PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
الجمهـــوريـــــة الجزائـريـــــة الديـمقراطيـــــة الشعبيـــــة
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
وزارة التعليـم العالـــي والبحـــث العلمـي
HIGHER SCHOOL OF COMPUTER SCIENCE AND DIGITAL TECHNOLOGIES-BEJAIA
المـــدرسة العـــليا في علـــوم وتكنـولوجيـات الاعـــلام الالـــي بجـــاية

Dissertation Submitted to the Department Of Computer Science in Partial Fulfillment of the Requirements for Engineer's Degree in Computer Science

Specialty: Artificial Intelligence and Data Science

Submitted By:

Ms. MELIZOU Ouassila & Ms. TOUATI Hayet

Intelligent Video Recording Optimization using Activity Detection for ESTIN Surveillance Systems

Supervised by:

Dr. ELMIR Youssef
Estin

Members of jury:

▪ **Dr.** Leila CHELOUAH	President	Estin
▪ **Dr.** Lamia CHEKLAT	Examiner	Estin
▪ **Mr.** Anis Chawki ABBES	Examiner	Estin
▪ **Mrs.** Macilia Boukhama	Examiner	Estin

Academic year: 2023/2024

Acknowledgments

First and foremost, All Praises are for Allah almighty ,The most Gracious and most merciful, who gave us the strength, courage, health, and patience to successfully conduct our research.

In this regard, we would like to express our deepest gratitude to our esteemed supervisor, Dr. Elmir Youssef, for the time he dedicated and the valuable information he provided with interest and understanding.

We also extend our heartfelt thanks to our internship supervisor, Mr. Ridha Chekroune, for the time he invested, his assistance in the success and smooth running of our internship under the best conditions, and the precise information he shared with us. We are also deeply grateful to the examiners of our thesis and for agreeing to evaluate our work..

Moreover, we acknowledge all the teachers and administrative staff of the Graduate School of Computer Science and Digital Technologies (ESTIN of Bejaia) which contributed to our education throughout our university course.

Finally, we extend our profound and sincere thanks to our families for their encouragement and moral and financial support. We do not let this occasion pass without thanking everyone who contributed, directly or indirectly, to the completion of this thesis

Dedication

I dedicate this modest work to my dear family: my father, my mother, and my siblings. To my father, who guided me and fought for me to study and remain strong during the challenging periods of school, I want to thank him infinitely. To my dear, patient mother, thank you for your support and love. I also want to express my gratitude to my extended family—my grandparents, uncles, aunts, and cousins.

Finally, to my friends, whose love and encouragement meant a lot to me, thank you for having my back when I needed you and for listening to my exam complaints, especially my two best friends, Hanane and Kenza.

Touati Hayet.

With my sincere thanks and deep gratitude, I dedicate this humble work to my parents, who have sacrificed and given everything for my happiness and success. To my brother Mami, who has always been there for me. His unwavering belief in me and his constant encouragement have been invaluable. To my best friends Zahra, Dyhia khawla and Imene. To my cousins Nesrine, and Hanan and to all my friends

I am profoundly grateful for the unwavering support and love from my entire family, my uncles, my aunts and my cousins, whose encouragement and belief in me have been a constant source of strength. Their presence in my life has made all the difference, and I am truly blessed to have such a wonderful family.

Melizou Ouassila.

List of Figures

List of Tables

Contents

General Introduction

The widespread installation of surveillance cameras, now a common sight in our urban environment, has resulted in a massive surge of visual data. For example, it is estimated that by the end of 2024, more than 1.4 billion cameras will be installed worldwide according to Gartner, generating astronomical amounts of video data daily. This profusion of data poses major challenges in terms of storage, with hundreds of petabytes required to store this crucial information. In the face of this reality, the search for efficient methods to manage this mass of data and facilitate search becomes an imperative necessity.

Surveillance cameras, although crucial for security and surveillance, have created an information-saturated environment. Indeed, according to estimates, a single HD camera can generate nearly 650 megabytes of data per minute, which, multiplied by thousands of cameras in an urban area, creates a mountain of data that is difficult to manage. These data, which are often redundant or irrelevant, require careful consideration of how to store them optimally while ensuring the rapid retrieval of critical information.

The problem that emerges from this critical situation is centered on how to optimize the storage of surveillance camera data. How can the need to keep recordings be reconciled while avoiding the congestion of storage devices? The use of object and motion detection algorithms is emerging as a promising path to filter sequences, recording only those that are of particular interest. This approach thus poses itself as a potential response to the major challenge of information overload in the field of video surveillance.

In this study, we aim to develop an optimized solution for the existing video surveillance system at Estin, focusing specifically on enhancing storage efficiency and minimizing system complexity. The primary objective is to achieve these goals through the innovative application of object and motion detection algorithms. Our research also aims to implement a solution capable of discerning crucial scenes, thereby reducing the storage burden and facilitating efficient search operations.

This thesis systematically addresses these objectives and proposes solutions within the field of video surveillance through three main chapters:

1. State of the Art: this chapter reviews foundational concepts and various techniques used in intelligent surveillance systems. It provides a comparative analysis of their strengths and weaknesses, laying the

groundwork for our proposed optimizations.

2. Estin Surveillance System: in this chapter we conduct a detailed examination of the Estin video surveillance system. This includes defining video surveillance systems, describing their components, and evaluating different system architectures, illustrated with a specific case study.

3. Concept, Implementation, and Validation: this chapter details the design, implementation, and validation of our optimized system. It covers data preparation, algorithm selection, and rigorous evaluation to demonstrate effectiveness in enhancing storage efficiency and reducing system complexity.

We conclude with a summary of our findings and a discussion of future research directions.

Chapter 1

State of the Art

Introduction

In this chapter, we explore the current landscape of intelligent surveillance systems, examining foundational concepts and various techniques employed in the field. Our primary objective is to conduct a comprehensive review that highlights the evolution, strengths, and limitations of existing methodologies.

Through a comparative analysis, we assess the effectiveness of diverse surveillance techniques, ranging from traditional methods to advanced algorithms such as motion detection, object detection, and object tracking. This exploration not only informs our approach but also uncovers opportunities for innovation and enhancement.

By critically evaluating the state of the art, we aim to establish a robust foundation for proposing novel strategies in the optimization of video surveillance systems. This groundwork lays the foundation for subsequent chapters, where we will apply this knowledge to enhance the Estin surveillance system.

1.1 Foundations and Definitions

1.1.1 Artificial Intelligence

Artificial intelligence, or AI, enables computers and machines to simulate human intelligence and problem-solving by enabling computers and digital devices to learn, read, write, create, and analyze. capabilities [IBM24b].

1.1.2 Machine Learning

1.1.2.1 Definition

Machine learning (ML) is a subset of Artificial Intelligence (AI) and computer science that focuses on enabling AI systems to learn from data and improve over time without being explicitly programmed. It involves the use of algorithms that analyze data, identify patterns, and make predictions or decisions based on that data [IBM24b]. In essence, ML works through a three-step process:

1. **Decision Process**: ML algorithms analyze input data, which can be labeled or unlabeled, to make predictions or classifications about patterns in the data.

2. **Error Function:** an error function evaluates the accuracy of the model's predictions by comparing them to known examples or labels in the data.

3. **Model Optimization Process:** the algorithm adjusts its internal parameters or "weights" to minimize the error between its predictions and the actual outcomes in the training data. This iterative process continues until the model achieves a satisfactory level of accuracy [IBM24b].

1.1.2.2 Machine Learning Methods

1. **supervised machine learning :** Supervised learning, also referred to as supervised machine learning, relies on labeled datasets to train algorithms for accurate classification or prediction tasks. During training, the model iteratively adjusts its parameters based on the input data to minimize errors. This process, often accompanied by cross-validation, aims to prevent over-fitting or under-fitting of the model. Supervised learning plays a crucial role in addressing various real-world challenges, such as segregating spam emails from legitimate ones. Common techniques employed in supervised learning include neural networks, Naïve Bayes, linear regression, logistic regression, random forest, and support vector machine (SVM) [IBM24b].

2. **Unsupervised learning**: also termed unsupervised machine learning, employs machine learning algorithms to scrutinize and categorize unlabeled datasets into subsets known as clusters. These algorithms autonomously uncover latent patterns or data groupings without human guidance. This methodology's adeptness in identifying similarities and disparities in data renders it invaluable for exploratory data analysis, devising cross-selling strategies, segmenting customers, and performing tasks like image and pattern recognition. Additionally, unsupervised learning facilitates the reduction of model features through dimensionality reduction techniques, with principal component analysis (PCA) and singular value decomposition (SVD) being prominent methods. Neural networks, k-means clustering, and probabilistic clustering approaches are among the various algorithms utilized

in unsupervised machine learning [IBM24b] .

3. **Semi-supervised learning:** offers a happy medium between supervised and unsupervised learning. During training, it uses a smaller labeled data set to guide classification and feature extraction from a larger, unlabeled data set. Semi-supervised learning can solve the problem of not having enough labeled data for a supervised learning algorithm. It also helps if it's too costly to label enough data [IBM24b].

4. **Reinforcement machine learning:** is a machine learning model that is similar to supervised learning, but the algorithm isn't trained using sample data. This model learns as it goes by using trial and error. A sequence of successful outcomes will be reinforced to develop the best recommendation or policy for a given problem [IBM24b].

1.1.3 Deep Learning

1.1.3.1 Definition

The simple machine learning algorithms work well on a wide variety of important problems like classification , , regression etc . However,they have not succeeded in solving the central problems in AI such as recognizing speech or recognizing objects. The development of deep learning was motivated in part by the failure of traditional algorithms to generalize well on such Al tasks. the challenge of generalizing to new examples becomes exponentially more difficult when working with high-dimensional data, and how the mechanisms used to achieve generalization in traditional machine learning

are insufficient to learn complicated functions in high-dimensional spaces. Such spaces also often impose high computational costs. Deep learning was designed to overcome these and other obstacles [GBC16].

Deep learning is a subset of machine learning that uses multi-layered neural networks, called deep neural networks, to simulate the complex decision-making power of the human brain [HS24].

By strict definition, a deep neural network, or DNN, is a neural network with three or more layers. In practice, most DNNs have many more layers. DNNs are trained on large amounts of data to identify and classify phenomena, recognize patterns and relationships, evaluate possibilities, and make predictions and decisions. While a single-layer neural network can make useful, approximate predictions and decisions, the additional layers in a deep neural network help refine and optimize those outcomes for greater accuracy [HS24].

The above describes the simplest type of deep neural network in the simplest terms. However, deep learning algorithms are incredibly complex, and there are different types of neural networks to address specific problems or datasets. For example :

- Convolutional neural networks (CNNs): used primarily in computer vision and image classification applications, can detect features and patterns within an image, enabling tasks, like object detection or recognition. In 2015, a CNN bested a human in an object recognition challenge for the first time [HS24].
- **Recurrent neural networks (RNNs):** typically used in natural language and speech recognition applications as it leverages sequential or times series data [HS24].

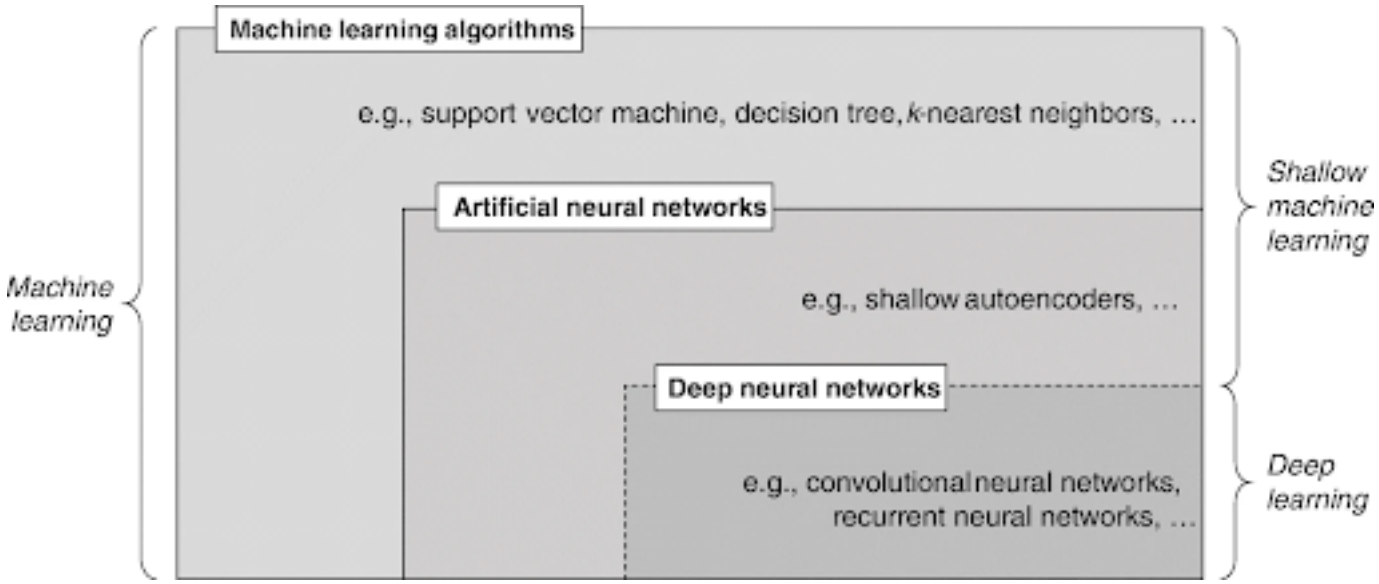


Figure 1.1: Venn diagram of machine learning concepts and classes [GBC16].

the figure 1.1 of the Venn diagram visually illustrates how these different concepts and classes overlap or are distinct from each other within the broader landscape of machine learning, providing a useful tool for understanding the relationships and distinctions between various approaches and techniques in the field.

1.1.4 Computer Vision

1.1.4.1 Definition

Computer vision, as a field of study, is primarily concerned with enabling computers to perceive visual data. At its core, the aim of computer vision problems is to analyze observed image data to gain insights into the surrounding world. The overarching objective is to decipher the contents of digital images, often achieved by developing techniques that mimic human visual capabilities. This entails extracting meaningful descriptions from images, which could range from identifying objects to generating textual descriptions or constructing three-dimensional models. Essentially, computer vision involves automating the process of extracting information from images, which can encompass tasks like object detection, recognition, determining camera positions, and organizing image content.

It's important to differentiate computer vision from image processing. While image processing involves modifying or enhancing existing images, focusing on aspects like brightness or color adjustments, it doesn't concern itself with interpreting the content of the image. However, computer vision systems may incorporate image processing techniques as a preliminary step, such as pre-processing raw images before analysis [Bro19].

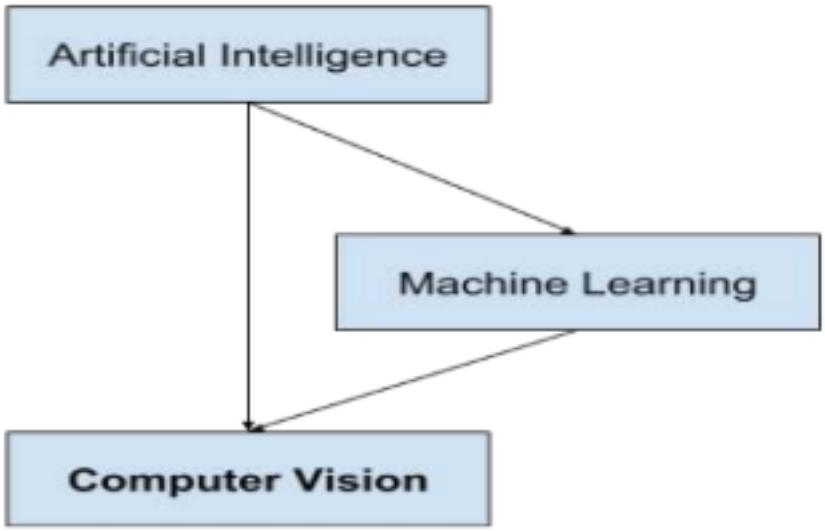


Figure 1.2: Overview of the Relationship of Artificial Intelligence and Computer Vision [Bro19].

Figure 1.2 illustrates the hierarchical relationship among Artificial Intelligence (AI), Machine Learning (ML), and specific application areas like Computer Vision. AI encompasses the entire field, providing the foundational goals and objectives. Machine learning is a critical component within AI, offering the methodologies and algorithms that enable systems to learn from data. Building upon machine learning, computer vision applies these techniques to visual data, allowing computers to gain insights from images and videos.

1.1.4.2 Tasks and Applications

As resources for developing computer vision applications become more accessible, it's crucial to address a fundamental question: What specific functions will these applications perform? Clarifying and defining precise computer vision tasks can focus and validate projects and applications, streamlining the initiation process.

Below are several examples of established computer vision tasks:

- Image classification involves analyzing an image and assigning it to a particular category (such as identifying a dog, an apple, or a person's face). Specifically, it accurately predicts that a given image belongs to a specific class. For instance, a social media platform might utilize image classification to automatically detect and segregate objectionable images uploaded by users [IBM24a].

- Object detection utilizes image classification to identify a specific class of objects and then locates and records their presence within an image or video. Applications include detecting defects on assembly lines or identifying machinery in need of maintenance [IBM24a].

- Object tracking involves monitoring or tracing an object once it has been detected. This task is often performed using sequential images or real-time video feeds. For example, autonomous vehicles must not only classify and detect objects like pedestrians, vehicles, and road infrastructure but also track their movements to avoid collisions and comply with traffic regulations [IBM24a].

- Content-based image retrieval leverages computer vision to browse, search, and retrieve images from extensive databases based on their content rather than metadata tags. This task can incorporate automatic image annotation to replace manual tagging, benefiting digital asset management systems and enhancing search and retrieval accuracy [IBM24a].

1.1.5 Pattern Recognition and Image Analysis

Pattern recognition and image analysis are closely related fields within the broader domain of computer vision and artificial intelligence [Arm]. While pattern recognition focuses on identifying regularities or patterns in data, image analysis specifically deals with processing and interpreting visual information contained in images [IBM24a] .

Pattern recognition techniques are applied to analyze various types of data, including images, signals, text, and more. In the context of image analysis, pattern recognition algorithms are used to detect and interpret patterns within images, such as shapes, textures, objects, or structures. These algorithms enable computers to understand and extract meaningful information from visual data [Arm]. Advanced topics in pattern recognition include:

- Statistical, structural, and syntactic pattern recognition : these are different approaches to classify data based on statistical analysis, structural relationships, or syntactic rules [Arm].

- Feature extraction and reduction : this involves identifying the most relevant data attributes (features) and reducing the dimensionality of the dataset to improve the performance of pattern recognition algorithms [Arm].

Image analysis, on the other hand, is a subset of computer vision that deals with processing and interpreting visual information contained in images. It encompasses various tasks such as image enhancement, feature extraction, image segmentation, and object detection, aiming to understand and analyze the visual content of images [Spr] [MDP].
Some of the current trends in image analysis are:

- Color and texture analysis: These techniques analyze the color and texture patterns within an image to identify objects or regions of interest [Arm].
- Image segmentation and compression: Segmentation divides an image into parts for easier analysis, while compression reduces the image size for storage and transmission without significantly affecting the quality [Arm].

Both fields are rapidly evolving with the advent of deep learning and neural networks, which have significantly improved the capabilities of both pattern recognition and image analysis systems. For instance, convolutional neural networks (CNNs) have become a staple in image classification tasks due to their ability to learn hierarchical representations of visual data.

1.2 Related Work and taxonomy

1.2.1 Motion Detection

1.2.1.1 Definition

In video surveillance, motion detection refers to capability of identifying changes in the position of objects within a specific area over time. Motion detection is usually a software-based monitoring algorithm,when it detects motions, signals the surveillance camera to begin capturing the event. Also called activity detection. An advanced motion detection surveillance system can analyze the type of motion to see if it warrants an alarm [MS16].

Human Motion:« *Human motion refers to physical position change in human body over time in daily life.* » [YI22].

1.2.1.2 Motion Detection Algorithms

1. **Background subtraction method**

   This algorithm involves creating a background model of the scene and subtracting it from the current frame to identify moving objects.Background subtraction is a foundational method for detecting moving objects in videos captured by static cameras. The essence of this approach involves comparing each frame of the video with a reference frame, often termed the "background copy" or "background replica." This reference frame ideally represents the scene without any moving objects and needs to be regularly updated to account for changes in lighting conditions and scene geometry.

   Traditionally, background subtraction involves detecting regions of motion by analyzing the differences between the current frame and the background reference frame. This method is highly

effective for motion detection and can provide valuable data, including information about detected objects [AY16].

However, more sophisticated models have expanded the concept of background subtraction beyond its literal interpretation. These advanced methods may incorporate additional features and techniques to enhance accuracy and robustness, such as adaptive background modeling, shadow removal, and probabilistic modeling [BKH+17].

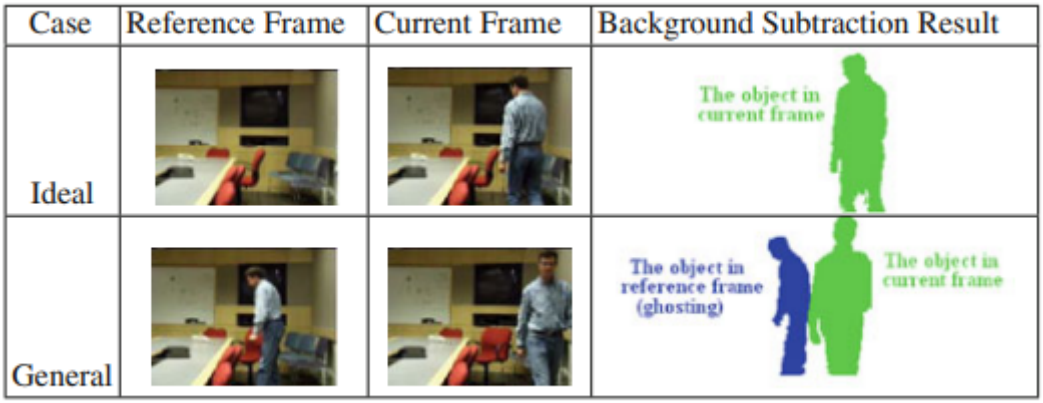


Figure 1.3: Background subtraction results [MMdP12].

The primary challenge in background subtraction is accurately initializing and continuously updating the background reference image. Effective methods for initialization and adaptation are crucial for ensuring accurate motion detection, especially in dynamic environments where lighting conditions and scene geometry may change frequently. Challenges include noise from poor-quality image sources, gradual variations in lighting conditions, small movements of non-static objects such as tree branches and bushes blowing in the wind, undeviating variations of objects in the scene such as cars that park or depart after a long period, sudden changes in light conditions such as sudden rain or the presence of a light switch, and movements of objects in the background that leave parts of it different from the background model. [RP13]. Many methods have been proposed based on this approach, such as background updating. Essentially, this time-differencing method proposes that a pixel is considered part of a moving object if its intensity has significantly changed between the current frame and the previous one. Mathematically, this is expressed as
$|I_t(x) - I_{t-1}(x)| < \tau$ where t is a predefined threshold [MMdP12].

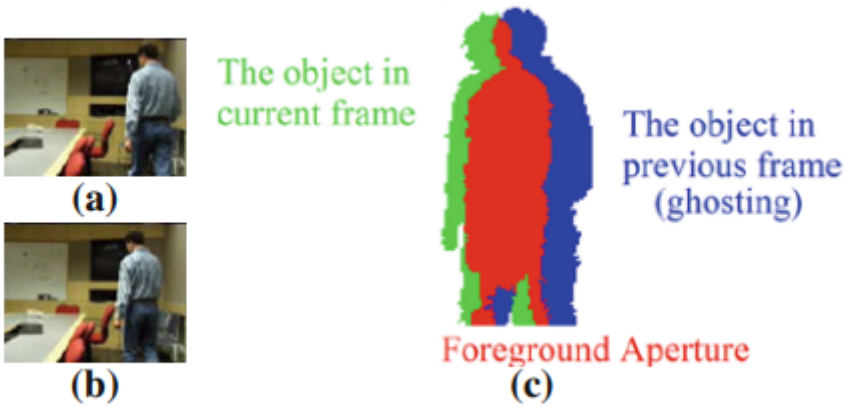


Figure 1.4: Drawbacks of adjacent frame difference approach [MMdP12].

2. **Optical Flow**

   The optical flow method detects moving objects or humans by analyzing their individual velocities. It serves as a crucial feature for both object detection and tracking, offering insights into the spatial arrangement of viewed objects and the rate of change in this arrangement. The optical flow describes the direction and time pixels in a time sequence of two consequent dimensional velocity vector, carrying direction, and the velocity of motion is assigned to each a given place of the picture [UMDS+19].

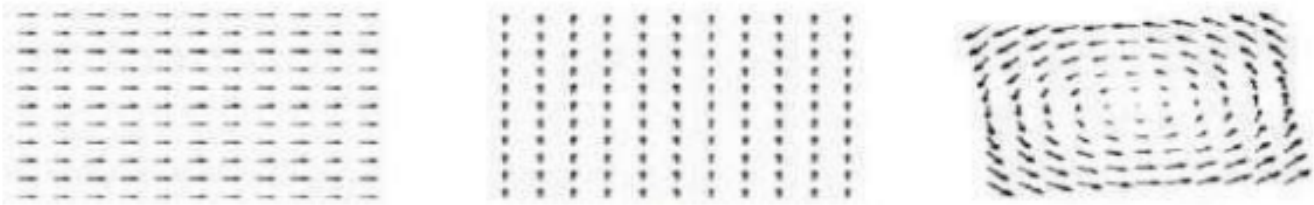


Figure 1.5: Example of optical flow [UMDS+19].

3. **Block Matching Algorithms**

   Pixel-based Block Matching Algorithms (BMA) are traditional algorithms used in video compression, motion estimation, and object tracking by finding correspondences between blocks of pixels in two sub-sequence frames. Block matching methods are effective yet computationally intensive [BRZ24].

   The process involves calculating a cost function at each possible location within a search window to find the best match for a macro-block in the reference frame. This helps in discovering temporal redundancy in the video sequence, thereby enhancing inter-frame video compression by referencing the contents of a macro-block to a known macro-block with minimal differences [PBJ+23]. In the block matching algorithm, a video frame (frame at time t) is divided into fixed-size macroblocks, and the next video frame (frame at time t + 1) is also divided into fixed-size macroblocks. Now, each

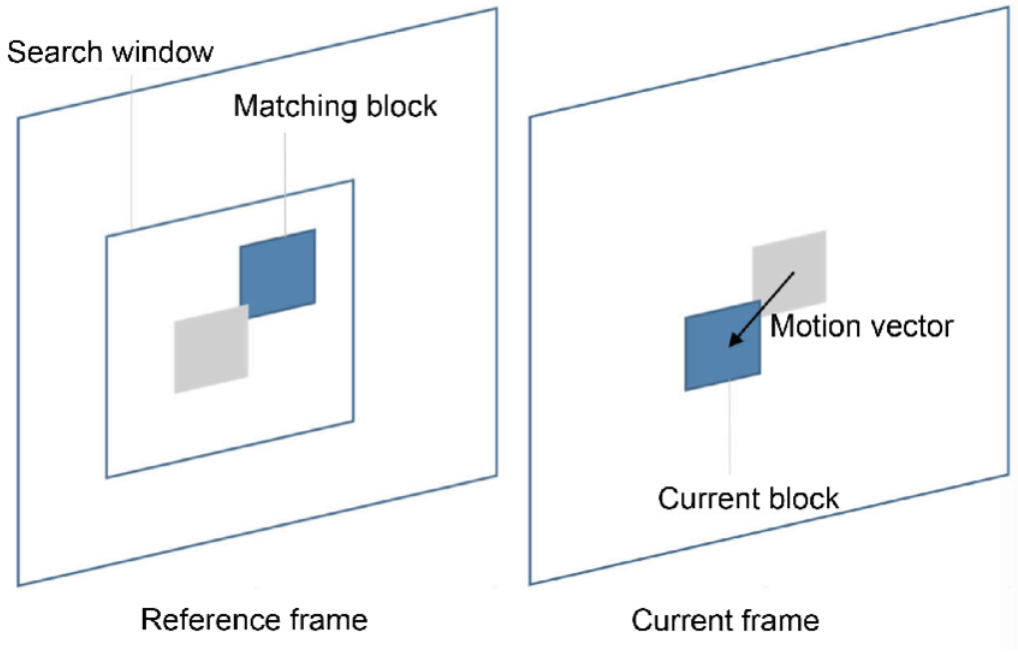


Figure 1.6: Block-matching motion estimation [YCS17a].

macroblock in the current frame is compared with the macroblocks in the previous frame. The video consists of moving objects, so the movement of objects from one frame to the next frame is observed by block matching algorithm.

1.2.1.3 Comparing Between Motion Detection Algorithms

After extensively studying the various motion detection techniques presented in related works, we conclude that background subtraction offers high detection accuracy. Despite it provides high speed and low computational complexity, making it ideal for real-time processing. On the other hand, techniques such as optical flow and block matching deliver high precision and more detailed results (e.g., motion direction). However, their significant computational complexity reduces their speed, making them less efficient for real-time detection, especially with limited resources. The table below summarizes our findings from analyzing the different related works.

Table 1.1: Comparative Table of motion detection algorithms.

algorithm	papers	precision	speed	computational complexity	Real-Time Efficiency
Background substraction	[AY16], [BKH+17], [RP13]	High	high	low	Ideal
Optical flow	[KMK16], [UMDS+19]	very high	low	high	Less Efficient
Block matching algorithme	[PBJ+23], [YCS17b]	very high	low	high	Less Efficient

1.2.2 Object Detection

1.2.2.1 Definition

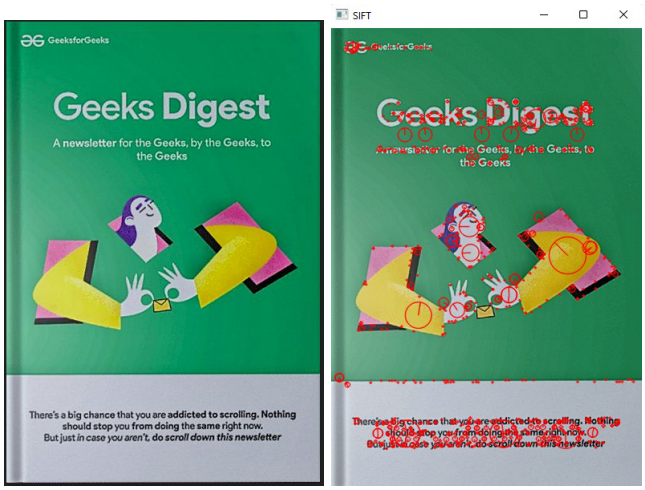
Object detection is the process of identifying and locating instances of specific objects within an image. Its objective is to accurately detect all occurrences of predefined object classes, such as people, cars, or

animals, amidst the potentially vast array of locations and scales within the image. This task requires robust algorithms capable of efficiently exploring the image space to recognize objects accurately, even in complex or cluttered scenes [AFG21].

1.2.2.2 Used Algorithms

1. **Traditional Methods**

   - **Definition**: Feature-based methods rely on handcrafted features extracted from images, such as edges, corners, or textures, to represent objects. These features are designed to capture distinctive characteristics of objects that can be used for detection [AFG21]. These methods typically involve extracting features from image patches or regions and then using these features for classification or matching against predefined templates.
   - **SIFT (Scale-Invariant Feature Transform)**: irrelevant to the scale and rotation of the image and the reference. This helps a lot when comparing real-world objects to an image, as it is independent of the angle and scale of the image. The key points of the images which need to be marked will be returned by this method [Geec].



(a) Input Image (b) Ouput image

Figure 1.7: Result of SIFT algorithm [Geec].

   - **Haar cascades:** are widely used in computer vision tasks, particularly for face detection, but they can be applied to detect other objects as well. They work by analyzing the intensity differences in various regions of an image using a set of rectangular features known as Haar-like features [Geeb].

Figure 1.8: Result of Output image using Haar cascade [Geeb].

- **Support Vector Machines (SVMs):** are pivotal in surveillance due to their robust classification, especially in complex, high-dimensional data. SVMs work by establishing a hyperplane to separate different classes, aided by support vectors that influence its position and a margin that maximizes the distance from this hyperplane to the nearest data points of each class. In human detection, SVMs excel at feature extraction using methods like Histograms of Oriented Gradients (HOG), which capture relevant image features. Trained on labeled data to discern humans from other objects, SVMs then classify new data, such as images or video frames, effectively identifying human presence amidst varying backgrounds or complexities [PP21].

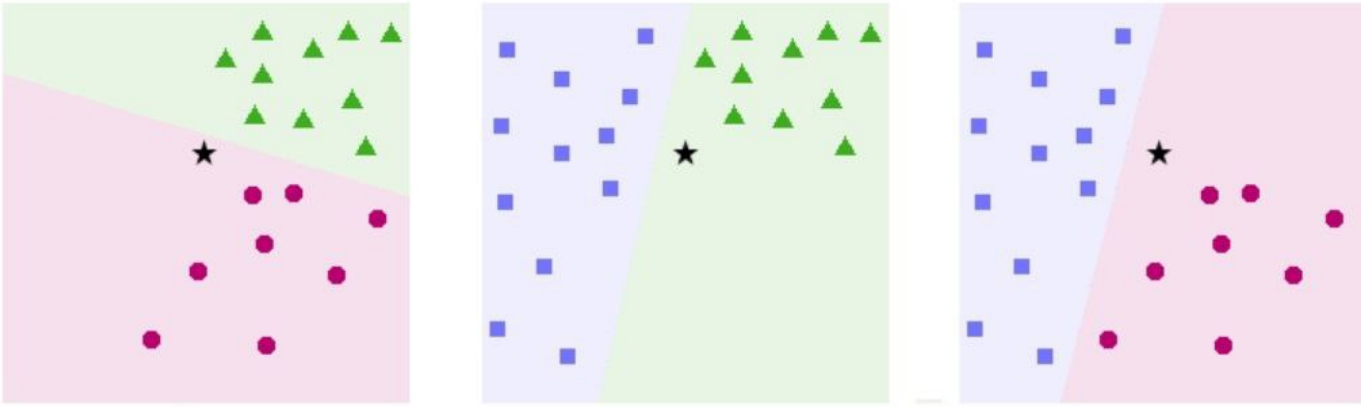


Figure 1.9: Example of use of the svm [Chr20].

2. **Neural Network Methods:** neural network methods leverage deep learning techniques, particularly convolutional neural networks (CNNs) that operates by employing convolutional layers that extract hierarchical features from input data through convolutional filters, capturing important patterns like edges, textures, and shapes. Subsequently, pooling layers downsample these feature maps, preserving essential information while reducing spatial dimensions for

computational efficiency. Fully connected layers then perform classification based on the extracted features, enabling CNNs to accurately detect humans in images or video frames amidst complex backgrounds. This process involves training the network using supervised learning with labeled data, where the CNN learns to recognize human presence through iterative adjustments of its internal parameters, ultimately leading to proficient classification capabilities in human detection tasks within surveillance systemtasks [AFG21].

These methods are effective for learning complex features directly from raw data and are particularly suited for large-scale datasets.

3. **Faster R-CNN:** short for "Faster Region-Convolutional Neural Network is an object detection architecture within the R-CNN family. The primary goal of the Faster R-CNN network is to develop a unified architecture that not only detects objects within an image but also precisely locates these objects [RHGS16]. It combines the benefits of deep learning, convolutional neural networks (CNNs), and region proposal networks (RPNs) into a cohesive network, significantly improving the speed and accuracy of the model.
   Faster R-CNN consists of two main components:

   (a) **Region Proposal Network (RPN):** this module is responsible for generating region proposals. It applies the concept of attention in neural networks, guiding the Fast R-CNN detection module to where to look for objects in the image [Geee].

   (b) **Fast R-CNN:** this module detects objects in the proposed regions generated by the RPN. The convolutional computations are shared across the RPN and the Fast R-CNN, reducing computational time and improving efficiency [Geee].

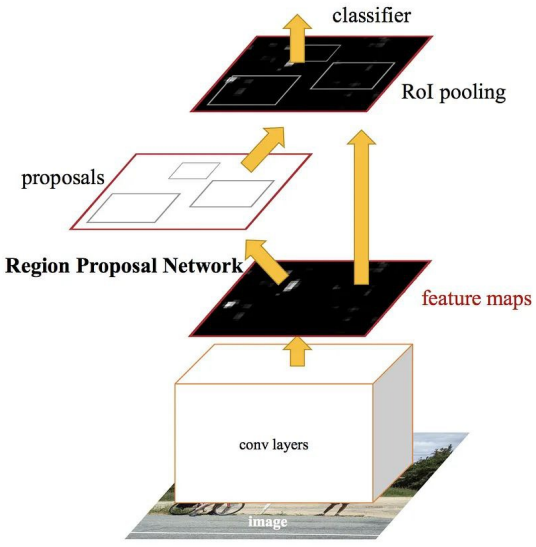


Figure 1.10: Fast RCNN Architecture [geea].

4. **YOLO (You Only Look Once) :** marked a significant advancement in object detection due to its innovative approach of performing detection and classification
   simultaneously in a single pass through a convolutional neural network (CNN). Unlike
   traditional methods that propose regions of interest in a separate initial step, YOLO divides the input image into a grid, with each cell responsible for predicting bounding boxes and class probabilities directly from features extracted by the CNN. This eliminates the need for multipleimage scans, combining real-time speed with high accuracy. YOLO's architecture includes several convolutional and pooling layers that capture useful patterns at different spatial scales, significantly reducing computational costs. It also uses predefined "anchors," or bounding boxes of various sizes and shapes, to improve detection across different object types and scales, enhancing its precision [Geed].

Figure 1.11: An object detection exemple using yolo [Dat23].

The figure 1.11 shows an input image divided into a grid , the second image displays the detected objects with bounding boxes and the final image highliths the identified object with refined bounding box .

1.2.3 Object Tracking

Object tracking involves the process of assigning a unique identifier to each initial detection of an object and then monitoring the movement of these objects across successive frames in a video, while maintaining consistency in the assigned IDs.

The primary objective of object tracking is to determine the state of a target or object across a sequence of images or frames. This state can be characterized by various attributes, such as its shape, appearance, location, or speed. However, object tracking poses significant challenges, as it requires the development

of algorithms capable of addressing various complexities. These challenges include managing variations in lighting conditions, accounting for changes in object appearance due to factors like camera rotation, handling occlusions when objects are obscured by other objects or elements in the scene, and addressing issues related to image quality [HAM21].

1. **Tracking Using Matching**

   Tracking using matching involves comparing a model representation of an object from frame T-1 with potential candidates in frame T, typically based on similarity measurements. The process involves calculating a cost function at each possible location within a search window to find the best match for a macro-block in the reference frame [BRZ24].

2. **Tracking by Detection**

   Tracking by detection involves building a model to distinguish objects from the background. Once an initial detection is made, it is associated with previous detections. Current research is increasingly focused on using neural networks to extract detections.

   Some tracking systems combine the speed of feature tracking with the accuracy of neural networks, creating a hybrid approach. In this method, neural networks are used for detections every N frames, while feature tracking is used for intermediate frames.

3. **Tracking as Regression**

   The network receives two images (previous and current frame) and directly predicts the object's location in the current frame, modeling changes in scale and aspect. However, it's limited to processing a single target and requires data augmentation techniques to learn various transformations of the targets [HAM21].

Conclusion

This introductory chapter lays the foundation for a comprehensive exploration of activity detection methods and the state-of-the-art techniques in surveillance technology. The delineation of machine learning (ML) methods, including supervised, unsupervised, and semi-supervised learning, provides a framework for understanding the intricacies of training AI models. Furthermore, the overview of computer vision algorithms offers insights into the practical applications of AI in surveillance technology. We also reviewed related work, studying various intelligent surveillance techniques, their strengths and limitations, and analyzing and comparing them to determine the most suitable approach for our situation. With this solid theoretical foundation established, the subsequent chapters delve deeper into the techniques, components, and architecture of a video surveillance system.

Chapter 2

Estin's Surveillance System

Introduction

Video surveillance has experienced significant expansion both technologically and economically in recent years. It has become one of the essential components of government security policies, ensuring public safety amidst the challenges posed by escalating criminal activities. This evolution addresses every citizen's fundamental need for security, offering reassurance and protection in the face of increasing delinquency and crime rates.

In this chapter, the intricacies of video surveillance systems are delved into, examining their diverse components, structural frameworks, various types, and expansive domains of application. We will start by presenting a comprehensive overview of video surveillance systems in general. Following this, we will study the specific video surveillance system implemented at ESTIN and explore its different characteristics.

2.1 Definition of Video Surveillance System

Video surveillance constitutes a sophisticated system incorporating strategically positioned cameras within a defined area to ensure continuous monitoring. These cameras are linked to a computer system capable of processing and analyzing incoming data. Originating in Germany in 1942 with Siemens AG's development for rocket observations, video surveillance systems have undergone significant evolution. Modern advancements have led to automated data analysis and integration, thereby reducing the necessity for extensive human involvement [Mok12].

Utilizing a network of cameras, monitors, and video interfaces, video surveillance involves observing scenes and scrutinizing behaviors indicative of impropriety or potential threats. Integration with autonomous artificial intelligence enhances the system's capabilities, elevating its efficiency and effectiveness.

Deployed in diverse settings, both indoor and outdoor, spanning structures and properties, these systems

operate continuously. They can be programmed to record upon motion detection or during specified periods, offering comprehensive monitoring and surveillance functionalities.

2.2 Video Surveillance System Components

2.2.1 Surveillance Cameras

Surveillance cameras are the eyes of the system, capturing video in real-time. They come in various types, including analog and digital (IP cameras), each suitable for different applications. Features like night vision, motion detection, and thermal imaging are common, enhancing the camera's ability to record in diverse conditions. Cameras can also vary in housing configuration to suit different installation environments.

Here are some of the IP camera models available on the market, each designed to meet specific surveillance needs [BHKNA16]:

1. **Fixed IP Camera:** unlike motorized cameras, a fixed IP camera monitors a specific location. Its main advantage is its effectiveness in deterring ill-intentioned individuals who quickly realize they are being filmed when they see the camera's lens directed toward them. This makes it an excellent choice for protecting residences or businesses.

2. **Motorized Dome IP Camera:** this type of camera can be installed anywhere, such as on walls or ceilings. A key feature is that it can be remotely controlled via a controller. It can zoom in on objects or people and perform 360-degree rotations, which is useful for sweeping the surroundings and obtaining a comprehensive view. Additionally, it allows for the scheduling of surveillance rounds at convenient times.

3. **Box IP Camera:** shaped like a box, this camera includes various types of lenses, making it versatile for different video surveillance needs. PTZ (Pan-Tilt-Zoom) and Dome PTZ IP Camera The PTZ camera can make horizontal movements and zooms, making it ideal for tracking a suspicious person and observing their actions in real-time. The dome PTZ camera offers the ability to monitor areas by moving through a 360-degree angle.

4. **Vandal-Proof IP Camera:** as the name suggests, this camera is designed to withstand external assaults, such as vandalism and tampering attempts. It is equipped with special surfaces that are resistant to breaks and shocks, making it the ideal surveillance camera for protecting valuable locations and enhancing security.

5. **Spy IP Camera:** due to its small size, the spy camera can easily be hidden in motion detectors and other small objects. It can also come in various forms, such as a pen, keychain, or watch. Its main

advantage is that it can better catch intruders attempting to infiltrate our home or business. Since it is easily concealed, it is not readily noticeable at first glance.

6. **Infrared IP Camera:** this night vision surveillance device can record in complete darkness with precision up to 30 meters. The infrared camera is practical for protecting against thieves and burglaries while you sleep peacefully.

2.2.2 Recorder and Storage

The video recorder processes and stores the videos captured by the cameras. There are two main types of recorders:

1. **Digital Video Recorders (DVRs):** these are used with analog cameras and are part of traditional surveillance systems.

2. **Network Video Recorders (NVRs):** these work with IP cameras and are more suited for modern, digital setups. NVRs often provide all-in-one solutions that simplify the surveillance system, especially in entry-level applications.

2.2.3 Transmission and Connectivity

The method of transmitting the video signal from the cameras to the monitoring site is crucial. Options include hard-wiring with cables such as CAT5e or CAT6 Ethernet cables, or using wireless technologies where cameras connect to a network via routers. The choice between wired and wireless systems depends on the specific installation site and the required flexibility [BHKNA16].

2.2.4 Storage and Compression

Storage is a critical component, with options ranging from local hard drives to cloud storage. Video compression technologies like MJPEG, H.264, or H.265 help in managing storage space and bandwidth by reducing the size of the video files without significant loss of quality (the most commonly used is H.264) [BHKNA16].

2.2.5 Supporting Technologies

Additional components like cables, routers, and Wi-Fi extenders might be necessary, depending on the system's configuration. For instance, wireless systems need a robust network setup to maintain a reliable connection across the surveillance area [BHKNA16].

2.2.6 Remote Viewing and Access

Modern IP video surveillance systems typically offer remote viewing capabilities, allowing users to monitor their property from anywhere via a mobile app. This feature requires internet access to connect to the NVR remotely [BHKNA16].

2.3 Video Surveillance System Architectures

The architecture of video surveillance systems is based on five fundamental decisions: encoding, storage, analytics, management, and monitoring. These decisions determine where the video is encoded, stored, analyzed, managed, and monitored .

1. **Encoding:** the first decision in a video surveillance system is where the video will be encoded. This can occur directly in the cameras, in separate encoders, or at the recorders. The choice affects the system's flexibility, cost, and complexity.
2. **Storage:** video data can be stored in several locations
   (a) **Local Storage:** directly on the camera or on a local recorder like DVR or NVR.
   (b) **Network Storage**: on network-attached storage (NAS) or storage area networks (SAN).
   (c) **Cloud Storage:** offsite storage that offers scalability and remote accessibility.
3. **Analytics:** video analytics can be processed at different points in the system:
   (a) **Edge Analytics:** directly on the camera.
   (b) **Centralized Analytics:** on dedicated servers or cloud-based services. This placement impacts the responsiveness and scalability of the analytic solutions.
4. **Management:** the management of video data and system operations can be handled through various platforms:
   (a) **Video Management Software (VMS):** for centralized management of video feeds.
   (b) **Cloud-Based Management:** for remote and scalable management options.
   (c) **Monitoring:** the final output of the video surveillance system is the monitoring interface, which can be stationed locally or accessed remotely, often via cloud services.

However, the complexity of interactions between system components, particularly across different equipment and models, presents challenges. Various methods have been proposed to address this issue, focusing on connecting cameras and gathering information through different architectural approaches.

Different architectural approaches categorize connections between main stations and devices into analog, digital, and network systems [EAAM21].

2.3.1 Analog Surveillance System

Historically, surveillance systems relied on analog technology for over two decades. Analog signal processing formed the foundational model for image transmission, exchange, and recording. This involved utilizing short-distance coaxial cables and long-distance transceiver optical fibers. While analog systems offered advantages such as image restoration, they also presented limitations including restricted transmission distance, complex engineering cabling, and inflexible application. The diagram below illustrates the components of an analog-based system, encompassing acquisition, storage, visualization, switching, and processing [EAAM21].

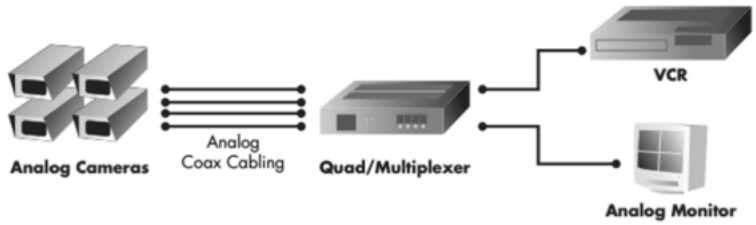


Figure 2.1: Example of an analog video surveillance system [EAAM21].

2.3.2 Digital Video Surveillance

The transition from analog to digital surveillance systems began with the introduction of digital video recorders (DVRs), stemming from analog technology. During this phase, digital image files are transmitted via computer network systems. Cameras connect to a video server through an IP network, enabling transmission over existing computer LANs or even the Internet. This setup allows for camera control and scene zooming via a computer terminal network, effectively transforming traditional systems into distributed ones.

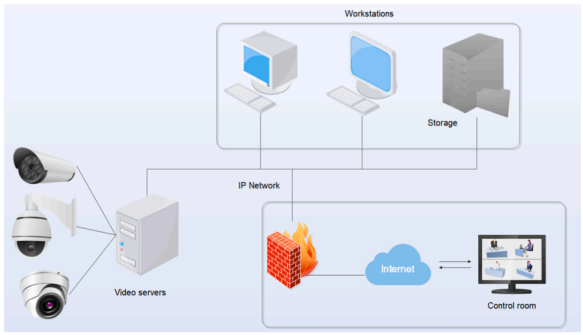


Figure 2.2: Example of a digital video surveillance system [EAAM21].

2.3.3 Network Surveillance System

The network surveillance system is founded on digital signal processing and incorporates network cameras or IP cameras, which are digital video cameras. This system allows for the direct connection of as many IP cameras as necessary to the IP network. Networking techniques are employed to achieve signal transmission, exchange, control, and video storage. Additionally, the system can perform centralized control and management of all devices, including cameras and sensors.

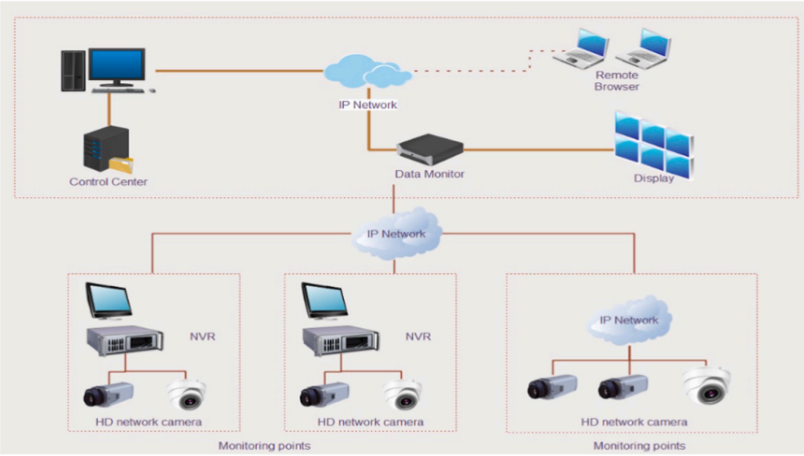


Figure 2.3: Example of a network video surveillance system [EAAM21].

The figure 2.3 illustrates the architecture of a network video surveillance system, detailing all devices and their connection methods [EAAM21].

2.4 Internship Location Presentation

For this project, we undertook our internship at ESTIN (École Supérieure en Sciences et Technologies de l'Informatique). This institution provided an ideal environment for conducting our research due to its advanced infrastructure and supportive academic community.The primary focus of our internship was to optimize the existing surveillance system currently implemented at ESTIN. This project aimed to Evaluate the current surveillance system in place at ESTIN, Identify areas of improvement and potential vulnerabilities , develop and implement optimization strategies using modern technologies of artificial intelligence.

2.4.1 General Presentation

ESTIN is a prestigious institution located in Amizour,17km from the city of Béjaïa, Algeria. Established in 2019, ESTIN aims to address the growing demand for qualified professionals in the IT sector.

The school follows a rigorous educational system similar to other elite institutions, including preparatory classes and competitive exams for entry into higher classes.

The mission of ESTIN is to provide top-tier education and training in computer science and digital technologies, fostering innovation and excellence. The vision is to become a leading center of excellence in higher education, scientific research, and technological development in Algeria and beyond.

ESTIN follows a structured educational system that includes,preparatory Classes, these classes prepare students for the competitive exams required for entry into the higher classes. ESTIN is home to the LITAN (Laboratory of Information Technology and Artificial Intelligence), which focuses on cutting-edge research in various fields of computer science and digital technologies.

2.4.2 ESTIN Organization Chart

The figure 2.4 illustrates the hierarchical organization of ESTIN, headed by the School Director, who is accountable to the Board of Directors and the Scientific Council. It depicts several deputy directors overseeing specific domains such as economic affairs, infrastructure, and external relations, along with directors managing human resources, studies, internships, and the library. Additionally, the Head of Department is responsible for educational services, and the Secretary General supervises various administrative functions, emphasizing both technical and external relations services.

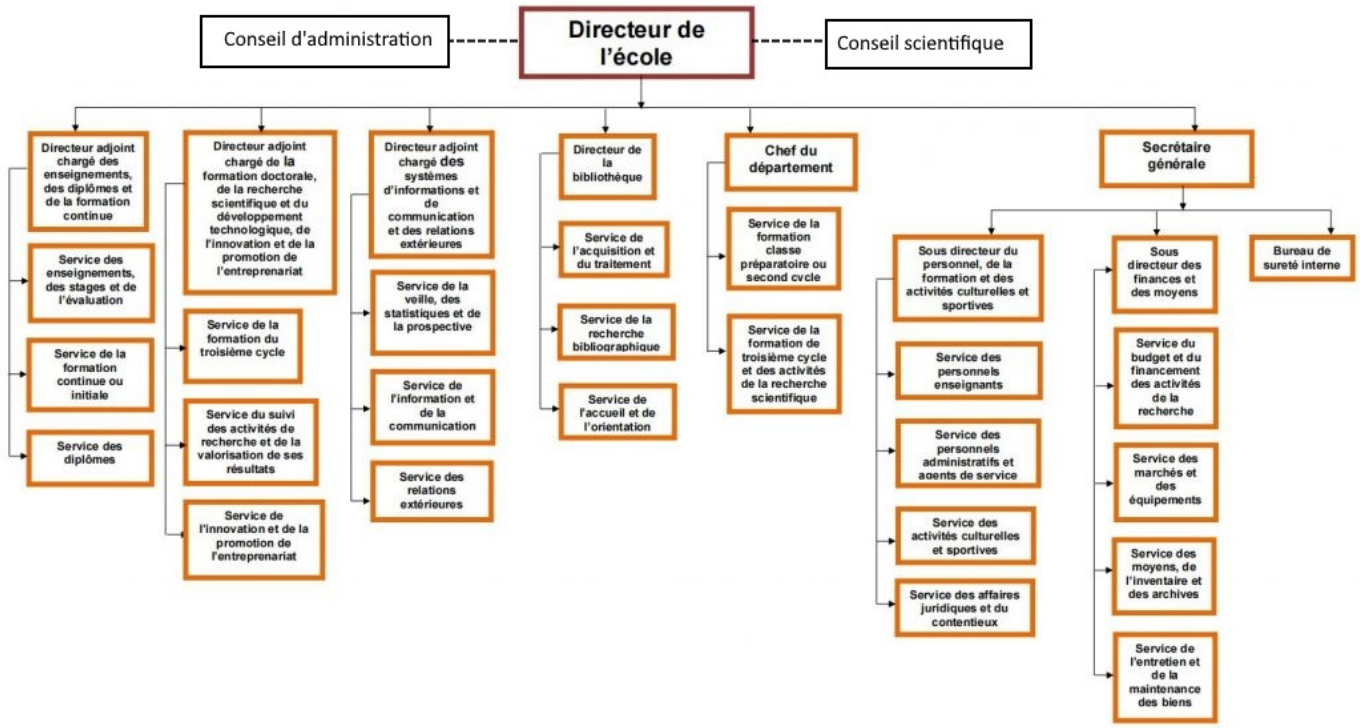


Figure 2.4: ESTIN organization chart [est].

2.4.3 Campus Layout and Surveillance Needs

The campus spans a substantial area with a buildable surface of 50,000 square meters. This extensive space is utilized efficiently with over five significant buildings dedicated to various functions. The campus consists of the following key buildings:

- **Administrative Building:** houses the offices of the administrative staff and management.
- **Teaching Buildings** (DW and PW): dedicated to theoretical and practical classes, accommodating numerous classrooms and lecture halls.
- **Laboratory Building:** equipped with state-of-the-art facilities for advanced research and experiments.
- **Library:** a resource-rich environment providing access to a vast collection of books, journals, and digital media.
- **Amphitheaters:** multiple amphitheaters for large gatherings, lectures, and presentations.

These buildings, along with numerous other facilities, create a complex and dynamic environment that requires a robust surveillance system to ensure safety and security. The strategic placement of surveillance cameras across these areas is essential for monitoring activities, preventing unauthorized access, vandalism, and other security threats.

The campus is equipped with a comprehensive surveillance system featuring high-resolution cameras strategically placed to cover critical areas such as entrances, hallways, common areas, and parking lots. This extensive camera network generates a vast amount of visual data daily, crucial for real-time monitoring and post-event analysis. However, this also presents significant challenges in terms of storage management and data processing efficiency. The campus hosts approximately 1,017 students, dozens of teachers, security agents, housekeepers, and various other workers. The high foot traffic and complex layout necessitate an effective surveillance solution to ensure safety and security, motivating the administration to implement this advanced system.

ESTIN has opted for the Hikvision Surveillance System Solution, a comprehensive and technologically advanced system designed to enhance security and surveillance capabilities in various environments, including educational institutions like schools. As of 2019, Hikvision held a commanding 43% share of the global market, underscoring its strong presence in the industry. Hikvision is renowned for its continuous technological advancements, ensuring it remains at the forefront of the surveillance industry and meets the evolving needs of its customers [Yol20].

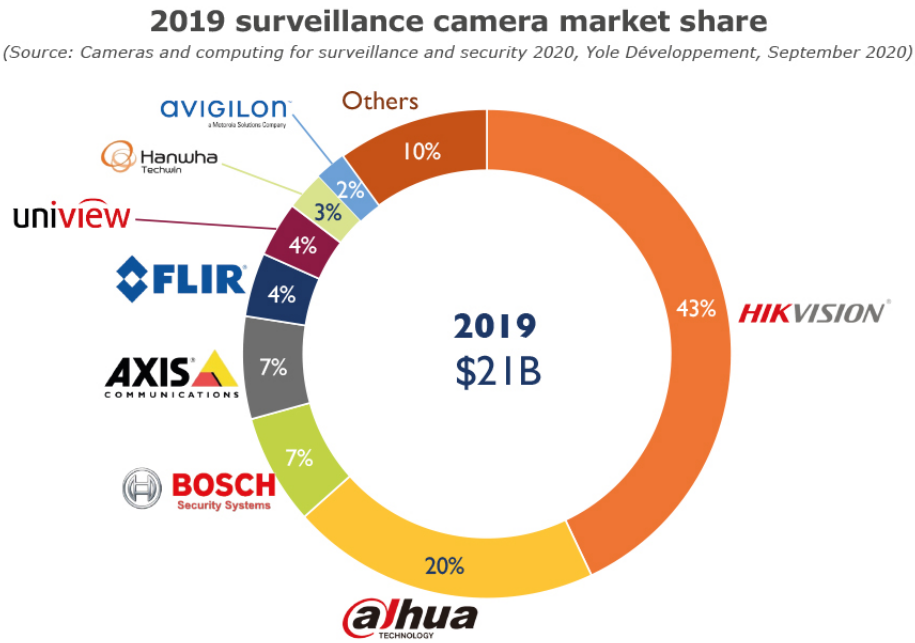


Figure 2.5: 2019 surveillance camera market share [Yol20].

2.5 Key Components and Features of Hikvision Surveillance System

2.5.1 Network Cameras

ESTIN uses three types of network cameras(Fixed IP Camera, Motorized Dome IP Camera and Box IP Camera) offered by Hikvision, the adoption of IP cameras signifies a shift toward advanced digital surveillance, allowing for remote monitoring, seamless integration with other security systems, these cameras are equipped with high-resolution sensors ensuring superior image quality and comprehensive coverage of the school premises.

2.5.2 Network Video Recorder

Is a device that records and stores video footage from surveillance cameras. It's like a digital VCR but designed for security cameras. It processes and stores video data directly from these network cameras without the need for analog-to-digital conversion. NVRs often have more advanced features and capabilities, including support for higher resolutions and additional functionalities.

2.5.3 Hikvision Management Software

Hikvision Management Software provides users with a user-friendly interface and powerful tools to efficiently manage and control their Hikvision surveillance system for effective security monitoring and management.s. Some key features and functions of the Hikvision Management Software include [Hik].

1. **Live Viewing:** users can view live video feeds from multiple cameras simultaneously, allowing for real-time monitoring of the surveillance system.

2. **Playback:** the software allows users to playback recorded video footage, search for specific events or incidents, and export video clips for further analysis or sharing.

3. **Remote Access:** the software supports remote access, allowing users to view live or recorded footage from their surveillance system via a computer or mobile device over the internet.

4. **Configuration:** users can configure and customize camera settings, recording schedules, motion detection settings, and other parameters to optimize the surveillance system for specific needs.For example, activate or deactivate motion detection.

5. **Alarm Notifications:** the software can send real-time alerts and notifications for events such as motion detection, camera tampering, or system errors.

6. **User Management:** the software includes user management features to control access permissions and user roles within the surveillance system.

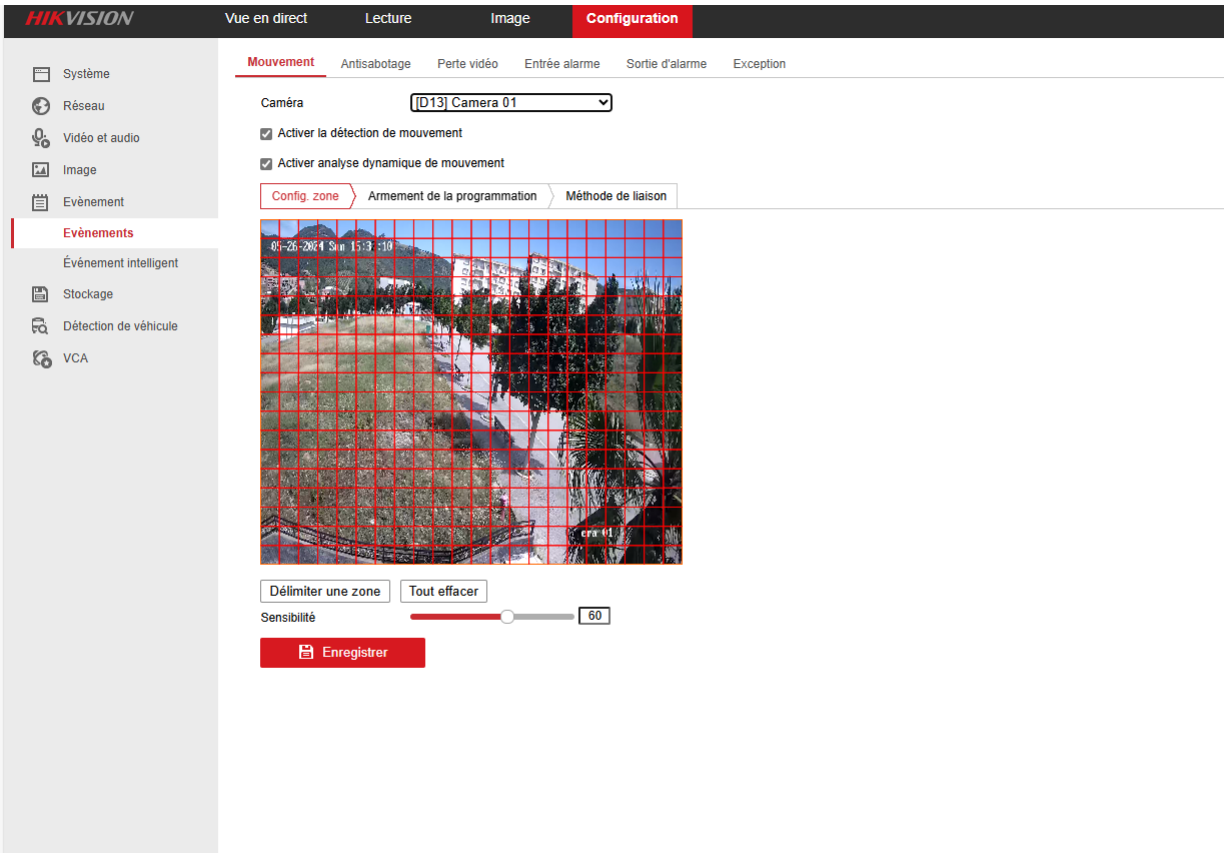


Figure 2.6: Interface of hikvision management software.

2.5.4 Architecutre

1. **IP Camera Connection:** IP cameras are connected directly to the NVR using a network (Ethernet) cable.
2. **Digital Recording:** the NVR processes and directly records the digital video signals from IP cameras.
3. **Storage:** like DVRs, NVRs have built-in hard drives for storing recorded video footage.
4. **Remote Access:** NVRs often support remote access, allowing users to view live or recorded footage from their cameras over a network, such as the internet.

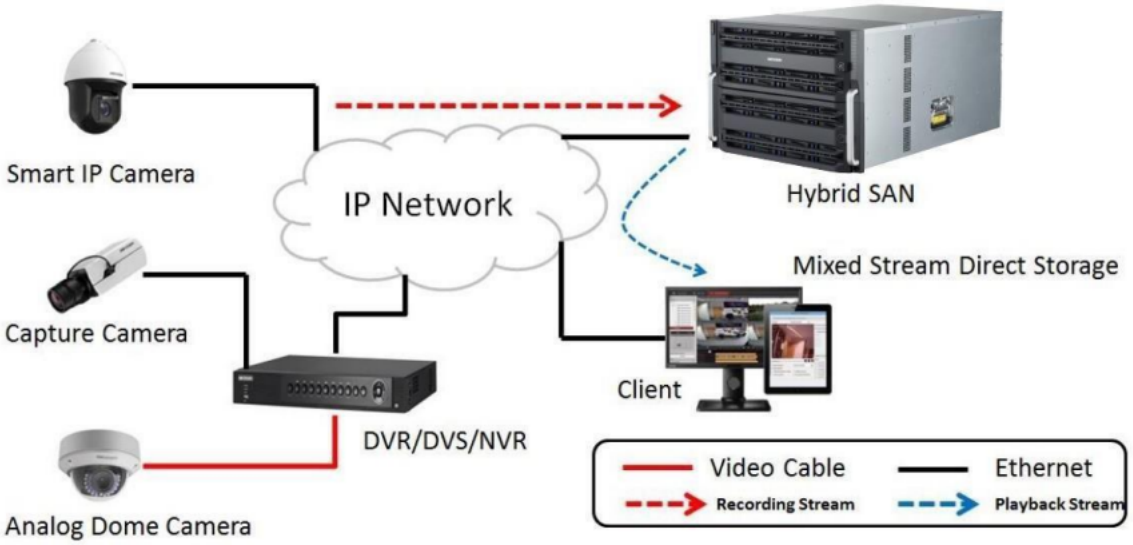


Figure 2.7: Hikvision surveillance system architecture [Hik].

Conclusion

In this chapter, various video surveillance system architectures and their critical components, including storage, compression, supporting technologies, remote viewing, and access, have been explored. The specific surveillance system implemented at ESTIN has been analyzed, examining its architecture, key components, and functionalities. The ESTIN surveillance system, based on the Hikvision solution, includes advanced features such as high-resolution network cameras, network video recorders (NVRs), and comprehensive management software. This system supports functionalities like motion detection recording, live viewing, playback, remote access, and alarm notifications.

However, one notable limitation of the current system is its inability to differentiate between significant and insignificant motion, such as the movement of trees, leading to unnecessary recordings. This inefficiency impacts storage space, especially given the large number of cameras employed at ESTIN and the sensitivity of the recording equipment. In the next chapter, a solution is proposed to address this problem by developing a method to record only important scenes, thereby optimizing storage space and improving the overall efficiency of the surveillance system.

Chapter 3

Proposed Solution : Conception, Implementation and Validation

Introduction

The primary objective of this chapter is to present the conception, implementation and validation of our intelligent video recording optimization system for surveillance systems. Our system aims to optimize video recording by incorporating activity detection techniques. By detecting relevant activities, such as object movements and motion, the system intelligently triggers video recording, thereby reducing storage space and facilitating efficient video analysis.

In this chapter, the system design is discussed, including the user interface, key components, and algorithms. The implementation details are elaborated upon, covering the development environment, coding, and integration of various modules. Furthermore, the challenges encountered during the implementation phase are examined, and the solutions adopted to overcome them are described. Finally, the validation methodology, experimental results, and a comprehensive discussion on the performance of the system compared to existing surveillance systems are presented.

3.1 Conception

3.1.1 System Design

3.1.1.1 Architecture Overview

The architecture of Our intelligent video recording optimization solution is primarily divided into several distinct modules, each responsible for a set of tasks. This modular design ensures the system's robustness, efficiency, and scalability. The system is implemented in Python, utilizing several specialized libraries and frameworks such as OpenCV, Streamlit, and Roboflow API.The main components of our system are:

1. **User Interface:** the user interface is designed using the Streamlit framework, a fast, user-friendly tool for developing data applications. The UI allows for real-time processing and display of video feeds, as well as user interaction through functional buttons like "Start Webcam" and "Stop Recording".

2. **Video Capture**: the video capture module uses OpenCV's Video Capture functionality to capture real-time video from the IP camera . The frames captured from this module are then forwarded for motion detection and object detection processes.

3. **Motion Detection**: the motion detection module leverages OpenCV's absdiff and threshold functions to detect motion in the video frames. The system considers a frame to have significant motion if the absolute difference of the grayscale values between the current frame and the previous one exceeds a certain threshold.

4. **Object Detection**: the object detection process in the system utilizes a fine-tuned YOLOv9 model deployed in the Roboflow API. The trained model is loaded using the get-model function, which enables the detection of objects in the frames where motion was detected.

5. **Video Recording**: when objects are detected in the frames, the system initiates the video recording module. The video recording process uses OpenCV's VideoWriter functionality. The recorded videos are systematically saved in a pre-designated "recordings" directory.

6. **Alerts and Notifications**: throughout the process, the system keeps the user informed using Streamlit's success function. This function is used to display important alerts and notifications to the user, such as the start or end of a recording.

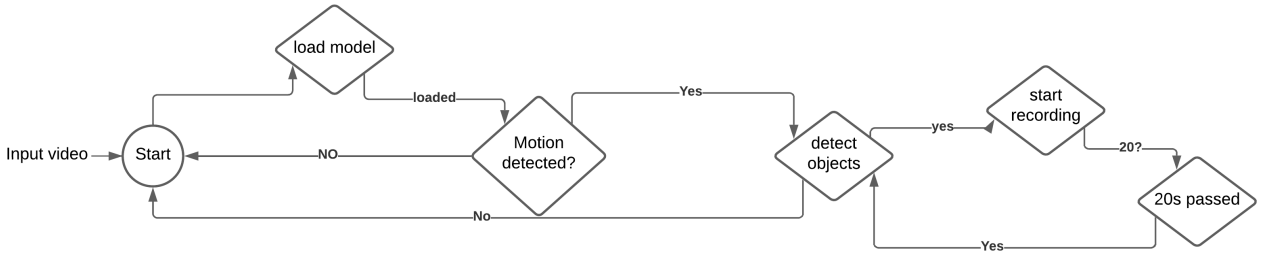


Figure 3.1: Flowchart explaining the global architecture of the proposed system.

The figure 3.1 illustrates the system's architecture, Detected motion triggers the object detection module using a YOLOv9 model via Roboflow API. The system includes a 20-second interval before re-evaluating scenes to avoid redundant detections, balancing computational load and capturing meaningful activity. Upon object detection, video recording is initiated using OpenCV's VideoWriter, and recordings are saved systematically.

3.1.2 Data Preparation

In this subsection, the process of preparing the data for training the model is outlined. The data preparation phase is crucial for ensuring the quality and effectiveness of the trained model. The following steps are involved in the data preparation process:

3.1.2.1 Data Collection

The data used for training the model was collected from various sources, including public datasets and online repositories [JK13] [Ver22]. The dataset consists of a combination of images of humans and cars. Here is an overview of the data collection process:

- **Human Detection Dataset:** a dataset containing 300 images specifically focused on human detection was obtained from Kaggle. The dataset consists of CCTV (closed-circuit television) footage of humans.
- **Car Detection Dataset:** another dataset containing 100 images of cars was sourced from the GitHub repository nguyentruonglau/cars-dataset [Lau], the dataset provides images of cars along with XML annotations.
- **Additional Images:** to augment the dataset and increase its diversity, additional images were collected from various websites and Google Images. These images were selected to cover a wide range of scenarios and backgrounds relevant to the problem domain.

3.1.2.2 Data Labeling

To train a supervised model, it is necessary to label the collected data. This involves manually annotating the objects of interest in the frames with their corresponding labels. In our case, the dataset was uploaded to the Roboflow platform for the labeling process. The objects were manually annotated into two different classes: person and car. This manual annotation process ensures that each object is accurately identified and labeled, providing the necessary labeled data for training a supervised machine learning model.

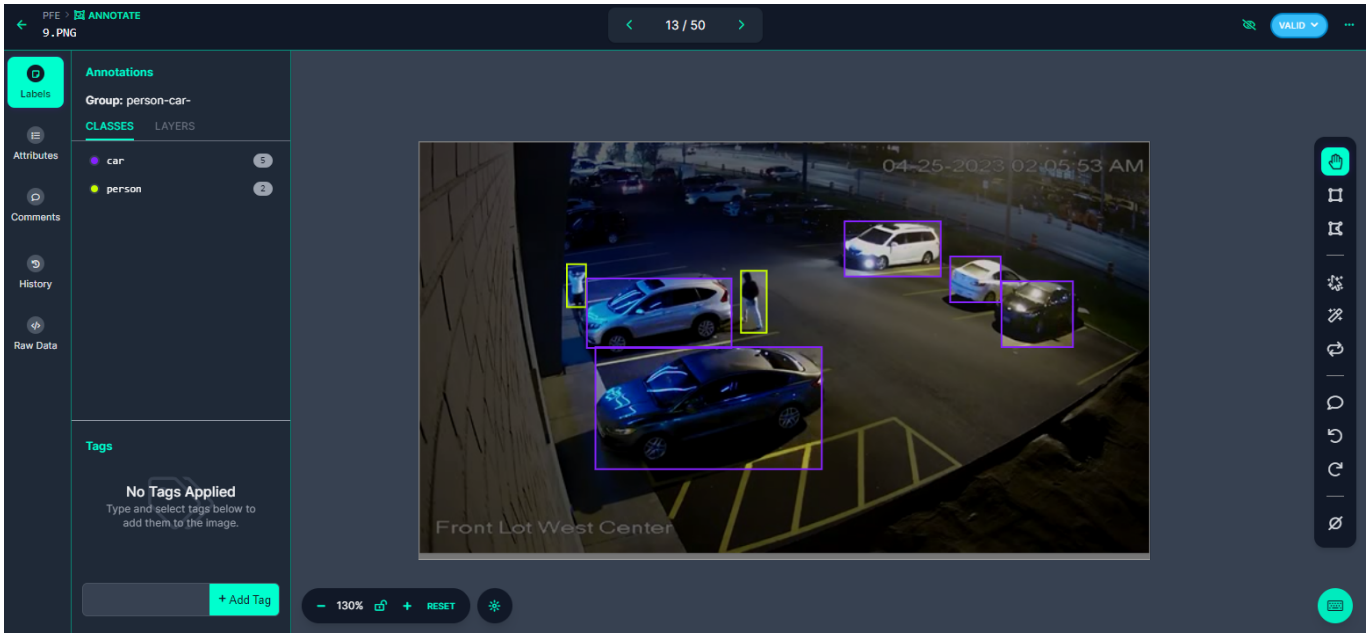


Figure 3.2: Annotation Process using roboflow annotation tool.

3.1.2.3 Data Preprocessing

After the data is collected, it undergoes a preprocessing stage to make it suitable for model training. This involves a series of steps, including:

- **Auto-Orient Preprocessing:** applying automatic orientation correction to the data to ensure consistency in the object's alignment.
- **resize Preprocessing:** resizing the data to a specific dimension, such as stretching it to 646x640 pixels, to achieve uniformity and compatibility with the model's requirements.
- **Auto-Adjust Contrast Preprocessing:** enhancing the contrast of the data using a technique called contrast stretching. This helps to improve the visibility of important features and details in the images.

3.1.2.4 Data Augmentation

In order to increase the diversity and robustness of the training data, data augmentation techniques can be employed. These techniques involve applying various transformations, such as rotation, scaling, flipping, or adding noise, to the labeled data. Data augmentation serves to reduce overfitting and enhance the generalization capability of the trained model.
for our case, Grayscale conversion was applied to 8 percent of the images. This process converts colored images into grayscale, which can help in reducing the computational requirements of the model and simplifying the learning process.

3.1.2.5 Data Splitting

In the data preprocessing pipeline, after the labeled and augmented data is prepared, it is typically split into training, validation, and testing sets. Each of these sets serves a specific purpose in the model development process.

- **Training Set:** the training set is used to train the model. It is the largest portion of the data and is used to optimize the model's parameters and learn the underlying patterns and relationships in the data.
- **Validation Set:** the validation set is used for hyperparameter tuning and model selection. It is a separate set of data that is not used during the training process. The model's performance is evaluated on the validation set, and this evaluation helps in selecting the best hyperparameters and configurations for the model. The validation set acts as a measure of how well the model is performing during training and helps in preventing overfitting.
- **Testing Set:** the testing set is used to evaluate the final performance of the trained model. It provides an unbiased assessment of the model's generalization capabilities on unseen data. The testing set is not used during the training or validation stages and is crucial for estimating the model's performance in real-world scenarios.

In this specific case, a split ratio of 70:20:10 was applied. This means that 70 percent of the data was allocated to the training set, 20 percent to the validation set, and 10 percent to the testing set. This split ratio is a common practice.

3.1.2.6 Data Description

The figure 3.4 presents a dashboard summarizing key statistics of an image dataset. It includes:

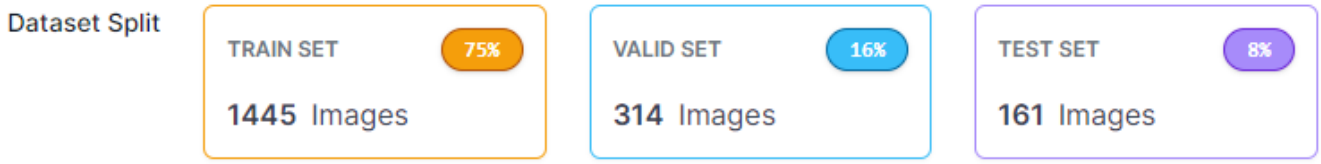


Figure 3.3: Split ratio after augmentation.

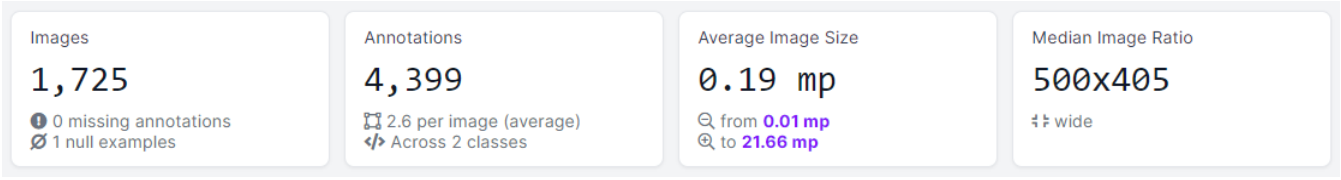


Figure 3.4: Comprehensive Dataset Statistics Dashboard.

- **Total Images:** the dataset contains 1,725 images, providing a substantial variety for training and testing.
- **Annotations:** with 4,399 annotations, each image has multiple labels, indicating a richly annotated dataset.
- **Average Image Size:** the average image size is 0.19 megapixels, suggesting a moderate resolution across the dataset.
- **Median Image Ratio:** the most common image dimension is 500x405 pixels, which could be the standardized format for the dataset.

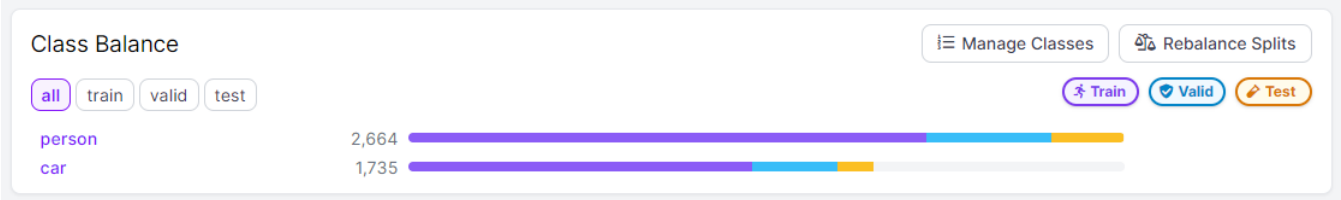


Figure 3.5: Class Distribution Visualization.

The figure 3.5 showcases the class balance across training, validation, and testing sets. It features:

- **Class Categories:** two main classes are highlighted, 'person' and 'car', with respective image counts.
- **Image Count:** the 'person' class includes 2,664 images, while the 'car' class comprises 1,735 images.
- **Visualization:** the data is represented through horizontal bars, with the length of each bar corresponding to the number of images in each class.

The figure 3.6 presents insights into the size and aspect ratio distribution of elements within a dataset. The second bar Aspect Ratio Distribution chart focuses on aspect ratios:

Size Category	Number of Images
Small	21
Medium	972
Large	582
Jumbo	150

Table 3.1: Size Distribution of Images.

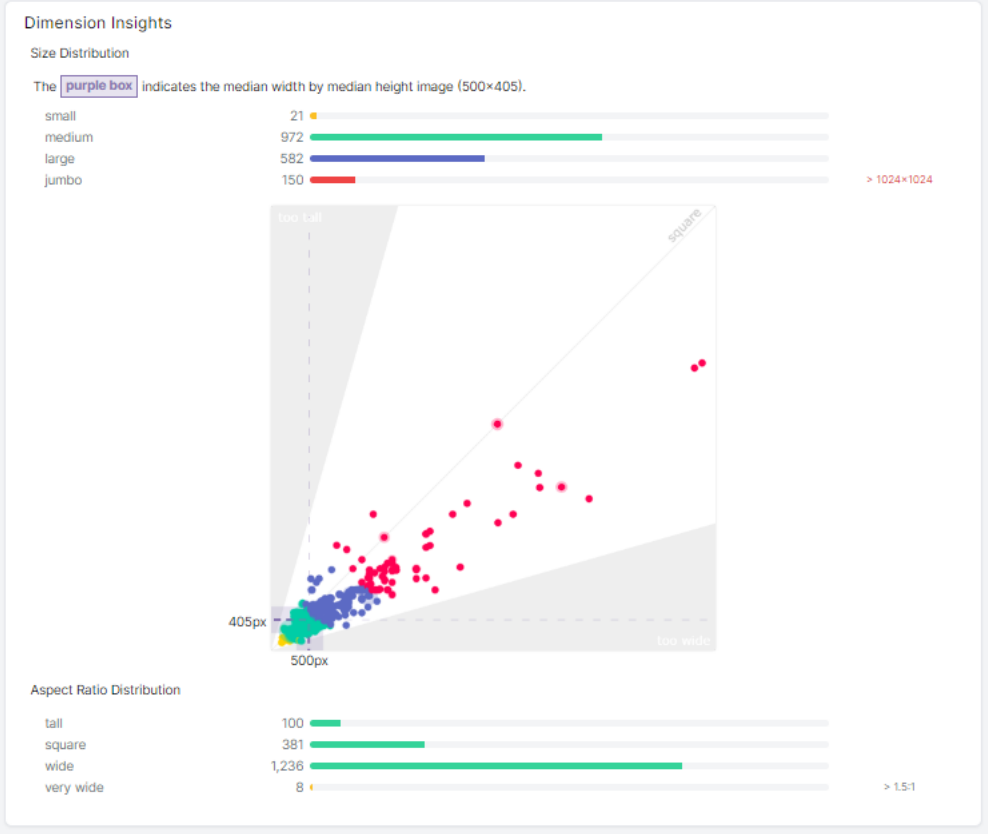


Figure 3.6: Dimension Insights, Size and Aspect Ratio Distribution.

- **Square:** there are 100 square images (where width equals height).
- **Wide:** the dataset includes 1,236 wide images.
- **Very Wide:** there are 8 very wide images.
- **the note "< 1:1":**indicates that the square ratios have a width-to-height ratio less than 1.

The figure 3.7 displays an annotation heatmap of the image dataset, highlighting the areas where objects are most frequently annotated.

- **Annotation Categories**: total Annotations 4,399, Car (1,735), Person (2,664)
- **Heatmap Interpretation:** the heatmap shows the density of annotations across the images.The color gradient ranges from blue (low density) to green and yellow (high density).The central area of the images has the highest concentration of annotations, as indicated by the green and yellow colors.The edges of the images have fewer annotations, shown in blue.

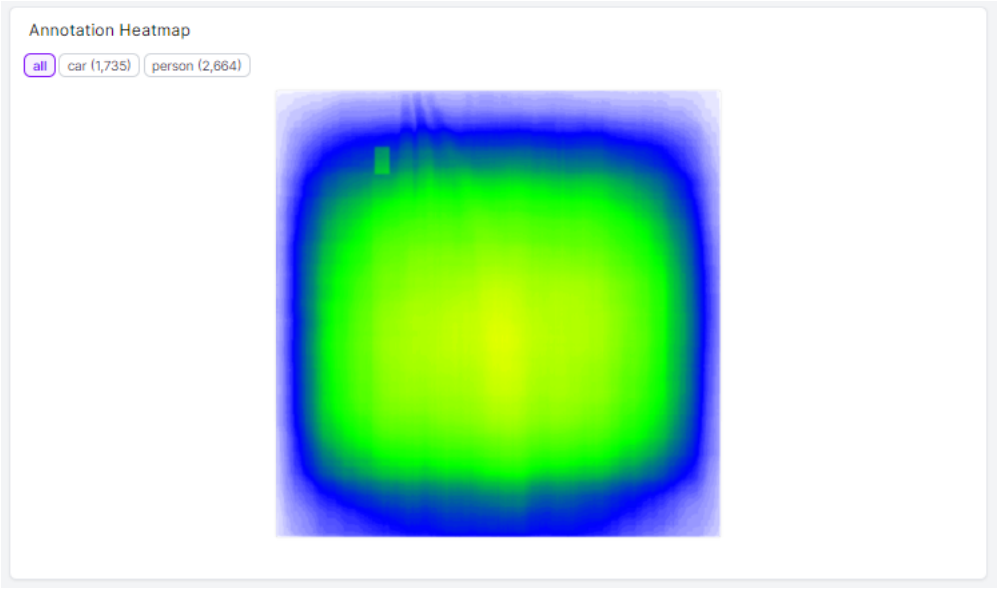


Figure 3.7: Annotation Heatmap, Car and Person Distribution.

This heatmap provides a visual representation of where objects, such as cars and people, are typically located within the images in the dataset. The central concentration suggests that objects are more likely to be annotated in the middle of the images.

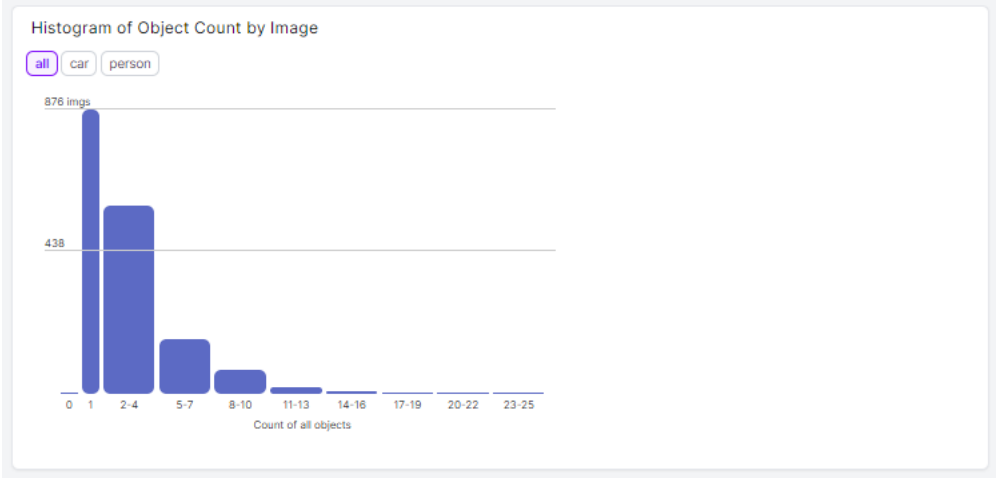


Figure 3.8: Histogram of Object Count by Image.

The figure 3.8 is a histogram that illustrates the distribution of the number of objects found in different images within a dataset. The x-axis represents the count of objects in an image, which are grouped into ranges, while the y-axis represents the number of images that fall into each object count range.

The histogram provides insights into the distribution of object instances across different images in the dataset. It allows us to observe the variation in object counts for the 'car' and 'person' classes and understand the frequency of occurrence within specific ranges.

3.1.3 Algorithm Selection

For the choice of algorithm, YOLO was selected due to its efficiency as a single-pass algorithm. Unlike traditional CNNs, which require two passes over the image—first to identify regions of interest and then to detect objects within those regions—YOLO processes the entire image in a single pass. This streamlined approach not only simplifies the detection process but also significantly reduces computational overhead. To further optimize the proposed system and minimize unnecessary model invocations, a basic motion detection algorithm based on the subtraction of successive frames was implemented. This method, detailed in the state-of-the-art section, was chosen for its simplicity and effectiveness. Its primary role is to activate the object detection model only when motion is detected, thereby conserving computational resources. Once motion is detected, the object detection model is triggered, and if the detection confirms the presence of relevant objects, the recording model is then activated to capture the event.

YOLOv9 was opted for over its predecessors due to its superior performance and innovative features. YOLOv9 delivers better results compared to earlier versions of YOLO, thanks to the integration of cutting-edge techniques such as Programmable Gradient Information (PGI) and the Generalized Efficient Layer Aggregation Network (GELAN). PGI enhances the model's learning capabilities by providing more informative gradients during training, while GELAN improves the efficiency of layer aggregation, leading to better utilization of network resources. These advancements not only boost YOLOv9's accuracy but also its overall efficiency, making it the ideal choice for our application. The combination of these novel techniques ensures that YOLOv9 remains at the forefront of object detection technology, offering unmatched performance and reliability.

3.1.3.1 YOLOV9

YOLOv9 is the latest iteration in the YOLO series of real-time object detection models. Released on February 21, 2024, by Chien-Yao Wang, I-Hau Yeh, and Hong-Yuan Mark Liao, YOLOv9 introduces several innovative techniques to enhance the efficiency and accuracy of object detection tasks [pap].

Traditional deep neural networks often suffer from issues such as vanishing and exploding gradients, which can hinder the training process and model performance. Techniques like batch normalization and advanced activation functions have been developed to mitigate these issues to a significant extent. However, YOLOv9 takes a deeper look at analyzing and addressing the problem of the information bottleneck, a challenge that was not explicitly tackled in previous YOLO versions.These breakthroughs ensure that YOLOv9 achieves exceptional real-time object detection performance, setting a new benchmark for precision and speed in this domain

3.1.3.2 Core Innovations of YOLOv9

According to Ultralytics, YOLOv9's advancements are fundamentally focused on addressing the challenges of information loss in deep neural networks. Its design is centered around the Information Bottleneck Principle and the innovative use of Reversible Functions, ensuring that YOLOv9 maintains high efficiency and accuracy. Its clever design allows the deep model to reduce the number of parameters by 49% and the number of calculations by 43% compared with YOLOv8. And it still has a 0.6 Average Precision improvement on the MS COCO dataset [vis].

3.1.3.3 YOLOV9 Architecture

The architecture of YOLOv9 incorporates several innovative features that distinguish it from its predecessors and contribute to its superior performance in real-time object detection tasks. Here's a breakdown of the key architectural components based on the provided search results:

1. **Programmable Gradient Information (PGI):** YOLOv9 introduces PGI, a mechanism designed to address the information bottleneck problem in deep neural networks. It focuses on the precise and efficient backpropagation of gradients, crucial for model training and efficiency .PGI integrates an Auxiliary Supervision Node and reversible functions to ensure efficient gradient flow and mitigate the risk of information degradation, especially in deeper layers.

2. **Information Bottleneck Principle**: The Information Bottleneck Principle reveals a fundamental challenge in deep learning: as data passes through successive layers of a network, the potential for information loss increases. This phenomenon is mathematically represented as:

$$I(X, X) >= I(X, f_theta(X)) >= I(X, g_phi(f_theta(X)))$$

   where mutual information is denoted, with f and g representing transformation functions with parameters θ and ϕ, respectively. YOLOv9 counters this challenge by implementing Programmable Gradient Information (PGI), which aids in preserving essential data across the network's depth, ensuring more reliable gradient generation and, consequently, better model convergence and performance [ult].

3. **Reversible Functions:** Reversible Functions, a fundamental aspect of YOLOv9's design, play a pivotal role in maintaining information integrity within the model. A function is considered reversible if it can be inverted without any loss of information, represented by :

$$X = v_zeta(r_psi(X))$$

Here, psi and zeta represent parameters for the reversible and its inverse function, respectively. This property is indispensable in deep learning architectures as it enables the network to uphold a seamless information flow, facilitating more precise updates to the model's parameters. YOLOv9 integrates reversible functions into its architecture to counteract the risk of information deterioration, particularly in deeper layers, thus safeguarding critical data for object detection tasks [ult].

4. **Generalized Efficient Layer Aggregation Network (GELAN):** GELAN is a significant innovation in YOLOv9, enhancing the model's learning capacity and computational efficiency It merges the features of CSPNet and ELAN, focusing on gradient path planning and ensuring a versatile structure accommodating various computational blocks [rP].

5. **Multi-Level Auxiliary Information:** YOLOv9 employs specialized networks to combine gradient information across the model's layers, addressing the challenge of information loss in deep supervision models

6. **Superior Performance Benchmarks :** YOLOv9 achieves a higher mean Average Precision (mAP) compared to its predecessors, demonstrating its superior performance in real-time object detection tasks

7. **Efficient Training and Inference:** The architecture of YOLOv9, particularly the PGI mechanism, is designed to facilitate efficient training and inference processes, ensuring that critical information is fully comprehended by the model.

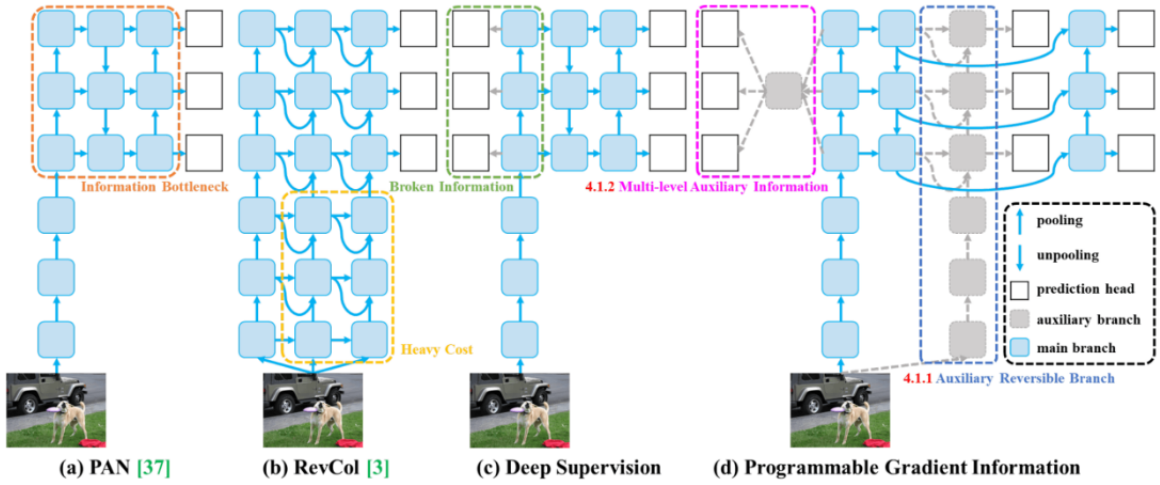


Figure 3.9: YOLOV9 architecture [Hik].

3.1.3.4 YOLOv9 Performance Comparison

The comparison between YOLOv9 and state-of-the-art (SOTA) models highlights notable advancements across several metrics.YOLOv9 surpasses existing methods in terms of parameter efficiency, utilizing fewer parameters while maintaining or enhancing accuracy. Additionally, YOLOv9 exhibits superior computational efficiency, outperforming both models trained from scratch and those based on depth-wise convolution and ImageNet-pretrained models [enc].

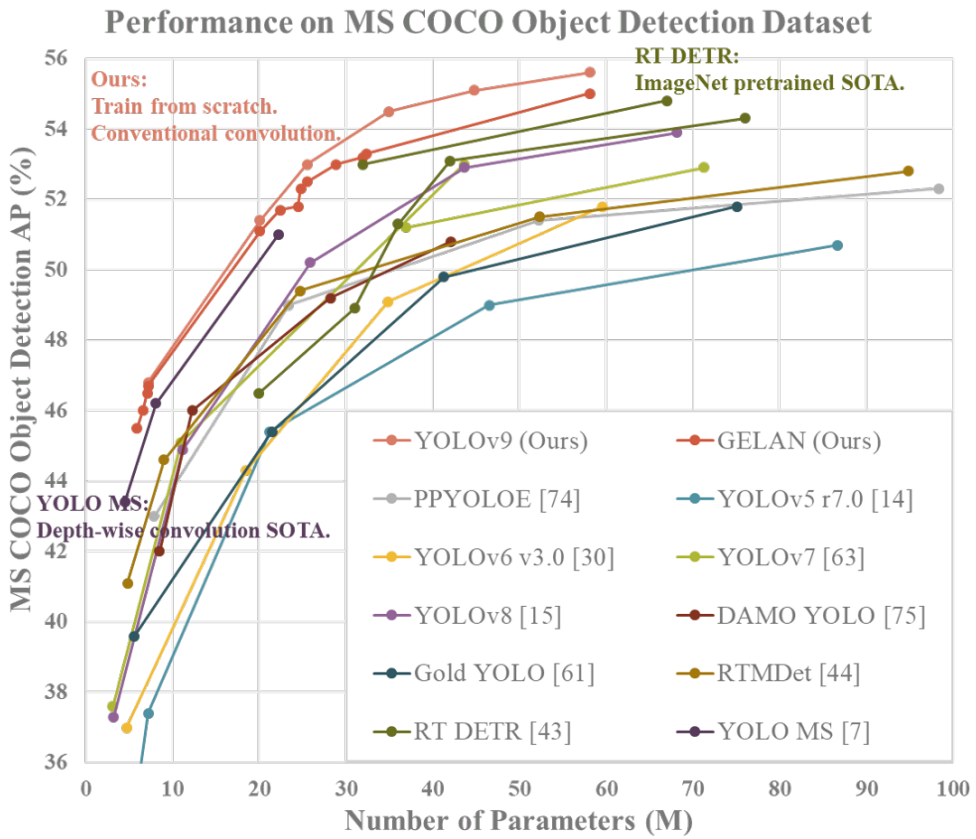


Figure 3.10: Comparison of YOLOv9 performance with state-of-the-art Models [rP].

This figure shows a comparison between YOLOv9 and the most effective models, including YOLO MS-S for lightweight models, YOLO MS for medium models, YOLOv7 AF for general models, and YOLOv8-X for large models.

When compared to YOLO MS for lightweight and medium models, YOLOv9 shows significant improvements, with around 10% fewer parameters and requiring 5-15% fewer computations, while still achieving a 0.4-0.6% increase in Average Precision (AP).

In comparison to YOLOv7 AF, YOLOv9-C offers a substantial reduction of 42% in parameters and 22% in computations, while maintaining a similar AP of 53Lastly, when contrasted with YOLOv8-X, YOLOv9-E demonstrates a decrease of 16% in parameters and 27% in computations, along with a notable improvement of 1.7% in AP. [rP]

Paper	Algorithm	Key Performance
[RHGS16]	Faster R-CNN	mAP: 73.2% (PASCAL VOC), 21.9% (MS COCO)
[PP21]	Motion Pattern Analysis	Recognition: 95.6% (PETS), 95.1% (UMN)
[BKH$^+$17]	Neural Networks	Recognition: 94%
[BMB21]	STDnet-ST++	mAP Improvement: 2.3% (USC-GRAD-STDdb), 1.0% (UAVDT)
[B$^+$22]	YOLOv3	mAP: 76% (cats), 72% (persons)

Table 3.2: Comparative Performance of related work.

3.2 Implementation

3.2.1 Object Detection Model Development

3.2.1.1 Setup and Environment Preparation

The model development was conducted on Google Colab, utilizing the free NVIDIA T4 GPU provided by the platform. This setup provided a robust environment for training and fine-tuning the YOLOv9 model. The operating system on Colab is Linux-based, and the programming was done using Python. To facilitate deep learning model implementation, PyTorch, an open-source machine learning library, was used. Additionally, the YOLOv9 repository was cloned from GitHub, which included all necessary requirements that were installed via the provided requirements file.

3.2.1.2 Data Handling

The used dataset was comprehensively prepared and managed using the Roboflow platform. This process included annotation, augmentation, and splitting of the dataset, all handled directly within Roboflow. The platform provided tools to annotate the dataset, ensuring that each image was accurately labeled for training. Data augmentation techniques were applied to increase the diversity of the training

set, enhancing the model's ability to generalize from the data. The dataset was then split into training, validation, and test sets to facilitate effective model evaluation.

To integrate the dataset into our Colab environment, The Roboflow API was used to clone the dataset directly into Colab. The following code snippet demonstrates how this was achieved:

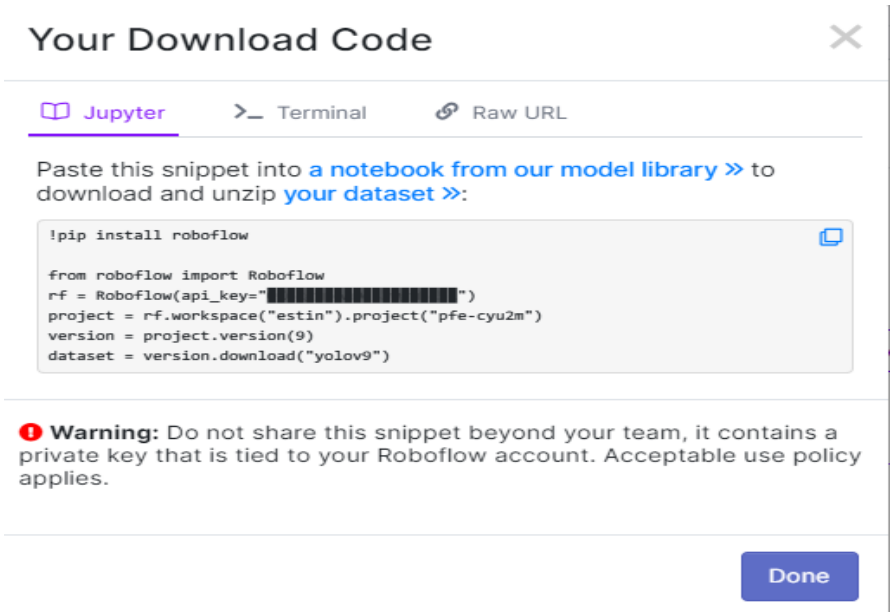


Figure 3.11: Code snippet demonstrates how to clone the dataset directly into Colab.

This approach ensured that the used dataset was readily available for model training and evaluation, with minimal setup required within the Colab environment.

3.2.1.3 Model Training

1. **Model Pre-training**

   the YOLOv9 algorithm implementation began with cloning the YOLOv9 repository from GitHub into the Colab environment. The necessary dependencies were installed according to the instructions provided in the repository. The model utilized pre-trained weights from the MS COCO dataset, which were downloaded and used as a starting point for fine-tuning with the used custom dataset. The pre-training of the YOLOv9 model was done using the following script from the GitHub repository:

```
# Train YOLOv9 models
python train_dual.py --workers 8 --device 0 --batch 16 --data data/coco.yaml --
    img 640 --cfg models/detect/yolov9-c.yaml --weights '' --name yolov9-c --hyp
    hyp.scratch-high.yaml --min-items 0 --epochs 500 --close-mosaic 15
```

   - –workers 8: Specifies 8 CPU workers for data loading, speeding up the data loading process.
   - –device 0: Utilizes the first GPU for training.
   - –batch 16: Sets the batch size to 16, processing 16 images before updating model parameters.

- –data data/coco.yaml: Points to the COCO dataset configuration file.
- –img 640: Resizes input images to 640x640 pixels.
- –cfg models/detect/yolov9-c.yaml: Specifies the model architecture configuration file.
- –weights ”: Indicates training from scratch with no pre-loaded weights.
- –name yolov9-c: Names the training run "yolov9-c".
- –hyp hyp.scratch-high.yaml: Points to the hyperparameter configuration file optimized for training from scratch.
- –min-items 0: Uses images with zero objects for training.
- –epochs 500: Trains for 500 epochs.
- –close-mosaic 15: Closes mosaic augmentation after 15 epochs to help the model focus more on original images in later stages of training.

This pre-trained model served as the basis for further fine-tuning on the custom dataset.

2. **Model fine-tuning**

The training process for the custom dataset was executed using the following modified script:

```
!python train.py --img 640 --batch 16 --epochs 25 --data {dataset.location}/data
   .yaml --cfg yolov9.yaml --weights yolov9-c.pt --name yolov9_results
```

- –img 640: Specifies the input image size. Images are resized to 640x640 pixels before being fed into the model. This parameter affects both the model’s accuracy and the computational resources required. Larger images can improve detection performance but require more memory and processing power.
- –batch 16: Sets the batch size for training. A batch size of 16 means that 16 images are processed before updating the model’s parameters. This parameter impacts training speed and model performance. A larger batch size can make training faster and more stable, but it requires more memory.
- –epochs 25: Sets the number of training epochs. An epoch is one complete pass through the entire training dataset. Training for 25 epochs means the model will see the entire dataset 25 times, which helps in learning the patterns and features more effectively. However, too many epochs can lead to overfitting, where the model performs well on the training data but poorly on unseen data.

- –data dataset.location/data.yaml: Points to the data configuration file. This YAML file contains information about the dataset, such as the paths to the training, validation, and test datasets, as well as the class names. It ensures that the model knows where to find the data and how it is structured.
- –cfg yolov9.yaml: Specifies the model configuration file. This YAML file defines the architecture of the YOLOv9 model, including the number of layers, types of layers, and other architecture-specific parameters. It ensures that the model is built according to the desired specifications.
- –weights yolov9-c.pt: Indicates the initial weights for the model. The file yolov9-c.pt contains pre-trained weights from the initial training on the MS COCO dataset.
- –name yolov9_results: Sets the name for the training run.

3. **Examine Training Results**

After completing the training process of the YOLOv9 model, A detailed evaluation was conducted to understand the model's performance using both qualitative and quantitative metrics. The validation phase involved assessing the model's predictions against a separate validation dataset to ensure that the model generalizes well to unseen data.

- **Validation Setup** was conducted using the following parameters:
  - Data Configuration: The data configuration was set to use the prepared dataset located at /content/yolov9/pfe-9/data.yaml.
  - Weights: The model weights used for validation were the best weights obtained during training, located at /content/yolov9/runs/train/exp/weights/best.pt.
  - Batch Size: The batch size was set to 16, indicating that 16 images were processed simultaneously during validation.
  - Image Size: The images were resized to 640x640 pixels for consistency with the training process.
  - Confidence Threshold: A confidence threshold of 0.001 was used, meaning predictions with confidence scores below this threshold were ignored.
  - IoU Threshold: The Intersection over Union (IoU) threshold for evaluating detections was set to 0.5.
  - Maximum Detections: The maximum number of detections per image was limited to 300.
- **Model Performance Metrics:** The evaluation metrics provide a comprehensive understanding of the model's performance. The key metrics include Precision (P), Recall (R), mean Average

Precision at IoU threshold 0.5 (mAP50), and mean Average Precision across IoU thresholds from 0.5 to 0.95 (mAP50-95).

– **Precision (P):** Precision measures the accuracy of the positive predictions made by the model. It is calculated as the ratio of true positive detections to the total number of positive detections (true positives + false positives). For the proposed model, the overall precision is 0.869, indicating that approximately 87% of the time, the model's positive predictions were correct.

Class	Precision
Car	0.855
Person	0.884

Table 3.3: Precision Metric results.

– **Recall (R):** Recall measures the model's ability to detect all relevant instances in the dataset. It is calculated as the ratio of true positive detections to the total number of actual positives (true positives + false negatives). The overall recall for the proposed model is 0.824, suggesting that the model correctly identified about 82% of all objects that should have been detected.

Class	Recall
Car	0.83
Person	0.819

Table 3.4: Recall Metric results.

– **Mean Average Precision (mAP)**: mAP is a comprehensive metric used to evaluate the precision and recall across all classes and at different IoU thresholds. It is the mean of the average precision values for each class. The mAP50 is the mean Average Precision at an IoU threshold of 0.5, which for the proposed model is 0.891, indicating high precision and recall at this threshold. The mAP50-95, which averages precision across IoU thresholds from 0.5 to 0.95, is 0.558, showing the model's robustness across different IoU settings.

Category	mAP50	mAP50-95
Car	0.899	0.638
Person	0.883	0.478

Table 3.5: Mean Average Precision (mAP) results.

These metrics indicate that the model performs well across different classes, with high precision and recall values, especially in detecting cars and persons. The mAP50-95 values, although lower than mAP50, still reflect the model's ability to generalize across various IoU thresholds.

Figure 3.12: Model Performance Metrics.

- **Confusion Matrix:** In object detection, the confusion matrix is used to describe the performance of the model by comparing the predicted labels to the true labels. The four key components of the confusion matrix are:
  - True Positives (TP): Correctly detected objects.
  - False Positives (FP): Incorrectly detected objects (model detected something that is not present).
  - False Negatives (FN): Objects that were present but not detected by the model.
  - True Negatives (TN): Correctly identified background (not typically considered in object detection).

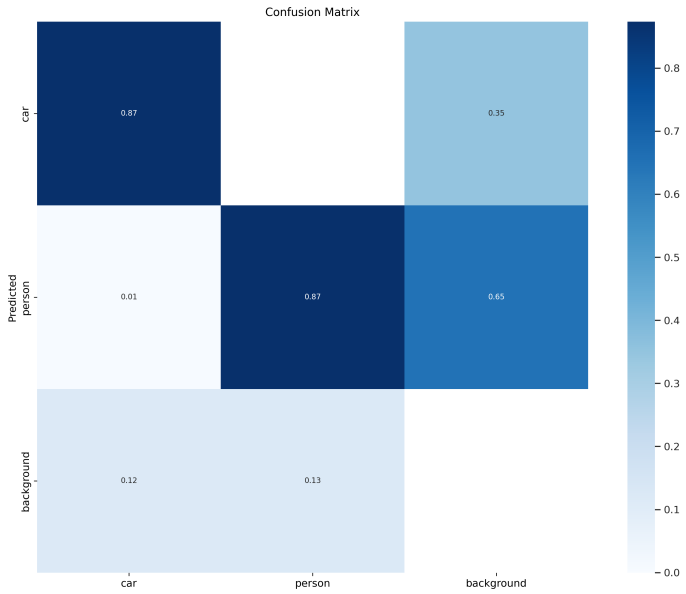


Figure 3.13: Confusion matrix.

Performance Metric	Time (milliseconds per image)
Pre-processing Time	0.5
Inference Time	25.1
Non-Maximum Suppression (NMS) Time	5.2

Table 3.6: Performance Speed Metrics.

The speed metrics highlight the efficiency of the model in making predictions. The inference time of approximately 25.1 milliseconds per image is particularly notable, demonstrating the model's capability to process images in real-time, which is essential for applications requiring quick decision-making.

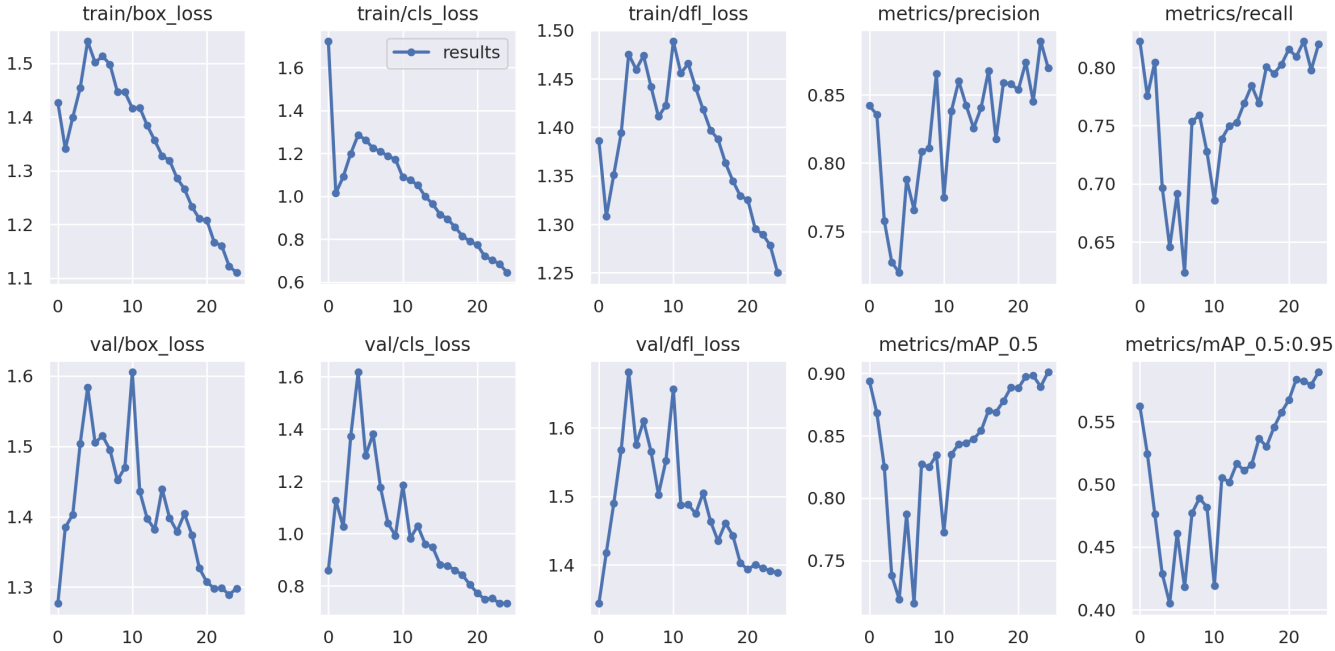


Figure 3.14: Training and Validation Performance Metrics of the YOLOv9 Model.

- **Visualization of Training Progress:** the training progress is visualized through various plots that show the evolution of different loss components and performance metrics over epochs. Below is a detailed explanation of each plot in the provided figure:
  - train/box_loss: This plot shows the training loss related to bounding box regression. A decreasing trend indicates that the model is improving its accuracy in predicting the bounding box coordinates for detected objects. The box loss starts around 1.5 and decreases steadily, reaching approximately 1.1 by the end of the training, indicating consistent improvement.
  - train/cls_loss: This plot shows the training loss related to classification. It measures how well the model is learning to classify objects within the bounding boxes. The cls_loss starts at around 1.6 and drops to approximately 0.6, showing a significant reduction, which indicates better classification performance.
  - train/dfl_loss: The Distribution Focal Loss (DFL) is another component of the training loss, which helps in refining the bounding box predictions. The DFL loss starts high, around 1.5, and decreases to about 1.25, indicating that the model is becoming more precise in its predictions.
  - metrics/precision: This plot shows the precision of the model during training. An increasing trend signifies that the model is making fewer false positive errors over time. The precision starts around 0.65 and improves to about 0.85, showing the model's increasing accuracy.
  - metrics/recall: This plot shows the recall of the model, indicating its ability to detect all relevant instances. An upward trend is observed, starting from around 0.65 and reaching

approximately 0.82, showing the model's improving detection capability.

- val/box_loss: Similar to the training box loss, this plot shows the validation loss for bounding box regression. Although more fluctuating, the overall trend is downward, starting around 1.6 and decreasing to approximately 1.3, indicating better generalization of the model to validation data.
- val/cls_loss: This plot shows the validation classification loss. The cls_loss decreases from around 1.6 to about 0.8, reflecting better classification performance on the validation set.
- val/dfl_loss: The validation Distribution Focal Loss shows a similar downward trend, starting around 1.6 and decreasing to about 1.4, indicating improved prediction precision on the validation set.
- metrics/mAP_0.5: This plot shows the mean Average Precision at an IoU threshold of 0.5 during training. The upward trend, from around 0.45 to approximately 0.90, indicates significant improvement in the model's performance.
- metrics/mAP_0.5:0.95: This plot shows the mean Average Precision averaged across IoU thresholds from 0.5 to 0.95. The metric starts at about 0.40 and increases to approximately 0.75, demonstrating the model's robustness across varying IoU thresholds.

These visualizations provide a clear indication of the model's learning progress and its eventual stabilization at higher performance levels. They are essential for diagnosing the training process and ensuring that the model does not overfit or underfit the training data.

In summary, the detailed examination of the training and validation results confirms that the YOLOv9 model, fine-tuned with the proposed custom dataset, performs well in detecting objects of interest with high accuracy and efficiency. The model's robust performance metrics and real-time processing capabilities make it suitable for deployment in proposed intelligent video recording optimization solution.

3.2.1.4 Comparative Analysis of Trained Models

The comparative analysis aids in identifying the most suitable model for deployment based on predefined criteria and objectives. By considering various performance metrics, computational resources, and deployment constraints, an informed decision can be made about which model is best suited for the intended application. Comparative analysis doesn't stop at model selection; it also informs iterative improvement strategies. By understanding how different models perform, areas for refinement and optimization can be identified. leading to continuous improvement in model performance over time.

Based on the comparative analysis, the 7th Model is chosen for the proposed system due to its highest mAP, precision, and recall, making it the most suitable choice for deployment in the system. This model

Table 3.7: Comparative Table of Trained Models.

Model	Dataset Size	Epochs	Precision	Recall	mAP (0.5)
1st Model	664 Images	50	77.9%	73.7%	76.8%
2nd Model (yolov8s Model)	664 Images	50	70.0%	71.7%	74.3%
3rd Model	1177 Images	30	81.6%	73.3%	80.9%
4th Model	1469 Images	30	85.3%	75.2%	82.4%
5th Model	1294 Images	30	83.6%	75.1%	82.2%
6th Model	1622 Images	25	86.5%	76.7%	85.5%
7th Model	1920 Images	25	87.0%	82.0%	90.1%
8th Model	1927 Images	100	85.1%	82.6%	88.7%

demonstrates the best balance of precision and recall, ensuring high accuracy in detecting and classifying objects in the video feeds.

3.2.1.5 Model Deployment

The deployment process begins with the preparation and deployment of the YOLOv9 model, which entails converting it into a deployable format and configuring it for integration into the application environment. Following deployment, the model is made accessible through an API, facilitating its seamless integration into the application ecosystem. The deployment process ensures that the model can effectively perform inference tasks, providing accurate predictions based on input data. By following this process, the model becomes ready for deployment in real-world scenarios, contributing its object detection capabilities to the application's functionality.

3.2.2 Activity Detection Model

3.2.2.1 Introduction

Our model is designed to detect the presence of individuals and vehicles within a given scene, starting recording when these objects are detected and stopping when no objects are present. This functionality is crucial in surveillance and security systems, ensuring a timely and appropriate response to human or vehicular activity. In the following section, an overview of the model's architecture is provided, the input data it processes is described, and the interface and tools utilized in its development are discussed.

3.2.2.2 Overview of our Program

1. **Initialization:** import necessary libraries like OpenCV, supervision, Streamlit, PIL, tempfile, datetime, collections and Create the output directory for recordings if it does not exist

2. **Load Object Detection Model :** load the pre-trained object detection model using the get-model

function.

3. **The detect motion function :** it analyzes consecutive frames in a video stream to identify motion. It converts the current frame to grayscale, calculates the absolute difference between it and the previous grayscale frame (if available), and thresholds the difference to create a binary image. It then determines motion based on the number of non-zero pixels in this binary image. Finally, it returns a boolean indicating the presence or absence of motion and the current grayscale frame.

4. **the object detection function:** this function processes a frame to detect objects using a pre-trained object detection model. It converts the frame to a PIL image, applies inference using the model, filters detections based on confidence, annotates the frame with bounding boxes and labels, and returns the annotated frame along with the detections.

5. **Function for real-time Processing with camera:** this function is designed to handle real-time video processing from an ip camera or a webcam by integrating motion detection and object detection using a pre-trained model. This function initializes variables to control recording, motion detection, . It continuously processes each frame captured from the camera. The current frame is displayed using Streamlit, and both motion and object detection are performed on it. Recording starts when motion and objects are detected and continues as long as an object is present. If no motion is detected for a specified duration, recording stops. Additionally, the function monitors user input to allow for stopping the recording. Finally, it releases the webcam capture and writer resources, ensuring proper cleanup of the resources. This setup enables efficient and automated real-time video processing and recording based on detected activities.

6. **Function for video processing:** this function processes a video by capturing its frames, analyzing them for motion and the presence of objects (such as people or cars), and saving the relevant frames until no objects are detected in the scene. It serves as an alternative method for testing models in scenarios where an IP camera is unavailable. Additionally, this function is useful for video summarization, as it retains only the scenes with human activity, effectively summarizing the video by focusing on significant events. Our model combines two algorithms: one for motion detection and the other for object detection. The motion detection algorithm avoids unnecessary calls to the object detection model when there is no movement, as object detection is computationally expensive. After extensive experimentation, the proposed approach was optimized to achieve the best detection with minimal cost by invoking the object detection algorithm once every second. For a camera that captures 25 frames per second, this optimization reduces the number of calls to the object detection model from 250 to just 10 over 10 seconds, effectively saving 240 model calls.

Algorithme 1 : Intelligent video surveillance Recording algorithm.

```
Require: Surveillance video stream
Ensure: Recorded video files with object activity
  Load the object detection Yolov9 model
  Initialize variables for motion detection and recording status
  while video stream exists do
    Read the current frame from the video stream
    Use the motion detection algorithm to detect motion in the current frame
    if motion is detected then
      Use the object detection model to detect objects in the current frame
      if an object is detected then
        if not currently recording then
          Start recording and note the time of last detection and currently recording == True
        end if
      end if
    end if
    if currently recording then
      if less than 20 seconds have passed since the last detection then
        if objects are detected then
          Update the time of last detection
        end if
        Write the current frame to the video file
      else
        if no objects are detected then
          Stop recording and currently recording == False
        else
          Write the current frame to the video file and Update the time of last detection
        end if
      end if
    end if
  end while
  if Stop recording button = True then
    Stop recording and currently recording == False
  end if
```

3.2.2.3 Used Packages and Tools

To develop proposed model, Visual Studio was utilized as an integrated development environment (IDE) for its robust features and user-friendly interface. The operation was conducted within a Conda environment provided by Anaconda, incorporating numerous pre-installed Python packages essential for machine learning and data processing. This environment simplifies package management and ensures compatibility between different libraries. The programming language used for development was Python,This comprehensive setup facilitated efficient development, testing, and deployment of the proposed model.

3.2.2.4 Programming language

To implement our solution we used Python, which is a high-level, interpreted programming language known for its readability, versatility and extensive support for machine learning frameworks. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python is widely used in various fields such as data science, artificial intelligence, scientific computing and web developmentand . Its extensive standard library and active community contribute to a rich ecosystem of third-party packages, making it a powerful tool for both beginners and experienced developers.

3.2.2.5 Used Packages

1. **OpenCV (Open Source Computer Vision Library):** is a comprehensive library designed for real-time computer vision. It provides tools for image and video processing, including capturing, manipulating, and analyzing visual data. in this project, it was used to capture video frames from cameras, process these frames (e.g., resizing, color conversion), and display the processed frames. It also provides functionalities for drawing bounding boxes on images.

2. **Streamlit:** is an open-source app framework for creating and sharing data science and machine learning web applications. It allows developers to build interactive and visually appealing applications with minimal code.

   Streamlit is used to create a user interface for the real-time video detection application. It can display video feeds, show detection results, and provide interactive controls (e.g., buttons, sliders) for adjusting parameters or settings of the detection model.

3. **Supervision:** it's a Python package and a versatile tool designed to facilitate the development of computer vision applications. It is model-agnostic, meaning it can be integrated with various classification, detection, or segmentation models. The package provides connectors for popular libraries such as Ultralytics, Transformers, and MMDetection. The package supports object detection and tracking, allowing us to visualize and analyze the movement of objects within a scene.

4. **Inference:** inference is an open source package developed by Roboflow containing there primary deployment capabilities. It is performant, flexible, and highly integrated wih Roboflow projects making it extremely easy to deploy computer vision models into nearly any environment.Inference enables the deployment of a wide range of pre-trained and foundational models without an API key. To access thousands of fine-tuned models shared by the Roboflow Universe community [rP].

5. **OS module:** The os module in Python provides a way of using operating system-dependent functionality, such as reading or writing to the file system, managing paths, and interacting with

the environment. is used to handle file paths, create directories to save detection results, and manage environment variables. It ensures that the application can interact with the underlying operating system effectively.

6. **PIL (Pillow):** Pillow is a fork of the Python Imaging Library (PIL) and adds some user-friendly features. It is used for opening, manipulating, and saving many different image file formats. It's used to handle image files within the application. For example, it can convert images between different formats, resize images, or apply transformations that are not directly supported by OpenCV.

3.2.3 User Interface

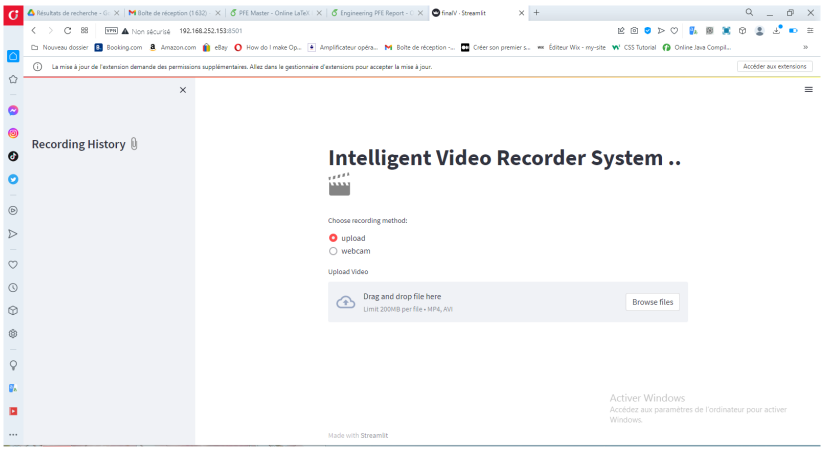


Figure 3.15: First menu page after choosing upload video option.

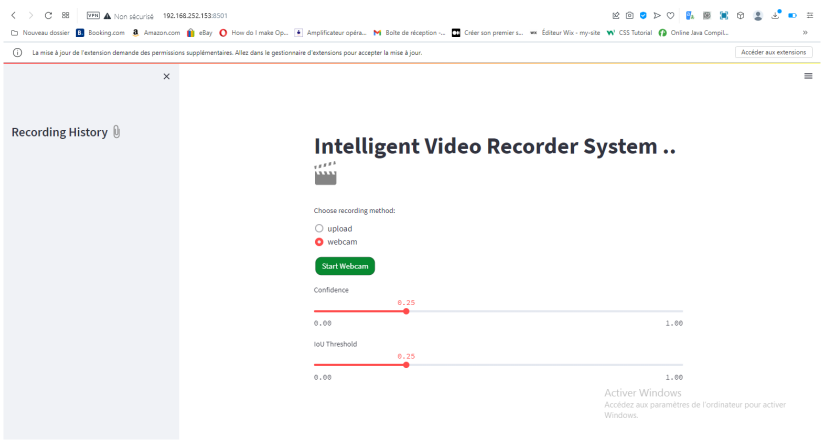


Figure 3.16: First menu page after choosing webcam streaming.

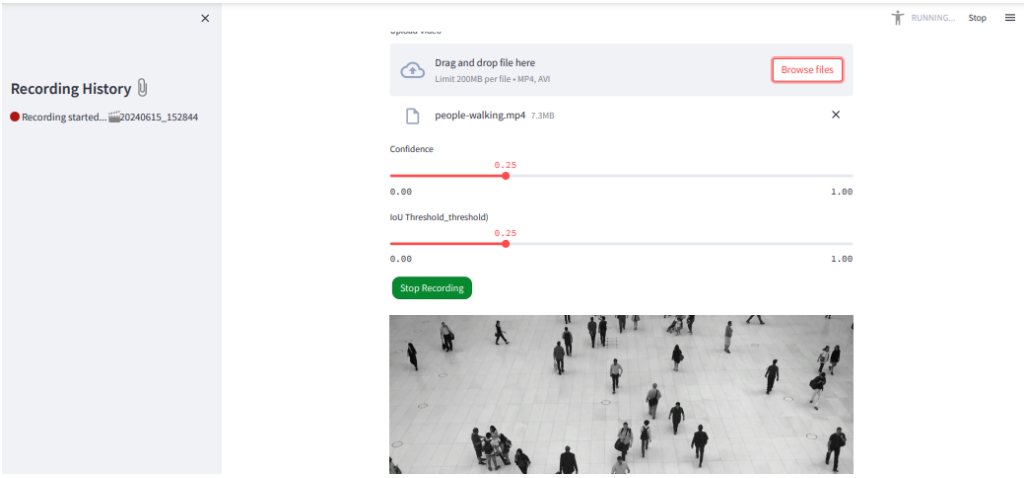


Figure 3.17: After uploading the video and start recording.

3.3 Validation

3.3.1 Test Setup

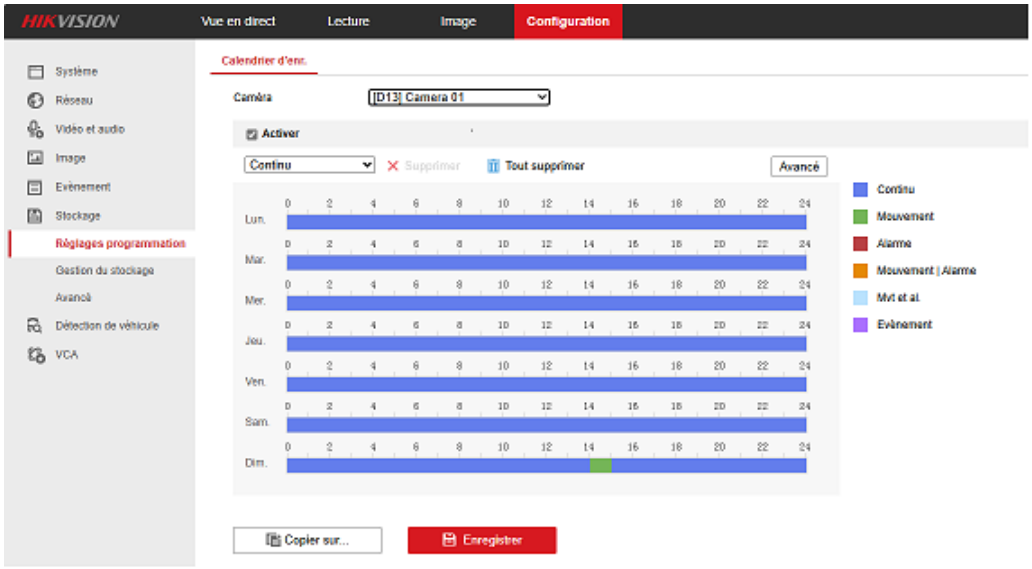
To validate our proposed application performance and this thesis objective, which is the Optimization of Recording Storage for Surveillance Systems, a comparative study was conducted with the Hikvision surveillance software used at ESTIN. The tests involved running both systems simultaneously under identical conditions to measure their effectiveness in optimizing storage by comparing the length of recorded videos.

3.3.1.1 Equipment and Configuration

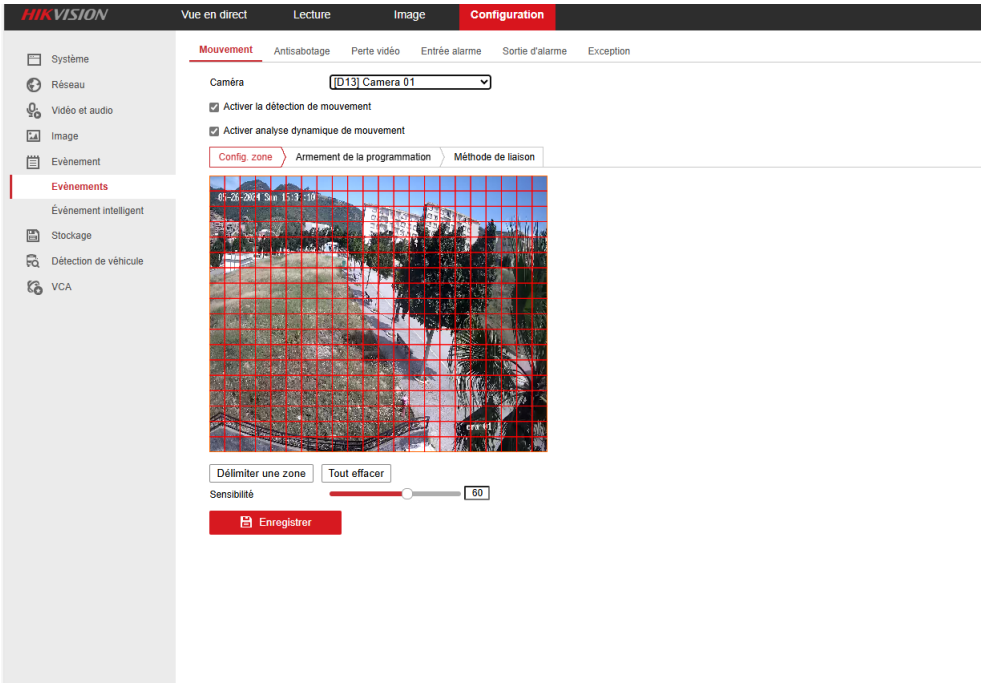
For the equipment and configuration, two types of cameras were utilized: a fixed camera and a flexible dome camera, strategically positioned in front of ESTIN's main gate and the block of TP rooms. The Hikvision system was configured with its specific hardware and software setup, while the proposed system was deployed on a local machine, specifically an i7 Core ThinkPad. This setup provided a controlled environment to accurately measure and compare the performance of both systems in terms of storage optimization.

3.3.1.2 Real-Time Test

- **Objective:** Compare the storage optimization between the proposed system and the Hikvision software over a one-hour period with motion detection enabled.
- **Methodology**: Both systems were launched at the same time and ran for one hour. The motion detection feature was activated on both systems.
- **Metrics Collected:** Length of the recorded videos from both systems.



(a) Hikvision System with Motion Detection Enabled for 1 hour



(b) Screenshot of the Hikvision System with Motion Detection Enabled

Figure 3.18: Hikvision System Configurations.

3.3.1.3 Video Upload Test

- **Objective:** Compare the performance of the proposed system and the Hikvision software using a pre-recorded video.
- **Methodology:** A 15-minute video was processed by both systems with motion detection enabled.
- **Metrics Collected:** Length of the resulting videos after processing.

3.3.2 Performance Metrics

The performance of the two systems was evaluated based on the following criteria:

- **recorded Video Length**: The total duration of the videos recorded by each system under the same conditions. A shorter recorded length indicates better storage optimization.
- **Detection Accuracy:** The ability to correctly identify and record relevant activities (human and vehicular) without missing any significant events.

3.3.3 Validation Results

Test Type	System	Video Length	Format Type	Storage
1-Hour Real-Time Test	Using Hikvision SS	60 minutes	MP4	1917 Mo
	Using Activity Detection System	13:45 minutes	AVI	271,6 Mo
15-Minute Video Test	Using Hikvision SS	6:53 minutes	MP4	379,8 Mo
	Using Activity Detection System	6:20 minutes	AVI	62,4 Mo

Table 3.8: Comparison of Recorded Lengths, Format Types, and Storage for Different Test Types.

3.3.4 Discussion

The validation results demonstrate that the proposed system significantly outperforms the Hikvision software in terms of optimizing storage. During the one-hour real-time test, conducted with the first camera in front of the top gate of the school from 2 PM to 3 PM, the proposed system recorded only 13:45 minutes of video compared to Hikvision's 60 minutes. Despite enabling the intelligent movement option on the Hikvision system, it recorded the entire duration of the test. This substantial reduction in recorded length by the proposed system highlights its superior ability to filter out irrelevant footage and focus on significant activities, thus optimizing storage usage.

Similarly, in the 15-minute video upload test, conducted with the flexible camera in front of the PW building from 2:30 PM to 2:45 PM, the proposed system reduced the recorded length to 6:20 minutes, whereas Hikvision's system recorded 6:53 minutes. The intelligent movement option on the Hikvision system was enabled and did work in this case, capturing the necessary scenes, although it experienced some bugs in certain scenes. Nonetheless, the proposed system still demonstrated improvements in efficiency by minimizing storage through accurate detection and recording of pertinent activities.

These results validate the effectiveness of proposed model in optimizing recording storage for surveillance systems, making it a valuable tool for enhancing the efficiency and cost-effectiveness of surveillance operations.

3.3.5 Estimated Storage Savings

To estimate the storage savings achieved by the proposed Activity Detection System, the data from the 1-hour real-time test was extrapolated across longer time periods. The 1-hour test showed a significant reduction in both the length of the recorded video and the storage required.

Time Period	Hikvision SS (MB)	Activity Detection (MB)	Storage Saved (MB)
1 Hour	1917	271.6	1645.4
1 Day	460680	28464	432216
1 Week	3224760	199248	3025512
1 Month	13820400	854400	12966000

Table 3.9: Estimated Storage Savings.

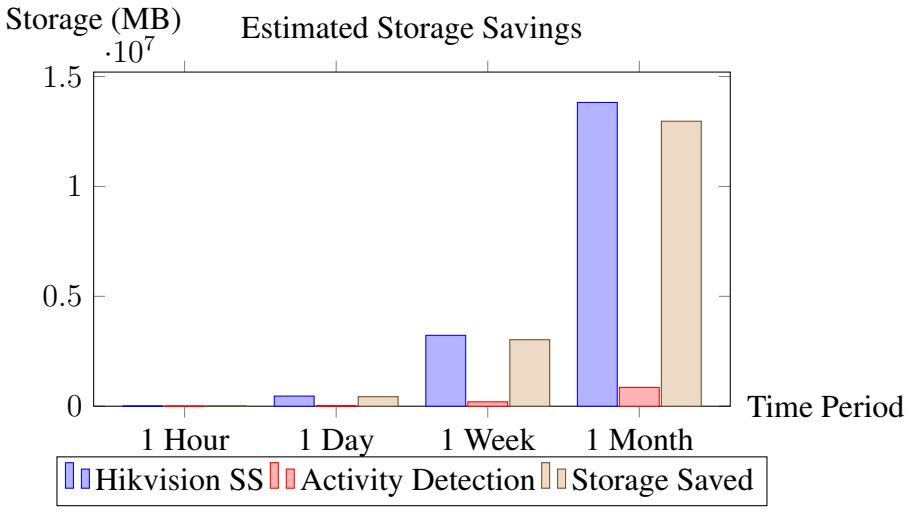


Figure 3.19: Chart of Estimated Storage Savings.

By extrapolating these results, it is evident that the proposed system can significantly reduce storage requirements over extended periods, offering daily, weekly, and monthly savings of 17.15 GB, 119.89 GB, and 514.29 GB respectively. These savings demonstrate the effectiveness and efficiency of the Activity Detection System in optimizing video storage for surveillance purposes.

3.3.6 Screenshots of Different Scenes

To illustrate the detection capabilities of the proposed activity detection model under different conditions, screenshots of the proposed system were captured in various scenarios. These screenshots demonstrate how well the system performs in daytime, nighttime, and different weather conditions.

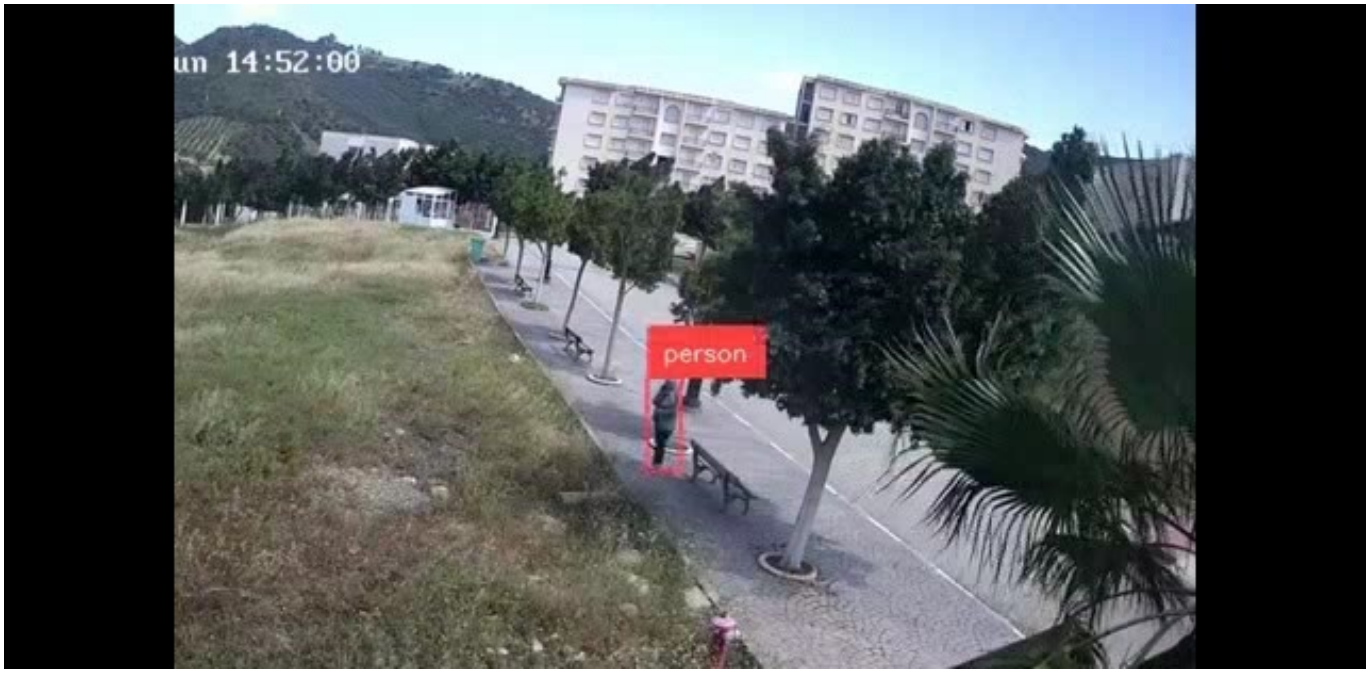


Figure 3.20: Daytime Detection Capabilities.

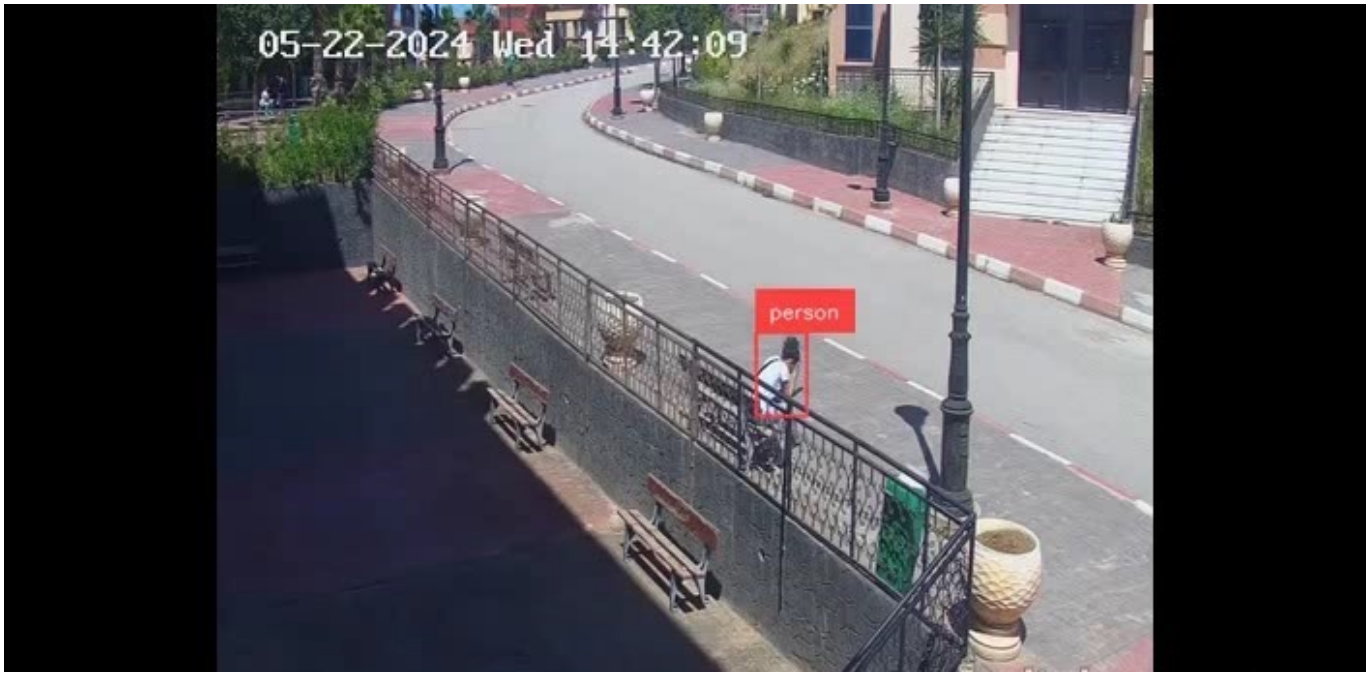


Figure 3.21: Detection Capabilities from an another camera.

Description: Screenshots taken during daylight conditions to show the clarity and accuracy of the system in detecting objects such as individuals and vehicles .

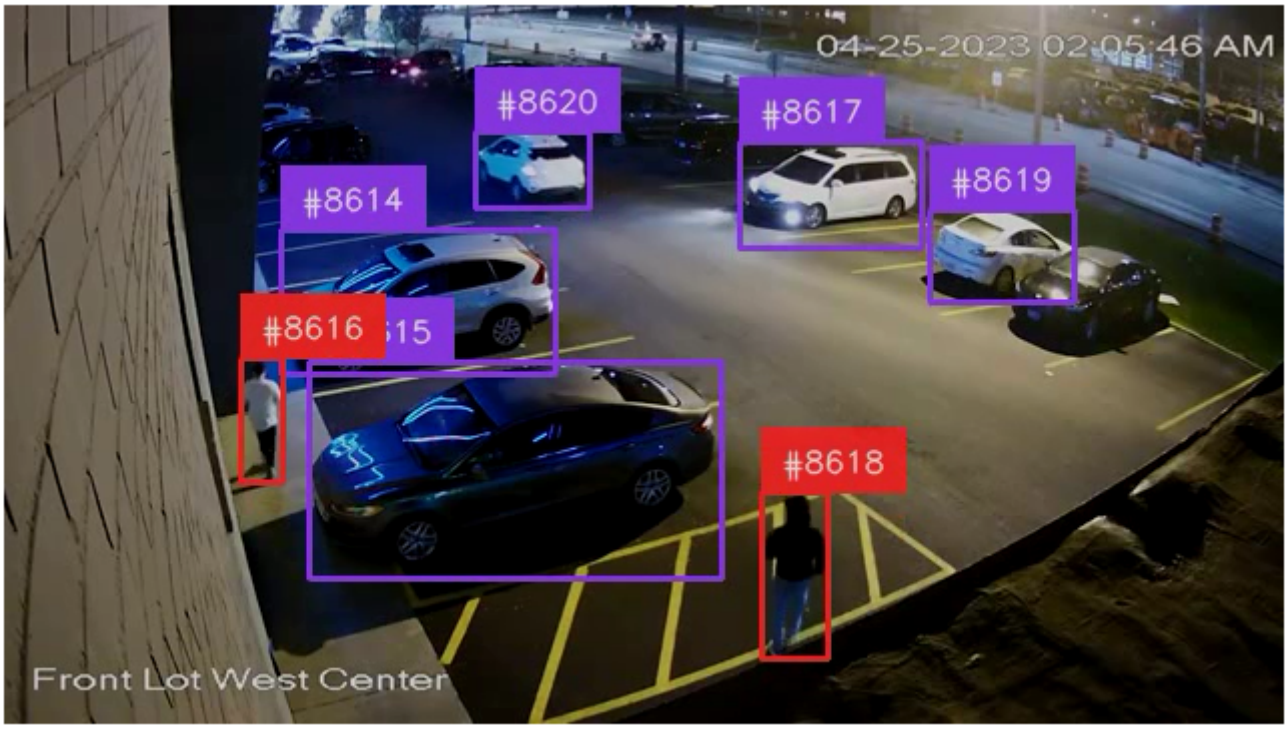


Figure 3.22: Nighttime Detection Capabilities.

Description: Screenshots taken during nighttime to evaluate the system's performance in low-light conditions.

Conclusion

In conclusion, this chapter presented the conception, implementation, and validation of an intelligent video recording optimization system for surveillance systems. The system successfully integrates activity detection techniques, such as object detection and motion detection, to optimize video recording based on relevant activities. Through the implementation process, various challenges were overcome, including the synchronization of motion and object detection modules. The system was validated through extensive testing, and the results demonstrated its effectiveness in reducing storage space while accurately capturing important activities.

The implementation of this system opens up possibilities for enhancing surveillance systems by reducing the amount of irrelevant video data collected and improving the overall efficiency of video analysis. The system can be further expanded and optimized in the future by exploring additional algorithms, enhancing the user interface, and integrating with advanced machine learning techniques. Overall, this chapter contributes to the field of intelligent video recording optimization for surveillance systems, providing a valuable solution for improved video management and analysis.

General Conclusion and Perspectives

In this thesis, The domain of intelligent video recording optimization was explored by leveraging activity detection techniques. The primary objective was to create a system for detecting important scenes in video surveillance systems before initiating recording, thereby optimizing the process to achieve real-time detection with high precision and minimal resource consumption.

This study involved a detailed examination of various models, and through comparative analysis, a hybrid system was selected. This system begins with a motion detection algorithm; successive frame subtraction was chosen because it is less costly and avoids unnecessary model calls when there is no movement. When motion is detected, the YOLOv9 model is invoked to determine if there is significant activity, such as the presence of people or vehicles. YOLOv9 was chosen for its superior detection capabilities and improved processing time. The model achieved an accuracy of 87%, which is a better precision compared to other models reviewed in the state-of-the-art, while also optimizing resource usage.

This research concluded that our approach, combining the strengths of various methods, could potentially offer a more balanced solution. Specifically, integrating motion detection techniques for initial activity screening with advanced object detection algorithms for precise identification and tracking could optimize both performance and resource usage. This approach was particularly tailored to optimize the recording system existing at ESTIN, which presented specific challenges in terms of storage and resource management.

The results of validation show a significant optimization in both time and storage, demonstrating the effectiveness of the proposed method. Future efforts should focus on refining the model structures to enhance the precision of detection and localization while maintaining computational efficiency. utilizing more powerful training devices can significantly enhance the accuracy and speed of the models. The technologies and methodologies developed in this project have promising applications beyond the security sector, potentially benefiting any field requiring real-time object detection and intelligent video recording.

In conclusion, while our project has made significant strides in optimizing intelligent video recording through activity detection, there remains room for improvement. Specifically, with the presence of high resources, we can develop more complex motion detection techniques and enhance the object detection model. We encountered difficulties collecting data, so if we trained the model on a larger dataset with data from high-quality surveillance cameras, this would augment its precision and significantly improve

performance. The insights gained from this research provide a solid foundation for future advancements, paving the way for more effective and efficient surveillance systems.

Appendix A

Glossary

Application Programming Interface (API): a set of routines, protocols, and tools for building software applications.

Deep Learning: a subset of machine learning where artificial neural networks, algorithms inspired by the human brain, learn from large amounts of data.

Intersection over Union (IoU): a metric used to measure the accuracy of an object detector on a particular dataset.

mean Average Precision (mAP): a metric used to evaluate the accuracy of object detection models.

Principal Component Analysis (PCA): a statistical procedure that transforms possibly correlated variables into a smaller number of uncorrelated variables called principal components.

Singular Value Decomposition (SVD): a mathematical technique used in machine learning to reduce the dimensionality of data.

Support Vector Machine (SVM): a supervised machine learning algorithm which can be used for both classification or regression challenges.

Appendix B

Acronyms

AI	Artificial Intelligence
AI HLEG	Artificial Intelligence High-Level Expert Group
API	Application Programming Interface
AVI	Audio Video Interleave
CCTV	Closed Circuit Television
CLI	Command-line interface
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture
DNN	Deep Neural Network
DQN	Deep Q-Networks
DW	Directed Work
GFLOPs	Giga Floating Point Operations Per Second
GPU	Graphics Processing Unit
HD	High Definition
IoU	Intersection over Union
LSTM	Long Short-Term Memory
mAP	mean Average Precision
mAP@0.5	mean Average Precision at Intersection over Union threshold of 0.5
ML	Machine Learning
MP4	MPEG-4 Part 14

Ms COCO	Microsoft Common Objects in Context
NMS	Non-Maximum Suppression
PCA	Principal Component Analysis
PIL	Python Imaging Library
PR	Pattern Recognition
PX	Pixel
PW	Practical Work
RCNN	Region-based Convolutional Neural Network
RPN	Region Proposal Network
SVD	Singular Value Decomposition
SVM	Support Vector Machine
SS	Surveillance System
SSD	Single Shot MultiBox Detector
STDnet-ST++	Spatio-temporal ConvNet for small object detection
TP	True Positive
TPU	Tensor Processing Unit
TN	True Negative
UI	User Interface
YOLO	You Only Look Once
YAML	Yet Another Markup Language

Bibliography

[AFG21] Yali Amit, Pedro Felzenszwalb, and Ross Girshick. Object detection. In K. Ikeuchi, editor, *Computer Vision*. Springer, 2021.

[Arm] Arm's glossary entry on pattern recognition.

[AY16] Sadoun Abdelbaki and Ouellabi Yacine. Détection et suivi des objets mobiles: Application dans un environnement de foule. Master's thesis, Université ECHAHID HAMMA LAKHDAR D'EL OUED, Algérie, 2016.

[B+22] Noureddine BOUMEDIENE et al. *Détection d'objet en temps réel en utilisant une approche basée sur l'apprentissage profond*. PhD thesis, Université Ibn Khaldoun-Tiaret-, 2022.

[BHKNA16] Amal Ben Hamida, Mohamed Koubaa, Henri Nicolas, and Chokri Ben Amar. Video surveillance system based on a scalable application-oriented architecture. *Multimedia Tools and Applications*, 75, 2016.

[BKH+17] Mohanad Babiker, Othman Omran Khalifa, Kyaw Kyaw Htike, Aisha Hassan, and Muhamed Zaharadeen. Automated daily human activity recognition for video surveillance using neural network. *2017 IEEE 4th International Conference on Smart Instrumentation, Measurement and Application (ICSIMA)*, 2017.

[BMB21] Brais Bosquet, Manuel Mucientes, and Víctor M Brea. Stdnet-st: Spatio-temporal convnet for small object detection. *Pattern Recognition*, 116, 2021.

[Bro19] Jason Broenlee. *Deep learning for computer vision*. 2019.

[BRZ24] Mohammadreza Bayat, Liu Rongke, and Haleh Zarrini. Utilizing motion direction of video capturing satellites for complexity reduction in video compressor block-matching techniques. 2024.

[Chr20] Walid Chrimni. Comprendre les support vector machines (svm), september 2020.

[Dat23] DataScientest. You only look once (yolo): Tout savoir, 2023.

[EAAM21] Omar Elharrouss, Noor Almaadeed, and Somaya Al-Maadeed. A review of video surveillance systems. *Journal of Visual Communication and Image Representation*, 77, 2021.

[enc] encord. Yolov9: Sota object detection model explained.

[est] estin. estin presentation.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. The MIT Press, 2016.

[geea] Faster r-cnn | ml - geeksforgeeks. https://www.geeksforgeeks.org/faster-r-cnn-ml/.

[Geeb] GeeksforGeeks. Face detection using cascade classifier using opencv-python. https://www.geeksforgeeks.org/face-detection-using-cascade-classifier-using-opencv-python/.

[Geec] GeeksforGeeks. Feature detection and matching with opencv-python. https://www.geeksforgeeks.org/feature-detection-and-matching-with-opencv-python/.

[Geed] GeeksforGeeks. Probabilistic predictions with gaussian process classification (gpc) in scikit learn. https://www.geeksforgeeks.org/probabilistic-predictions-with-gaussian-process-classification-gpc-in-scikit-learn/.

[Geee] GeeksforGeeks. R-cnn | region based cnns. https://www.geeksforgeeks.org/r-cnn-region-based-cnns/.

[HAM21] ABDELAZIZ HAMADI. Real time multi-object tracking using deep learning. 2021.

[Hik] Hikvision. Introduction key feature.

[HS24] Jim Holdsworth and Mark Scapicchio. Deep learning. https://www.ibm.com/topics/deep-learning?, june 2024.

[IBM24a] IBM. What is computer vision? https://www.ibm.com/topics/computer-vision, june 2024.

[IBM24b] IBM. What is machine learning? https://www.ibm.com/topics/machine-learning?, june 2024.

[JK13] Jia Deng Li Fei-Fei Jonathan Krause, Michael Stark. Stanford cars dataset. 3D Object Representations for Fine-Grained Categorization, 4th IEEE Workshop on 3D Representation and Recognition, at ICCV 2013 (3dRR-13). Sydney, Australia. Dec. 8, 2013, 2013. Data source and banner image: https://www.kaggle.com/datasets/jessicali9530/stanford-cars-datasetl.

[KMK16] Ibrahim Kajo, Aamir Saeed Malik, and Nidal Kamel. An evaluation of optical flow algorithms for crowd analytics in surveillance system. In *2016 6th International conference on intelligent and advanced systems (ICIAS)*. IEEE, 2016.

[Lau] Nguyen Truong Lau. Cars Dataset. https://github.com/nguyentruonglau/cars-dataset/. Accessed: 2024-07-04.

[MDP] Mdpi's topic collection on "applications in image analysis and pattern".

[MMdP12] Ester Martínez-Martín and Ángel P del Pobil. *Robust motion detection in real-life scenarios*. Springer Science & Business Media, 2012.

[Mok12] Djamila Mokhtari. Université de montréal département d'informatique et de recherche opérationnelle faculté des arts et des sciences,". *Détection des chutes par calcul homographique*, 2012.

[MS16] Sumati Manchanda and Shanu Sharma. Analysis of computer vision based techniques for motion detection. In *2016 6th international conference-cloud system and big data engineering (confluence)*. IEEE, 2016.

[pap] paperspace. Yolov9: Exploring object detection with yolo model.

[PBJ+23] Arup Kumar Pal, Bhaskar Biswas, Mihir Digamber Jichkar, Adarsh Nandan Jena, and Manish Kumar. Object detection driven composite block motionestimation algorithm for high-fidelitysurveillance video coding. 2023.

[PP21] Shankargouda Patil and Kappargaon S. Prabhushetty. An efficient motion based group level activity recognition for intelligent video surveillance. *2021 Asian Conference on Innovation in Technology (ASIANCON)*, 2021.

[RHGS16] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6), 2016.

[rP] rPetru Potrimba. What is new in yolov9? an architecture deep dive.

[RP13] Rupali S Rakibe and Bharati D Patil. Background subtraction algorithm based human motion detection. *International Journal of scientific and research publications*, 3(5), 2013.

[Spr] Pattern recognition and image analysis. *Journal by Springer*.

[ult] ultralytics. Yolov9 a leap forward in object detection technology.

[UMDS+19] Amin Ullah, Khan Muhammad, Javier Del Ser, Sung Wook Baik, and Victor Hugo C. de Albuquerque. Activity recognition using temporal optical flow convolutional features and multilayer lstm. *IEEE Access*, 7, 2019.

[Ver22] Konstantin Verner. Human detection dataset, 2022. CCTV footage of humans. Available on Kaggle: https://www.kaggle.com/konstantinverner/human-detection-dataset.

[vis] visio.ai. Yolov9: Advancements in real-time object detection (2024).

[YCS17a] Zhuge Yan, Siu-Yeung Cho, and Sherif Shaker. Predictive block-matching algorithm for wireless video sensor network using neural network. *Journal of Computer and Communications*, 05, 01 2017.

[YCS17b] Zhuge Yan, Siu-Yeung Cho, and Sherif Welsen Shaker. Predictive block-matching algorithm for wireless video sensor network using neural network. *Journal of Computer and Communications*, 5(10), 2017.

[YI22] Junzi Yang and Ajune Wanis Ismail. A review: deep learning for 3d reconstruction of human motion detection. *International Journal of Innovative Computing*, 12(1), 2022.

[Yol20] Yole Intelligence. Cameras and computing for surveillance and security 2020, 2020.

Summary

Surveillance systems often face the challenge of managing extensive amounts of footage, much of which is irrelevant, leading to inefficient storage and difficulty in event retrieval. This thesis addresses this issue by proposing an optimized video recording solution that focuses on activity detection. The proposed approach utilizes a hybrid method combining motion detection via frame subtraction and object detection using You Look Only Once model . This strategy aims to record only scenes with human activity, thereby reducing unnecessary footage and optimizing storage usage. The developed model demonstrates superior performance, achieving precision metrics of 0.855 for car detection and 0.884 for person detection, highlighting its effectiveness in enhancing the efficiency of surveillance systems. However, some limitations remain, such as false positives and false negatives in bad weather conditions like powerful winds.

Key Words : activity detection, video surveillance, object detection, YOLOv9, motion detection, recording optimization, Background subtraction , Surveillance system, Optical flow , Machine learning, Deep learning, YOLO, CNN, Faster R-CNN.

Résumé

Les systèmes de surveillance rencontrent souvent le défi de gérer des quantités importantes de séquences, dont une grande partie est inutile, ce qui entraîne une utilisation inefficace du stockage et des difficultés à récupérer les événements pertinents. Cette thèse aborde ce problème en proposant une solution d'enregistrement vidéo optimisée axée sur la détection d'activité. L'approche proposée utilise une méthode hybride combinant la détection de mouvement par soustraction de trames et la détection d'objets avec YOLOv9. Cette stratégie vise à n'enregistrer que les scènes avec une activité humaine, réduisant ainsi les séquences inutiles et optimisant l'utilisation du stockage. Le modèle développé démontre des performances supérieures, atteignant des précisions de 0,855 pour la détection des voitures et de 0,884 pour la détection des personnes, soulignant son efficacité pour améliorer l'efficacité des systèmes de surveillance.Toutefois, certaines limitations subsistent, notamment des faux positifs et des faux négatifs dans des conditions météorologiques défavorables comme les vents violents.

Mots clés : détection d'activité, surveillance vidéo, détection d'objets, YOLOv9, détection de mouvement, optimisation de l'enregistrement, Systèmes de surveillance, Soustraction de fond , Flux optique , Apprentissage automatique , Apprentissage profond, YOLO, CNN .

ملخص

غالبًا ما تواجه أنظمة المراقبة تحديًا في إدارة كميات كبيرة من اللقطات، والتي يكون معظمها غير ذي صلة، مما يؤدي إلى استخدام غير فعال للتخزين وصعوبة في استرجاع الأحداث الهامة. تتناول هذه الأطروحة هذه المشكلة من خلال اقتراح حل لتسجيل الفيديو الأمثل يركز على اكتشاف النشاط. تعتمد الطريقة المقترحة على استخدام طريقة هجينة تجمع بين اكتشاف الحركة عبر طرح الإطارات واكتشاف الكائنات باستخدام $YOLOv9$. تهدف هذه الاستراتيجية إلى تسجيل المشاهد التي تحتوي على نشاط بشري فقط، مما يقلل من اللقطات غير الضرورية ويعزز استخدام التخزين. يظهر النموذج المطور أداءً فائقًا، حيث يحقق دقة ٨٥٥.٠ لاكتشاف السيارات و٤٨٨.٠ لاكتشاف الأشخاص، مما يبرز فعاليته في تحسين كفاءة أنظمة المراقبة. ومع ذلك، هناك بعض النقائص مثل التصنيف الإيجابي الخاطئ والتصنيف السلبي الخاطئ في بعض الحالات ذات الطقس السيء مثل الرياح القوية.

الكلمات المفتاحية اكتشاف النشاط، المراقبة بالفيديو، اكتشاف الأجسام، $YOLOv9$ ، اكتشاف الحركة، تحسين التسجيل. أنظمة المراقبة، الطرح الخلفي، التدفق البصري، تعلم الآلة، التعلم العميق، $YOLO, CNN, FasterR-CNN.$
