabilities. The proposed methodology, experimental setup, and results presented in the paper highlight both the potential and the challenges of implementing machine learning-based gesture recognition in practical applications.

A. Key findings

Recognition Accuracy: The CNN model trained on the LeapGesture dataset achieved a recognition accuracy of only 30% for ten types of hand gestures. This level of performance indicates challenges in the model's ability to generalize from training data to real-world application. However, when employing MediaPipe, which utilizes pre-trained models optimized for hand tracking and gesture recognition, the accuracy and robustness improved significantly, demonstrating the utility of combining multiple approaches.

Integration of CNN with MediaPipe: The integration of a custom CNN with MediaPipe showcased an innovative approach to enhance gesture recognition systems. While the CNN provided a basic framework for learning gesture features, MediaPipe offered advanced capabilities for real-time hand tracking and landmark localization, which were crucial for the practical deployment of the system.

Application in Presentation Control: The system successfully implemented basic control gestures, such as moving to the next or previous slide, which were reliably recognized in a live setting. This application illustrates the practical utility of gesture recognition technologies in creating more interactive and intuitive user interfaces.

B. Future Work

Enhancement of Gesture Recognition Accuracy: Investigate methods to improve the accuracy and robustness of gesture recognition algorithms, such as exploring advanced CNN architectures, incorporating additional training data, or implementing ensemble learning techniques.

Gesture Library Expansion: Developing algorithms that can learn and reliably recognize a broader array of gestures would significantly enhance the system's utility across different applications.

User-Centric Design: Future iterations could benefit from a user-centered design approach, involving potential end-users early in the design process to ensure the system meets practical needs and usability standards.

C. Summary

This project has laid a foundational framework for gesture-based interaction systems, with specific application in presentation control. While there are notable challenges to be addressed, the integration of CNN and MediaPipe technologies offers a promising path forward. Future work focused on improving the technology and expanding its capabilities will be crucial in realizing the full potential of gesture-based human-computer interfaces.

References

[1] G. D. Nagalapuram, R. S., V. D., D. D., and D. J. Nazareth, "Controlling Media Player with Hand Gestures using Convolutional Neural Network," in 2021 IEEE Mysore Sub Section International Conference (MysuruCon), Hassan, India, 2021, DOI: 10.1109/MysuruCon52639.2021.9641567.

[2] Prathyakshini and Prathwini, "Hand Gesture Controlled Video Player Application," in 2023 7th International Conference on Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India, 2023, DOI: 10.1109/ICECA58529.2023.10395578.

[3] V. Joseph, A. Talpade, N. Suvarna, and Z. Mendonca, "Visual Gesture Recognition for Text Writing in Air," in 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2018, DOI: 10.1109/ICCONS.2018.8663176.