Traffic Eye - Aerial Traffic Monitoring system

Keva Harris James, Samuel Hurtado, Milind Pansare, Saivineeth Suram

Version Number: 01

Version Date: March 01, 2023

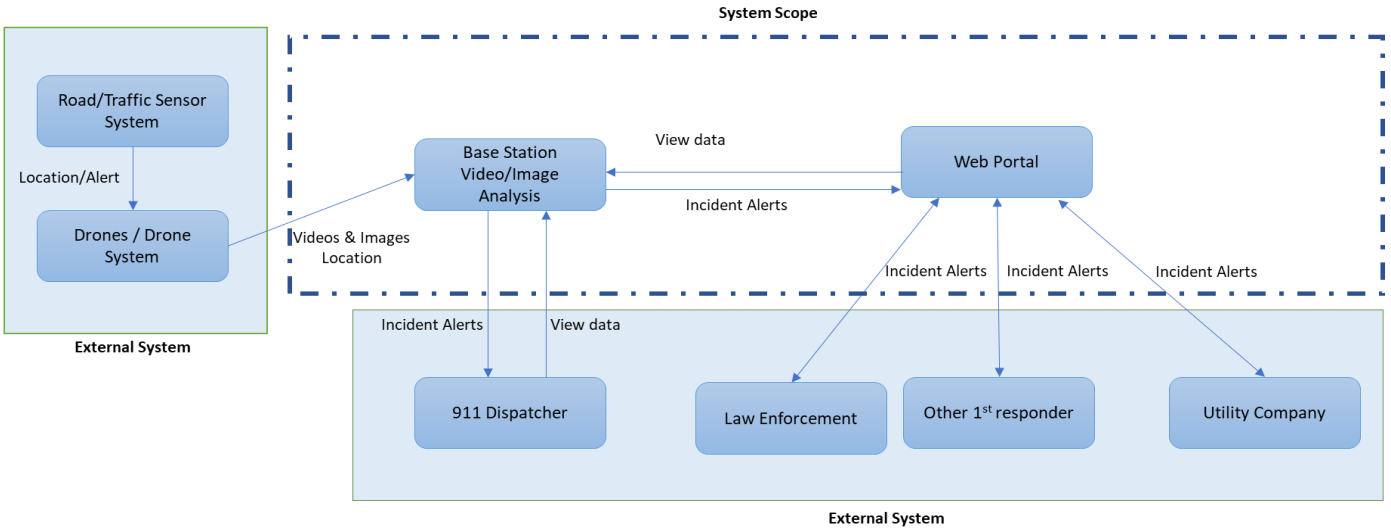
Abstract

Traffic incidents have the potential to impact the well being, and even the life of drivers, not to mention the potential cost in millions of dollars in losses due to property damage as well as lost productivity in traffic delays. The goal of this project is to develop a system that efficiently ameliorates these issues using data analysis software. The program involves using Artificial Intelligence to detect aerial shots of different traffic incidents and then send out appropriate signals for first responders, depending on the type of incident. Using Python libraries for the core program, and a JavaScript based user interface, the team will train the model with various types of traffic accidents with varying levels of severity. The video and picture data will be used to train the model, and elicit a proper response signal to the various emergency response departments such as police and fire departments, highway cleanup, or sewer departments for burst pipes and other first responders. Due to the scope of this course, the team will presuppose a working drone is able to send the video files, coordinates, and any other relevant information. The focus of development will be on the video processing and a UI based portal for interaction with a 911 dispatcher. The team will also presuppose the existence of communication of the system with the appropriate proper governmental departments such as police or fire rescue.

Use Case Details:

As an external system, the Aerial Traffic monitoring system will detect road congestion and will dispatch drones to detect the reasons. A drone will fly through the congested traffic path and capture videos of the accident site and transmit the location and video/pictures to the base station for further analysis. The Base Station will analyze the data, generate alerts and reports, and will dispatch to another external system for first responders with details about emergency and needs. The proposed software system is for the video processing Base Station and UI Portal for the user interaction. Drone video capturing, streaming and interfacing with the first respondent systems is out of the project scope.

System Diagram



Functional Requirements

System requirements - The base station analysis system will

- Receive video footage from the congested traffic
- Store the video data securely
- Process the video and alert the vehicle crash
- Process the video and alert the utility (water, power line) breakages
- Send alerts that will contain the pictures and location information
- Host a web portal

User requirements - The web portal will:

- Allow authenticated users to login
- List all the alerts
- Show the alerts on map
- Provide an option to write comments/notes for each of the listed alerts
- Have an option to define the priority of the alert (critical, high, low)
- Give the option to close the alert
- Integrate with external first respondent systems (police, fire, hospitals) to notify the alerts

Specific User Roles Requirement

- ***Drone:*** images, videos, and location
  - Update logs
  - Upload captured images and files
- ***Dispatcher***: login data (username, password, authentication)
  - View incident reports
  - Access images, videos

- ***Dispatcher supervisor***: login data (username, password, authentication)
    - View incidents reports
    - Access videos and images
    - Generate reports
    - Define incidence priority (low, high, critical)
- ***First responders:*** login data (username, password, authentication)
    - View incident reports
    - Access images

Non Functional Requirements

- Response time - Video processing should be done in less than 2 mins
- Capacity - System to support 1000 concurrent users
- Data confidentiality
- Legal, regulatory, compliance
- Maintainability
- Availability - System availability should be 95% with 5% downtime for system maintenance and updates

Technology Stack

- Python v3.8
- Tensorflow v2.11.0
- Pytorch 1.13
- NoSql CouchDB / MongoDB
- ReactJS 17.0.0
- GitHub - code repository

Milestones

- Acquiring video files
- Reading Video files
- Processing the video file to detect incidence
- Creating alert record into the database table
- Portal to view the database entries

Timelines

No.	Start Date	End Date	Description
	03/05	03/11	***Project Analysis and Design***
1			Project documentation
2			Design algorithms & choose models
3			Document Requirements and Planning
	03/12	03/13	***Environment Setup***
4			Environment setup - Colab
5			Environment setup - local
	03/14	03/31	***Base Station Design and development***
6			Gathering Video Files, Repository setup
7			Coding to process Video file in Tensorflow/Pytorch
8			Model training - Video file - Object detection
9			Alert creation/insertion in DB
10			List alerts in Python notebook
11	04/01	04/05	***Base Station Testing and debugging***
	04/01	04/10	***Portal builds***
12			Login Page
13			Alert list view
14			Alert Map view
15	04/10	04/15	**System Integration Testing and defect fixing**
16	04/16	04/19	**Demo**