**Final Year Project**

**Inspection AI**

**Inspecting defects in EV batteries**

**FYP Team**

**Abbas Mustafa**

**Shayan Bakht**

**Ali Chohan**

**Supervised By**

**Mr. Hasan Raza**

**Mr Amad Nadeem**

**FAST School of Computing**

**National University of Computer and Emerging Sciences**

**Islamabad, Pakistan**

**2021**

**Anti-Plagiarism Declaration**

This is to declare that the above FYP report produced under the:

**Title: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

is the sole contribution of the author(s) and no part hereof has been reproduced on as it is

basis (cut and paste) which can be considered as Plagiarism. All referenced parts have

been used to argue the idea and have been cited properly. I/We will be responsible and

liable for any consequence if violation of this declaration is determined.

Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Student 1

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student 2

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student 3

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Supervisor (Faculty)

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Authors’ Declaration***

This states Authors’ declaration that the work presented in the report is their own, and has

not been submitted/presented previously to any other institution or organization.

Table of Contents

[Introduction 4](#_Toc144131196)

[Literature Review 5](#_Toc144131197)

[Project Vision 6](#_Toc144131198)

[Software Requirement Specification 7](#_Toc144131199)

[Iteration Plan 8](#_Toc144131200)

[Iteration 1 9](#_Toc144131201)

[Iteration 2 10](#_Toc144131202)

[Implementation Detail 11](#_Toc144131203)

[User Manual 12](#_Toc144131204)

[References 13](#_Toc144131205)

[Appendices 14](#_Toc144131206)

# Introduction

# Literature Review

# Project Vision

# Software Requirement Specification

## Features:

### Real-time Video Processing:

The system should be able to process video streams from smart spectacles in real time.

### 3D Model Generation:

Convert the captured video frames into a 3D model of the inspected object.

Provide a user-friendly interface to visualize and manipulate the 3D model.

### Object Detection and Recognition:

Detect and recognize objects within the captured video.

Highlight and label recognized objects in both the video stream and the 3D model.

### Anomaly Detection:

Identify and flag anomalies or defects on the inspected object based on predefined criteria.

Provide notifications and alerts when anomalies are detected.

### Measurement and Dimension Analysis:

Allow users to measure dimensions of objects within the 3D model.

Provide accurate measurements for inspection and analysis.

### Annotation and Reporting:

Enable users to annotate detected anomalies and add notes for further analysis.

Generate detailed inspection reports with annotated images and 3D models.

### Integration with Data Systems:

Provide integration with data storage and management systems to save inspection data and reports.

Enable seamless data sharing and retrieval for future reference.

### Augmented Reality (AR) Overlay:

Overlay relevant information, such as object specifications or inspection guidelines, onto the user's view through the smart spectacles.

### User Authentication and Authorization:

Implement secure user authentication to ensure only authorized personnel can access the system.

Assign different permission levels for various user roles.

### User-Friendly Interface:

Design an intuitive and user-friendly interface for controlling the smart spectacles and interacting with inspection results.

## Functional Requirements:

### Video Stream Capture:

Capture high-quality video streams from the smart spectacles' cameras.

Ensure synchronization between the video streams and the 3D model generation.

### 3D Model Reconstruction:

Process captured video frames to reconstruct an accurate 3D model of the inspected object.

Utilize computer vision and depth-sensing techniques for precise reconstruction.

### Object Detection Algorithms:

Implement advanced object detection algorithms to identify and locate objects within the video stream.

Utilize deep learning models for object recognition.

### Anomaly Detection Algorithms:

Develop algorithms to detect anomalies based on predefined patterns or criteria.

Ensure high accuracy and low false-positive rates.

### Measurement and Dimension Calculation:

Implement algorithms to calculate accurate measurements and dimensions from the 3D model.

### AR Overlay Generation:

Generate AR overlays with relevant information and annotations to enhance the user's view.

### Data Storage and Management:

Provide a reliable database to store inspection data, 3D models, annotations, and reports.

Ensure data security and integrity.

### Notification and Alert System:

Implement a notification system to alert users in real time when anomalies are detected.

### User Authentication and Roles:

Develop a secure authentication mechanism with role-based access control.

Administer user roles such as inspectors, supervisors, and administrators.

### Integration with External Systems:

Provide APIs or integration points for connecting the inspection AI system with external data and reporting systems.

## Quality Attributes:

### Accuracy:

The system should exhibit a high level of accuracy in object detection, anomaly recognition, and measurement calculations to ensure reliable inspection results.

### Performance:

The system should process video streams and generate 3D models in real time with minimal latency to support efficient and effective inspections.

### Scalability:

The system should be designed to handle an increasing number of concurrent users and video streams without compromising performance.

### Reliability:

The system should be highly reliable, minimizing downtime and ensuring consistent availability for inspection tasks.

### Security:

Data transmission, storage, and access should be secured using encryption and proper authentication mechanisms to protect sensitive inspection data.

### Usability:

The user interface should be intuitive and user-friendly, allowing inspectors to easily control the smart spectacles and interact with inspection results.

### Interoperability:

The system should be designed to integrate with various data storage systems, APIs, and external reporting tools commonly used in the inspection industry.

### Maintainability:

The system's architecture and codebase should be well-organized and documented to facilitate future maintenance and updates.

### Adaptability:

The system should be adaptable to different types of objects and inspection environments, accommodating a variety of inspection scenarios.

### Privacy:

The system should adhere to privacy regulations and guidelines, ensuring that captured video and inspection data are handled with appropriate consent and safeguards.

## Non-Functional Requirements:

## Response Time:

The system should respond to user interactions and anomaly detections within a predefined acceptable time frame.

## Processing Speed:

Video processing and 3D model generation should be completed within a reasonable time to provide real-time insights.

## Data Storage Capacity:

The system should be able to handle a large volume of inspection data, including 3D models, video streams, and reports.

## Security Measures:

Ensure data encryption during transmission and storage, implement secure authentication and authorization mechanisms, and conduct regular security audits.

## Compatibility:

The system should be compatible with a variety of smart spectacle models, operating systems, and hardware configurations.

## Scalability Limits:

Define the maximum number of concurrent users, video streams, and objects that the system can handle while maintaining performance.

## Accuracy Metrics:

Define accuracy metrics for object detection, anomaly recognition, and measurement calculations, and ensure that the system meets or exceeds these metrics.

## Backup and Recovery:

Implement regular data backups and establish a disaster recovery plan to minimize data loss in case of system failures.

## User Training and Support:

Provide comprehensive user training materials and support resources to help inspectors effectively use the system.

## Regulatory Compliance:

Ensure that the system complies with industry-specific regulations and standards related to inspections and data handling.

# Iteration Plan

## Iteration 1:

1. Data Collection
2. Model Creation for 2D image to Depth Map
3. Model creation for 2D image + Depth map to 3D model
4. Creation of Front End
5. Implementation of Login System

## Iteration 2:

1. Annotation of Data for detection
2. Model training for detection
3. Deployment of model with front end and 3D generation pipeline
4. Adding database backend to store detections along with IDs
5. Model Optimization to work in Realtime

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Week** | | | | | | | | | | | | |
| **Module** |  | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** |
| **Iteration 1** |  |  |  |  |  |  |  |  |  |  |  |  |
| **Iteration 2** |  |  |  |  |  |  |  |  |  |  |  |  |

# Iteration 1

### Data Collection

1. Collection of 2D images (around six images per object from multiple angles) along with their 3D models for training a 2D to 3D model via scrapping from the internet and using prebuilt conversion models to convert task specific images and objects.
2. Generation of depth maps for collected images

## Model Creation for 2D image to Depth Map

Creating an architecture based on GANs to convert given images into a depth map to specify the third dimension in an image and later use the said depth map for geometric measurements to help creation of 3D model

## Model creation for 2D image + Depth map to 3D model

Using the previous model and preprocessing create a pipeline to feed all six images ( plus depth maps ) to train another GANs based model to convert 2D images into a 3D model (.obj file)

## Creation of Front End

Using UI UX and web dev to create a user-friendly design for a web app (based on Flask or Fast API) that integrates previously created models. That provides livestream as well as detections on the livestream and creates a log file for found anomalies

## Implementation of Login System

Creation of a login system integrated into the front end along with a database (firebase or AWS/ google big cloud) to stop unwanted access to the application and provide hierarchy-based feature control

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Week** | | | | | | | | | | | | |
| **Module** |  | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** |
| **Iteration 1** |  |  |  |  |  |  |  |  |  |  |  |  |
| **Iteration 2** |  |  |  |  |  |  |  |  |  |  |  |  |

# Iteration 2

# Implementation Detail

# User Manual

# References

# Appendices