HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY

FACULTY OF COMPUTER SCIENCE AND ENGINEERING

COMPUTER ENGINEERING PROJECT

DESIGN AND IMPLEMENT A SMART POLE SYSTEM

MAJOR: COMPUTER ENGINEERING

THESIS COMMITTEE:	**COMPUTER ENGINEERING CLC**
SUPERVISOR(S):	**DR. LÊ TRỌNG NHÂN**
	MR. NGUYỄN THIÊN ÂN
MEMBER SECRETARY:	**MR. NGUYỄN THIÊN ÂN**

Student 1:	LÝ GIA HUY	2053038
Student 2:	VƯƠNG NHẬT ANH	2052855
Student 2:	NGUYỄN TRỌNG DUY	1852296

Comment

. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .

Signature

STUDENT INFORMATION

List of project authors:

1. **Lý Gia Huy** - ID: 2053038
    - Phone number: +84 919 137 649
    - Email: huy.ly2053038@hcmut.edu.vn
2. **Vương Nhật Anh** - ID: 2052855
    - Phone number: +84 586 577 422
    - Email: anh.vuong0110@hcmut.edu.vn
3. **Nguyễn Trọng Duy** - ID: 1852296
    - Phone number: +84 372 025 082
    - Email: duy.nguyen.11281227@hcmut.edu.vn

COMMITMENT

This project was initially inspired by our supervisor's concept. We conducted all the implementation ourselves over the course of the semester. We consulted various open-source projects, articles, and data sources, and you can find references to these resources at the end of the thesis. We take full responsibility for any potential copyright infringements within the thesis. The thesis adheres to the guidelines and requirements established by the Faculty of Computer Science and Engineering at Ho Chi Minh University of Technology and has been completed accordingly.

Sincerely,

Lý Gia Huy
Vương Nhật Anh
Nguyễn Trọng Duy

ACKNOWLEDGEMENT

First and foremost, we would like to express our deepest gratitude to our supervisors, Dr. Lê Trọng Nhân, and Mr. Nguyễn Thiên Ân, for their unwavering support and guidance throughout this project. Both have been instrumental in shaping our journey of knowledge, providing constant presence, heartfelt encouragement, and expert advice whenever needed. Their invaluable guidance, inspiration, and suggestions have been crucial in navigating the challenges of our research. Without their dedicated involvement and assistance at every step, this project would never have been accomplished.

We extend our sincere appreciation to the esteemed educators of the Faculty of Computer Science and Engineering, and to the Ho Chi Minh City University of Technology as a whole. Over the past four years, their unwavering commitment to imparting knowledge has been an enduring source of learning. Their support, encouragement, and insightful ideas have significantly contributed to the successful completion of the project.

Last but certainly not least, we cannot overlook the significance of acknowledging our friends and family. The unwavering love and blessings from our late parents, the care and companionship of friends and acquaintances who kept our spirits high, have all played a pivotal role in guiding us to this significant stage in our lives. We are grateful for their unwavering support during challenging moments, as they motivated us to overcome obstacles and pursue our aspirations.

In closing, we extend our heartfelt wishes for your continued well-being and success in all your noble endeavors.

ABSTRACT

The rapid urbanization of modern cities demands innovative solutions to enhance urban infrastructure, improve public safety, optimize resource management, and foster sustainable development practices. This thesis presents the design, implementation, and evaluation of a Smart Pole System, aimed at addressing these challenges through the integration of advanced Internet of Things (IoT) technologies and Artificial Intelligence (AI). Developed under the auspices of Ho Chi Minh City University of Technology, this project contributes significantly to the emerging field of smart city solutions.

The Smart Pole System is designed as a multifunctional urban infrastructure that integrates several key technologies: environmental sensors, high-definition security cameras, dynamic lighting systems, digital advertising displays, and emergency response features. Each pole is equipped with sensors that monitor air quality, temperature, and humidity, providing real-time data crucial for environmental and health-related decision-making. Security cameras augmented with AI capabilities enhance public safety through real-time surveillance and anomaly detection. Additionally, the poles' adaptive lighting systems use motion detection to manage illumination based on pedestrian and vehicular presence, significantly reducing energy consumption.

Central to the Smart Pole's functionality is its ability to serve as a node within a larger network of interconnected devices, facilitating seamless communication and data exchange. This networked approach allows for synchronized operations across multiple poles, enhancing the system's efficiency and responsiveness. The poles are also equipped with digital screens that display real-time public information, advertisements, and emergency alerts, thereby supporting both economic and safety objectives.

Methodologically, the project adopts a systems engineering approach to design and implement the hardware and software components of the Smart Pole. The evaluation of the system involves both simulation and field testing to validate the functionality, reliability, and effectiveness of each integrated component. Preliminary results demonstrate that the Smart Pole System not only enhances urban infrastructure but also promotes sustainable practices through energy conservation and improved resource management.

In conclusion, the Smart Pole project demonstrates the transformative potential of integrating IoT and AI technologies into urban infrastructure. By addressing critical urban challenges such as safety, energy consumption, and environmental monitoring, this project provides a solid foundation for future smart city developments. Following this study, we will explore expanding the Smart Pole system to other urban areas, adapt it for broader environmental variables, and enhance its data analytics capabilities. This expansion aims to establish a globally scalable model that contributes significantly to the broader goals of sustainable urban development, ultimately paving the way for smarter, more sustainable cities worldwide.

Project keyword: IoT, NBIoT, MQTT, Smart pole, Object tracking, YOLOv9.

Table of contents

List of figures

CHAPTER 1

INTRODUCTION

1.1 Motivation

As cities around the globe continue to grow both in population and complexity, the concept of **Smart Cities** has emerged as a critical solution to address the inherent challenges of urbanization. **Smart Cities** leverage digital technology and the **Internet of Things (IoT)** to improve municipal services, enhance quality of life, and reduce environmental footprint. The aim is to create more sustainable and efficient urban environments through the optimization of city operations and services, such as transportation, energy consumption, and waste management. By doing so, **Smart Cities** not only aim to enhance the urban living experience but also strive to meet the future needs of their growing populations in an environmentally responsible and resource-efficient manner.

In **Viet Nam**, the development of smart cities is a dynamic and evolving field that aligns with the country's rapid urbanization and modernization needs. Since the national policy first mentioned the concept of smart cities in **2016**, significant strides have been made. A key government policy issued in **2018** aims to transform Vietnam's four largest cities - Hanoi, Ho Chi Minh City, Da Nang, and Can Tho - into smart cities by **2025** or **2030**. These initiatives focus on enhancing urban infrastructure through technological solutions that improve energy efficiency, traffic management, security, and overall urban governance.

Ho Chi Minh City, for example, has committed to becoming a smart city by **2025** with strategic plans to implement a shared database, a smart administration center, and enhance information security measures. The city has established an Intelligent Operations Center and is actively developing an open data portal to provide accessible information on public services to its citizens and businesses, which promotes transparency and efficiency.

Viet Nam's approach to smart cities incorporates a comprehensive strategy involving not just technological advancements but also sustainable and social development goals. The ongoing collaborations between Vietnamese cities and international institutions, including educational and research organizations, are crucial in ensuring that these smart city projects are sustainable, innovative, and aligned with global best practices.

Figure 1.1: Thủ Đức City signs an agreement with Hồ Chí Minh City Department of Science and Technology to develop into an innovative urban area and smart city by 2025. (Photo by VNA/VNS)[1]. It aims to promote the use of science and technology and innovations in management, administration and socio-economic development.

In the heart of these technological urban advancements lies the deployment of innovative infrastructure and services designed to improve connectivity, resilience, and public welfare. **Smart Cities** utilize a network of **sensors**, **cameras**, **wireless devices**, and **data centers** to collect and analyze data on everything from traffic patterns to energy use, facilitating real-time responses to city needs. This interconnectedness allows for more informed decision-making and the ability to address issues proactively, thereby enhancing the efficiency of city operations and services. The integration of **renewable energy sources**, **smart grids**, and **electric vehicle charging stations** further exemplifies the commitment of **Smart Cities** to sustainability and environmental stewardship. A pivotal component in the realization of these technologically advanced urban environments is the implementation of "**Smart Poles**." These innovative structures go beyond the traditional role of street lighting, integrating various technologies to enhance connectivity, safety, and efficiency within urban landscapes.

[1] https://vietnamnews.vn/society/1111586/thu-duc-to-develop-into-innovative-hub-smart-city.html

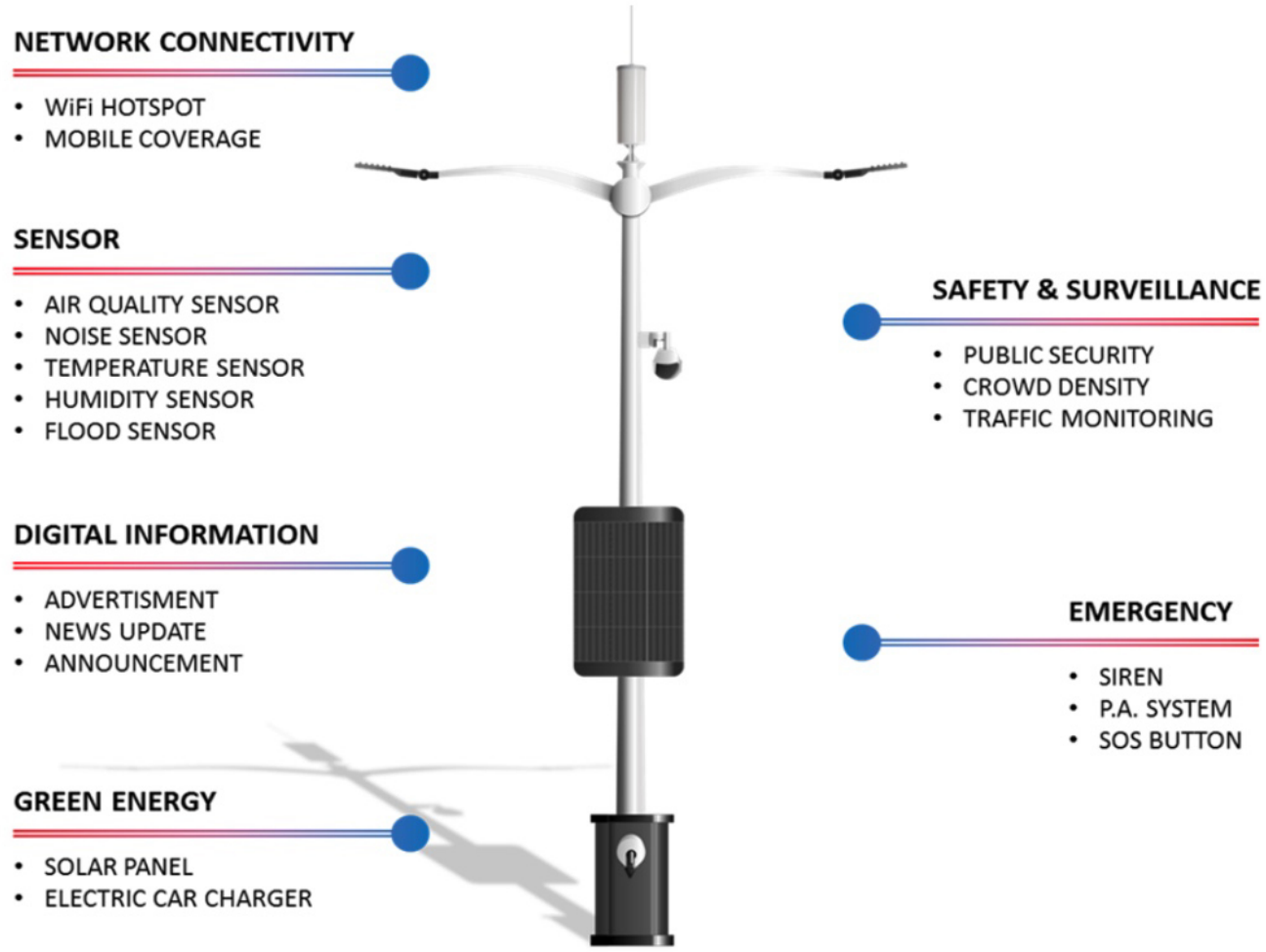


Figure 1.2: This infographic illustrates a multi-functional Smart Pole, designed to enhance urban living through technology. It integrates various features including network connectivity (Wi-Fi hotspot, mobile coverage), an array of environmental sensors (air quality, noise, temperature, humidity, flood), safety and surveillance equipment (cameras for public security, traffic, and crowd density monitoring), digital information displays (advertisements, news updates), green energy components (solar panels, electric car chargers), and emergency response tools (siren, P.A. system, SOS button). (Picture by Telco services) [2]

[2]https://tpworks.com.my/smart-pole/

1.2 Proposed solution

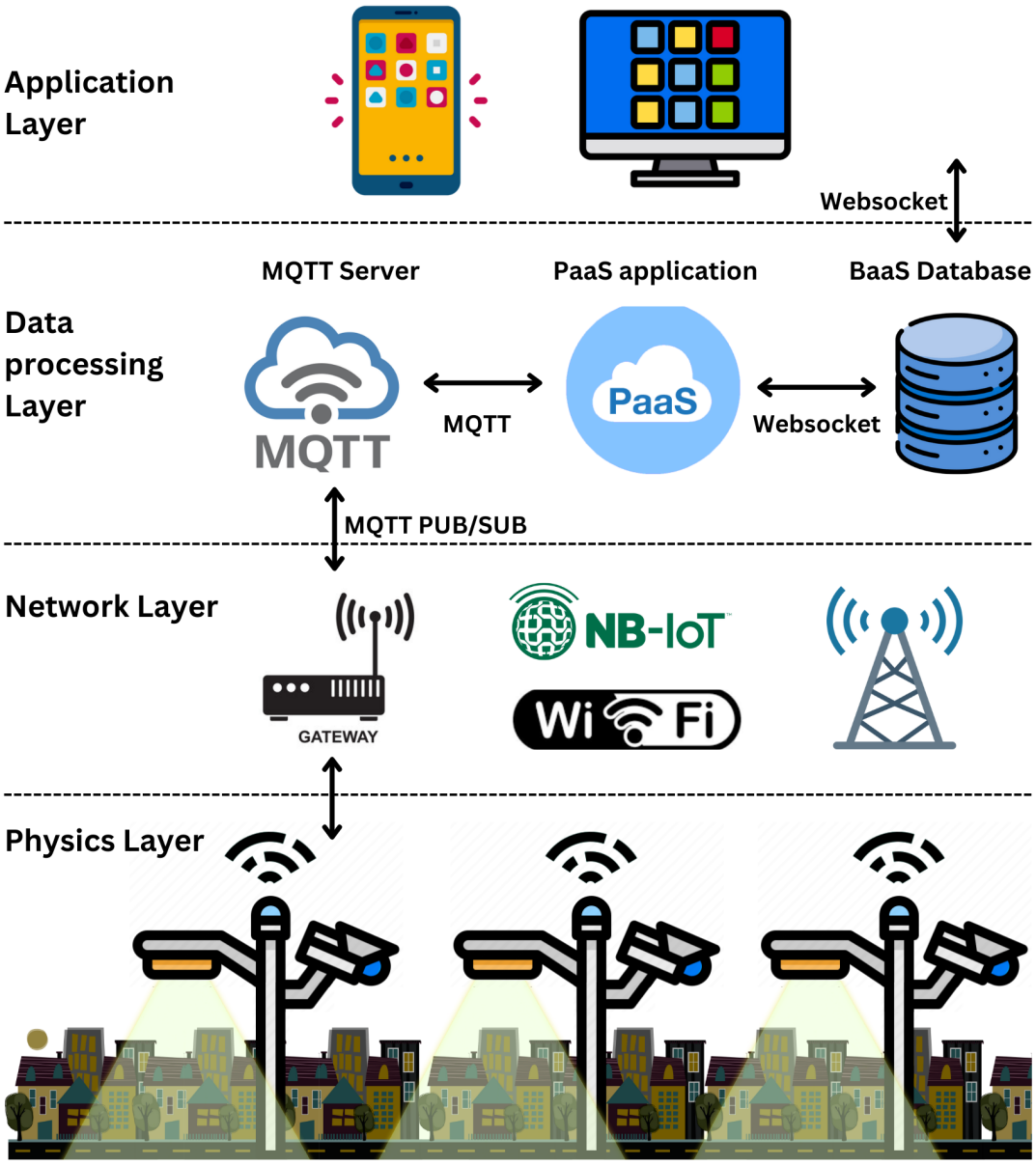


Figure 1.3: This diagram showcases the IoT architecture for managing multiple smart poles across a smart city, incorporating layers from the physical infrastructure to application interfaces. It details communication technologies at the network layer, MQTT protocols at the data processing layer, and user interactions at the application layer.

For the reasons clearly stated in the previous section, **Smart Cities** and **energy demand** have become global issues, including in **Viet Nam**, which is currently following the trend of development and eagerly seeking more technological solutions to contribute to the country's growth. Capturing the general development trend of the world and the need for **smart cities**, we apply the knowledge and theories learned at the university to propose a solution named

Design and Implement a Smart Pole System. With our project, we provide an **IoT** based solution that is not just a product but a **multi-functional, multi-device, multi-platform system**.

Our proposed **Smart Pole** system aims to revolutionize urban infrastructure by integrating advanced **IoT** technologies. These poles will be equipped with various **sensors** to monitor environmental conditions such as **air quality**, **noise levels**, and **weather changes**. Additionally, they will support enhanced connectivity options including **Wi-Fi hotspots** and **mobile data services**, which are critical for ensuring continuous communication in a modern urban setting.

The system also includes safety and **surveillance features** to improve public security. **Cameras** installed on the smart poles will help in monitoring public spaces and managing traffic effectively. In emergencies, the integrated PA system and **SOS buttons** on the poles will provide immediate assistance to the public, thus enhancing safety and quick response capabilities.

Implementing this smart pole system will therefore not only enhance urban life but also pave the way for a sustainable and technologically advanced future, demonstrating our commitment to contributing meaningful solutions to Vietnam's urban development challenges.

1.2.1 Aim of the Study

The primary aim of this study is to design and implement a sophisticated Smart Pole System within the smart city framework, employing cutting-edge Internet of Things (IoT) and Artificial Intelligence (AI) technologies. The project specifically focuses on the following technological advancements:

- **Integration of AI in Surveillance Cameras:** Enhance public security by equipping smart poles with AI-enabled cameras capable of real-time face recognition, traffic monitoring, and incident detection, thus facilitating proactive urban management.
- **Adoption of NBIoT for NEMA Socket Technology:** Utilize Narrowband Internet of Things (NBIoT) technology in NEMA socket-enabled lighting systems. This innovation allows for more reliable and energy-efficient control of street lighting infrastructure, even in challenging urban environments.
- **Development of a Multi-Platform Lighting Control Application:** Implement an application that enables the remote control and scheduling of street lights across various platforms, enhancing operational efficiency and energy conservation in urban lighting

management.

This study aims to demonstrate the practical applications of these technologies in urban settings and evaluate their impact on enhancing the efficiency, safety, and sustainability of city infrastructures.

1.2.2 Significance of the Study

The significance of this study lies in its potential to transform urban areas into more livable, safe, and efficient environments. This project addresses several critical aspects:

- **Enhancing Urban Safety and Security:** By incorporating surveillance and emergency response technologies, smart poles can significantly improve public safety and facilitate faster response times during emergencies.
- **Environmental Monitoring:** Smart poles equipped with environmental sensors can play a crucial role in pollution control and climate monitoring, contributing to healthier urban living conditions.
- **Improved Urban Planning:** Data collected from these smart poles can provide valuable insights into urban dynamics, which can assist city planners and policymakers in making informed decisions to optimize city operations.
- **Promotion of Sustainable Practices:** The integration of renewable energy sources and green technologies in smart poles underscores the commitment to sustainable urban development, aligning with global sustainability goals.

This study will not only demonstrate the technological feasibility of the Smart Pole System but also highlight its broader implications for societal benefits and urban development.

1.2.3 Limitations of the Study

The project is divided into two phases, and this report concludes Phase 1. As a result, we face several limitations:

- Limited Project Duration: The timeframe for this phase of the project is 14 weeks. This includes mechanical work, which means certain implementations may not be fully completed within this phase.
- Short Data Collection Period: The data sampling period for the project is confined to 2 weeks. This duration is insufficient to demonstrate the long-term operational capabilities of the system.

In addition, there are some constraints that the entire project process, including both phases, will encounter:

- Limitations Regarding Sample Product: The prototype developed for sampling purposes is singular. Due to financial constraints, as the cost of the product is sponsored by our project advisor, we are not in a position to independently finance the production of multiple units.
- Geographical Limitations: The geographic scope of our sampling is confined to District 10 of Ho Chi Minh City, Vietnam. Consequently, the findings may not be generalizable or applicable to other regions with different environmental conditions or air quality challenges.

1.3 Overview of the Study

This project is structured into five distinct chapters, each focusing on a crucial aspect of the study:

1. **Chapter 1: Introduction**
   This chapter provides an overview of the project, explaining the reasons for choosing the topic and outlining the content and objectives of the study. It sets the stage for the subsequent chapters by introducing the key themes and issues that the project addresses.
2. **Chapter 2: Background Knowledge**
   The second chapter delves into the basic knowledge and theoretical background necessary for understanding the system. It lays the groundwork for the development of the main system, discussing relevant concepts, principles, and pre-existing research that inform the project.
3. **Chapter 3: Design and Implementation**
   Focused on the core of the project, Chapter 3 details the design and implementation of the system. This is the most critical part of the study, where the practical application of theories and concepts discussed in the previous chapters comes to fruition. It includes the methodologies used, development processes, and technical details of the system's construction.
4. **Chapter 4: Conclusion and System evaluation**
   In this chapter, the findings and outcomes of the project are critically analyzed. It provides a summary of the results, discusses the implications of these findings, evaluates the effectiveness of the system, and reflects on the study's overall achievements and limitations.
5. **Chapter 5: Expansion and Future Development**
   The final chapter explores potential avenues for future research and development. It suggests enhancements and modifications that could improve the system and discusses how the project could be expanded or adapted to other contexts or applications in the future.

CHAPTER 2

BACKGROUND KNOWLEDGE

2.1 Introduction to YOLOv9: Foundation of Object Detection System

2.1.1 Overview of YOLOv9

In the dynamic field of computer vision, the YOLO (You Only Look Once) architectures are renowned for setting benchmarks in real-time object detection. YOLOv9, the latest version, continues this tradition by incorporating state-of-the-art enhancements that boost both its accuracy and processing speed. This section presents an in-depth overview of YOLOv9, emphasizing its critical role in advancing object detection technologies.

YOLOv9 has made significant strides in neural network architecture, training processes, and execution speed, making it exceptionally effective for real-time applications. Notable improvements include a sophisticated feature extraction mechanism, refined anchor box optimization, and the adoption of cutting-edge loss functions. These advancements collectively heighten the model's detection capabilities and operational efficiency.

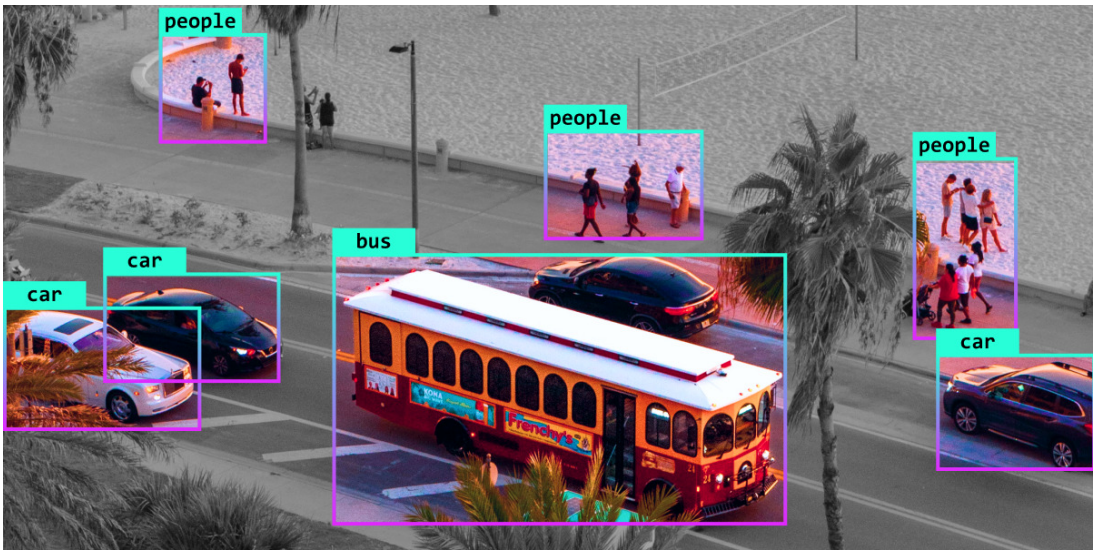


Figure 2.1: Real-time Object Detection Using YOLOv9. This image illustrates the capabilities of YOLOv9 in a real-world setting, showcasing its precision in simultaneously detecting multiple objects such as people, cars, and buses in a complex urban environment. Each object is accurately identified and bounded, demonstrating the model's efficiency and effectiveness in real-time object detection.

Further discussion explores the various applications of YOLOv9, showcasing its adaptability across different scenarios such as automated surveillance, traffic management, and more. The section also delves into the technical enhancements that underpin YOLOv9's superior

performance, focusing on its optimized convolutional layers and improved region proposal algorithms.

This overview of YOLOv9 articulates its architectural nuances, its significance in the continuum of YOLO models, and its impactful contributions to object detection technology. The insights provided highlight the technical excellence of YOLOv9 and its practical implications in solving real-world challenges.

2.1.2 Evolution and Significance

Historical Context and Evolution from YOLOv1 to YOLOv9

The YOLO series has revolutionized the field of real-time object detection since the release of YOLOv1 in 2015 by Joseph Redmon and his colleagues. YOLOv1 introduced a novel approach by integrating both the detection and classification steps into a single model that processes the entire image at once. This method dramatically accelerated the detection process and set a new standard in object detection efficiency.

Subsequent iterations, YOLOv2 and YOLOv3, introduced significant enhancements such as batch normalization, anchor boxes, and the use of finer-grained features, which improved the accuracy and speed of the detections. YOLOv4 and YOLOv5 further advanced these concepts by incorporating features like Cross-Stage Partial networks (CSPNet), mish activation, and enhanced data augmentation, which contributed to better model robustness and performance.

With YOLOv6 through YOLOv8, there was a continued focus on optimizing the network to enhance speed and efficiency, making it suitable for real-time applications on devices with limited computational power. These versions refined the balance between accuracy and speed, making the YOLO family increasingly versatile for various applications.

YOLOv9, the latest version, builds upon these advancements by integrating cutting-edge AI techniques, which improve the model's accuracy and efficiency even further. These include optimizations in feature extraction, advanced anchor box techniques, and the implementation of more effective loss functions that minimize errors during training, enhancing the model's overall performance.

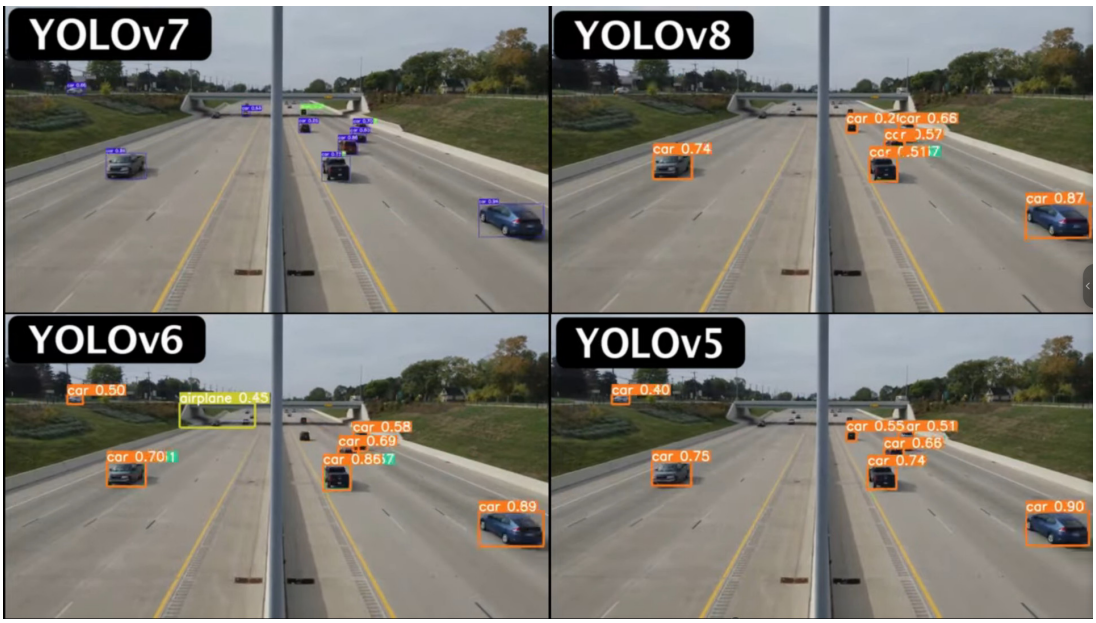


Figure 2.2: The image displays a comparison of object detection performance across four versions of the YOLO model (YOLOv5 to YOLOv8) on a highway scene. Each panel shows how each version detects and labels vehicles with varying levels of accuracy and confidence scores, illustrating the progression in YOLO's detection capabilities over time.

Key Improvements and Their Impact on the Field of Object Detection

Each version of YOLO has introduced key innovations that have significantly impacted the field of object detection. The transition from complex, multi-stage detection systems to a unified, efficient architecture has allowed for real-time applications, fundamentally changing how video is analyzed for surveillance, autonomous driving, and interactive systems.

The introduction of anchor boxes, for instance, improved the accuracy of bounding box predictions, while advancements in network architecture and training processes have made YOLO models quicker and more accurate. These improvements have not only expanded the use cases for YOLO in industrial and commercial applications but also set new benchmarks for future research and development in object detection technologies.

2.1.3 Training and Loss Optimization: Minimizing Information Loss in YOLOv9

YOLOv9 significantly advances beyond its predecessors by directly tackling the crucial problem of information loss during data processing—a challenge often referred to as the information bottleneck, which YOLOv7 did not fully address. By integrating two groundbreaking features, Programmable Gradient Information (PGI) and Generalized Efficient Layer Aggregation Network (GELAN), YOLOv9 specifically targets and effectively minimizes information loss within deep learning models. This strategic enhancement not only

optimizes the processing efficiency but also greatly improves the accuracy and robustness of the model's performance.

Programmable Gradient Information (PGI)

Programmable Gradient Information (PGI) is a sophisticated framework that enhances the training dynamics of deep learning models, specifically targeting the prevalent issues of information loss and error propagation in object detection tasks. The subsequent figure will depict how various methods and architectures, including Path Aggregation Network (PAN), Reversible Columns (RevCol), traditional deep supervision, and Programmable Gradient Information (PGI), manage the flow of information differently.

PGI is composed of three interconnected components:

- **Main Branch**: This component is designed for rapid and accurate object detection, minimizing unnecessary computations. It foregoes additional components during the prediction phase to maintain high processing speeds without increasing computational costs.
- **Auxiliary Reversible Branch**: This branch addresses a significant challenge in deep neural networks: the loss of information as networks become deeper and process data through sequential layers. By employing a reversible architecture, this branch preserves information, ensuring more reliable gradient calculations and more precise parameter updates.
- **Multi-Level Auxiliary Information**: This component enhances the model's ability to make accurate predictions across objects of various sizes by compensating for the potential information loss in the early layers of the network.

During the training of object detection models, shallow features extracted from the early layers of the network are typically utilized, primarily aiding in the detection of smaller objects. However, this often results in a loss of information pertinent to larger objects.

To overcome this challenge, the multi-level auxiliary information technique aggregates details from all target objects, encompassing both small and large objects. This consolidated information is then channeled to the main branch, which is responsible for generating the final predictions. By integrating this technique, the main branch is better equipped to produce precise predictions for objects of varying sizes, significantly enhancing the model's overall effectiveness and accuracy.

Generalized Efficient Layer Aggregation Network (GELAN)

YOLOv9 introduces the Generalized Efficient Layer Aggregation Network (GELAN), an in-

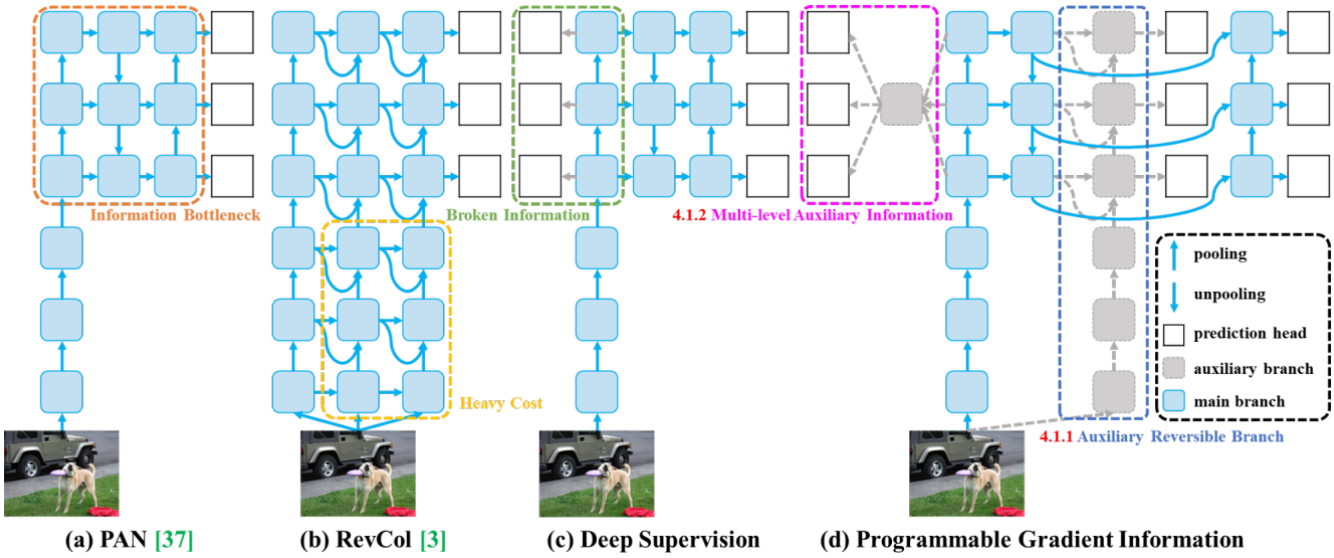


Figure 2.3: PGI Architecture

novative architecture that enhances the principles established by YOLOv7's Efficient Layer Aggregation Network (ELAN) but with added flexibility and efficiency.

GELAN is founded on two pivotal principles:

- **Gradient Path Planning**: This principle ensures that essential information is maintained and flows smoothly throughout the network, effectively preventing any loss during data processing. By optimizing the paths along which gradients travel, GELAN ensures that crucial information is not diluted or lost as it moves through various layers of the network.
- **Reversible Functions**: These functions are integral to GELAN's architecture, allowing the network to preserve important details that might otherwise be discarded during deeper processing stages. Reversible functions enable the network to backtrack and recover any vital information, thus ensuring that the integrity of the data is maintained, leading to more accurate and reliable outcomes.

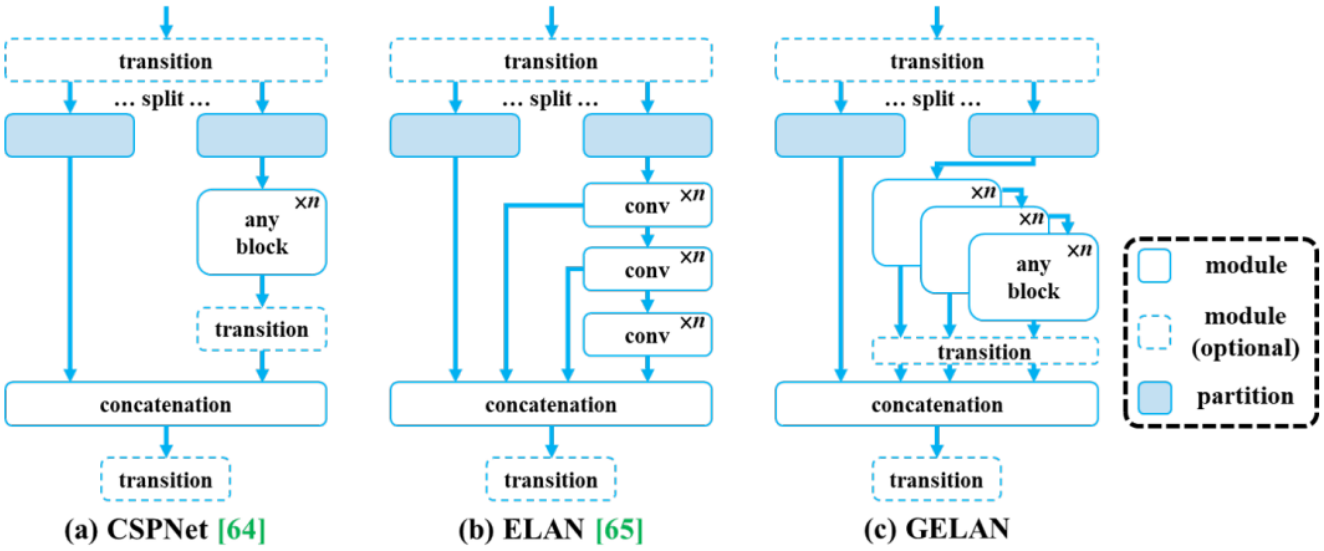


Figure 2.4: GELAN Architecture

Together, while Programmable Gradient Information (PGI) is designed to preserve information within the network, GELAN leverages this preserved information to maximize efficiency and effectiveness in object detection within the YOLOv9 framework. This strategic utilization of information not only enhances the performance of the model but also significantly improves its accuracy in detecting objects across various scenarios.

2.1.4 Experiments and Performance Evaluation

Range of YOLOv9 Models

The YOLOv9 series comprises five models: YOLOv9-n (nano), YOLOv9-s (small), YOLOv9-m (medium), YOLOv9-c (compact), and YOLOv9-e (extended), each with varying parameter counts and performance levels. These models cater to a spectrum of needs, from lightweight applications to those demanding higher performance.

Model	Test Size	APval	AP50val	AP75val	Param.	FLOPs
YOLOv9-T	640	38.3%	53.1%	41.3%	2.0M	7.7G
YOLOv9-S	640	46.8%	63.4%	50.7%	7.1M	26.4G
YOLOv9-M	640	51.4%	68.1%	56.1%	20.0M	76.3G
YOLOv9-C	640	53.0%	70.2%	57.8%	25.3M	102.1G
YOLOv9-E	640	55.6%	72.8%	60.6%	57.3M	189.0G

Table 2.1: YOLOv9 models Performance Comparison

This illustration demonstrates how YOLOv9 models achieve impressive accuracy on the

COCO dataset while maintaining efficiency by using fewer parameters, thus striking a balance between model complexity and performance.

YOLOv9 COCO Benchmarks

The evaluation of YOLOv9 on the MS COCO object detection dataset demonstrated its superior performance over existing real-time detection models that were trained from the ground up. It excelled across multiple metrics, including the number of parameters, computational requirements, and overall accuracy, setting new benchmarks in the field. Notably, YOLOv9 achieved remarkable results by providing higher precision while utilizing fewer parameters and consuming less computational power compared to both lightweight and medium-sized models. This efficiency underscores its robust design and optimization.

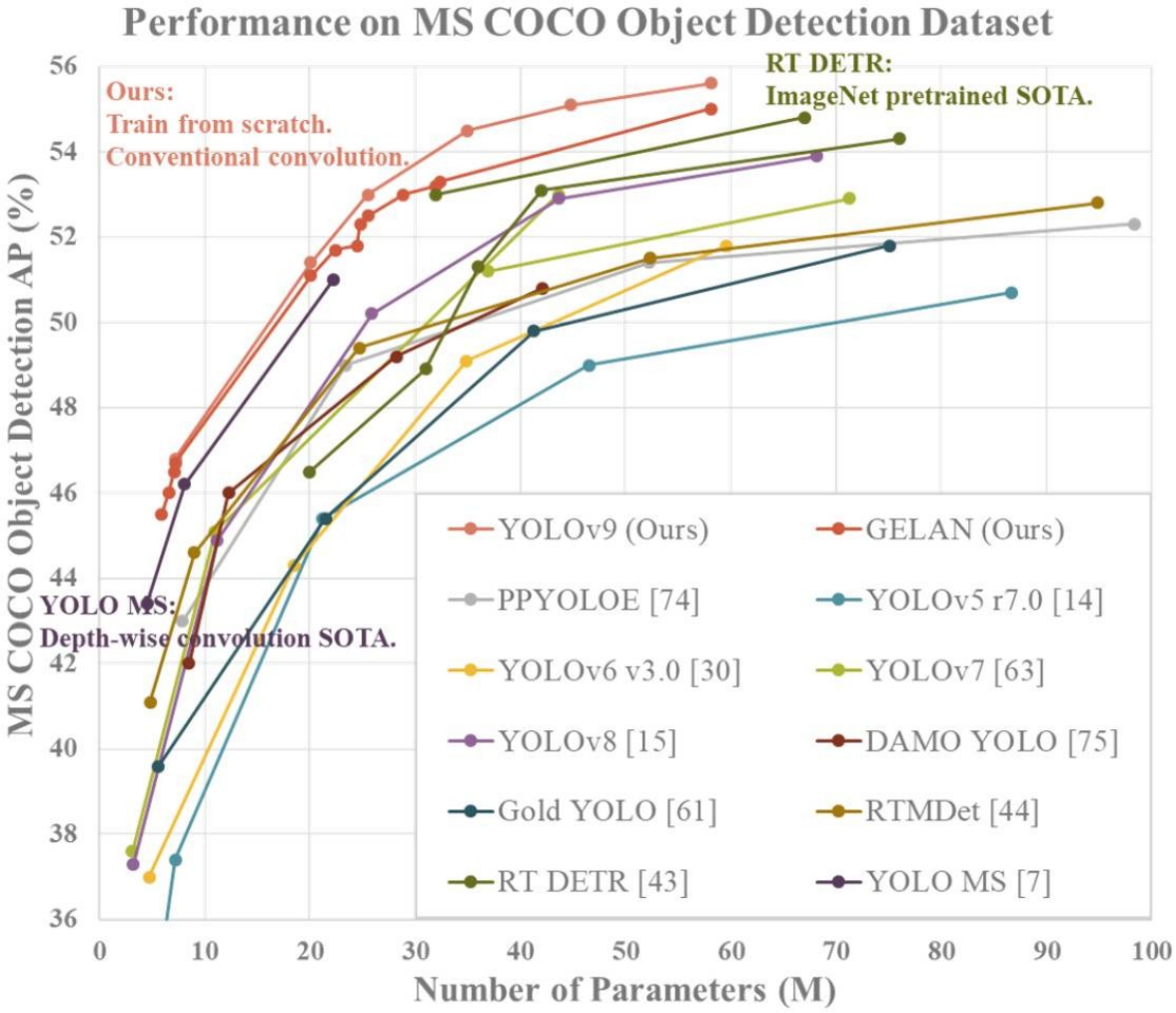


Figure 2.5: Comparisons of the real-time object detectors on MS COCO dataset

Moreover, YOLOv9 also showcased exceptional effectiveness in using its parameters more judiciously than models that rely on datasets pre-trained on ImageNet. This advantage highlights YOLOv9's capability to maximize the impact of its architectural enhancements and

learning algorithms, making it not only more efficient but also potentially more adaptable to a variety of real-world applications where resource constraints are a significant consideration.

2.2 Deepsort Algorithm: Real-time Object Tracking Technique

2.2.1 Introduction to Tracking

Object tracking in deep learning is the process of predicting the positions of objects throughout a video by leveraging both their spatial and temporal features. This is generally achieved through a two-step approach: first, an object detection module identifies and localizes objects in each frame using detectors such as YOLOv4 or CenterNet; second, a motion predictor uses the past information of the objects to forecast their future movements.

The Necessity of Trackers

Trackers are crucial when object detectors fail, for instance, under occlusion or when objects overlap significantly. In such cases, trackers continue to predict the position of objects. Additionally, unlike detectors that only locate objects, trackers assign and maintain unique IDs for each object, ensuring continuity across frames. This ID assignment is essential for applications requiring the monitoring of objects over time, making trackers faster and suitable for real-time applications due to their ability to maintain object continuity without repeatedly detecting them.

Real-World Applications

Trackers have a broad range of applications:

- **Traffic Monitoring**: They help manage traffic flow and detect violations, contributing to systems like Automatic License Plate Recognition.
- **Sports**: Trackers facilitate player and ball tracking, aiding in game analysis and foul detection.
- **Multi-Cam Surveillance**: Using re-identification, trackers can maintain an object's ID across different cameras, enhancing security systems by allowing for continuous tracking of individuals even when they move out of a camera's field of view and re-enter into another's.

Types of Trackers

Trackers are classified into:

- **Single Object Trackers:** These focus on tracking one object despite the presence of multiple objects in the frame. They are particularly fast and are used in applications where one object's movement is of interest.
- **Multiple Object Trackers (MOTs):** These are capable of tracking several objects si-

multaneously, recognizing and maintaining the identities of multiple different objects through various frames.

- **Tracking by Detection:** This method relies on an object detector to identify objects first, then tracks them across frames.
- **Tracking without Detection:** Here, objects are manually initialized in the first frame, and their movements are tracked through subsequent frames, often used in simpler or controlled scenarios.

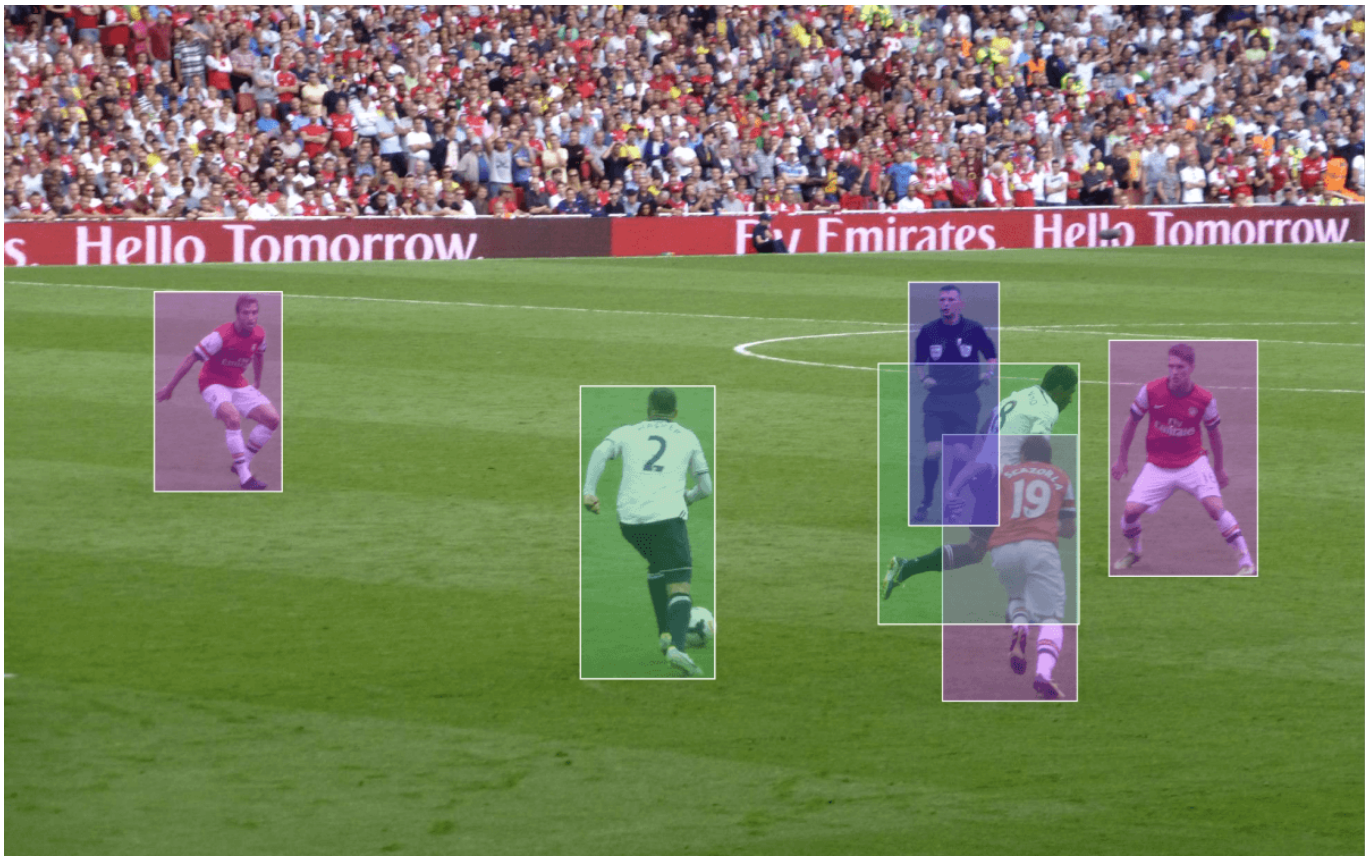


Figure 2.6: A football match scene with players highlighted by bounding boxes, showcasing how object tracking is applied to analyze player movements in sports.

2.2.2 DeepSORT: Advancing Object Tracking

2.2.2.1 Introduction to DeepSORT

After establishing the fundamental importance of tracking in various real-time applications, it becomes evident that while traditional tracking methods have their strengths, they often fall short in complex dynamic environments. DeepSORT (Deep Simple Online and Real-time Tracking) emerges as a pivotal solution in the landscape of object tracking by integrating deep learning techniques to enhance tracking accuracy and robustness. This section

explores how DeepSORT addresses specific challenges faced by traditional tracking systems and significantly improves upon them.

Enhancing Accuracy with Deep Features

DeepSORT extends the SORT algorithm by incorporating a deep neural network that extracts rich feature representations of objects. This advancement allows for more precise discrimination between objects, especially in scenarios where they interact closely or undergo significant appearance changes due to occlusion or varying angles and lighting conditions.

Robust Tracking in Complex Scenarios

By utilizing deep learning, DeepSORT can maintain consistent tracking even when objects are temporarily occluded or merged within the frame. The high-dimensional feature space generated by the neural network provides a robust basis for re-identifying objects once they reappear, ensuring continuity and reliability that is crucial for real-time monitoring and surveillance.

Efficient Real-Time Application

DeepSORT manages to achieve this enhanced performance without sacrificing the speed necessary for real-time application. It cleverly balances computational efficiency and accuracy, making it an ideal choice for scenarios that require both rapid processing and high reliability, such as traffic management and sports analytics.

2.2.2.2 The architecture of DeepSORT

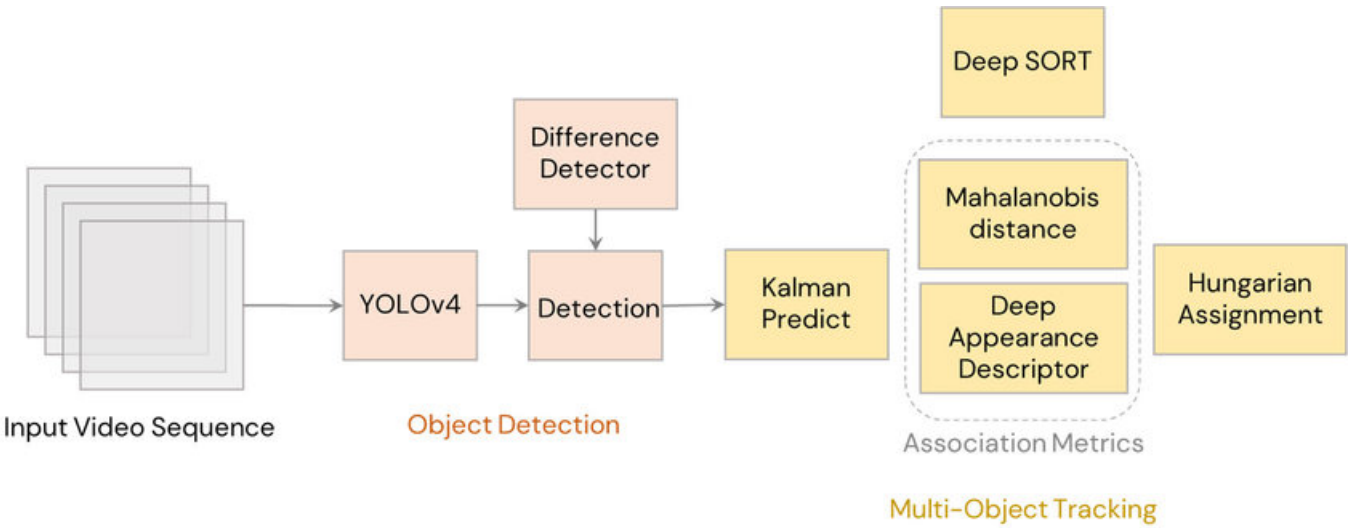


Figure 2.7: Multi-object tracking using YOLOv4 and Deep SORT

The architecture of DeepSORT is structured to enhance the accuracy and robustness of object tracking. It can be dissected into the following steps:

- **Detection:** Utilize an object detection algorithm (such as YOLO) to identify objects in each frame of a video.
- **Feature Extraction:** Extract deep appearance feature vectors for each detected object using a CNN-based feature extractor (e.g., ResNet).
- **Data Association:** Associate detected objects across frames using a matching algorithm (e.g., the Hungarian algorithm) that considers both spatial location and appearance features.
- **Track Management:** Manage tracks by updating the state of each track (e.g., position and velocity) based on associated objects and their appearance features.
- **Track Pruning:** Remove tracks that have not been associated with any objects for a certain number of frames or have low confidence scores.

The deep appearance features employed in DeepSORT are learned during training from a substantial dataset of object images. By incorporating appearance features alongside location and motion information, DeepSORT can handle scenarios where objects may temporarily disappear or occlude each other, resulting in more accurate and robust object tracking.

2.2.2.3 Advantages of DeepSORT

DeepSORT presents numerous advantages that elevate its performance above other object tracking algorithms:

- **Enhanced Tracking Accuracy:** DeepSORT surpasses traditional methods by incorporating deep appearance features alongside spatial and motion information. This integration enables DeepSORT to accurately track objects even in challenging scenarios where objects may momentarily vanish or overlap, ensuring robust and precise object tracking.
- **Multi-Object Handling:** DeepSORT excels in tracking multiple objects concurrently while preserving their individual identities over time, even amidst crowded scenes. This capability is pivotal for applications requiring the monitoring of numerous objects simultaneously, contributing to comprehensive surveillance and analysis.
- **Real-Time Performance:** Engineered for real-time operations, DeepSORT demonstrates swift processing capabilities suitable for diverse applications such as surveillance systems, robotics, and autonomous vehicles. Its ability to deliver timely results makes it indispensable for dynamic environments where prompt decision-making is imperative.
- **Low Computational Cost:** Despite its sophisticated features, DeepSORT maintains

computational efficiency. The utilization of a deep feature extractor does not impose a significant computational burden, ensuring that the algorithm remains computationally lightweight while delivering high-quality tracking results.

- **Seamless Integration with Existing Detectors:** DeepSORT seamlessly integrates with prevalent object detectors like YOLO, facilitating the creation of end-to-end object detection and tracking pipelines. This interoperability streamlines the implementation process, allowing users to leverage existing detection frameworks while benefiting from DeepSORT's advanced tracking capabilities.

By amalgamating deep learning techniques with state-of-the-art tracking methodologies, DeepSORT emerges as a formidable solution that addresses the complexities of modern object tracking tasks, offering unparalleled accuracy, efficiency, and versatility across various domains.

2.2.2.4 Limitations of DeepSORT

Despite its prowess, DeepSORT encounters several limitations that impact its performance:

- **Dependence on Object Detection:** DeepSORT's effectiveness hinges on the accuracy of the underlying object detection algorithm. Inaccurate or inconsistent detections produced by the object detector can compromise DeepSORT's tracking performance, leading to errors in object association and tracking.
- **High Memory Usage:** The utilization of deep appearance features in DeepSORT contributes to high memory consumption, particularly when tracking a large number of objects. This memory overhead can become a limiting factor, especially in resource-constrained environments.
- **Sensitivity to Parameter Tuning:** DeepSORT's performance is intricately tied to the selection of various parameters, including the choice of distance metric for matching, association threshold, and minimum track length. Fine-tuning these parameters for specific applications can be challenging and time-consuming, potentially impeding the algorithm's efficacy.
- **Limited Adaptability to Appearance Changes:** While DeepSORT can effectively track objects with moderate appearance variations, it may struggle when confronted with objects undergoing significant changes in appearance, such as alterations in shape or texture. This limitation can affect tracking accuracy, particularly in dynamic environments with diverse object appearances.

- **Challenges with Occlusions:** Similar to many object tracking algorithms, DeepSORT faces difficulties in tracking objects that are occluded by other objects, particularly when the occluding objects share similar appearances. Occlusions can disrupt object continuity and hinder accurate tracking, posing a challenge for DeepSORT in complex scenarios.

2.2.3 Integration of YOLOv9 and DeepSORT for Superior Object Tracking

The fusion of YOLOv9 for object detection with DeepSORT for object tracking creates a powerful tool capable of addressing complex challenges in real-time tracking scenarios. This section explores how the combination of these two advanced technologies enhances both detection accuracy and tracking stability across various applications, such as surveillance and sports analytics.

2.2.3.1 Detection with YOLOv9

YOLOv9 brings the latest advancements in object detection, characterized by its exceptional speed and accuracy. Utilizing a highly refined convolutional neural network, YOLOv9 excels in detecting objects with high precision across diverse environments. This capability forms the foundation for effective tracking, as accurate initial detection is critical for the subsequent tracking stages.

2.2.3.2 Feature Extraction and Transition to Tracking

Upon detecting objects, the process transitions to tracking, where DeepSORT takes over. DeepSORT extends the capabilities of YOLOv9 by applying a convolutional neural network to extract deep appearance features from each detected object. These features are crucial for distinguishing between different objects and maintaining track consistency, especially in crowded scenes or when objects have similar appearances.

2.2.3.3 Advanced Data Association with DeepSORT

DeepSORT employs sophisticated data association techniques to maintain accurate tracking across frames. It uses the Hungarian algorithm, which considers both the Mahalanobis distance for predicting object motion and the cosine distance for appearance similarity. This method ensures robust track continuity even when objects overlap or become occluded, significantly reducing identity switches and track fragmentation.

2.2.3.4 Seamless Integration for Enhanced Performance

The integration of YOLOv9 and DeepSORT provides a seamless workflow where detection and tracking are handled efficiently. YOLOv9's rapid detection allows DeepSORT to focus on tracking without the burden of detecting objects in every frame, thus optimizing computational resources and enabling real-time performance. This integration is particularly effective in dynamic environments where speed and accuracy are paramount.

2.2.3.5 Practical Applications and Future Prospects

The combination of YOLOv9 and DeepSORT is revolutionizing fields such as automated surveillance, where precise and continuous tracking of objects is necessary. It also plays a crucial role in sports analytics by providing detailed movement patterns of players and objects. Looking forward, this integrated approach has the potential to expand into more complex applications like autonomous driving and complex event recognition, where robust and accurate tracking provides critical inputs for decision-making processes.

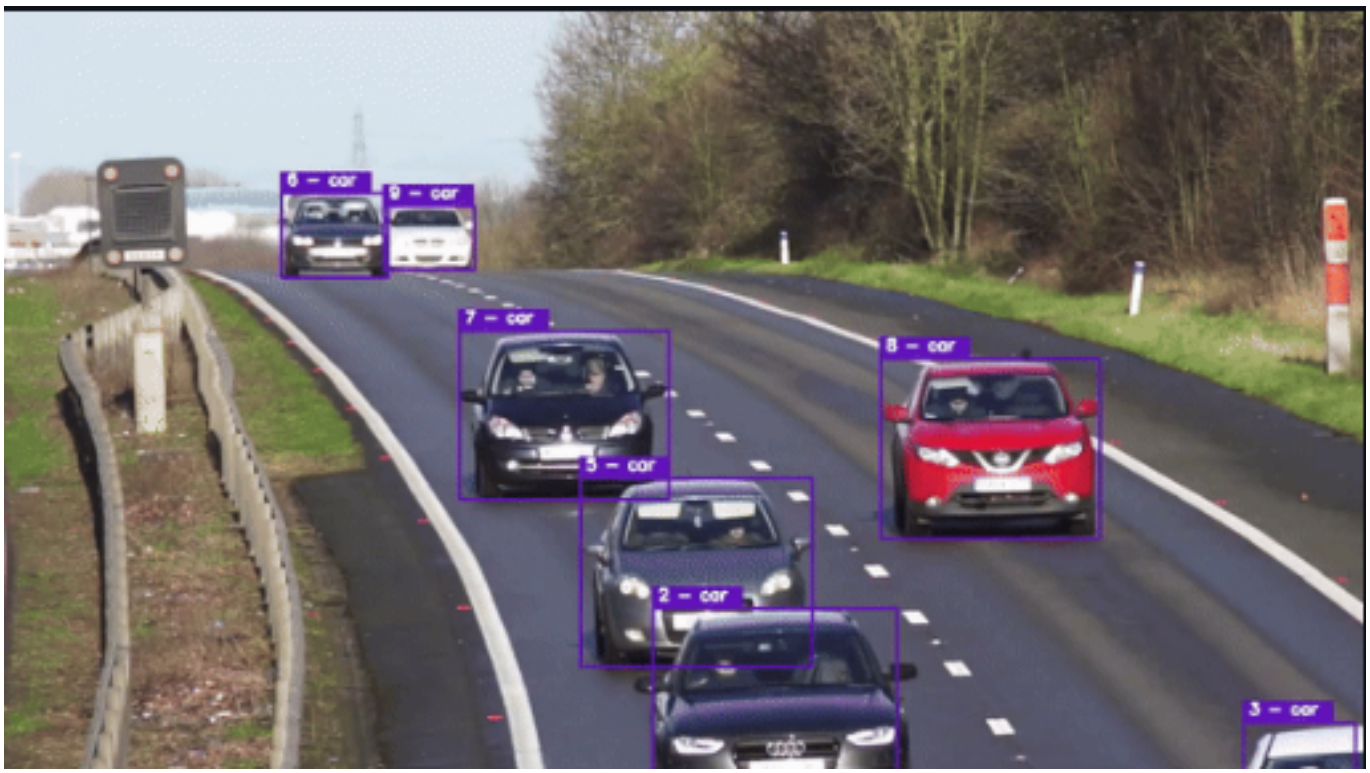


Figure 2.8: The integration of YOLOv9 and DeepSORT excels in tracking vehicles, demonstrating their effectiveness in traffic monitoring scenarios.

2.3 Introduction to Internet of Things

The Internet of Things (IoT) represents a transformative evolution in the digital landscape, where the internet extends beyond the traditional computer networks and connects a diverse range of devices and everyday objects. At its core, IoT is a multi-layered ecosystem that enables seamless interaction between the physical and digital worlds. Based on the IoT system architecture proposed in Timothy Chou's book "Precision"[1], the architecture of an IoT system can be understood through its five fundamental layers, each serving a distinct but interconnected function:

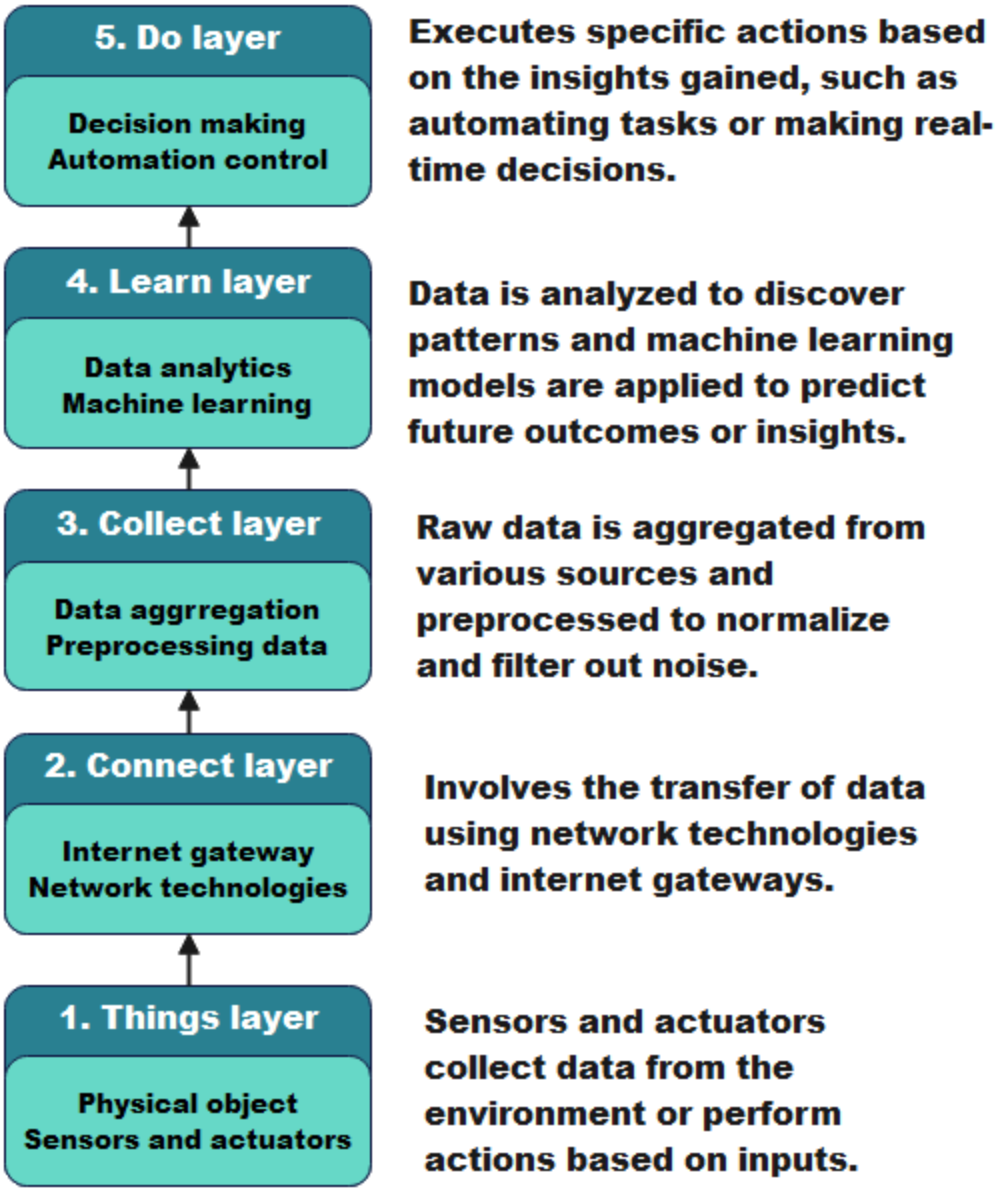


Figure 2.9: IOT Architecture proposed by Timothy Chou

[1]Timothy Chou, *Precision: Principles, Practices and Solutions for the Internet of Things*

- **Things Layer**
  The foundation of any IoT system is the Things Layer, which comprises physical objects equipped with sensors and actuators. These components are essential for collecting data directly from the environment or performing actions based on inputs. This layer serves as the direct interface with the physical world, ensuring the capture of real-world data for further processing.
- **Connect Layer**
  The Connect Layer is crucial for the data transmission collected by physical devices. It uses network technologies and internet gateways to ensure data flows from the Things Layer to the subsequent layers for additional processing. This layer is vital for maintaining robust and secure communication between devices and the rest of the IoT framework.
- **Collect Layer**
  Following data collection, the Collect Layer focuses on data aggregation and preprocessing. It aggregates data from various sources, normalizes it to ensure consistency, and filters out noise to maintain data quality and reliability. This step is essential for preparing the data for deeper analysis and insight extraction.
- **Learn Layer**
  The Learn Layer involves data analysis and the application of machine learning models to processed data. This layer analyzes the data to discover patterns and apply predictive models to forecast future outcomes or derive meaningful insights, enabling adaptive and intelligent behaviors in real-world applications.
- **Do Layer**
  The final layer, the Do Layer, involves executing specific actions based on the insights gained from data analysis, such as automating tasks or making real-time decisions. It demonstrates the practical utility of IoT by applying the derived insights into actionable tasks in various applications.

Fundamentally, IoT represents an advanced network that seamlessly connects the physical with the digital, allowing devices to interact, make decisions, and execute tasks with little human oversight. This cutting-edge technology has extensive applications in multiple sectors, such as home automation, industrial monitoring, healthcare, and environmental observation. Understanding the complex functionality of IoT requires an examination of its essential components and their roles within the broader ecosystem. Each component, from

the devices themselves to the user interfaces, plays a crucial role in the integrated operation of IoT systems. Below, we outline the main components of IoT architecture and explain how they work together to enable a wide range of applications across different sectors:

2.3.1 IoT Devices

IoT devices are at the core of the Internet of Things ecosystem, equipped to sense, process, and communicate data across networks. These devices can range from simple sensors and actuators to complex embedded systems that integrate microcontrollers or microprocessors, allowing them to perform sophisticated computations.

A typical IoT device is built around an embedded system, which is essentially a combination of hardware and software designed to perform specific functions. The main components of an IoT device include:

- **Microcontroller or Microprocessor:** The brain of the device, capable of processing data, executing commands, and managing device operations based on the embedded software.
- **Transducers:** These are sensors and actuators. Sensors collect environmental data by detecting changes in their surroundings, such as temperature, humidity, motion, or light. Actuators, on the other hand, perform actions based on the processed data, like opening a valve, turning on a light, or adjusting a thermostat.
- **Communication Module:** This component is crucial for connectivity. It allows the device to connect to the internet or other devices via various communication protocols such as Wi-Fi, Bluetooth, Zigbee, or cellular networks. The choice of communication technology depends on the range, power consumption, and data transmission requirements of the application.
- **Power Supply or Battery Management:** Since many IoT devices are placed in remote or hard-to-reach locations, efficient power management is critical. This could involve batteries, solar power, or other forms of energy harvesting, depending on the device's environment and usage.

These devices collect data continuously from their environment, which is then processed either locally or sent to a gateway or directly to the cloud for further analysis. The ability of these devices to operate autonomously with minimal human intervention is what makes IoT transformative. They can adjust to new inputs and conditions in real-time, making them integral to applications ranging from home automation and industrial monitoring to smart

cities and healthcare.

IoT devices are also designed with security considerations in mind, as they often handle sensitive or critical data. Security features might include encryption, secure boot, and regular firmware updates to protect against vulnerabilities.

In summary, IoT devices are dynamic and adaptable components that act as the interface between the physical and digital worlds. Their design and capabilities are continually evolving to meet the demands of increasingly sophisticated IoT applications.

2.3.2 Gateway IoT

IoT devices connect to the internet using a gateway, which serves as a conduit for data from sensors to the cloud. Typically, the gateway collects local device data and transmits it to the cloud where it's processed and displayed to users through web applications. A gateway may be a specific hardware device or a software solution, facilitating connections among various data sources and destinations. Communication technologies like WiFi, Bluetooth Low Energy (BLE), ZigBee, Z-wave, LoRa, NB-IoT, and LTE-M enable IoT devices to connect to a gateway, which then provides internet access and possibly supports device-to-device communications.

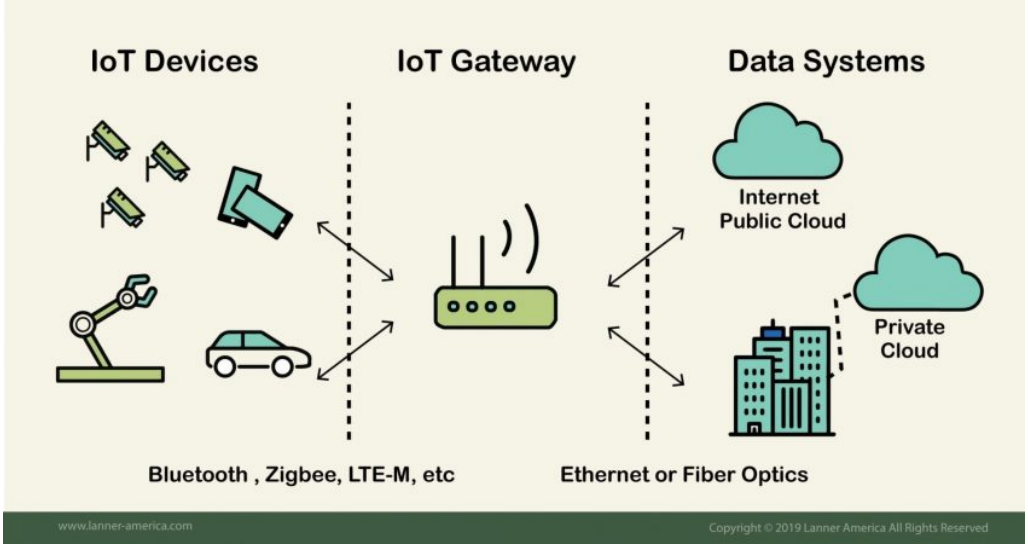


Figure 2.10: This diagram illustrates the connectivity framework of a typical IoT system, featuring various IoT devices (such as sensors and vehicles) that communicate through protocols like Bluetooth, Zigbee, and LTE-M to an IoT Gateway. The gateway then facilitates data transmission over Ethernet or Fiber Optics to centralized data systems, including both Internet Public Cloud and Private Cloud infrastructures. This setup enables efficient data flow and integration across different platforms in an IoT ecosystem.

2.3.3 Cloud and Data Processing

The sensor data must be stored in the cloud. In simpler IoT setups, the cloud might primarily function as a network router between devices. In more complex configurations, it integrates closely with the devices to manage and process raw data. The processed data can be transmitted back to the IoT devices through the gateway or to users via some form of user interface.

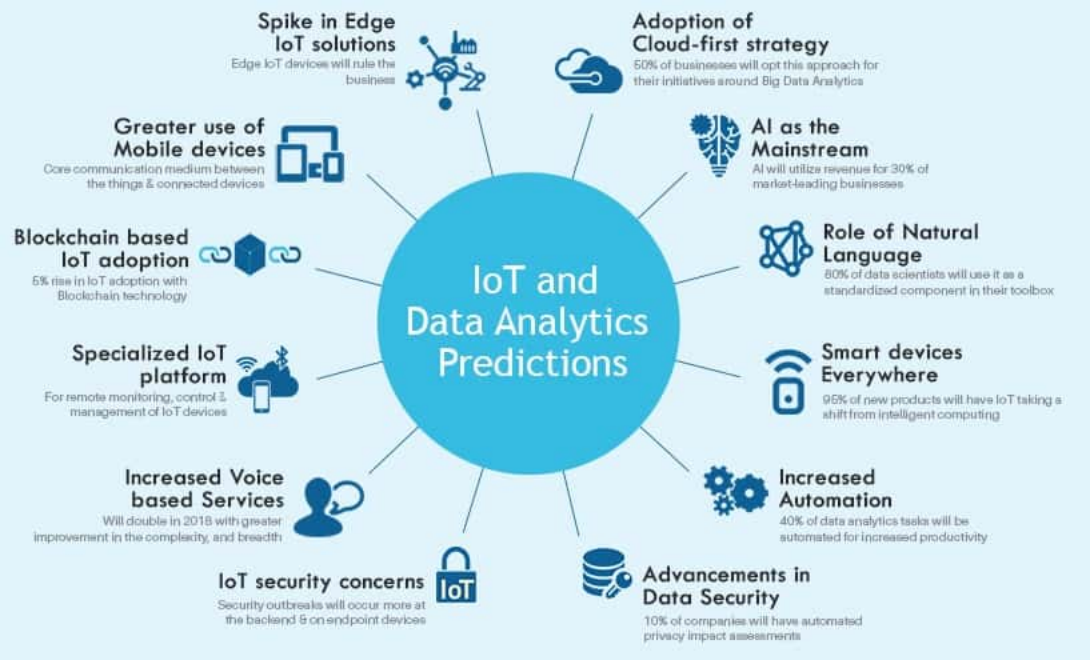


Figure 2.11: Infographic illustrating key trends and predictions in IoT and Data Analytics for the near future, highlighting advancements in mobile technologies, edge solutions, AI, natural language processing, and security. The graphic emphasizes the integration of IoT with blockchain, increased automation, and the widespread adoption of smart devices across various sectors

2.3.4 User Interface

The user interface is pivotal for enabling interactions between humans and IoT devices. It allows users to easily configure devices and access processed data. The user interface, consisting of screens, buttons, icons, and forms, enhances user interaction with the system, as seen in various software and applications on computers and smartphones.

2.3.5 Application of IoT

The significant applications of IoT span military operations, solid waste management, smart metering, healthcare, transportation, vehicular systems, precision agriculture, pilgrim mon-

itoring, consumer asset tracking, home automation, disaster response, environmental monitoring, smart grids, and surveillance. Furthermore, IoT continues to show vast potential for adoption in additional domains.

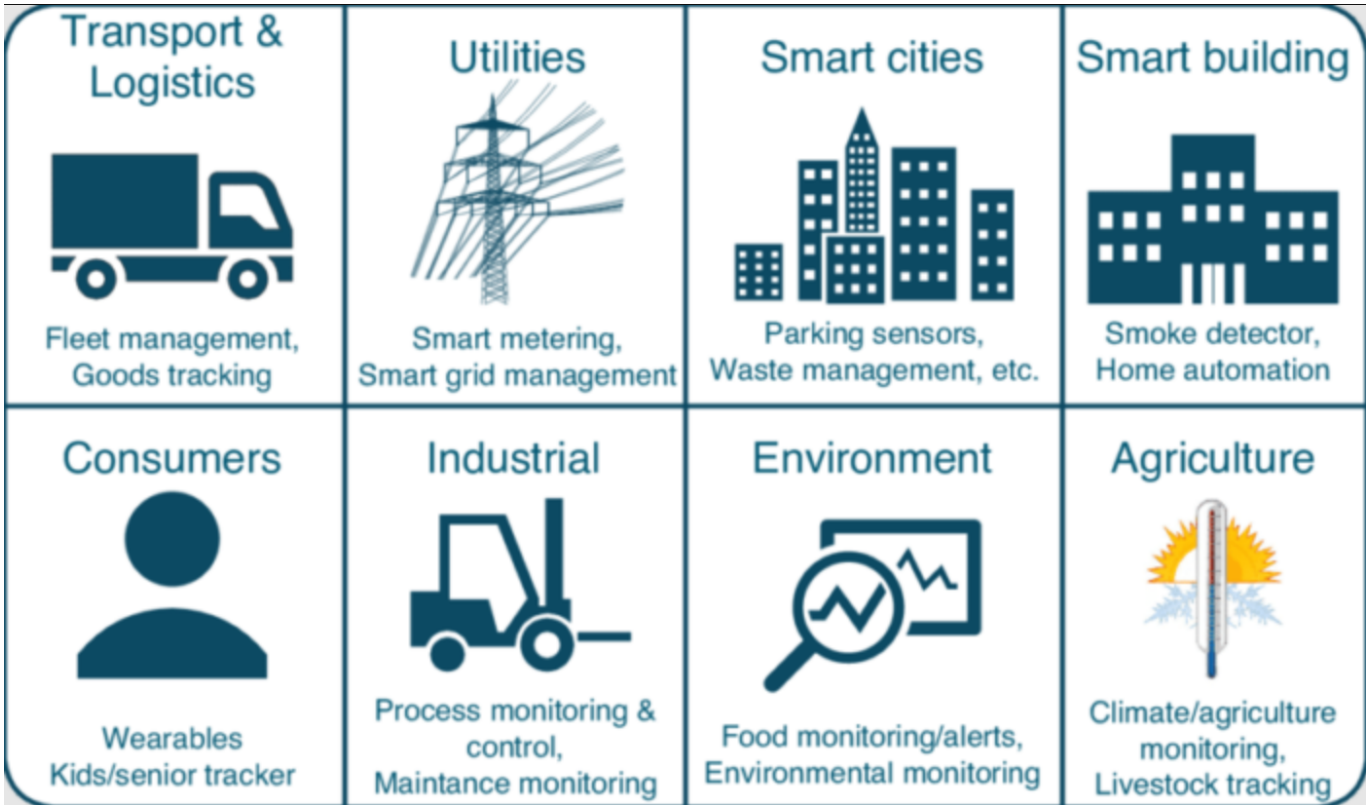


Figure 2.12: This diagram categorizes various IoT applications into sectors such as Transport & Logistics, Utilities, Smart Cities, Smart Building, Consumers, Industrial, Environment, and Agriculture. Each category is associated with specific use cases like fleet management, smart metering, parking sensors, and climate monitoring, demonstrating the extensive integration of IoT technologies in enhancing sector-specific operations.

2.4 End to End Communication based MQTT Protocol

MQTT stands for MQ Telemetry Transport, defines a protocol for IoT devices to publish and subscribe to data over the Internet. Widely used for messaging and data exchange in IoT and industrial IoT (IIoT), MQTT employs an event-driven approach and the publish/subscribe (Pub/Sub) pattern. Communication between the sender (Publisher) and receiver (Subscriber) occurs via Topics, and their connection is managed by the MQTT broker. The broker filters and distributes incoming messages to Subscribers, ensuring effective communication.

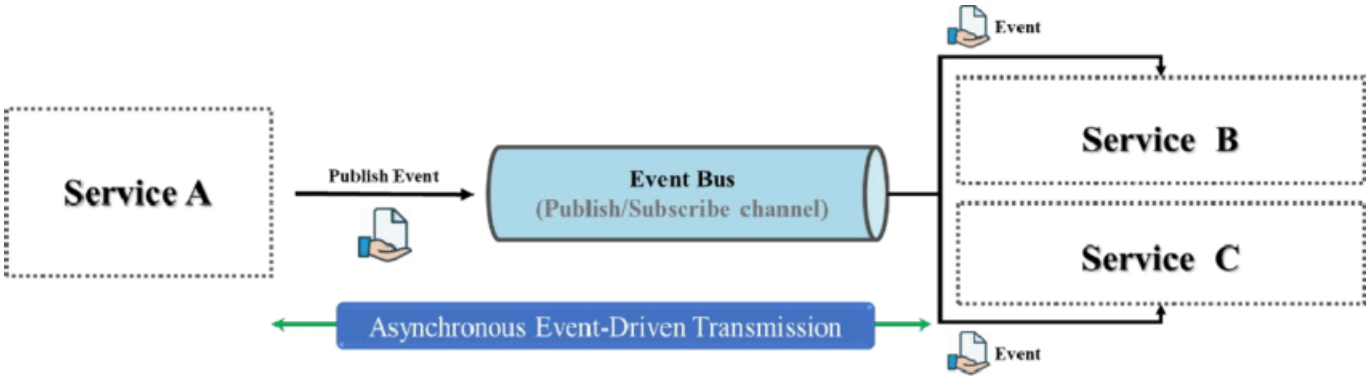


Figure 2.13: MQTT operates on a publish-subscribe model, which is an alternative to the traditional client-server model where each client communicates directly with an endpoint. In MQTT, clients publish messages to a topic, and other clients subscribe to receive messages from those topics. This decouples the producers of data from the consumers of data, thereby enhancing message routing and scaling capabilities.

2.4.1 Role of MQTT Broker

The central component of the MQTT protocol is the MQTT broker. The broker is responsible for managing the transmission of all messages between the sender (publisher) and the receiver (subscriber). It handles the connection, session, authentication, and also the authorization of clients, making sure messages are securely and reliably delivered to the correct destinations.

2.4.2 MQTT Packet Format

MQTT works over IANA registered TCP port numbers 8883 and 1883. The port number 8883 is used for using MQTT over SSL/TSL, and 1883 is used for non-TLS communications.In MQTT, MQTT brokers play a major role. A client otherwise called as a publisher usually sends its messages to a broker (MQTT Server) on different topic-tags and consumers

otherwise known as receivers subscribe to these topics to receive the messages.

A plaintext MQTT message has a fixed-length header of 2 Bytes, an optional message-length header, and a message payload header. The first eight bits of MQTT TCP Packet Format is used as fixed MQTT header, where the first four bits represent the Message Type, the fifth bit is used for the duplicate (DUP) flag, sixth and seventh bits are used for the QoS (Quality of Service) level, the eighth bit is used for “retain" message service. The second 8 bits are reserved for the variable length header, and the optional header which can be used for TLS payloads.

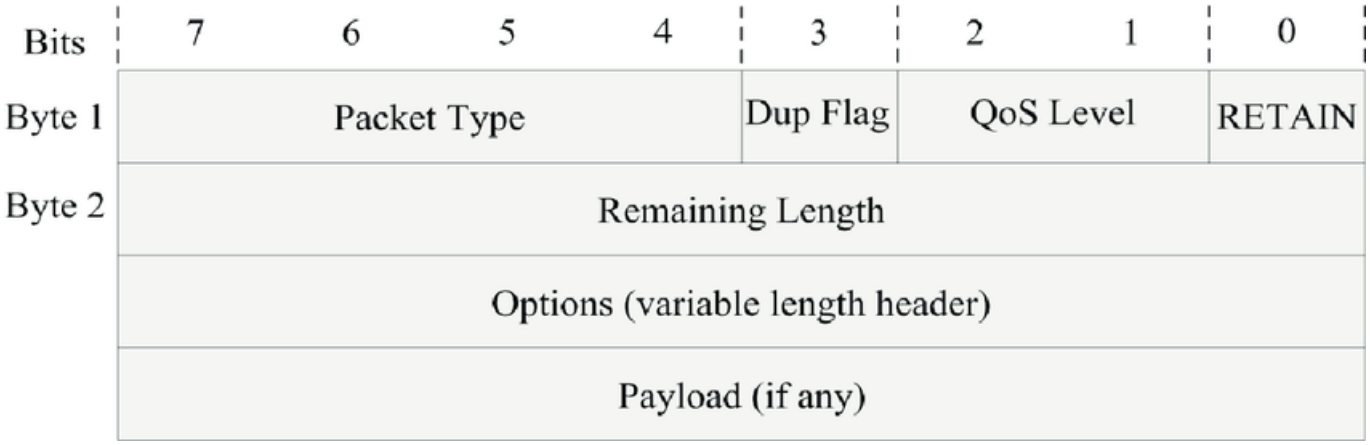


Figure 2.14: MQTT frame format of CONNECT packet

2.4.3 Quality of Service Levels

MQTT offers three levels of Quality of Service (QoS) to define the guarantee of message delivery:

- **QoS 0** (At most once) - where messages are delivered according to the best effort of the underlying network. No confirmation is given on delivery, and the message is not stored and retransmitted by the sender.
- **QoS 1** (At least once) - where messages are assured to arrive but duplicates can occur.
- **QoS 2** (Exactly once) - which provides the highest level of assurance where each message is received only once by the counterpart.

2.4.4 Advantages of MQTT

The design of MQTT includes several features that make it highly suitable for IoT applications:

- **Lightweight Protocol** - Minimal bandwidth usage, efficient battery use, and reduced data packets size.
- **Efficient Distribution** - Efficient mechanisms to distribute messages to multiple clients.
- **Reliable Message Delivery** - Offers message delivery assurance even in unreliable networks.
- **Secure Communication** - Supports secure communication mechanisms such as TLS/SSL to encrypt messages.

2.4.5 Applications of MQTT

Due to its efficient and flexible nature, MQTT is widely used in various applications including smart home devices, automotive, telecommunications, and more. It facilitates effective communication between devices and serves as a critical component in the architecture of Internet of Things.

In conclusion, MQTT provides a robust and effective protocol for IoT communication, ensuring that data flows efficiently between devices and applications, even in challenging network environments.

2.5 NBIoT Efficient Long-Range Connectivity

2.5.1 What is NBIoT?

Narrowband Internet of Things (NB-IoT) is a telecommunications technology that is part of the global standards defined by the 3rd Generation Partnership Project (3GPP) under the LTE Advanced Pro specifications. It is engineered specifically for connecting a large number of devices in a network with very low power consumption and deep penetration. Unlike standard mobile networks, NB-IoT operates in a limited frequency band which enables it to connect devices under challenging conditions, such as underground or in remote areas.

NB-IoT supports efficient connection of devices over long ranges, making it ideal for applications that require only sporadic data transmission, such as sensor reporting, meter reading, and status checks. The protocol is designed to handle small amounts of infrequent data from devices that need a long battery life and can benefit from a simple and powerful connection framework.

Technical aspects of NB-IoT include:

- **Low Power Consumption:** Devices connected via NB-IoT can have battery lives that extend up to 10 years, due to its minimal energy requirements.
- **Cost-Effectiveness:** The technology reduces the cost per connection because it simplifies the device and network architecture.
- **Excellent Coverage:** Enhanced indoor and underground coverage is possible with a range increase of about 20 dB compared to GSM networks.
- **High Connection Density:** Supports over 100,000 connections per square kilometer.

2.5.2 NBIoT in Vietnam

The adoption of NB-IoT in Vietnam reflects a broader trend within Asia, where countries are embracing this technology to bolster their IoT infrastructures and develop smarter, more connected environments. Vietnam's approach to NB-IoT deployment is strategic, focusing on sectors where IoT can have the most immediate and profound impact, such as agriculture, industrial automation, and smart cities.

The implementation of NB-IoT by Viettel, Vietnam's largest telecommunications company, is aimed at supporting a variety of applications that contribute to smart city initiatives, such as smart parking, air quality monitoring, and smart metering. These developments align with

Viettel's vision of transforming urban management and services through advanced telecommunications and IoT technologies. Viettel's commitment to these technologies is reflected in its inclusion among the first 50 companies worldwide to successfully deploy an NB-IoT network, as recognized by the Global System Mobile Association (GSMA).

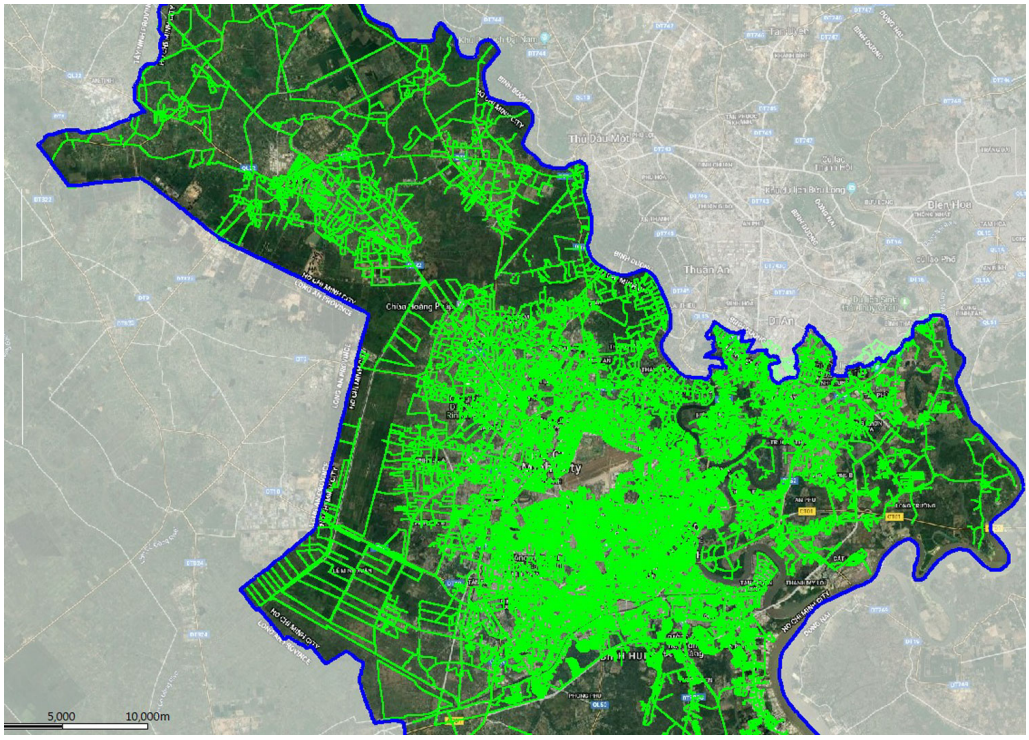


Figure 2.15: Viettel's NB-IoT coverage map in Ho Chi Minh City (Photo: Dieu Linh) [2]:Viettel has made significant strides in deploying Narrowband Internet of Things (NB-IoT) technology across Vietnam. As of 2019, Viettel had successfully established the first commercial IoT network in Vietnam, distinguishing itself by deploying over 1,000 NB-IoT stations to cover the entirety of Ho Chi Minh City. This deployment is part of a broader strategy to enhance IoT connectivity across major cities, including Hanoi, and gradually across the nation.

2.5.2.1 Government Initiatives and Partnerships

Vietnam's government has identified NB-IoT as a key technology in its "National Digital transformation program to 2025 with a vision to 2030." Significant investments are being made in partnerships with leading telecom providers to create a robust infrastructure capable of supporting wide-scale IoT deployments. Initiatives such as the Smart City project in major urban centers like Hanoi, Ho Chi Minh City, and Da Nang are foundational to this effort.

[2]https://dangcongsan.vn/kinh-te/viettel-phu-song-cong-nghe-nbiot-toan-tp-ho-chi-minh-534976.html

2.5.2.2 Telecommunications Infrastructure

Leading Vietnamese telecom operators like Viettel, Vinaphone, and Mobifone have started rolling out NB-IoT networks across the country. Viettel, for example, has launched its NB-IoT network in 30 provinces, with plans for national coverage by the end of the year. This rollout is supported by the development of NB-IoT modules and devices that are tailored to local market needs and conditions.

2.5.2.3 Applications in Key Sectors

In agriculture, NB-IoT is being used to enhance traditional practices through smart agriculture solutions. Sensors deployed in fields collect data on soil moisture and nutrient levels, which is then analyzed to optimize irrigation and fertilization. In industrial settings, NB-IoT facilitates the monitoring of equipment to predict failures before they occur, thereby reducing downtime and maintenance costs.

In urban management, NB-IoT is integral to developing smart city infrastructure, managing everything from traffic and waste collection to energy use in buildings. The technology's ability to provide real-time data supports dynamic decision-making and improved public services.

2.5.2.4 Challenges and Future Directions

Despite its promising deployment, NB-IoT in Vietnam faces challenges such as spectrum allocation, device compatibility, and ensuring interoperability across different networks and platforms. Future directions involve addressing these challenges, enhancing the security of IoT devices, and continuing to innovate in NB-IoT applications to maintain sustainable growth and leverage IoT's full potential to transform society.

2.5.2.5 Research and Development

Research institutions and universities in Vietnam are actively engaged in developing new applications and improving NB-IoT technologies. These efforts are crucial for advancing the technology and ensuring it meets the specific needs of the Vietnamese market, thereby supporting the country's vision of becoming a leading hub for IoT innovation in the region.

In conclusion, NB-IoT holds substantial promise for Vietnam, driving not only technological innovation but also societal advancements by enabling a more connected and efficient environment. The continued development and deployment of NB-IoT technology are expected to play a critical role in Vietnam's digital transformation journey.

2.6 Cloud-hosted database

A cloud database is a database that typically runs on a cloud computing platform, and access to the database is provided as-a-service[**?**]. It allows users to store and manage their data in the cloud rather than on local servers or personal computers. Here are some key theoretical aspects of cloud-hosted databases:

- **Accessibility and Scalability**
  One of the primary benefits of a cloud-hosted database is its accessibility. Users can access their databases from anywhere in the world, provided they have internet connectivity. This enhances collaboration among teams that are geographically dispersed. Moreover, cloud databases offer scalability, allowing the database to grow in size or performance capabilities as the user's needs expand, often with minimal downtime.
- **Cost-Effectiveness**
  Cloud-hosted databases can be more cost-effective than traditional on-premises databases. They typically operate on a pay-per-use model, which means that users only pay for the storage and processing they actually use. This can significantly reduce the costs of data management, particularly for small to medium-sized businesses or projects with variable demands.
- **Performance and Reliability**
  Cloud providers invest heavily in maintaining high-performance computing resources and redundant infrastructure. This means that cloud-hosted databases can deliver high availability, with many services offering uptime guarantees. Data is replicated across multiple locations, which contributes to both the reliability and the disaster recovery capabilities of the service.
- **Security and Compliance**
  Security is a critical consideration for cloud-hosted databases. Providers generally offer a suite of security features, including firewalls, encryption, and access controls, which protect against unauthorized access and data breaches. Furthermore, many cloud providers are compliant with various regulations and standards, which is vital for users who operate in regulated industries.

There are several types of cloud-hosted databases, including:

- SQL Databases: Also known as relational databases, they use structured query language (SQL) for defining and manipulating data. Examples include MySQL, PostgreSQL, and Microsoft SQL Server.
- NoSQL Databases: These are useful for unstructured or semi-structured data and do not require a fixed schema. They are often chosen for their flexibility and scalability. Examples include MongoDB, Cassandra, and Couchbase.
- NewSQL Databases: These databases combine the scalability of NoSQL systems with the ACID (Atomicity, Consistency, Isolation, Durability) guarantees of SQL databases.
- Data Warehousing Services: Designed for analytics and reporting, these services can handle large volumes of data and complex queries. Examples include Amazon Redshift and Google BigQuery.

In summary, cloud-hosted databases provide a flexible, scalable, and often more cost-effective alternative to traditional database systems. They are integral to many modern web applications and services and are a key component of the overall cloud computing paradigm.

2.6.1 Platform as a Service (PaaS)

Platform as a Service (PaaS) is a cloud computing model that provides a platform allowing customers to develop, run, and manage applications without the complexity of building and maintaining the infrastructure typically associated with developing and launching an app. PaaS can be delivered in two ways: as a public cloud service from a provider, where the consumer controls software deployment with minimal configuration options, or as a private service (software or appliance) inside the firewall.

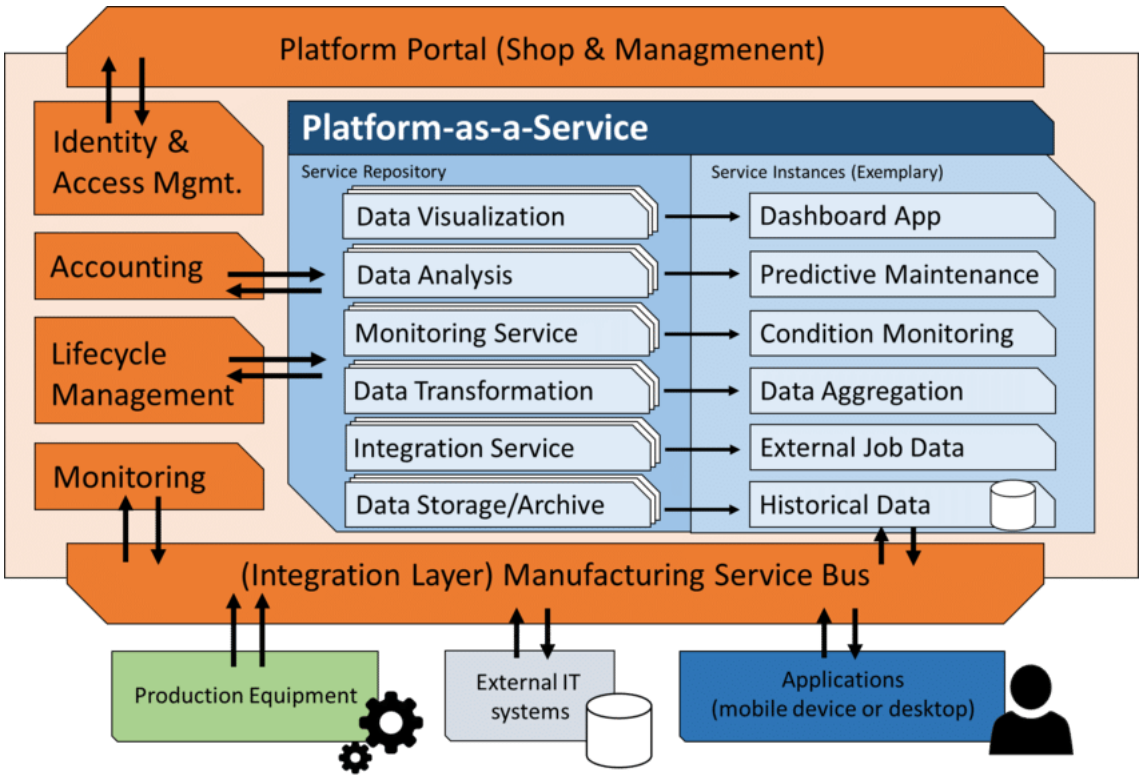


Figure 2.16: PaaS platform architecture

2.6.1.1 Characteristics of PaaS

PaaS services are hosted in the cloud and accessed by users via their web browser. Key characteristics include:

- **Simplified Development Environment**: Developers can focus on building software without worrying about operating systems, software updates, storage, or infrastructure.
- **Scalability**: PaaS provides scalability which can adjust to changes in demand without the need for physical changes to the infrastructure.
- **Multi-tenancy**: Multiple users can access a single piece of hardware through virtualization technology.
- **Access to a Variety of Services**: These services can include team collaboration, web service integration, database integration, security, scalability, storage, persistence, state management, application versioning, and more.

2.6.1.2 Advantages of PaaS

Using a PaaS brings several advantages, such as:

- **Cost-Effective Development and Deployment**: Reduces the cost of underlying hardware, software, and hosting.

- **Increased Development Speed**: PaaS offers built-in software components and tools, which can drastically reduce coding time.
- **Management Simplicity**: The PaaS provider manages security, servers, storage, and networking, enabling developers to focus on the app development.

2.6.1.3 Use Cases of PaaS

PaaS is used in several scenarios including:

- Developing new applications rapidly.
- Analyzing or mining data.
- Adding web services to existing applications.
- Streamlining workflows when multiple developers are working on the same development project.
- Supporting the creation of customizable applications for businesses.

In summary, **PaaS** represents a significant shift in how businesses develop and deploy applications. It offers a high level of efficiency, flexibility, and scalability, making it an attractive option for companies looking to optimize their development resources.

2.6.2 Backend as a Service (BaaS)

Backend as a Service (BaaS), also known as "mobile backend as a service" (MBaaS), is a cloud computing model that provides developers with a way to connect their web and mobile applications to cloud services via APIs (Application Programming Interfaces) and SDKs (Software Development Kits). BaaS significantly simplifies backend development tasks, allowing developers to focus on the frontend and user experience.

2.6.2.1 Characteristics of BaaS

BaaS platforms typically offer a range of features, which include:

- **Database Management**: Cloud-based database services that can store and retrieve data.
- **User Authentication**: Tools for managing user accounts, authentication, and profiles.
- **Push Notifications**: Service for sending notifications to users.
- **Cloud Storage**: For storing files and media.
- **Server Code Execution**: Capability to run server-side logic.

- **APIs and SDKs**: For seamless integration with various platforms and technologies.

2.6.2.2 Advantages of BaaS

Using BaaS offers several advantages, such as:

- **Rapid Development**: Significantly reduces development time by removing the need to build and maintain server-side infrastructure.
- **Scalability**: Easily scales with user demand without manual intervention.
- **Cost-Effective**: Reduces costs associated with backend development and maintenance.
- **Real-Time Data Synchronization**: Ideal for applications requiring real-time data updates.

2.6.2.3 Applications of BaaS

BaaS is particularly useful in scenarios such as:

- Developing mobile and web applications with minimal backend setup.
- Implementing applications that require real-time data updates, like chat applications.
- Startups and small teams looking to launch applications quickly with limited resources.
- Projects that require rapid prototyping and testing.

In essence, **BaaS** is a crucial tool for modern application development, offering a streamlined approach to building and managing the backend. It enables developers to focus on creating rich user experiences while leveraging the power and scalability of cloud computing.

2.6.3 Websocket

The integration of WebSockets into our air quality monitoring system represents a significant step towards real-time web communication. WebSockets provide a full-duplex communication channel over a single, long-lived connection, which allows for instant data transfer between the client and the server. This section will discuss the implementation and advantages of using WebSockets for our system's data flow control.

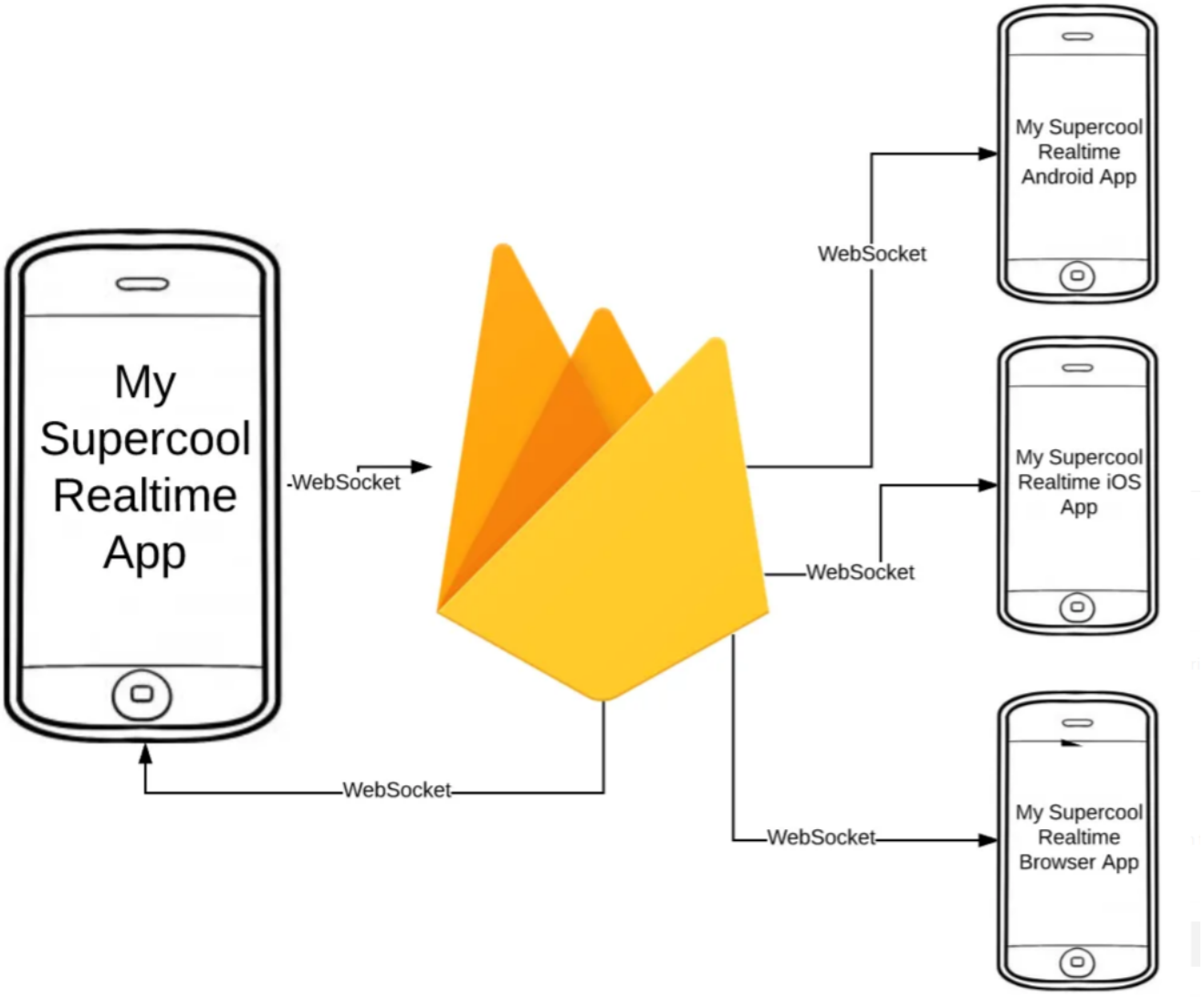


Figure 2.17: This diagram illustrates the real-time data flow between a central server and various client applications across different platforms. Using WebSockets, the server efficiently distributes live data updates to an Android app, an iOS app, and a browser app, ensuring a synchronous and interactive user experience.

2.6.3.1 WebSocket Protocol

WebSocket is a protocol that enables two-way communication between a client and a server on the web. It is designed to be implemented in web browsers and web servers but can be used by any client or server application. The WebSocket protocol is an independent TCP-based protocol, which is part of the HTML5 specification.

2.6.3.2 Advantages of WebSocket in IoT

The use of WebSockets in IoT applications, such as our air quality monitoring system, offers several benefits:

- **Real-Time Data Streaming:** WebSockets allow for the continuous flow of data between sensors and user interfaces without the need to refresh the page or poll the server for updates.
- **Low Latency:** By maintaining an open connection, WebSockets minimize the latency that can occur in traditional HTTP requests, ensuring that data updates are almost instantaneous.
- **Bi-Directional Communication:** Unlike HTTP, where communication is typically initiated by the client, WebSockets allow the server to push messages to the client as soon as data is available.
- **Efficient Resource Utilization:** WebSockets reduce network overhead by eliminating the need for repeated HTTP handshakes, thus conserving bandwidth and server resources.

2.6.3.3 Implementation Strategy

Our implementation strategy for WebSockets involves the following steps:

1. Establishing a WebSocket Connection: When a user opens the dashboard, the client application initiates a WebSocket handshake with the server. Once the handshake is completed, a persistent connection is established.
2. Data Transmission: Sensor data, once collected and processed, is pushed to the server using the MQTT protocol. The server then transmits this data to the client through the open WebSocket connection.
3. Data Reception: The client-side of the dashboard listens for incoming messages over the WebSocket connection. When new data arrives, it is immediately rendered on the

user interface.

4. Handling Disconnections: In case of disconnection, the system is designed to automatically attempt to re-establish the WebSocket connection, ensuring minimal disruption to the data stream.

2.6.3.4 System Integration

The WebSocket protocol has been integrated into the system architecture as a bridge between the data processing layer and the application layer. It facilitates a seamless and efficient transfer of data to the end-user interface, enabling users to monitor air quality metrics in real time.

2.6.3.5 Conclusion

The implementation of WebSockets in our air quality monitoring system underscores our commitment to providing a responsive and interactive user experience. As we continue to refine our system, the WebSocket protocol will remain a pivotal component in achieving our objective of delivering accurate and up-to-the-minute environmental data.

CHAPTER 3

DESIGN AND IMPLEMENTATION

3.1 System design overview

Based on the architecture discussed in Chapter 1 (Introduction), we propose a four-layer system architecture specifically including the Physical layer, Network layer, Data Processing layer, and Application layer. The specific architecture with the following components is as follows:

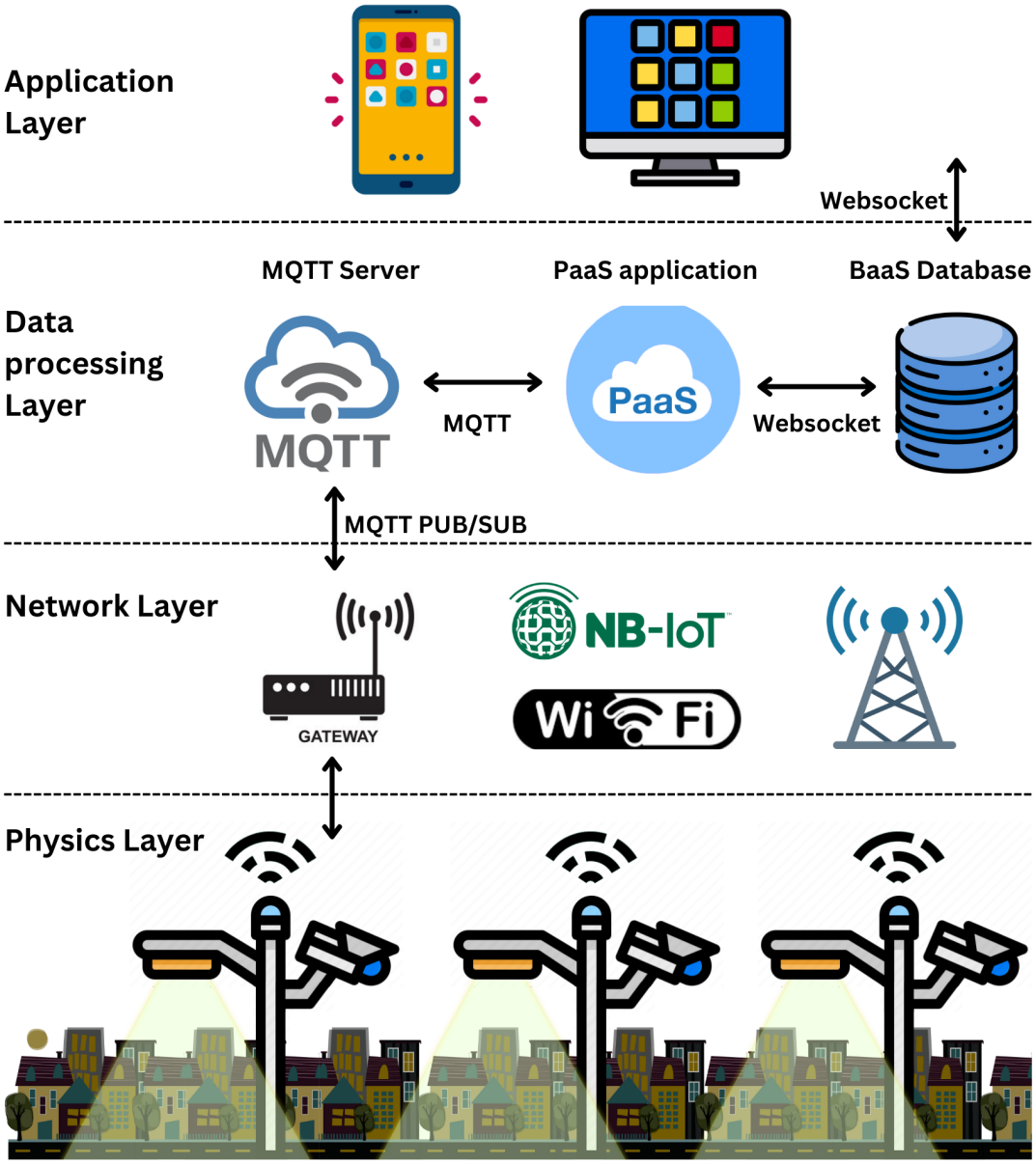


Figure 3.1: Proposed 4-layer IoT architecture in chapter 1

From this general architecture, we will analyze each part of the system to gain a more detailed perspective on each layer and its components.

3.2 Physics Layer

The physical layer consists of sensors, and actuators. Through this, we will analyze the structure of the sensor components first, followed by the structure of the actuators.

3.2.1 Physical architecture of the sensors

The sensor system will utilize the all-in-one M5Stack AirQ device [1]. This device allows us to update environmental sensor values such as Particulate matter, CO2 levels, temperature, humidity. To analyze the data read/write architecture of the system, we divide the electronic components of the sensor into blocks according to the following architecture:

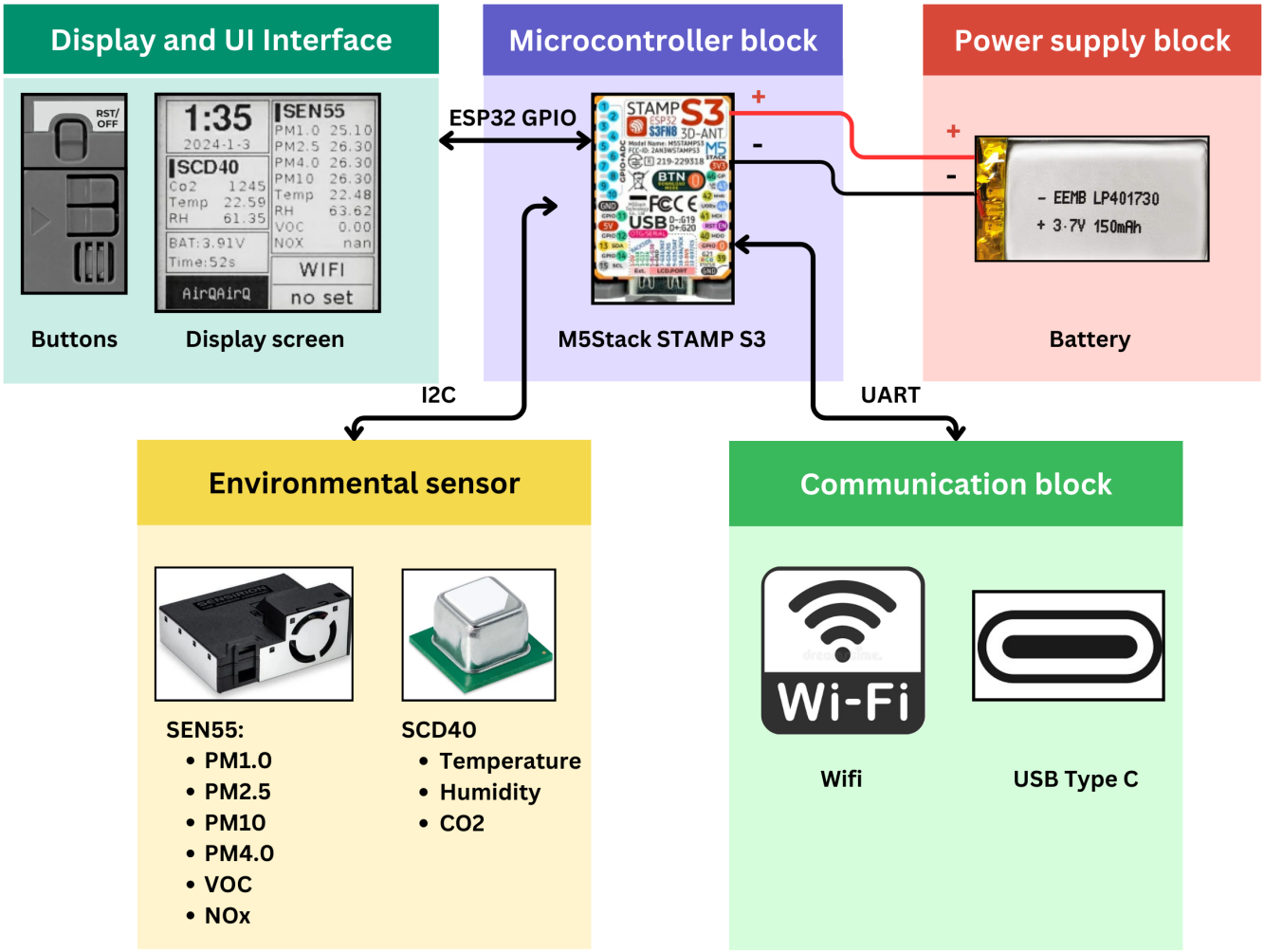


Figure 3.2: The architecture of the sensors in the physcial layer

1. **Power Supply Block:** The power supply block is responsible for providing the necessary power to all components of the M5AirQ. This includes converting the input voltage

[1] We will discuss about this device in section 3.3 (Hardware specification)

to the levels required by various parts of the circuit. Commonly, the power supply would include components for managing battery charging (if applicable), voltage regulation (stepping down the voltage to usable levels like 3.3V or 5V), and power distribution to other blocks. Components might include voltage regulators, capacitors for smoothing out fluctuations, and potentially a battery management system if the device is battery-powered.

2. **Environmental Sensor Block:** This block consists of various sensors integrated into the M5AirQ to monitor environmental parameters. Based on the typical usage of the M5AirQ, these sensors could include particulate matter sensors, gas sensors (for CO2, VOCs, etc.), temperature sensors, and humidity sensors. Each sensor outputs data that needs to be processed and interpreted by the main control unit. The sensors are usually connected via a communication protocol like I2C, as it supports multiple sensors over the same bus.

3. **Microcontroller Block:** At the heart of the M5AirQ is the microcontroller unit (MCU), which handles data acquisition, sensor management, data processing, and communication with external devices. The MCU reads the sensor data, processes it according to the programmed algorithms, and then can output this data to a display or over a network. The control block would also include interfaces for programming the device, debugging, or interfacing with other systems. This could involve standard programming interfaces like USB or UART, and in some devices, wireless communication modules (like Wi-Fi or Bluetooth) for remote data access and control.

4. **Display and User Interface Block:** If the M5AirQ includes a display, this block would manage the visual output of the device. Displays in such devices are typically used to show real-time data, system status, or configuration menus. This block would include the display hardware (like an LCD or OLED screen), drivers for the display, and potentially buttons or touch interfaces for user interaction.

5. **Communication Block:** This block handles all external communications. Depending on the capabilities of the M5AirQ, it could include components for wired communication (like USB or Ethernet) and wireless communication (like Wi-Fi, Bluetooth, or even LoRa for longer-range applications). This block is crucial for integrating the M5AirQ into larger systems or networks, allowing for data logging, remote monitoring, and control capabilities.

3.2.2 Physical architecture of the actuators

The main device is a streetlight controller compliant with the NEMA standard, which includes a central microcontroller and peripherals. The sensors collect environmental monitoring data and send it to the central microcontroller. From there, the controller transmits the data to the platform using an NB-IoT connection supported by mobile network base stations

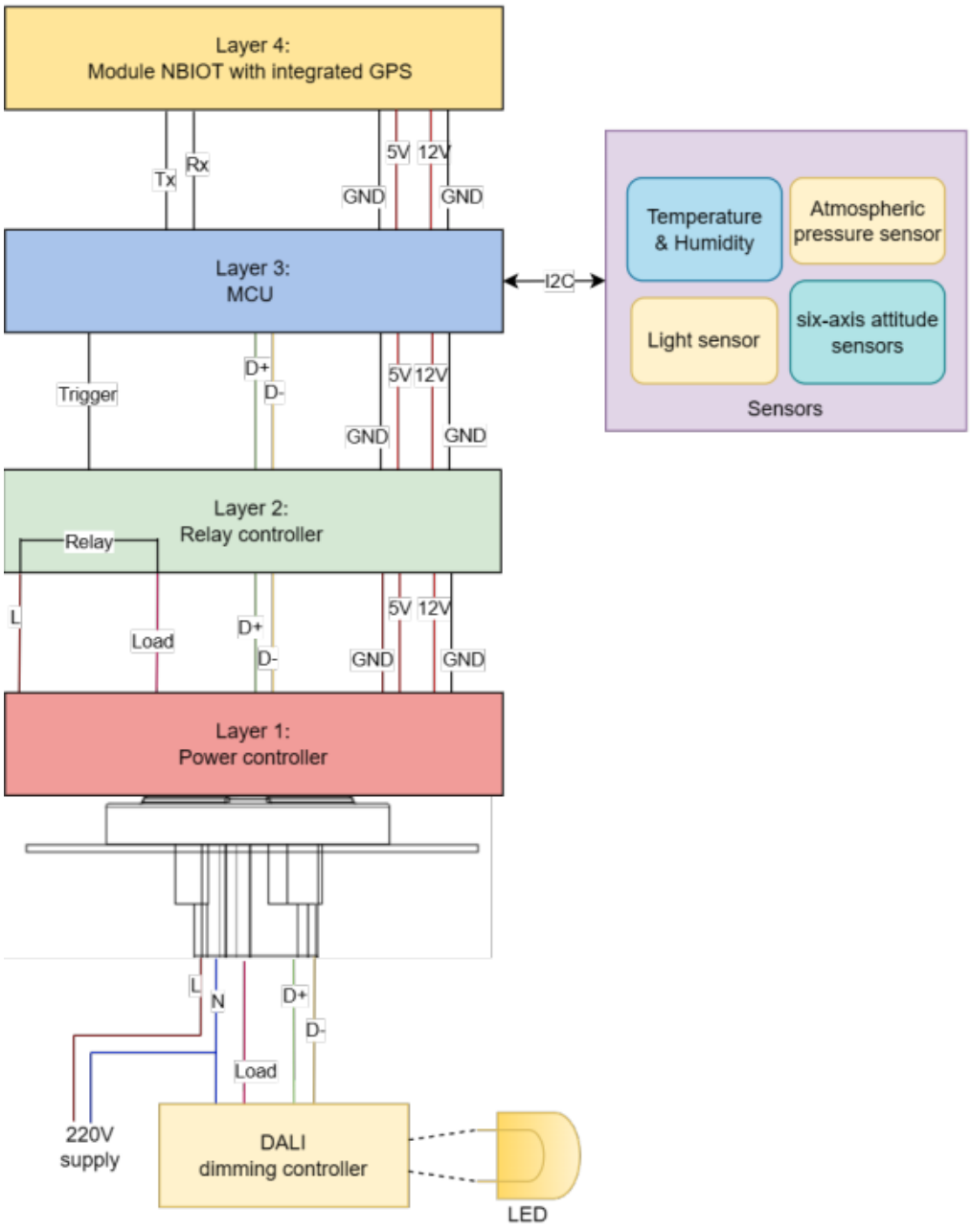


Figure 3.3: The architecture of the actuators in the physical layers for the standard NEMA streetlight controller

The NEMA streetlight controller is designed to manage the operations of urban street lighting efficiently. The hardware layers of this controller consist of several key components:

- **Microcontroller:** At the core of the controller is a robust microcontroller that processes input from various sensors and manages output to the lighting system and other peripherals.
- **Sensors:** Integrated sensors in the system detect environmental conditions such as ambient light, motion, and even weather conditions, enabling adaptive lighting control that enhances energy efficiency.
- **Communication Module:** To facilitate IoT connectivity, a communication module supports NB-IoT technology, allowing the controller to communicate with a central management system for real-time monitoring and control. This module ensures that data transmission is secure and reliable, leveraging mobile network infrastructures.
- **Power Supply:** A dedicated power supply unit ensures reliable operation, equipped with fail-safes to handle power surges and outages, maintaining lighting without interruption.
- **Actuators:** Actuators within the system execute the commands sent from the microcontroller, adjusting the lighting based on collected data and predefined algorithms, which can include dimming lights during low traffic hours to save energy.

These components are encapsulated within a hardware architecture that is optimized for low power consumption and high durability, suited for the rigorous demands of public infrastructure.

3.3 Network Layer

In this network layer section, we will discuss the intermediary connection technologies between components. Therefore, let's review the connection architecture to gain an overall perspective on the system connectivity of our devices throughout the entire system.

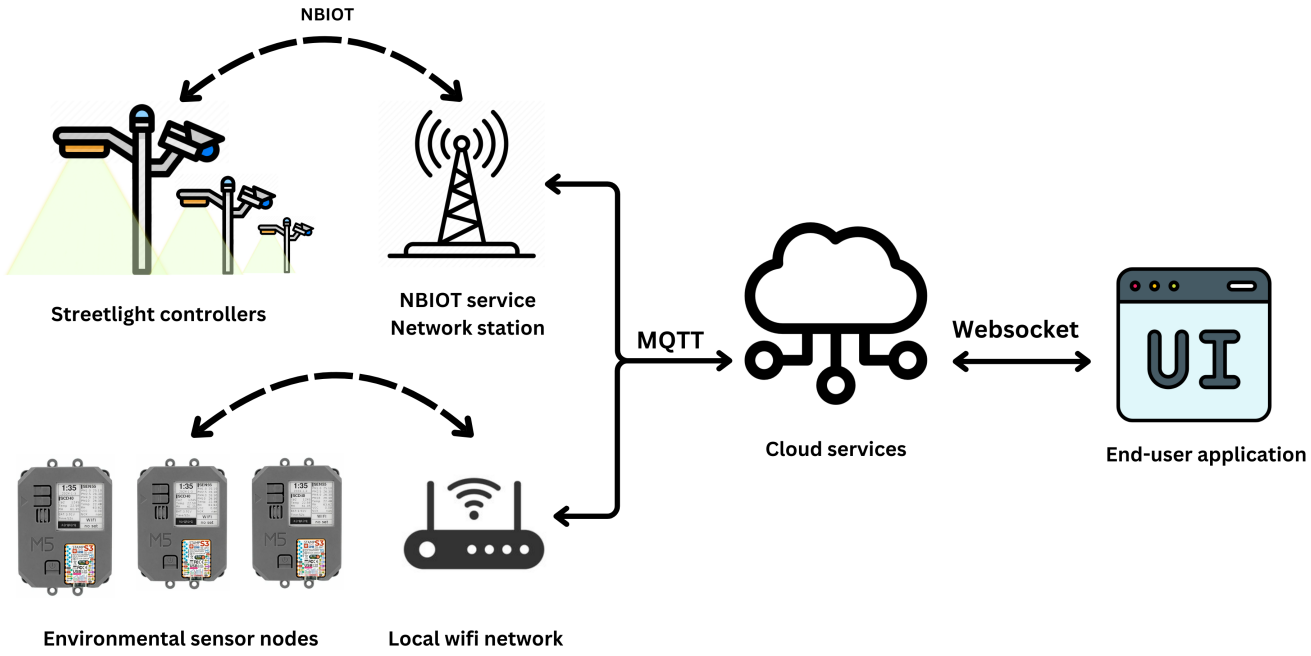


Figure 3.4: This diagram depicts the advanced network architecture of a smart street lighting system, emphasizing the critical role of technology in connectivity. Environmental sensor nodes are interconnected through a local WiFi network, and streetlight controllers seamlessly integrate with the broader **NB-IoT service network station**. This station utilizes the **MQTT** protocol to efficiently transmit data to cloud services, where it is then made available to end-users through a **Websocket**-enabled user interface (UI) application, facilitating real-time monitoring and operational control.

With the mentioned connection architecture, the implementation of the Network layer will be emphasized by connectivity technology **NB-IoT**. Through this, we will sequentially analyze the communication methods between components using these technologies.

Implementing NB-IoT communication in the Network Layer.
Implementing Narrowband IoT (NB-IoT) communication within the network layer involves several key steps, each crucial for ensuring reliable and efficient data transmission between devices and the network. Below, we detail the process, following the sequential steps as illustrated in the provided schematic diagram.

1. **Initialization:** Begin by initializing the modem to prepare it for connection.

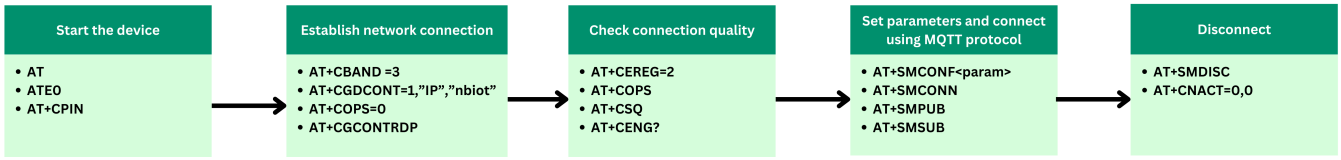


Figure 3.5: Step-by-step to connect and send data through NBIoT

- Send basic ‘AT‘ commands to check the readiness of the device.
- Disable echo with ‘ATE0‘ to streamline communication.
- Verify the SIM card’s status with ‘AT+CPIN?‘ to ensure readiness for network registration.

2. **Network Setup:** Configure the network settings after the device is initialized.
   - Set the communication band specific to the network provider and region with ‘AT+CBAND=3‘.
   - Define the PDP context for the NB-IoT connection with ‘AT+CGDCONT=1,"IP","nbiot"‘.
   - Automatically select the network operator with ‘AT+COPS=0‘.
   - Display the PDP addresses with ‘AT+CGCONTRDP‘, confirming the device’s registration and connectivity status.
3. **Quality Check:** Ensure the network connection is of good quality before data transmission.
   - Set the device to report network registration status and location information with ‘AT+CEREG=2‘.
   - Confirm the current network operator details with ‘AT+COPS?‘.
   - Check the signal quality with ‘AT+CSQ‘.
   - Query detailed network information with ‘AT+CENG?‘ for troubleshooting and optimizing network performance.
4. **MQTT Setup and Communication:** Set up MQTT for data transmission over the NB-IoT network.
   - Configure MQTT session parameters with ‘AT+SMCONF‘.
   - Establish an MQTT connection with ‘AT+SMCONN‘.
   - Publish messages to a topic on the MQTT broker with ‘AT+SMPUB‘.

- Subscribe to a topic to receive messages with ‘AT+SMSUB‘.

5. **Connection Termination:** Properly close the network connection to end the session.
    - Disconnect the MQTT session with ‘AT+SMDISC‘.
    - Deactivate the NB-IoT connection with ‘AT+CNACT=0,0‘, ensuring the modem properly disconnects from the network.

Following these steps ensures that NB-IoT devices can communicate efficiently over the network, leveraging MQTT for real-time data exchange and control, integral for smart city applications and other IoT ecosystems.

3.4 Data Processing Layer

In the data processing layer, the primary tasks involve data **formatting**, **storage**, and **processing**. This layer handles three types of information, each crucial for the integrated system's functionality:

1. **Actuator Control Information:** This includes commands and control signals sent to actuators within the system, enabling real-time interaction with the physical environment.
2. **Sensor Data:** Information collected from various sensors deployed within the system, which may include environmental data, operational metrics, and other sensor readings essential for monitoring and control.
3. **Camera Image Data:** Visual data captured by surveillance cameras integrated into the system. This data is particularly important as it supports security and monitoring functions.

Given these types of data, the information from sensors and actuators is processed in a similar manner, focusing on efficient data integration and real-time responsiveness. Meanwhile, camera data undergoes a more complex processing path. It is not only captured and stored but also analyzed with the help of Artificial Intelligence (AI) technologies within this layer. The AI algorithms applied to the image data enable advanced features such as motion detection, object recognition, and even behavioral analysis, enhancing the system's capabilities in surveillance and security.

To streamline the discussion and clarity of implementation, this section is divided into two parts:

- **Processing of Sensor and Actuator Data:** This part covers the mechanisms and technologies used to handle data from sensors and actuators. It includes data collection, preprocessing, formatting, and transmission to ensure that the data is actionable and available for real-time decision-making processes.
- **AI-Based Image Data Processing:** Focusing on the camera data, this part discusses the integration of AI technologies to process and analyze image data. The processing involves not just basic image capture and storage but also the application of AI models that can extract valuable insights from the visual data, thereby contributing significantly to the objectives of surveillance and monitoring.

Through efficient data processing strategies and the use of advanced AI, the data processing layer plays a pivotal role in ensuring that the system not only collects data but also transforms it into meaningful information that supports all operational aspects of the smart system.

3.4.1 Processing of Sensors and Actuators Data

The efficient processing of data from sensors and actuators is critical for the responsive operation of smart lighting systems. This subsection delves into the data flow, data transformation, and subsequent actions taken based on the data received from sensors and commands sent to actuators.

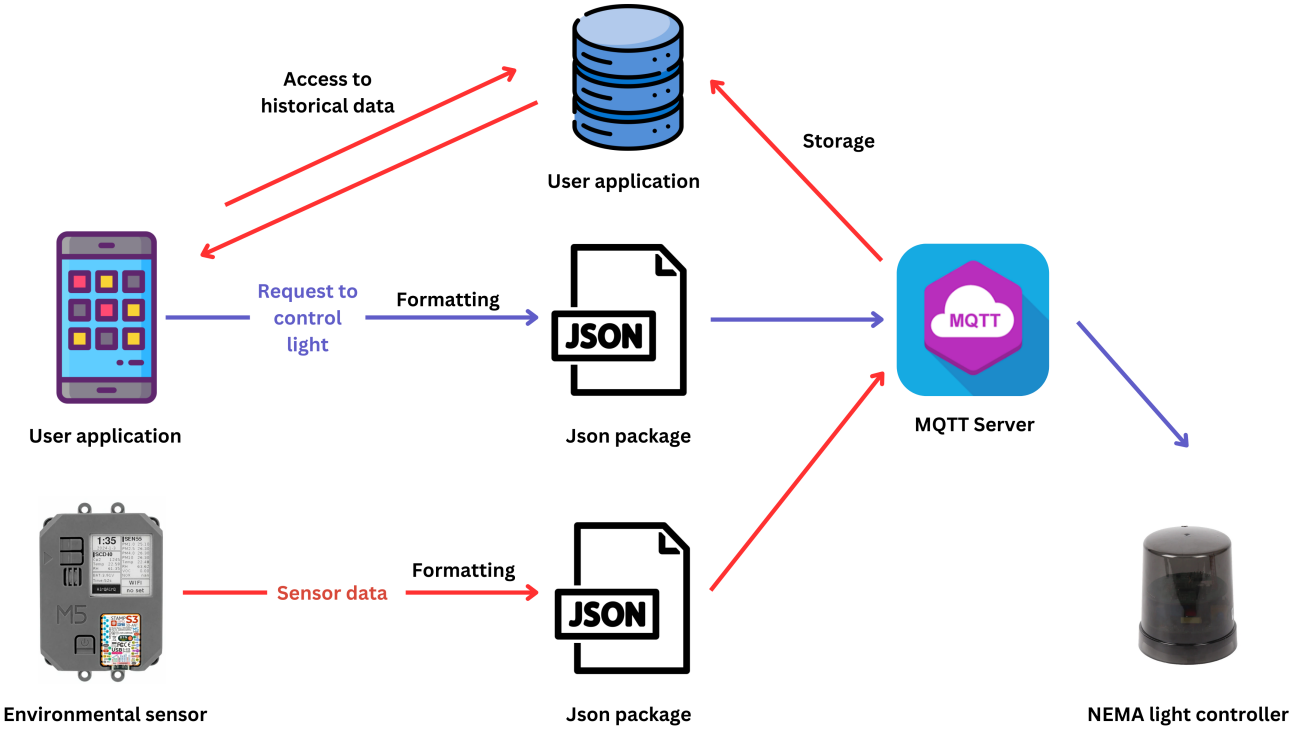


Figure 3.6: This diagram illustrates the data flow within a smart lighting system, highlighting the interactions between environmental sensors, user applications, and the NEMA light controller. Data from the environmental sensors are formatted into JSON packages, which are then sent to an MQTT server for efficient data handling. Simultaneously, user applications send control requests, which are also formatted as JSON and communicated via the MQTT server. This centralized server manages all incoming data and commands, ensuring that commands are relayed back to the NEMA light controller for actuation and that sensor data is stored for both immediate use and historical analysis. This setup exemplifies a seamless integration of sensor inputs and user interactions in modern IoT environments.

- **Data Formatting:** Environmental data collected by sensors are formatted into struc-

tured JSON packages. This standardization facilitates easier data parsing and processing by subsequent systems. Similarly, actuator commands generated by user applications are encapsulated in JSON format to ensure consistency and compatibility across the network.

- **MQTT Communication:** The use of the MQTT protocol allows for lightweight and efficient transmission of both sensor data and control commands. MQTT's publish-subscribe model is particularly advantageous in this setup, enabling the MQTT server to efficiently distribute commands to the appropriate actuators while also handling incoming sensor data.
- **Data Handling and Storage:** Once received by the MQTT server, sensor data are stored in a centralized database for both immediate and historical analysis, supporting real-time operational decisions and long-term strategic planning. The data storage is structured to support quick retrieval and scalable storage solutions.
- **Actuator Control:** Control signals sent from user applications are processed by the MQTT server and then relayed to the respective actuators, such as the NEMA light controller. This ensures that user commands are executed accurately and promptly, adjusting the lighting based on environmental conditions or user preferences.

The integration and processing of sensor and actuator data as described not only optimize the operational efficiency of the smart lighting system but also enhance user interaction, making the system more adaptive and responsive to environmental changes and user needs.

3.4.2 AI Image Data Processing by using Object tracking based Yolov9 and DeepSORT

3.4.2.1 Overview

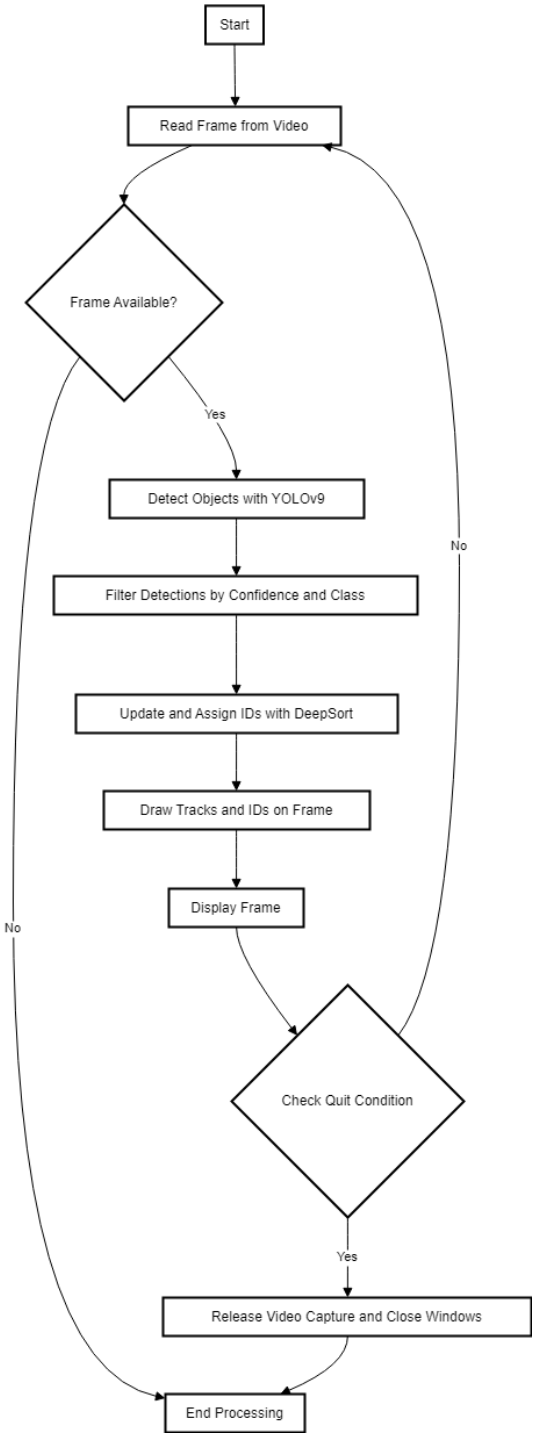


Figure 3.7: Real-time object tracking with YOLOv9 and DeepSORT flowchart

The implementation of the object tracking system using YOLOv9 and DeepSORT is designed to provide real-time monitoring and tracking capabilities. This process is primarily geared toward enhancing security measures by accurately identifying and tracking various objects within video footage. The flowchart and the provided Python script outline a comprehensive system that integrates cutting-edge technologies for effective object detection and tracking.

Video Capture and Frame Processing

The process begins with the initialization of a video capture object, which reads frames from a specified video source. This could be a video file or a live video stream. Each frame from the video source is then analyzed to determine if it is valid. If a frame is not available (indicating the end of the video or a streaming issue), the process moves towards a graceful termination, where resources are released and the application is properly closed.

Object Detection with YOLOv9

For frames that are successfully captured, the system employs the YOLOv9 model to detect objects within the frame. YOLOv9 is utilized here due to its high accuracy and

speed, which are crucial for real-time processing scenarios. The detection step involves analyzing the frame to locate objects and classify them based on a trained model. The model predicts bounding boxes and class labels for each detected object, and these are filtered based on a predefined confidence threshold to ensure the relevance and accuracy of the detections.

Tracking with DeepSORT

Post-detection, the DeepSORT algorithm is applied. This step is critical as it associates the detected objects across multiple frames, providing a consistent track ID for each object. DeepSORT uses a combination of motion and appearance information to maintain tracking even when objects move or temporarily get occluded. It updates the tracks based on new detection data and adjusts the tracked object's paths accordingly.

Drawing and Displaying Results

Once objects are detected and tracked, their current positions are annotated on the frame. This includes drawing bounding boxes around each object and labeling them with class names and track IDs. These visual enhancements help users to easily identify and follow the objects in the video.

Continuous Processing and User Interaction

The system is designed to run in a continuous loop, where it processes each frame sequentially until an exit condition is met. This condition is typically triggered by user input, such as pressing a specific key. Upon triggering the exit condition, the system performs cleanup by releasing the video capture resources and closing any active windows, ensuring an orderly shutdown.

3.4.2.2 Application of SmartPole Surveillance Cameras

In our Smart Pole project, object tracking plays a crucial role in enhancing operational efficiency and energy conservation. The primary application of this technology is to automate the control of street lighting based on the presence of vehicles within the area covered by each pole.

Energy Conservation

Using YOLOv9 combined with DeepSORT, our system can detect and track moving objects such as cars and bicycles. When the tracking system identifies a vehicle entering the vicinity of a smart pole, it triggers the street lights to turn on, providing necessary illumination. Conversely, the lights are programmed to turn off when no vehicles are present, significantly

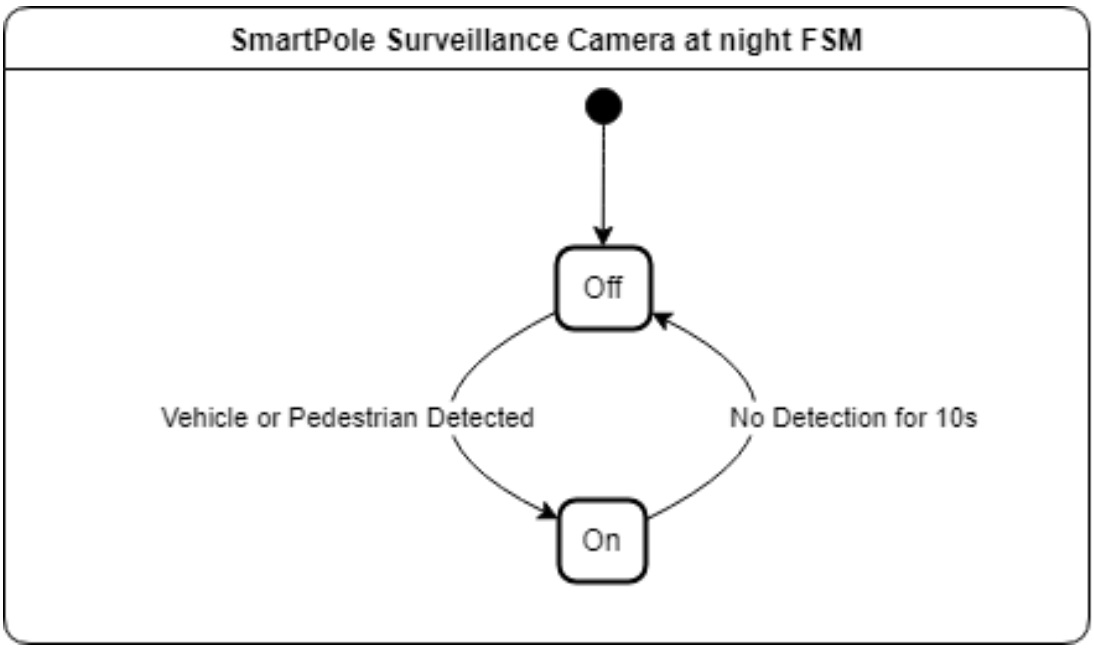


Figure 3.8: Finite State Machine for Surveillance Camera at night

reducing unnecessary energy consumption. This automated light control not only saves energy but also extends the lifespan of the lighting fixtures.

Future Development and Scalability

While the current implementation focuses on detecting vehicles to control lighting, the decision to utilize a robust tracking system like DeepSORT provides several advantages for future enhancements. The tracking capability allows us to easily integrate additional features without extensive modifications to the core system.

- **Speed Calculation:** In the future, we plan to calculate the speed of passing vehicles. This feature will utilize the tracking data
- **Traffic Flow Analysis:** By continuously tracking vehicles, the system can analyze traffic patterns and provide real-time data on traffic density and flow. This information could be used to optimize traffic light control and reduce congestion.
- **Enhanced Security Features:** The tracking system can be adapted to alert security personnel about unusual activities or unauthorized vehicle movements at odd hours, enhancing safety and security in urban areas.
- **Pedestrian Tracking for Crosswalk Safety:** Extending the system to track pedestrians can improve safety at crosswalks. Smart poles can be programmed to increase illumination and flash warning lights when pedestrians are detected, especially during night or adverse weather conditions.

Incorporating advanced object tracking into the Smart Pole project not only fulfills the immediate goal of energy conservation but also sets a foundation for future smart city solutions. The scalability of the tracking system opens up numerous possibilities for enhancing urban infrastructure, making cities safer, more efficient, and environmentally friendly.

3.5 Application layer

The application layer in the Smart Pole project serves as the operational core that integrates and manages the diverse functionalities embedded within each pole. This layer is crucial for translating technological capabilities into practical, user-centered applications that enhance urban environments.

Strategic Integration of Smart Technologies

The application layer orchestrates the interaction between various smart technologies to achieve optimal efficiency and responsiveness. By coordinating the operation of lighting systems, environmental sensors, security cameras, and communication devices, the layer ensures that each component operates synergistically. This integration supports dynamic responses to real-time urban conditions, enhancing the effectiveness of each smart pole's contribution to city operations.

Key Applications

- **Dynamic Urban Illumination:** Smart poles automatically adjust lighting based on pedestrian and vehicular activity, detected via integrated sensors. This not only improves energy efficiency but also ensures safety in urban areas during night-time or low-light conditions.
- **Public Safety and Emergency Response:** In emergencies, smart poles act immediately to enhance safety. They can light up strategically to guide evacuation routes or highlight areas of concern, simultaneously alerting emergency services through integrated communication systems.
- **Environmental Monitoring and Public Health:** Constantly monitoring environmental parameters, smart poles provide data crucial for managing urban air quality and public health. This real-time data supports proactive city planning and immediate responses to environmental hazards.
- **Seamless Connectivity:** Serving as nodes within a broader urban communication network, smart poles provide essential connectivity that supports everything from public Wi-Fi access to the backbone of IoT infrastructures, facilitating a myriad of smart city applications.
- **Community Engagement and Information:** Utilized as digital signages, smart poles display real-time public information, emergency alerts, and community messages, keeping the public informed and engaged with city events and announcements.

- **Resource Efficiency:** The application layer optimizes the use of urban infrastructure. By managing resources efficiently, such as reducing power consumption when not required, the smart poles exemplify sustainable urban development.

The application layer is pivotal in transforming individual technological components into a coherent, interactive system that not only fulfills specific functional requirements but also enhances the broader objectives of urban management and sustainability. Through this layer, smart poles become more than just infrastructure—they evolve into active participants in the smart city ecosystem, adapting to and anticipating the needs of the urban environment.

3.6 Software requirements specification

3.6.1 Functional Requirements

The smart pole integrates various functionalities, making it an indispensable asset for modern cities. From efficient lighting management to emergency response capabilities, this intelligent pole serves as a beacon of progress. These are some key features:

3.6.1.1 Light Control and Brightness Adjustment

Requirement	The Smart Pole empowers users to control the ON/OFF state of its light.
Details	Through an MQTT server, users toggle the light, adapting to changing needs. Gradual brightness adjustment ensures optimal illumination for diverse scenarios, from bustling streets to tranquil parks.

3.6.1.2 Electrical Device Charging Plug-In

Requirement	Convenience meets functionality with the Smart Pole's charging point.
Details	Standard electrical outlets or USB ports cater to device charging. Compatibility with common charging cables (USB-A, USB-C) ensures seamless user experience.

3.6.1.3 Emergency Button and SOS Signal

Requirement	Urgency demands action, and the Smart Pole responds swiftly.
Details	The emergency button triggers immediate alerts, sending distress signals to the central server. Audible alarms or visual indicators enhance situational awareness.

3.6.1.4 Air Quality Sensor

Requirement	The Smart Pole breathes with the city, monitoring air quality.
Details	Sensors measure temperature, humidity, and air pollution levels. Real-time data transmission informs urban planning and health initiatives.

3.6.1.5 Security Camera

Requirement	The watchful eye of the Smart Pole detects relevant objects.
Details	Computer vision algorithms identify cars, pedestrians, pets, and anomalies. Lighting adjusts based on camera input, enhancing safety and energy efficiency.

3.6.1.6 Advertisement Display Screen

Requirement	The Smart Pole doubles as an advertising canvas.
Details	Administrators remotely manage ad content, ensuring timely messaging. Synchronized ad schedules across multiple poles create a cohesive urban advertising network.

3.6.1.7 Wireless Connectivity and Network Coordination

Requirement	Seamless communication weaves through the cityscape.
Details	5G small cells and Wi-Fi hotspots enable high-density network deployment. Smart poles coordinate lighting patterns, emergency alerts, and data exchange in an interconnected fabric.

3.6.2 Non-funtional requirements

In the pursuit of creating intelligent and sustainable urban environments, the Smart Pole emerges as a multifunctional fixture that transcends mere illumination. Beyond its primary role as a light source, the Smart Pole integrates a myriad of features—ranging from environmental monitoring to emergency response mechanisms. However, the success of this urban innovation hinges not only on its functionalities but also on its non-functional attributes.

- **Durability and Weather Resistance:**

  The Smart Pole stands sentinel in the face of nature's fury. Its housing material must withstand extreme temperatures, relentless rain, and gusty winds. Corrosion-resistant alloys and robust construction ensure longevity, while an IP65 or higher rating shields against dust ingress and water jets. Vandal-resistant design deters mischief, safeguarding both functionality and aesthetics.

- **Energy Efficiency and Independence:**

Energy conservation lies at the core of sustainability. The Smart Pole optimizes power usage through high-efficiency LED lighting, solar panels, and wind turbines. Excess energy is stored in batteries, ensuring uninterrupted operation even during cloudy days. By reducing reliance on the grid, the Smart Pole contributes to a greener urban footprint.

- **Security and Privacy:**

  Beyond aesthetics, security is paramount. Tamper-resistant features prevent unauthorized access, while flame-retardant materials enhance fire safety. The integrated camera system, crucial for object detection, adheres to privacy regulations. Balancing public safety with individual privacy remains a delicate yet essential aspect of the Smart Pole's design.

- **Seamless Connectivity and Coordination:**

  Wireless veins pulse through the cityscape. The Smart Pole communicates seamlessly via 5G small cells and Wi-Fi hotspots. Low latency ensures real-time data exchange, while redundant communication paths enhance reliability. Interconnected poles synchronize lighting patterns, emergency alerts, and data flow, creating an urban fabric that adapts harmoniously.

- **Maintenance and Proactive Serviceability**

  Behind the sleek facade lies practicality. Access panels grant maintenance crews swift entry for inspections, repairs, and upgrades. Component lifespans are estimated, ensuring timely replacements. Remote diagnostics tools detect anomalies proactively, minimizing downtime. The Smart Pole's vigilance relies on efficient serviceability.

3.6.3 Use-case Diagram

3.6.3.1 Whole system

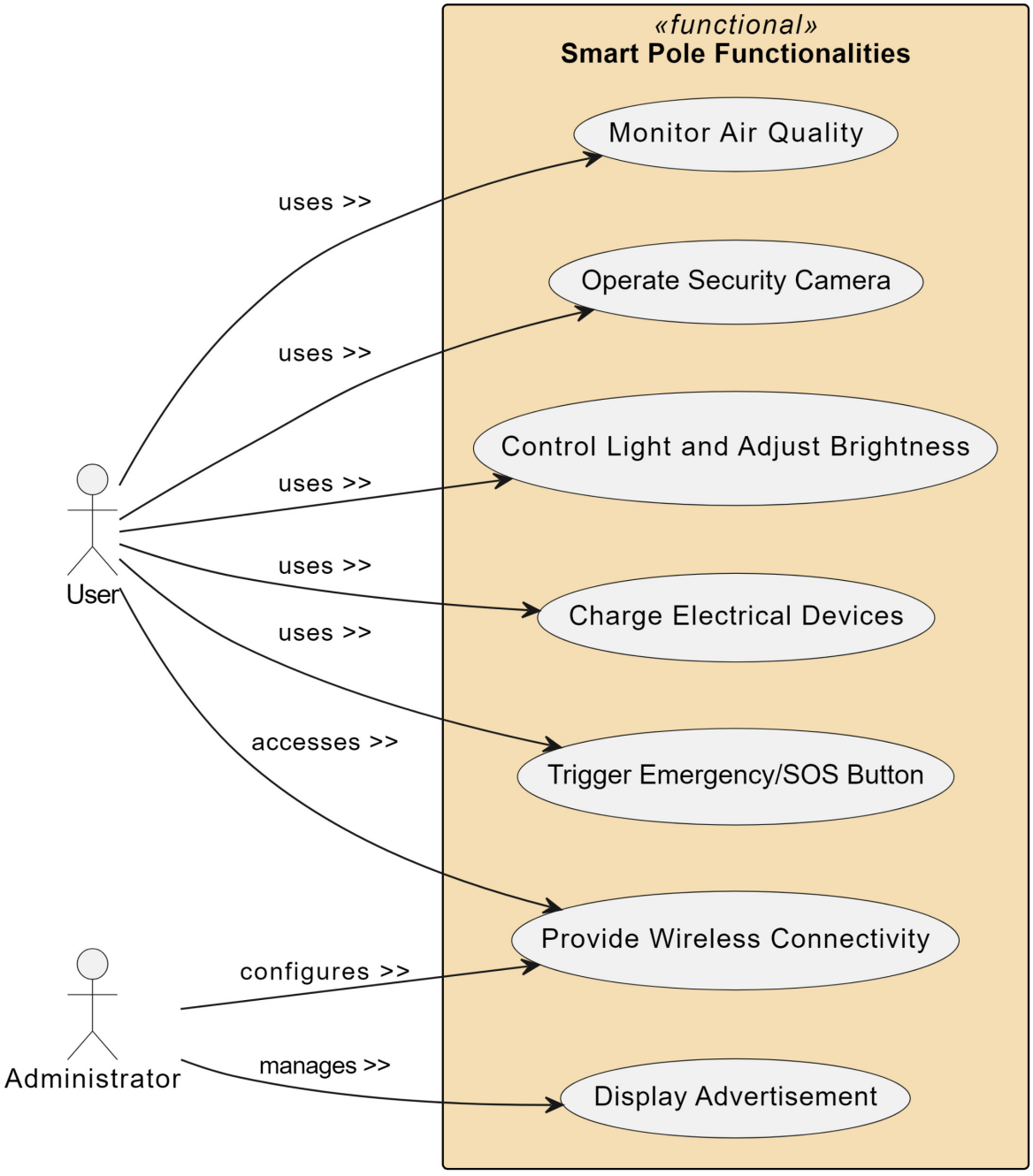


Figure 3.9: Usecase diagram of system

3.6.3.2 Module: Control light and Adjust Brightness

Use Case Name	Control Light and Adjust Brightness
Primary Actor	User
Goal	To control and adjust the lighting of the smart pole.
Description	Users can control the ON/OFF state of the smart pole's light and adjust its brightness to suit various environmental conditions. The system allows control through an MQTT server, enabling users to toggle lights and modify brightness levels to ensure optimal illumination for different settings such as busy streets or tranquil parks.
Trigger	User interacts with a mobile app or physical interface.
Preconditions	The smart pole must be operational and connected to the control network.
Postconditions	The lighting state is changed as per the user's preferences.

3.6.3.3 Module: Charge Electrical Devices

Use Case Name	Charge Electrical Devices
Primary Actor	User
Goal	To provide power outlets for charging electronic devices.

Description	The smart pole includes standard electrical outlets and USB ports that allow users to charge their devices conveniently. It supports common charging cables such as USB-A and USB-C, catering to a wide range of mobile phones and other electronic devices.
Trigger	User plugs a device into the charging point.
Preconditions	The charging facility is functional and accessible.
Postconditions	The user's device is charged.

3.6.3.4 Module: Trigger Emergency/SOS Button

Use Case Name	Trigger Emergency/SOS Button
Primary Actor	User
Goal	To enable users to initiate an emergency alert.
Description	In cases of urgency, users can press an emergency button located on the smart pole. This action triggers an immediate alert to the central server, which may activate audible alarms or visual indicators to notify nearby individuals and authorities.
Trigger	User presses the emergency button.
Preconditions	The emergency system is active and monitored.
Postconditions	Emergency services are alerted, and response mechanisms are activated.

3.6.3.5 Module: Monitor Air Quality

Use Case Name	Monitor Air Quality
Primary Actor	System
Goal	To continuously monitor and report the air quality in its vicinity.
Description	The smart pole is equipped with sensors that measure temperature, humidity, and air pollution levels. This data is transmitted in real-time to assist urban planning and health monitoring initiatives.
Trigger	Sensors collect environmental data at set intervals.
Preconditions	Sensors are calibrated and operational.
Postconditions	Air quality data is updated and available for analysis.

3.6.3.6 Module: Operate Security Camera

Use Case Name	Operate Security Camera
Primary Actor	System
Goal	To enhance area security through surveillance.
Description	Equipped with advanced computer vision algorithms, the smart pole's security cameras can detect and analyze movements of cars, pedestrians, and other entities. The system adjusts lighting based on camera inputs to improve visibility and energy efficiency.
Trigger	Motion or presence detected by the camera.
Preconditions	Cameras are functional and the area is within camera view.

Postconditions	Relevant footage is captured and stored for security purposes.

3.6.3.7 Module: Display Advertisement

Use Case Name	Display Advertisement
Primary Actor	Administrator
Goal	To manage and display digital advertisements.
Description	Administrators can remotely manage content displayed on the smart pole's digital screens. This feature supports synchronized ad schedules across multiple poles, creating an integrated advertising network within the urban space.
Trigger	New ad content is scheduled for display.
Preconditions	Advertisement content is prepared and scheduled.
Postconditions	Ads are displayed as per the schedule.

3.6.3.8 Module: Provide Wireless Connectivity

Use Case Name	Provide Wireless Connectivity
Primary Actor	User and Administrator
Goal	To offer high-speed internet access and network coordination.
Description	The smart pole includes 5G small cells and Wi-Fi hotspots that provide extensive wireless network coverage. Administrators configure and manage the network, while users can connect to access high-speed internet.
Trigger	Device attempts to connect to the network.

Preconditions	Network equipment is active and operational.
Postconditions	Users gain network access, and data is exchanged seamlessly across the network.

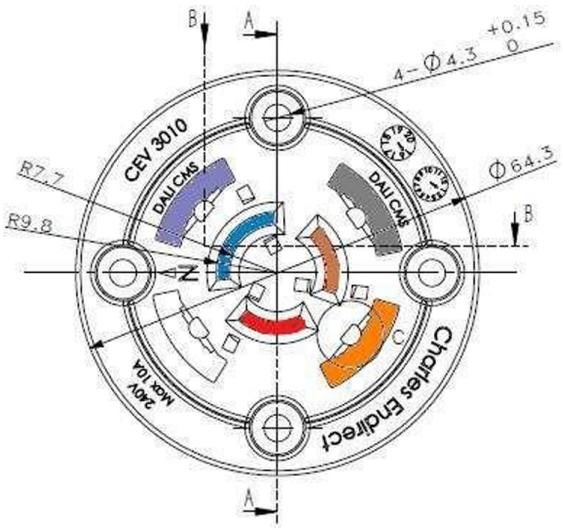
3.7 Hardware specification

3.7.1 NEMA socket

The NEMA 7-pin socket from Charles Endirect is designed to meet the requirements of BS5792 and ANSI C136.41-2013. Designed to IP2X standard, this socket not only ensures protection against accidental finger contact but also prevents intrusion from objects with a diameter larger than 12mm. Additionally, this product integrates support for DALI (Digital Addressable Lighting), enhancing its control capabilities in lighting systems.

With this special feature, the socket is expected to become a popular choice in urban street lighting systems. Installation and connection to the light become simpler, and the sturdy protective layer on top effectively prevents the intrusion of dust, increasing the lifespan and performance of the product.

The socket includes 3 power pins and 4 data pins. Among these, the neutral pin is green, while the 2 red and brown pins are connected together with a switch and linked to the power supply.



(a) Schematic for NEMA socket

(b) Real image for NEMA socket

Figure 3.10: Standard NEMA socket schematic and reality

3.7.2 M5Stack AirQ:

The M5Stack AirQ represents a pivotal component in our smart pole project, providing real-time air quality monitoring solutions. Designed for efficiency and accuracy, the AirQ integrates multiple sensors to assess ambient air quality. Let's delve into its hardware speci-

Figure 3.11: M5Stack AirQ: Environment sensor

fications and capabilities.

HARDWARE SPECIFICATION:

- **SEN55 Air Quality Sensor:**
  - Measures:
    - PM1.0, PM2.5, PM4, and PM10 particle concentrations.
    - Temperature and humidity levels.
    - VOC (volatile organic compounds) concentrations.
  - Ensures precise data collection ensures comprehensive air quality assessment.
- **SCD40 CO2 Sensor:**
  - Specifically detects carbon dioxide (CO2) levels.
  - Essential for assessing indoor air quality and occupant health.
- **Main Controller (MCU):**
  - Powered by the StampS3 microcontroller (ESP32S3FN8) with 8M Flash memory.
  - Efficient data processing and storage capabilities.
- **Display:**
  - Features a 1.54-inch e-ink screen with a resolution of 200x200 pixels.
  - Clear visual representation of collected air quality data.
  - Even during power outages, the final data screen remains visible.
- **Battery and Power Management:**

  - Equipped with a 450mAh battery for extended operation.
  - RTC low-power power management circuit conserves energy.
  - Prolongs battery life for continuous monitoring.

- **Data Upload and Cloud Integration:**
  - Preconfigured firmware uploads data to M5's EZDATA cloud platform.
  - Automatic monitoring pages for convenient remote access and data management.
  - Users can monitor air quality data anytime, anywhere via cloud-based access.

- **Installation Flexibility:**
  - LEGO mounting holes, adhesive magnets, and detachable mounting ears.
  - Versatile installation options in homes, schools, industrial facilities, and hospitals.
  - Enables long-term online air quality monitoring.

3.7.3 ESP32 board:

The ESP32 is a versatile and powerful microcontroller unit (MCU) designed by Espressif Systems. As a successor to the popular ESP8266, the ESP32 offers significant upgrades, making it an ideal choice for various applications, including our smart pole project. Let's explore the ESP32's features, specifications, and its role in enhancing connectivity.

KEY FEATURES OF ESP32:

- **Dual-Core 32-bit LX6 Microprocessor:**
  - The ESP32 features either a single-core or dual-core Tensilica Xtensa LX6 microprocessor.
  - Clock frequency can reach up to 240 MHz, providing ample processing power for complex tasks.

- **Ample Memory Resources:**
  - 520 KB of SRAM (Static Random-Access Memory) ensures efficient data handling.
  - 448 KB of ROM (Read-Only Memory) stores essential firmware and code.
  - An additional 16 KB of RTC (Real-Time Clock) SRAM for time-sensitive operations.

- **Wi-Fi and Bluetooth Connectivity:**

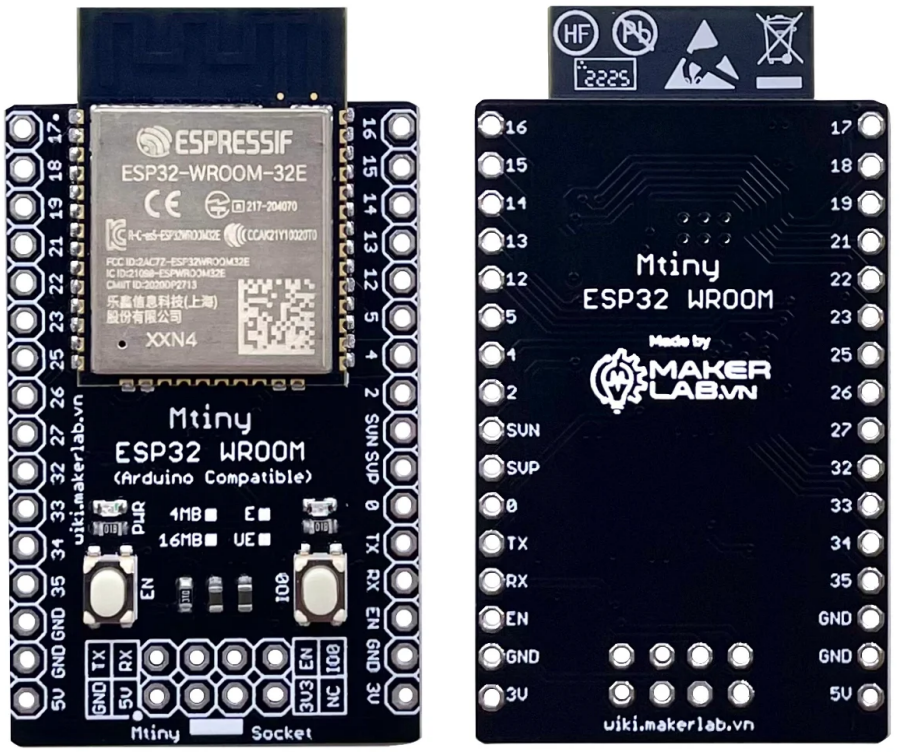


Figure 3.12: ESP32 Board

  - Supports 802.11 b/g/n Wi-Fi connectivity with speeds up to 150 Mbps.
  - Dual-mode Bluetooth:
    * Classic Bluetooth v4.2 for legacy devices.
    * BLE (Bluetooth Low Energy) for energy-efficient communication.

- **Integrated RF Components:**
  - ESP32 includes a Power Amplifier, Low-Noise Receive Amplifier, Antenna Switch, Filters, and RF Balun.
  - Simplifies hardware design by reducing external components.

- **Ultra-Low-Power Technology:**
  - Manufactured using TSMC's 40 nm ultra-low-power process.
  - Ideal for battery-operated applications such as wearables, audio devices, and smartwatches.

APPLICATIONS OF ESP32:

- **IoT (Internet of Things) Devices:**

  - ESP32's Wi-Fi and Bluetooth capabilities make it a go-to choice for IoT projects.
  - Home automation, smart appliances, and environmental monitoring benefit from its connectivity.
- **Wearable Electronics:**
  - ESP32's low power consumption suits wearables like fitness trackers and smartwatches.
  - Real-time data collection and wireless communication enhance user experience.
- **Industrial Automation:**
  - ESP32 integrates seamlessly into industrial systems.
  - Monitor sensors, control actuators, and communicate data over Wi-Fi or Bluetooth.
- **Audio Equipment and Smart Speakers:**
  - High-resolution ADCs (Analog-to-Digital Converters) and DACs (Digital-to-Analog Converters) enable audio processing.
  - Bluetooth connectivity for wireless audio streaming.
- **Wireless Sensor Networks:**
  - ESP32's range and low power make it suitable for large-scale sensor networks.
  - Collect data from remote locations and transmit it to a central hub.

3.7.4 M5Stack Unit NB-IoT:

The M5Stack Unit NB-IoT is a wireless communication module specifically designed for narrowband IoT (NB-IoT) applications. As part of our smart pole project, this module plays a crucial role in enhancing connectivity and enabling efficient data transmission. Let's explore its features, specifications, and applications.

KEY FEATURES OF M5STACK UNIT NB-IOT:

- **Global Coverage and Multi-Band Support:**
  - The Unit NB-IoT operates across the Cat-NB frequency band, making it applicable globally.
  - It supports multiple bands, including: B1, B2, B3, B4, B5, B8, B12, B13, B17, B18, B19, B20, B25, B26, B28, B66, B70, B71, B85.

- **Strong Signal Access Capability:**
  - The built-in SIM7020G communication module ensures robust signal reception.
  - Improved communication quality and signal stability enhance overall performance.
- **AT Command Control:**
  - Developers can control the Unit NB-IoT using AT commands.
  - This flexibility allows customization and integration into various IoT applications.
- **Nano SIM Card Support:**
  - The module accommodates Nano SIM cards, ensuring compatibility with standard SIM sizes.
- **External Antenna Interface:**
  - The integrated SMA external antenna interface enhances signal reception.
  - Reliable communication quality even in challenging environments.
- **Data Transmission Speeds:**
  - Downlink (DL): Up to 126 kbps.
  - Uplink (UL): Up to 150 kbps.

Applications of M5Stack Unit NB-IoT

- **Smart Metering:**
  - Monitor utility meters (electricity, water, gas) remotely.
  - Efficiently collect consumption data and optimize resource management.
- **Asset Tracking:**
  - Track valuable assets (e.g., containers, vehicles) across large areas.
  - NB-IoT's long-range capabilities ensure reliable tracking.
- **Remote Monitoring:**
  - Monitor environmental conditions (temperature, humidity) in remote locations.
  - Ideal for agriculture, weather stations, and wildlife tracking.
- **Telemedicine and Healthcare:**
  - Enable remote patient monitoring and health data transmission.

  - NB-IoT's low power consumption suits wearable medical devices.

- **Shared Bicycles and Scooters:**
  - Track and manage shared mobility solutions.
  - NB-IoT's low cost and long battery life make it suitable for this application.

3.8 Microcontroller Program Implementation

3.8.1 NEMA smart pole controller:

3.8.1.1 The operation flow of the controller utilizing NBIoT technology

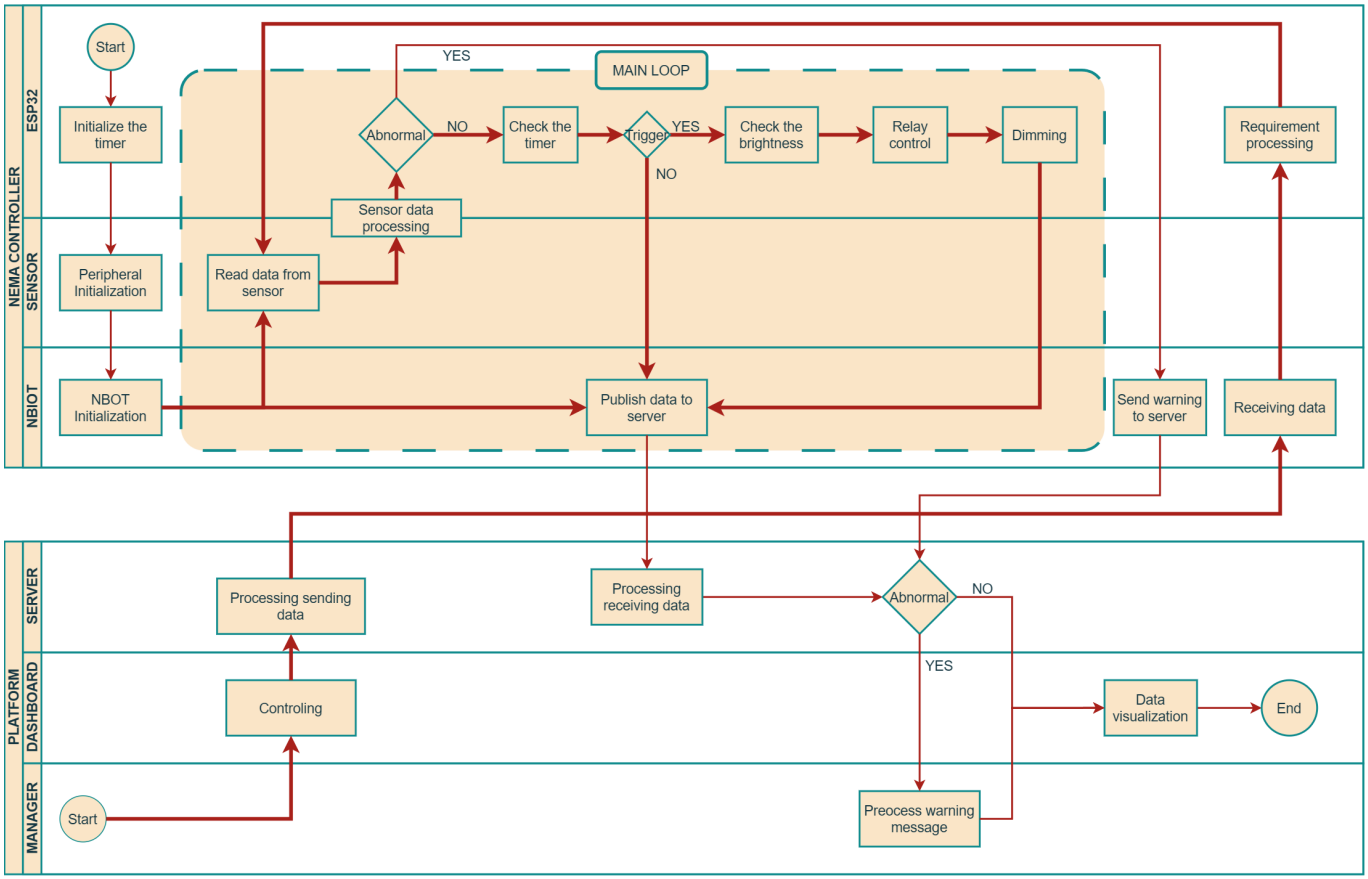


Figure 3.13: The main operation flow of the controller

As the swimlane model above illustrates, there are two separate pools: the NEMA Controller and the Platform.

At the **NEMA controller**, the ESP32 central processing unit will play a primary role in reading sensors, controlling the brightness of the lights, and communicating with the NBIoT module. Meanwhile, the NBIoT module handles two-way communication with the external internet network, specifically transmitting and receiving data with the Platform. We will delve into the "main loop," the primary loop of the controller, to analyze its functions and behaviors:

- Beginning with the initialization of the timer, peripherals such as GPIO sensor, UART pin with the NBIoT module, and initializing parameters as well as connecting to the NBIoT module's platform as shown in Figure 3.13.

- Sensor reading function via the I2C protocol; currently, the existing sensor is the M5Stack's IMU Pro supporting a 6-axis sensor (3-axis rotation, 3-axis acceleration), integrated with temperature and pressure sensors.
- Sensor data processing function: from the data obtained from the IMU Pro, the ESP32 will preprocess the data to check if the light is knocked over or if the controller is overheated or malfunctioning, broken, or damaged so that the pressure changes. If any abnormalities occur, the ESP32 will perform the "send alert to the server" function and turn off the lights; maintenance will be determined by management.
- Timer check function: when no abnormal data occurs, proceed to check the timer to see if it is time for the lights to turn on and how bright they are. If it is not yet time, only publish sensor data to the server.
- Check if the current brightness matches the scheduled time. Depending on the schedule, close or open the relay, then adjust the dimming voltage using PWM to control the appropriate brightness for the lights.
- Synthesize sensor data and current brightness, along with the necessary network parameters, to send to the server.
- The loop will continue after a 5-minute delay.

For the **Platform pool**, the management object has the function of controlling the Dashboard, through which to communicate with the street light controller. The server part will process incoming and outgoing data, then display it visually on the Dashboard.

- Control function: currently, the manager can control the brightness of a specific light or request to read all the data of the light immediately.
- Data processing function: the data is preprocessed at the existing controller, with the limitation that only real-time data is processed. This means that historical data stored in the database will be used to check and predict maintenance time alerts.
- Visualize data: display a map of the lights, display parameters of each light such as network parameters, temperature, pressure, tilt, and display energy usage level. Additionally, there is a brightness control slider, and a button to request immediate data transmission.

3.8.1.2 Central control unit utilizing ESP32

Module LED status

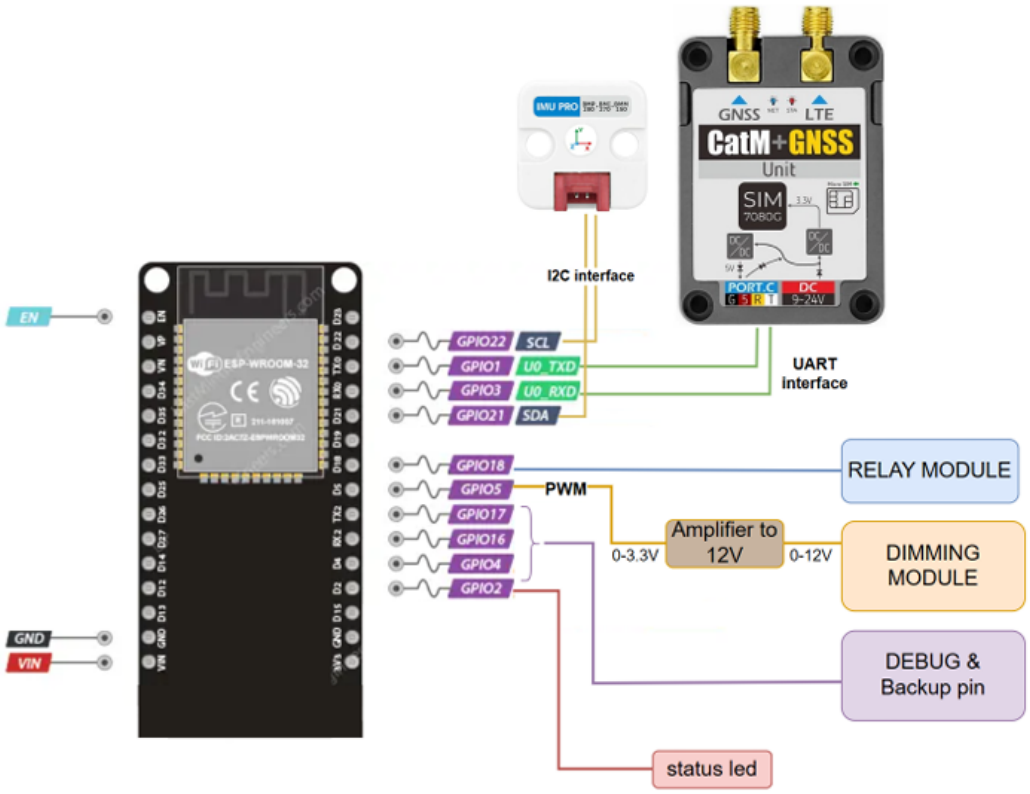


Figure 3.14: The pinout diagram of ESP32 with various modules

First, let's start with the status LED, which is a small but crucial module for checking the system's operation status. The blinking pattern of the small LED helps us quickly determine if the controller is operational or not. The LED states are represented as follows:

- Off: Indicates that the ESP32 is not operating, is faulty, or has no power supply.
- Fast blinking (500ms per blink): Indicates stable connection to the server.
- Slow blinking (1000ms per blink): Indicates normal operation but with a sensor error or inability to connect to the server. To diagnose the error with the sensor or server, you can cross-reference by observing the blinking LED on the NBIoT module mentioned in the NBIoT module implementation section.

Dimming Module

To control the brightness of the streetlights (dimming), we need to adjust the voltage at the D+ and D- terminals in the 0V $\longleftrightarrow$ 10V signal range. To implement this function, two issues need to be addressed simultaneously:

- The voltage at GPIO5 pin is typically only at 0V and 3.3V. Therefore, we need to adjust the output voltage as an analog signal within the 0V $\longleftrightarrow$ 3.3V range.

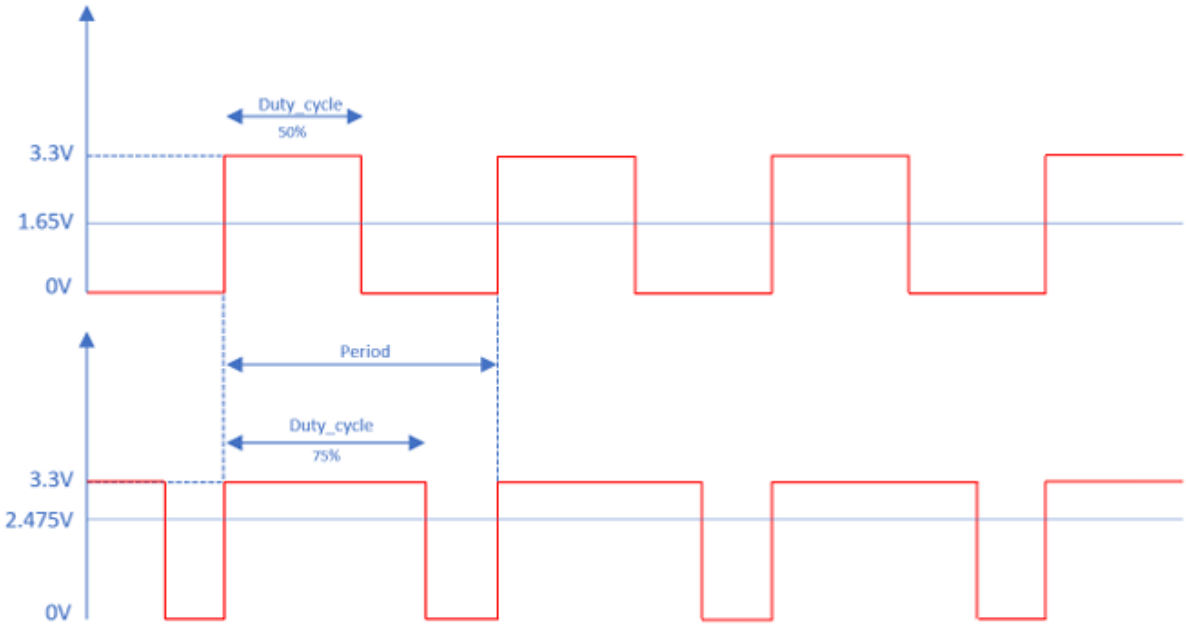


Figure 3.15: Adjusting the pulse width affects the output voltage

- The power supply for the streetlights requires a control voltage level of up to 10V to adjust the brightness. Thus, we need to amplify the output voltage from 0V $\longleftrightarrow$ 3.3V to 0V $\longleftrightarrow$ 12V.

To address the first issue, the team will use Pulse Width Modulation (PWM) technique, which adjusts the load voltage by changing the width of the square wave pulse, thereby changing the voltage.

As shown in the diagram above, the duty cycle is the percentage of time the voltage is at a high level compared to the entire cycle. Based on the duty cycle, we can generate the desired voltage using the formula *Vout = Vmax * DutyCycle*. For example, to generate a voltage of 2.475V, we can use a duty cycle of 75%.

However, in practice, using PWM (Pulse Width Modulation) techniques on microcontrollers can be quite complex. The ESP32 provides the necessary APIs to set up parameters and use PWM easily as follows:

- `ledc_timer_config(ledc_timer_config_t)`: used to configure parameters such as resolution, frequency, speed mode, timer sequence number, and clock source.
- `led_set_duty(uint_32 duty_in_tick)`: to assign the duty cycle to the ESP32. The function requires passing the parameter not in percentage form, but in

the corresponding tick number (calculating tick numbers will be explained below).

- `led_update_duty()`: to activate the assigned duty cycle as mentioned above.

Relay Module However, when the dimming control voltage is 0V, the LED light remains dimly lit. To completely turn off the light, we need another approach. As shown in Figure 3.4, the LED light is controlled by a DALI (Digital Addressable Lighting Interface) module, which is powered through the LOAD wire. Therefore, we need to completely cut off the power from the LOAD wire to the DALI module, and a relay module is typically used to address this issue. The act of completely cutting off the power supply to the light will be controlled by a signal from the ESP32 to the relay module. It is important to note that although the power to the LED light is completely cut off, the controller continues to operate thanks to the power directly supplied from the 220V source.

3.9 Utility software related to project

3.9.1 Android studio

Android Studio is the official Integrated Development Environment (IDE) for Android app development, endorsed by Google. It encompasses a wide array of tools designed to enhance the productivity and efficiency of developers creating Android applications.

Figure 3.16: Our choice for mobile app development, Android Studio, is favored for its comprehensive suite of tools, official Google support, and robust performance profiling, making it the premier environment for crafting cutting-edge Android applications.

3.9.1.1 Advantages of Android Studio

- **Comprehensive Development Tools:** Android Studio provides developers with a powerful code editor, a flexible Gradle-based build system, and a fast emulator for testing apps across a wide range of Android devices.
- **Intuitive User Interface Design:** With the Layout Editor and ConstraintLayout, developers can easily create complex layouts while ensuring compatibility across different screen sizes and resolutions.
- **Performance Monitoring:** Built-in profiling tools help in monitoring an application's memory, CPU, and network usage, leading to better optimized and smoother apps.
- **Google Cloud Integration:** Simplified authentication with Google services and integration with Google Cloud Platform allow for easy addition of cloud-based backend solutions.
- **Instant Run Feature:** Changes to code can be viewed almost instantaneously without the need to restart the app, significantly speeding up the development process.

3.9.1.2 Disadvantages of Android Studio

- **System Resource Intensive:** Android Studio requires a considerable amount of system resources, which can slow down the development on older machines.
- **Steep Learning Curve:** For beginners, the wide array of features and the complexity of the IDE can be overwhelming.
- **Gradle Build Times:** Projects with complex dependencies can experience slower build times, which can impede rapid testing and iteration.
- **Emulator Performance:** While powerful, the emulator can be slow to boot and may run sluggishly on systems with limited resources.
- **Compatibility Issues:** Occasionally, updates bring changes that can cause compatibility issues with existing projects, plugins, or dependencies.

3.9.1.3 Conclusion

Android Studio, while providing a robust environment for app development with many beneficial features, also poses challenges, particularly to those with less powerful hardware or those new to Android app development. However, its integration with Google's ecosystem and its comprehensive toolset make it an indispensable asset for serious Android developers.

3.9.2 Firebase Realtime Database

The Firebase Realtime Database is a NoSQL database hosted in the cloud, allowing you to store and synchronize data in real-time among your users. When you build cross-platform apps with our Apple platforms, Android, and JavaScript SDKs, all of your clients share one Realtime Database instance and automatically receive updates with the newest data.

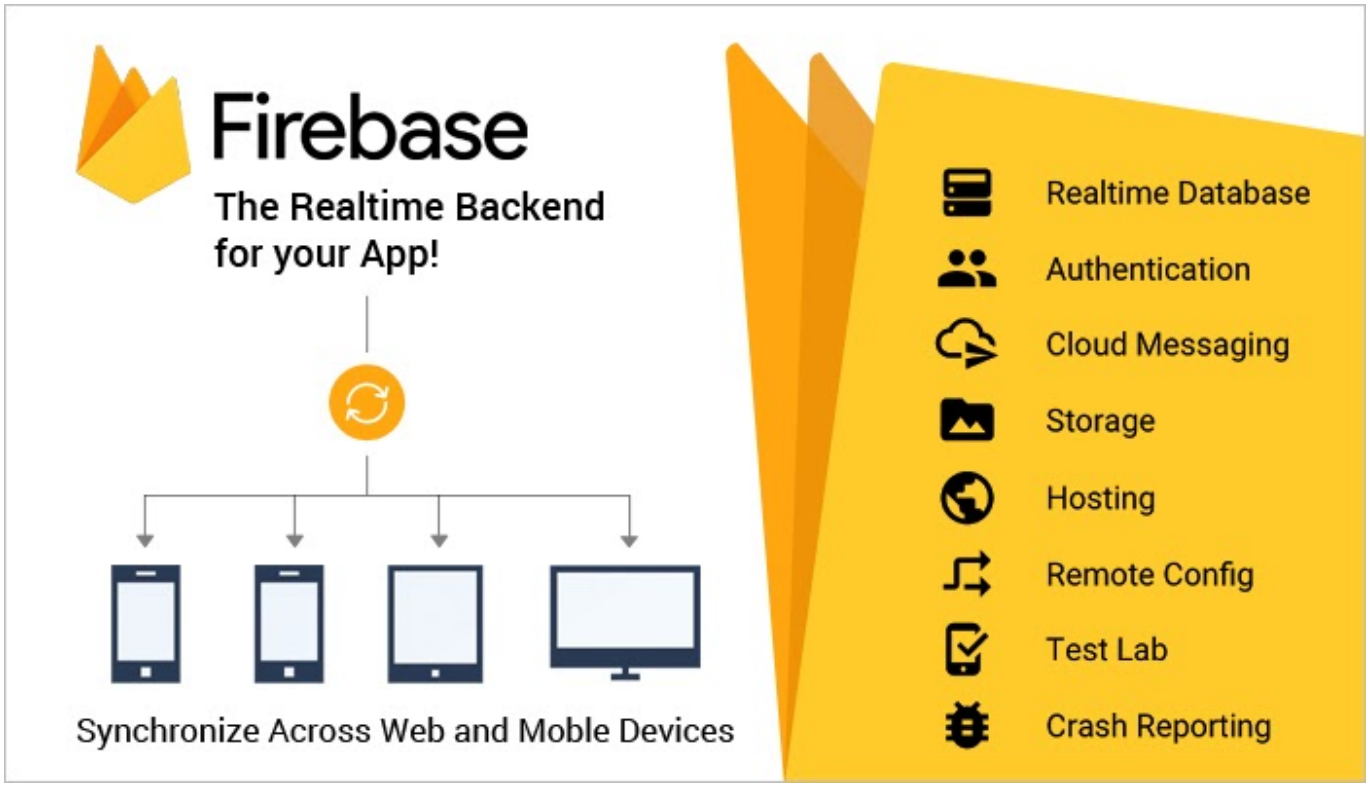


Figure 3.17: We utilize Firebase for storing sensor data, owing to its resource efficiency, swift response times, and flawless integration with Android Studio. Firebase enhances our project with a robust and adaptable backend solution.

3.9.2.1 Key Features

- **Real-time Synchronization**: Data is synchronized across all clients instantly and automatically. When you update data, every connected device receives that update within milliseconds.
- **Offline Support**: Firebase apps remain responsive even when offline as the Firebase Realtime Database SDK persists your data to disk. Once connectivity is reestablished, the client device receives any changes it missed.
- **Accessible from Client Devices**: The database can be accessed directly from a web browser or mobile device, simplifying the development process by reducing the need

for a server-side layer.

- **Scalability**: Despite being a NoSQL database, it provides scalability to accommodate your application's growth with ease.

3.9.2.2 Advantages

- **Effortless Setup**: Developers can set up and start using the database without the hassles of setting up a server.
- **Data Storage and Retrieval**: Provides APIs for reading and writing data, listening for changes, and securing data access with robust authentication and authorization.

3.9.2.3 Disadvantages

- **Data Structure Limitations**: As a NoSQL database, it requires careful planning of data structure to ensure efficient data retrieval and updates.
- **Query Limitations**: It does not support sophisticated querying like SQL databases, which can be a limitation for complex data structures.
- **Cost Scaling**: Depending on the project, as the number of users grows, the cost can increase significantly, especially if not managed properly.

3.9.2.4 Use Cases

Firebase Realtime Database is ideal for building real-time collaborative applications, enabling real-time chat applications, and syncing various states across user devices, like online presence or real-time gaming.

3.9.2.5 Conclusion

Firebase Realtime Database offers a unique set of features that make it an attractive choice for developers looking to build real-time, collaborative web and mobile applications. Its real-time syncing capabilities, combined with ease of use and setup, make it a powerful tool for developers. However, considerations around data structuring and cost management should be taken into account when deciding to use Firebase Realtime Database for your application.

CHAPTER 4

CONCLUSION AND SYSTEM EVALUATION

4.1 Project Overview

Throughout the course of the Smart Pole project, significant progress has been made in several key areas, each contributing to the transformation of urban infrastructure through intelligent technology integration:

- **Four-Layer IoT Architecture:** The project developed a robust, four-layer architecture that underpins the entire smart pole system. This architecture comprises the perception layer, network layer, platform layer, and application layer, each designed to handle specific functionalities ranging from data collection to user interaction. This structured approach ensures scalability and flexibility, allowing the system to integrate new technologies as they become available.
- **Object Tracking with YOLOv9 and DeepSORT:** The integration of advanced object tracking capabilities using YOLOv9 and DeepSORT represents a major technical milestone. These technologies enable the smart poles to perform real-time surveillance and monitoring of urban spaces. By accurately tracking the movement of vehicles and pedestrians, the system enhances public safety and improves the city's traffic management capabilities, providing a proactive approach to urban security and logistics.
- **Remote Light Control via NBIoT:** Implementing remote control for street lighting through Narrow Band Internet of Things (NBIoT) technology has led to significant advancements in energy management. This feature allows city administrators to control lighting based on real-time data, such as traffic density and pedestrian patterns, thereby optimizing energy usage and reducing operational costs. The implementation of NBIoT highlights the project's commitment to energy efficiency and sustainability in smart city management.

4.2 Project result

4.2.1 Remote Light Control via NBIoT

This section outlines the results of the implementation of the Smart Pole system, demonstrating the effectiveness of the hardware integration and functionality.

4.2.1.1 Electronic Components Inside the Smart Pole

The first image illustrates the electronic components inside the Smart Pole, highlighting the control systems and sensors that facilitate its operations.

Figure 4.1: Electronic components inside the NEMA light device

4.2.1.2 Simulation and Real Product

The second image compares the simulated model of the Smart Pole with the actual product installed in an office environment. This visualization showcases the design accuracy and real-world application of the Smart Pole.

Figure 4.2: Comparison between simulated model and actual Smart Pole installation

4.2.1.3 Demonstration of Functionality: Light OFF and ON

The third image demonstrates the functionality of the Smart Pole's lighting system, showing the light turned off and on. This test confirms the operational status and responsiveness of the integrated lighting system.

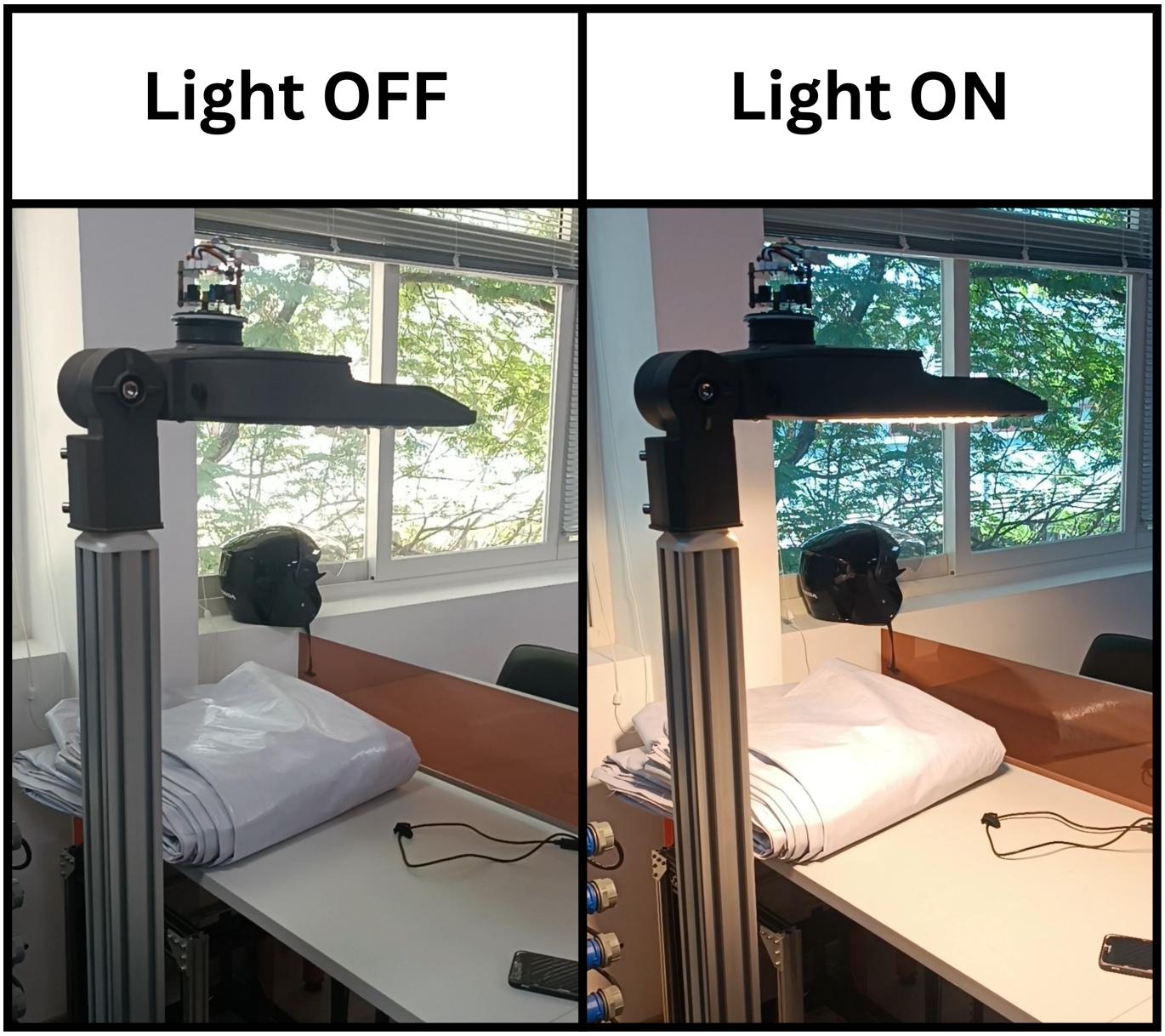


Figure 4.3: Smart Pole operational demonstration with light off and on

4.2.2 Air Quality Data from Smart Pole Sensors

The Smart Pole is equipped with advanced sensors that continuously monitor various environmental parameters. This section provides an overview of the types of data these sensors collect, which are crucial for assessing urban air quality and environmental conditions.

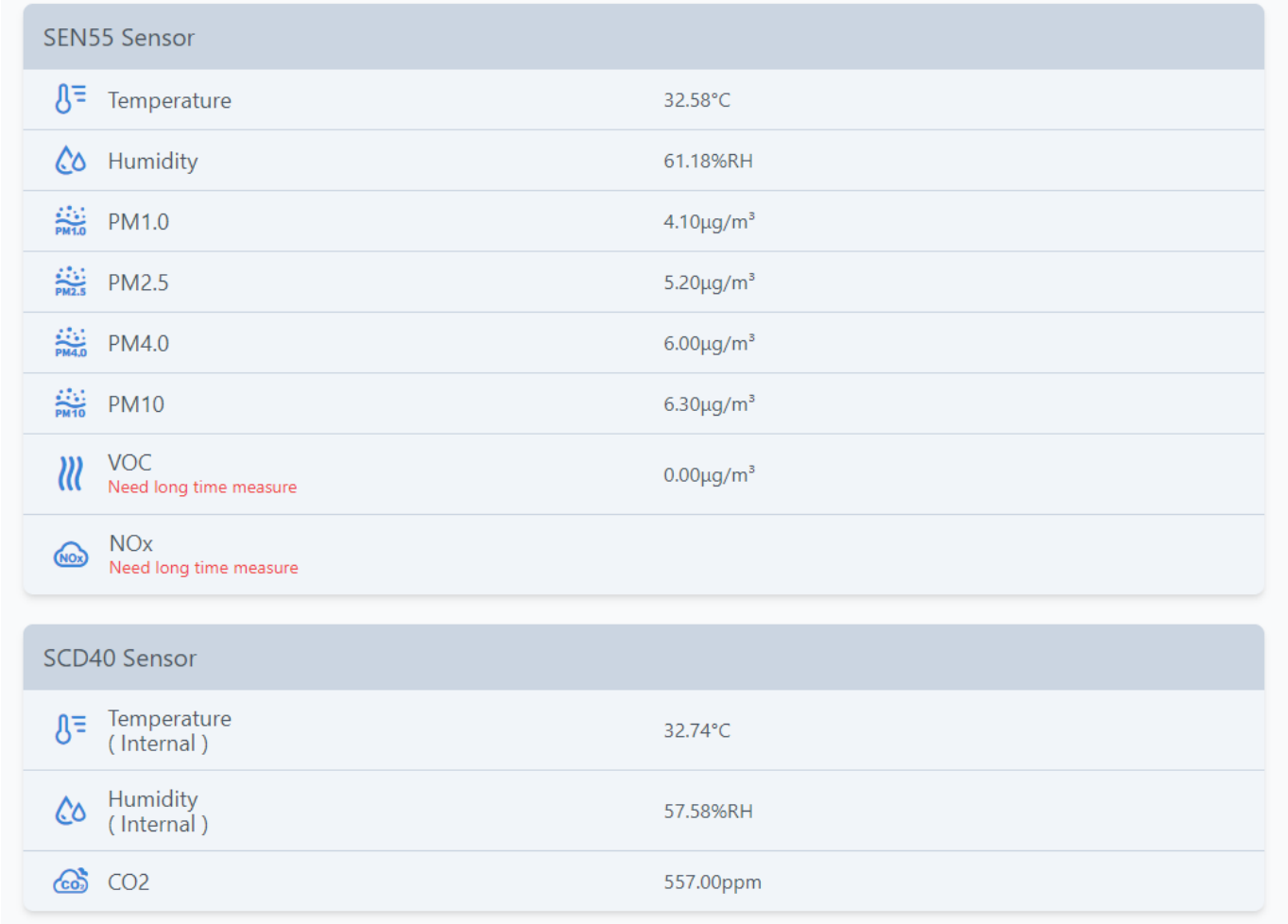


Figure 4.4: Each sensor in the Smart Pole is designed to provide real-time data crucial for the active management of urban environments, helping city planners and residents make informed decisions about health and safety based on air quality indices.

4.2.3 SEN55 Sensor Capabilities

The SEN55 sensor in the Smart Pole system measures a comprehensive set of air quality and meteorological parameters, including:

- Temperature
- Humidity
- Particulate Matter across different sizes (PM1.0, PM2.5, PM4.0, PM10)

- Volatile Organic Compounds (VOCs) - noted that prolonged measurement periods may be required for accurate readings
- Nitrogen Oxides (NO_x) - also requiring extended measurement times to ensure precision

4.2.4 SCD40 Sensor Capabilities

In addition to the external environmental measurements, the Smart Pole features the SCD40 sensor for internal monitoring, which includes:

- Internal Temperature
- Internal Humidity
- Carbon Dioxide (CO_2) levels

4.2.5 Object Tracking via SmartPole Camera

4.2.5.1 Object Tracking Using Traffic Video

We showcase the application of the Yolov9 model to effectively track vehicles in a busy urban traffic scene. The displayed image demonstrates the model's robust capability to identify and track multiple cars simultaneously under varying weather conditions. Each vehicle is accurately bounded by a colored box, labeled with a unique identifier, ensuring clear visibility of tracking results. This capability is crucial for applications in traffic management, autonomous driving, and security surveillance, illustrating the model's precision and reliability in real-time vehicle tracking. This test run ensures that the model operates as expected. Following this verification, we will integrate the model with the camera system of a smartPole to enhance our urban traffic monitoring capabilities. This integration aims to leverage real-time data for more effective traffic management and enforcement, contributing to smarter city infrastructure.

Figure 4.5: Real-Time Tracking of Multiple Vehicles in Urban Traffic

4.2.5.2 Object Tracking Using Hikvision SmartPole Cameras

Building on the initial success of our object tracking model as discussed previously, we now turn our attention to a practical application within an educational environment. The following image illustrates the application of our enhanced tracking technology in room 301B9

at HCMUT. This segment highlights the effectiveness of our system in a real-world setting, where it can identify and monitor individuals, providing valuable data for security and space management. This technology not only enhances security but also supports behavioral studies and space utilization analytics in academic settings.

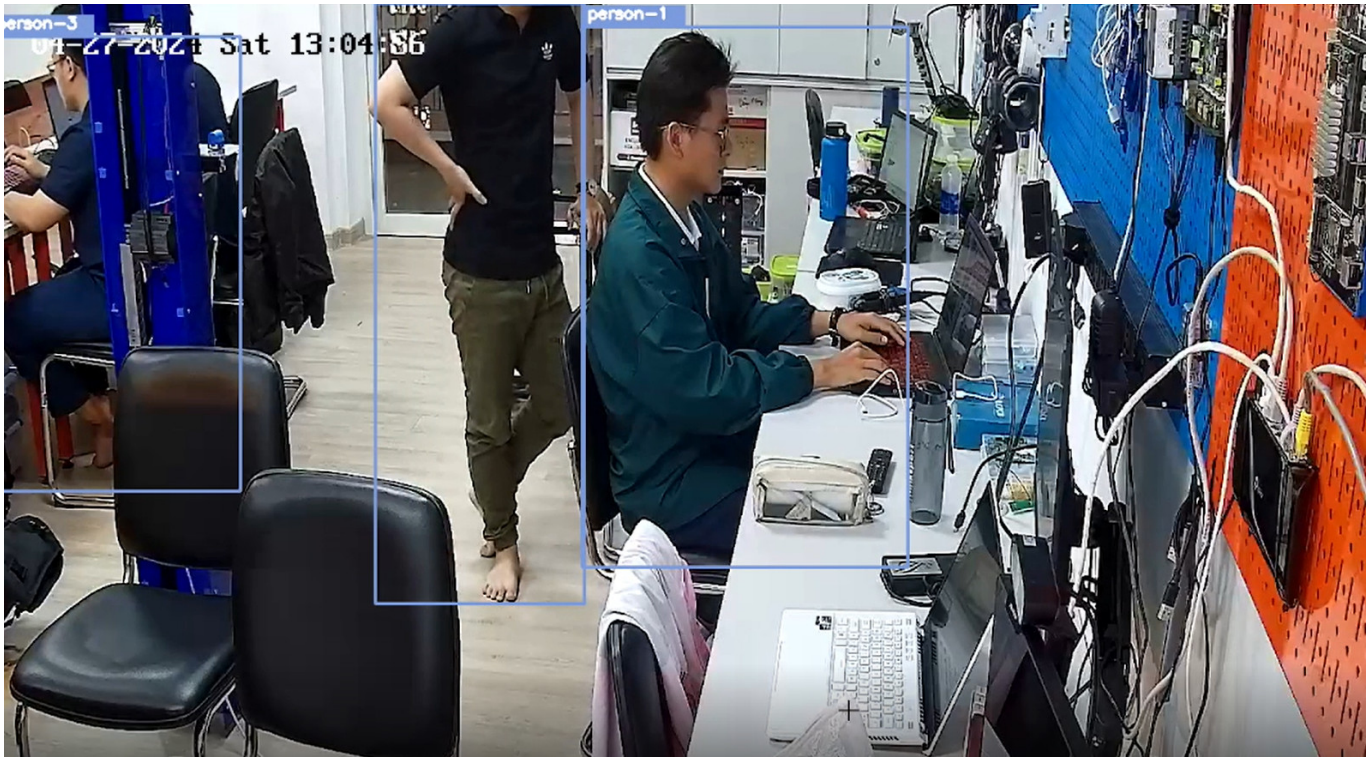


Figure 4.6: Utilizing Hikvision SmartPole Cameras for Real-Time Tracking in HCMUT Room 301B9

4.3 The Limitations in the Project

While the project has made commendable progress, several limitations have been identified that could impact the full realization of its potential:

- **Energy and Latency Measurement:** Comprehensive evaluations of the system's energy consumption and latency have not yet been conducted, especially across different connectivity standards. These measurements are essential for assessing the efficiency and responsiveness of the smart poles, which are critical metrics for the scalability of the system across larger urban areas.
- **Application Layer:** The project team's limited experience in application development and user interface design has constrained the implementation of a robust and intuitive user application. This application is crucial for enabling efficient management and real-time interaction with the smart pole system by city administrators and maintenance teams.
- **Device Layer:** The selection process for an appropriate microcontroller unit (MCU) that can manage not only the advertisement display but also critical functionalities such as emergency response and network broadcasting is still ongoing. This selection is vital for ensuring the reliability and effectiveness of the integrated systems within each pole.
- **Connectivity Layer:** Although a suitable model for detecting people and vehicles has been developed, establishing a seamless method for real-time communication between the cameras and the video processing unit has proven challenging. Enhancing this layer is crucial for the real-time data processing and analytics needed for the smart pole's operational success.

These identified limitations outline the need for targeted research and development efforts to address the gaps in technology implementation and system integration. Moving forward, the project aims to refine these areas to enhance the functionality and reliability of the smart poles, thereby ensuring their effective deployment in enhancing urban living environments.

CHAPTER 5

EXPANSION AND FUTURE DEVELOPMENT

5.1 Enhancements to the Smart Pole System

As we advance the capabilities of the Smart Pole System, we focus on not only refining existing features but also incorporating new functionalities that align with the needs of modern urban environments and technological advancements.

5.1.1 User Interface Improvements

One of our key objectives is to enhance the user interface (UI) of the management software used for controlling and monitoring the Smart Pole System. The improved UI will offer a more intuitive and user-friendly experience, enabling city administrators and technicians to interact more efficiently with the system. This includes streamlined navigation, real-time data visualizations, and easier access to control features, all designed to facilitate quicker decision-making and responsiveness to urban needs.

5.1.2 Integration of Television on Demand (TVOD)

Incorporating TVOD capabilities into the digital displays of Smart Poles will transform how public information and entertainment are delivered in urban spaces. This feature allows for the broadcasting of a wide range of content, including news, emergency broadcasts, advertisements, and entertainment, directly to the urban populace. TVOD will enable city administrators to deploy targeted content based on location and time, enhancing the way information is disseminated and consumed in public areas.

5.1.3 Leveraging Artificial Intelligence for Data Processing

To further enhance the functionality of the Smart Pole System, we plan to integrate Artificial Intelligence (AI) technologies to optimize the processing and analysis of vast amounts of data collected by the system's sensors. AI algorithms will be used to analyze patterns in environmental data, traffic flows, and public interaction. This will not only improve the accuracy of real-time responses but also enable predictive analytics for better planning and resource allocation. AI-enhanced features will include anomaly detection in traffic or environmental conditions and adaptive lighting and display content based on real-time crowd data analysis.

5.2 Future Directions

Looking forward, the Smart Pole System is set to evolve with ongoing technological innovations and the changing dynamics of urban development. Future iterations of the system might include advanced connectivity options such as 5G integration, which would significantly increase data transfer speeds and enable more sophisticated IoT applications. Additionally, sustainability will be a focus, with increased use of renewable energy sources like solar panels and integration of electric vehicle (EV) charging stations directly into the Smart Poles.

5.3 Collaborative Opportunities

We aim to expand our collaboration with technology developers, urban planners, and governmental agencies to explore new applications and integrations for the Smart Pole System. By fostering these partnerships, we can ensure that the system remains at the forefront of smart city innovations, adapting to the needs of cities and their inhabitants worldwide.

5.4 Conclusion

The expansion and continuous development of the Smart Pole System represent our commitment to building smarter, more responsive urban environments. Through enhancements in user interface, integration of TVOD, and the application of AI in data processing, we are setting new standards for what urban infrastructure can achieve. As we move forward, our focus will remain on innovation, usability, and sustainability, ensuring that the Smart Pole System continues to lead in transforming urban landscapes.

Bibliography

[1] Timothy Chou, *"Precision: Principles, Practices and Solutions for the Internet of Things"*, McGraw Hill Education (India) Private Limited, 2017.

[2] Robert C.Martin, *"Clean Code - A Handbook of Agile Software Craftsmanship"*, Pearson Education, 2008

[3] Se Dong Min, *"A DPN (Delegated Proof of Node) Mechanism for Secure Data Transmission in IoT Services"*, 2016. Available: ResearchGate, accessed on: 5/3/2024.

[4] Dr. Samer Jaloudi, *"MQTT for IoT-based Applications in Smart Cities"*, 2019. Available: ResearchGate, accessed on: 5/3/2024.

[5] VNA/VNS, *"Thủ Đức to develop into innovative hub, smart city"*. Available online: vietnamnews.vn, accessed on 5/5/2024.

[6] Đảng cộng sản Việt Nam, *"Viettel phủ sóng công nghệ NB-IoT toàn TP Hồ Chí Minh"*. Available online: dangcongsan.vn, accessed on 5/5/2024.

[7] Gaudenz Boesch, *"YOLOv9: Advancements in Real-time Object Detection (2024)"*. Available online: viso.ai/computer-vision/, accessed on 5/5/2024.

[8] Chien-Yao Wang, I-Hau Yeh, Hong-Yuan Mark Liao, *"YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information"*. arXiv preprint arXiv:2402.13616 (2024). accessed on 5/5/2024.

[9] Lakshitha Vimuth, *"Mastering YOLOv9: The Ultimate Guide for Enhanced Object Detection"*. Available online: lvimuth.medium.com, accessed on 5/5/2024.

[10] ML Wires, *"YOLOv9, the latest breakthrough in real-time object detection"*. Available online: ML WIRES, accessed on 5/5/2024.

[11] Deci Team, *"The History of YOLO Object Detection Models from YOLOv1 to YOLOv8"*. Available online: deci.ai, accessed on 5/5/2024.

[12] Ankan Ghosh, *"YOLOv9: Advancing the YOLO Legacy"*. Available online: learnopencv, accessed on 5/5/2024.

[13] Pamudu123 Ranasinghe, *"YOLOv8 Comparison with Latest YOLO models"*. [Online]. Available: `https://youtu.be/QOC6vgnWnYo?si=WMdpj5dxVjb0UwfZ`, accessed on 5/5/2024.

[14] Sanyam, *"Understanding Multiple Object Tracking using DeepSORT"*. Available online: learnopencv, accessed on 5/6/2024.

[15] Shoeb Ahmad, *"Object Tracking with DeepSORT | By Shoeb Ahmad"*, 2023. Available: ResearchGate, accessed on: 5/6/2024.

[16] Super Annotate, *"Computer vision in sports: applications, challenges, and sports datasets"*,2023. Available online: Super Annotate, accessed on 5/6/2024.

[17] Sujan Shrestha, *YOLOv9_DeepSORT*. [Online]. Available: `https://github.com/sujanshresstha/YOLOv9_DeepSORT`, accessed on 5/6/2024.