

SRS REPORT

ON

**“SURROUND VIEW SYSTEM (IMAGE PROCESSING)”**

SUBMITTED BY

**SIB20 – GHUTUKADE AKASH BHAGWAN**

**SIB21 – GUDE HASRHWARDHAN SHAILESH**

**SIB27 – KAZI DANISH TANVIR**

**SIB39 – MUNGASE AASHISH VITTHAL**

GUIDED BY

Mr. Nitin Gulhane

DEPARTMENT OF INFORMATION TECHNOLOGY

INTERNATIONAL INSTITUTE OF INFORMATION

TECHNOLOGY, PUNE – 411057

SAVITRIBAI PHULE PUNE UNIVERSITY

2024-2025

**Acknowledgement**

With immense pleasure, we are presenting this project report on “The Surround View System (SVS)” as a part of the Project Based Learning in S.E. Information Technology at INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY, HINJEWADI, PUNE.

It gives us the privilege to complete this report under the valuable mentorship of **Mr. Nitin Gulhane.** His guidance, cooperation, and encouragement have made headway in the project.

We are extremely grateful to **Dr. Jyoti Surve** (H.O.D of Information Technology) and **Dr. Vaishali Patil**, Principal, Information Technology of Information Technology for providing all facilities and help for the smooth progress of the report work. We would also like to thank **Mr. Nitin Gulhane** and all the Staff Members of the Information Technology Department, Management, friends, and our family members, who have directly or indirectly guided and helped us with the preparation of this report and gave us unending support right from the stage the idea was conceived.

Students Name & Signature

**Ghutukade Akash Bhagwan.**

**Harshwardhan Shailesh Gude**

**Danish Tanvir Kazi**

**Aashish Vitthal Mungase**

**Contents**

**1 Introduction To Project 7**

1.1 Objective 7

1.2 Scope of the Project 8

1.3 Problem Definition 8

**2 Literature Survey 9**

**3 Software Requirements Specification 11**

3.1 System Requirements 12

3.1.1 User Interface 14

3.1.2 Hardware Interface 14

3.1.3 Software Interface 14

3.2 Functional Requirements 14

3.3 Non-Functional Requirements 15

**4 Technical Specification 19**

**5 Result Discussion 21**

**6 Glossary 24**

**7 Conclusion 25**

**8 References 26**

**List of Tables**

Sr.No Name Page No

1 Literature Survey 12, 13

**Abstract**

The Surround View System (SVS) is an advanced image processing solution designed to enhance vehicle

safety by providing a seamless 360-degree view using multiple camera feeds. This abstract presents an overview of SVS, highlighting its key functionalities and advantages.

Modern vehicles require enhanced situational awareness to assist drivers in maneuvering through complex

environments. Traditional side and rear-view mirrors have limitations, leading to blind spots and increased accident risks. SVS addresses these challenges by integrating multiple vehicle-mounted cameras and employing real-time image stitching techniques to generate a bird’s-eye view of the surroundings.

SVS utilizes high-resolution cameras strategically placed around the vehicle, capturing live feeds that are

processed through advanced computer vision algorithms. The system aligns and stitches images seamlessly, eliminating blind spots and providing drivers with a comprehensive real-time view. Additionally, SVS incorporates object detection and obstacle marking, ensuring increased situational awareness, especially during parking and low-speed maneuvers.

The system automatically adjusts views based on the vehicle’s movement, dynamically switching between

different perspectives such as front, rear, side, and composite top-down views. It also supports integration with automotive infotainment systems, offering an intuitive and interactive display for the driver.

SVS features a robust backend powered by deep learning models such as VGG-16 for efficient image

processing. The architecture ensures low-latency processing to maintain real-time feedback. Furthermore, the system enables data logging and analytics, allowing manufacturers to analyze driving patterns and improve vehicle safety measures.

To enhance user experience, SVS provides customizable UI settings where drivers can personalize viewing

modes, adjust contrast, and enable automatic warnings for nearby objects. The system also supports over-the-air (OTA) updates, ensuring continuous improvements and adaptation to evolving road conditions.

By offering a real-time 360-degree visualization, SVS significantly reduces accident risks, enhances parking

efficiency, and improves overall driving safety. This intelligent vehicle assistance technology marks a significant step towards autonomous driving and advanced driver-assistance systems (ADAS).

**Chapter 1**

**Introduction to Project Topic**

## 1.1 Overview

**Project Overview**

The Surround View System (SVS) is an advanced driver assistance system designed to improve vehicle safety and maneuverability by providing a 360-degree view of the vehicle's surroundings. By utilizing multiple cameras placed around the vehicle, the system captures real-time images, processes them using computer vision techniques, and displays a composite view on the vehicle’s multimedia system. This technology aids drivers in parking, navigating tight spaces, and detecting obstacles, ultimately reducing the risk of accidents and improving overall driving convenience.

**1.2 Brief description**

The SVS employs a combination of image processing, graphics rendering, and real-time visualization techniques to create a seamless surround view. The system integrates four high-resolution cameras positioned at the front, rear, left, and right sides of the vehicle. These images are processed using OpenCV, OpenGL, and OpenSceneGraph (OSG) to generate 2D/3D composite views, including a bird’s-eye perspective. The system provides users with multiple viewing modes and includes features such as object detection and parking assistance, making it an essential tool for modern vehicles.

**Techniques and Algorithms**

**Image Stitching:** Uses feature detection and image warping techniques to combine images from multiple cameras into a unified 360-degree view.

**Object Detection**: Implements deep learning-based algorithms such as YOLO (You Only Look Once) or Haar cascades to identify pedestrians, vehicles, and obstacles.

**Perspective Transformation**: Adjusts camera views using OpenCV’s transformation functions to create a realistic bird’s-eye view.

**Real-time Rendering:** Utilizes OpenGL and OSG for rendering smooth, distortion-free 2D/3D images with minimal latency.

**Edge Detection and Segmentation:** Applies computer vision techniques like Canny edge detection and contour analysis for object boundry identification.

**1.3 Problem Defination**

Parking and maneuvering in confined spaces are common challenges faced by drivers, often resulting

in accidents and vehicle damage. Conventional rear-view mirrors and parking sensors offer limited assistance, as they do not provide a complete perspective of the vehicle’s surroundings. Drivers frequently encounter blind spots, making it difficult to detect objects, pedestrians, or other vehicles. The SVS addresses this issue by offering a full 360-degree view, thereby minimizing blind spots and enhancing driver awareness. Through real-time image processing and intelligent detection mechanisms, the system improves safety, reduces collision risks, and enhances driving efficiency.

**1.4 Applying Software Engineering Approach**

The development of the Surround View System follows a structured software engineering methodology, ensuring high-quality implementation and maintainability. The key phases of the software development life cycle (SDLC) applied to this project include:

**Requirement Analysis:** Gathering functional and non-functional requirements for the SVS, ensuring compatibility with vehicle systems and user needs.

**System Design:** Architecting the system to incorporate efficient image processing, rendering techniques, and user interface components.

**Implementation:** Developing software modules using OpenCV for image processing, OpenGL for rendering, and OSG for real-time visualization.

**Testing & Validation:** Conducting unit testing, integration testing, and system testing to ensure reliable performance and accuracy.

**Deployment and Maintenance:** Integrating the system with vehicle multimedia interfaces and performing periodic updates to improve functionality and security.

**Chapter 2**

**Literature Survey Table:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Sr**  **.**  **no** | **PAPER**  **NAME** | **YEAR**  **(PUBLICATIO**  **N OF PAPER)** | **AUTHOR** | **OBJECTIVE** | **METHODOLOGY** | | **CONCLUSION** | |
| 1. | 360°  Surround  View System with Parking  Guidance | 2014 | Mengmeng Yu, Guanglin Ma | To develop a real-time 360degree  surround  system with parking aid features. | Implemented four fisheye cameras  around the vehicle to capture the  surroundings.  Developed algorithms for image  distortion correction and  seamless stitching to create a unified top-  down view, enhancing parking assistance and  blind spot detection. | | The system  effectively  provides drivers  with a  comprehensive  view, improving  parking safety  and reducing  blind spots. | |
| 2. | A Surround  View Image Generation  Method with  Low Distortion for Vehicle  Camera Systems  Using a  Composite Projection | 2017 | Kunio Nobori, Norimichi  Ukita, Norihiro Hagita | To propose a method for generating  surround view  images with minimal  distortion for  vehicle camera systems. | Introduced a composite projection technique that  combines multiple camera feeds. The  method corrects lens  distortion and stitches  images to produce a coherent surround  view, facilitating  better driver assistance during parking and low-speed maneuvers. | | The proposed method  successfully  reduces image distortion,  providing a  clearer and more  reliable surround view for drivers. | |
| 3. | **Real-Time 3D**  **Surround View**  **System for Vehicle Based on Panoramic**  **Stitching Image** | 2021 | A. A. Knyaz,  A. V. Vizilter,  A. V. Knyaz,  V. M. Kniaz | | To develop a real-time 3D  surround view system for  vehicles using multiple  cameras. | Utilized multiple cameras to capture the vehicle's environment.  Applied image processing techniques to stitch images and  render a 3D model of the surroundings, enhancing driver awareness and safety. | | The system provides an  effective 3D  representation of the vehicle's  surroundings, improving situational  awareness and safety. |
| 4. | Surround-View  Fisheye Optics in  Computer Vision and Simulation:  Survey and  Challenges | 2022 | M. Mutsch,  M. Weinmann,  R. Klein | | To survey the use of fisheye  optics in surround-view systems and identify associated challenges. | Conducted a comprehensive review  of existing literature on fisheye optics in  computer vision.  Analyzed various simulation models and their effectiveness in  replicating real-world  fisheye distortions for system development and testing. | | Identified key challenges in  simulating fisheye optics and  provided  recommendations  for future research to enhance the accuracy and reliability of  surround-view systems. |

**Chapter 3**

**Software Requirements Specification**

## 3.1 Introduction

Software requirements are expressed in a software requirement document. The Software Requirements Specification (SRS) is the official statement of what is required from the system developers. This document includes both requirement definition and requirement specification. It is not a design document but rather a detailed description of the functionalities, constraints, and dependencies of the system.

###### **3.1.1 Purpose**

The goal of this document is to outline the features, requirements, and interface of the Surround View System (SVS). It clarifies the objectives of the intended project and the measures required for successful implementation. The system aims to enhance vehicle safety by providing a 360-degree view using real-time image processing and computer vision techniques.

###### **3.1.2 Intended Audience and Reading Suggestion**

This document is intended for developers, project managers, automotive engineers, and researchers interested in implementing or improving surround view technology. It provides a comprehensive guide for designing, implementing, and integrating the SVS into modern vehicles.

###### **3.1.3 Project Scope**

The scope of the project includes designing and implementing a real-time image processing system that integrates multiple cameras to generate a seamless surround view. The project consists of five phases:

**Requirement Gathering** – Researching existing techniques and collecting data on image processing for surround view systems.

**Dataset Collection & Processing** – Capturing images from vehicle-mounted cameras and preparing them for training.

**System Development & Testing** – Implementing and testing real-time image stitching, object detection, and rendering algorithms.

**User Interface & Integration** – Developing a UI for real-time display and integrating with vehicle multimedia systems.

**Deployment & Optimization** – Deploying the system and optimizing its performance for accuracy and efficiency.

**3.1.4 Design and Implementation Constraints**

Hardware Limitations: The system requires high-resolution cameras and a powerful onboard computing unit to process images in real-time.

Software Dependencies: The system will rely on OpenCV, OpenGL, and OSG for image

processing and rendering.

Latency Constraints: The system must operate with minimal delay to provide real-time feedback

to the driver.

###### **3.1.5 Assumptions and Dependencies**

##### **Assumptions:**

The vehicle has pre-installed cameras at appropriate positions. The computing hardware supports real-time image processing. The system operates under normal lighting and weather conditions.

##### **Dependencies:**

Image processing libraries like OpenCV.

GPU acceleration for rendering and real-time processing.

Compatibility with vehicle multimedia display systems.

##### **3.2 System Feature**

The Surround View System consists of two main components: frontend (user interface) and backend (image processing engine).

###### **3.2.1 System Feature 1**

Frontend -

1.The system should provide an intuitive user interface for real-time visualization.

2.Users should be able to switch between different viewing modes (e.g., top-down, split-screen, rear view).

3.The system should display detected objects with visual indicators.

4.The interface should be optimized for minimal distractions to the driver.

###### **3.2.2 System Feature 2**

Backend-

1.The system should fetch and process images from multiple cameras.

2.The system should stitch images seamlessly to create a 360-degree composite view.

3.The system should perform real-time object detection and highlight obstacles.

4.The system should minimize processing latency for real-time feedback.

## 3.3 External Interface Requirements

###### **3.3.1 User Interfaces**

1.The interface should display a real-time stitched surround view.

2.Users should be able to configure viewing angles and display settings.

3.The interface should integrate with the vehicle's infotainment system.

###### **3.3.2 Hardware Interfaces**

**Cameras:** High-resolution cameras mounted on front, rear, and sides of the vehicle.

**Processing Unit:** Embedded GPU or CPU capable of real-time image processing.

**Display Unit:** Vehicle’s infotainment screen or dedicated display panel.

**Storage:** SSD or onboard storage for caching images and logs.

###### **3.3.3 Software Interfaces**

**Operating System:** Linux-based OS for embedded automotive applications.

**Libraries:** OpenCV for image processing, OpenGL/OSG for rendering.

**Networking:** Integration with vehicle CAN bus for vehicle status updates.

## 3.4 Functional Requirements:

**Few of its functional requirements are as follows:**

1. Image Capture: The system should continuously capture images from multiple cameras.

2. Image Stitching: The system should merge images into a seamless 360-degree view.

3. Object Detection: The system should highlight obstacles using computer vision techniques.

4. Rendering: The system should display real-time images with minimal latency.

5. User Interaction: The system should allow drivers to change viewing modes dynamically.

## 3.5 Non-Functional Requirements:

**1.Security:** Ensure data integrity and protect against unauthorized access.

**2.Usability:** Provide an intuitive interface for drivers with minimal learning curve.

**3.Reliability:** The system should function in various environmental conditions.

**4.Performance:** The system should process images in real-time (<50ms delay).

**5.Scalability:** The system should support future enhancements like AI-based recognition.

###### **3.5.1 Software Quality Attributes**

**1.Availability:** The system should operate continuously without failure.

**2.Portability:** The system should be compatible with various vehicle models.

**3.Reliability:** The system should provide accurate and consistent image stitching.

**4.Flexibility:** The system should adapt to different camera configurations and resolutions

## 3.6 Analysis Model/Sequence Flow Diagram

###### **3.6.1 Flow of the Code:**

**System Inputs:**

**•** A set of images stored in the folder “C:/Users/Lenvo/Desktop/New folder/Input Images/\*.jpg”

**Main Processing Steps:**

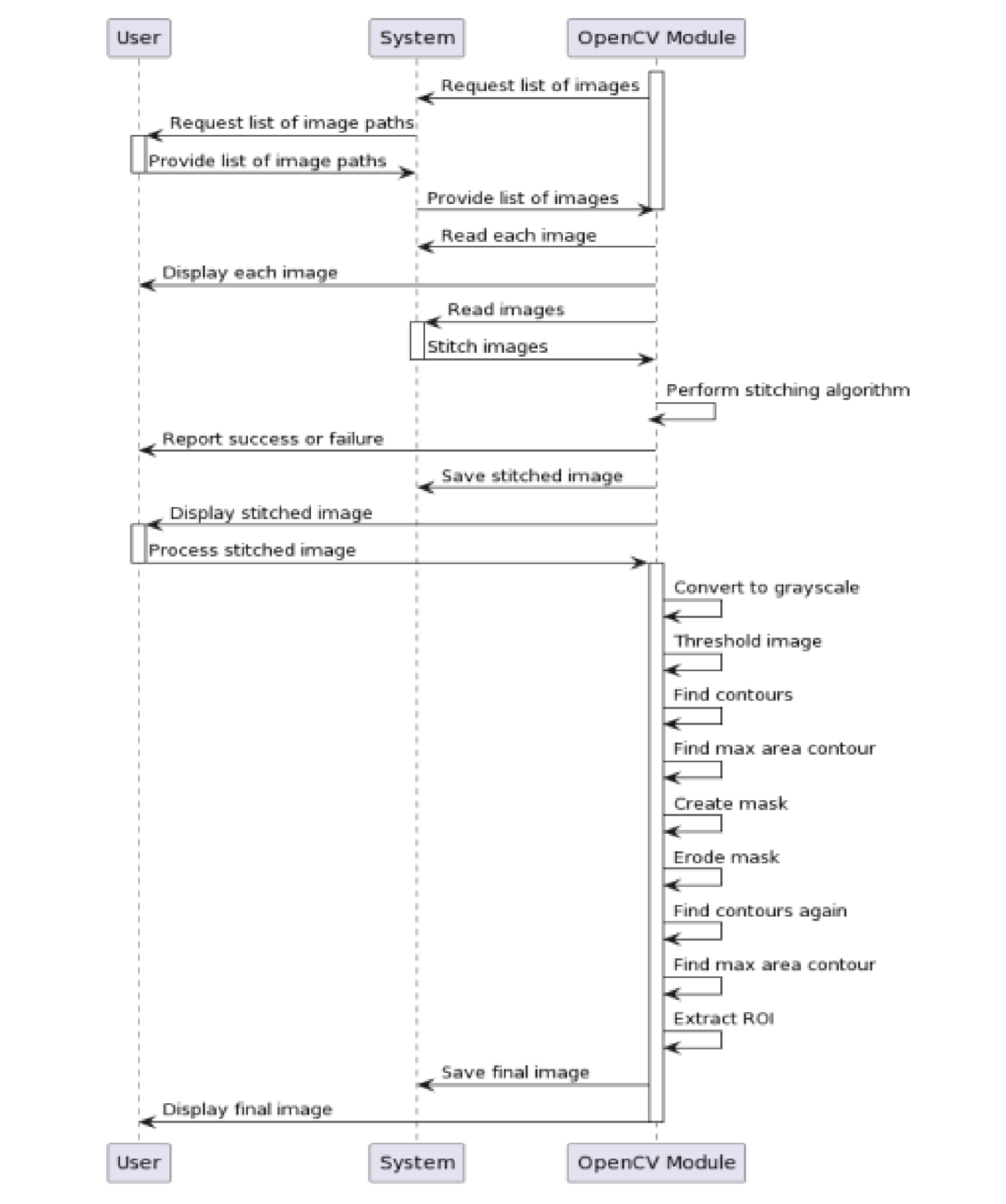
|  |  |  |  |
| --- | --- | --- | --- |
|  | **1.** | **Image Reading and Display:**  o Use cv::glob to find the image paths. o Read each image using cv::imread. o Display each image using cv::imshow and wait for user input. |  |
|  |  |  |  |
|  | **2.** | **Image Stitching:**   * Create a cv::Stitcher object with mode = cv::Stitcher::PANORAMA. o Set stitching parameters (optional, based on needs):   ▪ imageStitcher->setWaveCorrection(true) (enabled by default)  ▪imageStitcher>setWaveCorrectKind(cv::detail::WAVE\_CORRECT\_  HORIZ) (default)   * Stitch the images using imageStitcher->stitch(images, stitched\_image). o Check the status and report an error if unsuccessful. |  |
|  | **3.** | **Stitched Image Processing:** |  |

* Save the stitched image "C:/Users/Lenvo/Desktop/New folder/Output Images/stitched.jpg"
* Display the stitched image. o Add a 10-pixel black border using cv::copyMakeBorder. o Convert to grayscale using cv::cvtColor. o Threshold the image using cv::threshold to create a binary image.
* Display the threshold image. o Find contours using cv::findContours.
* Find the contour with the maximum area using findMaxAreaContour. o Create a mask around the largest contour using cv::rectangle. o Erode the mask until it's empty using cv::erode and cv::subtract. o Find contours again in the eroded mask and extract the largest one.
* Get the bounding rectangle of the largest contour.
* Extract the Region of Interest (ROI) from the stitched image using the bounding rectangle.
* Save the processed stitched image "C:/Users/Lenvo/Desktop/New folder/Output Images/stitched\_processed.jpg".
* Display the processed stitched image.

##### **System Outputs:**

* Processed stitched image with the extracted ROI saved at "C:/Users/Lenvo/Desktop/New folder/Output Images/stitched\_processed.jpg"
* Displayed processed stitched image

**b) Sequence Diagram:**



## 3.6 System Implementation Plan

#### **3.6.1 Implementation Platform**

Below is the specification of the platform on which the proposed approach will be implemented and tested:

##### **Programming Languages: C++, Python**

* **Image Processing Libraries:** OpenCV, OpenGL, OpenSceneGraph (OSG)
* **Hardware:** Embedded GPU for real-time processing
* **Operating System:** Linux-based automotive OS
* **User Interface:** HTML, CSS, JavaScript for vehicle display integration

### **3.6.2 Dataset**

The system will use datasets consisting of:

**1.Camera Feeds:** Images from front, rear, and side-mounted cameras.

**2.Calibration Data:** Data for stitching images correctly.

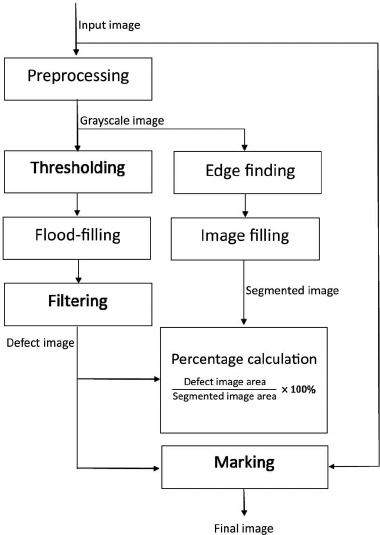
**3.Obstacle Detection Data:** Annotated images for training object detection models.

**4.Vehicle Positioning Data:** Inputs from vehicle sensors for accurate stitching.

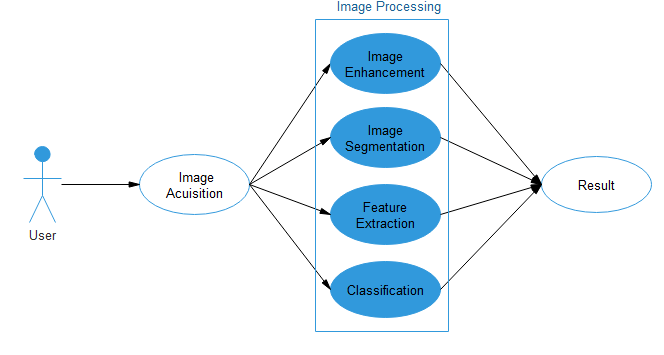
**5.Environmental Factors:** Image variations under different lighting and weather conditions.

This document provides a detailed roadmap for developing the Surround View System, ensuring that it meets safety, efficiency, and usability standards.

**Data Flow Diagram :**

****

**Use Case Diagram:**

****

**Chapter 4**

**Technical Specification**

## 4.1 Technology Details used in the project:

* **Camera Specification:**
* Number of Cameras: Typically, these systems utilize four wide-angle cameras mounted around the vehicle—one at the front, one at the rear, and one on each side mirror ▪ Field of View: The horizontal imaging radius can reach up to 194 degrees,
* Ensuring overlapping fields of view for seamless image stitching.
* **Image Processing :**
* Stitching Technology: Advanced algorithms merge indivisuals camera feed into a

cohesive 2D or 3 D bird’s eye , facilitating accurate obstacle view.

* Output Resolution : System may support high – definition outputs; for example

some deliver up to 2.5K resolution for clear visual representation .

* Frame Rate: A standard output frame rate is 30 frames per second, providing smooth

real – time video.

* **Display :**
* Type & Size: Many systems are paired with in – vehicle HD touch displays, commonly around 10.1 inches in size, offering intuitive user interface.
* Brightness & Contrast: Displays typically have brightness level around 800cd/m2 and contrast ratios of 6000:1, ensuring visibility under various lighting conditions.
* **Additional Features :**
* Recording Capabilities: Some system incorporate build in digital video recorders that support loop recording, motion detection and G – sensor – based video locking for event recording.

**Smart Parking Assistance :**

Automatic Parking – Works with the vehicle's self-parking system to park automatically.

Steering Guidance – Displays trajectory lines based on steering angle.

Obstacle Proximity Alerts – Combines ultrasonic sensors with camera feeds for precise parking.

##### **Smart Mirror Integration :**

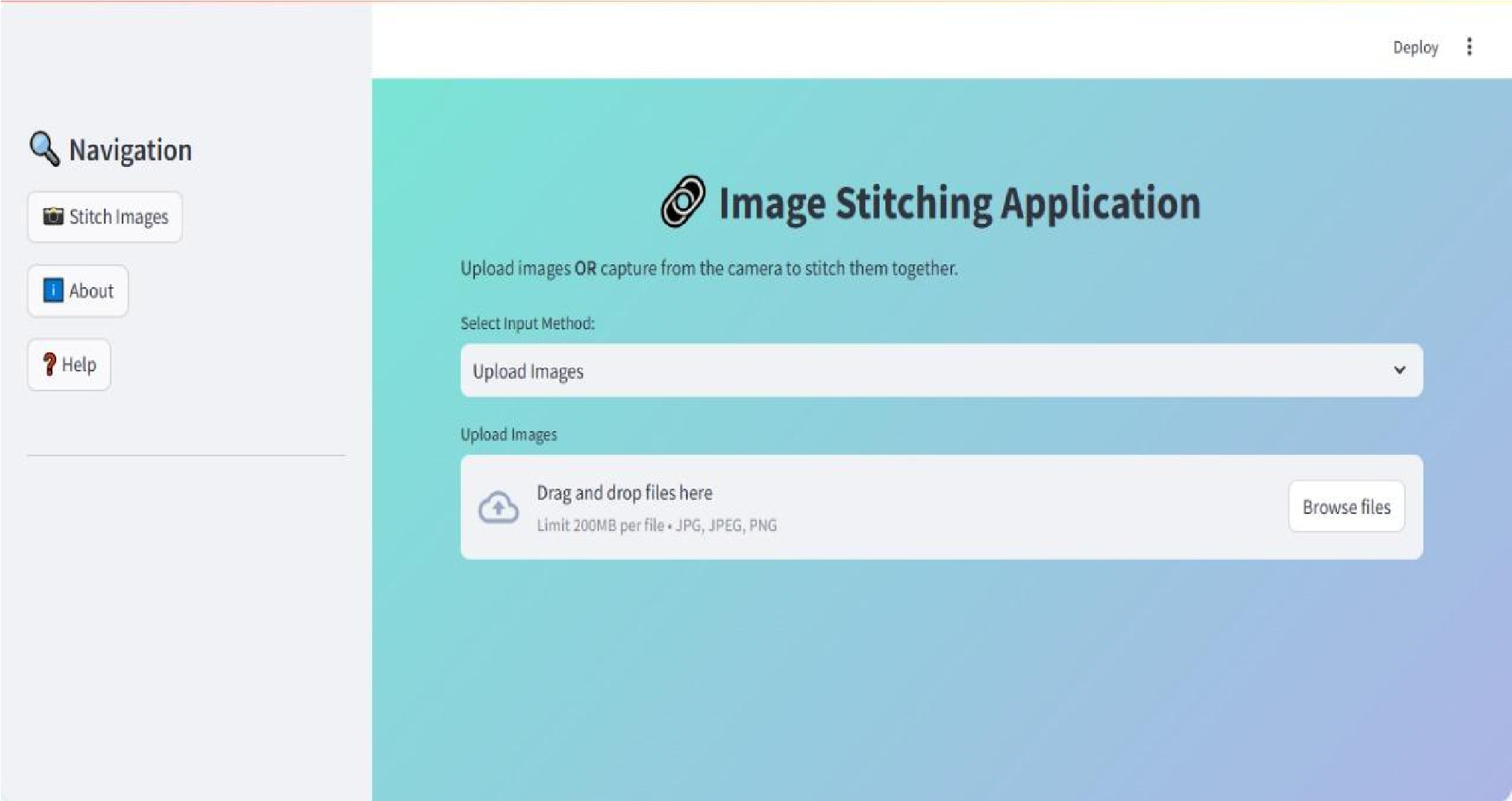
Displays 360° camera feed directly on the rearview mirror.

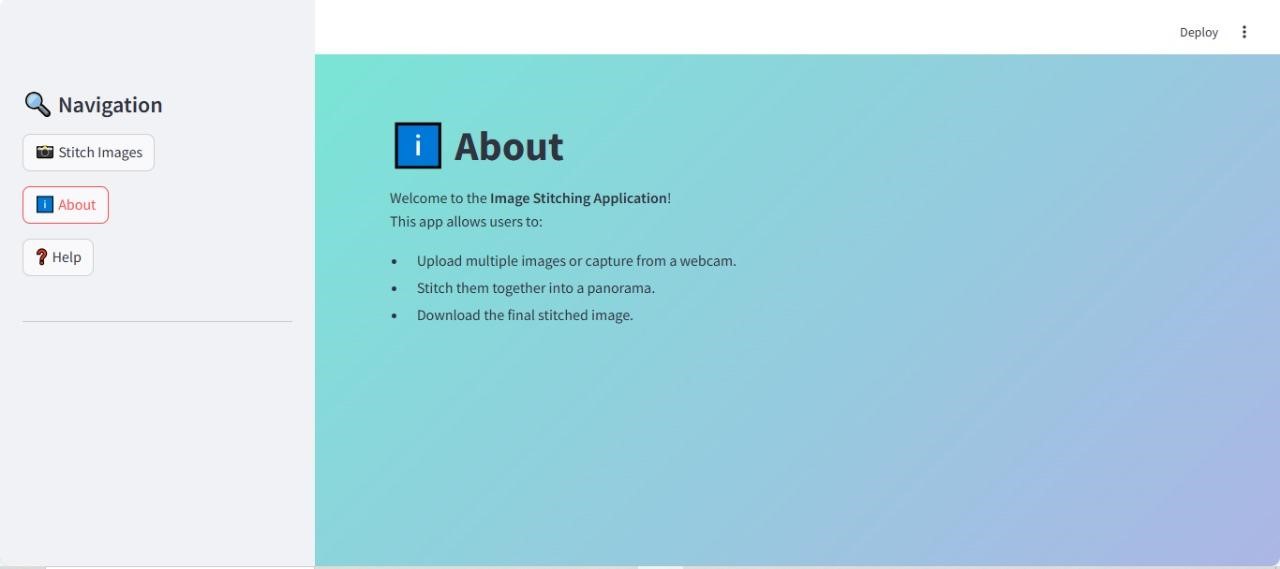
Provides a clear, unobstructed view even if passengers or cargo block the rear window.

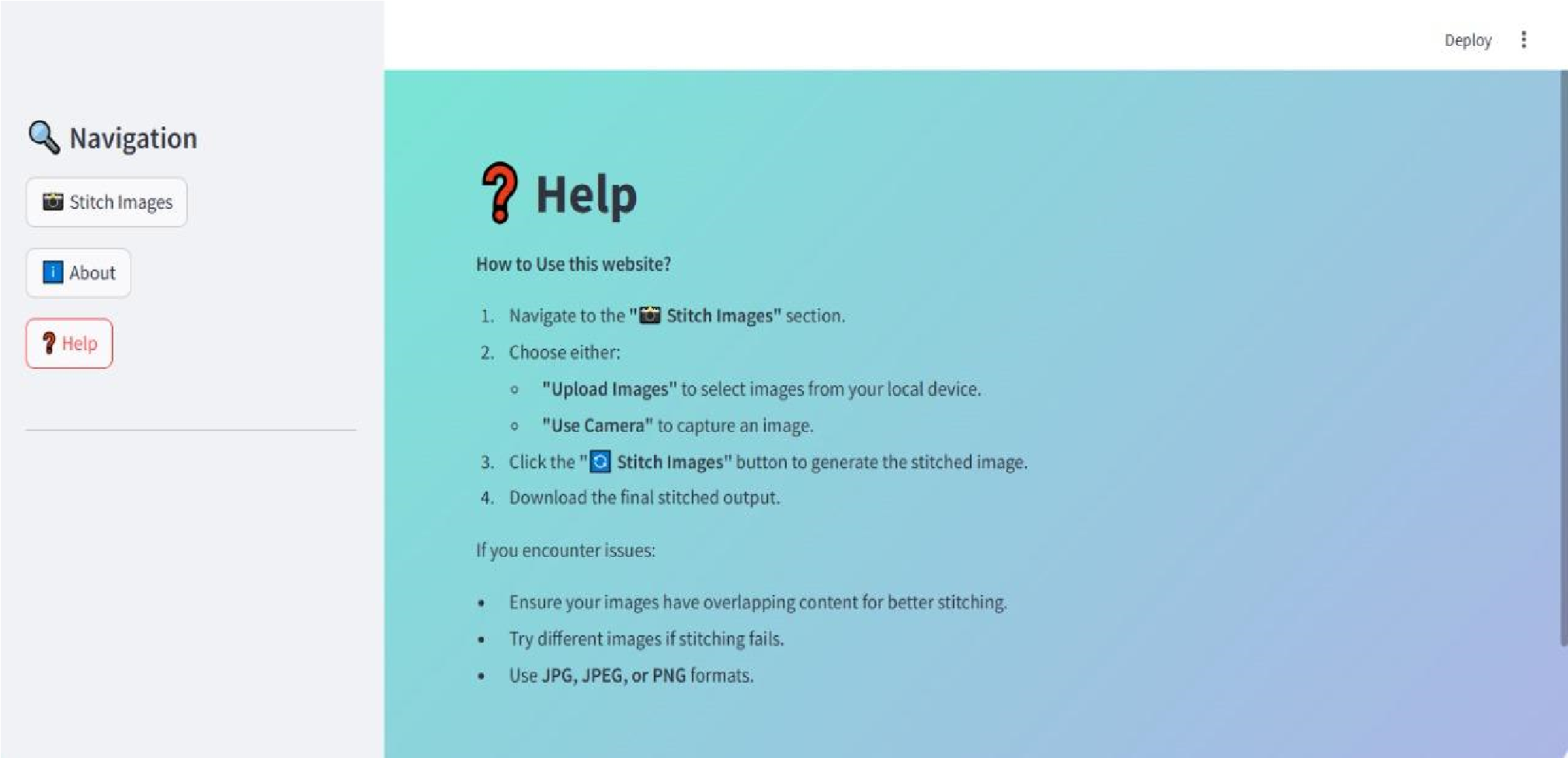
Works as a digital rearview mirror with enhanced night vision.

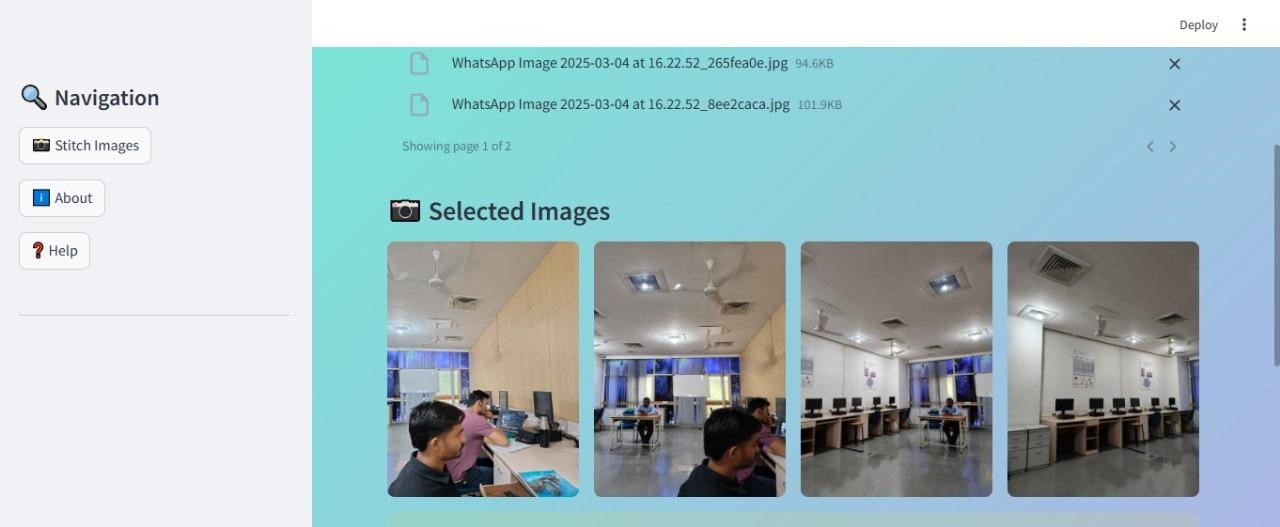
**Chapter 5**

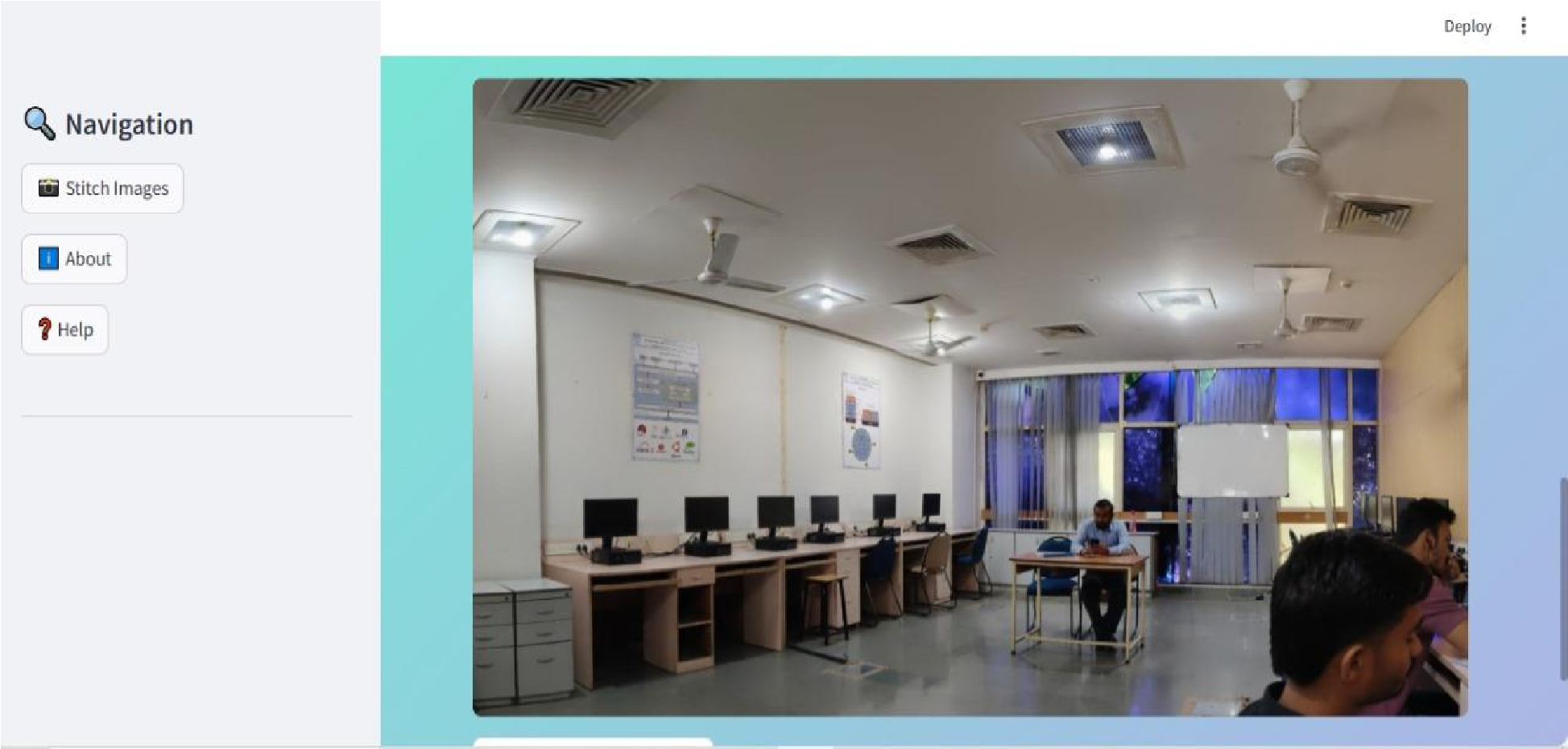
**Result Discussion:**











**Chapter 6**

**Glossary :**

1. ADAS (Advanced Driver Assistance Systems): A set of smart features like lane departure warning, collision alerts, and automatic braking that improve safety.

2. AI-Based Object Detection: Uses artificial intelligence to recognize pedestrians, vehicles, and obstacles in real time.

3. Augmented Reality (AR) Display: Overlays navigation and warnings on the camera feed for enhanced visualization.

4. Bird’s-Eye View: A top-down 360° image generated by stitching together multiple camera feeds.

5. Blind Spot Monitoring (BSM): Alerts the driver if another vehicle is in their blind spot.

6. Blending Algorithm: A technique used to merge images from multiple cameras into one seamless view.

7. CAN Bus (Controller Area Network): A communication system that allows vehicle components, including cameras, to exchange data.

8. Camera Synchronization: Ensures all four cameras capture images at the same time for seamless stitching.

9. Collision Avoidance System: Uses cameras and sensors to detect obstacles and warn the driver.

10. Dashcam Mode: A feature that allows the 360° camera system to record video footage while driving or when parked.

11. Digital Signal Processing (DSP): A technology used to enhance image quality and reduce latency.

12. Dynamic Parking Guidelines: Adjustable lines on the display that show the vehicle’s projected path based on steering input.

13. Edge Blending: A process that smooths the transition between multiple stitched images to avoid visible seams.

14. Electronic Control Unit (ECU): The brain of the 360° Surround View System, processing camera data and generating images.

15. Extended Parking Mode: Keeps cameras active when the vehicle is parked for security and surveillance.

**Chapter 7**

**Conclusion**

The **360° Surround View System** is a groundbreaking vehicle technology that enhances **safety, convenience, and driving confidence**. By using multiple **wide-angle cameras**, advanced **image stitching algorithms**, and **real-time processing**, it provides a **bird’s-eye view** of the vehicle’s surroundings, reducing blind spots and assisting in parking, maneuvering, and obstacle detection.

##### **Modern systems go beyond basic** surround-view functionality **by integrating with** ADAS (Advanced Driver Assistance Systems)**, offering features like** pedestrian detection, lane departure warnings, night vision, and smart parking assistance**. Some high-end versions also incorporate** AI-based recognition, augmented reality (AR) overlays, and 3D viewing modes **for a more interactive experience.**

With the continuous evolution of **autonomous driving, AI, and V2V communication**, future 360° camera systems will become even more **intelligent, responsive, and efficient**. This technology not only enhances safety but also contributes to a more **seamless and stress-free driving experience** in urban, highway, and off-road conditions.

**Chapter 8**

**References :**

1. [https://www.researchgate.net/publication/336716292\_Online\_Complaint\_ Management\_Systems](https://www.researchgate.net/publication/336716292_Online_Complaint_Management_Systems)
2. https://www.academia.edu/9234897/Project\_Report\_On\_Surround \_View\_System\_Submitted By
3. https:/www.researchgate.net/publication/2469166\_for\_image\_

proceesing\_using\_opencv

1. https:/www.researchgate.net/publication/by/real-time\_3D\_system
2. https:/www.ieee.com/986452/Project\_report\_on\_Opencv