Developing a portable railroad crossing monitoring system based on artificial intelligence and image processing technology

Final Report

by

Yu Qian
Office: (803)777-8184
Email: yuqian@sc.edu
Department of Civil and Environmental Engineering
University of South Carolina

Yuche Chen, University of South Carolina
Dimitris Rizos, University of South Carolina
Gurcan Comert, Benedict College, North Carolina A&T State University

August 2024

Center for Connected Multimodal Mobility (C^2M^2)

200 Lowry Hall
Clemson, SC 29634

DISCLAIMER

ACKNOWLEDGMENT

The research team sincerely thanks the Center for Connected Multimodal Mobility (C2M2) for their partial support of this project. We also extend our gratitude to the CSX Corporation and the City of Columbia for their significant support and invaluable guidance throughout the project.

Technical Report Documentation Page

1. Report No.	**2. Government Accession No.**	**3. Recipient's Catalog No.**
4. Title and Subtitle Developing a portable railroad crossing monitoring system based on artificial intelligence and image processing technology		**5. Report Date** August 2024 **6. Performing Organization Code**
7. Author(s) Yu Qian, Ph.D. ORCID: 0000-0001-8543-2774 Yuche Chen, Ph.D. ORCID: 0000-0003-2577-2448 Dimitris Rizos, Ph.D. ORCID: 0000-0001-5764-7911 Gurcan Comert, Ph.D., ORCID: 0000-0002-2373-5013		**8. Performing Organization Report No.**
9. Performing Organization Name and Address Department of Civil and Environmental Engineering University of South Carolina 300 Main St. Columbia, SC 29208		**10. Work Unit No.** **11. Contract or Grant No.**
12. Sponsoring Agency Name and Address Center for Connected Multimodal Mobility (C2M2) Clemson University 200 Lowry Hall, Clemson, SC 29634		**13. Type of Report and Period Covered** Final Report, September 2023-August 2024 **14. Sponsoring Agency Code**
15. Supplementary Notes (Delete and insert information here or leave blank)		

16. Abstract

This project addresses the urgent need for improved railroad grade crossing safety, where traditional systems often fail to detect and manage unforeseen hazards such as vehicles or pedestrians obstructing tracks. This project develops a cost-effective, portable system for real-time identification, counting, and categorization of objects at railroad grade crossings, including vehicles and pedestrians. The system integrates a cost-effective camera and an edge-computing device, capable of performing object detection and classification in the railroad crossing area. A field-testable prototype has been assembled and proven effective in detecting track intrusion. Utilizing a specialized deep neural network and edge computing platform, the system could enhance collision warnings and inform traffic management and urban planning. Tested on the CDnet 2014 dataset, it achieves an F-measure of 87.67%, surpassing state-of-the-art models in foreground detection. Its deployment on a multi-core inference pipeline reduces latency from 120.82 ms to 49.63 ms and increases the frame rate from 8.28 to 20 FPS, ensuring real-time performance. Field testing validates its ability to identify track intrusions and enhance safety, representing a significant step toward proactive traffic management and reduced trespassing risks at railroad crossings. This approach could shift railroad crossing monitoring from passive to proactive traffic management, aiding urban development and reducing trespassing risks.

17. Keywords Machine vision, object detection, tracking	**18. Distribution Statement** No restrictions.		
19. Security Classif. (of this report) Unclassified	**20. Security Classif. (of this page)** Unclassified	**21. No. of Pages** 33	**22. Price** NA

Table of Contents

List of Tables

List of Figures

EXECUTIVE SUMMARY

Railroad safety remains critically challenged by accidental intrusions at rail crossings, a major cause of fatalities and injuries. Notable incidents, such as the June 2022 collision near Mendon, Missouri, and the June 2023 accident in Lower Richland County, South Carolina, highlight the urgent need for improved safety measures at these crossings. Despite significant advancements over recent decades, including the installation of signaling units and gate arms, collisions at grade crossings continue to result in approximately 250 deaths and 775 injuries annually in the United States.

The primary limitation of current warning systems is their high-cost and limited capability to detect and manage unexpected trespassing or obstructions at crossings. Existing systems alert only to approaching trains but fail to address unforeseen hazards like pedestrians, vehicles, or other objects that may be on the tracks. This gap necessitates the development of a more advanced, real-time monitoring system that enhances railroad safety and operational reliability. In response to this need, the rapid advancement of deep learning and Artificial Intelligence (AI) has led to the development of Convolutional Neural Networks (CNNs) for various computer vision applications, including railroad safety. However, existing models often struggle with detecting unanticipated objects and are typically designed for server-based environments with high-performance GPUs, limiting their applicability in field deployments.

This study presents a portable Railroad Crossing Monitoring System (RCMS) based on artificial intelligence and image processing technology. One core of the RCMS is the YOLOv8-FG model, an enhanced version of the YOLOv8 model, is specifically designed for real-time foreground detection at railroad crossings. Besides, RCMS integrates the capabilities of the Contrastive Language–Image Pre-training (CLIP) (Radford, 2021) and Deep-SORT (Wojke, 2017) algorithms to further extending its functionality for foreground classification and tracking. Thus, the RCMS can simultaneously perform detection, segmentation, classification, and tracking, ensuring comprehensive monitoring of non-compliant objects and unauthorized activities within railroad areas. The YOLOv8-FG model, tested on the CDnet 2014 dataset, achieves the highest F-measure of 87.67%, outperforming leading algorithms in foreground detection. Deployed on the cost-effective Nvidia Jetson AGX Orin platform, the model's efficiency is further enhanced with a multi-core inference pipeline, reducing latency from 120.82 ms to 49.63 ms and increasing FPS from 8.28 to 20. Real-time field testing has demonstrated the model's effectiveness, confirming its suitability for deployment on edge computing devices. This makes RCMS a promising tool for significantly enhancing safety and security at railroad crossings, addressing the limitations of current systems, and ultimately contributing to a proactive approach to railroad crossing management.

Future work could focus on optimizing our model for lightweight deployment on lower-end edge devices by employing techniques like pruning, quantization, and efficient architectures to reduce computational demands. These strategies aim to maintain high detection accuracy and speed while enabling real-time performance on resource-constrained hardware.

CHAPTER 1

Introduction

Railroad safety is critically undermined by accidental intrusions at rail crossings, which are a major cause of accidents. On June 27, 2022, a devastating collision occurred near Mendon, Missouri, when a train collided with a dump truck at a public crossing, resulting in four deaths and around 150 injuries. More recently, on June 13, 2023, another tragic incident took place in Lower Richland County, South Carolina, where a train struck a box truck, leading to one fatality and two injuries. These events highlight the pressing need for improved safety measures at rail crossings.

The Federal Railroad Administration (FRA) reports that the United States has approximately 210,000 public and private grade crossings, with the majority open to public access (Ogden & Cooper, 2019). Since the 1980s, there has been a consistent and significant decrease in collisions at these crossings (Lifesaver, 2023). This positive trend can largely be attributed to safety enhancements such as the installation of signaling units and gate arms at these critical junctions. Despite these considerable improvements in railway safety over recent decades, collisions at grade crossings continue to be a leading cause of railroad-related fatalities. According to FRA data, from 2018 to 2022, the U.S. averaged approximately 2,144 collisions at these crossings annually, resulting in about 250 deaths and 775 injuries each year (Lifesaver, 2023). These statistics underscore the significant societal burden these accidents pose, including disruptions to highway and rail operations and substantial adverse impacts on local economies and communities.

In recent years, North America has experienced a disturbing rise in accidents and fatalities on railroad rights-of-way (ROW), with trespassing incidents contributing to approximately 70% of these cases. Alarmingly, more than 60% of collisions occur at crossings equipped with automatic warning systems, and 34.7% happen at crossings featuring flashing lights and gates (FRA, 2019). This situation highlights a critical need for enhancements in the existing grade crossing warning systems. The main limitation of current systems is that while flashing lights and gate arms signal the presence of an approaching train, they are ineffective in detecting and managing unexpected trespassing or track fouling incidents, which may involve a pedestrian, vehicle, or an unforeseen obstruction on the crossings. In such situations, onboard engineers can only respond to unusual activity at the crossing that falls within their line of sight, often resulting in delayed and ineffective countermeasures.

This underscores the urgent need for an advanced, comprehensive warning system that provides real-time monitoring information at these crossings for both road traffic and incoming trains. Implementing such a system could significantly mitigate many of the risks currently associated with railroad crossings. As a result, this would greatly enhance the safety and reliability of these crossings, ensuring better protection for both rail operators and the general public.

In recent years, the rapid advancement of deep learning and Artificial Intelligence (AI) has seen Convolutional Neural Networks (CNNs) excel in various computer vision applications. Their adoption for enhancing railroad safety and track resilience has grown increasingly popular. CNN-based models can drastically improve detection efficiency and accuracy, minimizing human errors and facilitating auxiliary decision-making. Consequently, significant efforts have been dedicated to employing CNN-based real-time automatic outlier detection systems to enhance situational awareness and prevent accidental intrusions at railroad crossings.

While these earlier networks can successfully detect objects they were trained to recognize, they often struggle with items not included during model training. For instance, networks designed for pedestrian detection might fail to identify vehicles, and while some object detectors can recognize both pedestrians and vehicles, they may overlook obstacles like fallen trees (He, 2017). The unpredictable nature of intruding or trespassing objects—which can range from animals and dropped parcels to collapsed catenary and other unexpected items, in addition to vehicles and pedestrians—poses a substantial challenge, as the cause of an accident is often unforeseen.

Furthermore, most early object detection models were developed for use on servers equipped with high-performance GPUs and a consistent power supply. To our knowledge, few models have been specifically designed for field deployment, considering constraints related to computing resources and power supply. This gap underscores the need for developing robust, adaptable CNN models capable of operating effectively within the limitations of field deployment environments.

To tackle the challenges of object detection in rail crossing monitoring, the RCMS in this project incorporates an enhanced version of the YOLOv8 model (Ultralytics, 2023), named YOLOv8-FG, specifically developed for foreground detection. For further extending the functionalities if RCMS, it integrates the capabilities of CLIP and Deep-SORT for foreground classification and tracking. This comprehensive approach ensures a more accurate and efficient monitoring system capable of identifying and responding to any non-compliant objects or unauthorized activities within the railroad area.

Additionally, real-time field testing of the RCMS has demonstrated its feasibility and effectiveness. These tests confirm the model's suitability for deployment on edge computing devices, offering a robust solution that leverages the advantages of advanced AI capabilities while accommodating the limitations of field deployment environments such as restricted computing resources and power supply. This adaptability makes it a promising tool for enhancing safety and security at railroad crossings

CHAPTER 2
Literature Review

2.1 Object Detection and Foreground Detection

The safety of railway grade crossings has been a perennial concern and an area of intense research focus. However, the advent of artificial intelligence (AI) and machine learning technologies have injected a fresh impetus and research trajectory in this field. For instance, Zaman et al. (2019) employed Mask R-CNN for the detection of intrusion events on railroads. Concurrently, Sikora et al. (2020) introduced a railway crossing surveillance system, utilizing YOLO and SSD networks to monitor vehicles and pedestrians. Building on the principle of swift object detection, Guan et al. (2022) devised a high-speed obstacle detection algorithm for railway images, based on a streamlined YOLO-Tiny network and a swift region proposal. In parallel, Wang & Yu (2021) unveiled an innovative neural network, rooted in the SSD framework, for railway intrusion detection. Additionally, Zhang et al. (2022) developed a YOLO-based framework for the automatic identification of railroad trespassing incidents. However, it is vital to acknowledge that these methodologies predominantly leverage object detection techniques for rail-related monitoring. While they deliver essential functions for intrusion detection, they are fundamentally basic and do not provide an all-encompassing solution for rail crossing monitoring.

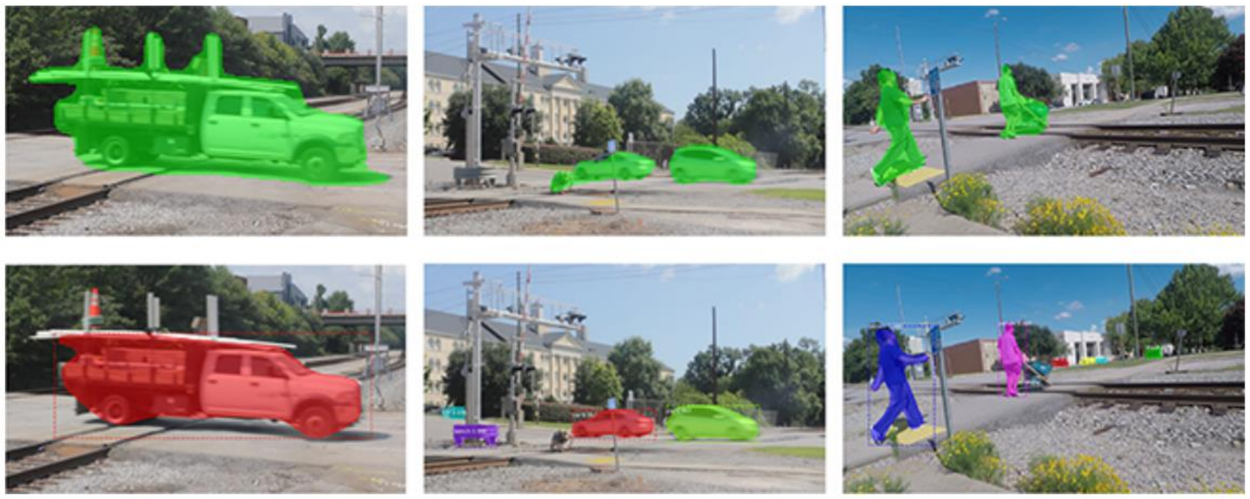


Figure 2.1 Differences between foreground segmentation and object detection (from top to bottom: foreground segmentation, object detection)

Object detection refers to the process of identifying and localizing instances of specific object classes within an image or a video frame (He, 2017). This technique primarily involves the use of deep learning algorithms and convolutional neural networks to recognize and classify different objects, such as cars, pedestrians, or animals, and determine their boundaries or locations within the scene. On the other hand, foreground detection, also known as change detection or background subtraction, is a technique that aims to distinguish the moving elements, referred to as the foreground, from the static scene, referred to as the background (Varadarajan, 2015). This is achieved by analyzing the differences between consecutive frames in a video sequence and identifying the regions with significant changes, which are then classified as the foreground.

The clear illustration of the differences between object detection and foreground detection in **Figure 2.1** shows that while object detection can recognize and classify common objects, such as pedestrians and cars, providing a more detailed understanding of the scene, it may encounter difficulties when faced with uncommon or irregular objects, such as pipes, a squatting person, or a wheelbarrow. This can lead to false-negative errors, where the object is not detected, or false-positive errors, where background objects are mistakenly identified as foreground objects.

On the other hand, Foreground segmentation does not classify objects with limited categories but detects all the outliers shown within a scenery as the Region of Interest (ROI). Therefore, the feature of foreground segmentation is particularly useful in applications where it is essential to detect and monitor the movement of objects within a given environment, such as in traffic monitoring or intrusion detection systems. This could be ideal for crossing monitoring because it could identify all the static and moving outliers within the railroad crossing area.

2.2 CNN-based Foreground Detection Algorithm

Previous research has utilized CNNs to segment video frames into foreground and background regions. As illustrated in **Figure 2.2**, based on the type of network input, these methodologies can be primarily classified into three categories: single-frame based models, multiple-frame based models, and background-frame(s) based models.

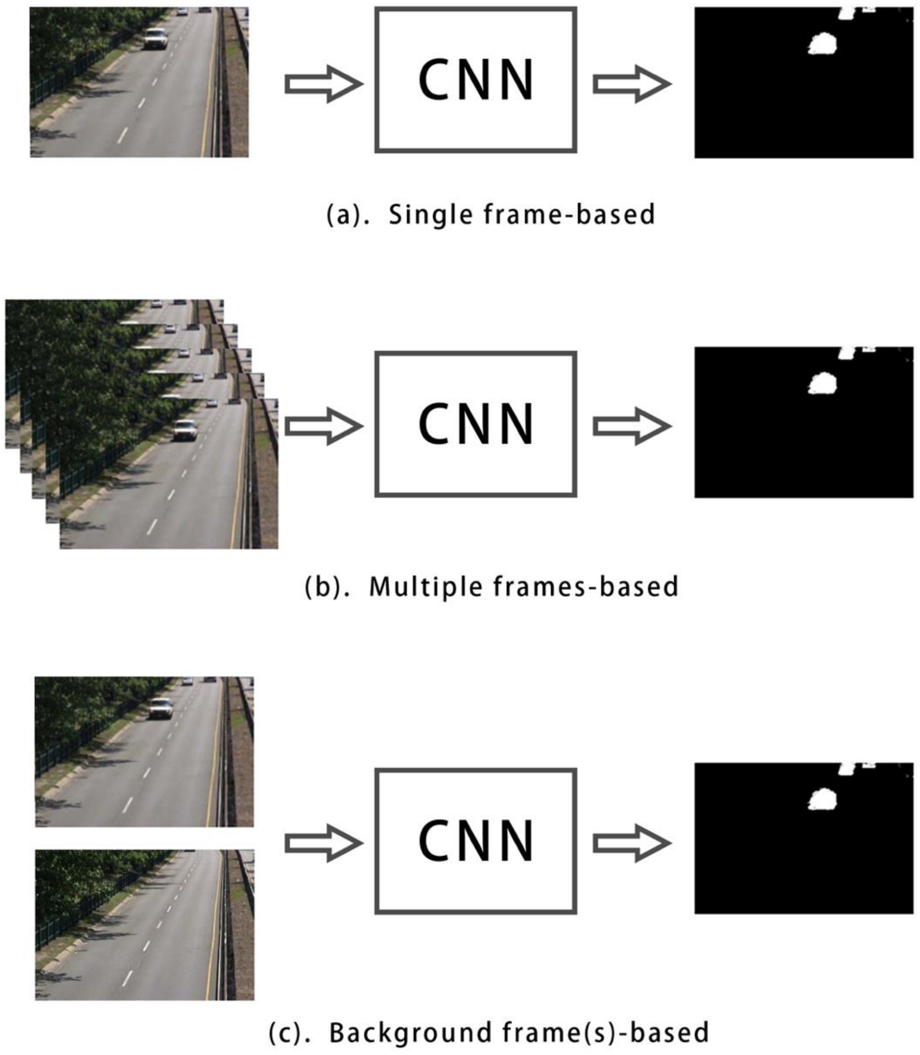


Figure 0.2 Three types of CNN-based foreground segmentation algorithms

Single-frame-based models operate by analyzing each frame independently to detect foreground elements. This approach is characterized by its simplicity and speed, which makes it particularly useful in scenarios where computational resources are limited or rapid response is crucial. The multi-scale architecture proposed by Lim & Keles (2020) exemplifies this method's ability to handle varying object scales within a frame. Rahmon et al. (2021) further enhanced the capability of single-frame models by integrating motion analysis through a motion U-Net, which combines the traditional algorithms of foreground segmentation with CNNs to improve the accuracy of motion detection. Xu et al. (2022) introduced an innovative approach in this category with their cascaded feature-mask fusion network, designed to refine the segmentation process by consecutively processing features and masks within a single frame. This method shows potential in enhancing detail recognition, though, like most single-frame models, it faces limitations when applied to unfamiliar scenes. The model's training on specific types of scenes can lead to a significant drop in performance when confronted with new or varied environments, highlighting a crucial drawback of the single-frame approach.

Expanding on the idea of temporal context, multiple-frames-based models utilize not only the current frame but also surrounding frames (either past or future) to predict foreground segments. This method helps in understanding motion and temporal consistency, which are often missed by single-frame models. For instance, Yang et al. (2017) demonstrated the use of a Fully Convolutional Network (FCN) that processes multiple frames to detect dynamic changes more effectively. Hou et al. (2021) introduced a lightweight, fast 3D CNN that processes video sequences to capture temporal and spatial dynamics simultaneously, providing a robust framework for environments where quick adaptation to changes is necessary. Meanwhile, Valipour et al. (2017) proposed a recurrent CNN approach, using the network's hidden states to maintain a memory of previous frames, thus enhancing the continuity in foreground detection. These models are generally more flexible in new scenes but might struggle with detecting stationary objects that do not vary across multiple frames, posing a challenge in scenarios where such objects could be critical.

The third category involves models that use a background reference frame(s) for comparison with current frames to detect foreground objects. This approach is highly effective in stable environments where the background remains mostly unchanged. Lin et al. (2018) employed an FCN that uses both the generated background and the current frame for segmentation, utilizing advanced background modeling techniques like SuBSENSE (St-Charles, 2014) to adaptively update the background in light of environmental changes. Vijayana et al. (2021) enhanced the recognition rate of moving objects by integrating optical flow analysis with background reference, allowing for a more dynamic understanding of object movements relative to the static background. These models are particularly adept at handling scenarios with a consistent background but may require high-quality background frames to maintain accuracy, a factor that can be limiting in environments with variable backgrounds.

Given the specific needs of rail crossing environments—where safety is paramount, and the ability to detect both static and moving objects is crucial—background frame(s)-based models offer a compelling solution. Rail crossings typically present a relatively stable background, making it easier to implement background frame(s)-based methods effectively. By employing traditional and advanced techniques for background generation, these models can swiftly create accurate background frames, thus ensuring consistent performance without the need for frequent retraining.

2.3 Edge Computing Platform for AI-based Surveillance Systems

AI-based surveillance systems deployed on edge computing devices must navigate a complex interplay of computational requirements, constraints, and optimization strategies to achieve efficiency and reliability. Edge devices are tasked with handling real-time data streams, performing on-device analysis, and ensuring timely responses while operating under significant hardware and environmental limitations.

Real-time processing is a fundamental requirement for AI-based surveillance systems, especially in latency-sensitive applications such as rail crossing monitoring. This entails rapid acquisition and processing of data streams, ensuring that anomalies or critical events are detected and addressed in time. The use of deep learning models for surveillance systems has raised the bar for computational demands. Running inference for these models requires significant computational power, especially when processing real-time image data from cameras.

The constraints of edge devices pose challenges to the effective deployment of the AI-based surveillance systems. One of the most significant limitations is hardware resources. Edge devices are typically built with low-power processors, such as ARM-based architectures, which lack the computational strength of traditional servers. This is compounded by limited RAM, which restricts the size of deep learning models and data that can be processed simultaneously. Furthermore, intermittent or low-bandwidth network connections in remote or mobile deployments restrict the feasibility of offloading heavy computations to the cloud, necessitating a balance between local processing and periodic data synchronization.

To overcome these challenges, optimizing the system is essential to enhance the performance of AI-based surveillance systems on edge devices. A key strategy involves employing lightweight deep learning models to reduce computational demands on resource-constrained hardware. Additionally, leveraging data parallelization can fully utilize the capabilities of multi-core computational platforms, thereby maximizing the efficiency of the deep learning model inference pipeline.

In this project, Jetson AGX Orin depicted in **Figure 2.3**, an edge computing platform was selected for model inferencing and deployment. Measuring just 100mm x 87mm x 70mm and weighing only 918 grams, it is specifically tailored for mobile or space-constrained environments where size and weight are critical considerations. This device is powered by a 12-core ARM CPU and is equipped with a GPU containing 2048 CUDA cores and 64 tensor cores. These components are built on the advanced Ampere architecture and operate at a frequency of 1.3 GHz, delivering a potent combination of efficiency and computing power. The AGX Orin achieves computational capabilities of up to 5.3 TFLOPs, which is significant for edge computing applications that require intensive data processing. Additionally, the device is designed to operate within a power envelope of 30 watts under normal conditions and can go up to 60 watts at peak performance.

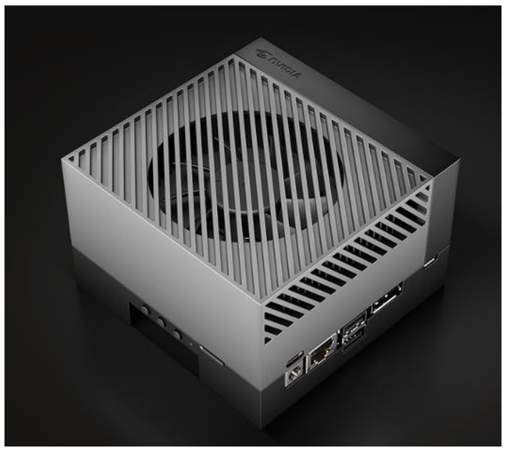
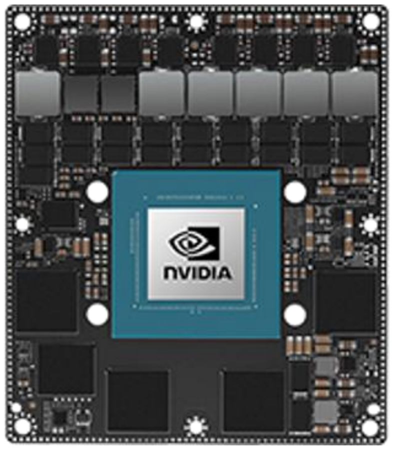



Figure 1.3 The edge device used in the work (Left: Nvidia Jetson AGX Orin; Right: Nvidia Jetson AGX Orin Core)

Deploying monitoring systems on edge computing devices requires a careful balance between computational requirements, hardware constraints, and optimization strategies. By leveraging lightweight models, efficient algorithms, hardware accelerators, and adaptive resource management, developers can overcome the inherent limitations of edge devices while delivering reliable, real-time monitoring solutions across diverse applications.

CHAPTER 3
Methodology

3.1 Overall Approach

Figure 3.1 showcases the inference pipeline of the RCMS. This system incorporates YOLO-FG and SuBSENSE for foreground detection. The SuBSENSE is a classical background modeling algorithm, while YOLO-FG is based on YOLOv8 network (Ultralytics, 2023). This advanced iteration introduces a novel component known as the input head, designed specifically to enhance the network's ability to identify and discern foreground objects within complex visual scenes. The primary innovation of the YOLOv8-FG model lies in its ability to effectively compare differences between current frames and static background frames, thereby isolating dynamic or static objects that constitute the immediate foreground. This capability is crucial for applications where precise detection of moving or relevant objects against a cluttered or changing background is essential.

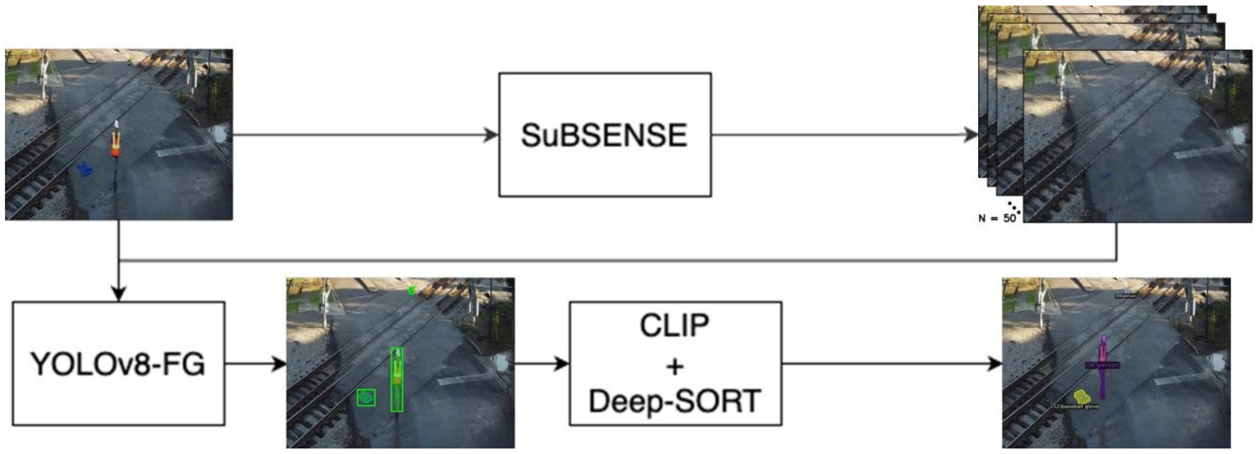


Figure 3.1 Overall approach of proposed Railroad Crossing Monitoring System

Moreover, RCMS integrates the capabilities of the Contrastive Language–Image Pre-training (CLIP) (Radford, 2021) and Deep-SORT (Wojke, 2017) algorithms into its architecture, further extending its functionality. The CLIP algorithm leverages vast amounts of textual and visual data to understand and classify images in a manner that mimics human visual and cognitive abilities. This integration allows to not only detect but also interpret and classify foreground objects in a context-aware manner, enhancing its applicability in scenarios requiring nuanced understanding of the scene.

The Deep-SORT algorithm complements this by providing robust tracking capabilities. It employs advanced data association techniques and motion prediction algorithms to track the trajectories of detected objects across successive frames. This feature is particularly beneficial in dynamic scenes where objects are in constant motion, ensuring that the monitoring of target entities remains consistent and reliable throughout the sequence of frames.

3.2 SuBSENSE

The SuBSENSE algorithm (St-Charles, 2014), as employed in the RCMS for foreground segmentation and background generation, is a sophisticated tool designed to tackle the complexities of dynamic video environments. This method is highly adaptive, leveraging advanced techniques to discern between foreground and background elements in each video frame it processes.

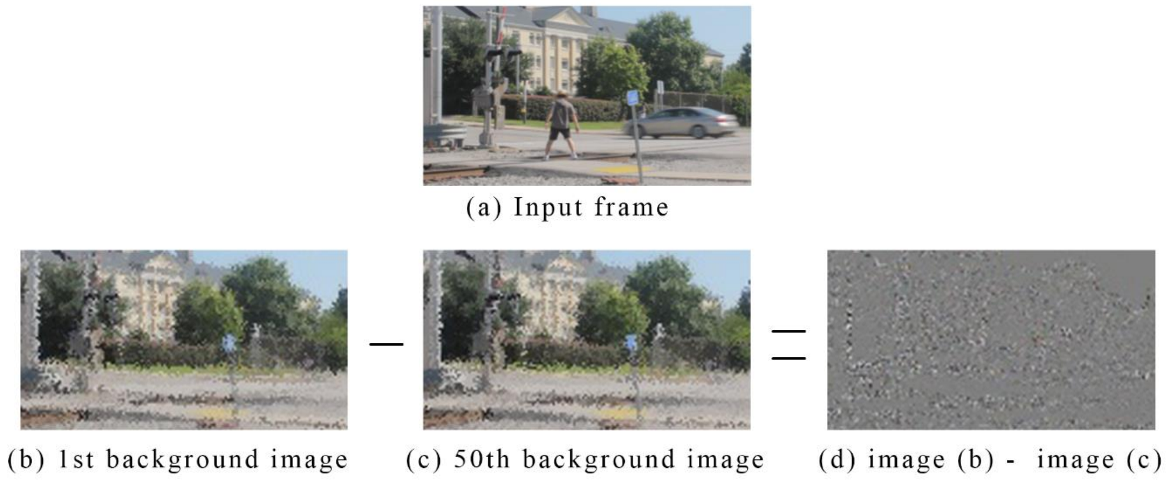


Figure 3.2 Example of the first and the last background images generated with SuBSENSE

Upon receiving a video frame, the SuBSENSE algorithm meticulously analyzes every pixel. It employs a dual-criterion approach to determine the nature of each pixel. The first criterion involves evaluating the pixel's RGB color values, which provide direct color information about the pixel. The second criterion uses Local Binary Similarity Pattern (LBSP) features, which are a form of texture descriptor that captures the local spatial pattern and contrast features of the pixel. These two data points together allow SuBSENSE to perform a robust comparison between the current pixel and a library of background images.

The background library is a central component of SuBSENSE's operation. It contains 50 background images, which are continuously updated and maintained to reflect the dynamic nature of the video environment. When deciding whether a pixel belongs to the background or the foreground, the algorithm checks for consensus among these images. If at least two of the background images agree that a pixel should be classified as part of the background, then it is deemed a background pixel. If not, it is classified as foreground.

The update mechanism of the background library is conservative yet effectively adaptive, employing a stochastic, two-step approach. When a pixel is confirmed as part of the background, one of the background images in the library has a chance (determined by the hyperparameter T) of updating that specific pixel to match the current frame's pixel. Furthermore, this update process may extend to one of the neighboring pixels as well, thus allowing the background model to adapt gradually and smoothly over time.

This stochastic nature of the updating process results in the generated background images often appearing blurry, as they represent a probabilistic consensus of many frames over time rather than a sharp snapshot of any single moment. To illustrate the subtle changes that occur in the background library, **Figure 2.3** compares the first and the last generated background images. Given the difficulty in discerning differences between these images with the naked eye due to their blurred characteristics, a differential image is created by subtracting these two background images. This differential image effectively highlights the small, often imperceptible changes that occur, providing a clearer understanding of how the background evolves in response to changes within the video stream.

3.3 YOLOv8-FG

3.3.1 Input head and its background generation model

To achieve foreground detection effectively, a nuanced enhancement is introduced by adding a specialized input head to the established YOLOv8 architecture (Ultralytics, 2023), resulting in the YOLOv8-FG model. This modification allows the model to process both the current video frame and background images simultaneously, setting the stage for accurate foreground detection. The innovative approach involves leveraging the differences between the current frame and the background, which is continually updated using the SuBSENSE algorithm, a robust method for background generation.

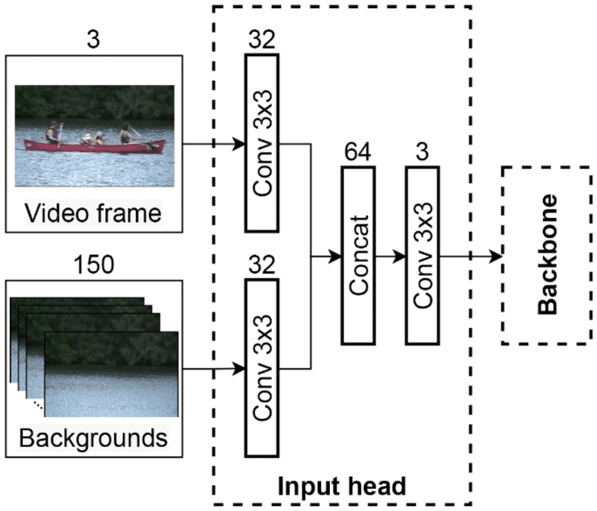


Figure 3.3 Input head structure of YOLO-FG

Figure 3.3 illustrates the sophisticated structure of the input head, which plays a pivotal role in ensuring the equilibrium of the data inputs. Specifically, the video frame is a standard RGB image with three channels, whereas the background images, derived from the SuBSENSE algorithm (St-Charles, 2014), consist of an ensemble of 50 RGB images, cumulatively presenting 150 channels. This vast difference in channel quantity could potentially skew the model's attention towards the background, overshadowing the video frame. Such an imbalance might impede the model's ability to discern subtle differences between the foreground and the background, crucial for accurate detection.

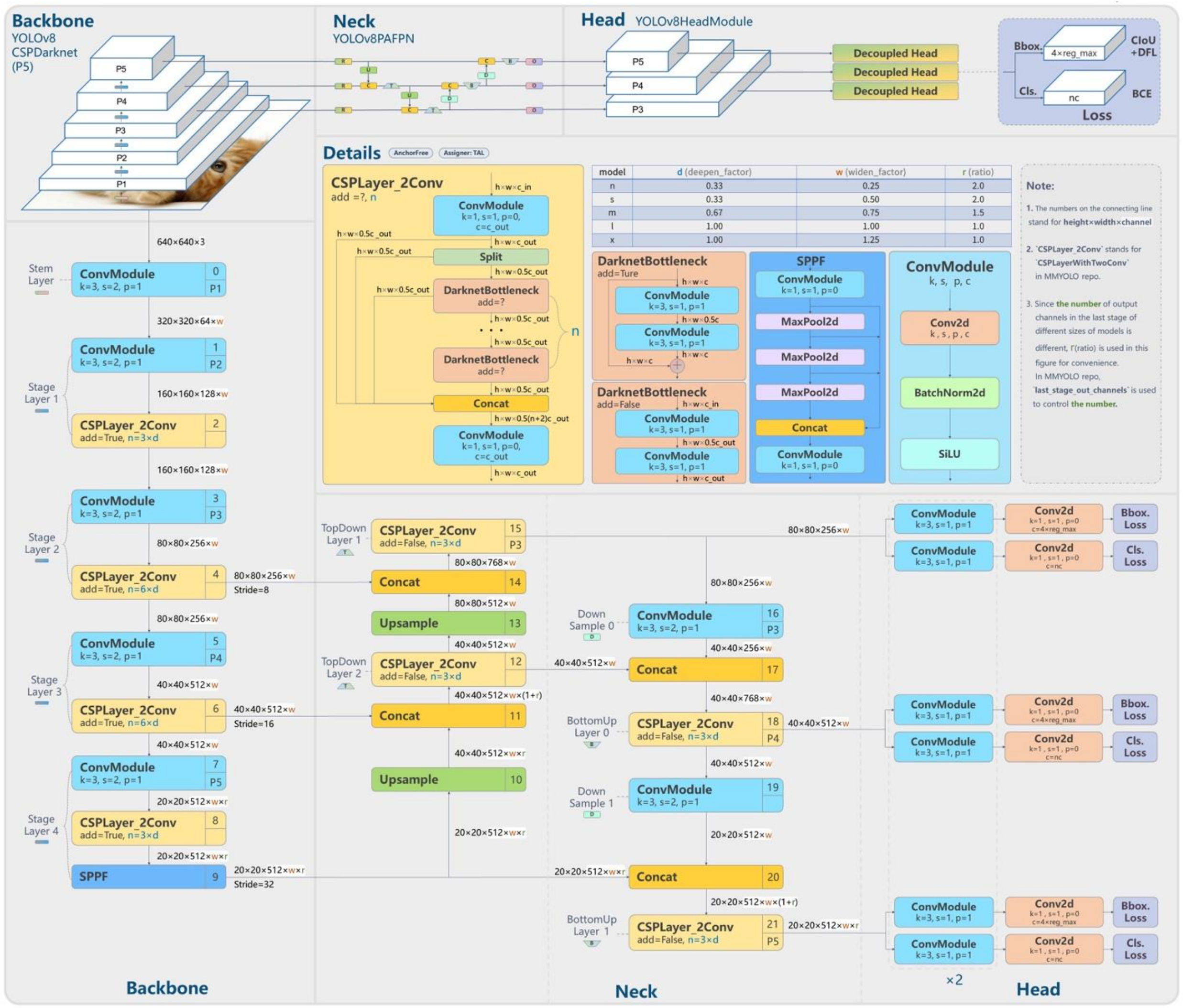
To counteract this potential bias, the input head is designed to harmonize the weight between the video frame and the background images. It accomplishes this by transforming both types of inputs into feature-maps of identical size. The input head features two distinct branches, each equipped with a single convolutional layer. The first branch processes the input frame, and the second manages the 50 background images. Through these branches, both the input frame and the background images are converted into a standardized set of feature maps, each with 32 channels.

Following this conversion, these two sets of feature maps are concatenated to form a combined 64-channel feature map. This concatenated feature map undergoes a final convolutional transformation to compress it into a 3-channel output. This compressed output, while retaining essential information, is designed to be compatible with the backbone of the YOLOv8 architecture without necessitating any further modifications. This elegant solution not only maintains the integrity and efficiency of the original YOLOv8 design but also enhances its capability to perform foreground detection with high precision. This approach ensures that the

YOLOv8-FG model remains robust and adaptable, capable of handling diverse and dynamic visual environments.

3.3.2 YOLOv8

YOLOv8-FG is built upon the YOLOv8, which is a significant enhancement from its predecessor, YOLOv5 (Jocher, 2022). Based on the official documentation and source code, key upgrades in YOLOv8 include a new backbone network and an innovative anchor-free detection head.



model	d (deepen_factor)	w (widen_factor)	r (ratio)
n	0.33	0.25	2.0
s	0.33	0.50	2.0
m	0.67	0.75	1.5
l	1.00	1.00	1.0
x	1.00	1.25	1.0

Note:
1. The numbers on the connecting line stand for height×width×channel
2. 'CSPLayer_2Conv' stands for 'CSPLayerWithTwoConv' in MMYOLO repo.
3. Since the number of output channels in the last stage of different sizes of models is different, r(ratio) is used in this figure for convenience. In MMYOLO repo, 'last_stage_out_channels' is used to control the number.

Figure 3.4. YOLOv8 network architecture (Ultralytics, 2023)

As shown in **Figure 3.4**, the backbone of YOLOv8-FG has seen a transformative update, most notably through the replacement of the C3 modules with the novel C2f modules. This new configuration leverages principles from the Efficient Layer Aggregation Network (ELAN), significantly enhancing the model's efficiency and performance. The introduction of C2f modules results in improved gradient information flow through the network, which in turn reduces the overall parameter count and boosts the computational efficiency. These improvements not only yield a quicker model but also enhance its performance by streamlining the detection process, leading to faster real-time applications.

The Spatial Pyramid Pooling Fusion (SPPF) module in YOLOv8 remains a critical component, inherited from YOLOv5. The SPPF module, which builds upon the traditional Spatial Pyramid Pooling (SPP) design, allows the network to handle variable input image sizes more efficiently. This flexibility greatly enhances the model's generalization capabilities across different visual contexts, making it robust against various scales and dimensions of input data. The optimization of processing speed within the SPPF module ensures that the model can perform efficiently without compromising on detection accuracy.

A major innovation in YOLOv8 is the overhaul of the detection head, transitioning from an anchor-based approach used in YOLOv5 to a new anchor-free methodology. This paradigm shift allows each pixel in an image to directly predict the coordinates of bounding boxes without relying on predefined anchor boxes. This change simplifies the learning process and reduces the complexity of the model. Additionally, by separating the box prediction and classification tasks into two distinct branches, YOLOv8 minimizes the conflict in learning these two different tasks simultaneously, thus enhancing the overall effectiveness and accuracy of the model.

3.4 CLIP

Empowering the foreground detection capacity to YOLOv8-FG deprives its object classification feature, which is the fundamental nature of object detection. However, as a rail crossing monitoring algorithm, it is essential to know the types of foreground objects to distinguish if it is a train or other objects like cars, pedestrians, etc. The train will not trigger the alarm, but other objects will if they stay in the rail crossing area for too long. Therefore, this study further integrates CLIP (Radford, 2021) to classify the detected foreground objects. By leveraging CLIP's robust image classification capabilities, the algorithm can accurately identify various types of objects in the monitored area. This approach ensures that trains passing through the crossing do not trigger unnecessary alarms, while other objects that pose potential risks are effectively detected and managed.

Contrastive Language–Image Pre-training (CLIP) is a model developed by OpenAI that leverages large-scale pre-training to learn visual concepts from natural language descriptions. As shown in **Figure 3.5**, The fundamental idea behind CLIP is to use contrastive learning, which aims to associate images and their corresponding textual descriptions in a shared latent space. This approach allows CLIP to perform a wide range of visual classification tasks without task-specific fine-tuning.

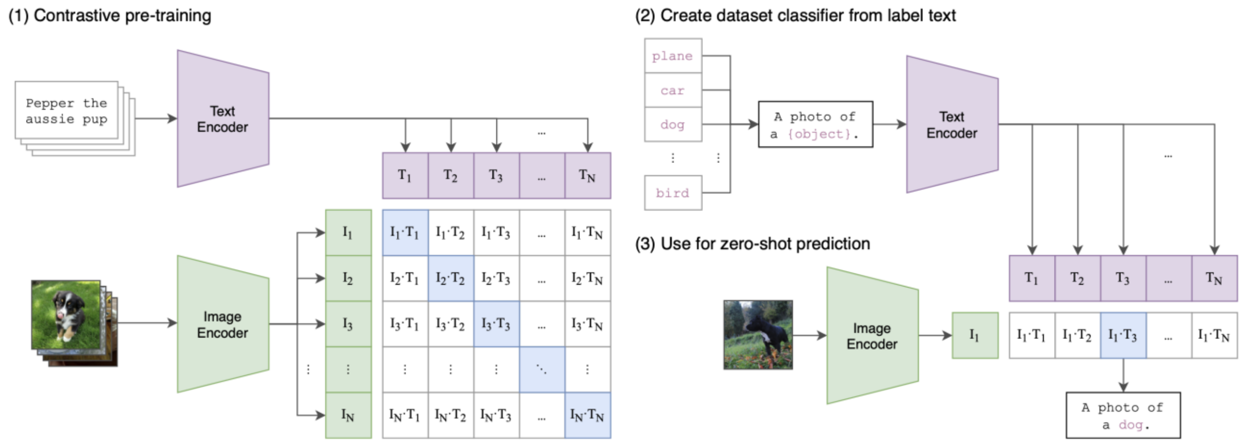

(1) Contrastive pre-training
Pepper the aussie pup
Text Encoder
Image Encoder
T1 T2 T3 ... TN
I1 I2 I3 ... IN
I1·T1 I1·T2 I1·T3 ... I1·TN
I2·T1 I2·T2 I2·T3 ... I2·TN
I3·T1 I3·T2 I3·T3 ... I3·TN
IN·T1 IN·T2 IN·T3 ... IN·TN
(2) Create dataset classifier from label text
plane
car
dog
bird
A photo of a {object}.
Text Encoder
(3) Use for zero-shot prediction
Image Encoder
I1
I1·T1 I1·T2 I1·T3 ... I1·TN
A photo of a dog.

Figure 3.5, Contrastive Language–Image Pre-training (CLIP) model architecture (Radford, 2021)

One of the significant advantages of CLIP is its ability to perform zero-shot learning, especially in open-world image classification scenarios. Zero-shot learning refers to the model's capability to recognize and classify images from categories it has never seen during training. In open-world image classification, the model encounters a diverse and potentially infinite set of categories. Traditional models struggle with such tasks because they require extensive labeled data for each category. However, CLIP overcomes this limitation by leveraging its learned associations between images and textual descriptions.

As shown in **Figure 3.4**, in practical terms, when given an image, CLIP can generate an embedding for it using the image encoder. Simultaneously, for a set of potential categories, the model can generate embeddings from their textual descriptions using the text encoder. By comparing the image embedding with the category embeddings, the model can identify the category with the highest similarity, effectively classifying the image without needing specific training on those categories. This ability to generalize from textual descriptions makes CLIP particularly powerful for open-world classification, where it can adapt to new categories on the fly.

3.5 Deep-SORT

YOLOv8-FG and YOLOv8 are foreground and object detectors, and they are designed to work on a per-frame basis, detecting objects within individual video frames. However, one of the core functions of this study is to identify outliers that remain in the rail crossing area for extended periods. To accomplish this, it is essential to employ a tracking algorithm that can consistently associate the same objects across successive frames. This tracking capability allows for the monitoring of objects over time, thereby enabling the detection of those that linger in the rail crossing area longer than expected, which is crucial for identifying potential hazards or anomalies.

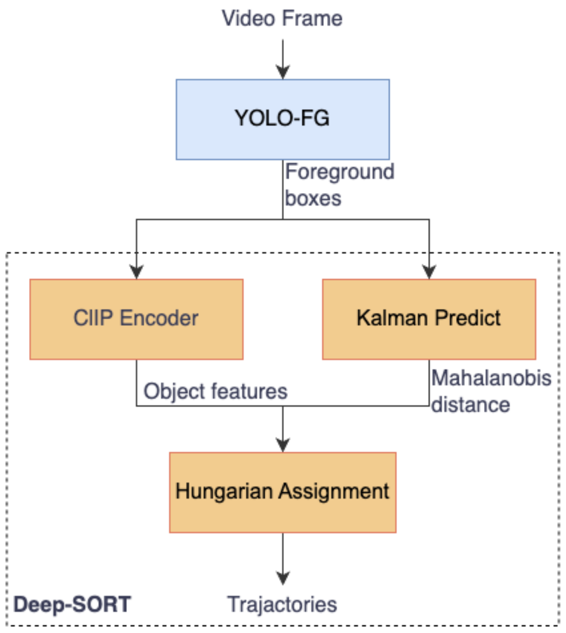


Figure 3.6, Flow chart of Deep-SORT

Deep SORT is a sophisticated tracking algorithm used primarily in computer vision tasks to track multiple objects in videos or image sequences (Wojke, 2017). It's an extension of the SORT algorithm that incorporates deep learning features for better object recognition and tracking.

As shown in **Figure 3.6**, once the foreground objects are detected by YOLOv8-FG, Deep SORT extracts deep features from these bounding boxes. Instead of using the raw pixel values, it passes the cropped regions corresponding to each bounding box through a deep neural network (e.g., ResNet) to obtain a high-dimensional feature vector for each object. Given that the CLIP model was already employed for object classification in the previous section, we can utilize the feature vectors produced by the CLIP encoder directly. This integration eliminates the need for an additional network for feature extraction.

Once these features are obtained, Deep SORT employs a data association algorithm that integrates the Kalman filter—a mathematical technique used for estimating the state of a linear dynamic system from a series of incomplete and noisy measurements. The Kalman filter is adept at predicting the future state (position and velocity) of each object based on its previously known trajectory. This prediction is crucial for handling frame-to-frame associations under motion.

With the predicted positions in hand, the next step in the Deep SORT pipeline involves the actual association of these predicted tracks with new detections in the current frame. To achieve this, the algorithm utilizes the Hungarian algorithm, an optimization approach that solves the assignment problem in polynomial time. The Hungarian algorithm compares the predicted tracks to new detections by considering both their spatial proximity, as measured by the Intersection over Union (IoU) of their bounding boxes, and their appearance similarity, as quantified by the cosine distance between their feature vectors.

The combination of these two metrics—IoU for spatial alignment and feature vector similarity for appearance—ensures that the tracking is both accurate and resistant to errors caused by occlusion or similar-looking objects entering the scene. This robust method of track-detection association allows Deep SORT to maintain consistent object identities across frames, even in complex, dynamic environments.

3.6 Inference Pipeline Optimization

In the context of advancing real-time object detection technologies, the RCMS presents a complex integration of techniques aimed at enhancing foreground detection, classification, and tracking. Incorporating sophisticated modules such as Subsense, CLIP, and Deep-SORT, each adding a layer of computational demands which, in turn, contribute to the potential bottleneck in processing speed when deployed in a sequential inference framework.

To elucidate, in a traditional single-threaded pipeline, a singular processing unit is responsible for executing all tasks sequentially. This process, as depicted in **Figure 3.7**, inherently leads to an accumulation of latency as each task must wait for the previous one to complete before it can begin. This setup, although straightforward, is not optimized for speed, especially in applications requiring real-time processing, such as autonomous driving systems or advanced surveillance technologies.

To counter these challenges, our study proposes a restructured approach to the inference pipeline of RCMS by leveraging a multi-processor model. This model utilizes the inherent capabilities of multi-core processing platforms, organizing the workflow into a parallel processing architecture. Here, different stages of the pipeline—namely foreground detection, object classification, and motion tracking—are assigned to separate processors.

Each processor operates independently, processing its designated task simultaneously with others. This division of labor is coordinated through the use of queues which facilitate the transfer of data between processors. These queues function akin to conveyor belts in an industrial assembly line, where each processor or 'worker' is responsible for a specific segment of the overall task. By enabling each segment of the pipeline to operate concurrently, the overall system latency is significantly reduced.

Furthermore, this multi-processor approach not only minimizes idle time but also ensures that the utilization of CPU resources is maximized. The performance of such a pipeline is constrained primarily by the slowest segment—often a factor of the computational complexity of a particular task and the efficiency of data transfer between processors. Nevertheless, even with these constraints, the multi-processor model demonstrates a marked improvement in processing speed compared to the traditional single-threaded approach.

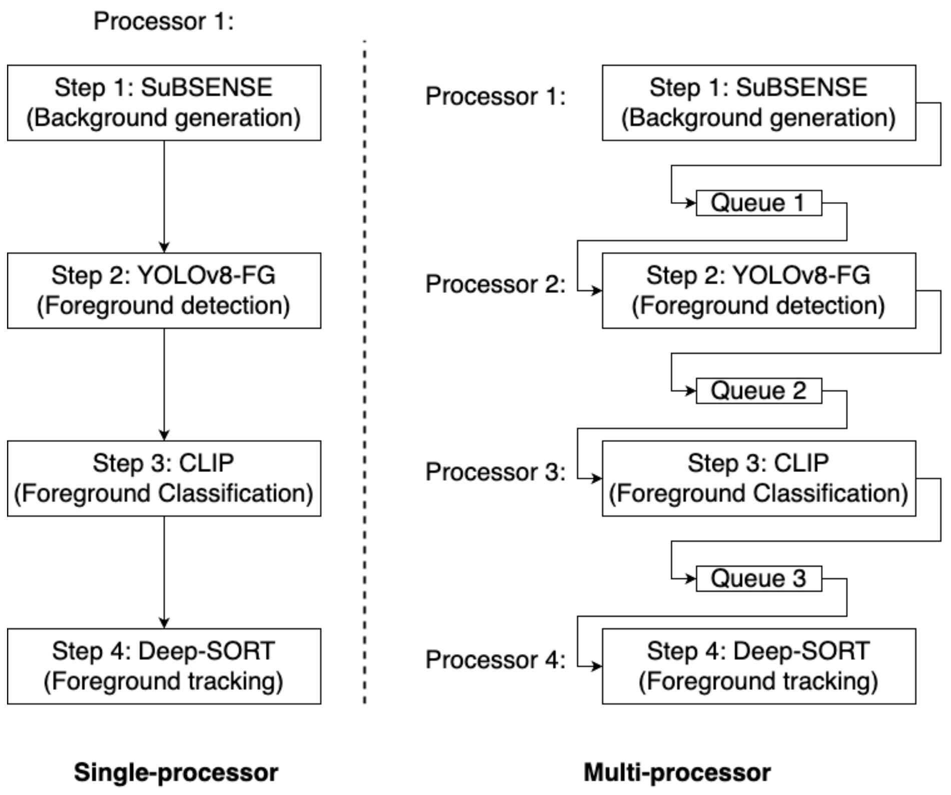

Processor 1:
Step 1: SuBSENSE (Background generation)
Step 2: YOLOv8-FG (Foreground detection)
Step 3: CLIP (Foreground Classification)
Step 4: Deep-SORT (Foreground tracking)
Processor 1:
Step 1: SuBSENSE (Background generation)
Queue 1
Processor 2:
Step 2: YOLOv8-FG (Foreground detection)
Queue 2
Processor 3:
Step 3: CLIP (Foreground Classification)
Queue 3
Processor 4:
Step 4: Deep-SORT (Foreground tracking)
Single-processor
Multi-processor

Figure 3.7. Single-processor vs multi-processor pipeline

CHAPTER 4
Model Development and Evaluation

4.1 Evaluation Methods

The RCMS is tailored for foreground detection, segmentation, classification and tracking, but its evaluation is currently constrained by the limited availability of comprehensive datasets. In particular, the model relies on the CDnet 2014 (Wang, 2014) dataset for testing purposes.

The CDnet 2014 dataset is a widely used benchmark in the field of change detection and foreground segmentation. Despite its richness in terms of content, CDnet 2014 only provides ground-truth data for foreground segmentation, lacking category and track ID information for each instance. Consequently, this limitation prevents a thorough evaluation of the YOLO-FG's mask head, and only the F-Measure metric can be used for performance assessment.

To showcase the capabilities of the CLIP and Deep-SORT, we will provide a series of demonstrations and qualitative results to highlight the RCMS's ability to accurately identify and track multiple objects of different categories in complex scenes.

As shown in **Eq. 1**, F-Measure is the harmonic mean of precision and recall; a high F-Measure reflects high precision and high recall. Precision is the percentage of detected foreground pixels correctly classified, and it can evaluate the anti-noise ability of the network, as shown in **Eq. 2**. Recall is the percentage of true foreground pixels classified correctly, and it can estimate the foreground recognition rate of the network, as shown in **Eq. 3**.

$$\mathrm{F-Measure} = \frac{2 \times Precision \times Recall}{Precision + Recall}, \tag{1}$$

$$Precision = \frac{TP}{TP+FP}, \tag{2}$$

$$Recall = \frac{TP}{TP+FN}, \tag{3}$$

where TP, FP, and FN are the numbers of true-positive, false-positive, and false-negative pixels, respectively.

4.2 Experiments on the CDNet 2014 Dataset

The experiments conducted on the CDnet 2014 dataset aim to evaluate the performance of the proposed YOLOv8-FG model for foreground segmentation in a variety of challenging scenarios. The YOLOv8-FG, a state-of-the-art deep learning model, is trained using a 7:3 split of the dataset—70% of the frames are utilized for training purposes while the remaining 30% serve as the test set. The CDnet 2014 dataset is an exemplary choice for testing such models due to its diverse array of video environments and scenarios, ranging from low-light conditions to dynamic weather situations, which effectively validate the robustness and adaptability of the YOLOv8-FG model.

In the evaluation process, the foreground segmentation results of the proposed YOLOv8-FG model are compared with those from several other leading algorithms. These include both contemporary deep learning models like RCSAFE, SimpleBSC, and DeepBS, and classical image processing methods such as SuBSENSE, PAWCS, and PBAS. In these comparisons, which are detailed in **Table 4.1**, the YOLOv8-FG model distinguishes itself by achieving the highest F-measure, a robust metric of accuracy, reaching an impressive 87.67%. This statistic not only underscores the efficacy of the YOLOv8-FG in foreground segmentation but also highlights its superiority over existing methods.

Table 4.1. F-measure comparison of different foreground algorithms among CDnet2014 dataset

Video categories	**YOLOv8-FG**	**RCSAFE (20)**	**SimpleBSC *(29)***	**DeepBS *(30)***	**SuBSENSE *(2)***	**PAWCS *(31)***	**PBAS *(32)***
Baseline	**96.16**	95.67	96.90	95.80	95.03	93.97	92.42
Bad weather	**92.34**	79.24	72.20	83.01	86.19	81.52	76.73
Dynamic background	**91.02**	82.19	82.30	87.61	81.77	89.38	68.29
Camera jitter	**87.08**	80.45	60.50	89.90	81.52	81.37	72.20
Intermittent object motion	78.32	**82.35**		60.98	65.69	77.64	57.45
Low framerate	**81.85**	82.14		60.02	64.45	65.88	59.14
Night videos	**70.48**	64.53	37.70	58.35	55.99	41.52	43.87
shadow	**94.09**	93.79	56.00	93.04	89.86	89.13	81.43
thermal	**94.59**	82.47	66.10	75.83	81.71	83.24	75.56
turbulence	**90.77**	77.47	57.80	84.55	77.92	64.50	63.49
Overall (exclude PTZ)	**87.67**	82.03	66.20	78.91	77.01	76.82	69.06

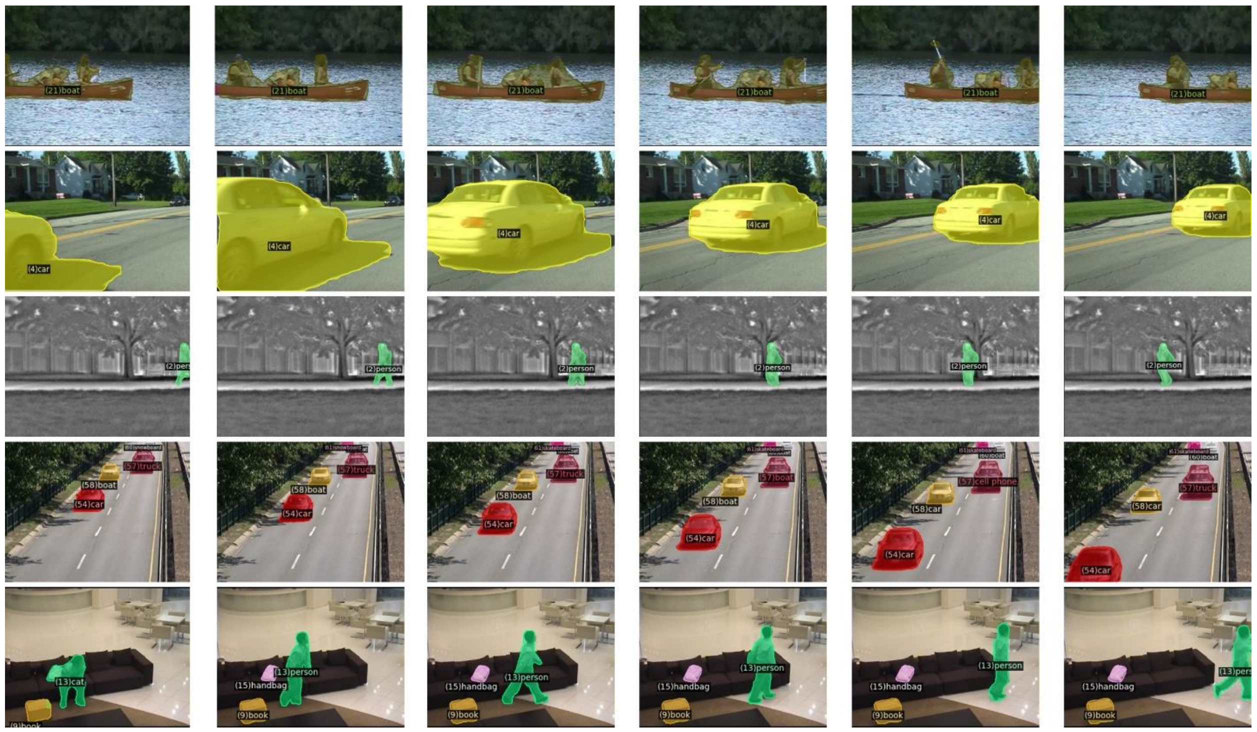


Figure 4.1. Example detection results on CDnet 2014 dataset

Further demonstrating the capabilities of the YOLOv8-FG model, **Figure 4.1** presents a series of detection results showcasing the model's effectiveness in real-world applications. Each detected object in the visual results is highlighted by a distinctively colored bounding box, which encapsulates the object's class label and tracking number. These visual aids illustrate the model's remarkable prowess in detecting and segmenting foreground objects from their backgrounds, accurately classifying them into their respective categories, and consistently tracking their movement across frames. This series of results not only provides a visual proof of the model's operational success but also serves as a clear indication of its potential applications in complex environments where accurate and reliable object detection and tracking are crucial.

4.3 Multi-processor Pipeline Implementation

Optimizing the inference speed of RCMS systems on edge devices such as the Nvidia Jetson AGX Orin is crucial for applications requiring real-time processing and analysis. The Jetson Orin, a compact edge-computing device measuring 110mm x 110mm x 71.65mm and weighing 918g, is an ideal platform for such tasks due to its robust computing capabilities and energy efficiency. It is powered by a 12-core ARM CPU and is equipped with 2048 CUDA cores and 64 tensor cores. The device operates with normal and peak power consumptions of 30W and 60W, respectively, balancing performance with energy consumption.

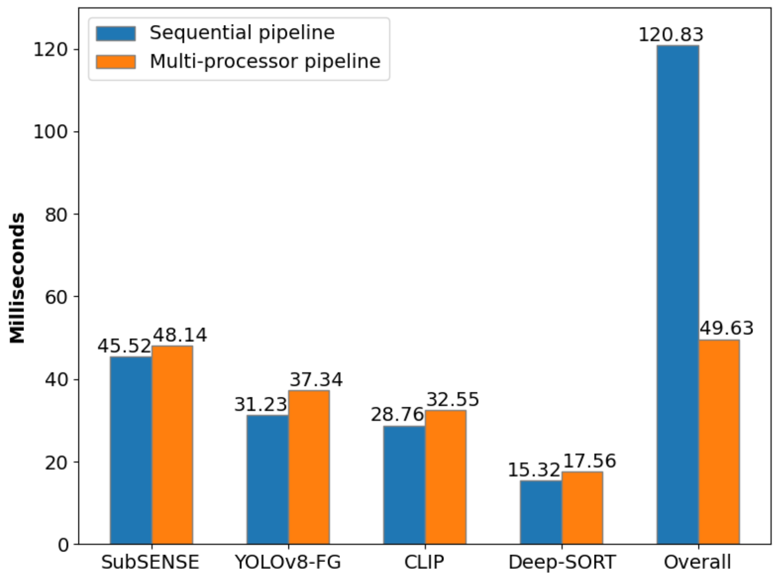


Figure 4.2 Comparison of sequential pipeline and multi-processor pipeline.

A detailed analysis of the device's performance, as presented in **Figure 4.2**, demonstrates some surprising results when comparing different processing pipelines. The single-threaded CPU pipeline processes tasks sequentially, moving from one processing phase to another. It starts with SubSENSE, progresses through various analysis stages, and ends with Deep-SORT. This pipeline results in a cumulative system latency of 120.82 milliseconds, translating to an inference speed of approximately 8.28 frames per second (FPS). The limitation here is evident as the pipeline does not leverage the multi-core architecture of the CPU, relying instead on a single thread for execution. This approach underutilizes the CPU's capabilities, particularly in multi-core processing and threading, which are essential for optimizing performance in complex computational tasks.

In contrast, the multi-processor pipeline adopts a parallel processing approach. Each segment of the pipeline is assigned to a separate CPU core, allowing simultaneous operation and significantly reducing overall latency. This methodology maximizes CPU utilization and enhances the system's inference speed by effectively distributing the workload across available resources. The key challenge in a multi-processor pipeline, however, is that its efficiency is dependent on the performance of its slowest segment. In this case, the SubSENSE segment has been identified as the slowest, with a latency of 48.14 milliseconds. Despite this bottleneck, the multi-processor pipeline achieves a reduced overall latency of 49.63 milliseconds and a frame rate of 20 FPS, which is notably higher by 10 FPS compared to the sequential pipeline.

This comparative analysis between the sequential and multi-processor pipelines on the Nvidia Jetson AGX Orin highlights the importance of architectural considerations in software design for edge computing devices. By effectively leveraging the multi-core capabilities of the

Jetson Orin, the multi-processor pipeline substantially improves performance, demonstrating the potential for real-time processing and analysis in edge-based applications. The insights gained from such analyses are vital for developers and engineers looking to optimize computer vision systems, ensuring that they not only meet the required accuracy standards but also perform efficiently in real-world environments.

CHAPTER 5
Field Testing

The online testing platform employed for our research experiments incorporates advanced hardware components, specifically a high-performance edge computing unit, the NVIDIA Jetson AGX Orin, and a high-definition Logitech HD Pro Webcam C920. These elements are graphically represented in **Figure 5.1** of our documentation. The platform's primary hardware, the NVIDIA Jetson AGX Orin, offers robust computational power ideal for running complex machine learning models directly on the edge, which is essential for real-time processing requirements in field applications.

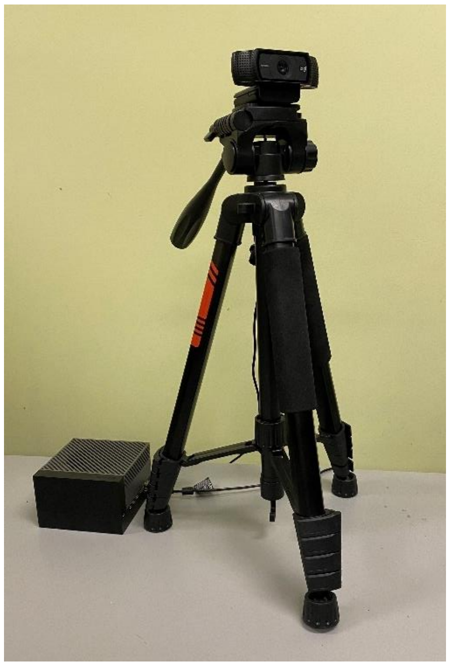
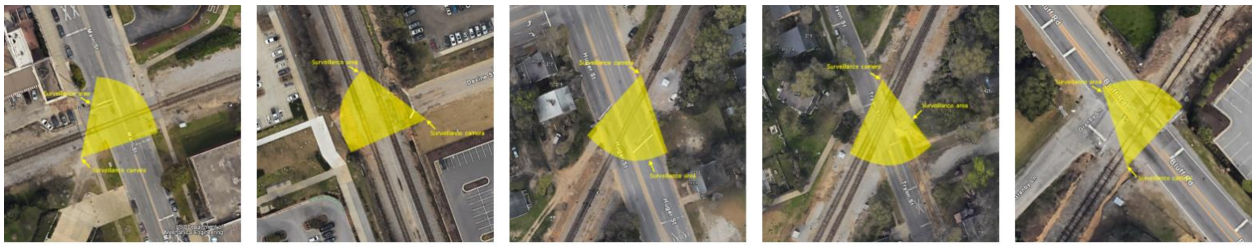


Figure 5.1 The hardware implementation



(a). 409 Main St, Columbia, SC 29201

(b). 718 Devine St, Columbia, SC 29201

(c). 230 Huger St, Columbia, SC 29201

(d). 216 Tryon St, Columbia, SC 29201

(e). 949 Rosewood Dr, Columbia, SC 29201

Figure 5.2 Field testing sites in Columbia, SC

As shown in **Figure 5.2**, field testing of our system was conducted at various locations in Columbia, SC. This diverse array of testing environments provided a broad spectrum of data, helping us assess the system's performance under different environmental conditions. The testing sites were chosen based on their unique characteristics, which included varying traffic patterns, pedestrian activity, and physical obstructions, all of which pose distinct challenges for automated detection systems.

As evidenced in **Figures 5.3-5.7**, the RCMS successfully identified an outlier in the rail crossing environment during one of the field tests. This detection is particularly noteworthy because it highlights the system's ability to discern and react to unexpected obstacles or anomalies effectively, thereby underscoring its practical effectiveness and reliability in real-world scenarios. The identified outlier, a temporarily placed construction sign not usually present at the site, was accurately detected and classified by our system, demonstrating its acute sensitivity to environmental changes.

The successful deployment and operation of our system demonstrate its potential in improving safety measures at rail crossings, a critical area of concern for urban transportation safety. This successful application not only validates the robustness of our proposed solution but also sets a precedent for future enhancements and implementations. The results we obtained are promising and pave the way for further research and refinement of the system to address broader safety challenges in rail transport and other related fields. The next steps include scaling the system for broader deployment across multiple cities and integrating additional sensors for enhanced multimodal data collection, which will allow for even more robust and fault-tolerant systems.

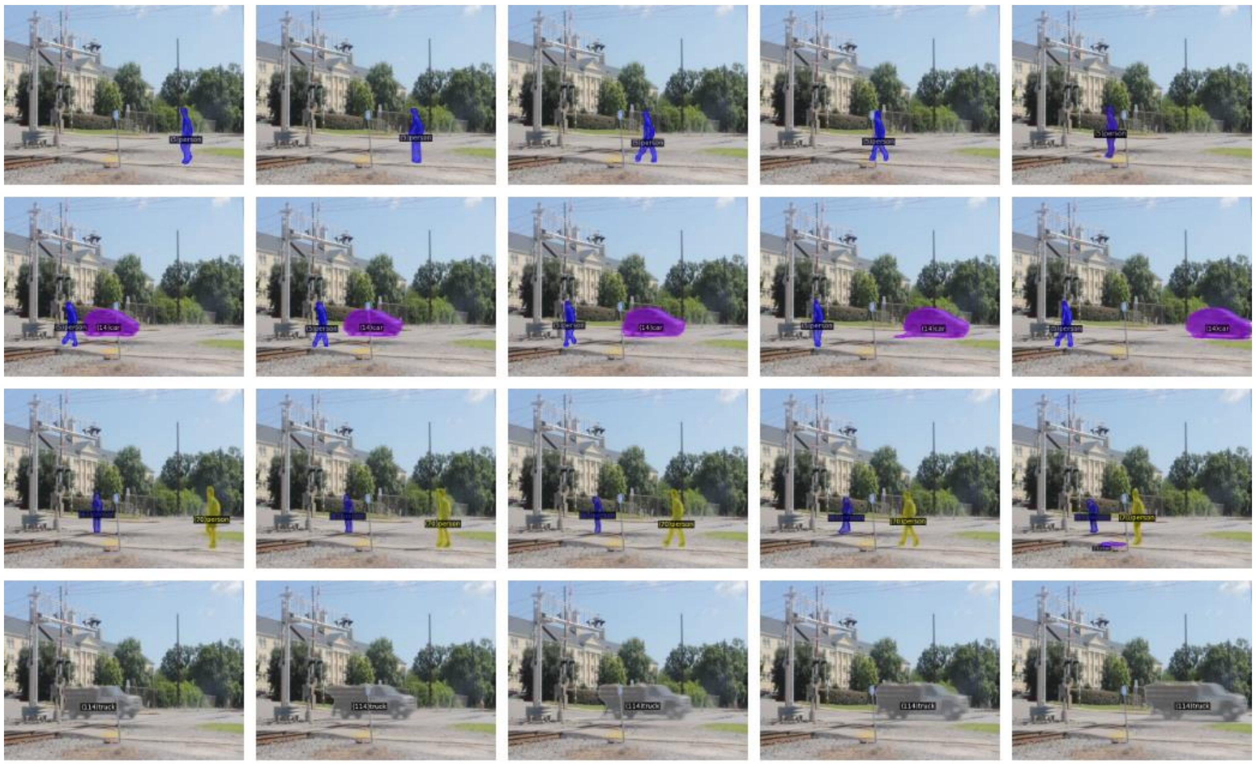


Figure 5.3 Field testing near 409 Main St, Columbia, SC 29201

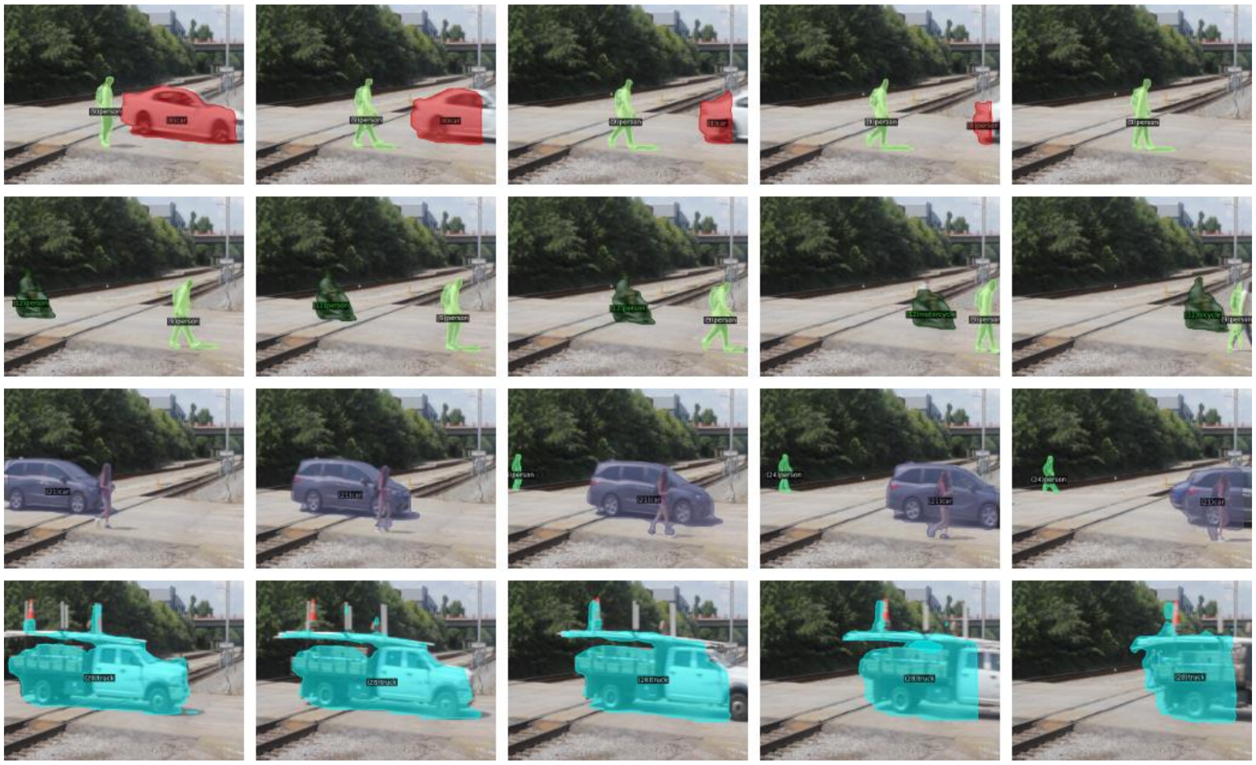


Figure 5.4 Field testing near 718 Devine St, Columbia, SC 29201

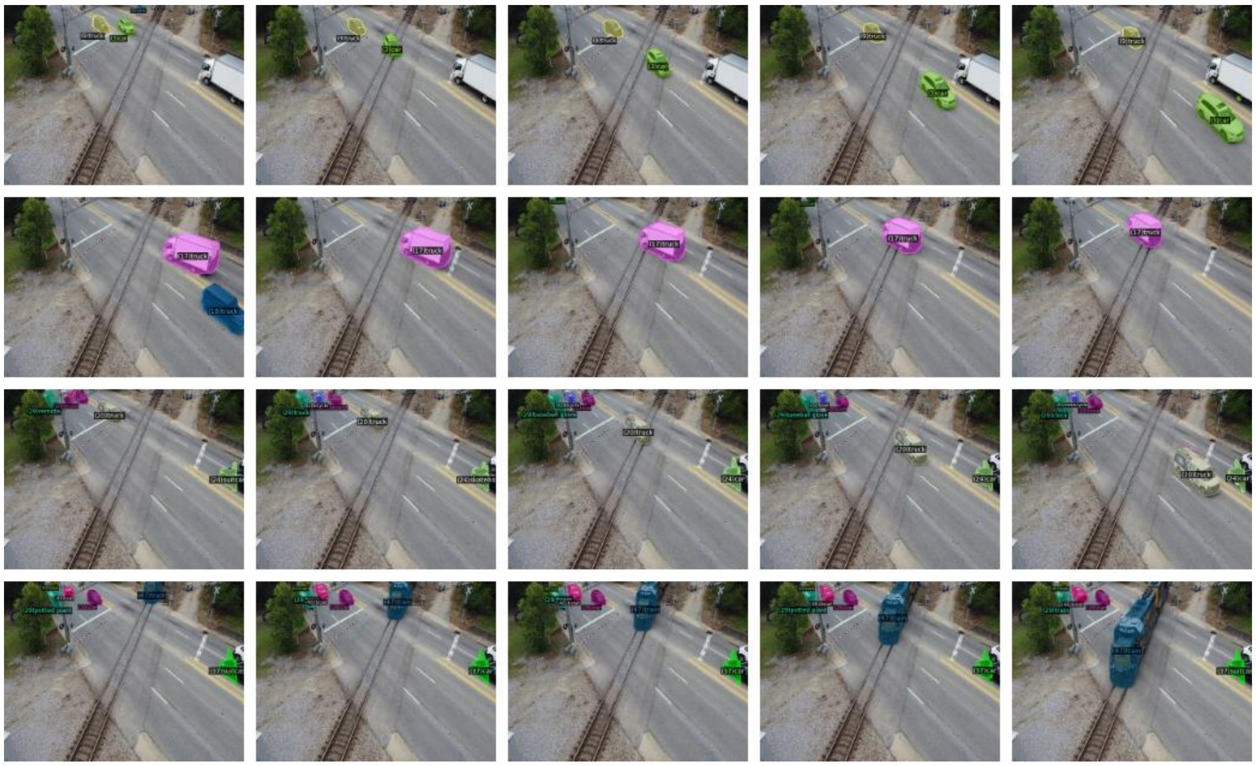


Figure 5.5 Field testing near 230 Huger St, Columbia, SC 29201

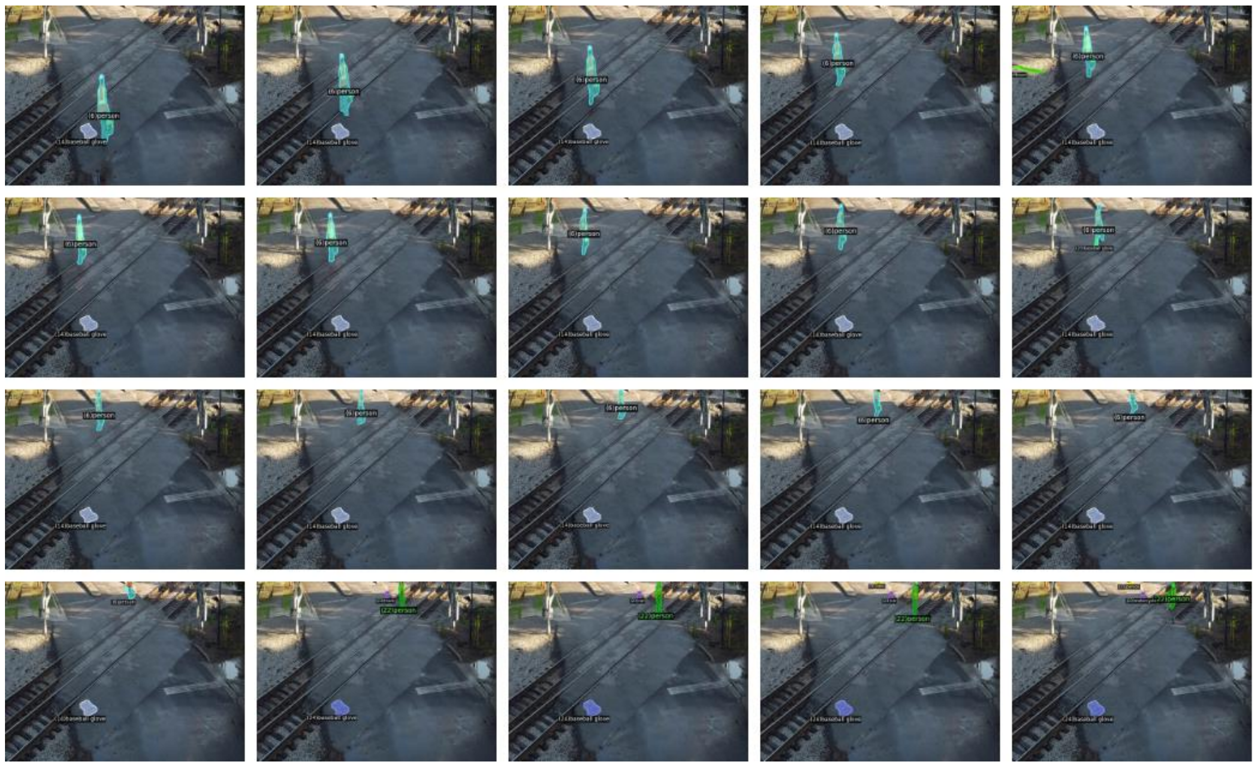


Figure 5.6 Field testing near 216 Tryon St, Columbia, SC 29201

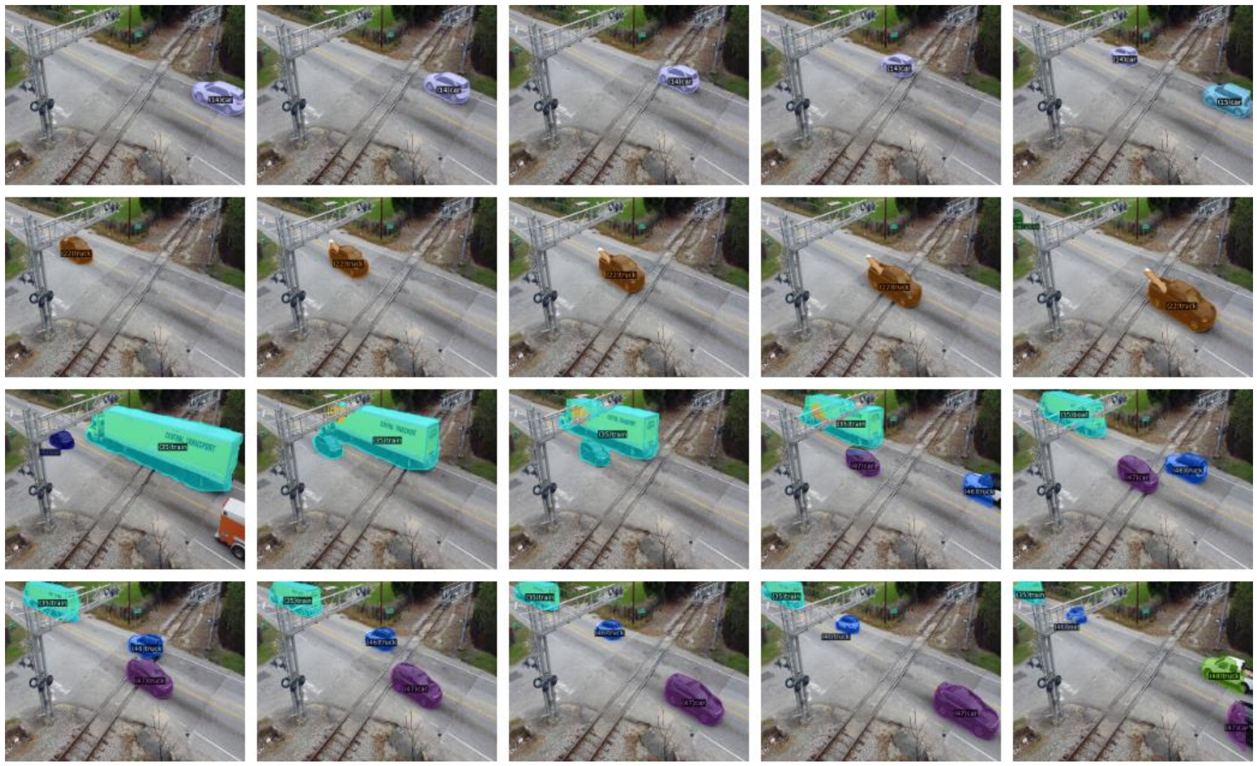


Figure 5.7 Field testing near 949 Rosewood Dr, Columbia, SC 29201

CHAPTER 6
Concluding Remarks

In this research, we introduced RCMS for railroad crossing monitoring, novel system for foreground detection, segmentation, classification and tracking, which is tailored for railroad crossing monitoring to improve railroad safety. The proposed method includes four parts: the SuBSENSE algorithm for background generation, then the YOLO-FG model inputs current frame and background images and outputs the foreground objects. As to know the type of the detected objects, the CLIP model is used to classify the detected boxes. Finally, Deep-SORT model tracks and associates the same objects across successive frames. This comprehensive system provides an effective solution for safeguarding the safety and security of railway grade crossings.

The experiments described utilize the CDnet 2014 dataset to assess the YOLOv8-FG model, specifically designed for foreground detection. The YOLOv8-FG model's performance is evaluated against several leading algorithms and is shown to excel by achieving the highest F-measure of 87.67%. This superior performance is highlighted through example detections in the results, where the model effectively identifies, classifies, and tracks various foreground objects with high accuracy.

This study optimizing RCMS systems on Nvidia Jetson AGX Orin for real-time applications. The analysis compares single-threaded and multi-processor pipelines. The single-threaded pipeline, using only one CPU core, achieves a latency of 120.82 milliseconds and 8.28 FPS. In contrast, the multi-processor approach, which leverages multiple cores for parallel processing, significantly reduces latency to 49.63 milliseconds and increases frame rate to 20 FPS. This demonstrates the effectiveness of utilizing the multi-core architecture to enhance performance in edge computing devices.

The field testing conducted at various railroad crossing areas highlights the robust performance of the proposed RCMS in different environments. This model is particularly adept at identifying any unauthorized or unexpected objects within the railroad crossing area. The effectiveness of RCMS in detecting anomalies and ensuring safety at railroad crossings is evident from its ability to accurately identify and segment objects that are not typically part of the railroad environment. This enhancement in detection capability is a significant step forward in the application of artificial intelligence in public safety and infrastructure monitoring.

Future work could focus on optimizing the RCMS model for lightweight deployment on lower-end edge devices by employing techniques such as model pruning to reduce redundant parameters, quantization to lower precision without sacrificing accuracy, and knowledge distillation to transfer the efficiency of smaller models. These approaches can significantly reduce the computational load and memory requirements, enabling the model to perform effectively on resource-constrained hardware. Additionally, exploring efficient neural network architectures like MobileNet or ShuffleNet and hybrid processing frameworks could further improve performance, ensuring real-time object detection and tracking with minimal latency even in challenging environments.

REFERENCES

FRA. (2019). Highway-Rail Grade Crossings Overview . Retrieved from https://railroads.dot.gov/program-areas/highway-rail-grade-crossing/highway-rail-grade-crossings-overview

Guan, L., Jia, L., Xie, Z., & Yin, C. (2022). A lightweight framework for obstacle detection in the railway image based on fast region proposal and improved yolo-tiny network. IEEE Transactions on Instrumentation and Measurement, 71, 1-16.

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 2961-2969).

Hou, B., Liu, Y., Ling, N., Liu, L., & Ren, Y. (2021). A fast lightweight 3D separable convolutional neural network with multi-input multi-output for moving object detection. IEEE Access, 9, 148433-148448.

Jocher, G., Chaurasia, A., Stoken, A., Borovec, J., Kwon, Y., Michael, K., ... & Jain, M. (2022). ultralytics/yolov5: v7. 0-yolov5 sota realtime instance segmentation. Zenodo.

Lifesaver, O. (2023, 6). Collisions & Casualties by Year. Retrieved from Operation Lifesaver: https://oli.org/track-statistics/collisions-casualties-year

Lim, L. A., & Keles, H. Y. (2020). Learning multi-scale features for foreground segmentation. Pattern Analysis and Applications, 23, 1369-1380.

Lin, C., Yan, B., & Tan, W. (2018). Foreground detection in surveillance video with fully convolutional semantic network. 2018 25th IEEE International Conference on Image Processing (ICIP) (pp. 4118-4122). IEEE.

Ogden, B. D., & Cooper, C. (2019). Highway-rail crossing handbook. United States. Federal Highway Administration.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., ... & Sutskever, I. (2021, July). Learning transferable visual models from natural language supervision. In International conference on machine learning (pp. 8748-8763). PMLR.

Rahmon, G., Bunyak, F., Seetharaman, G., & Palaniappan, K. (2021). Motion U-Net: multi-cue encoder-decoder network for motion segmentation. 2020 25th International Conference on Pattern Recognition (ICPR) (pp. 8125-8132). IEEE.

Sikora, P., Malina, L., Kiac, M., Martinasek, Z., Riha, K., Prinosil, J., . . . Srivastava, G. (2020). Artificial intelligence-based surveillance system for railway crossing traffic. IEEE Sensors Journal, 21, 15515-15526.

St-Charles, P.-L., Bilodeau, G.-A., & Bergevin, R. (2014). SuBSENSE: A universal change detection method with local adaptive sensitivity. IEEE Transactions on Image Processing, 24, 359--373.

Ultralytics. (2023). YOLO by Ultralytics,. Retrieved from https://github.com/ultralytics/ultralytics.

Valipour, S., Siam, M., Jagersand, M., & Ray, N. (2017, March). Recurrent fully convolutional networks for video segmentation. In 2017 IEEE Winter Conference on Applications of Computer Vision (WACV) (pp. 29-36). IEEE.

Varadarajan, S., Miller, P., & Zhou, H. (2015). Region-based mixture of gaussians modelling for foreground detection in dynamic scenes. Pattern Recognition, 48(11), 3488-3503.

Vijayan, M., Raguraman, P., & Mohan, R. (2021). A fully residual convolutional neural network for background subtraction. Pattern Recognition Letters, 146, 63-69.

Wang, Y., & Yu, P. (2021). A fast intrusion detection method for high-speed railway clearance based on low-cost embedded GPUs. Sensors, 21, 7279.

Wang, Y., Jodoin, P. M., Porikli, F., Konrad, J., Benezeth, Y., & Ishwar, P. (2014). CDnet 2014: An expanded change detection benchmark dataset. In Proceedings of the IEEE conference on computer vision and pattern recognition workshops (pp. 387-394).

Wojke, N., Bewley, A., & Paulus, D. (2017, September). Simple online and realtime tracking with a deep association metric. In 2017 IEEE international conference on image processing (ICIP) (pp. 3645-3649). IEEE.

Xu, C., Liu, H., Li, T., Zhang, Y., Li, T., & Li, G. (2022). Cascaded Feature-Mask Fusion for Foreground Segmentation. IEEE Open Journal of Intelligent Transportation Systems, 3, 340-350.

Yang, L., Li, J., Luo, Y., Zhao, Y., Cheng, H., & Li, J. (2017). Deep background modeling using fully convolutional network. IEEE Transactions on Intelligent Transportation Systems, 19, 254-262.

Zaman, A., Ren, B., & Liu, X. (2019). Artificial intelligence-aided automated detection of railroad trespassing. Transportation research record, 2673(7), 25-37.

Zhang, Z., Zaman, A., Xu, J., & Liu, X. (2022). Artificial intelligence-aided railroad trespassing detection and data analytics: Methodology and a case study. Accident Analysis & Prevention, 168, 106594.