Software Requirements Specification

for

AUTOMATIC DETECTION OF UNEXPECTED ACCIDENTS UNDER BAD CCTV MONITORING CONDITIONS IN TUNNELS

A.Anvesh – (227Z1A0511)
Ch .SaiVamshi – (227Z1A0536)
Ch.Edukondalu – (227Z1A0531)

Under the Guidance of Mrs. V.RANI SAILAJA

(ASSISTANT PROFESSOR)

Table of Contents

1. Introduction **1**
1.1 Document Purpose 1
1.2 Product Scope 1
1.3 Intended Audience and Document Overview 1
1.4 Definitions, Acronyms and Abbreviations 1
1.5 Document Conventions 2
1.6 References and Acknowledgements
2. Overall Description **2**
2.1 Product Overview 2
2.2 Product Functionality **2**
2.3 Design and Implementation Constraints 3
2.4 Assumptions and Dependencies 3
3. Specific Requirements **4**
3.1 External Interface Requirements 4
3.2 Functional Requirements 6
4. Other Nonfunctional Requirements **7**
4.1 Performance Requirements 7
4.2 Safety Requirements 7
4.3 Security Requirements 7
4.4 Software Quality Attributes 7

1. Introduction

1.1 Document Purpose

The purpose of this document is to describe the design and implementation of an on Python, which uses object tracking techniques to detect the occurrence of an accident. The system aims to provide timely care and alert emergency responders in the event of a traffic accident, thereby reducing mortality rates.

1.2 Product Scope

The product scope includes:

Development of an emergency alert system using Python and computer vision techniques.

Implementation of object tracking algorithms to detect accidents.

Automated reporting of accident sites to emergency responders.

Integration with existing transportation systems to improve travel safety.

1.3. Intended Audience and Document Overview

The intended audience for this document includes:

Software developers and engineers working on intelligent transportation systems.

Researchers and academics interested in computer vision and object tracking techniques.

1.4. Definitions, Acronyms, and Abbreviations

ITS: Intelligent Transportation Systems

CV: Computer Vision

OT: Object Tracking

API: A

pplication Programming Interface

GUI: Graphical User Interface

1.5. Document Conventions

This document uses the following conventions:

Bold text: Used to highlight important terms and concepts.

Italic text: Used to indicate variables and placeholders.

Code blocks: Used to display example code snippets.

1.6. References and Acknowledgements

This document references the following sources:

[1] "Intelligent Transportation Systems: A Survey" (Journal of Transportation Engineering, 2020)

[2] "Object Tracking in Computer Vision" (Book, 2019)

2 Overall Description

2.1 Product Overview

The emergency alert system is a software application designed to detect accidents and alert emergency responders in real-time. The system uses computer vision techniques to analyze video feeds from traffic cameras and detect accidents. Once an accident is detected, the system sends an alert to emergency responders with the location and severity of the accident.

2.2 Product Functionality

The emergency alert system has the following functionality:

Accident Detection: The system uses computer vision techniques to detect accidents in real-time.

Alert Generation: The system generates alerts when an accident is detected, including the location and severity of the accident.

Notification: The system sends notifications to emergency responders with the alert information.

Video Feed Analysis: The system analyzes video feeds from traffic cameras to detect accidents.

Design and Implementation Constraints

The emergency alert system has the following design and implementation constraints:

Real-time Processing: The system must process video feeds in real-time accidents and generate alerts.

Accuracy: The system must be accurate in detecting accidents and generating alerts.

Scalability: The system must be scalable to handle multiple video feeds and detect accidents in a timely manner.

Integration: The system must integrate with existing transportation systems and emergency response systems.

Assumptions and Dependencies

The emergency alert system has the following assumptions and dependencies:

Video Feed Quality: The system assumes that the video feeds from traffic cameras are of high quality and provide a clear view of the traffic.

Camera Placement: The system assumes that the traffic cameras are placed in strategic locations to provide a clear view of the traffic.

Network Connectivity: The system depends on network connectivity to send alerts to emergency responders.

Emergency Response System: The system depends on the emergency response system to receive and respond to alerts

3.Specific Requirements

3.1 External Interface Requirements

The external interface requirement for accident**s** detection in tunnels involves:

1. CCTV cameras providing video feeds.
2. Integration with tunnel management systems.
3. Alert systems for notifications.
4. Network connectivity for real-time data transmission.

Hardware Interfaces

The hardware interface for accident detection in tunnels involves CCTV cameras, video processing units, sensors, and alert systems that work together to capture, analyze, and respond to incidents in real-time.

Software Interfaces

The software interface for accident detection in tunnels involves video analytics, object detection, accident detection algorithms, and alert systems that work together to detect and respond to incidents in real-time.

3.2 Functional Requirements

3.2.1Accident Detection:

• The system must be able to detect accidents in real-time using sensors (e.g., vehicle crash
sensors, cameras) or image recognition software (e.g., through deep learning models for object

detection).

• The system must identify different types of accidents such as collisions, rollovers, and
pedestrian accidents.

3.2.2 Data Processing:

The system must process input data (from cameras, sensors, etc.) and analyze it using deep Learning algorithms (e.g., Convolutional Neural Networks, Recurrent Neural Networks).

• The system should extract critical information such as the severity of the accident, the location,
and the number of people involved.

3.2.3Alert Generation:

• Once an accident is detected, the system must immediately send an alert to emergency services
(police, fire, medical) with the necessary details like the location and type of accident.

• Alerts should be sent in multiple formats (e.g., SMS, email, app notifications) to ensure timely
delivery.

3.2.4 Location Tracking:

• The system must automatically detect the geographic location of the accident using GPS or a
similar system.

• It should integrate with mapping systems to provide accurate location data to emergency.

The system must process input data (from cameras, sensors, etc.) and analyze it using deep
learning algorithms (e.g., Convolutional Neural Networks, Recurrent Neural Networks).
• The system should extract critical information such as the severity of the accident, the location,
and the number of people involved.

3.2.5 Alert Generation:

• Once an accident is detected, the system must immediately send an alert to emergency services
(police, fire, medical) with the necessary details like the location and type of accident.
• Alerts should be sent in multiple formats (e.g., SMS, email, app notifications) to ensure timely
delivery.

3.2.6 Location Tracking:

• The system must automatically detect the geographic location of the accident using GPS or a
similar system.
• It should integrate with mapping systems to provide accurate location data to emergency.

4 .Other Nonfunctional Requirements

4.1. Performance:

• The system should process data in real-time with minimal delay (e.g., within seconds of an
accident occurrence).
• The deep learning model should perform at high speed, especially in processing higher solution
images or sensor data without latency

4.2. Scalability:

• The system should be scalable to handle a large number of accidents and locations.
• It must support expansion for integration with additional sensors or cameras across various
regions

4.3. Availability:

• The system should have high availability, ensuring 24/7 operation with minimal downtime.
• There should be redundancy for critical components (e.g., cloud backup, failover mechanisms).

4.4. Reliability:

• The system should reliably detect and report accidents without failure. If a failure occurs, it
must automatically recover or alert system administrators.
• It should ensure high accuracy in accident detection and reduce the number of false negatives
(i.e., missing real accidents).