SRS for Directional Tracking and Enumeration System

Prepared by:

Saurabh Deshmukh
Vivek Raj
Kartik Gade
Kaustubh Bhavsar
Prachi Chavan
Isha Agrawal

Organization:

Nanosoftrd
Jadhav complex,
Manik Chowk,
Chakan, Maharashtra
410501

- **Domain**: Software Development

Feb 13, 2024

CS349 Software Engineering
Guided by: Prof. Santosh Warpe

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Phase 1, Draft 1	13/02/24	Initial template, First Review completed	0.0

1. Introduction

1.1 Purpose

The purpose of this document is to define the requirements for the development of a software solution aimed at identifying the direction of moving cars or people in video frames and calculating their counts.

1.2 Intended Audience and Reading Suggestions

1. Researchers and developers in computer vision and video analytics.
2. Security companies and professionals interested in enhancing surveillance systems.
3. Government agencies involved in public safety and monitoring.
4. Technologists looking to integrate advanced motion tracking capabilities into their products or services.
5. The audience for this document includes the development team, client, and all persons involved in the implementation or evaluation of this software solution.

1.3 Project Scope

To tackle this project, we would likely need to employ computer vision techniques. Here's a general outline:

1. Object Detection: By using a pre-trained object detection model (such as YOLO, SSD, or Faster R-CNN) to detect cars or people in the video frames.

2. Direction Estimation: Once we have detected the objects (cars or people), you can track their movement across consecutive frames using techniques like optical flow or object tracking algorithms.

3. Direction Classification: Based on the tracked movement, classify the direction of each object (car or person) as left, right, up, or down. This could involve analyzing the change in position of each object between frames.

4. Counting: Finally, count the number of objects moving in each direction by keeping track of the direction classification results.

5. Visualization: Optionally, you can visualize the count of objects moving in each direction using graphical representations or overlays on the video frames.

1.4 References

Company: Nanosoftrd: AI based application development company

• Existing work on YOLOV7: https://github.com/ultralytics/ultralytics

• Research Paper:
https://www.researchgate.net/publication/376252973_Enhancing_Realtime_Object_Detection_with_YOLO_Algorithm

2. Overall Description

2.1 Product Perspective

A directional tracking and enumeration system is a product designed to monitor and track the movement of objects or individuals in a specific direction and provide accurate enumeration or counting information. This system can have various applications across different industries, such as retail, logistics, security, and more. Here's a breakdown of the product perspective for a directional tracking and enumeration system:

1. Purpose and Goals:
- Clearly define the purpose of the system, whether it's for retail analytics, crowd management, security surveillance, or any other specific application.
- Set clear goals for the system, such as providing real-time tracking, accurate counting, and generating insightful reports.

2. Technology and Hardware:
- Specify the technology used for tracking, which could include sensors (such as cameras, RFID, infrared), IoT devices, or any other relevant technology.
- Define the hardware requirements, considering factors like installation, maintenance, and scalability.

3. Software and Algorithms:
- Outline the software architecture, including algorithms for directional tracking and enumeration.
- Consider the use of machine learning or computer vision for enhanced accuracy in tracking and counting.
- Provide a user-friendly interface for configuration, monitoring, and reporting.

4. Directional Sensing and Tracking Accuracy:
- Discuss how the system will accurately sense and track the direction of movement.
- Address potential challenges like overlapping movements, varying speeds, and environmental factors that may impact tracking accuracy.

5. Real-time Monitoring and Reporting:
- Emphasize the importance of real-time monitoring capabilities for immediate decision-making.

- Develop a reporting system that provides actionable insights, trends, and historical data for analysis.

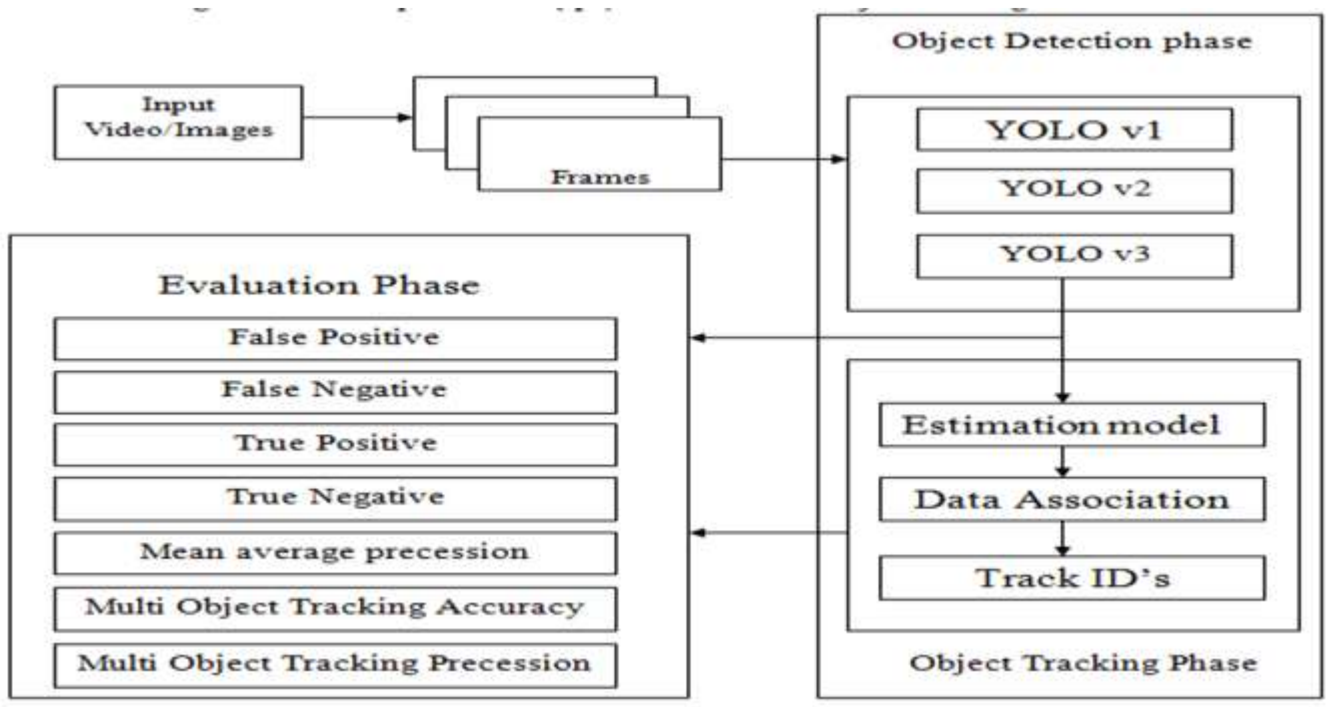
6. Integration and Compatibility:
- Ensure compatibility with existing infrastructure and systems, such as Point of Sale (POS) systems, security systems, or any other relevant technology.
- Provide options for integration with third-party software or analytics tools.

7. Scalability and Flexibility:
- Design the system to be scalable to accommodate varying sizes of spaces and different levels of tracking requirements.
- Consider flexibility in deployment, allowing the system to be used in diverse environments.

8. Security and Privacy:
- Implement robust security measures to protect the data collected by the system.
- Address privacy concerns by adhering to relevant regulations and providing features like anonymization of data.

9. Maintenance and Support:
- Develop a maintenance plan that includes regular updates, troubleshooting procedures, and customer support.
- Provide training materials and support for end-users.

10. Cost and ROI:
- Clearly outline the initial costs and ongoing expenses associated with the system.
- Highlight the potential return on investment (ROI) through improved efficiency, reduced losses, or other relevant benefits.



2.2 Product Features

- Develop an algorithm for detecting and tracking cars or people in video frames.
- Implement a direction identification mechanism to determine the movement direction of detected objects.
- Develop a counting mechanism to tally the number of objects moving in each direction.
- Provide a user-friendly interface for visualizing the results.

2.3 User Classes and Characteristics

Classes:

1. **VideoProcessor**: Handles input video processing and frame extraction.
2. **ObjectDetector**: Identifies objects (e.g., cars, people) in frames using computer vision or deep learning.
3. **DirectionEstimator**: Analyzes object trajectories to estimate their movement direction.
4. **CountingModule**: Tracks object counts for each direction (left, right, up, down).
5. **VisualizationModule**: Generates user-friendly visualizations (e.g., graphs, heatmaps) of object counts.

User Classes:

Client: The Software based Artificial Intelligence organization named "Nanosoftrd".
Developers : Students of MIT Academy of Engineering (6 members).
Monitoring personnel: The people incharge of monitoring all sensors in case of security or any defects.

Characteristics:

1. **Real-time Processing**: Provides quick analysis of video footage.
2. **Scalability**: Handles varying video resolutions, frame rates, and multiple streams.
3. **Accuracy**: Ensures precise object detection, direction estimation, and counting.
4. **Robustness**: Functions well under different lighting and occlusion conditions.
5. **Customization**: Allows user-defined settings for sensitivity, regions of interest, and alerts.
6. **Integration**: Easily integrates with existing surveillance systems or frameworks.
7. **User Interface**: Offers an intuitive interface for configuration and monitoring.
8. **Performance Optimization**: Ensures efficient processing even with large datasets.
9. **Security**: Maintains data privacy and security.
10. **Documentation and Support**: Provides comprehensive guides and support resources.

2.4 Operating Environment And Hardware Descriptions

For a directional tracking and enumeration system, the operating environment and hardware descriptions are crucial components to ensure the effectiveness and reliability of the system.

1. Indoor/Outdoor Considerations:
- Specify whether the system is designed for indoor, outdoor, or both environments.
- Consider environmental factors such as temperature variations, humidity, and exposure to sunlight.

2. Lighting Conditions:
- Address how the system performs under different lighting conditions, including low light, bright light, and artificial lighting.

3. Integration with Existing Systems:
- Specify compatibility with existing infrastructure, including POS systems, security systems, or other relevant technologies.

4. Durability and Reliability:
- Emphasize the durability of the system components to withstand environmental conditions and regular usage.
- Ensure reliability in continuous operation over extended periods.

Hardware Descriptions:

1. Sensors:
- Specify the type of sensors used for directional tracking, such as cameras, RFID, infrared sensors, or a combination.
- Provide details on the sensor's range, accuracy, and resolution.

2. Processing Unit:
- Describe the central processing unit (CPU) or processing architecture used for real-time data analysis and tracking algorithms.
- Highlight the system's processing power to handle simultaneous tracking and enumeration tasks.

3. Memory and Storage:
- Specify the amount of RAM and storage capacity required for efficient data processing and storage.
- Address whether the system uses cloud-based storage or on-site storage solutions.

4. Power Requirements:
- Outline the power requirements for the system, including voltage, power consumption, and any backup power options in case of outages.

5. IoT Devices:
- If applicable, describe any Internet of Things (IoT) devices used for connectivity and communication between system components.
- Detail how these devices contribute to the overall functionality of the system.

6. Mounting and Installation:
- Provide guidelines for the installation process, including mounting options and any specific requirements for optimal performance.

7. User Interface:
- Detail the hardware components involved in the user interface, such as touchscreens, buttons, or other interaction mechanisms.

9. Scalability:
- Address how the hardware is designed to scale for larger spaces or increased tracking requirements.

10. Maintenance and Upgrades:
- Provide information on how hardware maintenance is conducted and how the system accommodates future upgrades or enhancements.

2.5 Design and Implementation Constraints

1. **Computational Resources**: Limited processing power may restrict algorithm complexity and real-time capabilities.
2. **Memory Limitations**: Available memory may limit dataset sizes processed or stored in memory.
3. **Processing Time**: Real-time requirements necessitate efficient algorithms to meet frame processing deadlines.
4. **Accuracy and Reliability**: System must achieve reliable object detection and tracking despite variations in data quality and environmental conditions.
5. **Integration Compatibility**: System must integrate seamlessly with existing surveillance infrastructure, adhering to protocols and standards.
6. **Data Privacy and Security**: Compliance with data privacy regulations and security standards is crucial for protecting captured information.
7. **Environmental Conditions**: Performance may be affected by lighting, weather, or other environmental factors, requiring robustness in algorithm design.
8. **User Interface and Accessibility**: Interface design must consider user preferences, device compatibility, and accessibility requirements.
9. **Scalability and Flexibility**: System architecture should support scalable deployment and accommodate increasing data volumes or user demand.
10. **Regulatory Compliance**: Adherence to regulations and standards related to surveillance technology, data protection, and ethics is essential.

2.6 User Documentation

1. **Introduction**:
   - Overview and objectives of the system.
   - Target users and prerequisites.
2. **System Overview**:
   - Functionality and key features.
3. **Installation and Setup**:
   - Step-by-step setup instructions.
   - Hardware and software requirements.
4. **User Interface Guide**:
   - Navigation and controls overview.

- Visual aids for better understanding.

5. **Using the System**:
    - Starting, stopping, and configuring tracking.
    - Troubleshooting common issues.
6. **Data Management**:
    - Importing, exporting, and deleting data.
7. **Customization and Advanced Features**:
    - Advanced user options and customization.
8. **FAQs and Troubleshooting**:
    - Common questions and solutions.
9. **Support and Resources**:
    - Contact information and additional resources.
10. **Legal and Compliance**:
    - Legal and compliance information.

2.7 Assumptions and Dependencies

Direction tracking and enumeration system can be updated via new version of algorithm incase of important security batches with new compatible devices or for adding more functionalities. The Direction tracking and enumeration system can detect directions and elaborates the occurrence or frequency of a particular set of object.

3. External Interface Requirements

3.1 User Interfaces

1. **Main Dashboard**:
    - System status overview and key metrics.
    - Real-time visualizations of object counts and movements.
2. **Settings Panel**:
    - Customization options for system parameters and configurations.
3. **Video Playback Interface**:
    - Live or recorded video footage playback controls.
4. **Object Identification Panel**:
    - Displays detected objects and relevant information.
5. **Counting Summary Panel**:
    - Summary of object counts by direction.
6. **Visualization Tools**:
    - Various visualization options for tracking and counting data.
7. **Notification Center**:
    - Alerts about important events or updates.
8. **Help and Documentation**:

- Access to user guides, tutorials, and FAQs.

9. **User Profile and Settings**:
   - Personalization options and user settings.
10. **Error and Warning Messages**:
    - Informative messages for system issues or errors.

3.2 Hardware Interfaces

1. **p**.

3.3 Software Interfaces

1. **Operating System**: Interacts with the underlying OS for resource management.
2. **Communication Protocols**: Uses TCP/IP, UDP, etc., for data exchange.
3. **APIs**: Integrates with libraries like OpenCV, TensorFlow.
4. **Databases**: Stores and retrieves data using SQL, NoSQL, etc.
5. **Web Services**: Communicates with remote services for tasks like live streaming.
6. **UI Libraries**: Utilizes PyQt, Tkinter for GUI development.
7. **Data Formats**: Supports video, image files, JSON, XML.
8. **Interprocess Communication**: Facilitates data sharing between modules.
9. **Logging and Debugging**: Captures system events for troubleshooting.
10. **Algorithm**: YOLO ‘You only look once’ is an object detection software.

4. System Features

4.1 Video Input:

- This component is responsible for accepting video input from various sources such as cameras or video files.
- It interfaces with hardware devices or software APIs to capture live video streams or retrieve recorded video footage.
- The system may support multiple video formats and resolutions to accommodate different input sources.
- Input preprocessing tasks such as frame resizing, format conversion, or frame rate normalization may be performed to standardize the input data.

4.2 Object Detection:

- Object detection is a key component that identifies and tracks cars or people within the video frames.
- It utilizes computer vision algorithms, deep learning models, or a combination of both to detect objects of interest.

- Techniques like Haar cascades, HOG (Histogram of Oriented Gradients), or deep learning-based object detection frameworks such as YOLO (You Only Look Once) or SSD (Single Shot MultiBox Detector) may be employed.
- Object detection algorithms analyze each frame to identify bounding boxes around detected objects, along with confidence scores indicating the likelihood of correct detection.

4.3 Direction Identification:

- Once objects are detected and tracked, the system determines the direction of movement for each object.
- Direction identification algorithms analyze the trajectory of each object over time to estimate its movement direction.
- Techniques such as optical flow analysis, velocity vectors, or angle calculations may be used to determine the direction of movement.
- Object direction is classified into predefined categories such as left, right, up, or down based on the angle or directional vector of movement.

4.4 Counting Mechanism:

- The counting mechanism tallies the number of objects moving in each direction.
- It tracks the movement of each object and updates the count for the corresponding direction as objects move across the frame.
- Real-time or near real-time counting ensures accurate tracking and counting of objects as they move within the video frame.
- The counting mechanism may include features such as filtering out false positives, handling occlusions, or adjusting sensitivity thresholds to improve counting accuracy.

4.5 Visualization:

- Visualization components provide visualizations of the results through the user interface, enhancing the interpretability of the analysis.
- Graphs, charts, heatmaps, or annotated video overlays may be used to represent object movements and counts.
- Visualizations may include real-time updates of object counts, directional distributions, or historical trends for analysis.
- User-friendly interfaces allow users to interact with the visualizations, customize display settings, and gain insights from the analysis results.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The project must achieve real-time processing of video feeds, ensuring synchronization with the input stream's frame rate. It should scale efficiently to handle varying loads, including videos of diverse resolutions, lengths, and object densities. Accurate object detection and direction estimation algorithms are essential for reliable results. Resource optimization is critical, requiring efficient utilization of CPU, memory, and GPU resources. The system must remain robust to lighting changes, occlusions, and background clutter. Low latency is crucial for swift decision-making, necessitating rapid processing from detection to direction determination. Adaptability across environments, camera types, and weather conditions is necessary. Compatibility with multiple video formats and input sources is essential. Effective error handling mechanisms must gracefully manage instances of undetected objects or lost tracking. Security measures should protect the privacy and integrity of video data, especially in sensitive contexts.

5.2 Safety Requirements

Ensuring safety is paramount for the project's success. The system must adhere to rigorous safety requirements to prevent any potential harm or hazards. This includes robust testing procedures to identify and mitigate risks associated with object detection inaccuracies, such as misclassification or false positives. Additionally, the system should incorporate fail-safe mechanisms to handle unexpected errors or malfunctions, ensuring that it can gracefully recover from any disruptions without compromising safety. Furthermore, clear documentation and user instructions must be provided to facilitate safe and proper use of the system, minimizing the risk of accidents or misuse. Regular maintenance and updates are essential to address any emerging safety concerns and maintain the system's reliability over time. Overall, prioritizing safety at every stage of development and deployment is crucial to safeguarding both users and the surrounding environment.

5.3 Security Requirements

The security requirements for the project are crucial for protecting data and ensuring system integrity. This involves implementing robust encryption for data transmission and storage, enforcing strict access control measures, and conducting regular security audits. Monitoring system activity and compliance with relevant regulations are also essential. Continuous updates and patches must be applied to address emerging threats and vulnerabilities. Overall, a proactive approach to security is vital to mitigate risks and maintain the system's reliability.

5.4 Software Quality Attributes

Ensuring software quality attributes security requirements is paramount for system reliability. This entails implementing robust encryption, access controls, and regular security audits. Compliance with industry regulations and continuous monitoring of system activity are vital. Prioritizing security throughout development is essential for maintaining system resilience against evolving threats.

5.5 Business Rules

Business rules are essential guidelines governing system operation within organizational objectives. They encompass data handling protocols, privacy policies, and operational procedures. Compliance ensures adherence to data privacy regulations, operational efficiency, and user satisfaction. Additionally, they dictate adherence to industry standards, pricing models, and contractual obligations, fostering trust and alignment with organizational goals.

6. Other Requirements

Other requirements for the project include intuitive user interface design, comprehensive documentation, support and training provisions, and regulatory compliance. These elements are crucial for enhancing usability, facilitating system deployment and maintenance, providing assistance to users, and ensuring adherence to legal and industry regulations.

Appendix A: Glossary

1. SRS: Software Requirements Specification - A document that describes the intended behavior and functionality of a software system.

2. UI: User Interface - The visual elements and controls through which users interact with a software application.

3. API: Application Programming Interface - A set of rules and protocols that allows different software applications to communicate with each other.

4. CPU: Central Processing Unit - The main component of a computer responsible for executing instructions and performing calculations.

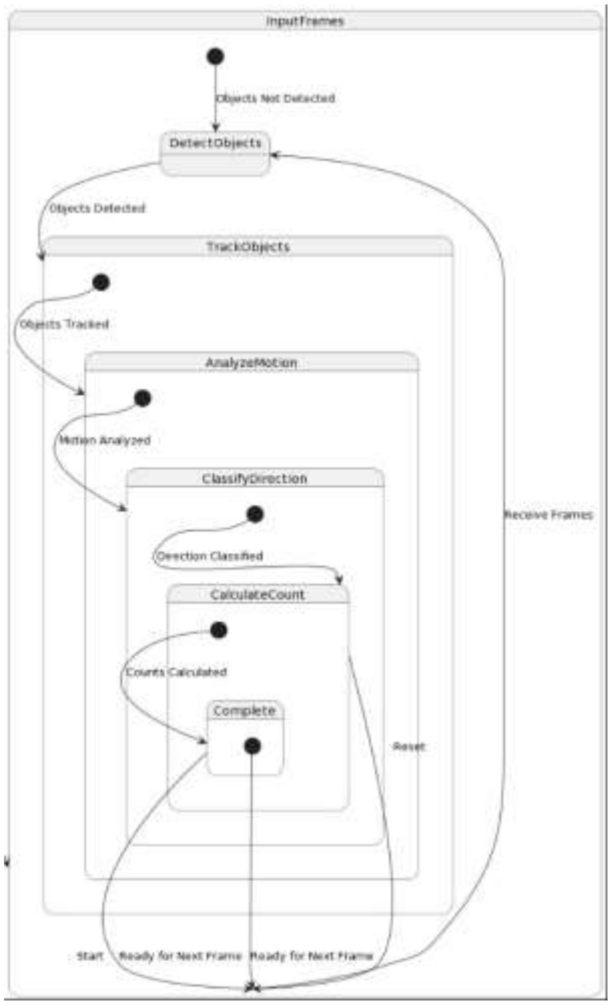
5. GPU: Graphics Processing Unit - A specialized electronic circuit designed to accelerate graphics rendering in computers.

6. TBD: To Be Determined - References in the SRS that require further clarification or decision-making before finalization.

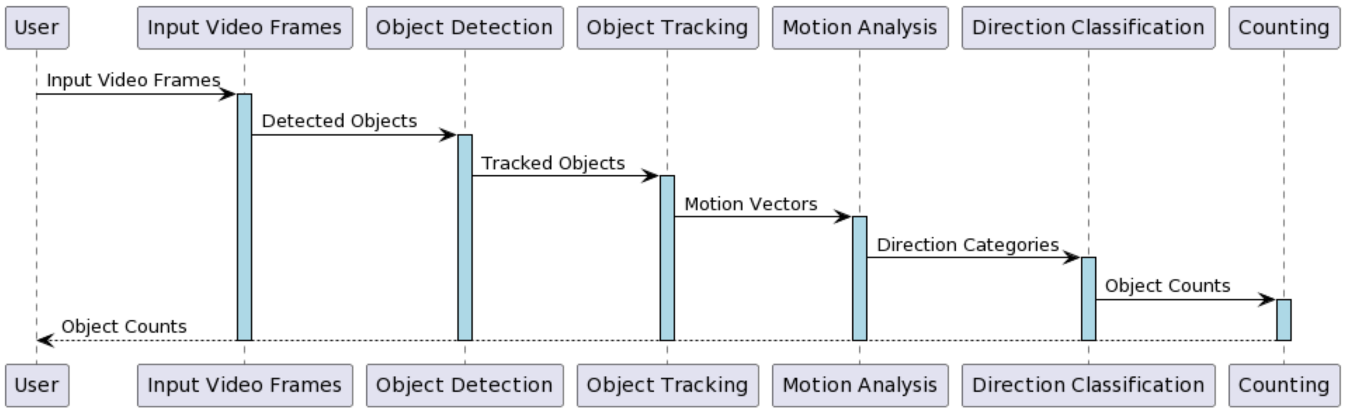
7. GDPR: General Data Protection Regulation - European Union regulation on data protection and privacy for all individuals within the EU and the European Economic Area (EEA).

8. HIPAA: Health Insurance Portability and Accountability Act - United States legislation that provides data privacy and security provisions for safeguarding medical information.

Appendix B: Analysis Models



State Machine Diagram



Time diagram

Appendix C: To Be Determined List

1. User authentication mechanism - TBD

2. Data storage architecture - TBD

3. Error handling strategy - TBD

4. Integration with external systems - TBD

5. Performance testing criteria - TBD