

STERILE BROWSING OF RADIOLOGY IMAGES

A PROJECT REPORT

Submitted by

BIBINUS DE BOSCO C - 312320205019
HARISH S - 312320205043

in partial fulfillment for the award of the degree

Of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY



St. JOSEPH'S COLLEGE OF ENGINEERING

(An Autonomous Institution)

OMR, Chennai 600 119

ANNA UNIVERSITY: CHENNAI

MARCH 2024

STERILE BROWSING OF RADIOLOGY IMAGES

A PROJECT REPORT

Submitted by

**BIBINUS DE BOSCO C - 312320205019
HARISH S – 312320205043**

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY



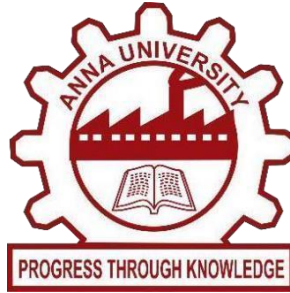
St. JOSEPH'S COLLEGE OF ENGINEERING

(An Autonomous Institution)

OMR, Chennai 600 119

ANNA UNIVERSITY: CHENNAI

MARCH 2024



BONAFIDE CERTIFICATE

Certified that this project report “**STERILE BROWSING OF RADIOLOGY IMAGES**” is the bonafide work of **BIBINUS DE BOSCO C (312320205019)** and **HARISH S (312320205043)** who carried out the project under my supervision.

SIGNATURE

Dr. R. Elavarasan M.E., Ph.D.

Supervisor

Assistant Professor,
Department of Information Technology,
St. Joseph's College of Engineering,
Chennai-600119.

SIGNATURE

Mrs. G. Latha Selvi M.E., (Ph.D).,

Head of the Department

Associate Professor & Head,
Department of Information Technology
St. Joseph's College of Engineering, OMR,
OMR, Chennai-600119.

CERTIFICATE OF EVALUATION

College name : St. Joseph's College of Engineering

Branch & Semester : Information Technology (VIII)

SL. NO	NAME OF THE STUDENT	TITLE OF THE PROJECT	NAME OF THE SUPERVISOR
1	BIBINUS DE BOSCO C (312320205019)	STERILE BROWSING OF RADIOLOGY IMAGES	Dr. R. Elavarasan M.E,Ph.D., ASSISTANT PROFESSOR
2	HARISH S (312320205043)		

The report of the project work submitted by the above students in partial fulfillment for the award of Bachelor of Technology Degree in Information Technology of Anna University was confirmed to be report of the work done by the above students and then evaluated.

Submitted to Project and Viva Examination held on_____.

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

The contentment and elation that accompany the successful completion of any work be incomplete without mentioning the people who made it possible. At the outset we like to express our sincere gratitude in thanking our beloved Chairman, **Dr. B. Babu Manoharan, M.A., M.B.A., Ph.D.**, for his constant guidance and support.

We would like to express our heartfelt thanks to our respected our Managing Director **Mr. B. Shashi Sekar, M.Sc.**, and our Executive Director, **Mrs. B. Jessie Priya, M.Com.**, for providing ample facilities in the institution.

We would like to extend our sincerest and most heartfelt gratitude to our beloved Principal **Dr. Vaddi Seshagiri Rao, M.E., M.B.A., Ph.D., F.I.E.**, for having encouraged us to do our under graduation in Information Technology in this esteemed college.

We express our sincere thanks and most heartfelt sense of gratitude to our Head of the Department **Mrs. G. Lathaselvi, B.E., M.E., (Ph.D.)**, for her guidance and assistance in solving the various intricacies involved in the project.

It is with a deep sense of gratitude that we acknowledge our indebtedness to our supervisor **Dr. R. Elavarasan, M.E., Ph.D.**, a perfectionist for his expert guidance and connoisseur suggestion.

Last but not the least, we thank our family members and friends who have been the greatest source of support to us.

ABSTRACT

Examining patient-specific image data, gathered from computed tomography scans and magnetic resonance imaging scans, during a surgical procedure requires doctor-computer interaction that supports medical imaging manipulation while allowing doctors' hands to remain sterile. However, traditional methods of human-computer interaction fail to provide an efficient method of medical image manipulation supporting users' focus of attention. With the advent of Artificial Intelligence, a new mode of interaction, Gesture based interaction is introduced. Gesture-Based interaction provides an efficient, intuitive, accurate and safe means of interaction without affecting the quality of work. A vision-based hand gesture recognition system has been proposed for surgeons to interact with medical image viewers during an operation. This system interprets the real-time user's hand gestures and translates it to appropriate commands which are then used for manipulation of radiology images.

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE NO
1	INTRODUCTION	1
	1.1 HEURISTIC	2
	1.2 SYSTEM OVERVIEW	4
	1.3 SCOPE OF THE PROJECT	6
2	LITERATURE SURVEY	9
3	SYSTEM ANALYSIS	16
	3.1 EXISTING SYSTEM	16
	3.2 PROPOSED SYSTEM	17
	3.3REQUIREMENTS SPECIFICATIONS	20
	3.3.1HARDWARE REQUIREMENTS	20
	3.3.2 SOFTWARE REQUIREMENTS	23
4	SYSTEM DESIGN	25
	4.1 SYSTEM ARCHITECTURE	25
5	MODULE DESCRIPTION	
	5.1 MODULE DESCRIPTION	30
	5.1.1 GESTURE RECOGNITION	30
	5.1.2 USER INTERFACE MODULE	31
	5.1.3 CALIBRATION AND ADAPTATION	32
	5.1.4 COMMUNICATION MODULE	33
6	CONCLUSION &FUTURE WORKS	37
	6.1 CONCLUSION	37
	6.2FUTURE WORKS	37
	APPENDIX 1	40
	APPENDIX 2	54
	REFERENCES	61

LIST OF ABBREVIATIONS

ABBREVIATION	DEFINITION
UI	User Interface
CT	Computed Tomography
MRI	Magnetic Resonance Imaging
JS	JavaScript
RTR	Real Time Recognition
AR	Augmented Reality
VR	Virtual Reality
EMR	Electronic Medical Record

LIST OF FIGURES

S.NO.	NAME	PAGE NO.
4.1	Architecture of proposed system	25

CHAPTER 1

INTRODUCTION

Humans can recognize body and sign language easily. This is possible due to the combination of vision and synaptic interactions that were formed along brain development. In order to replicate this skill in computers, some problems need to be solved: how to separate objects of interest in images and which image capture technology and classification technique are more appropriate, among others. In this project Gesture based Desktop automation, First the model is trained pre trained on the images of different hand gestures, such as showing numbers with fingers as 1,2,3,4. This model uses the integrated webcam to capture the video frame. The image of the gesture captured in the video frame is compared with the pre-trained model and the gesture is identified. If the gesture predicts is 0 - then images is converted into rectangle, 1 - image is Resized into (200,200), 2 - image is rotated by -45° , 3 - image is blurred, 4 - image is Resized into (400,400), 5 - image is converted into grayscale.

To ensure accurate recognition and efficient processing, various challenges need to be addressed. These include effective segmentation of the hand from the background in the captured images, robust feature extraction to represent the gestures, and optimization of the classification technique to handle real-time performance requirements. Additionally, considerations regarding lighting conditions, hand orientation, and potential occlusions must be accounted for to enhance the system's reliability and usability. As advancements in computer vision and machine learning continue to evolve, the capabilities of gesture-based interaction systems are expected to further improve, enabling seamless integration into various applications and workflows.

1.1 HEURISITCS

Efficiency in Image Manipulation:

The system should allow surgeons to efficiently manipulate medical images during a surgical procedure.

Interaction through gestures should streamline the workflow and not introduce unnecessary delays or complexities.

Intuitiveness of Gesture Commands:

Gesture commands should be intuitive and easily understandable by the surgeons, minimizing the learning curve.

The system should avoid ambiguity in interpreting gestures to prevent potential errors.

Accuracy in Gesture Recognition:

The hand gesture recognition system should accurately interpret and translate real-time user gestures into appropriate commands.

The accuracy of gesture recognition is crucial to ensure precise manipulation of radiology images.

Safety Considerations:

The system must not compromise the safety of the surgical procedure.

Surgeons' hands should remain sterile, and the interaction should not introduce any risk to the patient or disrupt the surgical environment.

Maintaining Surgeon Focus:

The interaction method should support the surgeon's focus of attention on the surgical task and not distract or overwhelm them with the interaction process.

Compatibility with Medical Imaging Data:

The system should effectively handle patient-specific image data from computed tomography (CT) scans and magnetic resonance imaging (MRI) scans. Ensure compatibility with various medical imaging formats and maintain the quality of the images during manipulation.

Real-Time Responsiveness:

The gesture recognition system should operate in real-time to provide immediate feedback to the surgeons. Delay in interpreting gestures may impact the surgeon's ability to make timely decisions during the procedure.

User Feedback and System Transparency:

Provide clear feedback to the surgeon about the interpreted gestures and executed commands.

The system should be transparent, helping users understand how their gestures are being interpreted.

Adaptability and Customization:

The system should be adaptable to different surgical scenarios and allow customization of gestures to cater to individual surgeon preferences.

Account for potential variations in hand movements and gestures among different users.

1.2 SYSTEM OVERVIEW

User Interface

Patient-Specific Image Data:

The system deals with image data acquired from CT scans and MRI scans, which are specific to each patient.

This data serves as the foundation for the surgeon's decision-making during the surgical procedure.

Gesture-Based Interaction:

The core interaction method involves gestures made by the surgeon in real-time. These gestures are captured and interpreted by a vision-based hand gesture recognition system.

Artificial Intelligence Integration:

Artificial Intelligence (AI) is employed to enhance the interaction paradigm.

The AI system interprets and understands the real-time gestures made by the surgeon.

Efficiency and Intuitiveness:

The system aims to provide an efficient and intuitive means of interaction.

Surgeons can manipulate medical images seamlessly without compromising the quality of work or introducing unnecessary complexities.

Sterile Environment Maintenance:

A crucial aspect of the system is the ability to allow doctors to interact with the computer without compromising the sterility of their hands.

Gesture-based interaction serves this purpose by eliminating the need for physical contact with traditional input devices.

Command Translation:

The recognized gestures are translated into appropriate commands that are then used for the manipulation of radiology images.

This translation ensures that the surgeon's intended actions are accurately reflected in the manipulation of the medical images.

Real-Time Gesture Recognition:

The system operates in real-time to provide immediate feedback to the surgeon.

Real-time gesture recognition ensures that there are minimal delays in the interaction, allowing for timely decision-making during the surgical procedure.

Safety Considerations:

The system is designed to prioritize safety during surgical procedures.

Surgeons can interact with the medical image viewer without introducing any risks to the patient or disrupting the surgical environment.

Compatibility and Integration:

The system is designed to be compatible with various medical imaging formats, specifically CT and MRI scans.

It seamlessly integrates into the existing surgical workflow, working alongside other medical equipment and software.

Quality of Work:

The introduction of Gesture-Based Interaction does not compromise the quality of the surgeon's work.

The system ensures that the manipulation of radiology images is accurate and aligned with the surgeon's intentions.

1.3 SCOPE OF THE PROJECT

Computer Vision and AI Integration:

This involves the incorporation of advanced computer vision algorithms and artificial intelligence techniques to enable real-time interpretation of surgeon hand gestures. By leveraging these technologies, the system can accurately analyze and understand the gestures performed by the surgeon, facilitating seamless interaction with medical imaging systems and other relevant equipment.

Gesture Recognition Models:

Developing and training machine learning models is crucial for accurately recognizing and interpreting predefined hand gestures used in medical contexts. These models need to be robust and capable of identifying subtle variations in hand movements to ensure precise manipulation of medical images and other associated tasks.

Integration with Imaging Devices:

To enhance workflow efficiency, the system must be compatible with existing medical imaging devices such as CT and MRI scanners. This ensures that high-quality medical images can be seamlessly captured, displayed, and manipulated using the gesture-based interface, without the need for additional hardware or complex setup procedures.

Medical Image Manipulation:

Enabling surgeons to perform various functions on medical images, including zooming, panning, rotating, and other relevant manipulations using recognized hand gestures.

User Interface Design:

Developing an intuitive and user-friendly interface for surgeons to interact with the system during surgical procedures.

Sterility Measures:

Implementing features that maintain a sterile environment by eliminating the need for direct physical contact with input devices

Emergency Overrides:

In any system where commands are executed based on gesture recognition, it's crucial to have mechanisms in place for immediate cessation of unintended commands or in case of emergency situations. This ensures that if there's a misinterpretation of a gesture or if something unexpected occurs, the system can be quickly halted to prevent any harm or errors.

Fail-Safe Protocols:

Fail-safe protocols are essential to minimize the risk of errors and enhance the overall reliability of the system. These protocols can include redundant systems, error detection and correction mechanisms, and graceful degradation strategies. By implementing fail-safe protocols, the system can continue to operate safely even in the event of component failures or unexpected circumstances.

Adaptability and Learning:

Providing a calibration process allows the system to adapt to individual surgeons' unique gestures and preferences. This calibration process ensures that the system recognizes gestures accurately and responds appropriately to the user's inputs. Additionally, incorporating machine learning techniques enables continuous improvement and adaptation to user behaviors over time, further enhancing the system's accuracy and efficiency.

Integration and Interoperability:

To seamlessly integrate into surgical workflows, the system must be compatible with existing hospital systems and medical equipment. This ensures that the gesture-based interface can communicate effectively with other devices and software used in the operating room, allowing for smooth coordination and workflow efficiency.

Data Transmission:

Establishing reliable communication channels between the various components of the system is essential for ensuring real-time responsiveness. This involves designing robust data transmission protocols that can handle the transfer of information between the gesture recognition module, the control system, and any other connected devices or systems.

Ethical Considerations:

Implementing measures to protect patient privacy and comply with healthcare regulations is paramount in any medical application. This includes ensuring that patient data is securely handled and only accessible to authorized personnel. Additionally, the system must adhere to ethical guidelines governing the use of technology in healthcare settings, prioritizing patient safety and well-being at all times.

CHAPTER 2

LITERATURE SURVEY

[1] “Hand Gestures Recognition Using Radar Sensors for Human-Computer-Interaction Supported by the Bio ad Medical Technology Development Program of the National Research Foundation (NRF)” by Shahzad Ahmed, Karam Dad Kallu, Sarfaraz Ahmed 2021

Human–Computer Interfaces (HCI) deals with the study of interface between humans and computers. The use of radar and other RF sensors to develop HCI based on Hand Gesture Recognition (HGR) has gained increasing attention over the past decade. Today, devices have built-in radars for recognizing and categorizing hand movements. In this article, we present the first ever review related to HGR using radar sensors. We review the available techniques for multi-domain hand gestures data representation for different signal processing and deep-learning-based HGR algorithms. We classify the radars used for HGR as pulsed and continuous-wave radars, and both the hardware and the algorithmic details of each category is presented in detail. Quantitative and qualitative analysis of ongoing trends related to radar-based HCI, and available radar hardware and algorithms is also presented. At the end, developed devices and applications based on gesture-recognition through radar are discussed. Limitations, future aspects and research directions related to this field are also discussed.

[2] “Gesture-controlled image system positioning for minimally invasive interventions Current Directions in Biomedical Engineering Benjamin Fritsch”, Thomas Hoffmann, Andre Mewes, Georg Rose 2021

This work examines how a touchless interaction concept contributes to an efficient, direct, and sterile interaction workflow during CT-guided interventions. Two hand gesture sets were designed specifically under consideration of the clinical workflow and the hardware capabilities. These were used to change the position of an X-Ray tube and detector of a CT scanner without breaking sterility

and are compared regarding usability and performance in a user study with 10 users. The user study revealed that it is possible to change the angle of the gantry within 10 seconds average in an experimental setup. A straight hand gesture showed higher acceptance than a pistol motivated gesture. Furthermore, the sequences were not optimal and confused the users. It turned out that it feels more natural to activate and confirm the system.

[3] “The Potential of Gesture-Based Interaction International Conference”, HCII Springer-Verlag, 2020

This work examines how a touchless interaction concept contributes to an efficient, direct, and sterile interaction workflow during CT-guided interventions. Two hand gesture sets were designed specifically under consideration of the clinical workflow and the hardware capabilities. These were used to change the position of an X-Ray tube and detector of a CT scanner without breaking sterility and are compared regarding usability and performance in a user study with 10 users. The user study revealed that it is possible to change the angle of the gantry within 10 seconds average in an experimental setup. A straight hand gesture showed higher acceptance than a pistol motivated gesture. Furthermore, the sequences were not optimal and confused the users. It turned out that it feels more natural to activate and confirm the system with the same gesture.

[4] “A Preliminary Study of Kinect-Based Real-Time Hand Gesture Interaction Systems for Touchless Visualizations of Hepatic Structures in Surgery Medical Imaging and Information Sciences Jiaqing LIU”, Tomoko Tateyama, 2014

We present a real-time hand gesture interaction system for the touchless visualization of hepatic structure in surgery, for which real-time visualization is important, particularly, during operations, but often faces the challenge of efficiently reviewing a 3D model of a patient's anatomy without requiring the

surgeon to touch an input device, such as a mouse or keyboard, to maintain a sterile field. To solve this problem, we developed a touchless system based on hand gesture interaction. We used a Microsoft Kinect sensor as an input device, which performs hand detection and tracking. Our approach combines three kinds of hand states and their movements to control the visualizations of hepatic structures. Experiments demonstrate that our system can realize real-time hand gesture interactions and visualizations.

[5] “Hand Gesture Recognition System Using Camera. International Journal of Engineering Research and Technology(IJERT)”, Viraj Shinde, Tushar Bacchav , Jitendra Pawar, Mangesh Sanap ,2014

Hand gestures are a form of nonverbal communication that can be used in several fields such as communication between deaf-mute people, robot control, human-computer interaction (HCI), home automation and medical applications. Research papers based on hand gestures have adopted many different techniques, including those based on instrumented sensor technology and computer vision. In other words, the hand sign can be classified under many headings, such as posture and gesture, as well as dynamic and static, or a hybrid of the two. This paper focuses on a review of the literature on hand gesture techniques and introduces their merits and limitations under different circumstances. In addition, it tabulates the performance of these methods, focusing on computer vision techniques that deal with the similarity and difference points, technique of hand segmentation used, classification algorithms and drawbacks, number and types of gestures, dataset used, detection range (distance) and type of camera used. This paper is a thorough general overview of hand gesture methods with a brief discussion of some possible applications.

[6] “Hand-gesture-based sterile interface for the operating room using contextual cues for the navigation of radiological images”. M. Jacob, J. Wachs, R. Packer Published in JAMIA Journal of the American... ,1 June 2013.

A method to improve the navigation and manipulation of radiological images through a sterile hand gesture recognition interface based on attentional contextual cues. Computer vision algorithms were developed to extract intention and attention cues from the surgeon's behavior and combine them with sensory data from a commodity depth camera. The developed interface was tested in a usability experiment to assess the effectiveness of the new interface. An image navigation and manipulation task was 14 performed, and the gesture recognition accuracy, false positives and task completion times were computed to evaluate system performance. Experimental results show that gesture interaction and surgeon behavior analysis can be used to accurately navigate, manipulate and access MRI images, and therefore this modality could replace the use of keyboard and mice-based interfaces.

[7] “Intension, Context and Gesture Recognition for Sterile MRI Navigation in the Operating Room Agency for Healthcare Research and Quality (AHRQ)” by Mithun Jacob ,Christopher Cange , Rebecca Packer, Juan P.Wachs, 2012

Human-Computer Interaction (HCI) devices such as the keyboard and the mouse are among the most contaminated regions in an operating room (OR). This paper proposes a sterile, intuitive HCI to navigate MRI images using freehand gestures. The system incorporates contextual cues and intent of the user to strengthen the gesture recognition process. Experimental results showed that while performing an image navigation task, mean intent recognition accuracy was 98.7% and that the false positive rate of gesture recognition dropped from 20.76% to 2.33% with context integration at similar recognition rates.

[8] “A Gesture-based Tool for Sterile Browsing of Radiology Images Journal of the American Medical Informatics Association (JAMIA)” by Juan P. Wachs, Helman I. Stern, Jon Handler ,2008

The use of doctor-computer interaction devices in the operation room (OR) requires new modalities that support medical imaging manipulation while allowing doctors’ hands to remain sterile, supporting their focus of attention, and providing fast response times. This paper presents “Gestix,” a vision-based hand gesture capture and recognition system that interprets in real-time the user’s gestures for navigation and manipulation of images in an electronic medical record (EMR) database. Navigation and other gestures are translated to commands based on their temporal trajectories, through video capture. “Gestix” was tested during a brain biopsy procedure. In the in vivo experiment, this interface prevented the surgeon’s focus shift and change of location while achieving a rapid intuitive reaction and easy interaction. Data from two usability tests provide insights and implications regarding human-computer interaction based on nonverbal conversational modalities

[9] “Vision Based Hand Gesture Recognition World Academy of Science, Engineering and Technology”, Pragati Garg, Naveen Aggarwal, Sanjeev Sof at 2009

This technology delves into vision-based hand gesture recognition, crucial for human-robot interaction (HRI). It outlines processes such as data acquisition, gesture detection, feature extraction, and classification. Highlighting the importance of natural communication, it addresses challenges in accurately capturing and interpreting gestures. Experimental evaluations assess the efficacy of gesture recognition techniques, shedding light on their practical implications. The study underscores the significance of ongoing research to improve system robustness and reliability. Future directions emphasize advancements in HRI,

aiming to enhance the seamless interaction between humans and robots across diverse applications. By leveraging insights from this analysis, researchers strive to develop more intuitive and efficient gesture recognition systems, paving the way for enhanced human-robot collaboration in various domains, including healthcare, manufacturing, and assistive technology.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

Voice Recognition Systems:

Some medical facilities utilize voice recognition systems to interact with radiology images in sterile environments. These systems may employ microphones to capture both gestures and voice commands, allowing surgeons to manipulate images without physical contact.

Touch Displays:

Touch display technologies are being explored to enable hand interaction with medical images.

Surgeons can navigate through images by making touch the display for input devices.

Gesture Recognition Systems:

Some medical institutions may develop in-house solutions tailored to their specific needs.

These systems might integrate off-the-shelf gesture recognition hardware with custom software for medical image manipulation.

Research Prototypes:

Academic and research institutions often develop prototypes to explore the feasibility of These prototypes may serve as inspiration for your project and provide insights into design considerations and challenges.

Commercial Medical Imaging Software with Gesture Support:

Some commercial medical imaging software solutions may integrate gesture support for certain functions.

These features may not be specifically designed for sterile environments, but they showcase the potential for incorporating gestures into medical image manipulation.

DISADVANTAGES:

Gesture Complexity: Users may struggle with learning and executing complex gestures efficiently, hindering workflow.

Recognition Accuracy: Inconsistent gesture recognition, especially in varied environments, can lead to errors and frustration.

User Fatigue: Prolonged use of gestures may cause physical strain and discomfort, impacting user productivity.

Integration Challenges: Compatibility issues with existing systems and costly technical modifications may impede implementation.

Training Needs: Extensive user training is necessary for proficiency, potentially slowing down adoption and increasing costs.

3.2 PROPOSED SYSTEM

System Architecture:

Integration with Medical Imaging Devices:

The proposed system will seamlessly integrate with various medical imaging devices, such as CT and MRI scanners, establishing communication protocols for data exchange.

Gesture Recognition Module:

A dedicated module will be implemented using computer vision algorithms and machine learning models to accurately recognize and interpret surgeon hand gestures in real-time.

Hardware Components:**Gesture Recognition Cameras:**

Deploy high-resolution cameras strategically placed in the operating room to capture surgeons' hand gestures without obstructing the surgical workflow.

Communication Interface:

Utilize reliable communication interfaces to connect the gesture recognition module with medical imaging devices and other components of the system.

Gesture Recognition Algorithm:**Training Dataset:**

Collect and curate a diverse dataset of surgeon hand gestures for training the machine learning model.

Real-time Recognition:

Implement an algorithm capable of real-time gesture recognition, allowing for seamless and immediate translation of gestures into commands.

Sterile User Interface:**Touchless Display:**

Design a touchless or gesture-controlled display interface to enable surgeons to navigate through radiology images without physically touching any input devices.

Intuitive Controls:

Implement intuitive controls for common image manipulations (e.g., zooming, panning, rotation) using recognized hand gestures.

Sterility Measures:**Contactless Interaction:**

Minimize or eliminate the need for physical contact with the interface, ensuring a sterile environment during surgical procedures.

Calibration and Adaptation:**User Calibration:**

Develop a calibration process allowing individual surgeons to personalize the system to their unique gestures and preferences.

Adaptive Learning:

Implement machine learning techniques for continuous adaptation, refining gesture recognition based on the surgeon's behavior over time.

Emergency Overrides and Fail-Safe Mechanisms:

Emergency Stop Feature: Integrate an emergency stop mechanism to immediately halt any unintended commands or address critical situations.

Usability Testing and Validation:

Conduct extensive usability testing involving surgeons in simulated and real-world scenarios to validate the system's effectiveness.

Performance Metrics:

Evaluate the accuracy, responsiveness, and overall user satisfaction through quantitative and qualitative metrics.

Ethical Considerations and Compliance:

Patient Privacy:

Implement measures to protect patient privacy and adhere to healthcare data security standards.

Regulatory Compliance:

Ensure compliance with relevant healthcare regulations and standards governing medical device development and usage.

ADVANTAGES:

Reduced Contamination Risk: Gesture-based interaction eliminates the need for physical contact with devices, promoting a sterile environment for medical professionals.

Improved Workflow Efficiency: Intuitive gestures streamline image navigation, allowing for quicker analysis and diagnosis.

Enhanced Mobility: Users can interact with radiological images from various angles and distances, improving accessibility and flexibility in clinical settings.

Minimized Equipment Dependency: With gesture control, reliance on additional equipment like keyboards or mice is reduced, simplifying the workspace.

Patient Comfort: Eliminating physical interaction with equipment enhances patient comfort during image reviews, fostering a more positive experience.

DISADVANTAGES:

Learning Curve:

Users may need to invest time and effort into learning how to effectively use gesture-based tools, particularly if they are accustomed to traditional input methods. This learning curve could hinder adoption and potentially slow down.

Limited Gesture Vocabulary:

Gesture-based systems may have a limited vocabulary of recognized gestures, which could restrict the range of interactions possible with radiology images. This limitation may lead to difficulties in expressing nuanced commands or actions.

Physical Fatigue:

Extended use of gesture-based interfaces may lead to physical fatigue or discomfort, particularly if users are required to maintain specific hand positions or perform repetitive gestures for prolonged periods.

3.3 REQUIREMENT SPECIFICATION**3.3.1 Hardware Requirements****Functional Requirements:****Gesture Recognition:**

The system shall accurately recognize and interpret a set of predefined hand gestures relevant to medical image manipulation.

It shall provide real-time feedback on recognized gestures.

The system should support gestures for common actions, including zooming, panning, rotating, and other relevant manipulations.

System Integration:

The system shall seamlessly integrate with various medical imaging devices, such as CT and MRI scanners.

It shall establish communication protocols for data exchange between the gesture recognition module and medical imaging devices.

User Interface:

The system shall feature an intuitive, touchless, or gesture-controlled user interface for surgeons to interact with radiology images.

It should include controls for various image manipulations and intuitive navigation.

Calibration and Adaptation:

The system shall include a user-friendly calibration process to allow individual surgeons to personalize the gesture recognition.

It shall implement machine learning techniques for continuous adaptation, refining gesture recognition based on individual user behavior.

Sterility Measures:

The system shall minimize or eliminate the need for physical contact with the interface, ensuring a sterile environment.

Optionally, voice commands may be integrated as an alternative, reducing reliance on touch-based interaction.

Emergency Overrides and Fail-Safe Mechanisms:

The system shall include an emergency stop feature to immediately halt any unintended commands. It shall implement fail-safe protocols to minimize the impact of errors and enhance system reliability.

Usability Testing and Validation:

The system shall undergo extensive usability testing involving surgeons in simulated and real-world scenarios.

Performance metrics shall include accuracy, responsiveness, and overall user satisfaction.

Non-functional Requirements:

Performance:

The system shall provide real-time gesture recognition with minimal latency.

It shall support simultaneous interactions from multiple surgeons within the operating room.

Security:

The system shall implement security measures to protect patient privacy and comply with healthcare data security standards.

Regulatory Compliance:

The system shall adhere to relevant healthcare regulations and standards governing medical device development and usage.

Reliability:

The system shall have a high level of reliability, minimizing downtime and ensuring consistent performance during surgical procedures.

Scalability:

The system shall be designed to accommodate potential scalability requirements, considering advancements in technology and medical imaging.

Documentation:

The development process and system functionality shall be well-documented, providing comprehensive resources for users and administrators.

Training and Support:

The system shall offer training materials, including tutorials and documentation, to facilitate user familiarity.

Technical support shall be available for troubleshooting and addressing user inquiries.

3.3.2 SOFTWARE REQUIREMENT:

OpenCV (Open-Source Computer Vision Library):

OpenCV is a widely-used open-source library that provides a comprehensive set of tools and algorithms for computer vision tasks. It includes functionalities for image processing, feature detection, object recognition, and gesture recognition, making it indispensable for implementing the core functionalities of the project. With its extensive documentation and active community support, OpenCV enables developers to efficiently develop and deploy computer vision solutions for a wide range of applications.

Python:

Python is chosen as the primary programming language for the project due to its versatility, ease of use, and extensive libraries for machine learning and image processing. It provides a rich ecosystem of tools and frameworks, including TensorFlow, PyTorch, and scikit-learn, which are essential for implementing machine learning models and algorithms for gesture recognition. Python's readability and flexibility make it well-suited for rapid prototyping and experimentation, facilitating the development of complex systems such as the gesture-based tool for sterile browsing of radiological images.

Browser (Chrome recommended):

A modern web browser, preferably Google Chrome, may be required for certain aspects of the project, such as accessing online documentation, tutorials, and web-based user interfaces. Google Chrome is recommended for its compatibility with web technologies and developer tools, which streamline web development and debugging processes. Additionally, web-based interfaces may leverage HTML5, CSS, and JavaScript for interactive visualization and user interaction, making a modern browser an essential tool for testing and deployment.

Text Editor (Visual Studio Code recommended):

A text editor is essential for writing, editing, and managing code and configuration files related to the project. Visual Studio Code (VS Code) is recommended for its lightweight yet powerful features, including syntax highlighting, code completion, and integrated terminal and version control support. VS Code's extensive ecosystem of extensions further enhances productivity by providing additional functionalities for debugging, linting, and code formatting. Its cross-platform compatibility makes it suitable for developers working on different operating systems, ensuring a consistent development experience across teams.

Git and Version Control Systems:

Git, along with version control systems (VCS), such as GitHub or GitLab, is essential for managing project codebase, tracking changes, and collaborating with team members. Version control systems enable developers to maintain a history of project modifications, revert changes if necessary, and coordinate concurrent development efforts. By using Git and VCS, developers can work efficiently, ensure code quality, and streamline collaboration workflows throughout the project's lifecycle.

CHAPTER 4

SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE

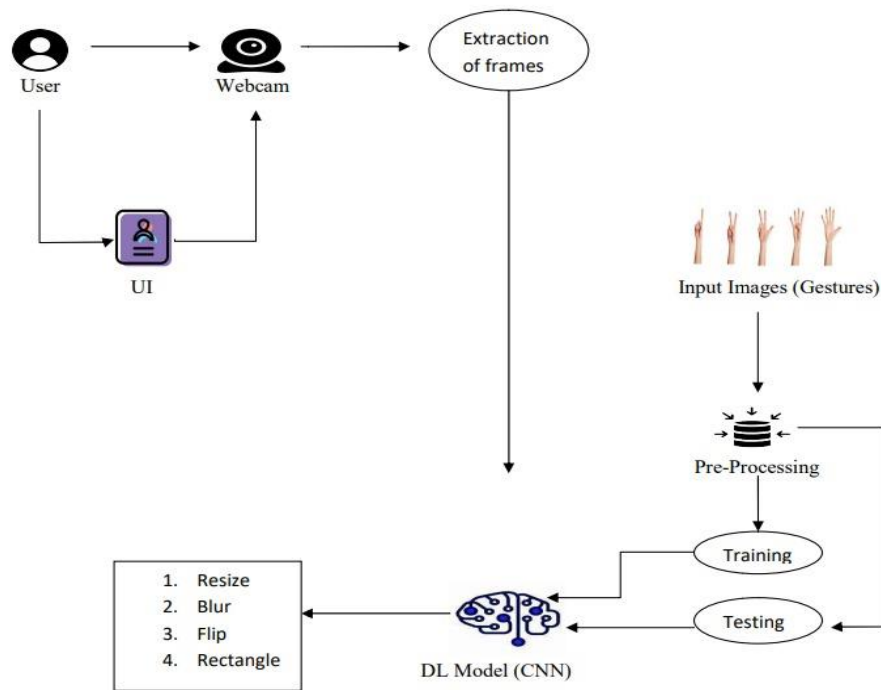


Fig:4.1 Architecture of proposed system

The system architecture for the "Gesture-based Tool for Sterile Browsing of Radiology Images" at (fig:4.1) involves various components working together to enable seamless interaction between the user (surgeon) and medical imaging devices. Here's an outline of the system architecture:

Hardware Components:

Gesture Recognition Hardware:

The Gesture Recognition Hardware includes high-resolution cameras strategically positioned within the operating room to capture hand gestures

accurately. These cameras are selected for their ability to capture detailed images with high clarity and minimal distortion, ensuring precise recognition of hand movements. The placement of cameras is carefully planned to provide comprehensive coverage of the surgical area, enabling efficient tracking of surgeon gestures from various angles. Additionally, depth sensors, such as Time-of-Flight (ToF) cameras, may be incorporated optionally to enhance gesture recognition precision. These sensors measure the distance to objects in the scene, providing depth information that can improve the accuracy of gesture recognition algorithms, especially in complex or dynamic environments.

Processing Unit:

A powerful central processing unit (CPU) or graphics processing unit (GPU) serves as the backbone of the gesture recognition system, responsible for real-time image processing and execution of gesture recognition algorithms. The CPU/GPU is selected based on its computational capabilities and ability to handle the intensive processing demands of gesture recognition tasks. It performs a series of complex operations, including image segmentation, feature extraction, pattern recognition, and gesture classification, with speed and accuracy. Additionally, sufficient RAM is essential to support the computational requirements of image processing and gesture recognition algorithms. Adequate RAM ensures smooth performance and responsiveness of the system, enabling timely execution of commands based on recognized gestures.

Display Interface:

Touchless or gesture-controlled displays for surgeons to visualize and interact with radiology images. The display interface serves as a crucial component of the system, providing surgeons with touchless or gesture-controlled displays to visualize and interact with radiology images during surgical procedures. Touchless displays eliminate the need for physical contact, minimizing the risk

of contamination in sterile environments. Gesture-controlled interfaces allow surgeons to manipulate images using intuitive hand gestures, enhancing workflow efficiency and reducing cognitive load during procedures. These interfaces are designed to be user-friendly and ergonomic, providing clear visualization of medical images while ensuring ease of interaction for surgeons.

Communication Interfaces:

High-speed communication interfaces to connect the gesture recognition module with medical imaging devices. High-speed communication interfaces facilitate seamless connectivity between the gesture recognition module and medical imaging devices. These interfaces ensure rapid transmission of data between the various components of the system, enabling real-time interaction with radiology images. By establishing reliable communication channels, the system can effectively integrate with existing medical imaging equipment, such as CT and MRI scanners, without compromising performance or data integrity.

Emergency Stop Mechanism:

Physical emergency stop button for immediate halting of unintended commands. An emergency stop mechanism is essential for ensuring the safety of surgical procedures by providing a means to immediately halt unintended commands or actions. A physical emergency stop button is incorporated into the system, allowing surgeons or other authorized personnel to quickly stop all ongoing operations in case of emergencies or unexpected situations. This button is strategically located within easy reach and is clearly labeled for rapid identification. When activated, the emergency stop mechanism triggers a shutdown procedure that halts all motorized components and disables any further command execution, helping to prevent potential harm to patients or damage to equipment.

Sterile Casing and Mounting:

Secure mounting system for cameras and components to ensure stability. To maintain a sterile environment in surgical settings, the system components are enclosed in sterile casings made of materials compatible with healthcare standards. These casings are designed to withstand rigorous sterilization processes, such as autoclaving or chemical disinfection, without compromising structural integrity or functionality. Additionally, a secure mounting system is employed to ensure stable positioning of cameras and other components within the operating room. This mounting system is adjustable to accommodate different surgical setups and provides reliable support to prevent accidental displacement during procedures. By incorporating sterile casings and secure mounting solutions, the system minimizes the risk of contamination and maintains the highest standards of hygiene in healthcare environments.

Software Components:**Gesture Recognition Module:**

The Gesture Recognition Module comprises sophisticated computer vision algorithms tailored for real-time recognition of predefined hand gestures. These algorithms leverage techniques such as image segmentation, feature extraction, and pattern recognition to analyze video frames captured by cameras in the surgical environment. By processing these frames, the module can identify specific hand gestures performed by surgeons with high accuracy and speed. Furthermore, machine learning models are integrated within the module to enable continuous adaptation and refinement of gesture recognition capabilities. These models are trained on large datasets of hand gesture samples and are capable of learning and adjusting to variations in gestures over time.

User Interface Software:

The User Interface Software is responsible for designing an intuitive and touchless interface for surgeons to interact with radiology images seamlessly. This software component incorporates user-friendly design principles to create an interface that is easy to navigate and understand. Surgeons can visualize and manipulate radiology images using touchless gestures, eliminating the need for physical contact with input devices and reducing the risk of contamination in sterile environments. Additionally, the software includes a gesture-to-command translation module that interprets recognized gestures and executes corresponding commands for image manipulation. By translating gestures into actionable commands, the User Interface Software enables surgeons to interact with radiology images effortlessly, enhancing workflow efficiency and surgical precision.

Calibration and Adaptation Software:

The Calibration and Adaptation Software provides a user-friendly platform for calibrating and personalizing the gesture recognition system according to individual user preferences. Surgeons can use this software to customize the system to their unique gestures and behavior, ensuring optimal performance during surgical procedures. The software guides users through a calibration process, collecting personalized gesture data and adjusting system parameters accordingly. Moreover, machine learning algorithms embedded within the software enable continuous adaptation based on individual user behavior. These algorithms analyze user interactions with the system and dynamically adjust gesture recognition models to improve accuracy and responsiveness over time. By facilitating personalized calibration and adaptive learning, the Calibration and Adaptation Software enhances the usability and effectiveness of the gesture recognition system in clinical settings.

CHAPTER 5

MODULE DESCRIPTION

5.1 MODULE DESCRIPTION

5.1.1 Gesture Recognition Module:

The Gesture Recognition Module is designed to recognize and interpret predefined hand gestures in real-time, facilitating medical image manipulation. Its components work together seamlessly to ensure accurate and efficient gesture recognition. It processes the captured images to extract relevant features that aid in gesture recognition.

Components:

Image Processing Algorithm:

This component applies computer vision techniques to analyze frames captured by cameras. It processes the captured images to extract relevant features that aid in gesture recognition. Applies computer vision techniques to analyze frames captured by cameras.

Machine Learning Models:

Trained to recognize specific gestures, these models continuously adapt based on user behavior. They play a crucial role in accurately identifying and interpreting hand gestures, even amidst varying environmental conditions or subtle differences in gestures.

Gesture Translation:

Once a gesture is recognized, this component translates it into corresponding commands for image manipulation. It ensures that the gestures performed by surgeons are accurately translated into actions within the system. To maintain a sterile environment in surgical settings, the system components are enclosed in

sterile casings made of materials compatible with healthcare standards. These casings are designed to withstand rigorous sterilization processes, such as autoclaving or chemical disinfection, without compromising structural integrity or functionality. Additionally, a secure mounting system is employed to ensure stable positioning of cameras and other components within the operating room. This mounting system is adjustable to accommodate different surgical setups and provides reliable support to prevent accidental displacement during procedures. By incorporating sterile casings and secure mounting solutions, the system minimizes the risk of contamination and maintains the highest standards of hygiene in healthcare environments.

Workflow:

Captures hand gestures using cameras.

Processes captured images using image processing algorithms.

Applies machine learning models for gesture recognition.

Translates recognized gestures into commands for the User Interface Module.

5.1.2 User Interface Module:

Provide an intuitive and touchless user interface for surgeons to interact with radiology images. The User Interface Module provides an intuitive and touchless interface for surgeons to interact with radiology images during surgical procedures.

Components:

Gesture-to-Command Translator:

Responsible for translating gesture commands received from the Gesture Recognition Module into specific image manipulations. It ensures that the gestures recognized by the system are accurately translated into actions within the user interface.

Workflow:

Displays radiology images on the touchless interface.

Listens for gesture commands from the Gesture Recognition Module.

Translates recognized gestures into appropriate commands for image manipulation.

Enables surgeons to interact seamlessly with radiology images using gestures.

5.1.3 Calibration and Adaptation Module:

The Calibration and Adaptation Module allows individual surgeons to personalize the gesture recognition system for optimal performance. Allow individual surgeons to personalize the gesture recognition system for optimal performance.

Components:**User Calibration Interface:**

Guides surgeons through a calibration process to customize the system to their unique gestures.

Adaptive Learning Algorithm: Utilizes machine learning to continuously adapt to individual surgeon behavior and improve recognition accuracy.

Workflow:

Initiates a user-friendly calibration process.

Captures personalized gesture data during calibration.

Utilizes adaptive learning algorithms to continuously refine gesture recognition based on individual behavior.

Emergency Overrides and Fail-Safe Module:

Ensure the safety of the surgical environment by implementing mechanisms to halt unintended commands. Utilizes machine learning to continuously adapt to individual surgeon behavior and improve recognition accuracy. It enables the system to learn from user interactions and refine its gesture recognition capabilities over time.

Components:

Emergency Stop Button:

A physical button that, when activated, immediately stops all ongoing commands and operations.

Fail-Safe Protocols:

Implemented within the Gesture Recognition and User Interface modules to minimize the impact of errors. Adaptive learning algorithms continuously refine gesture recognition based on individual behavior, ensuring optimal performance during surgical procedures.

Workflow:

Surgeon activates the physical emergency stop button in case of unexpected situations.

Emergency stop software halts all ongoing commands.

Fail-safe protocols prevent potential errors from affecting the system's overall performance.

5.1.4 Communication Module:

Facilitate seamless communication between the Gesture Recognition Module and medical imaging devices. The Communication Module plays a critical role in ensuring seamless interaction between the Gesture Recognition Module and

medical imaging devices within the system. It facilitates efficient data exchange and real-time communication to support the integration of gesture recognition capabilities into the surgical workflow.

Components:

Communication Protocols:

Communication protocols define the rules and formats for data exchange between modules within the system. These protocols establish a standardized framework for transmitting information, ensuring compatibility and consistency across different components. By adhering to established protocols, the Communication Module enables interoperability and smooth communication between the Gesture Recognition Module and medical imaging devices.

Real-Time Data Transmission:

Real-time data transmission capabilities enable responsive and continuous communication between modules. This ensures that information is relayed promptly between the Gesture Recognition Module and other system components, such as the User Interface Module and medical imaging devices. Real-time transmission of data allows for immediate feedback and timely execution of commands, enhancing the overall efficiency and effectiveness of the system.

Workflow:

Establishes communication channels between the Gesture Recognition Module and medical imaging devices.

Enables real-time data transmission to provide immediate feedback to the User Interface Module. The Communication Module establishes communication channels between the Gesture Recognition Module and medical imaging devices, such as CT and MRI scanners.

It ensures that data transmission occurs in real-time, providing immediate feedback to the User Interface Module based on recognized gestures.

By facilitating seamless communication, the Communication Module enables the Gesture Recognition Module to interact effectively with medical imaging devices, supporting the integration of gesture-based interaction into the surgical environment.

Integration Module:

The Integration Module is responsible for enabling seamless integration with medical imaging devices and other system components. It ensures compatibility and interoperability, allowing the Gesture Recognition Module to exchange data with external devices and software seamlessly.

Components:

Compatibility Interfaces:

Compatibility interfaces ensure compatibility with standard medical imaging communication protocols. These interfaces establish the necessary connections and protocols required to communicate with medical imaging devices, ensuring smooth integration and data exchange. Ensure compatibility with standard medical imaging communication protocols.

Integration Points:

Integration points define specific points for communication and data exchange with medical imaging devices. These points serve as interfaces through which the Gesture Recognition Module interacts with external devices, allowing for the exchange of data and commands. Define specific points for communication and data exchange with medical imaging devices.

Workflow:

The Integration Module integrates seamlessly

With various medical imaging devices, using established communication protocols and compatibility interfaces.

It defines integration points for communication and data exchange.

Facilitating the exchange of data between the Gesture Recognition Module and medical imaging devices.

CHAPTER 6

CONCLUSION AND FUTURE WORKS

6.1 CONCLUSION

The Gesture-based Tool for Sterile Browsing of Radiology Images introduces an innovative and efficient means of medical image manipulation in surgical settings. By leveraging touchless and gesture-based interaction, the system enhances sterility, ensures safety, and provides a personalized experience for surgeons. This advancement represents a significant step forward in improving surgical workflows and patient care.

6.2 FUTURE WORKS

Enhanced Gesture Recognition:

Investigate and implement advanced computer vision techniques and machine learning models to further enhance the precision and accuracy of gesture recognition. Explore the integration of 3D gesture recognition for more intricate commands.

Expanded Gesture Vocabulary:

Expand the set of recognized gestures to encompass a broader range of commands, allowing surgeons to perform more nuanced and complex manipulations of radiology images during surgical procedures.

Integration with Emerging Technologies:

Explore integration with emerging technologies, such as augmented reality (AR) and virtual reality (VR), to create a more immersive and interactive surgical environment. This could involve overlaying medical images onto the surgeon's field of view.

APPENDIX 1

SAMPLE CODE

APP.PY

```
from flask import Flask,render_template,request
# Flask-It is our framework which we are going to use to run/serve our
application.
#request-for accessing file which was uploaded by the user on our application.
import operator
import cv2 # opencv library
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import numpy as np

from tensorflow.keras.models import load_model #to load our trained model
import os
from werkzeug.utils import secure_filename

app = Flask(__name__,template_folder="templates") # initializing a flask app
# Loading the model
model=load_model('gesture.h5')

@app.route('/')# route to display the home page
def home():
    return render_template('home.html')#rendering the home page
```

```

@app.route('/intro') # routes to the intro page
def intro():
    return render_template('intro.html')#rendering the intro page


@app.route('/launch',methods=['GET','POST'])
def launch():
    return render_template("launch.html")


@app.route('/perform',methods=['GET', 'POST'])# route to show the predictions
in a web UI

def perform():
    if request.method == 'POST':
        print("inside image")
        file_loader = request.files['image']

        basepath = os.path.dirname(__file__)
        file_path = os.path.join(basepath, 'uploads',
secure_filename(file_loader.filename))
        file_loader.save(file_path)
        print(file_path)
        cap = cv2.VideoCapture(0)
        while True:
            _, frame = cap.read() #capturing the video frame values
            # Simulating mirror image
            frame = cv2.flip(frame, 1)

            # Got this from collect-data.py

```



```

# Coordinates of the ROI
x1 = int(0.5*frame.shape[1])
y1 = 10
x2 = frame.shape[1]-10
y2 = int(0.5*frame.shape[1])
# Drawing the ROI
# The increment/decrement by 1 is to compensate for the bounding box
cv2.rectangle(frame, (x1-1, y1-1), (x2+1, y2+1), (255,0,0) ,1)
# Extracting the ROI
roi = frame[y1:y2, x1:x2]

# Resizing the ROI so it can be fed to the model for prediction
roi = cv2.resize(roi, (64, 64))
roi = cv2.cvtColor(roi, cv2.COLOR_BGR2GRAY)
_, test_image = cv2.threshold(roi, 120, 255, cv2.THRESH_BINARY)
cv2.imshow("test", test_image)
# Batch of 1
result = model.predict(test_image.reshape(1, 64, 64, 1))
prediction = {'ZERO': result[0][0],
              'ONE': result[0][1],
              'TWO': result[0][2],
              'THREE': result[0][3],
              'FOUR': result[0][4],
              'FIVE': result[0][5]}
# Sorting based on top prediction
prediction = sorted(prediction.items(), key=operator.itemgetter(1),
reverse=True)

# Displaying the predictions

```

```
cv2.putText(frame, prediction[0][0], (10, 120),
cv2.FONT_HERSHEY_PLAIN, 1, (0,255,255), 1)
cv2.imshow("Frame", frame)
```

```
#loading an image
```

```
image1=cv2.imread(file_path)
```

```
if prediction[0][0]=='ONE':
```

```
    resized = cv2.resize(image1, (200, 200))
```

```
    cv2.imshow("Fixed Resizing", resized)
```

```
    key=cv2.waitKey(3000)
```

```
    if (key & 0xFF) == ord("1"):
```

```
        cv2.destroyWindow("Fixed Resizing")
```

```
elif prediction[0][0]=='ZERO':
```

```
    cv2.rectangle(image1, (480, 170), (650, 420), (0, 0, 255), 2)
```

```
    cv2.imshow("Rectangle", image1)
```

```
    cv2.waitKey(0)
```

```
    key=cv2.waitKey(3000)
```

```
    if (key & 0xFF) == ord("0"):
```

```
        cv2.destroyWindow("Rectangle")
```

```
elif prediction[0][0]=='TWO':
```

```
    (h, w, d) = image1.shape
```

```
    center = (w // 2, h // 2)
```

```
    M = cv2.getRotationMatrix2D(center, -45, 1.0)
```

```
    rotated = cv2.warpAffine(image1, M, (w, h))
```

```

cv2.imshow("OpenCV Rotation", rotated)
key=cv2.waitKey(3000)
if (key & 0xFF) == ord("2"):
    cv2.destroyWindow("OpenCV Rotation")

elif prediction[0][0]=='THREE':
    blurred = cv2.GaussianBlur(image1, (21, 21), 0)
    cv2.imshow("Blurred", blurred)
    key=cv2.waitKey(3000)
    if (key & 0xFF) == ord("3"):
        cv2.destroyWindow("Blurred")

def perform():
if request.method == 'POST':
    print("inside image")
    file_loader = request.files['image']

    filepath = os.path.dirname(__file__)
    file_path = os.path.join(filepath, 'uploads',
secure_filename(file_loader.filename))
    file_loader.save(file_path)
    print(file_path)
    cap = cv2.VideoCapture(0)
    while True:
        _, frame = cap.read() #capturing the video frame values
        # Simulating mirror image
        frame = cv2.flip(frame, 1)

# Got this from collect-data.py

```

```

# Coordinates of the ROI
x1 = int(0.5*frame.shape[1])
y1 = 10
x2 = frame.shape[1]-10
y2 = int(0.5*frame.shape[1])
# Drawing the ROI
# The increment/decrement by 1 is to compensate for the bounding box
cv2.rectangle(frame, (x1-1, y1-1), (x2+1, y2+1), (255,0,0) ,1)
# Extracting the ROI
roi = frame[y1:y2, x1:x2]

# Resizing the ROI so it can be fed to the model for prediction
roi = cv2.resize(roi, (64, 64))
roi = cv2.cvtColor(roi, cv2.COLOR_BGR2GRAY)
_, test_image = cv2.threshold(roi, 120, 255, cv2.THRESH_BINARY)
cv2.imshow("test", test_image)
# Batch of 1
result = model.predict(test_image.reshape(1, 64, 64, 1))
prediction = {'ZERO': result[0][0],
             'ONE': result[0][1],
             'TWO': result[0][2],
             'THREE': result[0][3],
             'FOUR': result[0][4],
             'FIVE': result[0][5]}
# Sorting based on top prediction
prediction = sorted(prediction.items(), key=operator.itemgetter(1),
reverse=True)

# Displaying the predictions

```

```

cv2.putText(frame, prediction[0][0], (10, 120),
elif prediction[0][0]=='FOUR':

    resized = cv2.resize(image1, (400, 400))
    cv2.imshow("Fixed Resizing", resized)
    key=cv2.waitKey(3000)
    if (key & 0xFF) == ord("4"):
        cv2.destroyWindow("Fixed Resizing")

elif prediction[0][0]=='FIVE':
    gray = cv2.cvtColor(image1, cv2.COLOR_RGB2GRAY)
    cv2.imshow("OpenCV Gray Scale", gray)
    key=cv2.waitKey(3000)
    if (key & 0xFF) == ord("5"):
        cv2.destroyWindow("OpenCV Gray Scale")

else:
    continue

interrupt = cv2.waitKey(10)
if interrupt & 0xFF == 27: # esc key
    break

cap.release()
cv2.destroyAllWindows()
return render_template("home.html")

```

```
if __name__ == "__main__":  
    # running the app  
    app.run(debug=True)
```

Model.js

```
$(document).ready(function () {  
    // Init  
    $('.image-section').hide();  
    $('.loader').hide();  
    $('#result').hide();  
  
    // Upload Preview  
    function readURL(input) {  
        if (input.files && input.files[0]) {  
            var reader = new FileReader();  
  
            reader.onload = function (e) {  
                $('#imagePreview').css('background-image', 'url(' + e.target.result +  
''));  
                $('#imagePreview').hide();  
                $('#imagePreview').fadeIn(650);  
            }  
            reader.readAsDataURL(input.files[0]);  
        }  
    }  
    $("#imageUpload").change(function () {  
        $('.image-section').show();  
        $('#btn-predict').show();  
    });  
});
```

```

    $('#result').text("");
    $('#result').hide();
    readURL(this);
});

// Predict
$('#btn-predict').click(function () {
    var form_data = new FormData($('#upload-file')[0]);

    // Show loading animation
    $(this).hide();
    $('.loader').show();

    // Make prediction by calling api /predict
    $.ajax({
        type: 'POST',
        url: '/predict',
        data: form_data,
        contentType: false,
        cache: false,
        processData: false,
        async: true,
        success: function (data) {
            // Get and display the result
            $('.loader').hide();
            $('#result').fadeIn(600);
            $('#result').html(data);
            console.log('Success!');
        },
    },

```

```
});  
});
```

MODEL BUILDING.PY

```
"""Import the libraries"""
```

```
import numpy as np  
import os  
import tensorflow as tf  
from tensorflow.keras.models import Sequential  
from tensorflow.keras.layers import Dense, Flatten, Dropout  
from tensorflow.keras.layers import Convolution2D, MaxPooling2D  
from tensorflow.keras.preprocessing.image import ImageDataGenerator  
from tensorflow.keras.preprocessing import image  
from tensorflow.keras.models import load_model  
import matplotlib.pyplot as plt  
import matplotlib.image as mpimg
```

```
"""Augment the data"""
```

```
train = ImageDataGenerator(rescale =  
1./255, shear_range=0.2, zoom_range=0.2, horizontal_flip=True)  
test = ImageDataGenerator(rescale = 1./255)
```

```
"""Loading and augmentation of given data"""
```

```
A_train = train.flow_from_directory(r'D:\College\7th semester\ibm\Final  
Deliverables\Project\Dataset\train', target_size=(64,64),
```



```

color_mode='grayscale',batch_size=3, class_mode='categorical')
A_test = test.flow_from_directory(r'D:\College\7th semester\ibm\Final
Deliverables\Project\Dataset\test', target_size=(64,64),
color_mode='grayscale',batch_size=3, class_mode='categorical')

print(A_train.class_indices)

print(A_test.class_indices)

"""Import Keras library"""

model = Sequential()

"""Add 1st Convolution Layer and Pooling layer"""

model.add(Convolution2D(32,(3,3),input_shape=(64,64,1),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))

"""Add 2nd Convolution Layer and Pooling layer"""

model.add(Convolution2D(32,(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))

"""Add Flatten layer"""

model.add(Flatten())

"""Add dense layers"""

```

```

model.add(Dense(units=512,activation='relu'))

model.add(Dense(units=6,activation='softmax'))

print(model.summary())

"""Compile the model"""

model.compile(metrics=['accuracy'],loss='categorical_crossentropy',optimizer='
adam')

"""Train the model"""

model.fit(A_train,steps_per_epoch =
594/3,epochs=25,validation_data=A_test,validation_steps=len(A_test))

"""Save the model"""

model.save('gesture.h5')

json_model = model.to_json()
with open("model-gesture.json","w") as json_file:
    json_file.write(json_model)
        processData: false,
        async: true,
        success: function (data) {
            // Get and display the result
            $('#loader').hide();
            $('#result').fadeIn(600);

```

```

        $('#result').html(data);
        console.log('Success!');
    },

    """"Test the model""""

    test_model = load_model('gesture.h5')
    img_path=r"D:\College\7th semester\ibm\Final Deliverables\Project\Model
    Building\test_image.jpg"
    model.add(Dense(units=512,activation='relu'))

    model.add(Dense(units=6,activation='softmax'))
    }
    $("#imageUpload").change(function () {
        $('.image-section').show();
        $('#btn-predict').show();
        $('#result').text("");
        $('#result').hide();
        readURL(this);
    });

    // Predict
    $('#btn-predict').click(function () {
        var form_data = new FormData($('#upload-file')[0]);

        // Show loading animation
        elif prediction[0][0]=='FOUR':

        resized = cv2.resize(image1, (400, 400))

```

```

cv2.imshow("Fixed Resizing", resized)
key=cv2.waitKey(3000)
if (key & 0xFF) == ord("4"):
    cv2.destroyWindow("Fixed Resizing")

elif prediction[0][0]=='FIVE':
    gray = cv2.cvtColor(image1, cv2.COLOR_RGB2GRAY)
    cv2.imshow("OpenCV Gray Scale", gray)
    key=cv2.waitKey(3000)
    if (key & 0xFF) == ord("5"):
        cv2.destroyWindow("OpenCV Gray Scale")

#loading an image
image1=cv2.imread(file_path)
if prediction[0][0]=='ONE':

    resized = cv2.resize(image1, (200, 200))
    cv2.imshow("Fixed Resizing", resized)
    key=cv2.waitKey(3000)

    if (key & 0xFF) == ord("1"):
        cv2.destroyWindow("Fixed Resizing")

elif prediction[0][0]=='ZERO':

    cv2.rectangle(image1, (480, 170), (650, 420), (0, 0, 255), 2)
    cv2.imshow("Rectangle", image1)
    cv2.waitKey(0)
    key=cv2.waitKey(3000)

```

```
if (key & 0xFF) == ord("0"):
    cv2.destroyAllWindows()
```

```
elif prediction[0][0]=='TWO':
    (h, w, d) = image1.shape
    center = (w // 2, h // 2)
    M = cv2.getRotationMatrix2D(center, -45, 1.0)
    rotated = cv2.warpAffine(image1, M, (w, h))
    cv2.imshow("OpenCV Rotation", rotated)
    key=cv2.waitKey(3000)
    if (key & 0xFF) == ord("2"):
        cv2.destroyAllWindows()
```

```
elif prediction[0][0]=='THREE':
    blurred = cv2.GaussianBlur(image1, (21, 21), 0)
    cv2.imshow("Blurred", blurred)
    key=cv2.waitKey(3000)
    if (key & 0xFF) == ord("3"):
        cv2.destroyAllWindows()
```

```
$(this).hide();
```

```
$('.loader').show();
```

```
// Make prediction by calling api /predict
```

```
$.ajax({
    type: 'POST',
    url: '/predict',
    data: form_data,
```

```

contentType: false,
cache: false,
processData: false,
async: true,
success: function (data) {
    // Get and display the result
    $('#loader').hide();
    $('#result').fadeIn(600);
    $('#result').html(data);
    console.log('Success!');
}

```

```
print(model.summary())
```

```
"""Compile the model"""
```

```
model.compile(metrics=['accuracy'],loss='categorical_crossentropy',optimizer='adam')
```

```
img = mpimg.imread(img_path)
```

```
imgplot = plt.imshow(img)
```

```
plt.show()
```

```
imgload =
```

```
image.load_img(img_path,color_mode='grayscale',target_size=(64,64))
```

```
res = image.img_to_array(imgload)
```

```
print(res.shape)
```

```
A_train = train.flow_from_directory(r'D:\College\7th semester\ibm\Final
```

```

Deliverables\Project\Dataset\train', target_size=(64,64),
color_mode='grayscale',batch_size=3, class_mode='categorical')
A_test = test.flow_from_directory(r'D:\College\7th semester\ibm\Final
Deliverables\Project\Dataset\test', target_size=(64,64),
color_mode='grayscale',batch_size=3, class_mode='categorical')

print(type(res))
res = np.expand_dims(res,axis=0)
print(res.shape)

"""Predict the result"""

pred_res = np.argmax(test_model.predict(res),axis=-1)
print(pred_res)

index = ['0','1','2','3','4','5']
final_res = str(index[pred_res[0]])
print(final_res)

```


APPENDIX 2

SCREENSHOTS

The image is a composite of two screenshots. The top screenshot shows a Google Colab notebook titled 'Model_Building.ipynb'. The notebook interface includes a file explorer on the left showing a directory structure with folders like 'content', 'drive', 'MyDrive', 'Camera', 'Certificates', 'Classroom', 'Colab Notebooks', 'DATASET', 'TEST', 'TRAIN', and 'Read Me'. The main code area contains three sections: 'Compile the model' with a single line of code, 'Train the model' with a `model.fit` call and its output showing progress and metrics (ETA: 0s, loss: 0.9489, accuracy: 0.6684), and 'Save the model' with code to save the model as 'gesture.h5' and export it as a JSON file. A 'Test the model' section is partially visible at the bottom. The bottom screenshot shows a video player with the title 'Vision-Based Hand-Gesture Applications'. The video frame shows a man in a suit. To the left of the video is a large yellow hand emoji. The video player has a progress bar at 08:32 and a 'vimeo' logo.

HAND GESTURE RECOGNITION SYSTEM

[Home](#) [Introduction](#) [Launch](#)



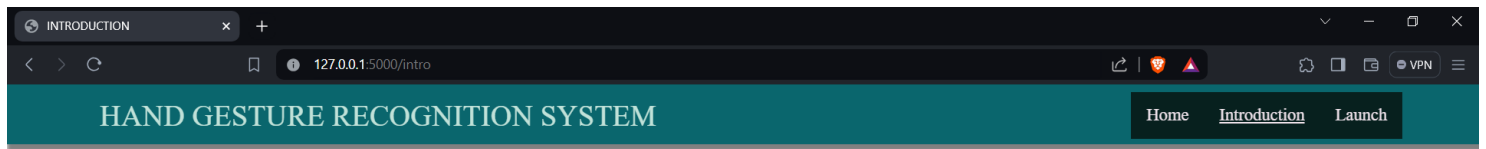
Vision-Based Hand-Gesture Applications

CACM

08:32

vimeo

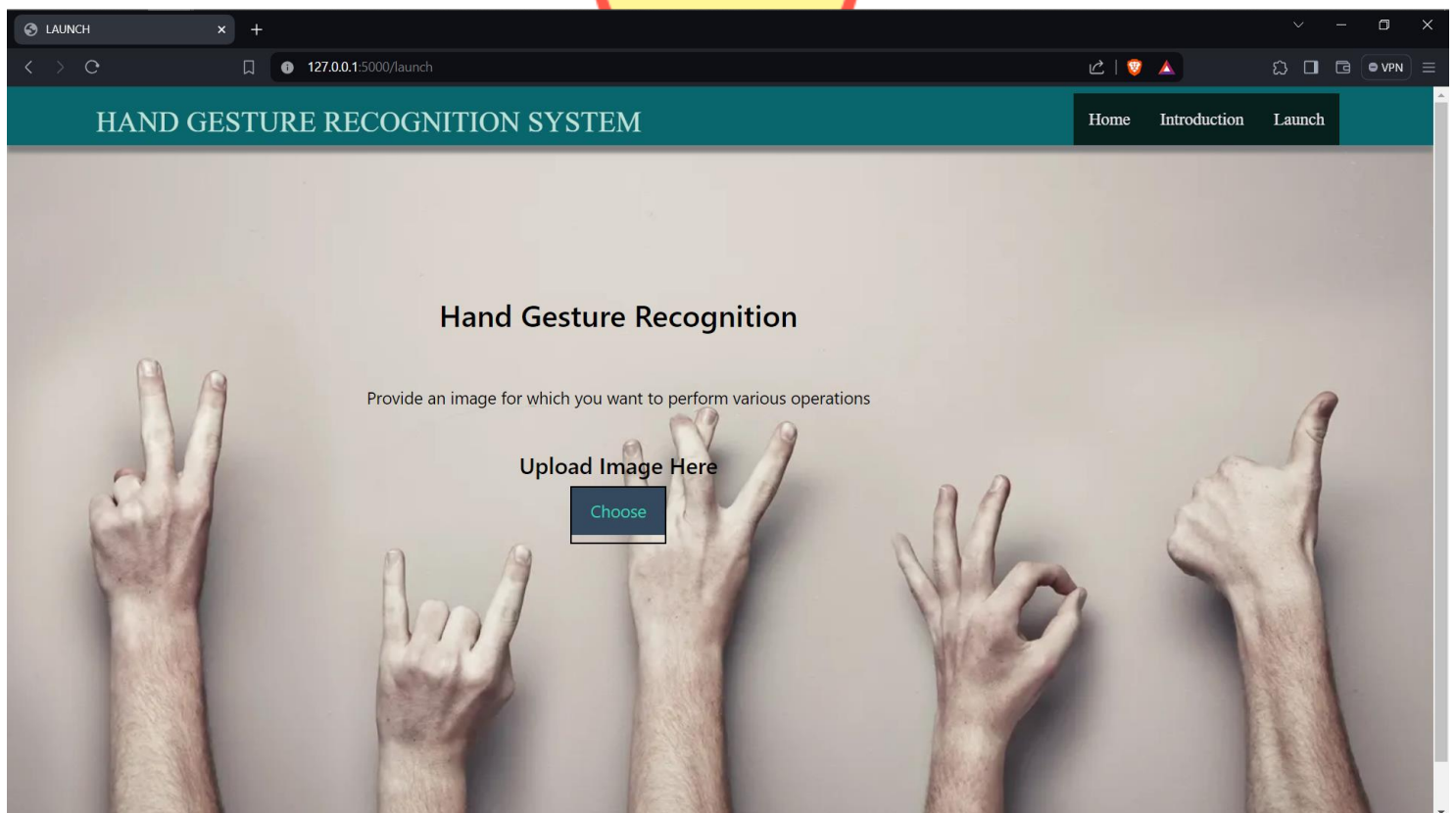
HAND GESTURE RECOGNITION OF RADIOLOGY IMAGES THROUGH STERILE BROWSING

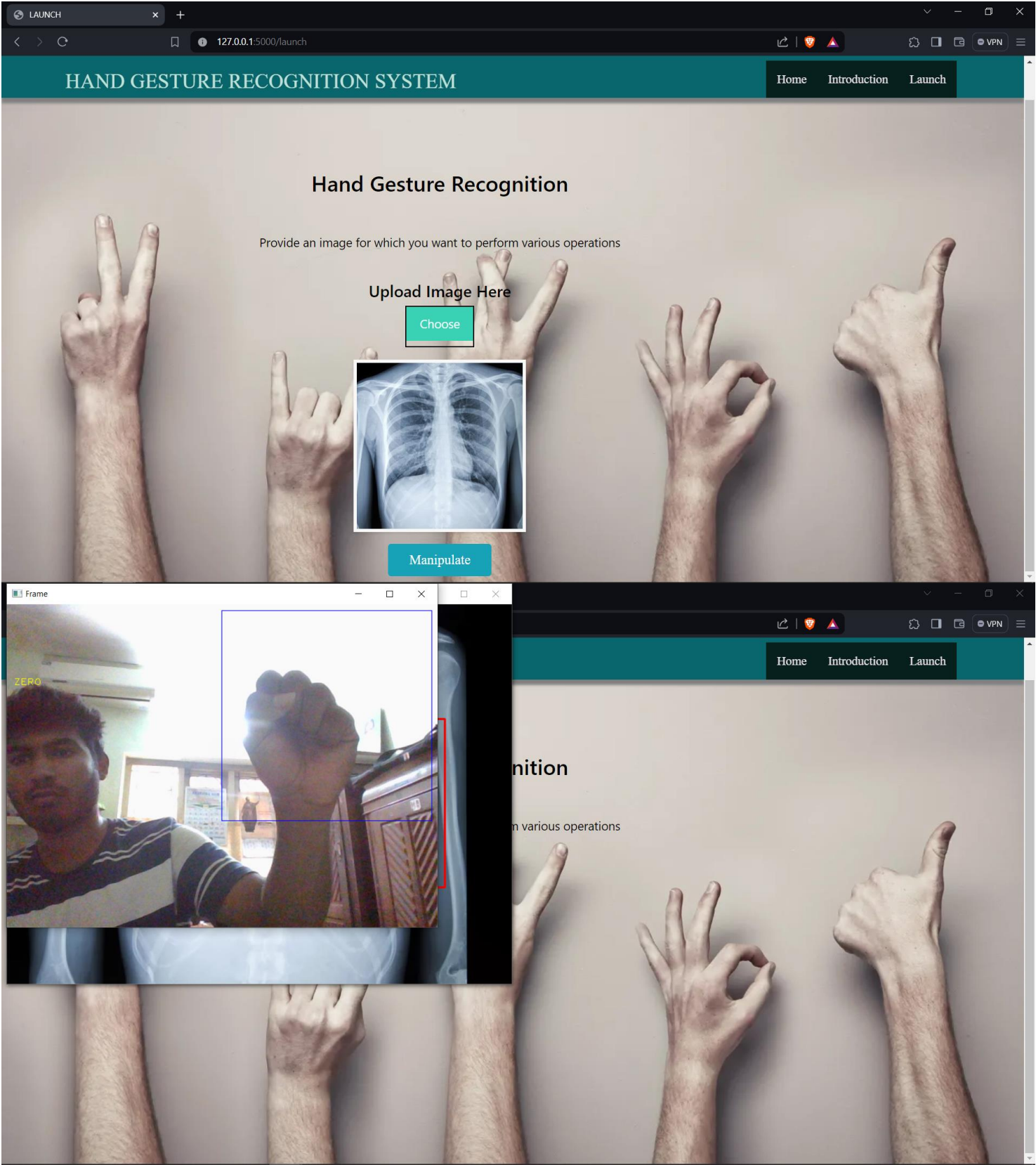


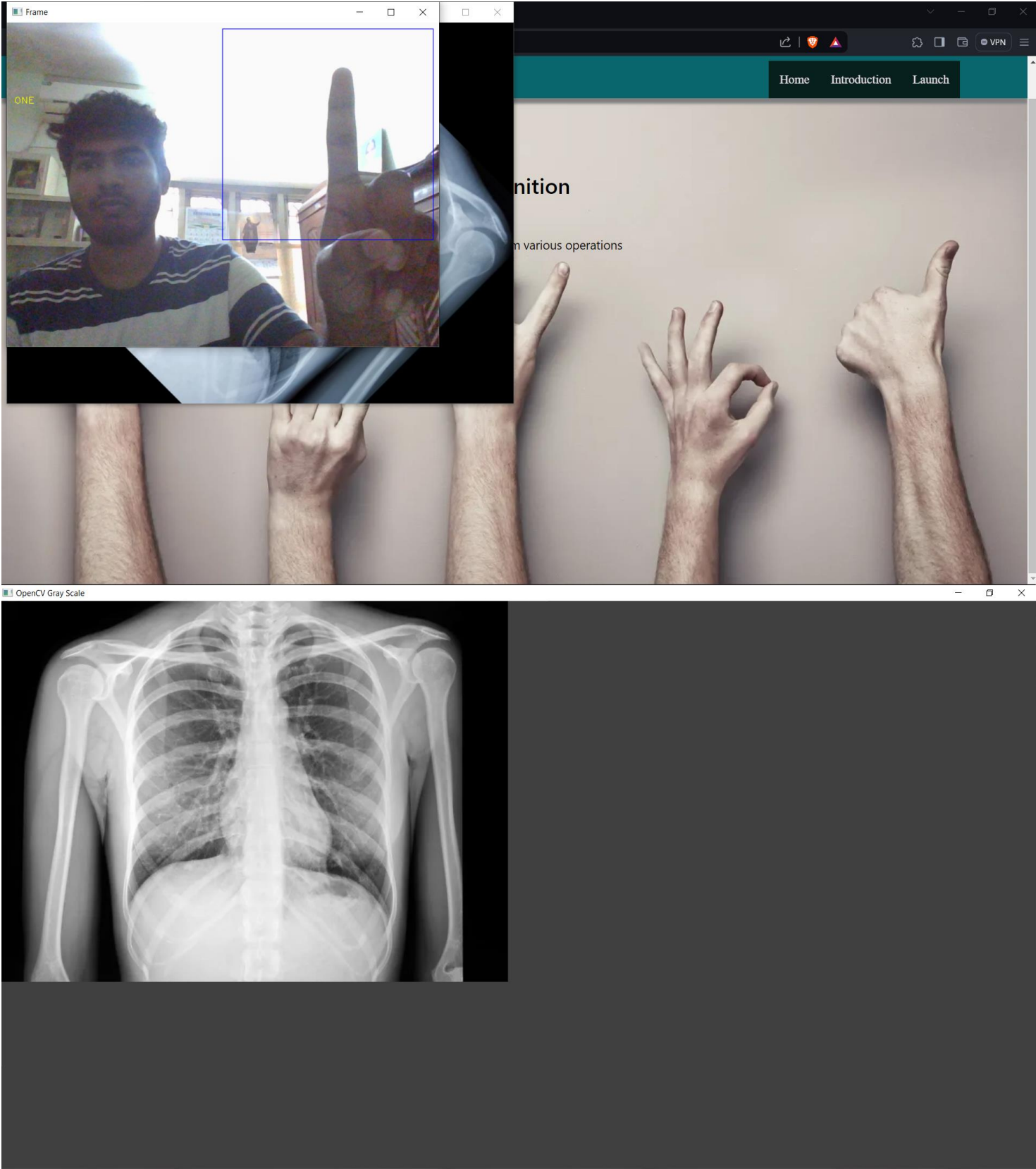
Hand Gesture recognition system provides us an innovative, natural, user friendly way of interaction with the computer which is more familiar to the human beings. In our project, the hand region is extracted from the background by using Region of interest. Then, we will be predicting the labels based on the CNN trained model weights of hand gestures using that predicted labels we apply if conditions to control some of the actions like reshaping , blur, flip of the given image.

The operations included in this project/system are as follows

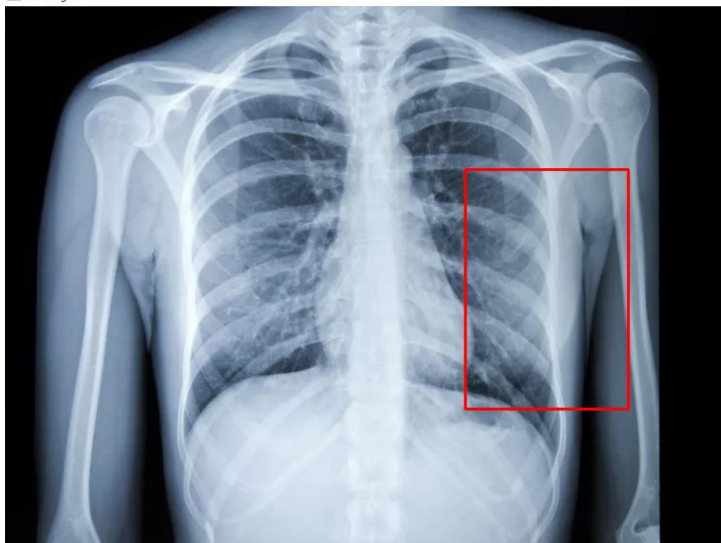
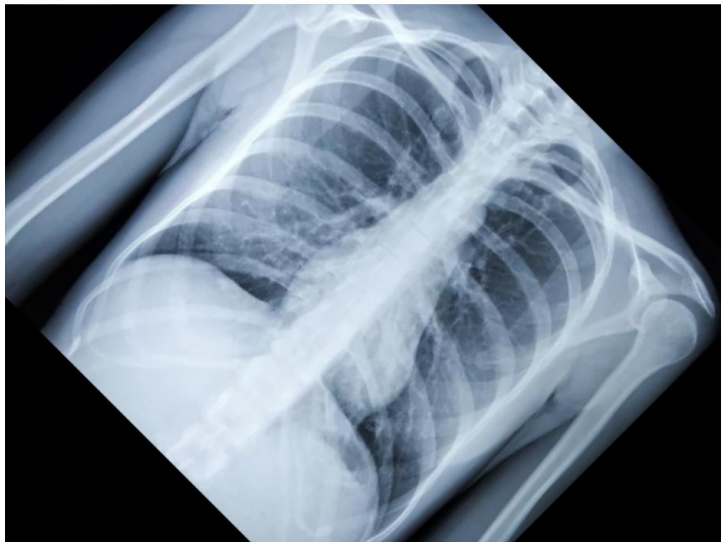
Hand Gesture	Function
1	Resize image into 200x200
2	45 Degree Right
3	Blur the image
4	Resize image into 400x400
5	Convert image into grayscale











REFERENCES

- [1]”Hand Gestures Recognition Using Radar Sensors for Human-Computer-Interaction Supported by the Bio ad Medical Technology Development Program of the National Research Foundation (NRF)” by Shahzad Ahmed, Karam Dad Kallu, Sarfaraz Ahmed 2021.
- [2]”Gesture-controlled image system positioning for minimally invasive interventions Current Directions in Biomedical Engineering “Benjamin Fritsch, Thomas Hoffmann, Andre Mewes, Georg Rose 2021
- [3]”The Potential of Gesture-Based Interaction International Conference”, HCII Springer-Verlag 2020
- [4]” A Preliminary Study of Kinect-Based Real-Time Hand Gesture Interaction Systems for Touchless Visualizations of Hepatic Structures in Surgery Medical Imaging and Information Sciences” Jiaqing LIU, Tomoko Tateyama 2014
- [5]”Hand Gesture Recognition System Using Camera. International Journal of Engineering Research and Technology(IJERT) “Viraj Shinde, Tushar Bacchav , Jitendra Pawar, Mangesh Sanap 2014.
- [6]”Hand-gesture-based sterile interface for the operating room using contextual cues for the navigation of radiological images.” M. Jacob, J. Wachs, R. Packer Published in JAMIA Journal of the American... 1 June 2013.
- [7]”Intension, Context and Gesture Recognition for Sterile MRI Navigation in the Operating Room Agency for Healthcare Research and Quality (AHRQ) “by Mithun Jacob ,Christopher Cange , Rebecca Packer, Juan P.Wachs 2012
- [8]”Vision Based Hand Gesture Recognition World Academy of Science, Engineering and Technology” Pragati Garg, Naveen Aggarwal, Sanjeev Sof at 2009
- [9]”A Gesture-based Tool for Sterile Browsing of Radiology Images Journal of the American Medical Informatics Association (JAMIA)” by Juan P. Wachs, Helman I. Stern, Jon Handler 2008.