INTEGRATING AI INTO CCTV SYSTEMS: A COMPREHENSIVE EVALUATION OF SMART VIDEO SURVEILLANCE IN COMMUNITY SPACE WITH ONLINE ANOMALY TRAINING

by

Shanle Yao

A thesis submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Master of Science in
Computer Engineering

Charlotte

2024

Approved by:

Dr. Hamed Tabkhivayghan

Dr. Asis Nasipuri

Dr. Ran Zhang

ABSTRACT

SHANLE YAO. Integrating AI into CCTV Systems: A Comprehensive Evaluation of Smart Video Surveillance in Community Space with Online Anomaly Training. (Under the direction of DR. HAMED TABKHIVAYGHAN)

This research introduces and evaluates an advanced AI-enabled Smart Video Surveillance (SVS) system designed to enhance safety across community spaces such as educational institutions, recreational areas, parking lots, small businesses, and in broader smart city applications. Our proposed system seamlessly integrates with existing Closed-Circuit Television (CCTV) systems and wired camera networks, making it easy to adopt and capitalizing on recent AI advancements. It uniquely employs metadata instead of pixel data for activity recognition to maintain privacy, adhering to stringent ethical standards. The SVS system features a cloud-based infrastructure and a mobile app for real-time, privacy-conscious alerts within communities.

In our comprehensive evaluation, conducted in a community college environment, we delve into AI-driven visual processing, statistical analysis, database management, cloud communications, and user notifications. The system was tested using sixteen CCTV cameras, achieving a consistent throughput of 16.5 frames per second over 21 hours with an average end-to-end latency of 26.76 seconds for detecting behavioral anomalies and alerting users. We also explore sophisticated data representation and visualization techniques such as Occupancy Indicators, Statistical Anomaly Detection, and Bird's Eye View. These tools help analyze pedestrian behaviors and enhance safety, offering intuitive visualizations and actionable insights for stakeholders like law enforcement, urban planners, and social scientists. The findings underscore the vital role of visualizing AI surveillance data in emergency management, public health, crowd control, resource distribution, predictive modeling, city planning, and informed decision-making.

This pioneering work is the first to examine the performance of a physical-cyber-

physical anomaly detection system, crucial for identifying potential safety incidents and guiding urban development.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AI Artificial Intelligence

API Application Programming Interface

AWS Amazon Web Services

CAGR Compound Annual Growth Rate

CCTV Closed-Circuit Television

CPU Central Processing Unit

CUDA Compute Unified Device Architecture

FPS Frames Per Second

GPU Graphics Processing Unit

IoT Internet of Things

IP Internet Protocol

IT Information Technology

NMS Non-Maximum Suppression

ONNX Open Neural Network Exchange

PCP Physical-Cyber-Physical

SOTA Simple Online and Realtime Tracking

SVS Smart Video Surveillance

CHAPTER 1: INTRODUCTION

Video surveillance systems have long been acknowledged as crucial for maintaining public safety and security [1]. Recent progress in machine learning algorithms, involving advancements in both hardware and software, has resulted in the widespread adoption of Smart Video Surveillance (SVS) systems in various sectors [2]. The video surveillance industry has grown substantially, attaining a value of USD 53.7 billion in 2023. Projections indicate a Compound Annual Growth Rate (CAGR) of 9.2%, compared to the average CAGR of 5% in the information technology (IT) sector, estimated at USD 83.3 billion by 2028[1].

A primary concern is the effective oversight and interpretation of these video outputs [3]. Legacy video surveillance frameworks, dependent on human operators for manual video review, are becoming less feasible for two main reasons. Firstly, the vastness of video content renders comprehensive and effective simultaneous monitoring a daunting task [4], especially when pinpointing specific incidents amid hours of footage. Secondly, many global law enforcement bodies face personnel deficits [5], resulting in an inadequate workforce to oversee the expanding camera networks. This scenario underscores the imperative to transition to intelligent surveillance systems capable of independent video analysis and delivering meaningful insights.

With the increasing significance of SVS systems for public safety, features like anomaly detection and real-time notifications are vital, as Raty and Komninos et al. [6, 7]. These systems are central to safeguarding communities by constantly monitoring and responding to anomalies. Anomaly detection in our system focuses on three key aspects: behavioral, object, and statistical anomalies. The former detects

[1] https://www.marketsandmarkets.com/

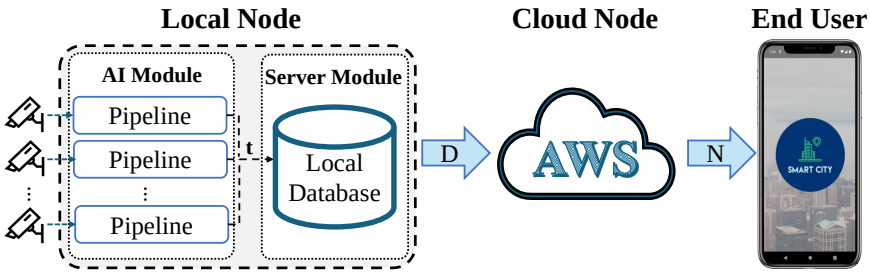

Figure 1.1: Conceptual Model.

unusual activities like trespassing or fighting, while the second identifies out-of-place items like unattended bags or unauthorized items and the last can detect unusual crowdedness grow with historical data. Swift recognition of these anomalies is crucial for preventing security breaches and facilitating rapid response from law enforcement. However, transitioning these systems from labs to real-world settings poses challenges. Latency can have unexpected consequences. Scalability is also necessary to cover diverse areas effectively. Privacy must also be addressed to protect individuals' rights while maintaining robust security measures [8, 9].

Research on AI-enabled computer vision and alert systems has mainly focused on lab performance, with limited real-world testing data [10, 11, 12, 13, 14]. Studies on alert systems prioritize latency and throughput over full performance analysis [15, 16, 17, 18], and many overlook real-world conditions like scalability in testbed environments or the practical transition of these systems for public safety [19, 20, 21, 22]. Overall, existing research work only stays at evaluation in a highly controlled lab environment and often reports their results based on standard datasets while running one or two algorithms in isolation. They also lack full end-to-end evaluation. This indicates a need for empirical studies on their real-world effectiveness and challenges.

This research introduces and evaluates an AI-based real-time Smart Video Surveillance (SVS) system in a practical, real-world setting to overcome prevalent challenges.

Deployed in a community college, the system adeptly handles real-world constraints, including computational limitations and the integration of legacy infrastructure such as Internet Protocol (IP) and Closed-Circuit Television (CCTV) cameras that utilize LAN connectivity. As illustrated in Figure 1.1, the study adopts a conceptual end-to-end model. Drawing inspiration from previous research by Pazho et al. [14] and [23], this work achieves comprehensive end-to-end integration, embracing community-in-the-loop principles. It synergizes real-time video analytics and edge computing within community spaces, facilitating immediate notification to communities and stakeholders. The system comprises four key components: (1) an AI pipeline for real-time, privacy-preserving video analytics; (2) a local node that aggregates processed information from multiple AI pipelines across various cameras; (3) a cloud node for external communication beyond the monitored physical environment; and (4) end-user applications designed to keep communities and stakeholders informed and engaged.

Our approach also employs data analysis techniques to generate statistical outputs that assist human observers in comprehending historical data and trends. Such approach could help the application users has better understanding of the environment rather than just anomaly notifications. These methodology could also help the end users to make their own decisions even more. First, we used statistical analytics to provide stats such as headcounts, the average number of people detected hourly, and the total number of people over time to provide general overview of the raw AI data. Furthermore, we employ visualization techniques on top of these statistical outputs to enhance situational awareness. For instance, by extracting and analyzing data such as bounding boxes and human trajectories, we applied a transformation matrix considering the camera angle to get the bird's eye view at specific times. We also utilized the historical data of each camera to generate occupancy indicators that represent a meaningful visualization of current people at each location regarding the historical data. Also, we present the statistical anomalies by extracting the normal

distribution of the data trend per location.

This research evaluates the Physical-Cyber-Physical system's performance under varying real-world conditions, such as different crowd densities and extended operational durations. We monitored up to 16 existing CCTV cameras. Our results, obtained on a server with a 16-core CPU and four 10496 CUDA-cores GPUs, demonstrate the system's robustness and efficiency. Specifically, the system maintains a throughput of 16.5 FPS and a latency of 6.5 seconds during a 21-hour continuous operation. Our real-world experiments also show that the system can detect suspicious objects within an average of 4.7 to 19.68 seconds and notify end-users of anomalous behavior within 9.34 to 26.76 seconds, depending on the camera setup. These results validate the system's effectiveness, scalability, and reliability, emphasizing the role of hardware in achieving these outcomes.

In summary, the contributions of this research are:

- To the best of our knowledge, this research is the first to propose, deploy, and measure an end-to-end AI-based real-time video surveillance that can run over existing legacy infrastructure for real-time situational awareness such as peak hour analysis at the desired location, rapid decision-making in case of emergency, and enhancing safety and operational efficiency in various high-stakes environments.

- We delve into the valuable insights extracted from data visualization, emphasizing their potential role in augmenting safety measures.

- This research performs the extensive evaluation and real-world experimental study over existing CCTV cameras, focusing on AI Pipeline latency, throughput, scalability, and endurance of the proposed system over the increasing number of cameras with vulnerable environmental conditions and density of human subjects in the scene and set the first benchmark for future studies.

- To the best of our knowledge, this research is the first to evaluate user-in-the-loop anomaly detection aspects of video surveillance (Physical-Cyber-Physical aspects), representing the total end-to-end time from when an anomaly appears in front of the camera to the instant the end user receives a notification which is crucial in applications such as emergency responses, security monitoring in sensitive locations, and monitoring critical infrastructure.

The rest of this paper is organized as follows. Section 2 begins by reviewing the relevant works on this topic. Section 3 describes the system features. Sections 4 and 6 presents our experiments' test setup and results. Section 6.4 focuses on the latency measurement of anomaly detection in the real world, and Section 7 concludes this research.

CHAPTER 2: LITERATURE REVIEW

Machine learning and artificial intelligence advancement in various domains have paved the way for complex data analysis and visualization. The criticality of analyzing the outputs of these machine learning models is evident in numerous applications, ranging from urban planning to safety considerations. However, the effectiveness of these models heavily relies on the presentation and interpretation of the analyzed data.

The advent of machine learning techniques, especially when combined with IoT devices, has revolutionized urban planning and smart cities. Mahdavinejad et al. underline the importance of such methods in forecasting urban congestion and paving the way for solutions [24]. Complementing this, Zanella et al. argue for the accessibility of urban IoT data to authorities and citizens, fostering civic participation and swift responsiveness to urban challenges [25]. Taking this a step further, Rathore et al. delve deep into the capacities of IoT systems for urban development, focusing on vehicular traffic datasets [26]. However, the true potential of these systems is realized when combined with big data analytics, as highlighted by Al Nuaimi et al., who stress iterative improvements based on analytical feedback [27].

Transitioning from holistic urban planning, it's imperative to focus on specific aspects, like traffic and pedestrian safety, which are paramount for the efficient functioning of a city.

Safety remains at the forefront of urban considerations, and several studies have dedicated efforts to analyze traffic patterns and potential hazards. Mouchili et al. venture into vehicle traffic data, shedding light on anomalies and parking occupancy [28]. Abberley et al. expand this scope by using machine learning to study traffic accidents,

revealing patterns similar to human traffic analytics [29]. Similarly, Bharadwaj et al. apply image processing techniques on highway images, concentrating on vehicle placements with a unique approach to data representation [30]. The realm of pedestrian safety isn't untouched either, with Chen et al. employing machine learning on Street View images to detect pedestrians, albeit without delving into intricate visualizations [31]. However, safety isn't confined to traffic. He et al. highlight the use of machine learning in detecting crime-facilitating factors in neighborhoods, emphasizing the dire need for robust visual representation for impactful insights [32].

SVS research is generally divided into two main areas: real-time object tracking and anomaly detection in both vehicular and human contexts [33][34]. Some studies have focused on using lightweight models for edge devices to minimize latency as demonstrated by Pramanik et al. [35]. In 2023, an innovative SVS system was introduced that employs advanced pre-processing techniques specifically for fall detection as introduced by Singh et al. [36].

The preceding year witnessed the emergence of the E2E-VSDL as introduced by Gandapur [37] method, employing a combination of BiGRU and CNN to compare video frames with pre-identified anomaly videos within their trained model, resulting in approximately 98 percent accuracy with the available dataset. Another notable contribution is the TAC-Net methodology [38], which introduces a novel deep contrastive learning approach for SVS. TAC-Net employs a self-supervised learning model to capture high-level semantic features, leading to superior performance in addressing anomaly scenarios compared to existing state-of-the-art methods on popular datasets in 2021.

A range of Smart Cities studies aims to bridge the gap between research and real-world applications. [39] A 2019 paper as introduced by Alshammari et al. [40] presents a functional SVS system implemented with surveillance cameras, moving beyond simulation-based evaluations. To enhance anomaly detection models and achieve

better training outcomes, researchers from RV College of Engineering Bengaluru [41] focus on refining object detector model localization, improving validation accuracy by around 98.5 percent. The paper published by Y. Yuan et al. [42] brings a very advanced methodology about using decentralized framework into Cyber-Physical system to optimize the traffic and pedestrian waiting time, while the response time of the system is discussed in a real world scenario. Publication [43] with a very impressive self-awareness Cyber-Phycical system also lacks of data with system latency which we values this a lot in our testbed community.

Optimizing large-scale SVS systems has become a focus of recent research, with various systems [44] concentrating on optimizing data transformation, communication, and even incorporating blockchain technology in diverse domains. These efforts go beyond model performance to consider the broader scalability of the entire system.

Efficient detection of suspicious objects is crucial for real-life security applications. However, widely used object classification datasets like COCO often lack harmful object categories. A study on Real-Time Abnormal Object Detection introduced by Ingle et al [45] demonstrates the training and implementation of an abnormal object detection model in smart cities, achieving an accuracy of approximately 90 percent. Additionally, a 2023 publication focuses on person re-identification using deep learning-assisted techniques [46], highlighting the significance of spatial and channel attention mechanisms in achieving improved re-identification scores.

While significant progress has been made in SVS research, a notable gap remains regarding comprehensive testbed support to evaluate real-world SVS system performance. Motivated by this gap, our study aims to demonstrate the capabilities of a state-of-the-art SVS system within a public setting, specifically a community college. In the subsequent sections, we present our achievements and offer insights into the potential of our endeavors.

CHAPTER 3: System Feature

Our proposed system represents a pioneering advancement in the realm of AI-based real-time video surveillance, uniquely designed to integrate seamlessly with existing legacy CCTV camera infrastructures, and it is a Physical-Cyber-Physical (PCP) system to alert the end users about anomalies. The proposed system comprises four distinct components: (1) AI Module/Modules, (2) Server Module, (3) Cloud Node, and (4) End user devices. Figure 3.1 provides an architectural overview of the end-to-end system. One five-stage computer vision pipeline is assigned to each on-site camera within the testbed. The output generated from these pipelines is subsequently channeled to a global tracker and a database for human re-identification and statistical analyses. The cloud node facilitates the transmission of the processed data to the end users, enabling the visualization of the data and notifications. In the following, we describe each component in detail.

3.1 AI Module/Modules

As Figure 3.1 shows, the AI Module is architected as a modular, multi-stage pipeline tailored for computer vision tasks. This modularity offers the flexibility to adapt the system for optimal performance in real-world security applications for real-world response instead of running it sequentially.

Inspired by [14], AI Module runs the deep learning pipeline within the physical environment next to the cameras. The images are then organized into batches of 30 frames, aligning with the window size prerequisites for high-level tasks. The batched image data is first processed by an object detector [47][48], identifying and localizing objects like people and high-priority items like guns and knives. Object Anomalies are

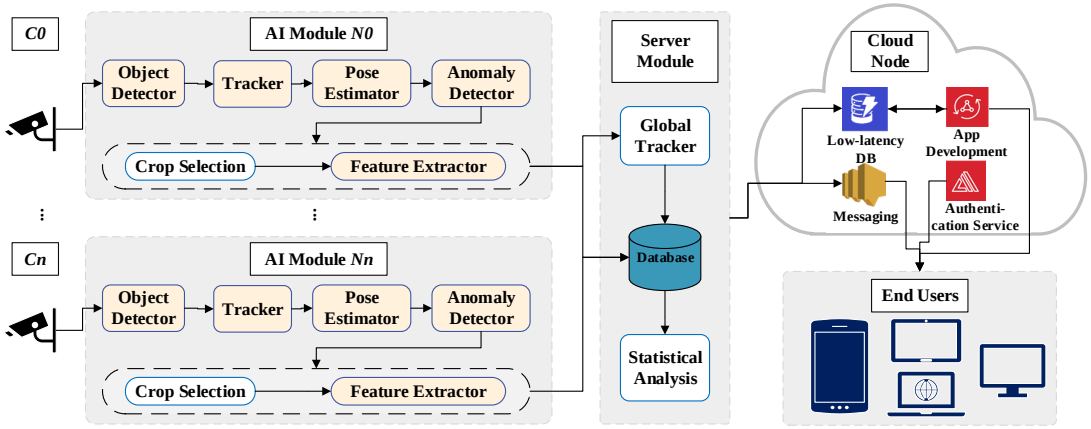

Figure 3.1: End-to-end detailed system. C_0 represents the camera *ID* and for each camera, one AI Module N_0 including multi-AI-vision-models pipeline is assigned with. All the AI modules send processed data to one Server Module database, and Server Module will re-identify the human track ID based on the feature extractor data. The statistical analyzer analyzes all the data in the database stored across all the cameras and communicates the results with Cloud Node. Cloud-native services are utilized to host the end users' applications.

flagged at this stage. The bounding box of detected persons is then sent to a tracking algorithm [49], where tracklets with local IDs are created. These IDs and bounding boxes are used in the subsequent pose estimation stage, where HRNet [50] extracts 2D skeletal data for each individual.

3.1.1 Object Anomaly Detection

For object anomaly detection, the object detection methodology employed leverages the YOLO algorithm [47][48]. This approach facilitates the identification of all objects of interest. Models can be tailored using custom datasets, encompassing both conventional object classes and anomalous ones. Upon detecting anomalous objects via YOLO, alerts are transmitted to the Cloud Node through a rule-based messaging service, triggering notifications on end users' devices. The objective is to optimize object anomaly detection for efficiency, aiming to minimize response latency.

3.1.2 Behavioral Anomaly Detection

Our system's AI pipeline uses the Graph Embedded Pose Clustering for Anomaly Detection (GEPC) [51] to analyze 2D pose-estimation data in 30-frame windows with a 20-frame stride. It scores individual movements in batches, contributing to a frame anomaly score. Each frame gets a scene anomaly score if it contains person detections. The pipeline then selects the most informative frame from each batch to enhance database efficiency. Next, a feature extraction algorithm [52] is used for human re-identification across different camera feeds. This process ensures that processed data, without pixel data, is effectively stored on the server for extended analysis and cross-camera identification.

3.2 Server Module

The server module serves as a repository for both metadata and historical data, which are transmitted from each of the AI modules. This data is systematically stored within a database, which is structured utilizing MySQL. Table 3.1 exemplifies the data storage schema within this database. It comprises several columns, each providing unique insights.

The database logs entries with precise timestamps in the "Record_Time" column, identifies the camera source in the "Camera_ID" column, and classifies detected objects using the "Class_ID" column, based on the COCO dataset, which "0" is human. The "Bounding_Box" column details the position and dimensions of bounding boxes around detected objects with the X and Y coordinates of the top-left corner of the bounding box and its width and height. In contrast, the "Feature" column stores human feature data as Tensors from the OSNet for re-identification. The "Local_ID" column assigns unique identifiers for object tracking within a single camera view, whereas the "Global_ID" column provides universal identifiers for cross-camera tracking. Lastly, the "Anomaly_Score" column rates the likelihood of an

Table 3.1: Example visualization of the database at Server Node

Record Time	Camera	Class ID	Bounding Box	Feature	Local ID	Global ID	Anomaly Score
00:00:00	1	0	[x, y, w, h]	Tensors	15	1001	40
00:00:01	2	0	[x, y, w, h]	Tensors	21	1001	40
....	...	..	...	...	..	..	..
23:59:59	1	0	[x, y, w, h]	Tensors	9999	1001	40

anomaly on a scale from negative infinity to 40, with 40 indicating no anomalies.

The server module primarily handles global tracking and statistical analysis. Global tracking uses the cosine similarity method to compare the feature data of individuals from all cameras. This approach allows the system to re-identify people across different cameras and assign a unique global identifier to each pedestrian within a specific time interval.

As shown in Figure 3.1, we perform a statistical analysis on the dataset housed within the Server Module. This analysis leverages the data depicted in Table 3.1 and encompasses a variety of data analytics techniques. This step holds pivotal significance within the overarching system for two primary reasons. The system's design significantly enhances privacy and usability for real-world applications in two ways. First, not transmitting raw data to the Cloud Node reduces the reversibility of pipeline output, thereby bolstering privacy. Second, the analysis provides end users with valuable insights beyond raw data. This includes real-time metrics such as location-specific headcounts and occupancy patterns and longitudinal analyses of population trends and resource consumption, all illustrated through occupancy indicators and heatmaps. These features collectively improve the system's utility for end users.

3.3 Cloud Node

As shown in Figure 3.1, the system leverages a suite of cloud-native services, including robust data storage, management solutions, and API generators, to meet its objectives [53] to achieve its goals. The system architecture aims to minimize the

time lag between detecting an anomaly and the notification received by the end-user. We utilize a rule-based messaging service to send real-time push notifications. The service generates specific topics and messages based on the type of detected anomalies. These notifications are disseminated using the JSON protocol and can be received via email, text, or app notifications. We primarily use email notifications to mitigate potential latency issues due to service outages.

We used a low-latency database service as our cloud-based data storage solution, enabling real-time data access. The database is organized into two types of tables: one for tracking objects across each camera and another for storing analytical results over time. These tables are indexed by timestamps and camera IDs, allowing for easy data retrieval.

For the smartphone application, we use an application development kit to generate the necessary APIs. This kit employs a GraphQL schema to pull data from existing tables, optimizing data retrieval and minimizing over-fetching, shown as Figure 3.1 [54]. User management and authentication are other aspects of the mobile application that cloud services handle [55]. The specifics of our implementation are beyond the scope of this paper. In summary, our developed end-to-end system architecture offers an efficient and scalable platform for the smartphone application, which is responsible for delivering notifications to the end user.

3.4 End User Devices

The system's primary goal is to secure and timely notify end users in case of detected anomalies. To achieve this, we use a smartphone application that provides users with a wealth of real-time data [8]. This includes the statistical analysis in section 3.2 and detailed information on detected anomalies. Our system is engineered to notify users promptly if any anomalies are identified. Users can choose their preferred notification methods.

The smartphone application has been designed to cater to a broad audience, offering

seamless functionality across different operation systems and platforms. This cross-platform consistency ensures that users can access the same features and benefits regardless of their chosen device or operating system. This approach not only enhances accessibility but also provides a unified user experience, optimizing the dissemination of crucial information to the end users.

CHAPTER 4: System Setup and Configuration

In our testbed, we deployed a network of 16 AXIS IP cameras, each operating at a frame rate of 30 FPS with 720p resolution, distributed across the college campus. Figure 4.1 visually represents the camera placements. Three of the 16 cameras were installed outdoors to monitor the parking areas. The remaining 13 were placed indoors across three separate buildings. Specifically, three cameras were allocated to oversee entry and exit points, another trio focused on vending machine areas, four captured hallway activities, and the last three monitored communal study spaces. The indoor cameras are located at the height of seven feet and six inches while the outdoor cameras are located at the height of ten feet and eight inches. Each camera's lens had a varifocal from 3.4 to 8.9 mm with a horizontal field of view of 100 to 36 degrees and a vertical field of view of 53 to 20 degrees. Around 35000 square feet of area is covered by thirteen indoor cameras and around 60000 square feet by three outdoor cameras.

The system runs on a dedicated server (physically located within the community college) with a single 16-core CPU with 1500 MHz clock speed and 252 GB memory and four 10496 CUDA-cores GPUs, each with 1700 GHz boost clock and 24 GB VRAM. We established multiple endpoints to facilitate comprehensive multi-user testing. An individual topic was created on the rule-based messaging console. Initially, four endpoints were used in controlled tests to measure end-to-end latency. This was later expanded to 50 endpoints for broader, open-world testing. We conducted experiments under three distinct scenarios, examining "AI Pipeline," "Load Stress," and "Endurance" to assess the system's real-world capabilities, performance metrics, and latency in real-time.

Figure 4.1: Locations of Cameras on the Campus. A maximum of 16 cameras are used; 3 of them are outdoors, and the other 13 are indoors

4.1 AI Pipeline Setup

The primary aim of the first scenario is to pinpoint the most effective configuration for the AI Module [14], tailored to meet real-world security demands as outlined in section 3. The configuration encompasses four core tasks: object detection, pedestrian tracking, human pose estimation, and human re-identification. Additionally, a high-level task-behavioral anomaly detection-integrates the results of these core tasks. Each core task offers a range of alternative methods, each with its trade-offs between accuracy and computational speed. While some SotA methods offer impressive accuracy, they often fall short in real-world SVS applications due to their high computational demands and complex parameter settings.

As demonstrated by Pazho et al. [14], the GEPC method [51] is particularly well-suited for behavioral anomaly detection. ByteTrack [49] and OSNet [52] excel in pedestrian tracking and human re-identification, respectively. The recently introduced YOLOv8 [48] stands out for its object detection capabilities and enhances human pose estimation through a top-down approach.

In this evaluation, our primary objective is to rigorously evaluate the performance of various algorithmic approaches to identify the most optimal one for integration into our system for real-world scenarios. We conducted a comprehensive performance evaluation, comparing the features of our existing system [14] against other popular methods recently developed in the field. The metrics for this comparative analysis included throughput, latency, and the number of detected outputs with their computation loads on GPU and CPU. We utilized video clips from the DukeMTMC dataset [56] for this evaluation, each lasting one minute and featuring different crowd densities. To ensure the scientific rigor of our experiments, these tests were initially performed on a separate laboratory server. Only after confirming the superiority of the chosen algorithm did we proceed with its full-scale integration into our designated testbed.

4.2 Load Stress Setup

We comprehensively examine the system's performance under escalating input loads in the second scenario. The focus is quantifying key metrics such as average latency and throughput across videos with varying crowd densities. We progressively increased crowd density. This rigorous evaluation provides valuable insights into the system's resilience and adaptability under increased workloads, a crucial factor for its real-world applicability.

To robustly assess the system's scalability, we conduct evaluations using different numbers of pipeline modules. This methodology allows us to gauge how the system adapts to varying levels of computational complexity. During the experimental phase, we designed concurrent tests that ran the system with one, four, eight, and twelve modules, all processing identical crowd density.

4.3 Endurance Setup

The final scenario focuses on deploying the system in a genuine real-world setting, utilizing all 16 cameras in our testbed. The primary aim is to assess the system's

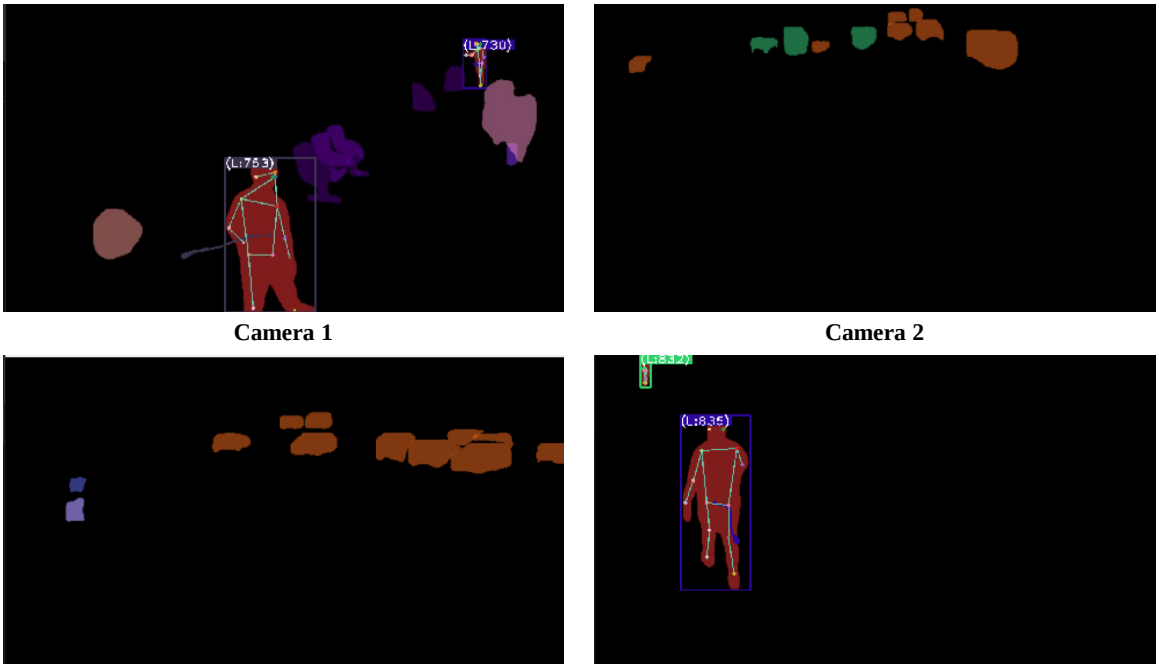

Figure 4.2: This represents an example of 4 pipelines running in the testbed with IP cameras deployed at different locations.

long-term stability and endurance. To this end, we conducted extended trials with configurations of eight, twelve, and sixteen modules, each running continuously for 21 hours. To ensure the video length does not significantly affect the results, we ran the system with 8 modules for a week period.

- Eight-Module Configuration: Covers one entrance point, one vending machine location, two hallway segments, and two common areas.
- Twelve-Module Configuration: Adds three additional outdoor cameras monitoring the parking lot and one more indoor camera at an entrance point.
- Sixteen-Module Configuration: Utilizes all available cameras for data acquisition.

As depicted in Figure 4.2, a sample output from the Local Node is displayed during a scenario with four operational cameras. For privacy reasons, instance segmentation masks have been applied to the visualizations. Specifically, Cameras 1 and 4 monitor

hallway points, while Cameras 2 and 3 oversee parking lot areas. This rigorous test aimed to validate the system's robustness and ability to maintain optimal performance under realistic, extended operational conditions.

CHAPTER 5: Data Analysis and Representation

In this section, we explore the detailed data stored in our database. The aim is to transform raw, unprocessed data into insightful and actionable information. Our exposition begins with presenting descriptive data, offering the end user a foundational and general grasp of the dataset's landscape. This preliminary understanding is then augmented with more sophisticated analyses designed to delve deeper into the data's nuances. Through these advanced analytical techniques, we endeavor to magnify the end user's situational awareness, equipping them with a more comprehensive and enriched perspective of the environment.

We gathered the data for data representation with 8 cameras from the testbed, 2 outdoor parking lot and six indoor ones, over 8 days, commencing on October 12^{th}, 2023, and concluding on October 20^{th}, 2023. Our deliberate selection of this timeframe encompassed 4 holidays, including 2 fall break days, 2 weekend days, and 4 working days.

5.1 Descriptive Data

In our descriptive data analysis, the linchpin is the Global ID, derived from the human feature data as outlined in Table 3.1. This Global ID is a unique identifier, ensuring consistent tracking across various data streams. Five foundational metrics have been presented in this analysis, designed to provide end users with a better understanding of the system's capabilities of the data. Although these data do not provide insightful information to the end user, they can provide a general overview of the traffic flow in the environment over time.

Current Number of People at Real-Time: AI pipelines allow the real-time

assessment of the number of individuals detected across all cameras. We can determine the current number of people under surveillance by querying the most recent timestamp entries from our database and aggregating the Global IDs.

Hourly Average Number of People Per Camera: By grouping the database entries by hour for each camera and averaging the unique Global IDs detected, we can compute the average number of individuals per camera identified hourly. This statistic gives the end user a general understanding of hourly people distribution.

Hourly Average Number of People Per Location: Beyond individual cameras, understanding the overall occupation trends in the environment is crucial. By categorizing entries based on the group of cameras and then by hour and subsequently counting and averaging unique Global IDs, we can determine the average traffic across all cameras. This data is incredibly valuable as it offers insights into the anticipated average number of individuals in the environment, leveraging historical data. Such information can prove instrumental for conducting evacuation drills and planning emergency procedures.

Total Number of People Over Time: The cumulative flow of people over a specified duration provides a broader understanding of traffic flow. By organizing database entries chronologically and counting unique Global IDs, we can chart a time series that depicts the total number of detected individuals, which is a very important metric for controlling the traffic of individuals.

Peak Hour Analysis: Identifying periods of maximum foot traffic is crucial for various operational decisions. We can pinpoint the peak traffic hours by aggregating data hourly and ranking these hours based on the count of unique Global IDs and use this data to optimize security personnel assignments.

5.2 Situational Awareness

Situational awareness is a key concept in various domains, encapsulating the ability to identify, process, and comprehend critical elements of information about the

environment. Situational awareness is about clearly understanding one's surroundings, which is essential for decision-making and proactive responses. Its value enables entities, individuals, or systems to anticipate needs and potential challenges, facilitating timely and informed actions. In the context of surveillance and safety, situational awareness is indispensable. This section delves into four key visualization techniques that support situational awareness: the Occupancy indicator, Statistical anomaly, Bird's eye view, and Heat map. We explain each technique's significance and the underlying methodology employed for its computation. Further, we underscore these visualizations' insights, accentuated with practical examples. When employed judiciously, these techniques serve as powerful tools, shedding light on patterns and anomalies elevating our understanding and proficiency in environment monitoring.

5.2.1 Occupancy Indicator

The Occupancy Indicator is a vital tool in video surveillance, aiming to provide contextual understanding concerning the number of people present in a specified location captured by a particular camera. It interprets raw data, such as a head count of seven individuals, into meaningful information by illustrating the relative occupancy level concerning the historical data, whether crowded or within the normal interval. For instance, a count of seven in a small room might signify a high occupancy, whereas in a larger hall, it might denote a low occupancy level. In emergencies like evacuations, knowledge of occupancy levels optimizes response effectiveness. Amid health crises, such as the COVID-19 pandemic, controlling occupancy is essential for public health. By providing a visual or numerical indicator, audiences can grasp the spatial dynamics at a glance.

Algorithm 1 demonstrates an algorithm to calculate and categorize the occupancy level at different camera locations using the data frame (`df`), which is continuously updated with new data. The algorithm iterates through each camera ID within the data frame and, for every camera, executes a loop every 5 seconds within a

Algorithm 1 Occupancy Indicator Algorithm

```
Input: data frame df, historical_data
for each camera_id in df do
    for every 5 seconds in record_time do
        current_number_of_people = length(unique(global_IDs))
    end for
    if current_number_of_people ≤ percentile(historical_data, 25) then
        occupancy = "Low Occupancy"
    else if current_number_of_people ≤ percentile(historical_data, 75) then
        occupancy = "Normal Occupancy"
    else
        occupancy = "High Occupancy"
    end if
    update_historical_data(historical_data, current_number_of_people)
end for
```

specified range of (`record_time`). Each iteration of this nested loop computes the current number of people by determining the length of the set of unique global IDs (`global_IDs`) present in the data frame during that time interval.

The occupancy level is then evaluated by comparing the `current_number_of_people` against the percentiles (25^{th} and 75^{th}) from the historical data (`historical_data`). Specifically, if the `current_number_of_people` is less than or equal to the 25^{th} percentile of the historical data, the occupancy is categorized as "Low Occupancy." If it falls between the 25^{th} and 75^{th} percentiles, it is categorized as "Normal Occupancy". Otherwise, if it is greater than the 75^{th} percentile, the occupancy is deemed as "High Occupancy".

Once the occupancy level is redetermined, the algorithm calls a function update_historical_data to update the historical data with the current_number_of_people. This updating step ensures the historical data remains current, allowing for more accurate and relevant occupancy level determinations in subsequent iterations. This process is carried out for every camera ID in the data frame, ensuring a comprehensive evaluation of occupancy levels across all monitored locations.

In Figure 5.1, we can observe a comparison of occupancy indicators for the same

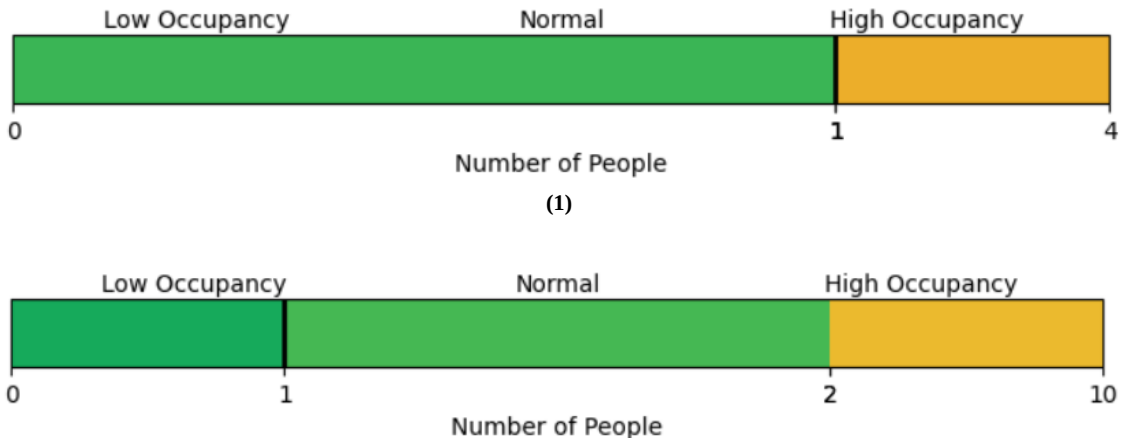

Figure 5.1: Comparing the Occupancy indicator of the same camera in an hour in two days. The graph (1) shows the Weekend. The graph (2) represents weekday.

camera during the same hour on two distinct days. Graph 1 illustrates the occupancy indicator for a weekend, whereas Graph 2 depicts the occupancy indicator for a weekday. The figure highlights that, during weekends, the detection of 2 people is categorized as "High Occupancy," whereas on weekdays, this number is considered "Normal."

5.2.2 Statistical Anomaly

In surveillance and safety, understanding statistical anomalies plays a pivotal role. This is primarily because anomalies, or deviations from the norm, often signal unexpected or unusual events. For instance, a sudden surge in crowd density in monitoring public spaces can indicate potential risks such as unauthorized gatherings, evacuations, or charges. By analyzing historical data, a baseline or 'norm' for crowd density can be established. Any significant deviation from this baseline, especially an increase in crowd density, would be considered an anomaly.

Being able to detect such anomalies in real time enables quick response mechanisms. For safety personnel, it provides an opportunity to proactively address potential

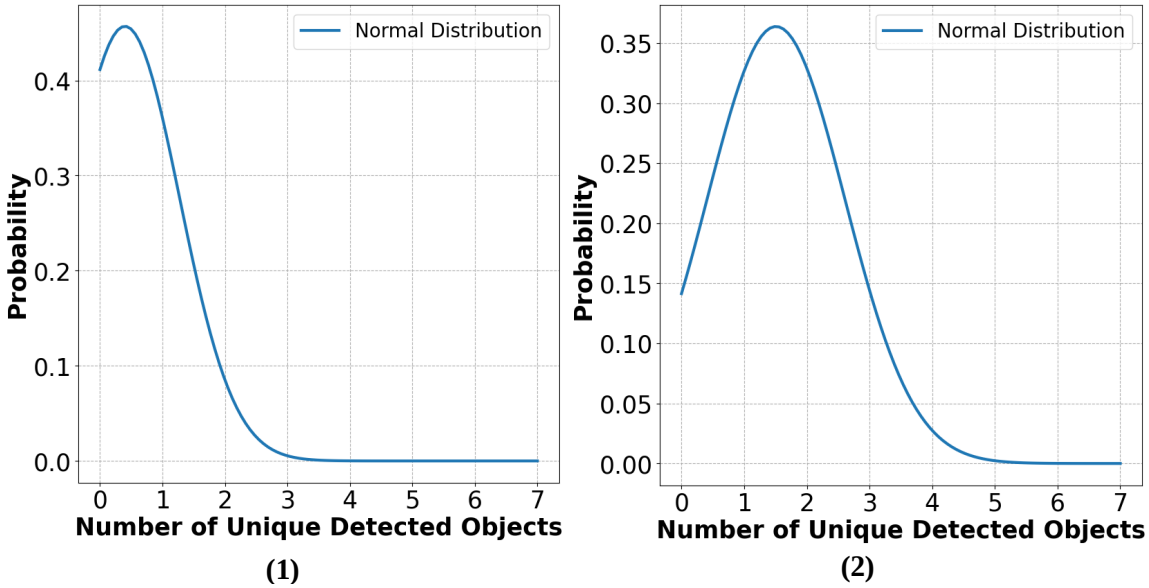

Figure 5.2: Comparing the normal distributions of the same camera in an hour period in two scenarios. The graph (1) shows the non-detections included in the calculation. The graph (2) represents excluding non-detections.

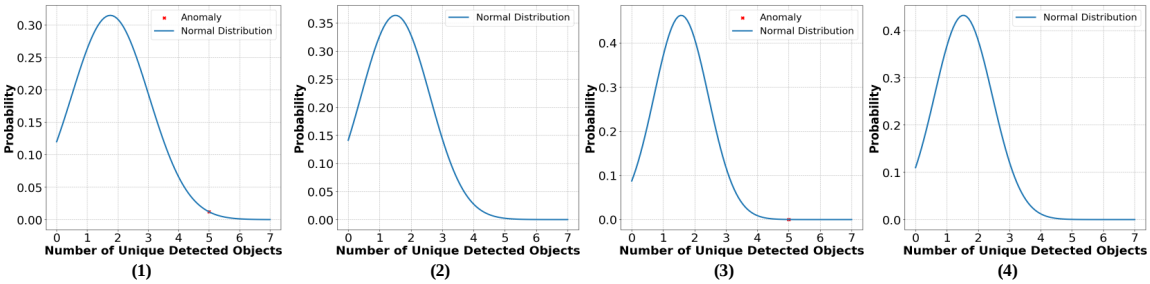

Figure 5.3: Comparing the normal distributions of the same camera in four scenarios. The graph (1) shows when statistical anomaly happens during weekdays. The graph (2) shows the normal distribution of an hour during weekdays. The graph (3) represents when statistical anomaly happens during weekends. The graph (4) represents the normal distribution of an hour during weekends.

Algorithm 2 Statistical Anomaly Detection

```
Input: df, start_time, end_time
Filter df for camera ID in the specified time range
Initialize mean to 0 and std to 1
Initialize detected_objects list
for each 5-second interval from start_time to end_time do
    Count unique detected objects in the interval
    if detected objects > 0 then
        Update detected_objects, mean, and std
    end if
end for
```

threats, ensuring the well-being and security of the public.

Algorithm 2 identifies unexpected numbers of detected individuals for each camera relative to historical trends. Recognizing that different locations display varying crowd densities over time, we compute these statistical anomalies hourly. The strategy involves constructing a normal distribution based on historical hourly data. Every 5 seconds, as new detections occur, this distribution is updated. As delineated in Algorithm 2, the mean and standard deviation for a specific hour are computed to characterize the data's distribution. Concurrently, current detections are compared against this historical backdrop. If the present number of detections exceeds two standard deviations from the mean, there's a less than 0.05 probability of such an occurrence. This implies that, with 95% confidence, such an event can be labeled a statistical anomaly.

We've intentionally omitted detections amounting to `0` in our approach. This decision stems from the rationale that, within a 5-second window, the absence of detections (i.e., detecting no individuals) is more probable than any detection. Including `0` detections in our computations would skew the results towards `0`, introducing a bias. This inclusion would result in diminished thresholds for defining anomalies, rendering the statistical analysis less meaningful and effective. Figure 5.2, compares the normal distributions of the detected objects under these two scenarios for the same camera during the same period. We did not exclude the zeros from our computations in the

graph (1). The graph (2) does not include zeros. We can see the skewness of graph (1) toward Zero, resulting in the mean between 0 and 1. In the other scenario, however, the mean is between 1 and 2. As a result, detecting 3 objects in the first scenario is considered an anomaly, while detecting the same number of objects in the second scenario is considered normal.

In Figure 5.3, we present a comparative analysis of four distinct scenarios captured at two separate temporal intervals of the same camera. Graphs (1) and (2) derive from weekday data. Specifically, Graph (1) illustrates a statistical anomaly, evidenced by detecting 5 individuals within 5 seconds, deviating from the expected probability distribution. In contrast, Graph (2) delineates the typical distribution over an hour-long weekday period. Graphs (3) and (4) are predicated on weekend data, with Graph (3) highlighting another statistical anomaly with 5 people detected. Graph (4) portrays the standard distribution over an hour during weekends. A comparative assessment of Graphs (2) and (4) reveals a higher mean value during weekdays for Graph (2), aligning with anticipated trends.

5.2.3 Bird's Eye View

A bird's eye view, often referred to as a top-down or overhead view, offers a unique vantage point that eliminates perspective distortion commonly associated with ground-level or diagonal images. This perspective allows for accurate spatial representation, ensuring that objects' relative positions and distances from one another are preserved [57]. In surveillance or monitoring, a bird's eye view ensures that the entire area of interest is observed without any hidden spots or overlapping regions, a feature often compromised in traditional camera views due to their limited field of view and perspective distortion [58].

Understanding how people are scattered in a specific area is crucial for various applications such as crowd management, security, and area planning. Accurate representation of people's positions helps determine crowd densities, identify potential

Algorithm 3 Bird's Eye View Transformation

Input: data frame df, camera_width, camera_height, min_theta, max_theta
Compute normalized values: df[normalized_W], df[normalized_H]
for each object in df **do**
    Compute $scale\_factor$ using df[normalized_H]
    Calculate centroid C_X and C_Y
    Compute $BirdsEye_X$ and $BirdsEye_Y$ using $scale\_factor$ and normalized values
end for
for camera_id from 1 to 8 **do**
    Define start_time and end_time
    Filter df based on camera_ID and time range
end for

choke points, and facilitate effective emergency responses. By utilizing a bird's eye view, these patterns can be more easily discerned, leading to better decision-making and prediction of crowd behaviors.

In Algorithm 3, a systematic approach is taken to calculate the Bird's Eye View coordinates for objects detected by cameras of the model AXIS P3225-VE Mk II Network Camera [1]. The primary objective of this transformation is to project the detected objects onto a top-down view, simulating an overhead perspective.

The first step involves normalizing the object's width and height by dividing them by the camera's resolution parameters, namely, `camera_width` and `camera_height`. This normalization ensures that the object dimensions are represented as scale-independent relative to the camera's resolution.

Next, a scale factor is computed for each object, a function of its normalized height and the angular field of view parameters, `min_teta` and `max_teta`. Notably, these angular parameters, `min_teta` and `max_teta`, are derived directly from the technical specifications provided by the camera producer. The scale factor is crucial for adjusting the object's dimensions in the Bird's Eye View, ensuring that objects farther away appear smaller than those closer to the camera, thereby preserving depth perception.

The centroid of each object is then determined by calculating the average of its

[1] https://www.axis.com/products/axis-p3225-ve-mk-ii/support#technical-specifications

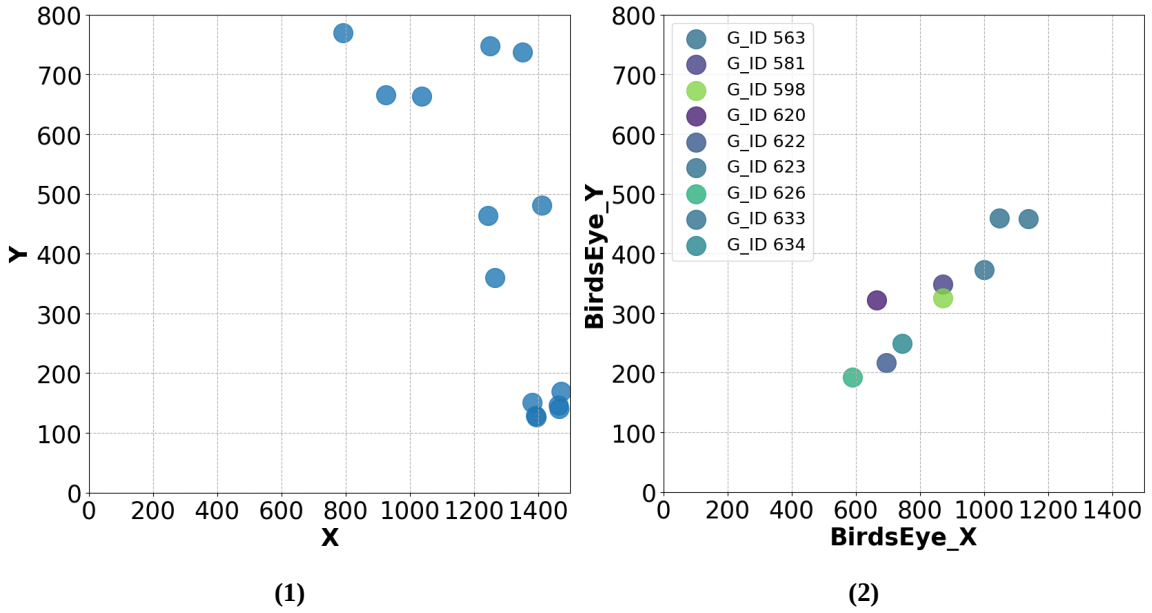

Figure 5.4: Comparing different views of camera 8 from 2023-10-17 12:17:42 to 12:17:47. The graph (1) generated directly from database shows the original camera view. The graph (2) represents the average birdseye view coordinates.

width and height dimensions. This centroid represents the object's central point, essential for accurate positioning in the Bird's Eye View.

Finally, the Bird's Eye View coordinates, `BirdsEye_X` and `BirdsEye_Y`, are calculated by applying the previously computed scale factor to the object's centroid coordinates. The result of this transformation is an accurate representation of the object's position in a top-down perspective.

The algorithm also includes a data filtering step, where records are filtered based on specific camera IDs and a specified time frame. Therefore, in every 5-second interval, we calculated the unique number of `global_IDs`, and based on that, we captured the number of detected objects. This ensures that only relevant data for a given camera and time range is considered for further analysis.

Figure 5.4 presents a comparative representation between the original perspective and the processed birdseye view from Camera 8, captured concurrently. The initial graph displays approximately 13 data points within a five-second interval. Post-processing, the bounding boxes corresponding to identical global IDs are averaged,

resulting in a birdseye representation of nine distinct individuals. This transformation underscores the efficacy of the bird's eye view process in refining and consolidating data for enhanced clarity and precision.

CHAPTER 6: Real-World Results and Evaluation

6.1 AI Pipeline Configuration Evaluation

Two challenging videos, Extreme Level and Heavy Level, depicting high crowd levels, were designated as input to facilitate a performance comparison across various methods. Each video lasted 60 seconds, operating at a frame rate of 60 per second, resulting in 120 batches for inference. To ensure statistical robustness, performance metrics were averaged over the middle 80 batches in each run, thus mitigating potential bias arising from the initial 20 batches (warm-up) and including the last 20 batches (cool-down).

Four distinct methods underwent testing:

- YOLOv5+HRNet: The original system method
- YOLOv8pose: A variant of YOLOv8 incorporating human pose estimation as a top-down method.
- YOLOv8pose-p6: The most complex model of YOLOv8 for human pose estimation.
- YOLOv8+HRNet: Object detection method with pedestrian tracking interacted.

For the AI Pipeline Configuration Test, Table 6.1 outlines the performance of each method across different crowd density levels using four distinct metrics: average latency and throughput, total detection count, average GPU and CPU memory usage, and total FLOPs of every AI models. For the "Extreme" crowd density video, YOLOv8pose exhibited superior latency and throughput performance, accompanied by the fewest detections. This method leverages a top-down approach, excelling

Table 6.1: Performance comparison among different system configurations. Data collected using lab server with a single node.

Crowd Density	Method	Latency (s)	FPS	Detections (counts)	GPU Memory (G)	CPU Memory (G)	Total FLOPs (B)
Extreme (∼50 detects per second)	YOLOv5+HRNet	14	14.82	1033	**5.3**	19.1	**126.08**
	YOLOv8pose-p6	3.7	26.39	911	12.2	12.68	1067.38
	YOLOv8pose	**2.2**	**49.07**	679	6.9	**12.38**	264.18
	YOLOv8+HRNet	*9.09*	*19*	***1217***	*5.5*	*19.3*	*182.18*
Heavy (∼20 detects per second)	YOLOv5+HRNet	3.47	**81.2**	237	**5.0**	15.28	**126.08**
	YOLOv8pose-p6	3.54	26.45	**358**	11.9	12.4	1067.38
	YOLOv8pose	**2.6**	45.11	308	6.9	**12.3**	264.18
	YOLOv8+HRNet	*2.76*	*67.63*	*342*	*5.5*	*15.4*	*182.18*

at processing larger human crops that show full-body movement while potentially ignoring smaller or just upper-body ones. In the real world, missing potential human detections could result in undetected objects and behavioral anomalies, which is unacceptable. Conversely, YOLO + HRNet enables the detection of more crops. In this context, YOLOv8+HRNet demonstrated a latency of 9.09 seconds, a throughput of 19 FPS, and comparable GPU memory usage of around 5.5 GB, better than the YOLOv5+HRNet method.

In the case of the "Heavy" crowd density video, where crowd density is lower than "Extreme," the YOLOv8+HRNet method maintained its competitive edge. With a latency of 2.76 seconds, a throughput of 67.63 FPS, 342 total detections, 5.5 GB GPU memory usage, and 15.4 GB CPU memory usage, it showcased a formidable performance against other methods.

6.2 Load Stress Evaluation

To rigorously assess the system's performance under varying crowd densities and parallel module operations, we curated ten videos with density levels ranging from 0 to 9. Each video lasted 150 seconds and operated at a frame rate of 30 frames per second. To ensure the statistical validity of our results, we averaged metrics over the central 100 batches in each run, excluding the initial 25 batches to account for system warm-up and the last 25 batches for cool-down effects.

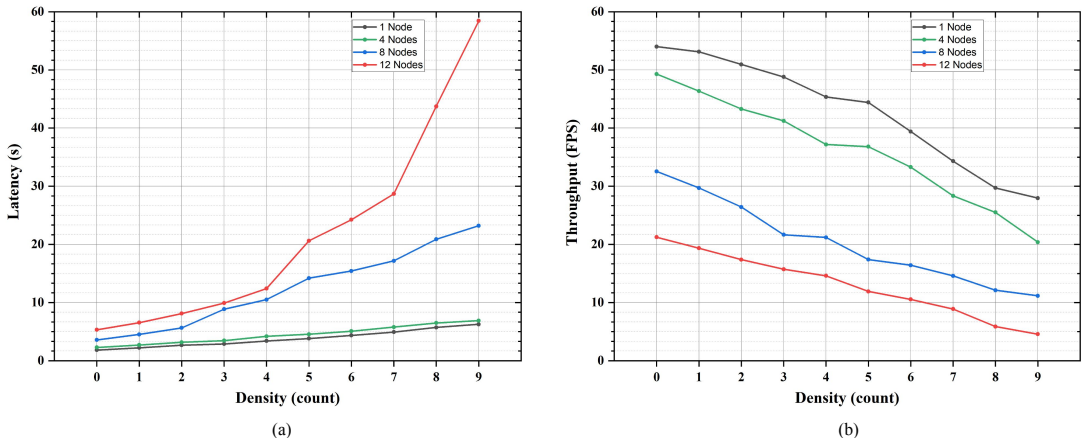

Figure 6.1: Throughput and latency trends with respect to crowd densities across different nodes number running in parallelism.

Figure 6.1 graphically presents the trends for throughput and latency. The X-axis, labeled "Density" (count), represents the average number of humans detected per batch (comprising 30 frames) for each input module in the experiment.

In tests with one and four cameras, our system maintained latency under 10 seconds, showcasing stability. However, with eight module s, latency increased beyond 20 seconds at higher density levels, particularly from level 5 onwards. A notable decrease in latency was observed with 12 nodes, especially at density levels 8 and 9, correlating with processing 108 individuals (12 nodes x 9 density). Throughput consistently declined linearly with increasing density and node count, reaching a low of 4.56 FPS at the highest density level 9 with 12 nodes. The observed decrease in performance can be primarily attributed to the computational demands of HRNet operating high density for keypoint processing. Additionally, the system's performance is constrained by the CPU computation limits. As the number of nodes and density increases, these factors collectively contribute to the bottleneck, leading to the observed decline in system throughput and latency.

6.3 Real-World Endurance Evaluation

The primary objective of this experiment is to validate the system's durability and capacity to maintain optimal performance within a realistic, sustained operational configuration. Figure 6.2 offers a detailed presentation of the latency and throughput trends with varying crowd density traffic. The experiment encompassed a continuous 21-hour runtime, employing different camera node configurations. The evaluation commenced at 2:00 p.m. on the first day and concluded at 11:00 a.m. the following day. During the experiment, we observed that system latency decreased when the server node's database reached around 50,000 queries. To address this and maintain privacy, we implemented an auto-reset feature for the database. It clears and resets after accumulating 50,000 queries or after 24 hours of run time.

Analyzing Figure 6.2, each data point epitomizes the average value over a 60-second interval. The "Density" metric signifies the sum of human detections per camera node within the middle 60-second window. This is attributed to the data collection period aligning with a summer break, resulting in lower human flow within the testbed, particularly about indoor camera nodes.

For configurations with 8, 12, and 16 camera nodes, we observed varying performance metrics: average latencies were 2.6-4.8s, 5.3-6.5s, and 6.7-10.5s, respectively, while corresponding throughputs ranged from 28.5-26.5 FPS, 20.5-18 FPS, to 16.5-14.5 FPS. As shown in Figure 6.3, the system's performance over a week, running with eight cameras, demonstrated consistent latency and throughput results about crowd density. Specifically, the latency ranged from 2.5-7.8s, and throughput varied from 22.3-29.8 FPS. These findings confirm that irrespective of crowd density, the latency increased. At the same time, the FPS decreased, aligning with expectations and underscoring that the length of the videos does not significantly impact the system's performance.

It is essential to note that the specific data spikes observed within the metrics for the eight and twelve-camera node configurations could stem from various factors,

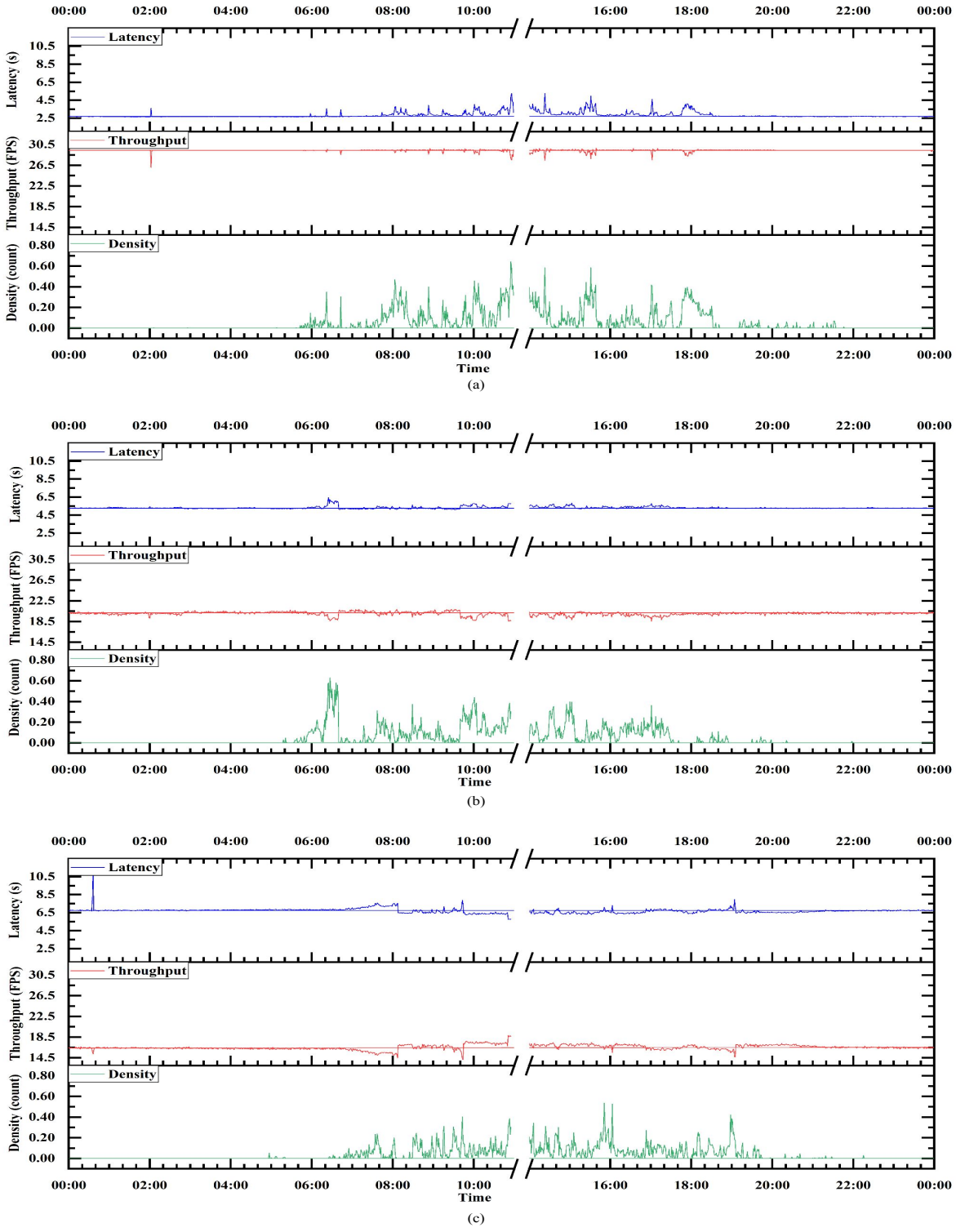

Figure 6.2: Latency and throughput trends concerning crowd densities during a 21-hour-length period for different camera nodes number (a) represents 8 camera nodes, (b) represents 12 camera nodes, (c) represents 16 camera nodes

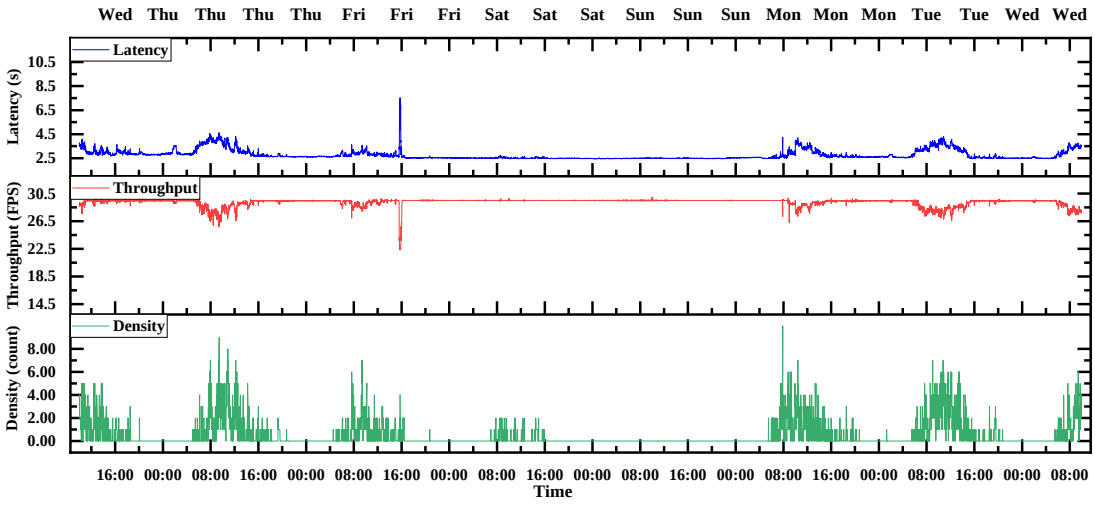

Figure 6.3: Latency and throughput trends concerning crowd densities during a week-long period for 8 camera nodes

including network irregularities within the environment (as the input streams originate from IP cameras) or CPU memory usage nearing its limits. However, in light of the comprehensive experiment, these occasional data point spikes remain within acceptable bounds. The system consistently maintained optimal performance levels, showcasing its robustness and capacity to effectively manage and adapt to various operational scenarios.

Knowing how much these key parameters, such as the number of cameras and crowd density, could affect system performance, such as latency and throughput in different public safety scenarios in case of emergency, is crucial in decision making and taking the proper actions. For example, retail and transport hubs could analyze foot traffic and crowd density, offering insights into peak hours and emergency response planning.

6.4 Physical-Cyber-Physical Evaluation

In this section, we dive into the details of setup and evaluation of both behavioral and object anomaly detection. We measure PCP latency, representing the total end-to-end time from when an anomaly appears in front of the camera to when the end user receives a notification.

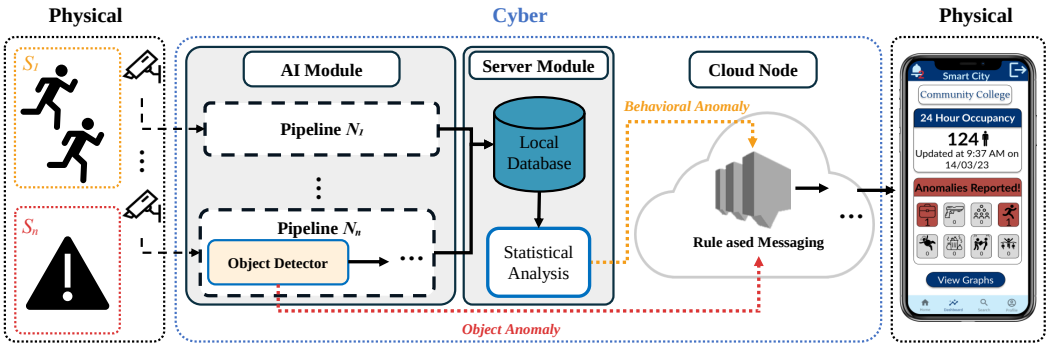

Figure 6.4: Notification's data flow through Physical-Cyber-Physical set up. S_1 represents the first scene that detects a behavior anomaly. The yellow line shows the notification data flow to the notification service. S_n shows the n^{th} scene where a suspicious object has been detected. The red line shows the notification data flow when detecting object anomalies.

In practical SVS deployments, using a small number of camera nodes is uncommon. Our tests involved four, eight, twelve, and sixteen nodes and simulated two distinct types of notifications: object anomaly and behavioral anomaly. The former measures the latency between detecting high-priority objects and alerting the end user, while the latter focuses on dangerous activities such as fighting or falling. Throughout these tests, scene density was carefully controlled, involving two individuals and three end-user devices receiving notifications. 400 data samples were collected across twenty experiments for each type of anomaly, offering a robust measure of the system's PCP latency under various conditions. In Figure 6.4, S_n with red dots represent object anomalies, notifications are directly transmitted through the notification service as soon as an anomaly object is detected from an object detector stage from AI node. S_1 with yellow dots represents behavioral anomalies that are subject to further scrutiny to minimize False Negatives (FN) and False Positives (FP). The behavioral anomaly score is compared with the two preceding scores in the server node database to ensure consistent anomaly detection. Once the analyzed anomaly score crosses a predefined threshold, notifications are dispatched via the notification service, yellow dotted line in Figure 6.4. The average PCP latency is shown in Table 6.2. It should be noted that

Table 6.2: Average P-C-P latency with different number of cameras

	Physical-Cyber-Physical Latency (s)	
Node	Object Anomaly	Behaviour Anomaly
4	4.7	9.34
8	10.99	14.53
12	15.76	18.45
16	19.68	26.76

Table 6.3: Average latency of cloud server with different number of cameras

	DynamoDB Latency (ms)		SNS Latency (ms)	AppSync Latency (ms)	
Nodes	GetItem	PutItem		Action	Statistical
4	14.6	17.5	140	105	14.4
8			150.5		
12			186		
16			172		

the timing data were manually recorded, as there is currently no scientific method for achieving extreme accuracy in measuring PCP latency. This manual approach may introduce some minor discrepancies in the time measurements. In the experiment, the camera node density was controlled as two humans in front of the camera. Each test was conducted a minimum of 20 times, with notifications transmitted to three end-user devices. The increase in camera numbers significantly affects the results. For instance, the object anomaly PCP latency rose from 4.7 seconds with four cameras to 19.68 seconds with sixteen. Given that each AI pipeline consumes up to 5.5 GB of GPU memory and the local node can handle a maximum of 24 GB, a GPU can support up to four pipelines simultaneously. Thus, the local node server's maximum capacity is limited to sixteen parallel AI pipelines. A similar trend is observed in behavioral anomaly latency, increasing from 9.34 seconds (4 cameras) to 26.76 seconds (16 cameras). These patterns are attributed to increased system traffic, overall latency increments (as shown in Table 6.3), and the multitasking overhead, ranging from footage loading to notification publishing on user devices. On the other hand, Table 6.2 indicates that the average object detection PCP latency for four cameras is around 4.7 seconds. In contrast, the behavioral anomaly PCP latency shows a 4-second

Table 6.4: Statistical PCP latency data for object and behaviour anomaly for different number of cameras

	Object Anomaly			Behaviour Anomaly		
Nodes	Min (s)	Max (s)	Standard deviation	Min (s)	Max (s)	Standard deviation
4	3.06	5.55	0.65	7.81	10.41	0.68
8	9.18	12.55	0.84	13.23	16.46	0.71
12	14.05	18.31	1.04	16.32	20.35	0.97
16	17.67	21.6	1.12	24.23	29.54	1.52

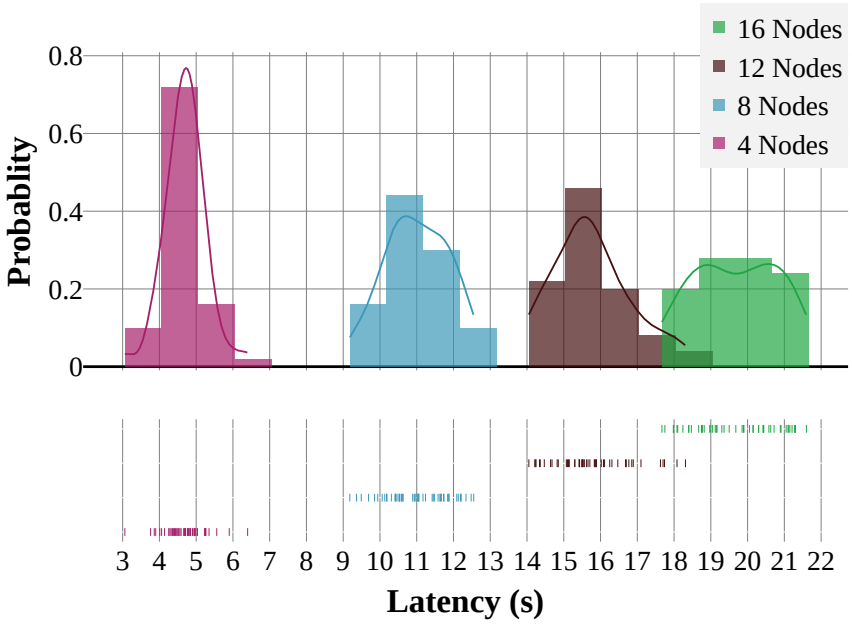

Figure 6.5: Object Anomaly Latency Rug plot with each run during PCP test. 50 data points for each camera number

increment. This observation is also similar among different experiments. The reason is that the behavioral anomaly notification needs to run through more stages and be recorded in the server node before being pushed to the notification service at the cloud node, as shown in Figure 6.4.

It's important to note that a latency of approximately three seconds was observed in the frame transmission from the cameras to the AI Nodes. This latency was slightly higher compared to other IP camera systems. The primary reason for this elevated latency can be attributed to the unique network configuration implemented within our

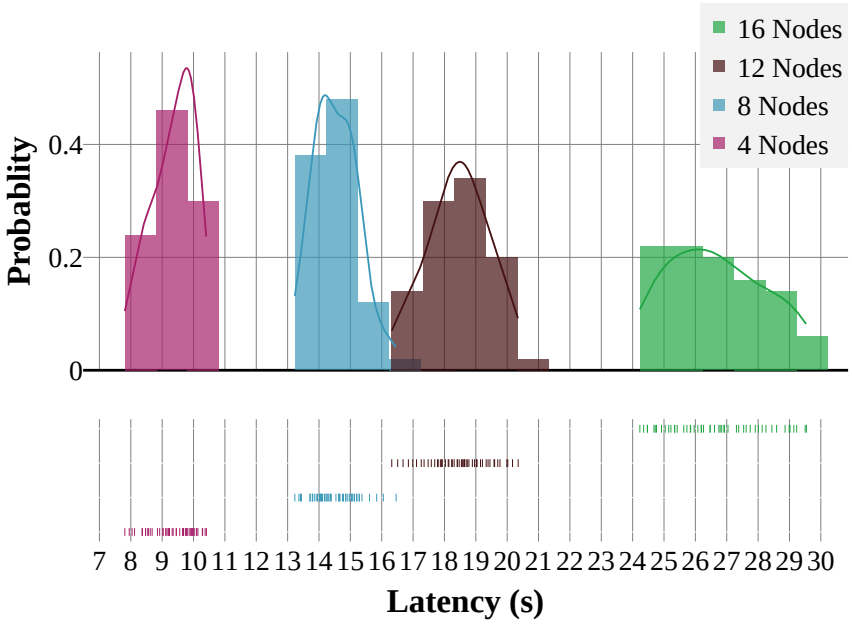

Figure 6.6: Behavior Anomaly Latency Rug plot with each run during PCP test. 50 data points for each camera number

testbed environment. Importantly, it is imperative to note that the PCP measurement results encompassed this latency incurred by the cameras.

Figures 6.5 and 6.6 show the distribution of 400 data points from our PCP experiments. Figure 6.5 illustrates that data points in the four-camera object detection PCP are less varied compared to the more scattered data in the sixteen-camera experiment. As reported in Table 6.4, the standard deviation (sd) of object detection PCP increases from 0.65 to 1.12 as camera numbers rise from four to sixteen. The minimal latency for detecting a suspicious object was 3.06s (four cameras) and 9.18s (eight cameras). The slight increase in sd across different node quantities indicates that increased node numbers and system traffic marginally broaden latency ranges, possibly due to increased system complexity, synchronization issues, resource contention, and network congestion.

Our system's evaluation of PCP latency with different node numbers at a real-world testbed demonstrates its utility in enhancing safety and operational efficiency. For

event venues and public spaces, the system could manage different locations, aiding in anomaly detection and resource allocation by notifying all the end users around 20 seconds, which could prevent further loss from danger. This analysis could provide communities with a clearer vision for managing and responding to anomalies with the SVS system, thereby bolstering public safety and efficiency.

CHAPTER 7: CONCLUSION AND FUTURE

SVS systems have emerged as powerful tools for enhancing public safety and security, with advancements in machine learning algorithms bolstering their reliability. As this technology is poised for substantial growth, it is crucial to address challenges related to scalability, latency, throughput, and privacy concerns for practical deployment. This thesis evaluated an SVS system's performance in a real world setting, demonstrating its capacity to deliver timely alerts with an end-to-end latency of under 30 seconds across 16 cameras. Comparatively, this system outperforms traditional police response benchmarks [1] [2], which are typically measured in minutes.

The potential of AI-driven SVS systems to improve urban safety and planning is profound. By integrating surveillance data with IoT sensors, we can achieve comprehensive environmental monitoring, capturing essential metrics such as air quality and noise levels. Furthermore, the use of better resource allocation algorithms can enhance the responsiveness of emergency services and traffic management systems in critical situations. Advanced computer vision techniques can also be employed to detect suspicious activities in densely populated areas, thereby bolstering public safety.

Additionally, the creation of simulation models for urban environments enables the testing of various planning scenarios, while the adoption of privacy-preserving techniques ensures the responsible usage of collected data. The integration of multimodal data sources, including transportation and weather information, provides a holistic view of urban dynamics, aiding informed decision-making.

[1] https://www.alicetraining.com/wp-content/uploads/2018/12/ALICE-Fact-Sheet-with-references.pdf

[2] https://leb.fbi.gov/image-repository/police-response-time-to-active-shooter-attacks.jpg/view

This study underscores that AI-driven SVS systems can be effectively implemented in real-world settings, enhancing surveillance capabilities while addressing privacy concerns. Future research should focus on refining algorithms, involving local communities in the deployment of SVS systems, and exploring the concept of "community-in-the-loop" to tackle scalability and privacy challenges. Continuously evaluating SVS performance across diverse environments will further enhance the system's reliability and efficiency.

REFERENCES

[1] D. Fraser, *Goals for MinneapolisâA City for the 21st Century*, pp. 83–103. Routledge, 2018.

[2] S. Nikouei, Y. Chen, S. Song, B. Choi, and T. Faughnan, "Toward intelligent surveillance as an edge network service (isense) using lightweight detection and tracking algorithms," *IEEE Transactions on Services Computing*, vol. 14, no. 6, pp. 1624–1637, 2019.

[3] N. Dilshad, J. Hwang, J. Song, and N. Sung, "Applications and challenges in video surveillance via drone: A brief survey," in *2020 International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 728–732, IEEE, 2020.

[4] J. Tang, Y. Zhou, T. Tang, D. Weng, B. Xie, L. Yu, H. Zhang, and Y. Wu, "A visualization approach for monitoring order processing in e-commerce warehouse," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 1, pp. 857–867, 2021.

[5] A. De Biasi, J. M. Krupa, A. Rzotkiewicz, E. McGarrell, G. Circo, and J. Liebler, "Insights from law enforcement personnel during the covid-19 pandemic: a focus on violence reduction and prevention initiatives and firearm violence," *Police Practice and Research*, vol. 24, no. 3, pp. 322–345, 2023.

[6] T. RÃ€ty, "Survey on contemporary remote surveillance systems for public safety," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 5, pp. 493–515, 2010.

[7] N. Komninos, E. Philippou, and A. Pitsillides, "Survey in smart grid and smart home security: Issues, challenges and countermeasures," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 4, pp. 1933–1954, 2014.

[8] B. Ardabili, A. Pazho, G. Noghre, C. Neff, S. Bhaskararayuni, A. Ravindran, S. Reid, and H. Tabkhi, "Understanding policy and technical aspects of ai-enabled smart video surveillance to address public safety," *arXiv preprint*, vol. arXiv:2302.04310, 2023.

[9] R. Xu, S. Nikouei, Y. Chen, A. Polunchenko, S. Song, C. Deng, and T. Faughnan, "Real-time human objects tracking for smart surveillance at the edge," in *2018 IEEE international conference on communications (ICC)*, pp. 1–6, IEEE, 2018.

[10] R. F. Mansour, J. Escorcia-Gutierrez, M. Gamarra, J. A. Villanueva, and N. Leal, "Intelligent video anomaly detection and classification using faster rcnn with deep reinforcement learning model," *Image and Vision Computing*, vol. 112, p. 104229, 2021.

[11] A. Anagnostopoulos, B. E. Griffiths, N. Siachos, J. Neary, R. F. Smith, and G. Oikonomou, "Initial validation of an intelligent video surveillance system for automatic detection of dairy cattle lameness," *Frontiers in Veterinary Science*, vol. 10, p. 1111057, 2023.

[12] C. Huang, Z. Wu, J. Wen, Y. Xu, Q. Jiang, and Y. Wang, "Abnormal event detection using deep contrastive learning for intelligent video surveillance system," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 8, pp. 5171–5179, 2021.

[13] J. Usha Rani and P. Raviraj, "Real-time human detection for intelligent video surveillance: An empirical research and in-depth review of its applications," *SN Computer Science*, vol. 4, no. 3, p. 258, 2023.

[14] A. Pazho, C. Neff, G. Noghre, B. Ardabili, S. Yao, M. Baharani, and H. Tabkhi, "Ancilia: Scalable intelligent video surveillance for the artificial intelligence of things," *arXiv preprint*, vol. arXiv:2301.03561, 2023.

[15] E. O. Asani, O. D. Akande, E. E. Okosun, O. T. Olowe, R. O. Ogundokun, and A. E. Okeyinka, "Ai-paas: Towards the development of an ai-powered accident alert system," in *2023 International Conference on Science, Engineering and Business for Sustainable Development Goals (SEB-SDG)*, vol. 1, pp. 1–8, IEEE, 2023.

[16] A. Dash, S. Bandopadhay, S. R. Samal, and V. Poulkov, "Ai-enabled iot framework for leakage detection and its consequence prediction during external transportation of lpg," *Sensors*, vol. 23, no. 14, p. 6473, 2023.

[17] V. Padmavathi and R. Kanimozhi, "Pandemic alert with smart covid-19 using blockchain-powered digital twins' collaboration," 2023.

[18] H. Cui, K. Hou, J. Zhang, S. Yan, M. Seraj, Y. Wang, M. Tavakoli, and T. Qiu, "Vision-based work zone safety alert system in a connected vehicle environment," *Transportation Research Record*, p. 03611981231165997, 2023.

[19] T. Dutta, A. Soni, P. Gona, and H. P. Gupta, "Real testbed for autonomous anomaly detection in power grid using low-cost unmanned aerial vehicles and aerial imaging," *IEEE MultiMedia*, vol. 28, no. 3, pp. 63–74, 2021.

[20] D. L. Marino, C. S. Wickramasinghe, V. K. Singh, J. Gentle, C. Rieger, and M. Manic, "The virtualized cyber-physical testbed for machine learning anomaly detection: A wind powered grid case study," *IEEE Access*, vol. 9, pp. 159475–159494, 2021.

[21] L. Maglaras, T. Cruz, M. A. Ferrag, and H. Janicke, "Teaching the process of building an intrusion detection system using data from a small-scale scada testbed," *Internet Technology Letters*, vol. 3, no. 1, p. e132, 2020.

[22] M. Zhang, J. Cao, Y. Sahni, Q. Chen, S. Jiang, and L. Yang, "Blockchain-based collaborative edge intelligence for trustworthy and real-time video surveillance," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 2, pp. 1623–1633, 2022.

[23] B. Ardabili, A. Pazho, G. Noghre, C. Neff, A. Ravindran, and H. Tabkhi, "Understanding ethics, privacy, and regulations in smart video surveillance for public safety," *arXiv preprint*, vol. arXiv:2212.12936, 2022.

[24] M. S. Mahdavinejad, M. Rezvan, M. Barekatain, P. Adibi, P. Barnaghi, and A. P. Sheth, "Machine learning for internet of things data analysis: A survey," *Digital Communications and Networks*, vol. 4, no. 3, pp. 161–175, 2018.

[25] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things journal*, vol. 1, no. 1, pp. 22–32, 2014.

[26] M. M. Rathore, A. Ahmad, A. Paul, and S. Rho, "Urban planning and building smart cities based on the internet of things using big data analytics," *Computer networks*, vol. 101, pp. 63–80, 2016.

[27] E. Al Nuaimi, H. Al Neyadi, N. Mohamed, and J. Al-Jaroodi, "Applications of big data to smart cities," *Journal of Internet Services and Applications*, vol. 6, no. 1, pp. 1–15, 2015.

[28] M. N. Mouchili, S. Aljawarneh, and W. Tchouati, "Smart city data analysis," in *Proceedings of the First International Conference on Data Science, E-learning and Information Systems*, pp. 1–6, 2018.

[29] L. Abberley, N. Gould, K. Crockett, and J. Cheng, "Modelling road congestion using ontologies for big data analytics in smart cities," in *2017 international smart cities conference (isc2)*, pp. 1–6, IEEE, 2017.

[30] N. Bharadwaj, P. Kumar, S. Arkatkar, A. Maurya, and G. Joshi, "Traffic data analysis using image processing technique on delhi–gurgaon expressway," *Current Science*, pp. 808–822, 2016.

[31] L. Chen, Y. Lu, Q. Sheng, Y. Ye, R. Wang, and Y. Liu, "Estimating pedestrian volume using street view images: A large-scale validation test," *Computers, Environment and Urban Systems*, vol. 81, p. 101481, 2020.

[32] L. He, A. Páez, and D. Liu, "Built environment and violent crime: An environmental audit approach using google street view," *Computers, Environment and Urban Systems*, vol. 66, pp. 83–95, 2017.

[33] A. Pazho, G. Noghre, A. Purkayastha, J. Vempati, O. Martin, and H. Tabkhi, "A survey of graph-based deep learning for anomaly detection in distributed systems," *arXiv preprint*, vol. arXiv:2206.04149, 2022.

[34] J. S., S. C., Y. E., and J. GP., "Real time object detection and trackingsystem for video surveillance system," *Multimedia Tools and Applications*, vol. 80, pp. 3981–96, 2021.

[35] A. Pramanik, S. Sarkar, and J. Maiti, "A real-time video surveillance system for traffic pre-events detection," *Accident Analysis and Prevention*, vol. 154, p. 106019, 2021.

[36] R. Singh, H. Srivastava, H. Gautam, R. Shukla, and R. Dwivedi, "An intelligent video surveillance system using edge computing based deep learning model," in *2023 International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT)*, pp. 439–444, IEEE, 2023.

[37] M. Gandapur, "E2e-vsdl: End-to-end video surveillance-based deep learning model to detect and prevent criminal activities," *Image and Vision Computing*, vol. 123, p. 104467, 2022.

[38] C. Huang, Z. Wu, J. Wen, Y. Xu, Q. Jiang, and Y. Wang, "Abnormal event detection using deep contrastive learning for intelligent video surveillance system," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 8, pp. 5171–5179, 2021.

[39] M. Ma, S. M. Preum, M. Y. Ahmed, W. Tärneberg, A. Hendawi, and J. A. Stankovic, "Data sets, modeling, and decision making in smart cities: A survey," *ACM Trans. Cyber-Phys. Syst.*, vol. 4, nov 2019.

[40] A. Alshammari and D. Rawat, "Intelligent multi-camera video surveillance system for smart city applications," in *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, pp. 0317–0323, IEEE, 2019.

[41] R. Franklin and V. Dabbagol, "Anomaly detection in videos for video surveillance applications using neural networks," in *2020 Fourth International Conference on Inventive Systems and Control (ICISC)*, pp. 632–637, IEEE, 2020.

[42] Y. Yuan, M. Ma, S. Han, D. Zhang, F. Miao, J. A. Stankovic, and S. Lin, "Deresolver: A decentralized conflict resolution framework with autonomous negotiation for smart city services," *ACM Trans. Cyber-Phys. Syst.*, vol. 6, nov 2022.

[43] L. Esterle and J. N. A. Brown, "I think therefore you are: Models for interaction in collectives of self-aware cyber-physical systems," *ACM Trans. Cyber-Phys. Syst.*, vol. 4, jun 2020.

[44] R. Wang, W. Tsai, J. He, C. Liu, Q. Li, and E. Deng, "A video surveillance system based on permissioned blockchains and edge computing," in *2019 IEEE international conference on big data and smart computing (BigComp)*, pp. 1–6, IEEE, 2019.

[45] P. Ingle and Y. Kim, "Real-time abnormal object detection for video surveillance in smart cities," *Sensors*, vol. 22, no. 10, p. 3862, 2022.

[46] M. Maqsood, S. Yasmin, S. Gillani, M. Bukhari, S. Rho, and S. Yeo, "An efficient deep learning-assisted person re-identification solution for intelligent video surveillance in smart cities," *Frontiers of Computer Science*, vol. 17, no. 4, p. 174329, 2023.

[47] G. Jocher, A. Chaurasia, A. Stoken, J. Borovec, NanoCode012, Y. Kwon, K. Michael, TaoXie, J. Fang, imyhxy, Lorna, Z. Yifu, C. Wong, A. V, D. Montes, Z. Wang, C. Fati, J. Nadar, Laughing, UnglvKitDe, V. Sonck, tkianai, yxNONG, P. Skalski, A. Hogan, D. Nair, M. Strobel, and M. Jain, "ultralytics/yolov5: v7.0 - yolov5 sota realtime instance segmentation." `https://doi.org/10.5281/zenodo.7347926`, 2022.

[48] G. Jocher, A. Chaurasia, and J. Qiu, "Yolo by ultralytics (version 8.0.0)." `https://github.com/ultralytics/ultralytics`, 2023.

[49] Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang, "Bytetrack: Multi-object tracking by associating every detection box," in *Computer VisionâECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23â27, 2022, Proceedings, Part XXII*, pp. 1–21, Cham: Springer Nature Switzerland, 2022.

[50] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *CVPR*, 2019.

[51] A. Markovitz, G. Sharir, I. Friedman, L. Zelnik-Manor, and S. Avidan, "Graph embedded pose clustering for anomaly detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10539–10547, 2020.

[52] K. Zhou, Y. Yang, A. Cavallaro, and T. Xiang, "Omni-scale feature learning for person re-identification," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 3702–3712, 2019.

[53] F. Dahunsi, J. Idogun, and A. Olawumi, "Commercial cloud services for a robust mobile application backend data storage," *Indonesian Journal of Computing, Engineering and Design (IJoCED)*, vol. 3, no. 1, pp. 31–45, 2021.

[54] S. Mohammed, J. Fiaidhi, D. Sawyer, and M. Lamouchie, "Developing a graphql soap conversational micro frontends for the problem oriented medical record (ql4pomr)," in *Proceedings of the 6th International Conference on Medical and Health Informatics*, pp. 52–60, 2022.

[55] H. Lessa, "Production-grade full-stack apps with aws amplify," in *AWS Re: Invent 2019*, 2019.

[56] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi, "Performance measures and a data set for multi-target, multi-camera tracking," in *European conference on computer vision*, pp. 17–35, Cham: Springer International Publishing, 2016.

[57] R. Liu, X. Wang, W. Wang, and Y. Yang, "Bird's-eye-view scene graph for vision-language navigation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10968–10980, 2023.

[58] N. Dubos, X. Porcel, M. A. Roesch, J. Claudin, R. Pinel, J.-M. Probst, and G. Deso, "A bird's-eye view: Evaluating drone imagery for the detection and monitoring of endangered and invasive day geckos," *Biotropica*, 2023.