Al Imam Mohammad Ibn Saud Islamic University

College of Computer and Information Sciences

Information Technology Department

Guiding glasses for blind people using computer vision and ChatGPT

By:

Name	**ID**	**Department**
Mohammed Fahad Al-Shammari	441015937	IT
Faisal Fahad Al-Qahtani	441014854	IT
Abdulalkreem Abdulmajeed Al-Otibai	440026009	IT

Supervisor:
Dr. Abdullah Hamad Almoqbil

Project Submitted in Partial Fulfillment for the Degree of B.Sc. in "Information Technology"

Winter 2024

Al Imam Mohammad Ibn Saud Islamic University

College of Computer and Information Sciences

Information Technology Department

Guiding glasses for blind people using computer vision and ChatGPT

By:

Name	**ID**	**Department**
Mohammed Fahad Al-Shammari	441015937	IT
Faisal Fahad Al-Qahtani	441014854	IT
Abdulalkreem Abdulmajeed Al-Otibai	440026009	IT

Supervisor Name	Abdullah Hamad Almoqbil
Date	2/1/2024
Signature	Almoqbil

Declaration

We are Mohammed Fahad Al-Shammari , Fisal Fahad Al-Qahtani and Abdulalkreem Abdulmajeed Al-Otibi being members of final year project group number 173, declare that this report contains only work completed by members of our group except for information obtained in a legitimate way from literature, company or university sources. All information from these other sources has been duly referenced and acknowledged in accordance with the University Policy on Plagiarism.

Furthermore, we declare that in completing the project, the individual group members had the following responsibilities and contributed in the following proportions to the final outcomes of the project:

Student ID	Responsibilities[1]	Contribution[2] %	Signature
441015937	Model training, integration and Team lead	40%	Mohammed
440026009	Objects' distance and direction detection	30%	Abdulalkreem
441014854	Raspberry and ChatGPT	30%	Faisal

[1] Write down your responsibilities in the project
[2] Must add to 100%

تعهد بحفظ حقوق الملكية الفكرية للكلية

أتعهد بعدم المشاركة في الفعاليات أو المبادرات أو المسابقات ذات العلاقة دون أخذ موافقة خطية مسبقة من الكلية، وأقر بمعرفتي أنني إذا خالفت هذا التعهد ستتم محاسبتي وفق اللوائح والأنظمة.

اسم الطالب	الرقم الجامعي	التوقيع
1. محمد الشمري	441015937	
1. عبدالكريم العتيبي	440026009	
2. فيصل القحطاني	441014854	

Acknowledgment

First and foremost, We would like to express my deepest gratitude to Almighty Allah for His bountiful blessings and for granting us the ability to complete this project successfully.

We would like to acknowledge and express my profound gratitude to our esteemed Supervisor, Mr. Abdullah Almoqbil. His wisdom, guidance, and unwavering support have been the reason for the successful completion of this project. His invaluable expertise and mentorship have been essential in shaping the project's direction and ensuring its successful completion.

We would also like to extend my heartfelt appreciation to our families for their unwavering support, patience, and encouragement throughout this project. Your unwavering faith in us and your willingness to lend a helping hand whenever needed have been crucial throughout this journey.

We would like to acknowledge the invaluable contribution of our IT department. Over the past four years, their teachings have been Significant in our learning process. Their support and guidance have been invaluable in shaping our understanding and skills.

Lastly, we would like to thank our dear friend Majid Alanazi for his invaluable consultations and for testing our system. His expertise and insights have been Crucial in improving the quality of our work.

In conclusion, this project would not have been possible without the unwavering support, guidance, and contribution of all these individuals and groups. We are deeply grateful for their invaluable assistance and look forward to the opportunities to collaborate with them in the future.

Abstract

The project, "Guiding Glasses for Blind People Using Computer Vision and ChatGPT," addresses the challenges faced by visually impaired individuals in achieving independence and mobility. By leveraging computer vision technology and advanced AI systems, this initiative aims to create innovative assistive glasses. These glasses serve as a visual guide, providing real-time object detection, distance measurements, and text recognition without the need for a traditional cane. Additionally, integration with ChatGPT enables users to access information on-demand. This comprehensive solution represents a significant advancement in enhancing the daily lives of the visually impaired, offering a versatile tool that improves spatial awareness and object recognition. The project's innovative aspect lies in seamlessly combining multiple technologies into a wearable device, ultimately enhancing the quality of life and independence of visually impaired individuals. The Agile methodology is employed for efficient project management, ensuring a seamless integration of computer vision and ChatGPT technologies. The project's success relies on the robustness of existing libraries and frameworks for computer vision, such as OpenCV and TensorFlow, as well as the seamless integration of ChatGPT's API. In terms of hardware, the prototype requires components like Raspberry Pi 4 Model B, a microphone, a speaker, and a Camera for optimal performance.

Keywords:

AI , Model, Speech Recognition, object detection, Artificial Intelligence, Computer Vision, Deep Learning, Machine Learning, Supervised Learning

Abstract (in Arabic)

يعالج مشروع "النظارات التوجيهية للمكفوفين باستخدام الرؤية الحاسوبية و شات جي بي تي" التحديات التي يواجهها الأفراد ضعاف البصر في تحقيق الاستقلال والتنقل. ومن خلال الاستفادة من تكنولوجيا الرؤية الحاسوبية وأنظمة الذكاء الاصطناعي المتقدمة، تهدف هذه المبادرة إلى إنشاء نظارات مساعدة مبتكرة. تعمل هذه النظارات بمثابة دليل مرئي، مما يوفر اكتشافًا فوريًا للأشياء وقياسات المسافة والتعرف على النص دون الحاجة إلى عصا تقليدية. بالإضافة إلى ذلك، يتيح الارتباط مع شات جي بي تي للمستخدمين الوصول إلى المعلومات عند الطلب. يمثل هذا الحل الشامل تقدمًا كبيرًا في تحسين الحياة اليومية لضعاف البصر، حيث يقدم أداة متعددة الاستخدامات تعمل على تحسين الوعي المكاني والتعرف على الأشياء. ويكمن الجانب المبتكر للمشروع في الجمع بسلاسة بين تقنيات متعددة في جهاز يمكن ارتداؤه، مما يؤدي في نهاية المطاف إلى تحسين نوعية الحياة واستقلالية الأفراد ضعاف البصر.

سيتم استخدام منهجية اجايل لإدارة المشروع بكفاءة، مما يضمن التكامل السلس بين رؤية الكمبيوتر وتقنيات نشات جي بي تي.

يعتمد نجاح المشروع على قوة المكتبات والأطر الموجودة لرؤية الكمبيوتر، مثل TensorFlowو OpenCV بالإضافة إلى التكامل السلس لواجهة برمجة تطبيقات تشات جي بي تي. فيما يتعلق بالأجهزة، يتطلب النموذج الأولي مكونات مثل Raspberry Pi 4 Model B و كاميرا ومايكروفون وسبيكر لتحقيق الأداء الأمثل.

List of Abbreviations

AI	Artificial Intelligence
CV	Computer Vision
DL	Deep Learning
ML	Machine Learning

Table of Contents

List of Figures

List of Tables

Chapter One: Introduction

1 Introduction

1.1 Introduction

Blind people face great challenges to independence and mobility. As a response to this, innovative technological solutions have emerged to enhance the lives of the visually impaired, The project titled "Guiding Glasses for Blind People Using Computer Vision and ChatGPT" aims to leverage the capabilities of computer vision technology and advanced AI-powered systems to create novel assistive guiding glasses that empower blind individuals to navigate their surroundings with greater confidence and autonomy.

The field of assistive technology has seen significant advancements, with AI-powered solutions playing a pivotal role in enhancing the lives of individuals with visual impairments. One such innovation is the development of smart eyeglasses equipped with artificial intelligence capabilities to detect objects and provide detailed measurements, thereby assisting users in navigating their surroundings.

1.2 Problem Definition

It's eyeglasses that act like an eye to guide blind people to walk without a cane give them continuous live detection of the objects around them like tables, doors, people, and all objects that they face, and measure the destination of each object near him and update measurement every meter, the glasses will be connected to ChatGPT to search any information that he would like.

Many individuals with visual impairments face challenges in accurately perceiving and interacting with their environment due to limitations in object recognition and measurement capabilities. Traditional assistive devices often fall short in providing real-time, contextually relevant information, leading to potential hazards and reduced independence for users.

We may face some Challenges:

1. Object Detection:
    - Develop an object detection algorithm capable of identifying various objects in real-time.
    - Optimize the model for both speed and accuracy to ensure prompt feedback to the user.
2. Object Classification:
    - Implement a classification system to categorize detected objects into relevant classes (e.g., class chairs, tables, garbage, etc.).
3. Measurement Estimation:
    - Integrate measurement estimation algorithms to provide accurate dimensions and distances of detected objects.
4. Environmental Adaptability:
    - Ensure the system can adapt to varying lighting conditions.

1.3 Project Scope

The primary objective of this project is to design, develop, and Guide glasses for blind people equipped with advanced computer vision object detection and ChatGPT and measurement capabilities to enhance the independence and safety of visually impaired individuals in the College of Computer and Information Sciences in IMAMU.

In the future, we plan to increase the project scope to cover and detect all objects in real life to help as many people as possible from walking in streets normally without a cane to reading words and numbers and recognizing people's faces.

Our project should detect every object and able a blind person to move freely in the College of Computer and Information Sciences on the IMAMU campus without any help from a cane or a friend.

1.4 Local Impact

Our eyeglasses focus on revolutionizing the lives of visually impaired individuals in our community. This technology offers a range of transformative benefits:

- Enhanced Mobility and Independence: Visually impaired individuals will experience a significant boost in their ability to navigate and interact with their surroundings.
- Improved Safety and Accident Prevention: With the ability to detect objects in their path, users can avoid potential hazards and obstacles, reducing the likelihood of accidents and falls.
- Increased Accessibility in Everyday Activities: Tasks that were once challenging or impossible, such as identifying and locating specific objects in a room, become achievable with the assistance of our designed eyeglasses.
- Empowerment in Educational Settings: Students with visual impairments will gain a powerful tool to facilitate their learning experiences. identifying classroom materials, and participating in interactive activities become more accessible.
- Greater Employment Opportunities: By providing visually impaired individuals with a tool that enhances their ability to engage with their environment, eyeglasses can open up new employment opportunities. This increased independence and capability can lead to greater job prospects and career advancement.
- Economic Impact: The adoption of this technology within the community can spur local economic growth. It may lead to the development of specialized training programs, maintenance services, and potentially even local production or assembly facilities for these eyeglasses.

1.5 Global Impact

The deployment of eyeglasses equipped with object detection and measurement capabilities holds immense potential for transforming the lives of visually impaired individuals worldwide. This innovative technology is poised to have far-reaching effects on a global scale:

- Universal Accessibility: The widespread adoption of AI-powered eyeglasses can break down barriers for visually impaired individuals across the globe. It can facilitate greater inclusion in education, employment, and daily activities, leading to improved quality of life for millions of people.
- Education: By enhancing the ability of visually impaired students to engage with educational materials, our eyeglasses can play a pivotal role in improving educational outcomes in regions where visual impairment poses a significant barrier to learning.
- Safety and Mobility in Urban Environments: In densely populated urban areas, where navigating crowded streets and public transportation systems can be challenging for visually impaired individuals, the object detection capabilities of these eyeglasses can significantly improve safety and mobility.
- Global Workforce Integration: Access to these eyeglasses can empower visually impaired individuals to participate in a wider range of industries and occupations, contributing to a more diverse and inclusive global workforce.
- Technological Diplomacy and Cooperation: The development and distribution of AI-powered eyeglasses can serve as a testament to global technological collaboration, showcasing the potential for innovation to address pressing social challenges on an international scale.

1.6 Aims and Objectives

The primary goal of this project is to design and develop AI-powered smart eyeglasses capable of accurately detecting objects in a user's vicinity and providing precise measurements, thereby enabling individuals with visual impairments to navigate and interact with their environment more effectively.

Objective Definition	Technical requirements	Duration
Start collecting pictures for object detection model	- Labelimage - deep learning	25 Days
Start developing object distance detection	- Python -Whisper	15 Days
Start developing object direction detection.	- Python - Whisper	25 Days
Integration with ChatGPT	- ChatGPT Integration - Whisper	11 Days
Assembling prototype	-	12 Days

Table 1: Objectives

1.7 Alternative Solutions

Alternative Solutions	**Description**	**Benefit**	**Cost**
Use a pre-trained model	There are many pre-trained object detection models available online. These models have been trained on large datasets and can achieve good performance on a variety of tasks**.**	Time savings	Computational cost
Use hardware acceleration	Many hardware platforms, such as GPUs and FPGAs, offer hardware acceleration for object detection. This can significantly improve the speed of inference, without sacrificing too much accuracy.	Efficiency	Hardware and Software cost
Use a pre-trained classification model	There are many pre-trained classification models available online.	Reduced Development Time	Data Acquisition and Preparation
Cloud-based services	Several cloud-based services offer measurement estimation capabilities	Scalability	Pay-as-you-go pricing
Lighting Condition-Aware Object Detection Al	One solution is to develop object detection algorithms that are aware of the lighting conditions. These algorithms can adjust their sensitivity and threshold values based on the lighting conditions.	Reduced risk of accidents	developing

Table 2: Alternative Solutions

1.8 Method / Approach

We chose the Agile methodology to make and manage our project, as we saw it's the best fit for our project because it relies on a lot of simultaneous steps from collecting the data to choosing and training the model and testing.

The success of our project depends on the integration of thee advanced technologies: computer vision for object, direction, distance detection, ChatGPT as an AI assistant, and MongoDB as a cloud database. Both technologies have already demonstrated their potential in various applications, making them technically feasible for your guiding glasses.

Computer Vision:

Existing computer vision libraries and frameworks like OpenCV, provide robust tools for object recognition, obstacle detection, and scene analysis.

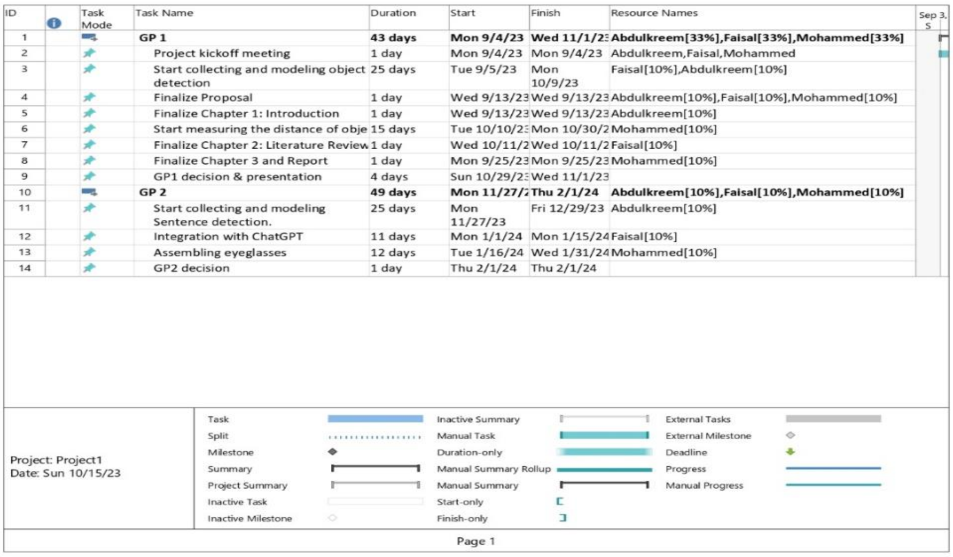
ChatGPT Integration:

Incorporating ChatGPT into your system requires integrating its API, which can be achieved through coding and cloud services.

MongoDB:

Is an open-source, NoSQL, cloud database known for its high performance, scalability, and flexibility.

1.9 Project Timeline



ID		Task Mode	Task Name	Duration	Start	Finish	Resource Names
1			**GP 1**	**43 days**	**Mon 9/4/23**	**Wed 11/1/2**	**Abdulkreem[33%],Faisal[33%],Mohammed[33%]**
2			Project kickoff meeting	1 day	Mon 9/4/23	Mon 9/4/23	Abdulkreem,Faisal,Mohammed
3			Start collecting and modeling object detection	25 days	Tue 9/5/23	Mon 10/9/23	Faisal[10%],Abdulkreem[10%]
4			Finalize Proposal	1 day	Wed 9/13/23	Wed 9/13/23	Abdulkreem[10%],Faisal[10%],Mohammed[10%]
5			Finalize Chapter 1: Introduction	1 day	Wed 9/13/23	Wed 9/13/23	Abdulkreem[10%]
6			Start measuring the distance of obje	15 days	Tue 10/10/2	Mon 10/30/2	Mohammed[10%]
7			Finalize Chapter 2: Literature Review	1 day	Wed 10/11/2	Wed 10/11/2	Faisal[10%]
8			Finalize Chapter 3 and Report	1 day	Mon 9/25/23	Mon 9/25/23	Mohammed[10%]
9			GP1 decision & presentation	4 days	Sun 10/29/2	Wed 11/1/23	
10			**GP 2**	**49 days**	**Mon 11/27/2**	**Thu 2/1/24**	**Abdulkreem[10%],Faisal[10%],Mohammed[10%]**
11			Start collecting and modeling Sentence detection.	25 days	Mon 11/27/23	Fri 12/29/23	Abdulkreem[10%]
12			Integration with ChatGPT	11 days	Mon 1/1/24	Mon 1/15/24	Faisal[10%]
13			Assembling eyeglasses	12 days	Tue 1/16/24	Wed 1/31/24	Mohammed[10%]
14			GP2 decision	1 day	Thu 2/1/24	Thu 2/1/24	

Figure 1: WBS 1

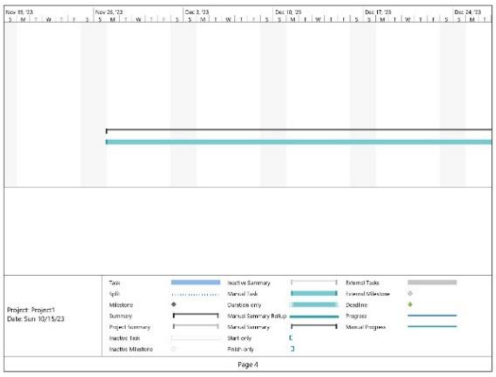


Figure 2: WBS 2

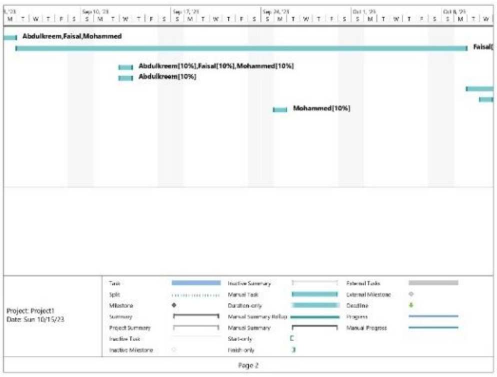


Figure 3: WBS 3

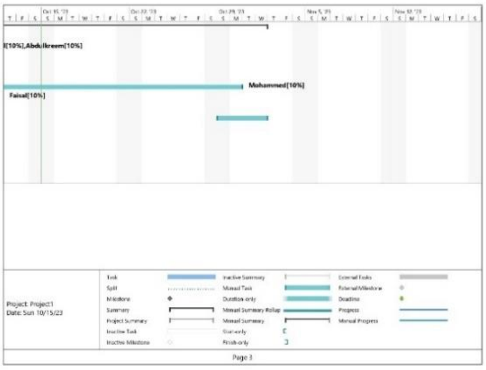


Figure 4: WBS 4

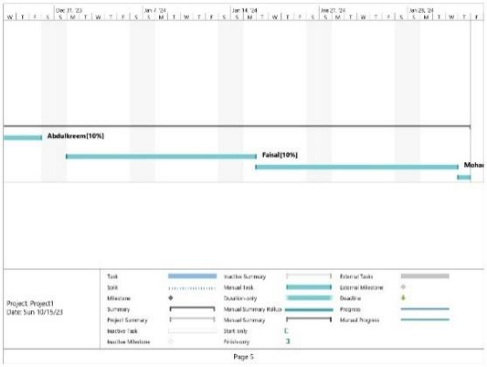


Figure 5: WBS 5

1.10Report Structure

In chapter one, we introduced a project titled "Guiding Glasses for Blind People Using Computer Vision and ChatGPT" aimed at creating innovative assistive glasses for the visually impaired. These glasses leverage computer vision technology and advanced AI to empower blind individuals to navigate their surroundings independently. The project addresses challenges faced by visually impaired individuals in object recognition and measurement, aiming to provide real-time, contextually relevant information.

we outlined the local and global impacts of these glasses, highlighting benefits such as enhanced mobility, improved safety, increased accessibility, empowerment in educational settings, greater employment opportunities, and potential economic impact on the local community. We also defined the project scope, and objectives, and presented alternative solutions. The methodology chosen for the project is Agile, which involves simultaneous steps from data collection to model training and testing and emphasizes the integration of computer vision and ChatGPT technologies.

In chapter two, we will conduct a comprehensive review of the existing literature related to our project, This includes exploring the evolution and principles of Artificial Intelligence (AI), focusing on its application in assistive technology for the visually impaired. We will also delve into prior research efforts in areas such as real-time object detection, visual odometry for navigation, text detection in images, and distance measurement using computer vision techniques. This retrospective analysis will serve as a valuable framework for informing our approach and contributing to the advancement of assistive technology for the visually impaired.

In chapter three, we focus on designing a comprehensive solution for visually impaired individuals. This solution leverages advanced technologies such as computer vision and ChatGPT. Our primary goal is to provide detailed environmental analysis and enable natural language interaction for the users. We discuss the system perspective, including dependencies on external components like the Computer Vision System and ChatGPT Integration. Key features include real-time environmental analysis, obstacle detection, object recognition, text-to-speech conversion, and user preference customization. We collect user requirements through interviews and surveys to guide the system design.

In chapter Four, we covers the implementation details of a project, including the choice of Python as the programming language, development environments, and MongoDB for database management. It discusses the user interface setup with Raspberry Pi 4 and associated hardware, and details the implementation of login/signup functionality using MongoDB Atlas and Python. The chapter also lists various packages and libraries used, such as Whisper_Mic, pyttsx3, and OpenCV, and describes specific procedures like object direction determination and search engine integration with ChatGPT.

In chapter Five, we covers the comprehensive testing process for the "Guiding Glasses for Blind People Using Computer Vision and ChatGPT" system, focusing on unit, integration, and user acceptance testing. It details the evaluation of system components like Whisper, YOLOv8, Distance and Direction functionality, and ChatGPT integration, and discusses the challenges faced, such as limitations in processing power and speech recognition difficulties, along with strategies to overcome them. The chapter concludes with user feedback, test cases, and a summary of the testing phase's results and lessons learned.

In chapter six, we concludes the project on "Guiding Glasses for Blind People Using Computer Vision and ChatGPT," summarizing the achievements and the challenges encountered. It then outlines future work to expand the project scope, enhance measurements, improve pathfinding, add text detection and reading capabilities, and move towards manufacturing the glasses.

1.11 Summary

The project "Guiding Glasses for Blind People Using Computer Vision and ChatGPT" aims to address the challenges faced by visually impaired individuals in achieving independence and mobility. By leveraging computer vision and advanced AI systems, the project aims to create innovative assistive glasses that empower blind individuals to navigate with confidence. The field of assistive technology has seen significant advancements, with smart eyeglasses equipped with AI capabilities emerging as a promising solution. These glasses can detect objects and provide detailed measurements, enhancing users' ability to interact with their environment.

The main problem being addressed is the limited independence and mobility experienced by visually impaired individuals due to challenges in object recognition and measurement. Traditional assistive devices often fall short of providing real-time, contextually relevant information, leading to potential hazards and reduced independence.

The project faces challenges in areas such as object detection, classification, measurement estimation, and adaptability to varying lighting conditions.

On a local level, the glasses aim to revolutionize the lives of visually impaired individuals in the community by enhancing mobility, safety, and accessibility. This technology offers benefits such as improved safety, accessibility, empowerment in educational settings, increased employment opportunities, and potential economic impact.

Globally, the project's impact is substantial, promising universal accessibility, improved education, enhanced safety in urban environments, greater workforce integration, and showcasing global technological collaboration.

Alternative solutions include using pre-trained models, hardware acceleration, pre-trained classification models, cloud-based services, and implementing lighting condition-aware object detection algorithms. The chosen Agile methodology is well-suited for managing the project's simultaneous steps, from data collection to model training and testing. The success of the project hinges on the integration of computer vision and ChatGPT technologies, both of which have demonstrated their potential in various applications.

The project's importance lies in its potential to significantly enhance the lives of visually impaired individuals, providing them with greater independence, mobility, and safety. The methodology combines cutting-edge technologies to create a novel solution that builds upon previous related work in assistive technology.

Chapter Two: Literature Review

2 Literature Review

2.1 Introduction

Chapter 1 introduces the project, "Guiding Glasses for Blind People Using Computer Vision and ChatGPT," aiming to empower visually impaired individuals through innovative technology. It highlights the challenges faced by blind people and the emergence of assistive technologies. The focus is on smart eyeglasses equipped with AI for object detection and measurements. The problem is defined as creating eyeglasses that act as a guide, providing real-time object detection and interaction with ChatGPT. The chapter emphasizes challenges in object recognition, classification, measurement, and adaptability to lighting conditions. It discusses the local and global impacts of the project, focusing on enhanced mobility, safety, accessibility, education, employment, and economic growth. The project scope is defined, targeting the College of Computer and Information Sciences at IMAMU, with future expansion plans. The aims and objectives are outlined, detailing technical requirements and durations for various project phases. Alternative solutions and the chosen Agile methodology are presented. This chapter establishes a strong foundation for the project's purpose, challenges, and goals.

In this chapter, our attention turns towards establishing the foundational knowledge in the realm of Artificial Intelligence (AI) and providing an overview of previous endeavors in this field. By delving into the background of AI, we aim to provide a comprehensive understanding of the evolution and principles that underpin this transformative technology. Additionally, we will explore prior initiatives and research efforts that have paved the way for the innovative project at hand. This retrospective analysis will serve as a valuable framework, allowing us to draw upon existing knowledge and experiences, ultimately informing our approach to developing the Guiding Glasses for Blind People using Computer Vision and ChatGPT. Through this contextual exploration, we endeavor to build upon the advancements of the past, propelling us toward a more inclusive and technologically empowered future for visually impaired individuals.

2.2 Background

2.2.1 The importance of guiding glasses in the AI field

AI-powered eyeglasses that detect objects and provide object measurements hold significant importance in the field of AI for several reasons:

Technology Advancements: These AI-powered eyeglasses represent a remarkable advancement in assistive technology

Object Detection Challenges: Developing AI algorithms for real-time object detection, especially in varying lighting conditions, presents complex challenges. These challenges push the boundaries of computer vision and AI research, fostering innovation in the field.

Real-World Applications: Beyond assisting visually impaired individuals, AI eyeglasses can find applications in various fields, such as healthcare, manufacturing, and navigation systems, showcasing AI's versatility and potential impact across industries.

AI eyeglasses have the potential to break down barriers for visually impaired individuals worldwide. This aligns with the global accessibility and inclusion goals of AI technology.

The development and distribution of such AI-powered devices can serve as a testament to global technological collaboration and cooperation, illustrating AI's potential to address pressing social challenges on an international scale.

2.2.2 The history of AI

The history of Artificial Intelligence (AI) begins in the 1950s with the coining of the term by John McCarthy and the establishment of AI as a field of study at the Dartmouth Conference. Early AI focused on symbolic reasoning and problem-solving. In the 1970s and 1980s, AI faced a period of reduced funding and interest, known as the "AI Winter."

In the 1980s, expert systems and knowledge-based AI gained prominence, using symbolic logic to represent knowledge. This period saw practical applications in fields like medical diagnosis and chemical analysis. The 1990s and early 2000s brought a focus on machine learning, enabling computers to learn from data.

The late 1990s and 2000s witnessed practical applications in speech recognition and computer vision. Breakthroughs in deep learning in the 2010s led to advancements in natural language processing and game-playing, with notable achievement.

Recent years have seen the development of powerful AI models like GPT-2, GPT-3, GPT-3.5, and GPT-4 for natural language processing. AI continues to advance, with ongoing research and applications in various fields.

2.2.3 The definitions of fundamental concepts in the field.

Artificial Intelligence (AI): AI refers to the simulation of human intelligence in machines that are programmed to think and learn like humans. It involves tasks such as problem-solving, speech recognition, learning, and planning.

Machine Learning (ML): Machine Learning is a subset of AI that focuses on the development of algorithms that enable computers to learn from and make decisions or predictions based on data, without being explicitly programmed.

Deep Learning (DL): Deep Learning is a subfield of machine learning that uses neural networks with multiple layers to process and learn from data. It is particularly effective for tasks like image and speech recognition.

Neural Network (NN): A neural network is a set of algorithms, modeled loosely after the human brain, that is designed to recognize patterns. It interprets sensory data through a kind of machine perception, labeling, or clustering of raw input.

Natural Language Processing (NLP): NLP is a branch of AI that deals with the interaction between computers and human language. It enables machines to understand, interpret, and generate human language in a valuable way.

Computer Vision (CV): Computer Vision is a field of AI that enables computers to interpret and understand visual information from the world. It involves tasks like object recognition, image segmentation, and object tracking.

Reinforcement Learning: Reinforcement Learning is a type of machine learning algorithm that learns by interacting with an environment. It receives feedback in the form of rewards or penalties and uses that information to make decisions.

Supervised Learning: In supervised learning, the model is trained on a labeled dataset, where the input data is paired with the correct output. The goal is to learn a mapping from inputs to outputs.

Unsupervised Learning: Unsupervised learning involves training a model on an unlabeled dataset, allowing it to discover patterns and structures in the data without any predefined outputs.

Feature Engineering: Feature engineering is the process of selecting and transforming the raw data into a format that is suitable for machine learning algorithms. It involves creating new features or modifying existing ones.

2.3 Related Work

2.3.1 Real-Time Object Detection Using Deep Learning and Open CV [1]

This paper discusses various approaches for object detection like

1- HOG In this approach we input the object features calculated from each window obtained by running a sliding window to SVM (Support Vector Machine) algorithm to create classifiers.

HOG features are computational inexpensiveness.

HOG Problem Very sensitive to image rotation and Final feature descriptor grows larger and it takes too much time to extract objects.

Figure 6: HOG

2- Regional-based Convolutional Neural Networks (R-CNN) in this approach Sliding window method needs to capture all the possible locations for objects in an image and also at different scales too, this is complex.

R-CNN gives rise to a more accurate result than HOG-based implementation.

RCNN technique is slower and very expensive, using the sliding window concept

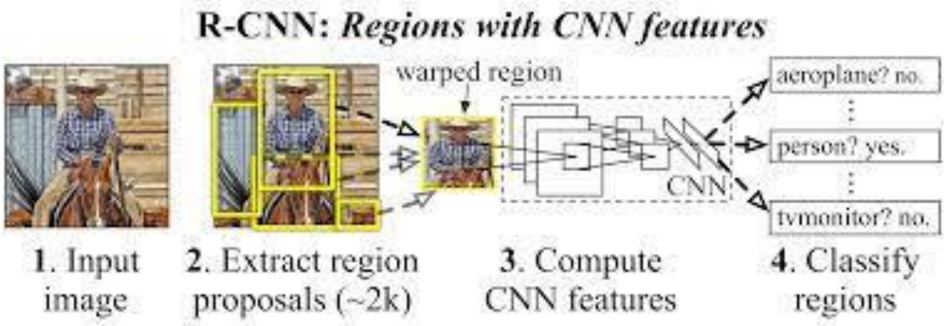


Figure 7: R-CNN

2.3.2 Improving Computer Vision-Based Perception for Collaborative Indoor Navigation [2]

This paper discusses the development of an enhanced visual odometry solution for indoor navigation, specifically focusing on the challenges of using a small, low-cost RGB-D camera. Visual odometry is a technique used to estimate a camera's motion and position based on visual input. The primary problem addressed in this research is the "scale issue" that arises when using a monocular camera, which makes it difficult to accurately determine distances. To mitigate this issue, the authors employed an RGB-D camera.

2.3.3 Object Detection with Deep Learning: A Review [3]

This paper provides an in-depth review of the field of deep learning-based object detection, highlighting its significant contributions and applications in recent years. Deep learning has gained attention due to its capacity to handle challenges like occlusion, scale transformation, and background variations effectively. The paper focuses on various object detection frameworks, most of which build upon the R-CNN architecture, and discusses their modifications and adaptations to address specific sub-problems.

2.3.4 Text Detection in Natural and Computer-Generated Images [4]

This paper addresses the challenging task of text detection in computer vision

The proposed approach uses a multi-step process:

A- Maximally Stable Extremal Regions (MSER): identify potential text region candidates within the image. MSER is a method used for stable region detection in images.

Figure 8: Sample regions

B- Geometric Elimination: helps reduce the number of false positives and improves the accuracy of text region identification by determining to eliminate too small regions in an image.

Figure 9: 1306 regions

C- Stroke Width Transformation (SWT) is an image processing operation that calculates the width of text strokes (lines) in a given image region. It associates each pixel in the region with its estimated stroke width value.

Figure 10: 980 regions

D- Connecting characters: refers to the process of joining or linking individual characters within a text or handwriting recognition context.

E- OCR Engine Assistance: refers to a set of techniques and processes aimed at improving the performance and accuracy of Optical Character Recognition (OCR) engines.

Figure 11: Final bounding boxes

2.3.5 Computer Vision-Based Distance Measurement System Using Stereo Camera View [5]

This paper discusses the development and evaluation of a computer vision system that utilizes a stereo camera system for measuring the distances between human faces and a screen,

The result of this paper is

The actual distances ranged from 71 cm to 132 cm, and the proposed system's measurements were generally very close to the actual values.

For instance, actual distances of 71 cm, 74 cm, 75 cm, and 79 cm were measured as 70 cm, 72 cm, 73 cm, and 77 cm by the system, respectively.

The experimental results suggest that the proposed computer vision system is successful in accurately measuring distances using a stereo camera setup

Category 1: Object Detection

Paper [1]: Real-Time Object Detection Using Deep Learning and Open CV

Approaches: HOG-based object detection and R-CNN.

Similarities and Differences:

Both approaches aim at object detection.

HOG relies on SVM for classification, while R-CNN uses CNNs.

HOG is computationally less expensive but sensitive to image rotation. R-CNN is more accurate but slower and computationally expensive.

Both face challenges in handling scale variations and complexity.

Shortcomings:

HOG is sensitive to image rotation.

R-CNN is computationally expensive and slower due to the sliding window concept.\

Paper [3]: Object Detection with Deep Learning: A Review

Focus: In-depth review of deep learning-based object detection methods.

Highlights: Discusses various frameworks building upon R-CNN architecture.

Applications: Addresses challenges like occlusion, scale transformation, and background variations effectively.

Category 2: Visual Odometry for Indoor Navigation

Paper [2]: Improving Computer Vision-Based Perception for Collaborative Indoor Navigation

Focus: Enhancing visual odometry for indoor navigation using a small RGB-D camera.

Problem Addressed: The scale issue in monocular visual odometry.

Approach: Utilizing an RGB-D camera to improve accuracy in estimating distances for indoor navigation.

Category 3: Text Detection in Images

Paper [4]: Text Detection in Natural and Computer-Generated Images

Approach: Multi-step process involving MSER, geometric elimination, SWT, connecting characters, and OCR engine assistance.

Focus: Detecting text in images, with techniques to reduce false positives and improve accuracy.

Category 4: Distance Measurement Using Stereo Camera

Paper [5]: Computer Vision-Based Distance Measurement System Using Stereo Camera View

Focus: Development and evaluation of a computer vision system for distance measurement using stereo cameras.

Results: The system successfully measured distances close to actual values, demonstrating accuracy in distance measurements.

Similarities:

Both Paper [1] and Paper [3] focus on object detection using different techniques, with Paper [3] providing a comprehensive review of deep learning-based object detection methods.

Differences:

Paper [2] addresses visual odometry for indoor navigation, while Papers [4] and [5] focus on text detection and distance measurement using computer vision techniques, respectively.

Shortcomings:

Each paper addresses specific challenges within their respective domains, but detailed shortcomings are not explicitly mentioned in the provided information.

A visual representation of this categorization could be presented in a figure, with each category branching out to the specific papers and their respective focuses, approaches, and results. This helps in providing a clear overview of the research landscape.

Proposed Contribution: The proposed project seeks to integrate and advance existing technologies by creating a comprehensive solution that combines computer vision, object detection, text recognition, navigation assistance, and conversational AI. By combining these elements into a single wearable device, it aims to offer a more holistic and versatile tool for the blind, addressing multiple aspects of their daily lives. The system's ability to continuously detect and measure objects in real-time, read text and numbers, and access information via ChatGPT adds a new dimension of functionality that can significantly enhance independence and mobility.

Innovation: The innovative aspect of this research project lies in the integration of multiple technologies into a single wearable device, the "Guiding Glasses." This integration allows for real-time object detection and measurement, text recognition, and on-demand information retrieval, all in one package. Furthermore, the project leverages state-of-the-art AI technology like ChatGPT to provide a natural and accessible interface for the user. The innovation lies in creating a comprehensive and user-friendly solution that addresses the multifaceted challenges faced by visually impaired individuals, ultimately improving their quality of life and independence.

2.4 Summary

Chapter 2 provides an extensive background on Artificial Intelligence (AI) and its historical evolution. It emphasizes the pivotal role of AI-powered guiding glasses in revolutionizing assistive technology for visually impaired individuals. The chapter outlines the significance of object detection and measurement capabilities in AI eyeglasses, demonstrating their potential applications beyond aiding the visually impaired. The historical timeline of AI development, from its inception in the 1950s to the present day, is discussed, highlighting key milestones and shifts in focus. Fundamental concepts in the field of AI, such as Machine Learning, Deep Learning, Neural Networks, and Natural Language Processing, are defined to establish a solid foundation for subsequent technical discussions. Furthermore, the chapter introduces related work, categorizing research papers based on their focus areas: Object Detection, Visual Odometry for Indoor Navigation, Text Detection in Images, and Distance Measurement Using Stereo Camera. Each category is summarized, highlighting the approaches and contributions of the respective papers. The proposed contribution of the project is introduced, emphasizing the integration of various technologies into the "Guiding Glasses" to offer a comprehensive solution for the visually impaired. The project's innovation lies in its ability to address multiple facets of the daily lives of visually impaired individuals through a unified wearable device, ultimately enhancing their independence and mobility.

Chapter Three: System Analysis and Design

3 System Analysis and Design

3.1 Introduction

In this chapter, we delve into the comprehensive solution designed to empower visually impaired individuals in navigating their surroundings. Leveraging cutting-edge technologies, including computer vision and ChatGPT, the system aims to provide detailed environmental analysis and natural language interaction. However, it's important to note that the system may rely on various external components, such as the Computer Vision System, ChatGPT Integration, and specific hardware for optimal functionality.

The core of our system lies in the computer vision module, which interprets the environment through advanced image processing algorithms. Dependencies may include essential libraries like OpenCV and hardware components like cameras and Raspberry Pi.

ChatGPT integration introduces a unique dimension, enabling natural language interaction for tasks like inquiring about surroundings, reading signs, and obtaining information. Dependencies may involve language processing APIs, real-time internet connectivity, and adherence to ChatGPT's system requirements.

we outline the functional and non-functional requirements that define the capabilities and performance standards of our system.

3.2 System Perspective

Our system is designed as a comprehensive solution to assist visually impaired individuals in navigating their surroundings. The system, while leveraging computer vision and ChatGPT, may have dependencies on external components. Specifically:

1-Computer Vision System:

The computer vision module is a core component that relies on image processing algorithms to interpret the environment. Dependencies might include libraries like OpenCV and hardware components such as cameras and Raspberry Pi.

2- ChatGPT Integration:

The integration with ChatGPT for natural language interaction could involve dependencies on language processing APIs, internet connectivity for real-time responses, and adherence to ChatGPT's system requirements.

3- Hardware Platform:

The glasses may depend on specific hardware components for optimal functionality, such as microphones, speakers, cameras, and eyeglasses.

our system stands out due to several key features:

-Our system employs cutting-edge computer vision techniques to provide detailed environmental analysis, allowing for precise navigation and object recognition and the distance between the visually impaired individuals and the objects around them.

- Conversational AI with ChatGPT:

The integration of ChatGPT adds a unique dimension, enabling natural language interaction for tasks like asking about surroundings, reading signs, or obtaining information.

-Real-time Updates and Adaptability:

The system prioritizes real-time data updates, ensuring that users receive the latest information about their surroundings. Additionally, it adapts to user preferences and dynamically adjusts guidance based on changing conditions.

-User-Friendly Interface:

Emphasis has been placed on creating an intuitive and user-friendly interface, making the system accessible to users with varying levels of technological expertise.

3.3 Requirements Elicitation Techniques

When we started gathering requirements, we chose to do it in two ways, Quantitative and qualitative research techniques, first, we did a one-to-one interview with a blind individual here are the questions we asked and his answers:

- What are the difficulties that blind people face in their daily lives, and the glasses can solve them?

As for the blind, whose faces are difficult to move around and the places are not prepared for them, they need things to help them with mobility, such as glasses to help them with a topic that can be read to them and shown to them.

- OK, can glasses replace the stick?

It is possible if the technology is very good. It is possible.

- what is the best distance to alert things in front of you?

Approximately one meter, a meter, and a half is the appropriate distance.

- is ChatGPT voice search service useful? Can it be useful in any possible way?

It will be very useful if it is unconventional if the things in it differ from their counterparts in the market.

- what is the appropriate cost for glasses that a blind person can afford?

I mean, for the blind, tools are very expensive. As for these tools, it would be better to have cheap prices, and their cost is cheaper, because most of the tools for the disabled are very expensive, especially the special things related to the visual disability.

Can it mean from one thousand five hundred to two thousand appropriate?

Maybe why not

- Are there any suggestions or additions that can be made to the glasses?

You learn from the things in the market and you are better than them and you start where others left off.

That was the end of the interview.

We also surveyed all the people, and we received 58 answers until now.

Do you know someone who is blind?

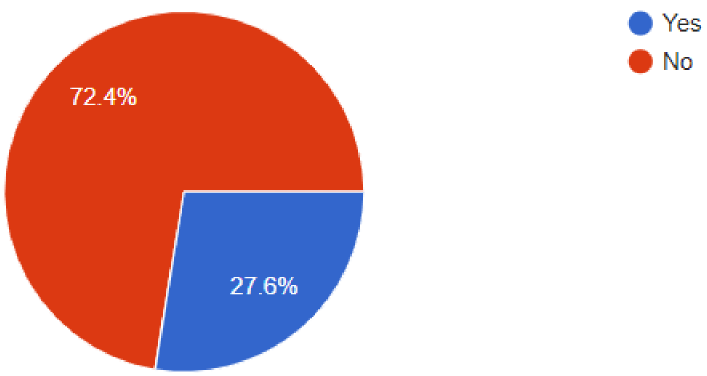


Figure 12: First answer analysis

Is it possible that these glasses save blind people from needing a cane?

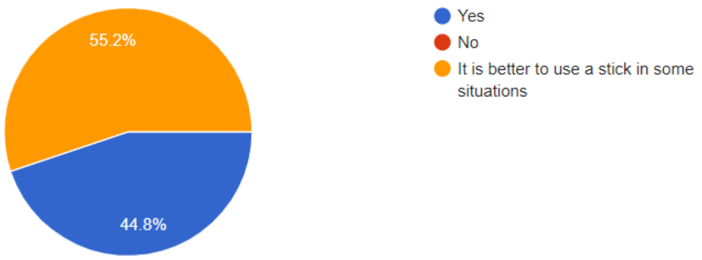


Figure 13: Second answer analysis

What is the best distance to alert the presence of objects?

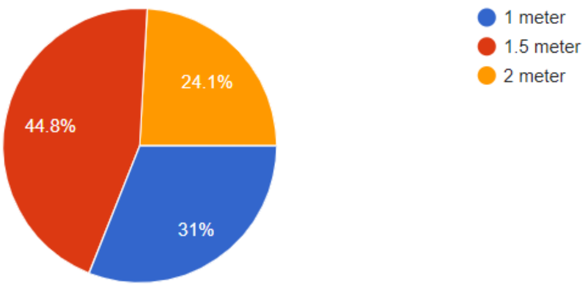


Figure 14: Third answer analysis

Is the search service in ChatGPT useful or not?

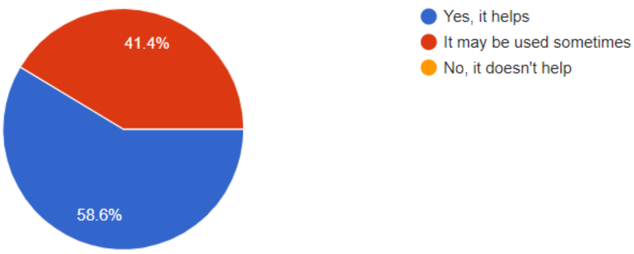


Figure 15: Fourth answer analysis

What is the appropriate cost for the glasses?

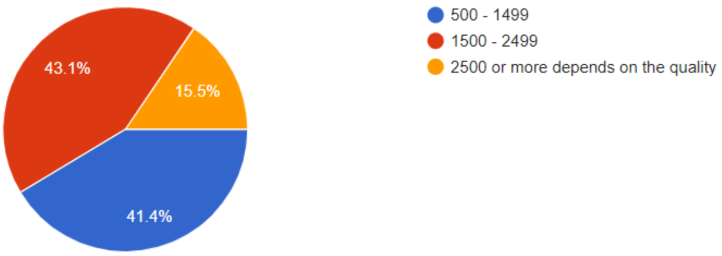


Figure 16: Fifth answer analysis

3.4 System Requirement

3.4.1 Functional Requirement

-User Registration:

The system shall allow visually impaired users to register by providing necessary personal information such as password codes.

- Real-time Environmental Analysis:

The system shall use computer vision technology to perform real-time analysis of the user's environment, identifying obstacles, objects, and spatial information.

-Natural Language Interaction:

The system shall facilitate natural language interaction, allowing users to ask questions about their surroundings, request information, and initiate dialogues for assistance.

-Obstacle Detection and Navigation:

The system shall identify obstacles in the user's path and provide real-time audio guidance to navigate around them.

-Object Recognition:

The system shall recognize and describe objects in the user's surroundings, providing information about their characteristics.

-Text-to-Speech Conversion:

The system shall convert text-based information (such as ChatGPT responses) into speech to communicate information to the user.

-User Preferences Configuration:

The system shall allow users to configure preferences, such as the level of detail in environmental descriptions, speech rate, and preferred interaction modes.

-Continuous Monitoring and Adaptation:

The system shall continuously monitor the user's environment, adapting its guidance based on real-time changes and dynamic conditions.

Object detection use case:

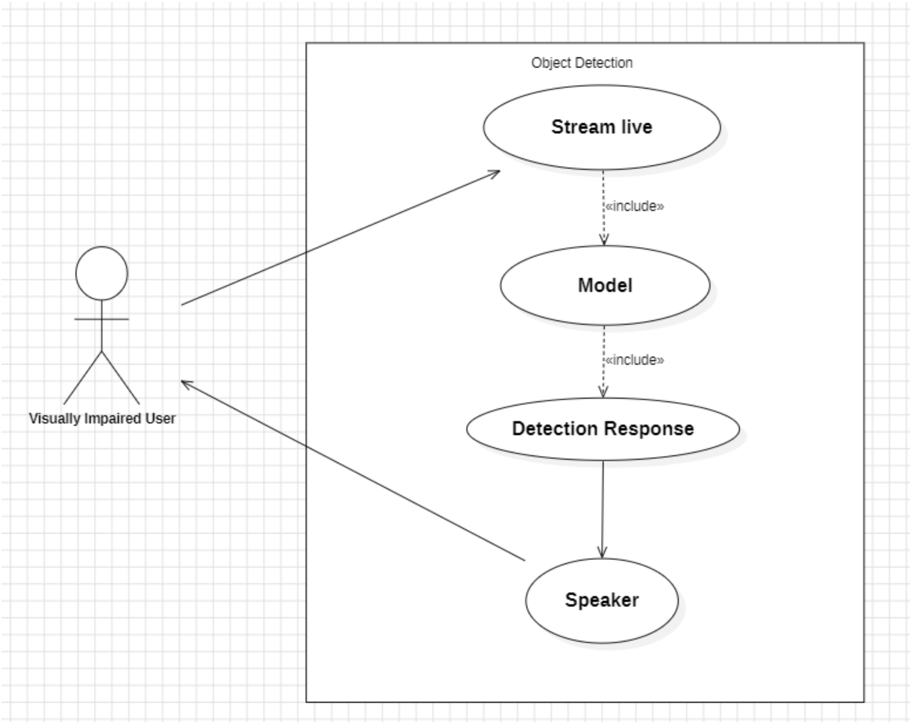


Figure 17: Object detection use case

Search in ChatGPT use case:

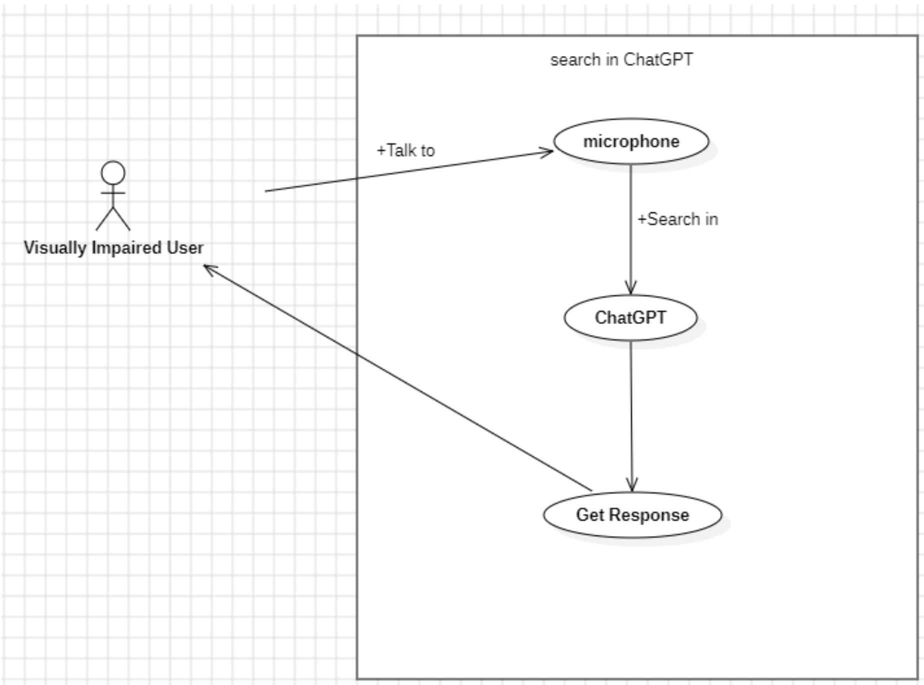


Figure 18: Search in ChatGPT use case

3.4.2 Non-Functional Requirement

- Accuracy: The glasses must be able to accurately detect and identify objects in the environment.
- Speed: The glasses must be able to detect and identify objects in real time so that the user can avoid obstacles and navigate safely.
- Reliability: The glasses must be reliable and work consistently in all lighting conditions and environments.
- Safety: The glasses must be safe for the user to wear and use. They should not obstruct the user's vision or cause any discomfort.
- Usability: The glasses should be easy for the user to learn and use. The controls should be simple and intuitive.
- Comfort: The glasses should be comfortable to wear for extended periods. They should be lightweight and have a secure fit.
- Durability: The glasses should be durable and able to withstand everyday wear and tear.

3.5 User Interface Prototype

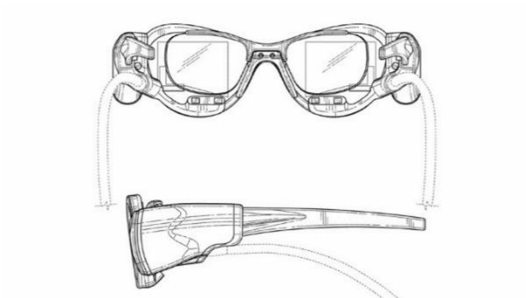


Figure 19: User interface prototype design 1

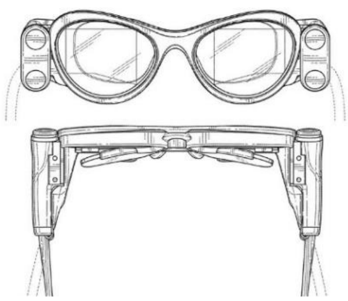


Figure 20: User interface prototype design 2

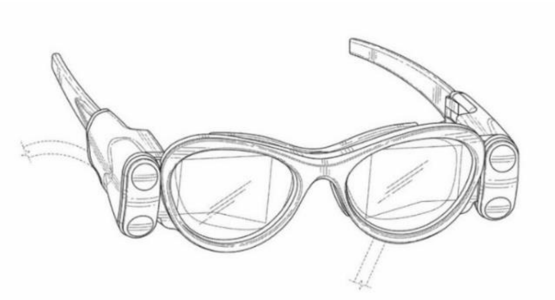


Figure 21: User interface prototype design.

3.6 System Design

3.6.1 Architectural Design

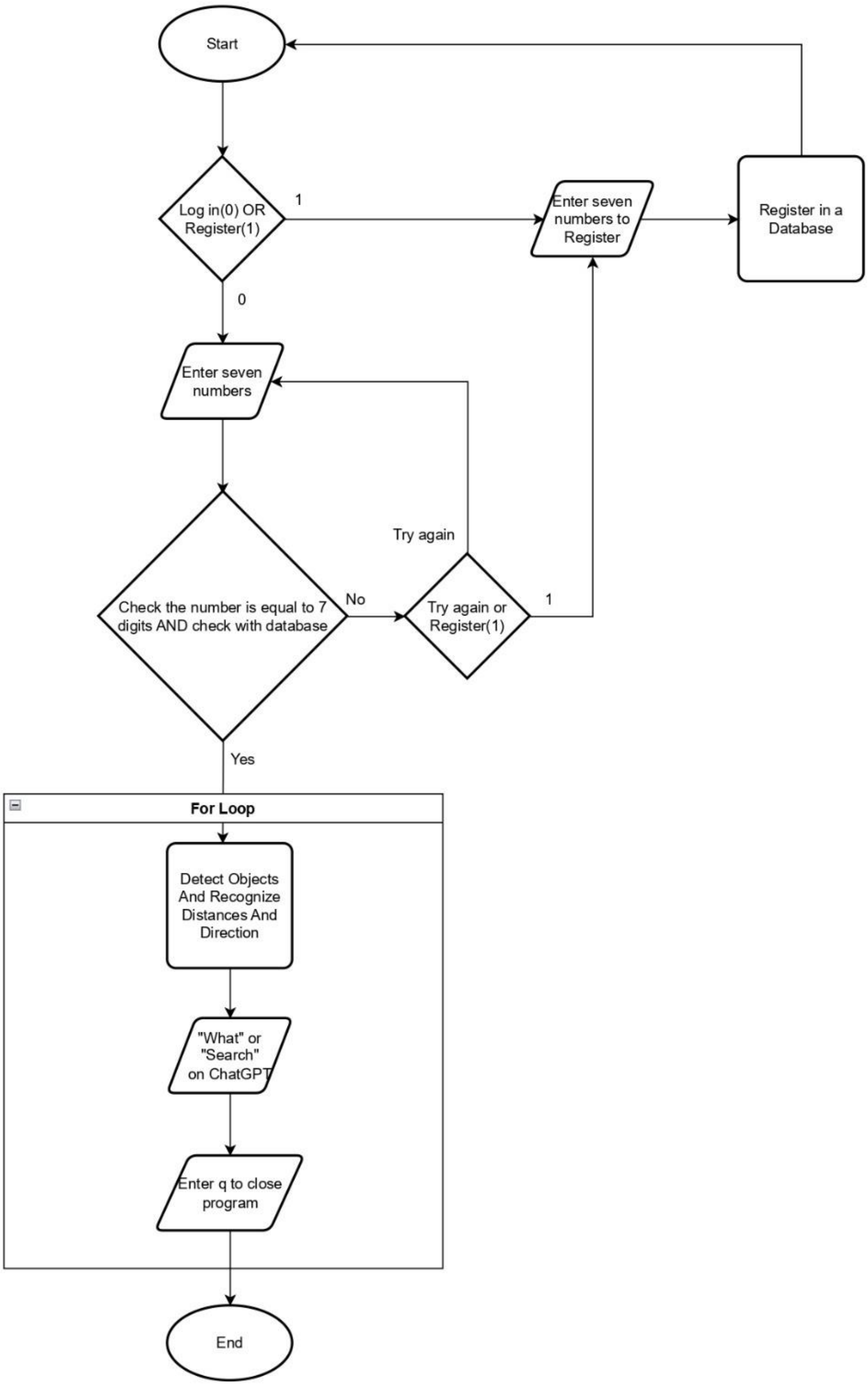


Figure 22: Architectural Design

3.6.2 Class Diagram

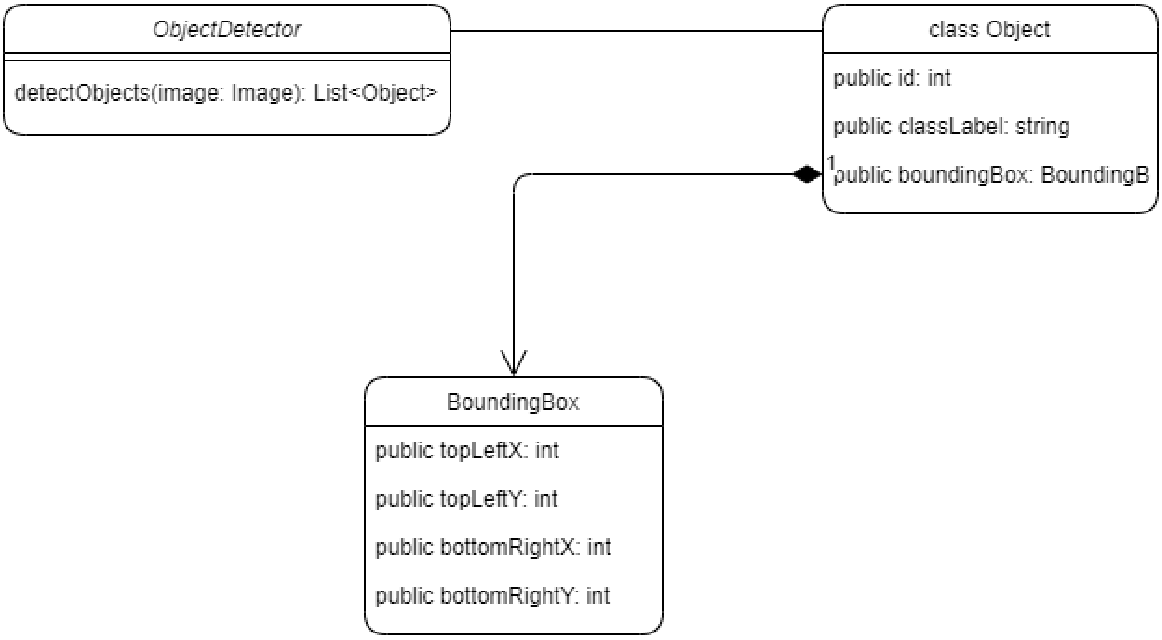


Figure 23: Class Diagram

3.6.3 Sequence Diagram

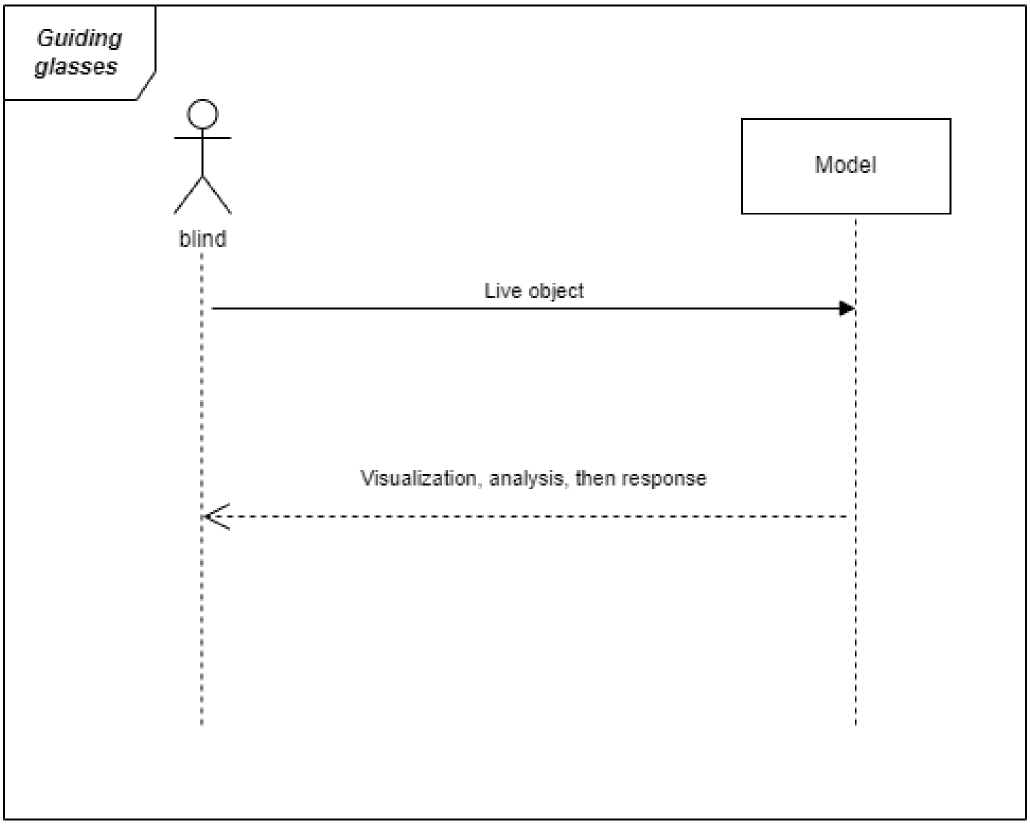


Figure 24: Sequence Diagram

3.6.4 Database Design (if any)

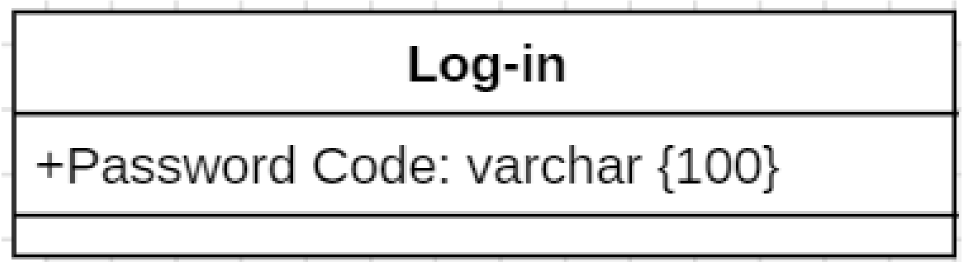


Figure 25: Database Design

3.7 Summary

The system aims to aid visually impaired individuals in navigation using computer vision and ChatGPT, potentially relying on external components. This includes a Computer Vision System using image processing with dependencies like OpenCV and hardware like cameras. ChatGPT Integration involves language processing APIs, internet connectivity, and adherence to ChatGPT's requirements. The Hardware Platform relies on components like microphones, speakers, cameras, and eyeglasses for optimal functionality.

Key features include advanced computer vision for detailed environmental analysis, natural language interaction with ChatGPT, real-time updates, and adaptability based on user preferences. Additionally, there's a strong emphasis on a user-friendly interface.

Requirements were gathered through one-on-one interviews and a subsequent survey. Functional requirements encompass user registration, real-time environmental analysis, natural language interaction, obstacle detection and navigation, object recognition, text-to-speech conversion, user preference configuration, and continuous monitoring and adaptation.

Non-functional requirements focus on the accuracy, speed, reliability, safety, usability, comfort, and durability of the glasses, ensuring their effectiveness and user-friendliness.

It’s estimated to take up to 11 weeks to be done.

In the next chapter, we will start implementing the project starting with Setting up a Computer Vision System by configuring the OpenCV library for image processing with our model and Ensuring the integration of hardware components like microphone, speakers, cameras, raspberry pi, and eyeglasses for optimal functionality.

Integrate ChatGPT to Establish natural language interaction.

Chapter Four: Implementation

4 Implementation

4.1 Introduction

In this chapter, we will delve into the programming languages, tools, user interface implementation, database implementation, packages, classes, and procedures used in our project. Firstly, we'll discuss the selection of Python as the primary programming language due to its versatility, extensive libraries, and compatibility with various hardware and software tools. Visual Studio Code, Jupyter Notebook, and Google Colab are chosen as development environments for their robust features, ease of use, and support for Python development. MongoDB is selected as the database management system due to its cloud-based scalability, flexibility, and seamless integration with Python, allowing for efficient storage and retrieval of data.

Secondly, we'll explore the user interface implementation, highlighting the main interfaces of our system and the rationale behind choosing Python alongside libraries like Whisper_Mic for voice interaction, pyttsx3 for text-to-speech conversion, and OpenCV for computer vision tasks. Additionally, we'll discuss the utilization of hardware tools such as Raspberry Pi 4 and Raspberry Pi Camera for capturing images and videos, along with batteries and connecting wires for power supply and connectivity. The implementation of login/signup functionality using MongoDB Atlas, Python, and Whisper_Mic library will be elucidated, detailing the process of greeting users, prompting them for login/signup, and storing/retrieving user information from the database.

4.2 Programming Languages and Tools

Programming Languages: Python

software tools: Visual Studio Code, Jupyter notebook, Google Colab, MongoDB, and OpenAI API.

hardware tools: Raspberry pi 4, V2.0 Raspberry Pi camera, 3.5 inch screen for Raspberry Pi, battery, Cooling Fan and connecting wires.

4.3 User Interface Implementation

The user interface implementation for our project involves integrating various components to create a seamless interaction experience. We utilize the Raspberry Pi 4 as the central processing unit, connecting it to a V2.0 Raspberry Pi camera for visual input. The 3.5-inch screen for Raspberry Pi serves as the display interface, providing users with real-time feedback and detection. To ensure uninterrupted operation, a battery is incorporated to power the Raspberry Pi.

Additionally, a cooling fan is integrated to regulate the temperature of the Raspberry Pi, ensuring optimal performance during extended use. Connecting wires are used to establish the necessary electrical connections between components, ensuring reliable communication and functionality. Together, these components form a cohesive interface that enables users to interact with the system seamlessly, facilitating a smooth and intuitive user experience.

Figure 26: User Interface Implementation 1

Figure 27: User Interface Implementation 2

4.4 Database Implementation (if any)

We choose MongoDB atlas because it offers a straightforward and intuitive way to interact with data using Python, can handle large amounts of data and high traffic loads, it offers cloud-based solutions such as MongoDB Atlas, which we used it implement our database. MongoDB Atlas for cloud-based scalability, flexibility, and easy management, while avoiding the rigid schema constraints and scaling limitations of MySQL.

We used the MongoDB atlas to implement the login/signup functionality using MongoDB Atlas and Python with the Whisper_Mic library:

1. Upon program start, greet the user and prompt them to choose between logging in or signing up.
2. If the user chooses to log in, ask the user to enter a 7-digit number. Check if the number exists in the database and log the user in if it does.
3. If the user chooses to sign up, ask the user to register a 7-digit number and verify it. Store the number in the database if it passes verification.

4.5 Packages and Classes Description

whisper_mic: that allows one to use a microphone with OpenAI whisper.

pyttsx3: is a text-to-speech conversion library in Python

Time: Python library for dealing with time conversions between universal time and arbitrary timezones.

PyMongo : distribution contains tools for interacting with MongoDB database from Python.

OpenAI: library provides convenient access to the OpenAI REST API

speech_recognition: Library for performing speech recognition, with support for several engines and APIs, online and offline.

Cv2: OpenCV (Open Source Computer Vision Library) is a library of programming functions mainly for real-time computer vision.

NumPy: is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices.

Ultralytics: provides interactive notebooks for YOLOv8, covering training, validation, tracking, and more.

Threading: A thread is a separate flow of execution. This means that your program will have two things happening at once.

Keras: provides a Python interface for artificial neural networks.

Gc: collector is keeping track of all objects in memory.

4.6 Procedures Description

This function returns the determined direction of an object:

The **direction_x(x_dir, frame_width)** function takes two arguments: x_dir, representing the center of the detected object pixel, and frame_width, representing the total width of the image. The function divides the image into seven segments based on the frame_width. Each segment corresponds to a specific direction, represented as the time on a clock face (e.g., "9 o'clock", "10 o'clock", etc.).

The function works by comparing the x_dir value with different ranges corresponding to each segment. For example, the first segment, which corresponds to "9 o'clock", includes pixels where x_dir is less than (frame_width/7 * 1 + frame_width/15). If x_dir falls within this range, the function assigns the string "9 o'clock" to the variable direction.
This function returns the direction of an object.

This Function to integrate and search with ChatGPT:

The **search_engine(word)** function that uses OpenAI's API to create an explanatory assistant. The assistant is tasked with explaining a given word in a simple sentence. The function works as follows:

1. An OpenAI client is initialized with an API key.
2. A content string is created, asking the assistant to explain the input word.
3. An assistant named "Explanatory" is created with the instruction to help users explain an object.
4. A new thread is created and a message is sent to the assistant in this thread with the content string.
5. A run is initiated for the assistant to address the user as "Mohammed".
6. The function enters a loop where it continuously checks the status of the assistant. If the assistant has completed its task, it retrieves the assistant's message, deletes the assistant, and returns the assistant's explanation. If the assistant hasn't finished yet, it prints a dot to indicate progress.

Detect every object on the camera:

The **video_to_images(video_path)** function uses OpenCV and a pre-trained deep learning model to process a video file, identify objects within each frame, and display information about these objects in real-time.

The function begins by opening the video file using cv2.VideoCapture(video_path).
Next, it enters a continuous loop where it reads each frame from the video file using cap.read(). For each frame, it makes predictions about the objects present in the frame using the model.predict() method.

Then, it iterates through the results of the predictions. For each detected object, it calculates the object's position and size in the frame, determines the object's class, calculates the distance of the object from the camera, and draws a bounding box around the object in the frame. It also displays the object's class and distance information above the bounding box.

Finally, after displaying the processed frame, it clears the session memory and waits for a key press to proceed to the next frame. If the 'q' key is pressed, it breaks the loop, releases the video capture, and ends the function.

4.7 Summary

This chapter outlines the programming languages, tools, and hardware used in the system, focusing on Python as the primary programming language and a variety of software tools including Visual Studio Code, Jupyter notebook, Google Colab, MongoDB, and the OpenAI API. Hardware tools include a Raspberry Pi 4, V2.0 Raspberry Pi camera, battery, and connecting wires. The choice of MongoDB Atlas as the database management system is justified by its compatibility with Python, scalability, and cloud-based solutions, contrasting with the rigid schema constraints and scaling limitations of MySQL. The chapter also describes the implementation of a login/signup functionality using MongoDB Atlas and Python with the Whisper_Mic library, outlining the procedures involved in greeting users, prompting for login or signup, and interacting with the database to authenticate users or store new registrations.

Additionally, the chapter provides a detailed description of packages and classes used in the system, including Whisper_Mic for voice interaction, Pyttsx3 for text-to-speech conversion, PyMongo for interacting with MongoDB, and others. It also describes specific procedures such as determining the direction of an object based on its position in an image frame and integrating a search engine with ChatGPT for explanatory purposes. Overall, the chapter provides a comprehensive overview of the programming languages, tools, hardware, database implementation, packages, classes, and procedures employed in the system's development and operation.

Chapter Five: Testing and Results

5 Testing and Results

5.1 Introduction

In Chapter 5 we delve into the comprehensive testing process undertaken for the system, focusing primarily on system testing, unit testing, integration testing, and user acceptance testing. Our goal is to ensure the robustness and functionality of the system, "Guiding Glasses for Blind People Using Computer Vision and ChatGPT," designed to address the challenges faced by visually impaired individuals through innovative technological solutions.

Throughout this chapter, we explore the testing methodologies employed to evaluate various components of the system, including Whisper (speech recognition), model object detection using YOLOv8, Distance and Direction functionality, and integration with ChatGPT. We examine the test cases designed to validate each component's performance and functionality under different scenarios, such as noisy environments, varying accents, and low-quality images. Additionally, we discuss the challenges encountered during testing, such as limitations in processing power, outdated code issues, and speech recognition difficulties, and the strategies employed to overcome these obstacles effectively.

By comprehensively testing the system's components and addressing any limitations or issues encountered, we aim to ensure the system's readiness for operational use and enhance the overall user experience for visually impaired individuals. Through meticulous testing and optimization, we strive to deliver a reliable, efficient, and user-friendly solution that significantly improves the daily lives of visually impaired individuals.

5.2 System Testing

5.2.1 Unit Testing

Model:
We test the model based on the difference between training set and validation set on several metrics

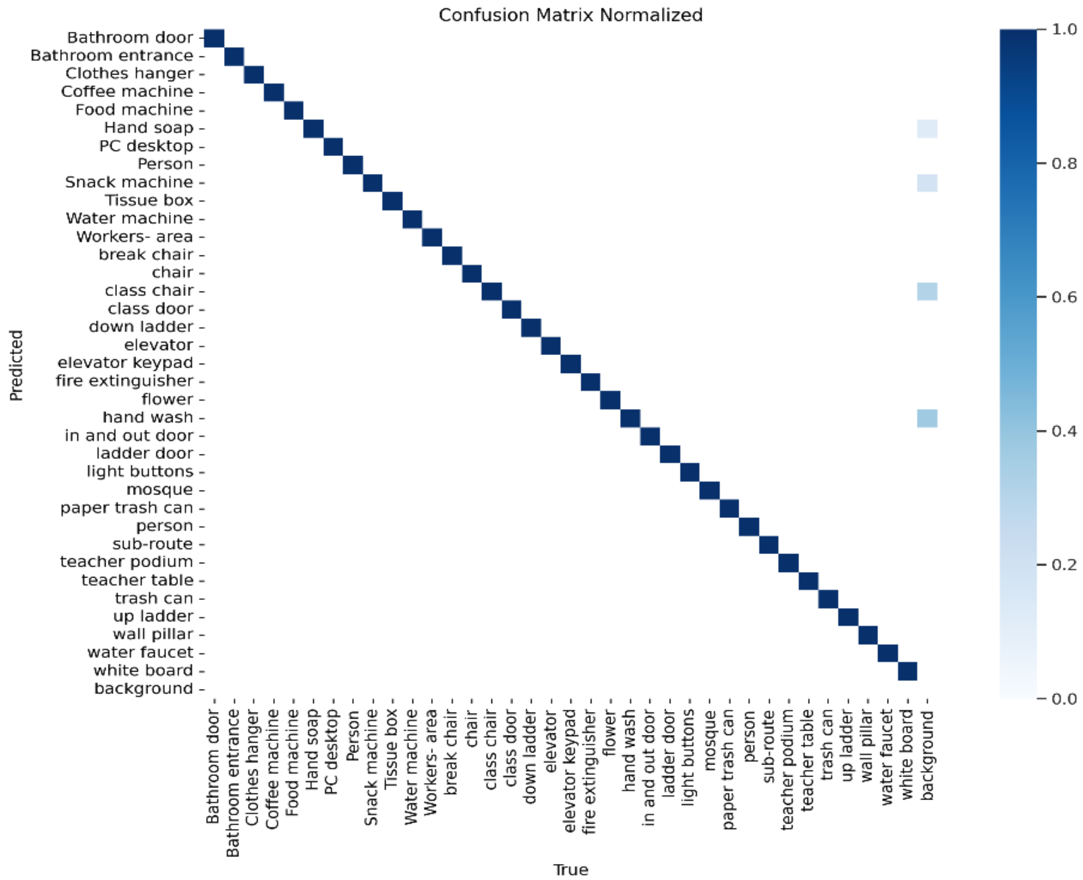


Figure 28: Confusion Matrix

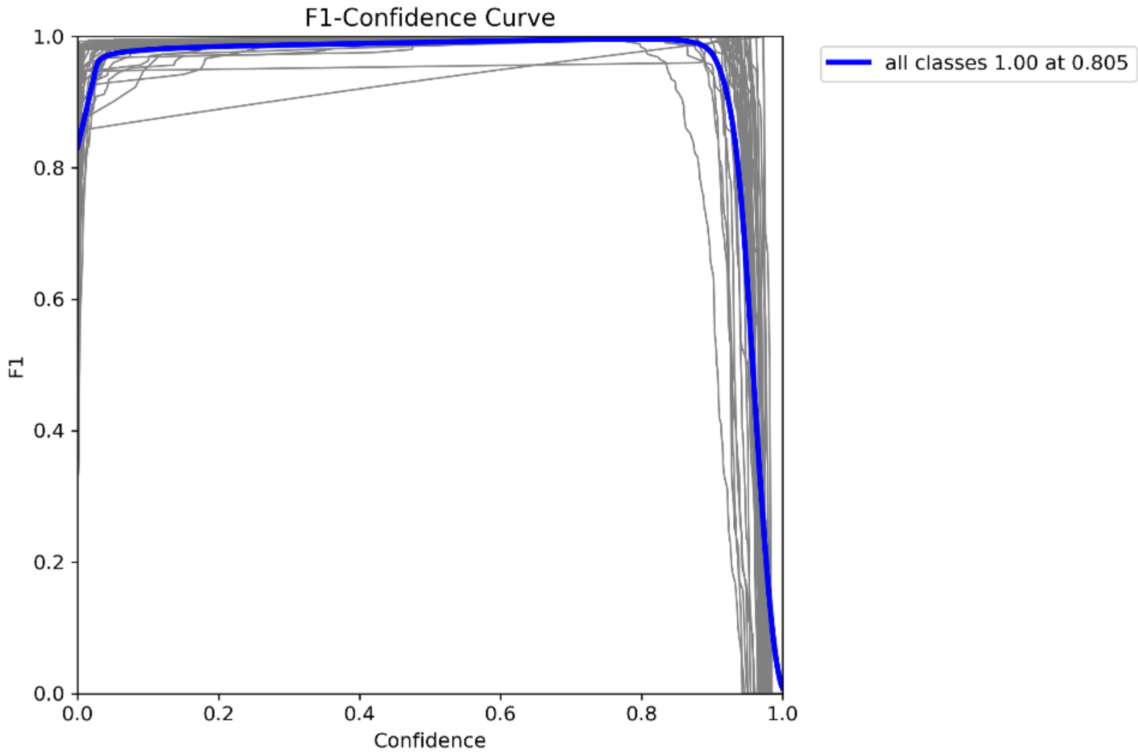


Figure 29: F1-Confidence curve

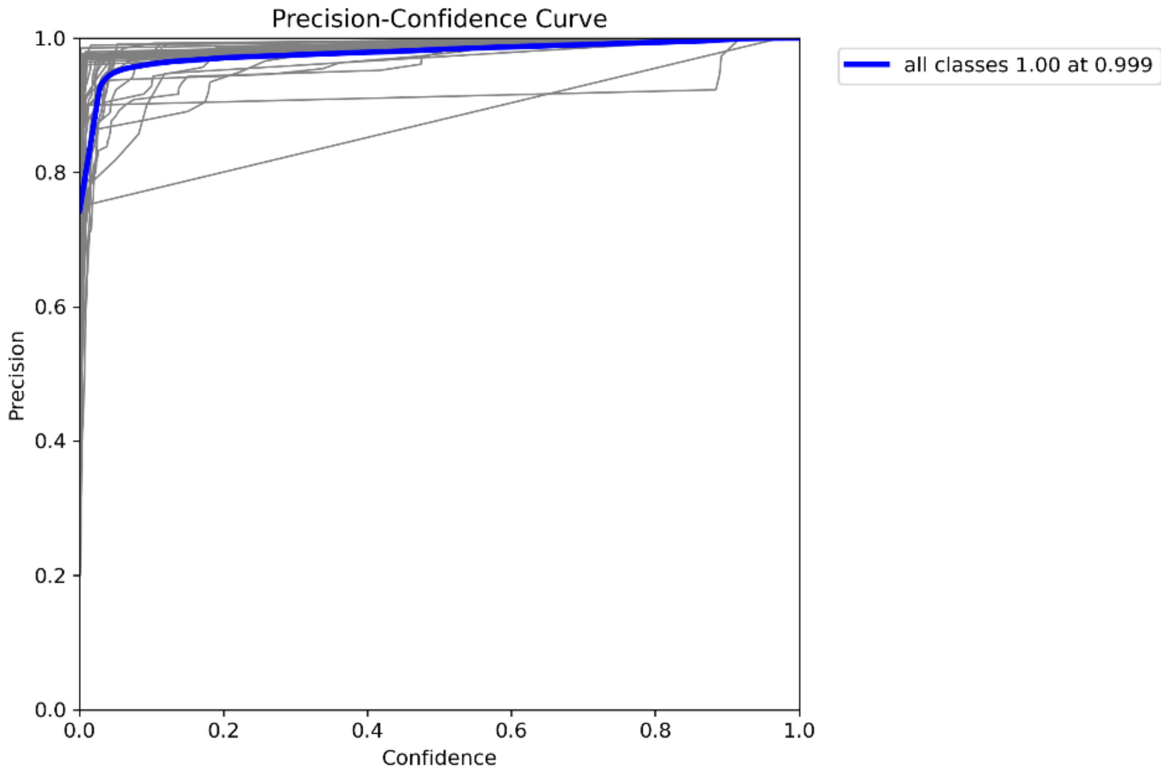


Figure 30: Precision-Confidence curve

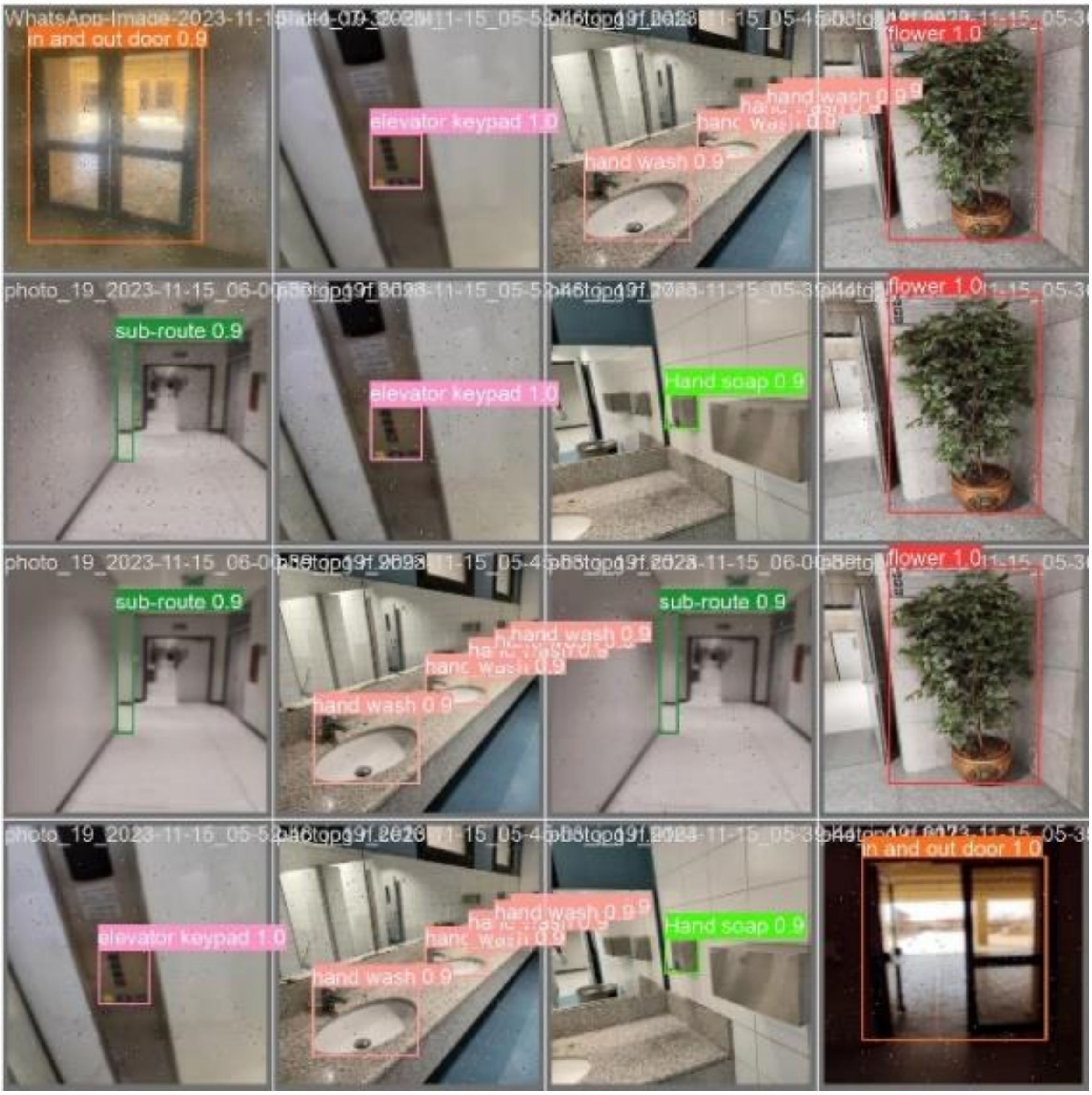


Figure 31: validation set prediction-1

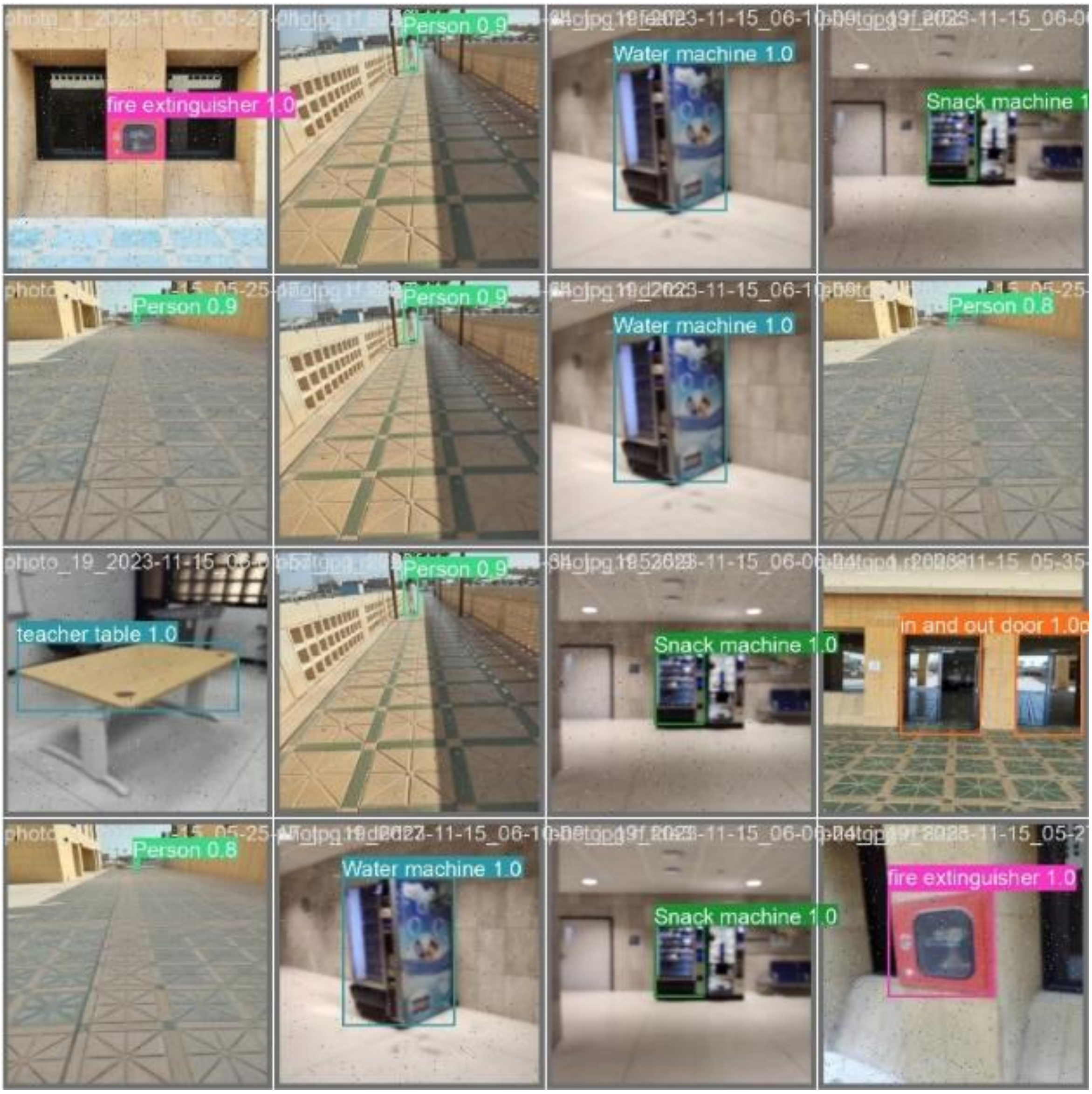


Figure 32: validation set prediction-2

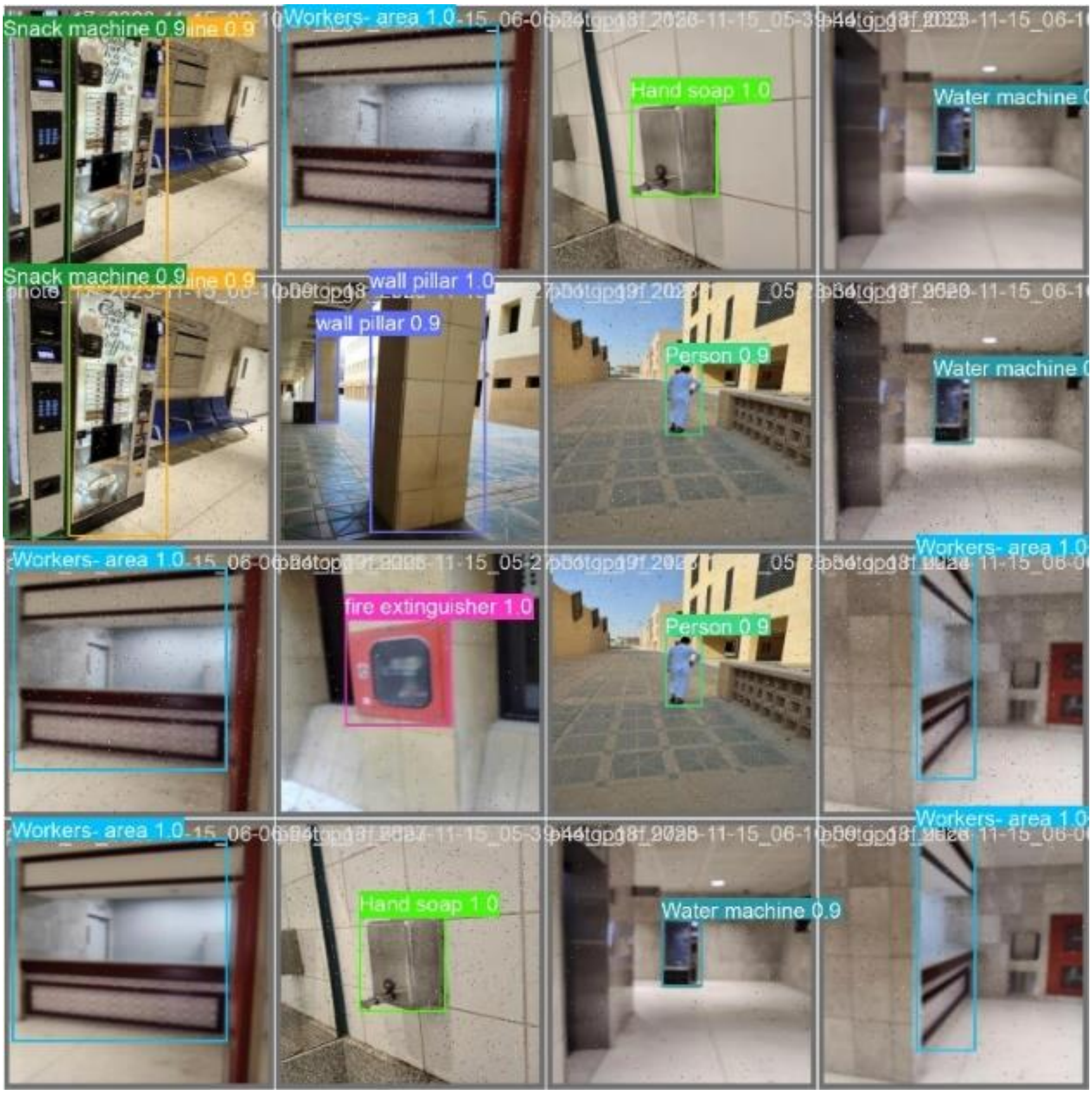


Figure 33: validation set prediction-3

ChatGPT:

We used this code for ChatGPT connection testing:

```
import openai
import time
client = openai.OpenAI(api_key="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")
assistant = client.beta.assistants.create(
    name="Explanatory",
    instructions="You are a an explanatory. You will help users to explain an object that are asked",
    tools=[{"type": "code_interpreter"}],
    model="gpt-4-1106-preview",
)
thread = client.beta.threads.create()
message = client.beta.threads.messages.create(
    thread_id=thread.id,
    role="user",
    content="What is a Car, explain it with a maximal of 2 sentence",
)
run = client.beta.threads.runs.create(
    thread_id=thread.id,
    assistant_id=assistant.id,
    instructions="Address this user as Mohammed.",
)
print("checking assistant status. ")
while True:
    run = client.beta.threads.runs.retrieve(thread_id=thread.id, run_id=run.id)
    if run.status == "completed":
        print("done!")
        messages = client.beta.threads.messages.list(thread_id=thread.id)
        print("messages: ")
        for message in messages:
            assert message.content[0].type == "text"
            if message.role == "assistant":
                print(message.content[0].text.value)
        client.beta.assistants.delete(assistant.id)
        break
    else:
        print("in progress...")
        time.sleep(5)
```

Database:

We used this code for Database connection Testing:

```
from whisper_mic import WhisperMic
import pyttsx3
import time
import pymongo
databasePin = []
tokenList = []
integer_number = [0,1,2,3,4,5,6,7,8,9]
string_number = ["zero","one","two","three","four","five","six","seven","eight","nine"]
client =
pymongo.MongoClient("mongodb+srv://moh:12345@mohammed.lvr6bpx.mongodb.net/
?retryWrites=true&w=majority")
database_names = client.list_database_names()
print(database_names)
db = client['GP']
collection = db["ob"]
# Find all documents in the collection
cursor = collection.find()
new_doc = {}
# Iterate through the cursor to print or process each document
for document in cursor:
    new_doc = document
print(new_doc)
last_key = list(new_doc.keys())[-1]
print(last_key)
last_key_number = last_key[-1]
print(last_key_number)
current_id = new_doc["_id"]
print(current_id)
def voice_notification(msg):
    engine = pyttsx3.init()
    newVoiceRate = 170
    engine.setProperty('rate',newVoiceRate)
    engine.say(msg)
    engine.runAndWait()
def speech_to_text(string):
    for item in string:
        if item in string_number:
            string[string.index(item)] = (integer_number[string_number.index(item)])
    return string
def remove_special_character(input_string, character_to_remove):
    return input_string.replace(character_to_remove, '')
def get_keys_by_value(dictionary, target_value):
    return [key for key, value in dictionary.items() if value == target_value]
def signup():
    global db
    global collection
    global tokenList
```

```
db = client['GP']
collection = db["ob"]
cursor = collection.find()
new_doc = {}
# Iterate through the cursor to print or process each document
for document in cursor:
    new_doc = document
mic = WhisperMic(english=True,model="tiny",energy=1000)
text = "Please register a 7 digit number"
voice_notification(text)
print("Please register a 7 digit number...")
while True:
    tokenList = []
    # text = mic.listen_loop(phrase_time_limit=5)
    mic.energy = 1500
    text = mic.listen(timeout=1)
    text = text.lower()
    if ("timeout" not in text):
        text = remove_special_character(text,".")
        text = remove_special_character(text,",")
        text = remove_special_character(text,"!")
        text = text.split()
        tokenList= speech_to_text(text)
        print(tokenList)
        if (len(tokenList) > 7):
            msg = "Input is higher than 7 digit, please try again"
            voice_notification(msg)
        elif (len(tokenList) < 7):
            msg = "Input is lower than 7 digit, please try again"
            voice_notification(msg)
        else:
            loginToken = "".join(tokenList)
            id = get_keys_by_value(new_doc, loginToken)
            if len(id) == 0:
                msg = "Account register succesfull, please say 0 for login and 1 for another
register"
                last_key = list(new_doc.keys())[-1]
                last_key_number = int(last_key[-1]) +1
                filter_criteria = {"_id": current_id}
                # Define the update operation (set a new value for the "age" field)
                dictKeys = "pin" + str(last_key_number)
                new_doc[dictKeys] = loginToken
                update_operation = {"$set": new_doc}
                result = collection.update_one(filter_criteria, update_operation)
                cursor = collection.find()
                for document in cursor:
                    new_doc = document
                print(new_doc)
                voice_notification(msg)
```

```
                    break
                else:
                    msg = "Account already exists, please try again"
                    voice_notification(msg)
def login():
    global db
    global collection
    global tokenList
    db = client['GP']
    collection = db["ob"]
    cursor = collection.find()
    new_doc = {}
    # Iterate through the cursor to print or process each document
    for document in cursor:
        new_doc = document
    mic = WhisperMic(english=True,model="tiny",energy=1000)
    text = "Please enter a 7 digit number"
    voice_notification(text)
    print("Please enter a 7 digit number...")
    while True:
        tokenList = []
        # text = mic.listen_loop(phrase_time_limit=5)
        mic.energy = 1500
        text = mic.listen(timeout=1)
        text = text.lower()
        if ("timeout" not in text):
            text = remove_special_character(text,".")
            text = remove_special_character(text,",")
            text = remove_special_character(text,"!")
            text = text.split()
            tokenList= speech_to_text(text)
            if (len(tokenList) > 7):
                msg = "Input is higher than 7 digit, please try again"
                voice_notification(msg)
            elif (len(tokenList) < 7):
                msg = "Input is lower than 7 digit, please try again"
                voice_notification(msg)
            else:
                loginToken = "".join(tokenList)
                id = get_keys_by_value(new_doc, loginToken)
                if len(id) != 0:
                    msg = "Login Succesful, Welcome"
                    id = id[0]
                    print(id)
                    voice_notification(msg)
                else:
                    msg = "Account not found, please try again"
                    voice_notification(msg)
def mic():
```

```
    mic = WhisperMic(english=True,model="tiny",energy=1000)
    time.sleep(2)
    text = "Welcome, please say 0 for Login and 1 for SignUp"
    voice_notification(text)
    while True:
        print("Please speak something...")
        # text = mic.listen_loop(phrase_time_limit=5)
        mic.energy = 1500
        text = mic.listen(timeout=1)
        text = text.lower()
        if ("timeout" not in text):
            text = remove_special_character(text,".")
            text = remove_special_character(text," ")
            text = remove_special_character(text,",")
            text = remove_special_character(text,"!")
            print(text)
            print(len(text))
            if (text == "zero" or text == "0"):
                login()
            elif (text == "one" or text == "1"):
                signup()
            else:
                msg = "is not a command"
                voice_notification(str(text + msg))
        mic()
```

Distance and Direction detection:

We used this code for Distance and Direction detection Testing:

```
import speech_recognition as sr
import cv2
import numpy as np
from ultralytics import YOLO
import pyttsx3
import math
class_names = ["person", "bicycle", "car", "motorbike", "aeroplane", "bus", "train",
"truck", "boat",
              "traffic light", "fire hydrant", "stop sign", "parking meter", "bench", "bird", "cat",
              "dog", "horse", "sheep", "cow", "elephant", "bear", "zebra", "giraffe",
"backpack", "umbrella",
              "handbag", "tie", "suitcase", "frisbee", "skis", "snowboard", "sports ball", "kite",
"baseball bat",
              "baseball glove", "skateboard", "surfboard", "tennis racket", "bottle", "wine
glass", "cup",
              "fork", "knife", "spoon", "bowl", "banana", "apple", "sandwich", "orange",
"broccoli",
              "carrot", "hot dog", "pizza", "donut", "cake", "chair", "sofa", "pottedplant",
"bed",
              "diningtable", "toilet", "tvmonitor", "laptop", "mouse", "remote", "keyboard",
"telephone",
```

```
              "microwave", "oven", "toaster", "sink", "refrigerator", "book", "clock", "vase",
"scissors",
              "teddy bear", "hair drier", "toothbrush"]
object_dimensions = {
    "bird" : "0.10",
    "cat" : "0.45",
    "backpack" : "0.55",
    "umbrella" : "0.50",
    "bottle" : "0.20",
    "wine glass" : "0.25",
    "cup" : "0.15",
    "fork" : "0.15",
    "knife" : "0.25",
    "spoon" : "0.15",
    "banana" : "0.20",
    "apple" : "0.07",
    "sandwich" : "0.20",
    "orange" : "0.08",
    "chair" : "0.50",
    "laptop" : "0.40",
    "mouse" : "0.10",
    "remote" : "0.20",
    "keyboard" : "0.30",
    "phone" : "0.15",
    "book" : "0.18",
    "toothbrush" : "0.16"
}
def direction_x(x_dir,frame_width):
    if x_dir < (frame_width/7 * 1 + frame_width/15):
        direction = "9 o'clock"
    elif x_dir >= frame_width/7 * 1 + frame_width/15 and x_dir < (frame_width/7 * 2 +
frame_width/15):
        direction = "10 o'clock"
    elif x_dir >= frame_width/7 * 2 + frame_width/15 and x_dir < (frame_width/7 * 3 +
frame_width/15):
        direction = "11 o'clock"
    elif x_dir >= frame_width/7 * 3 + frame_width/15 and x_dir < (frame_width/7 * 4 +
frame_width/15):
        direction = "12 o'clock"
    elif x_dir >= frame_width/7 * 4 + frame_width/15 and x_dir < (frame_width/7 * 5 +
frame_width/15):
        direction = "1 o'clock"
    elif x_dir >= frame_width/7 * 5 + frame_width/15 and x_dir < (frame_width/7 * 6 +
frame_width/15):
        direction = "2 o'clock"
    elif x_dir >= frame_width/7 * 6 + frame_width/15 and x_dir < frame_width/7:
        direction = "3 o'clock"
    else:
        direction = "None"
```

```
    return direction
def Vector_x(x1, x2, frame_width):
    obj_center_x = (x1 + x2) // 2
    camera_middle_x = frame_width // 2
    vector_x = obj_center_x - camera_middle_x
    return vector_x
def Vector_y(y1, y2, frame_height):
    obj_center_y = (y1 + y2) // 2
    camera_middle_y = frame_height // 2
    vector_y = obj_center_y - camera_middle_y
    return vector_y
real_width = 0.15
model = YOLO("yolov8n.pt")
clock_hand = [9,10,11,12,1,2,3]
def camera_setting(img):
    results = model.predict(img, stream=True)
    frame_height, frame_width, channels = img.shape
    center_x = int(frame_width // 2)
    center_y = int(frame_height // 2)
    radius = min(center_x, center_y) - 30
    for i in range(0, 7):
        angle = math.radians(30 *i)
        x = int(frame_width/15 + frame_width/7 * i)   # Frame Width / 15 = Offset of
starting clock position
        y = int(frame_height - radius * math.sin(angle))
        if i % 3 == 0:
            thickness = 3
        else:
            thickness = 2
        font = cv2.FONT_HERSHEY_SIMPLEX
        cv2.putText(img, str(clock_hand[i]), (x, y ), font, 0.5, (255, 0, 0), thickness)
    for r in results:
        boxes = r.boxes
        for box in boxes:
            x1, y1, x2, y2 = box.xyxy[0]
            x1, y1, x2, y2 = int(x1), int(y1), int(x2), int(y2)
            cls = int(box.cls)
            if class_names[cls].lower() == "elephant":
                real_width = float(0.55)
                # real_width = float(object_dimensions["elephant"])
                camera_width = x2 - x1
                camera_center = (x2-x1)/2 + x1
                distance = (real_width * frame_width) / camera_width
                angle_deg = math.degrees(math.atan2(Vector_y(y1, y2, frame_height),
Vector_x(x1, x2, frame_width)))
                print(camera_center)
                cv2.putText(img, direction_x(camera_center, frame_width), (10, 30),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)
```

```
            cv2.putText(img, "Distance: {:.2f} meters".format(distance), (x1, y1 - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 2)
            cv2.rectangle(img, (x1, y1), (x2, y2), (255, 0, 255), 3)
def camera_parser():
    image_path = 'Elephant.jpg'  # Replace with the actual path to your image
    # Read the image
    img = cv2.imread(image_path)
    camera_setting(img)
    # Check if the image is successfully loaded
    if img is not None:
        # Display the image
        cv2.imshow('Image', img)
        cv2.waitKey(0)
        cv2.destroyAllWindows()
    else:
        print(f"Error: Unable to load the image from {image_path}")
if __name__ == "__main__":
    # while True:
    camera_parser()
```

YOLOv8 Model for object detection:
We used this code for YOLOv8 Model for object detection Testing:

```
import cv2
import numpy as np
import torch
import uuid
import os
import time
from ultralytics import YOLO
model = YOLO('best.pt')
classes = ['Bathroom door', 'Bathroom entrance', 'Clothes hanger', 'Coffee machine', 'Food
machine', 'Hand soap', 'PC desktop', 'Person', 'Snack machine', 'Tissue box', 'Water
machine', 'Workers- area', 'break chair', 'chair', 'class chair', 'class door', 'down ladder',
'elevator', 'elevator keypad', 'fire extinguisher', 'flower', 'hand wash', 'in and out door',
'ladder door', 'light buttons', 'mosque', 'paper trash can', 'person', 'sub-route', 'teacher
podium', 'teacher table', 'trash can', 'up ladder', 'wall pillar', 'water faucet', 'white board']
result = model(source=0, show = True, conf=0.4,save= True)
```

Whisper(speech recognition):
We used this code for Whisper(speech recognition) Testing:

```
import openai
import time
from whisper_mic import WhisperMic
mic = WhisperMic(english=True,model="tiny")
print("test")
result = mic.listen()
print(result)
```

5.2.2 Integration Testing

During the integration process of Whisper (speech recognition) with object detection model, thorough testing of components was conducted to ensure seamless functionality. This involved validating the integration points between Whisper's speech recognition module and the model object detection system. Tests were designed to verify that Whisper accurately transcribed spoken commands and that these transcriptions effectively triggered the functionality of various commands. Various scenarios were tested to assess the robustness of the integration, including different environmental conditions, varying accents, and levels of background noise.

Additionally, the code for Distance And Direction with Database connection and ChatGPT connection was integrated and tested to ensure smooth communication between these components. This involved establishing connections to a database to store and retrieve user's PIN number, as well as integrating ChatGPT as an AI assistant.
Testing focused on validating the accuracy of distance and direction calculations, verifying the reliability of database interactions, and assessing the effectiveness of ChatGPT in understanding and responding to user queries related to distances and directions.
Throughout the integration process, testing methodologies such as unit testing employed to identify and address any issues or inconsistencies in each unit. The goal was to create a seamlessly integrated system where Whisper, model object detection, Distance And Direction functionality, and ChatGPT connection work harmoniously to provide users with a comprehensive and intuitive experience.

After integrating ChatGPT into our program, we encountered one notable issue when users posed questions to ChatGPT, causing significant lag and ultimately leading to the program shutting down unexpectedly.
we identified the factor contributing to this issue, the volume of incoming requests to system could have overwhelmed the system, causing it to struggle in handling concurrent interactions.
To solve this issue, we undertook several steps. This included optimizing the codebase to improve efficiency, implementing caching mechanisms to reduce redundant queries to ChatGPT, and scaling up the infrastructure to better handle peak loads.
Additionally, we conducted thorough performance testing and profiling to identify any bottlenecks or areas for optimization within the system.
through a combination of code optimizations, enhancements, and performance tuning, we were able to mitigate the lag and instability issues experienced after querying ChatGPT, ensuring a smoother and more reliable user experience.

5.2.3 User Acceptance Testing

Our friend Majid Alanazi tested the system and provided feedback on various aspects.
He expressed satisfaction with the Whisper and ChatGPT function as AI assistant, finding them to be effective assets. Regarding the object detection model, he was pleased with its performance but suggested expanding its scope beyond the college of computer and information sciences to cover all areas. Additionally, he noted that the distance measurement accuracy was not sufficient and recommended integrating sensors into the project to enhance accuracy. Despite these suggestions for improvement, Majid expressed overall satisfaction with the system. However, he mentioned that limitations of the Raspberry Pi were affecting the user experience, indicating a need for improvement in this area.

5.2.4 Test Cases

	Scenario	Expected Outcome	Actual result
Whisper	Speaks in a noisy environment with background	Whisper Expected to clear and filter the noise and get clear answers	As Expected
	Speaks with a heavy Arabic accent	Whisper effectively recognizes despite the accent.	As we tested it many several times on multiple users it recognizes effectively without any problems

Table 3: Whisper test cases

	Scenario	Expected Outcome	Actual result
YOLOv8 model for object detection	mage is blurry or low resolution	Model effectively handles the low-quality images and because of the trained data	It detected everything with a high confidence score
	multiple overlapping objects	Model effectively detects and identifies each individual object, providing separate outputs for each	It detect the first object that the camera sees without any problem

Table 4: YOLOv8 model for object detection test cases

	Scenario	Expected Outcome	Actual result
Distance And Direction detection	Valid coordinates for Distance detection	Expected exact Distance between the user and the object	It doesn’t detect the exact Distance effectively, there’s a need for sensors
	Valid coordinates for Direction detection	Expected exact Direction of the object	As expected, Directions were on point

Table 5: Distance And Direction detection test cases

	Scenario	Expected Outcome	Actual result
Database interactive	New User Registration	To save the new user PIN correctly	As expected
	Sign in	To sign in the user by his PIN correctly	As expected

Table 6: Database interactive test cases

	Scenario	Expected Outcome	Actual result
ChatGPT interactive	User asks a common question	ChatGPT responses with the right answer	As expected
	Multiple users simultaneously interact with ChatGPT	manages Multiple interactions effectively	The question gets miss understood with multiple people talking

Table 7: ChatGPT interactive test cases

5.3 Discussion

we faced some limitations based on the processing power of the raspberry pi 4, and we faced some issues regarding the some outdated codes like ssd mobilenet model it should be converted to a graph after the training and the conversion to graph is outdated by TensorFlow 2 so we used Yolov8 model instead, Which provide us with the ease of use for training and integrating also it provided us with great results which tested on F1 confidence curve ~0.80 score, and ~0.99 on the precision-confidence curve,
in real life it can detect the objects correctly.
we also faced some issues with speech-recognition library it cannot recognize voice correctly because of failed internet connection that happens with google so we change it to whisper_mic it worked better than we imagine and it gave us great testing results on a noisy background and with a heavy Arabic accent it worked almost flowless.

5.4 Summary

The integration testing phase of our project, focused on merging Whisper's speech recognition with the object detection model, uncovered critical insights and challenges. One notable hurdle arose from the limitations of the Raspberry Pi 4's processing power, prompting us to transition from an outdated SSD MobileNet model to the more efficient YOLOv8 model. Despite initial setbacks with outdated code and incompatible libraries, the adoption of YOLOv8 not only facilitated smoother integration but also yielded exceptional results, boasting F1 confidence curve scores nearing 0.80 and precision-confidence curves around 0.99. In real-world scenarios, the model demonstrated remarkable accuracy in detecting various objects, underscoring its efficacy in enhancing user experience.
Another significant challenge emerged from the limitations of the speech recognition library, exacerbated by unreliable internet connections affecting Google services. However, our adaptation to the whisper_mic library proved to be a game-changer, exceeding expectations in its ability to accurately transcribe speech even amidst noisy environments and heavy accents like Arabic. This unexpected success not only resolved our immediate issue but also highlighted the importance of adaptable solutions in overcoming unforeseen challenges. Ultimately, the integration testing phase not only validated the effectiveness of our chosen models and libraries but also underscored the critical role of adaptability and innovation in navigating obstacles encountered during the development process.

Chapter Six: Conclusion and Future Work

6 Conclusion and Future Work

6.1 Conclusion

The project, "Guiding Glasses for Blind People Using Computer Vision and ChatGPT," aims to address the challenges faced by visually impaired individuals in achieving independence and mobility. By leveraging computer vision technology and advanced AI systems, this initiative aims to create innovative assistive glasses. These glasses serve as a visual guide, providing real-time object detection, distance measurements, without the need for a traditional cane. Additionally, integration with ChatGPT enables users to asks for information on-demand.

The project faced several challenges, including developing an object detection algorithm capable of identifying various objects in real-time, implementing a classification system to categorize detected objects into relevant classes, integrating measurement estimation algorithms to provide accurate dimensions and distances of detected objects, and ensuring the system can adapt to varying lighting conditions. However, through the use of existing libraries and frameworks for computer vision, such as OpenCV and TensorFlow, as well as the seamless integration of ChatGPT's API, these challenges were successfully overcome.

The results of the project demonstrate the potential of computer vision and AI systems in enhancing the lives of visually impaired individuals. The glasses have proven to be an effective tool for enhancing spatial awareness and object recognition, thus offering a versatile solution that improves the daily lives of the visually impaired. The project's success also highlights the importance of seamless integration of multiple technologies into a wearable device, ultimately enhancing the quality of life and independence of visually impaired individuals.

Overall, this project represents a significant advancement in the field of assistive technology, offering a comprehensive solution that combines computer vision, object detection, navigation assistance, ChatGPT and AI. The innovation lies in creating a comprehensive and user-friendly solution that addresses the multifaceted challenges faced by visually impaired individuals, ultimately improving their quality of life and independence.

6.2 Future Work

Future work for the project "Guiding Glasses for Blind People Using Computer Vision and ChatGPT" could involve several key areas of expansion:

1. Expanding the Project Scope: Currently, the project is limited to the College of Computer and Information Sciences. To broaden its reach, the project could be expanded to include all places. This could include incorporating features such as voice guidance, haptic feedback, and compatibility with mobile devices.

2. Adding Sensors for More Accurate Measurements: The glasses currently rely on computer vision and AI for object detection and measurement. Adding sensors to the system could potentially provide more accurate measurements. For instance, ultrasonic sensors could be used to measure distances, while accelerometers could be used to detect changes in the user's orientation.

3. Improving Pathfinding: The glasses could be equipped with a feature that suggests the best path for the user to walk. This could involve using machine learning algorithms to analyze the user's surroundings and suggest the least obstructed path. Additionally, the glasses could be integrated with GPS technology to provide real-time navigation assistance.

4. Text Detection and Reading: Another area for future development could be the ability to detect and read text. This could involve using optical character recognition (OCR) technologies to identify text in the user's surroundings and read it aloud. This would be particularly useful for visually impaired individuals who rely on printed materials for information.

5. Manufacturing the Glasses: Once the design and functionality of the glasses have been finalized, the next step would be to manufacture them. This could involve partnering with a manufacturer experienced in creating wearable devices, and working closely with them throughout the manufacturing process to ensure the glasses meet the desired standards.

Each of these areas represents a significant opportunity for further development and improvement of the "Guiding Glasses" project. By addressing these areas, the project could become an even more powerful tool for visually impaired individuals, helping them navigate the world with greater ease and confidence.

References

[1] P.Devaki, S.Shivavarsha, G.Bala Kowsalya, M.Manjupavithraa, E.A. Vima "Real-Time Object Detection using Deep Learning and Open CV" in IJITEE ISSN: 2278-3075, Volume-8 Issue-12S, October 2019

[2] L. Ruotsalainen, A. Morrison, M. Mäkelä, J. Rantanen and N. Sokolova, "Improving Computer Vision-Based Perception for Collaborative Indoor Navigation," in IEEE Sensors Journal, vol. 22, no. 6, pp. 4816-4826, 15 March 15, 2022, doi: 10.1109/JSEN.2021.3106257.

[3] Zhong-Qiu Zhao, Member, IEEE, Peng Zheng, Shou-tao Xu, and Xindong Wu "Object Detection with Deep Learning: A Review" in arXiv:1807.05511v2

[4] A. C. Özgen, M. Fasounaki, and H. K. Ekenel, "Text detection in natural and computer-generated images," 2018 26th Signal Processing and Communications Applications Conference (SIU), Izmir, Turkey, 2018, pp. 1-4, doi: 10.1109/SIU.2018.8404600.

[5] E. DANDIL and K. K. ÇEVİK, "Computer Vision Based Distance Measurement System using Stereo Camera View," 2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), Ankara, Turkey, 2019, pp. 1-4, doi: 10.1109/ISMSIT.2019.8932817.

Appendix

A. Code Snippets

direction_x(x_dir,frame_width) function:

```
# Function to encode the direction
def direction_x(x_dir,frame_width):
## Take the center of the detected object pixel, and the total width of the
image, then divide it into 7 segments ##
    if x_dir < (frame_width/7 * 1 + frame_width/15):    # 1st segment, 9
o'clock
        direction = "9 o'clock"
    elif x_dir >= frame_width/7 * 1 + frame_width/15 and x_dir <
(frame_width/7 * 2 + frame_width/15):  # 2nd segment, 10 o'clock
        direction = "10 o'clock"
    elif x_dir >= frame_width/7 * 2 + frame_width/15 and x_dir <
(frame_width/7 * 3 + frame_width/15):  # 3rd segment, 11 o'clock
        direction = "11 o'clock"
    elif x_dir >= frame_width/7 * 3 + frame_width/15 and x_dir <
(frame_width/7 * 4 + frame_width/15):  # 4th segment, 12 o'clock
        direction = "12 o'clock"
    elif x_dir >= frame_width/7 * 4 + frame_width/15 and x_dir <
(frame_width/7 * 5 + frame_width/15):  # 5th segment, 1 o'clock
        direction = "1 o'clock"
    elif x_dir >= frame_width/7 * 5 + frame_width/15 and x_dir <
(frame_width/7 * 6 + frame_width/15):  # 6th segment, 2 o'clock
        direction = "2 o'clock"
    elif x_dir >= frame_width/7 * 6 + frame_width/15 and x_dir <
frame_width/7:  # 6th segment, 3 o'clock
        direction = "3 o'clock"
    else:
        direction = "None"
    return direction
```

search_engine(word) function:

```
# Function to integrate and search with ChatGPT
def search_engine(word):
    client = openai.OpenAI(api_key="sk-
hNqpxtZOodpdNCRS7BiBT3BlbkFJmVM5bK4scuknoXChUSXJ")
    content = "What is a " + str(word) + ", explain it with a simple
sentence and a maximal of 2 sentences"
    assistant = client.beta.assistants.create(
        name="Explanatory",
        instructions="You are a an explanatory. You will help users to
explain an object that are asked",
        tools=[{"type": "code_interpreter"}],
        model="gpt-4-1106-preview",
    )

    thread = client.beta.threads.create()

    message = client.beta.threads.messages.create(
        thread_id=thread.id,
        role="user",
        content=content,
    )
    run = client.beta.threads.runs.create(
        thread_id=thread.id,
        assistant_id=assistant.id,
        instructions="Address this user as Mohammed.",
    )
    print("checking assistant status. ")
    print("in progress...", end="")
    while True:
        run = client.beta.threads.runs.retrieve(thread_id=thread.id,
run_id=run.id)
        if run.status == "completed":
            print("done!")
            messages =
client.beta.threads.messages.list(thread_id=thread.id)
            print("messages: ")
            for message in messages:
                assert message.content[0].type == "text"
                if message.role == "assistant":
                    return message.content[0].text.value

            client.beta.assistants.delete(assistant.id)
            return None
        else:
            print(".", end="")
```

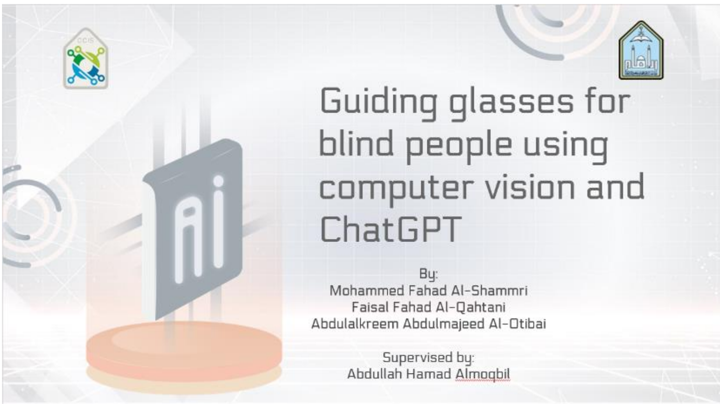
video_to_images(video_path) function:

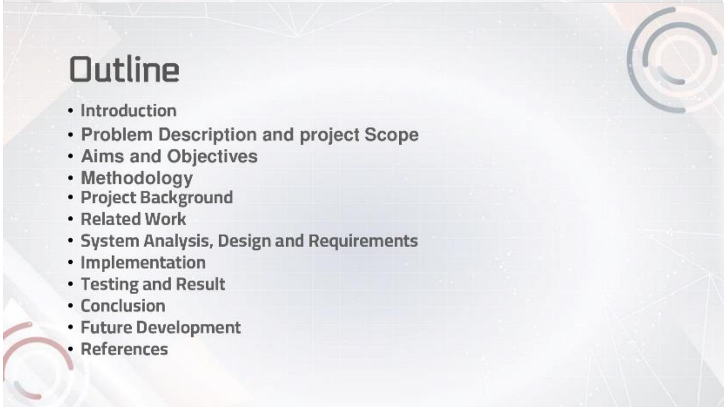
```
def video_to_images(video_path):
    cap = cv2.VideoCapture(video_path)
    global current_object_on_vision
    global object_on_vision_counter
    global object_on_vision
    global distance
    global direction
    if not cap.isOpened():
        print(f"Error: Could not open the video file {video_path}")
        return
    while True:
        # Read a frame from the video stream
        ret, frame = cap.read()
        results = model.predict(frame, stream=False, verbose=False)
        _, frame_width, _ = frame.shape
        # Reset the current object on vision
        current_object_on_vision = []
        for r in results:   # Iterate every detected object
            for box in r.boxes:
                x1, y1, x2, y2 = box.xyxy[0]    # Take the x,y coordinates
                x1, y1, x2, y2 = int(x1), int(y1), int(x2), int(y2)
                cls = int(box.cls)
                current_object_on_vision.append(cls)    # Add the detected
object into the current_object array
                real_width = class_width[cls]
                camera_width = x2 - x1
                camera_center = (x2-x1)/2 + x1
                direction = direction_x(camera_center, frame_width)     #
Check the direction of the object from the camera
                distance = (real_width * frame_width) / camera_width    #
Check the distance of the object from the camera
                mid_x = x1 + (x2-x1)//2                                 #
Take the middle coordinates
                distance_str = "{:.2f} meters".format(distance)
                cv2.putText(frame, class_names[cls], (x1, y1 - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 2)  # Display the name of the
object
                cv2.putText(frame, direction, (x2, y1 - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 0), 2)   # Display the direction
of the object
                cv2.putText(frame, distance_str, (mid_x, y1 - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 255), 2)   # Display the distance of
the object
                cv2.rectangle(frame, (x1, y1), (x2, y2), (255, 0, 255),
3)     # Put the rectangle on the detected object
            if not ret:
```

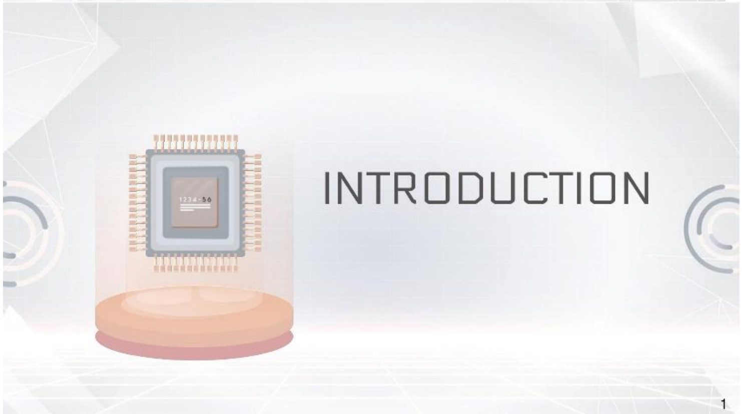
```python
                break
        # Clear memory for the next iteration
        K.clear_session()
        gc.collect()

        cv2.imshow('Video Stream', frame)
        # Check for the 'q' key to exit the loop
        if cv2.waitKey(25) & 0xFF == ord('q'):
            break
    # Release the video capture object
    cap.release()
    cv2.destroyAllWindows()
```

B. Presentation Slides







Problem Description and project Scope

- individuals with visual impairments face challenges in interacting with their environment
- Traditional assistive devices often fall short in providing real-time information.
- College of Computer and Information Sciences in IMAMU

3

Aims and Objectives

- primary goal is easing the live of visually impaired
- collecting pictures for object detection model
- developing object distance detection
- developing object direction detection.
- Integration with ChatGPT
- Assembling prototype

4

Methodology

- Agile methodology used for developing the project
- advanced technologies like OpenCV, MongoDB, ChatGPT AI assistance.

5

Project Background

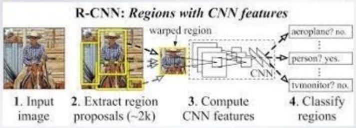
- The history of AI
- Main concepts in AI
- The importance of guiding glasses in the AI field

6

Related Work

▪ Real-Time Object Detection Using Deep Learning and Open CV

HOG



R-CNN

7

Related Work

▪ Improving Computer Vision-Based Perception for Collaborative Indoor Navigation
▪ Object Detection with Deep Learning: A Review
▪ Computer Vision-Based Distance Measurement System Using Stereo Camera View

8

Related Work

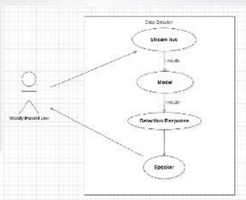
▪ Text Detection in Natural and Computer-Generated Images

9

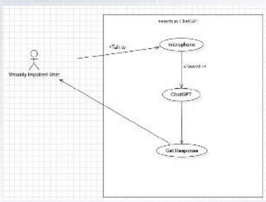
System Analysis, Design and Requirements

▪ System Perspective
▪ Requirements Elicitation Techniques
▪ Functional and Non-Functional Requirements

Use cases:



Object detection

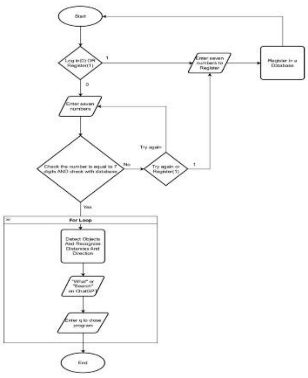


Search in ChatGPT

10

System Analysis, Design and Requirement

- Architectural Design



11

Implemention

- Python leads for project.
- Python libraries.
- MongoDB as a database.
- Software environments.

hardware ensemble:

12

Implemention

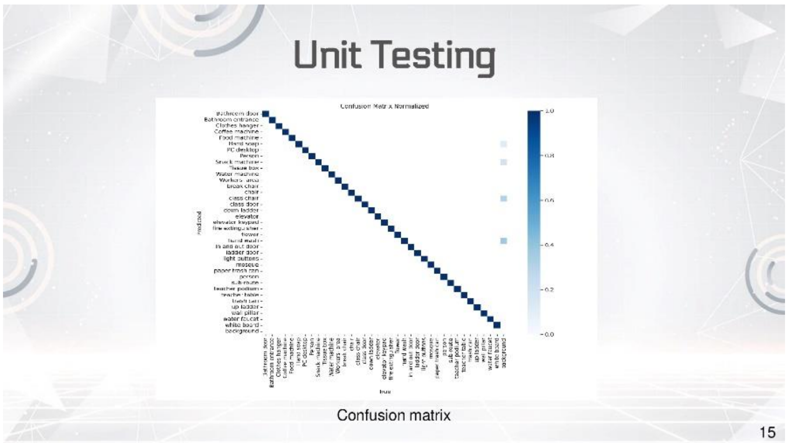
- Important functions

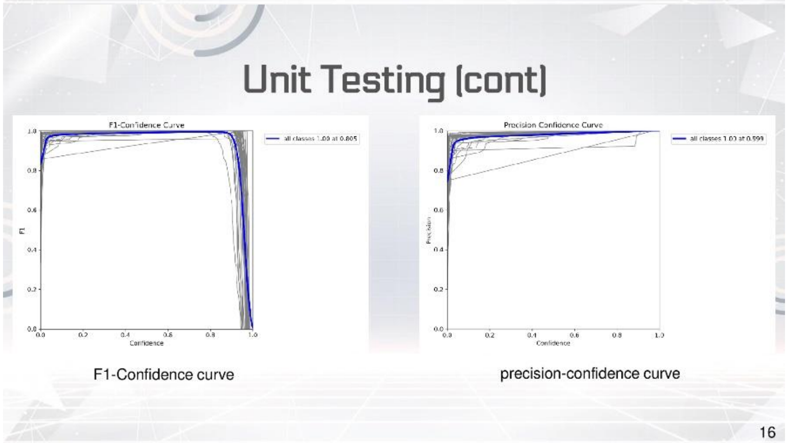
- direction_x(x_dir, frame_width)
- search_engine(word)
- video_to_images(video_path)

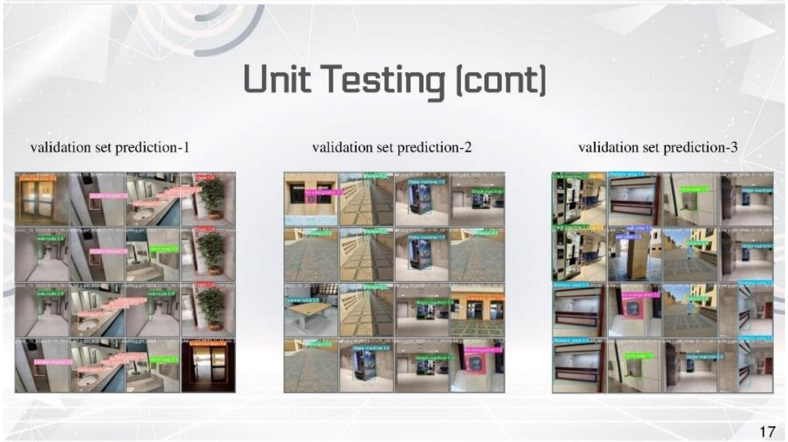
13

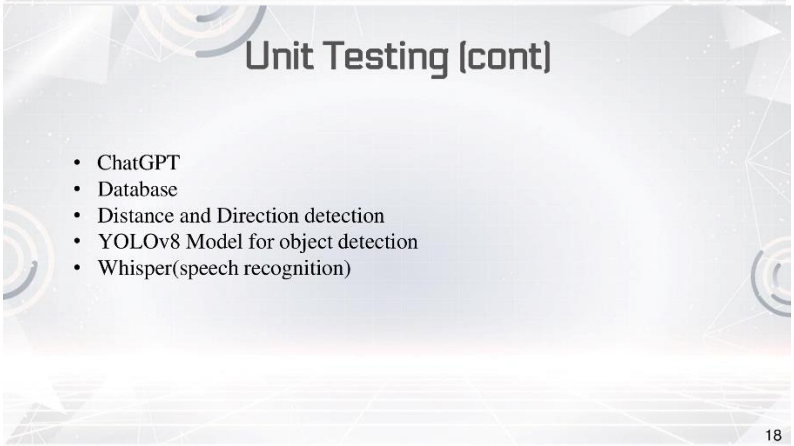
Testing and Result

14


Unit Testing
Confusion matrix
15


Unit Testing (cont)
F1-Confidence Curve
Precision-Confidence Curve
F1-Confidence curve
precision-confidence curve
16


Unit Testing (cont)
validation set prediction-1
validation set prediction-2
validation set prediction-3
17


Unit Testing (cont)
• ChatGPT
• Database
• Distance and Direction detection
• YOLOv8 Model for object detection
• Whisper(speech recognition)
18

Integration Testing

- Whisper Integration
- Distance & Direction, ChatGPT Connection
- Challenge & Solution
- Outcome

19

Conclusion

Enhancing the daily lives of visually impaired individuals.

AI to help humans.

20

Future Development

- Expand Project Scope.
- Sensor Integration
- Pathfinding Enhancement
- Text Detection and Reading
- Manufacturing Phase

21

References

[1] P.Devaki, S.Shivavarsha, G.Bala Kowsalya, M.Manjupavithraa, E.A. Vima "Real-Time Object Detection using Deep Learning and Open CV" in IJITEE ISSN: 2278-3075, Volume-8 Issue-12S, October 2019

[2] L. Ruotsalainen, A. Morrison, M. Mäkelä, J. Rantanen and N. Sokolova, "Improving Computer Vision-BasedPerception for Collaborative Indoor Navigation," in IEEE Sensors Journal, vol. 22, no. 6, pp. 4816-4826, 15 March 15, 2022, doi: 10.1109/JSEN.2021.3106257.

[3] Zhong-Qiu Zhao, Member, IEEE, Peng Zheng, Shou-tao Xu, and Xindong Wu "Object Detection with Deep Learning: A Review" in arXiv:1807.05511v2

[4] A. C. Özgen, M. Fasounaki, and H. K. Ekenel, "Text detection in natural and computer-generated images," 2018 26th Signal Processing and Communications Applications Conference (SIU), Izmir, Turkey, 2018, pp. 1-4, doi: 10.1109/SIU.2018.8404600.

[5] E. DANDIL and K. K. ÇEVİK, "Computer Vision Based Distance Measurement System using Stereo Camera View," 2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), Ankara, Turkey, 2019, pp. 1-4, doi:10.1109/ISMSIT.2019.8932817.

22


Acknowledgment


THANK
YOU !

C. Miscellaneous

Questionnaire form:

https://forms.gle/mbnzwg58aWWKtNnm6

Figure 34: Questionnaire form design 1

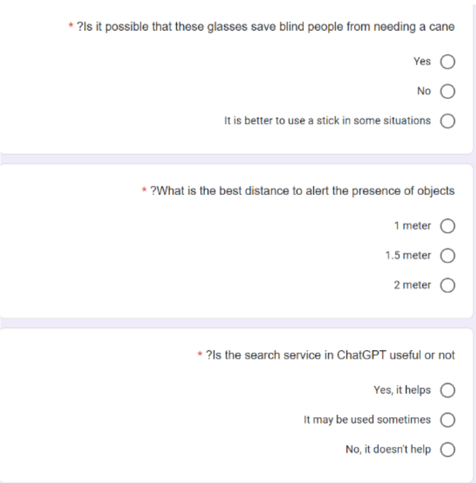


Figure 35: Questionnaire form design 2

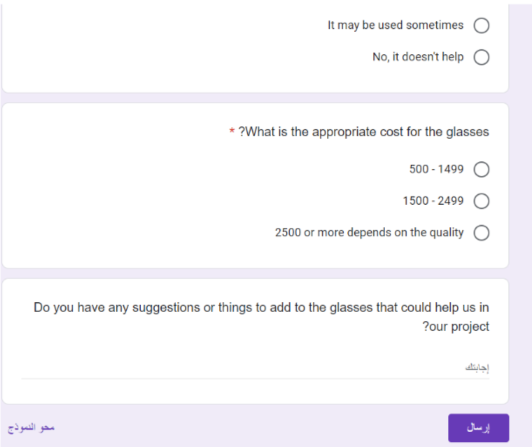


Figure 36: Questionnaire form design 3

D. Meetings

Week 1:

Discussed Topics and List of Accomplished Tasks
we then discussed some of the challenges we encountered during the project. These included: * Difficulties in finding a suitable dataset for the task, as the available data was either too small or not relevant to the problem domain. * Tuning the hyperparameters of the machine learning algorithm to achieve optimal performance. Conclusion: The discussion with supervisor was productive and insightful. we received valuable feedback and guidance on our progress with the ModelTrainingCode project. we are now confident that we can successfully complete the project and deliver a high-quality solution.

Week 2:

Discussed Topics and List of Accomplished Tasks
We Discussed various data sources, such as publicly available datasets, synthetic data generation techniques, and real-world data acquisition strategies, Discussed the challenges in optimizing object detection models, such as balancing precision and recall, handling complex backgrounds, and optimizing for real-time applications. The discussion concluded with a positive outlook on the project's potential to contribute to the field of object detection. The supervisor expressed enthusiasm for the proposed approach and provided valuable insights and suggestions for further refinement of the project plan.

Week 3:

Discussed Topics and List of Accomplished Tasks
Key Points Discussed: *Project Overview: we briefed Dr. Abdullah on our ongoing research project, DistanceAndDirectionTesting, which aims to develop a novel testing method for evaluating distance and direction perception in objects. *Technical Approach: We delved into the technical details of my approach. I explained the use of advanced sensors, computer vision algorithms, and machine learning techniques to accurately measure and assess the accuracy of distance and direction perception in objects. *Challenges and Solutions: We discussed the challenges I faced during the project, such as obtaining suitable testing environments, ensuring consistent lighting conditions. We brainstormed potential solutions and strategies to overcome these challenges and improve the robustness of the testing method.

Week 4:

Discussed Topics and List of Accomplished Tasks
The discussion with our supervisor was productive and informative, resulting in a shared understanding of the potential benefits and limitations of ChatGptTesting. We agreed on the importance of ongoing development and refinement to maximize the platform's impact in evaluating ChatGPT's performance and driving advancements in the field of natural language processing.

Week 5:

Discussed Topics and List of Accomplished Tasks
our supervisor acknowledged the potential benefits of WhisperTesting. We agreed to monitor its impact on test effectiveness and overall productivity. he discussion with our supervisor was productive, and we have a clear plan for implementing WhisperTesting, resulting in better software quality and efficient testing practices.

Week 6:

Discussed Topics and List of Accomplished Tasks
our discussion reinforced the critical role of DataTesting in ensuring data integrity and enabling data-driven decision-making. We recognized the need for a comprehensive and collaborative approach to DataTesting, involving various stakeholders and leveraging appropriate tools and techniques.

Week 7:

Discussed Topics and List of Accomplished Tasks
The discussion with our supervisor was productive and provided valuable insights into the potential benefits and challenges of incorporating whisper testing into our model training process. The constructive feedback and suggestions will help shape the implementation strategy and ensure a successful implementation of whisper testing, ultimately leading to improved model performance and reliability.

Week 8:

Discussed Topics and List of Accomplished Tasks
The discussion with our supervisor provided valuable insights into database testing challenges and effective strategies for addressing them. The emphasis on data security, leveraging automated tools, and fostering collaboration highlighted the importance of a comprehensive and collaborative approach to database testing.

Week 9:

Discussed Topics and List of Accomplished Tasks
- The discussion with our supervisor was productive and encouraging. - We aligned on the importance of exploring WhisperTesting with ModelObjectDetection and its potential impact on object detection and classification tasks. - We established a roadmap and timeline to initiate the project and achieve meaningful results.

Week 10:

Discussed Topics and List of Accomplished Tasks
- Final discussion and project delivery

E. User Manual

Overview

This project is a computer vision-based application aimed at assisting visually impaired individuals by detecting objects, measuring distances, and providing directional guidance. It also includes an integrated ChatGPT for answering questions via voice commands.

Prerequisites

Before starting, ensure you have the following installed:

- Python 3.8 or higher
- Ultralytics library (pip install ultralytics)
- OpenCV-Python library (pip install opencv-python)
- Pyttsx3 library (pip install pyttsx3)
- Pymongo library (pip install pymongo)
- Keras library (pip install keras)
- GC library (pip install gc-python-utils)

How to Run

1. Install all the prerequisites libraries.
2. Run the GP_Code.py program using the command line or by clicking the program.
3. Log in or sign up from the main menu. Choose 0 for login, 1 for signup.
4. After successful login, you will enter the detection program.
5. The program will automatically detect any object and announce it aloud using text-to-speech.
6. If you mention any keywords with what or search, it will automatically connect with ChatGPT to answer the question.

Key Features

- Object Detection: The application uses the YoloV8 model trained on 36 different classes to detect objects in the user's field of view.
- Distance and Direction Detection: The system provides real-time object detection, distance measurements, and directional guidance.
- Voice Assisted Command: Users can issue commands through voice, using Whisper-AI, to interact with the system.
- User Authentication: The system supports 7-digit pin user authentication for secure login.
- User Database: All users are stored in a MongoDB database for easy management.
- Integrated ChatGPT: The system integrates with ChatGPT to answer questions posed by the user via voice commands.

Usage Tips

- When interacting with the system, speak clearly and use simple language to ensure accurate recognition of your commands.
- Use keywords with what or search to engage with ChatGPT and get answers to your questions.
- Ensure the camera has a clear view of the area you wish to detect objects in for optimal performance.

Remember, this tool is designed to aid visually impaired individuals in their daily lives. It's important to use it responsibly and considerately.