

**COMPUTER VISION-BASED APPROACH TO REDUCE
ENERGY CONSUMPTION IN COMMERCIAL BUILDINGS**

BY

MUHAMMAD IBRAHIM SOBIH

A Thesis Presented to the
**COLLEGE OF GRADUATE AND INTERDISCIPLINARY STUDIES
KING FAHD UNIVERSITY OF PETROLEUM & MINERALS**

DHAHRAN, SAUDI ARABIA

1963 ١٣٨٣
In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

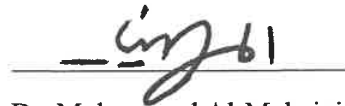
ELECTRICAL ENGINEERING

NOV 2023

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS
DHAHRAN- 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by Muhammad Sobih under the direction of his thesis advisor and approved by his thesis committee, has been presented and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE IN ELECTRICAL ENGINEERING**.



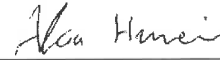
Dr. Mohammed Al-Muhaini
Department Chairman



Dr. Mohamed Abido
(Advisor)



Dr. Suliman Al-Homidan
Dean of Graduate Studies



Dr. Alaa Houssin
(Member)

Date



Dr. Aboubakr Salem
(Member)



© Muhammad Sobih

2023

Dedication

To my family, whose unwavering support and encouragement have been the guiding light throughout this journey. Your love, patience, and belief in me have fueled my determination and inspired me to reach for the stars. I dedicate this thesis to you, with heartfelt gratitude for always being my pillars of strength.

To my mentors, and colleagues, whose guidance, wisdom, and encouragement have enriched my academic journey. Your support and camaraderie have made the challenges more manageable and the victories more meaningful. This thesis is a testament to the power of collaboration and the importance of community in achieving our goals.

ACKNOWLEDGMENTS

First of all, I would like to thank my advisor Dr. Mohammad Abido for his continues support during the past two years. Without his guidance, this wok would not have been possible. Second, I would like to extend my thanks and appreciation to the MSc committee, mainly Dr. Abo Bakr Salem , and Dr. Alaa Hussian for their efforts in reviewing this work. Finally, I want to express my gratitude to my family members for providing me with unlimited support and continuous encouragement throughout my years of study.

Table of Contents

ACKNOWLEDGMENTS.....	v
LIST OF TABLES.....	viii
LIST OF FIGURES	ix
LIST OF ABBREVIATIONS.....	x
NOMENCLATURE.....	xi
ABSTRACT	xiii
ملخص الرسالة.....	xv
CHAPTER 1 INTRODUCTION.....	1
1.1 Overview	1
1.2 Motivation & Objectives	5
1.3 Thesis Contributions.....	6
1.4 Thesis Organization.....	7
CHAPTER 2 LITERATURE REVIEW.....	9
2.1 DSM Overview.....	9
2.2 Hardware Occupancy Methodologies.....	16
2.3 Deep Learning Methodologies	17
2.4 Summery	19
CHAPTER 3 PROPOSED METHODOLOGY.....	21
3.1 Deep Learning	21
3.2 Computer Vision	26
3.3 Proposed System Architecture	30
3.4 Summery	57
CHAPTER 4 TECHNOLOGY ASSESSMENT.....	58
4.1 Computation Hardware.....	58
4.2 Cameras	67
4.3 Actuators/ Thermostatically Controlled Loads	69
4.4 Software Libraries.....	73
4.5 Summery	75
CHAPTER 5 IMPLEMENTATIONS & RESULTS.....	76

5.1	Power System	76
5.1.1	Environment Modeling.....	78
5.2	Summery	99
CHAPTER 6 CONCLUSION		101
6.1	Conclusion.....	101
Appendex.....		105
References.....		111
Vitae		118

LIST OF TABLES

Table 4.1 Hardware Specification	67
Table 5.1 Parameters Used to Calculate the Power Consumption.....	82
Table 5.2 Energy Consumption In kWh For Different Setpoint Ac Temperature Concerning Three Different Outside Temperatures for Three Different Sub-Periods.....	86
Table 5.3 Shows the Total Power Consumption in kWh for Each Scenario for Different Ac Setpoint Ranging from 20° to 25° In Case 1.	87
Table 5.4 Shows the Total Energy Consumption in kWh for Each Scenario for Different Ac Setpoint Ranging from 20° to 25° In Case 2.	88
Table 5.5 Shows the Total Power Consumption in kWh for Each Scenario for Different Ac Setpoint Ranging from 20° to 25° In Case 1.	98

LIST OF FIGURES

Figure 2.1 Description of the proposed system components	20
Figure 3.1 Shallow neural network	21
Figure 3.2 The operation of the neuron where the inputs are multiplied by the weights and summed together. After that, the output will be fed to the activation function to produce the output.....	24
Figure 3.3 Each kernel in the convolutional layer slides on the input, multiplying the values of the kernel by the portion that is passed through the input to produce the output.....	27
Figure 3.4 The max-pooling operation for an input image with size of 4*4.....	28
Figure 3.5 Convolution Neural Network (CNN) to classify objects like human [58]	29
Figure 3.6 SSD model architecture composed of several convolutional layers to detect objects [51].....	40
Figure 3.7 Object detection, identification, and tracking.....	52
Figure 3.8 Tracking system occlusions robustness.	53
Figure 3.9 Counting region in the workplace	54
Figure 4.1 IoT edge device architecture.....	60
Figure 4.2 Workstation & server architecture.....	62
Figure 4.3 Cloud computing architecture.....	65
Figure 4.4 Targeted laboratory prototype for the proposed system.	72
Figure 5.1 Sample of single scenario of occupancy status for each room in the hotel building. Blue dot refers to occupied room.....	84
Figure 5.2 Room Model Block Diagram	94

LIST OF ABBREVIATIONS

CNN	:	Convolutional Neural Networks
CV	:	Computer Vision
DR	:	Demand Response
DSM	:	Demand Side Management
EE	:	Energy Efficiency
R-CNN	:	Region-based Convolutional Neural Networks
NN	:	Neural Networks
SSD	:	Single Shot Detectors

NOMENCLATURE

A_{wall}	The area of a wall
A_{window}	The area of a window
$A_{ceiling}$	The area of a ceiling
Blower cmd	The on/off thermostat signal
$\frac{dQ}{dt}$	The amount of energy supplied by the AC
g_j^m	The j-th ground truth box with m coordinates.
G_i	The amount of energy needed to keep a space in certain temperature T_i
H_{solar}	The solar radiation thermal power measured
L_{conf}	The confidence loss function between the ground truth boxes and detected boxes.
L_{loc}	The localization loss function of the bounding boxes coordinates between the ground truth and detected boxes.
l_i^m	The i-th predicted box with m coordinates.
R_{wall}	The thermal resistance of wall
R_{window}	The thermal resistance of window
$R_{ceiling}$	The thermal resistance of ceiling

$SHGC$	The solar heat gain coefficient
T_{err}	The difference between the set point of the thermostat and the actual room temperature
T_{out}	The outdoor temperature
T_i	The indoor temperature
x_{ij}^p	Indicator for matching i -th default box to the j -th ground truth box of category p .

ABSTRACT

Full Name : Muhammad Sobih
Thesis Title : Technology Assessment of Computer Vision-based Approach to Reduce Energy Consumption in Commercial Buildings
Major Field : Master of Science in Electrical Engineering
Date of Degree : Nov 2023

Advancements in technology have led to a significant increase in global electrical power consumption, driven by factors such as appliance efficiency and evolving user behavior. As demand continues to escalate, particularly in buildings where heating and cooling systems contribute substantially to energy consumption, the need for efficient energy management becomes paramount. In response to this challenge, researchers have turned their attention to Demand Side Management (DSM) programs, which focus on optimizing energy usage rather than solely increasing supply. These programs consider various factors such as price rates, building design, and the utilization of electrical equipment to achieve optimal results.

Among the key factors influencing energy reduction in DSM programs, occupancy status emerges as a critical consideration. However, accurately determining occupancy remains a challenge, with existing methods lacking precision, especially at the room/office level. Addressing this gap requires the development of accurate methodologies that are adaptable across different building types and seamlessly integrate with existing infrastructure, thereby minimizing implementation costs.

The primary objective of this thesis is to develop a reliable system for detecting occupancy in buildings using computer vision and image processing techniques. By leveraging surveillance camera systems, the proposed system aims to analyze camera feeds and extract occupancy data for specific locations within the building. This data will be utilized to adjust AC thermostat set points, consequently reducing energy consumption.

Unlike traditional occupancy detection methods such as motion sensors, which often prove unreliable, a computer vision-based approach offers greater precision and reliability. Integrated with existing surveillance systems, this system eliminates the need for additional hardware or sensors, thereby minimizing installation and maintenance costs. Furthermore, its real-time capabilities enable immediate adjustments to AC thermostat set points based on occupancy status, maximizing energy savings.

The anticipated benefits of the proposed system include more accurate occupancy data, cost savings, and real-time control over energy usage. To assess its effectiveness and robustness, multiple case studies, simulations, and tests have been conducted, providing valuable insights into its practical implementation and potential impact on energy efficiency in buildings. Through this thesis, the aim is to develop a precise, cost-effective, and real-time occupancy detection mechanism that can significantly optimize energy consumption and contribute to sustainability efforts in the built environment.

In summary, the proposed thesis aims to develop an accurate occupancy detection mechanism using computer vision and image processing techniques. The system will be integrated with the existing surveillance camera system to extract occupancy data, which will be used to adjust the AC thermostat set point temperature and reduce energy consumption.

ملخص الرسالة

أدى التطور في المجال التقني إلى زيادة استهلاك الطاقة الكهربائية على مستوى العالم. تساهم العديد من العوامل في هذا الاتجاه، بما في ذلك تحسين كفاءة الأجهزة الكهربائية وتغيير سلوك المستهلك. تعد المباني من أكثر الموارد التي تستهلك الطاقة الكهربائية، حيث يُصرف أكثر من نصف هذه الطاقة على أجهزة التبريد والتدفئة. يركز العديد من الباحثين على تطوير طرق جديدة لاستخدام الطاقة بشكل أكثر كفاءة من خلال التركيز على جانب المستهلك وليس جانب الإنتاج. تسمى هذه الطرق بإدارة جانب المستهلك، وتتضمن مجموعة متنوعة من التقنيات مثل أنظمة الطاقة الشمسية وأجهزة حفظ الطاقة.

أحد أهم العوامل التي تساهم في تقليل استهلاك الطاقة هو عدد الأشخاص الذين يشغلون المكان. تفنقر معظم برامج إدارة الطاقة إلى دقة معلومات عدد الأشخاص الذين يشغلون المبنى، مما يحد من فعاليتها. تهدف هذه الدراسة إلى تطوير نظام دقيق لتحديد عدد الأشخاص الذين يشغلون المبنى باستخدام أنظمة تحليل الفيديو. سيتم ربط النظام بنظام الكاميرات الأمنية للمبنى، وسيستخدم تحليل الصور لاستخراج معلومات الإشغال. ستستخدم معلومات الإشغال للتحكم في أنظمة التبريد والأنظمة الكهربائية الأخرى لتقليل استهلاك الطاقة.

تم اختبار النظام المقترح في عدد من البيئات المختلفة، وتم إثبات فعاليته في تقليل استهلاك الطاقة. أظهرت النتائج أن النظام يمكن أن يقلل من استهلاك الطاقة بنسبة تصل إلى 20%. يوفر النظام المقترح طريقة جديدة وفعالة لتحديد عدد الأشخاص الذين يشغلون المبنى. يمكن استخدام هذا النظام لتقليل استهلاك الطاقة وتحسين كفاءة الطاقة في المباني.

CHAPTER 1

INTRODUCTION

1.1 Overview

Industrialization and technological advancement have raised the increase in power demand all around the world. Many factors contribute to the increase in the energy consumption such as the user behavior, appliances efficiency, and many more. Meeting the demand by increasing the supply is a very expensive solution and governments cannot keep up with the rate of increase. Buildings are the most demanding source of electricity consumption and about half of its consumption is used on heating and cooling applications. Therefore, it is urgent to investigate new ways to optimize the current capacity of energy resources to cover more demand. Instead of optimizing the energy from the supply side, new Demand Side Management programs have been presented to reduce the energy consumption. These programs aim to reduce the energy consumption taking in consideration many factors such as the price rates, buildings design, electrical equipment and storage devices, distributed energy resources, and other factors. In the presented DSM programs, occupancy status detection is a spot of uncertainty. Accurate occupancy information can lead to better utilization of the energy resources. Different methodologies are used to predict the occupancy status of a building such as mathematical models, different sensor technologies, etc. However, they lack high accuracy, and setup complexity in real environments.

High demand for electricity has increased throughout the time. Due to the rapid urbanization trend, humanity became more electricity dependent. The demand for electricity is growing as the national population increases specially, with the growth of technology in modern societies [1]. Some studies suggest that world marketed energy consumption is projected to increase by 2030 to 33% [2] and by 56% by 2040 [3]. To cover the peak demands many countries have a capacity reserve of 20% [1], which will not be able to keep up with the rate of demand increase. Additionally, increasing the power generation involves generating more CO₂ emissions. According to The International Energy Agency (IEA) 70% of the world's energy production is through burning of fossil fuels, primarily coal (42%), and gas (21%) [1].

Nearly 40% of the energy consumption in many countries goes to buildings, and 60% of it is used in heating and cooling purposes [4]. Saudi Arabia is one the highest countries in building energy consumption where it poses 80% of the power consumption and 50% of it is used in heating and cooling applications [5]. The focus in this research will be on studying & reducing the energy consumption in buildings.

Building new power plants to cover the demand increase has many drawbacks among them the high cost and CO₂ emissions. Instead, researchers developed demand side management (DSM) algorithms to cover more demand with the same power capacity. These algorithms are based on optimizing the existing power resources by studying the factors that contribute to the high demand in the power consumption.

To build DSM algorithms, understanding the factors that contribute to increasing the energy consumption is very important. IEA and energy in the buildings and communities'

program (EBC) identified climate, buildings envelope, buildings energy and service systems, indoor sign criteria, building operation and maintenance along with occupant behavior to be the parameters that mostly contribute to the energy consumption in buildings [3]. In addition, the electricity market, pricing scheme, and the whole sale are also very important parameters in developing DSM schemes. Involving more factors will lead to better energy management and reduction in energy consumption. However, the trade-off will be the system complexity.

One of the big challenges in DSM programs is the uncertainty of the building's occupancy, price rates, and renewable power generation. Occupancy uncertainty is one of the major factors that contribute to higher energy consumption. Some of the presented DSM techniques rely on statistical approaches to predict the number of occupants. Some others rely on the history of data of the building to predict the number of people inside. These approaches lack the occupancy information on room/office/place level. Thus, determining the occupancy of each space/room/office in the building requires accurate methodologies that can operate irrespective of the building type and should be easy to integrate with infrastructure of the building to lower the cost of the implementation.

The occupancy information is very important when it comes to the demand side management. Accurate occupancy information enables extreme power consumption savings. This information will help boost the DSM algorithms to reduce energy consumption. In summary, the following issues need to be addressed.

- Thermal modelling of buildings to study the effect of the changing the AC set point temperature on the power consumption.

- The occupancy mechanism that eliminates the uncertainty of the occupancy information.
- The occupancy information that gives the highest level of details for each place and part of the building irrespective of the building type and usage.
- The integration to the existing surveillance infrastructure to reduce the implementation cost in real world environments.
- The integration to the controller device that manages load consumption based on the occupancy information.

Given this brief background, this work proposes a computer vision technique to monitor the occupancy of a building to give an accurate information about the occupancy of each room/place/office in a building at each instant. The proposed system is a retrofitting system, which means that it will use the already existing surveillance camera feeds without the need to add extra cost. A controlling strategy will be developed based on the occupancy information given by the occupancy detector. The controlling strategy for the AC thermostat set point temperature will be reset to a certain value. Different thermal models of the buildings are used to study the effect of changing the AC thermostat set point temperature on the energy consumption. A typical building will be taken as a case study to show the effect of changing the set point. Finally, a laboratory experiment will be conducted where the proposed system will be installed to monitor the occupancy and based on the collected information, the controller will take the proper actions to adjust the AC temperature. A real-time digital simulator (RTDS) is going to be used to simulate the AC loads.

1.2 Motivation & Objectives

The classical techniques that solve the occupancy problem make use of statistical models to predict the building occupancy profile or rely on hardware detector technologies. The issue with these approaches is accuracy and convenience. Regarding accuracy these models are not able to predict unusual occupancy profiles which will lead to wrong information. Misleading information will lead to unoptimized load control. Regarding convenience, these models are building-specific so for each building you have a different model to represent its occupancy profile which is not practical and adds complexity to the system.

Motivated by the recent advancements in the hardware accelerated device – GPUs & CPUs- deep learning specially the computer vision methodologies, Convolutional Neural Network is used to solve the occupancy detection problem. This approach will not be building depended as well as it will increase the accuracy dramatically.

This thesis aims to develop an advanced and accurate occupancy counting and detecting mechanisms based on computer vision and image processing techniques to be able to identify the occupancy status of a certain place in a building. The proposed approach will be integrated with the Closed-circuit television (CCTV) surveillance system to be able to analyze the camera feeds and to extract the occupancy information. This information will be used to control AC thermostat set point temperature to reduce the energy consumption. The analysis of the computer vision and image processing software will take place in a computational device. The power consumption reduction will be addressed as well as the

effect of changing the AC thermostat set point on the reduction in the power consumption.

The specific objectives of this thesis are as follows:

1. Developing an accurate occupancy model based on computer vision and image processing techniques.
2. Identifying a cost-effective hardware integrated with CCTV systems.
3. Building a laboratory prototype for accurate occupancy counting and detection systems.
4. Establishing a research laboratory for demand side management based on the advanced computer vision and image processing techniques.
5. Advance the kingdom with the modern technologies of energy saving using computer vision in image processing approaches.

1.3 Thesis Contributions

The proposed thesis work attempts to close some of the gaps found in the current literature. The work investigates the use of convolutional neural networks as the basic building block for human detectors, trackers, and counters. These operations will lead to accurate real time occupancy profiles. The proposed work is implemented and tested practically with simulated environments to calculate the energy savings resulting from accurate occupancy profile. This system can be implemented easily in any building type without additional cost for installation since it relies on the building infrastructure which will help in scalability.

Compared to the techniques in literature, the proposed approach:

- is considered an adaptive and robust model to predict the occupancy information.

- uses computer vision for accurate occupancy estimation.
- can give the highest level of details about the number of occupants in each room/office/place in the building.
- can work irrespective of the building type and usage.
- is easy to be integrated with the existing building infrastructure to reduce the implementation cost.

1.4 Thesis Organization

The thesis is organized as follows.

Chapter 2 provides an overall literature survey that covers different demand side management schemes specifically demand response schemes energy efficiency programs and IoT based DSM. It also provides a brief survey of hardware technologies used for detections. Finally, it provides a brief review of machine learning methodologies and statistical models used for occupancy profile prediction.

Chapter 3 provides details on the proposed methodology of the computer vision system. The different aspects of the computer vision system are explained, and further enhancements are presented.

Chapter 4 presents technology assessment of the hardware that can be used for the system implantation. Also presents the different libraries used to implement proposed methodology based on computer vision to detect occupancy status. Namely, Tensorflow, OpenCV, and sklearn are presented and discussed.

Chapter 5 presents thermal implementation of two environments along with multiple case studies that show the energy saving percentages when the proposed system is implemented. Results of implementation are also discussed in which different experiments are conducted to show the performance of the methodology and effect of varying different hyperparameters.

In chapter 6, a conclusion to the thesis and recommended future work are provided.

CHAPTER 2

LITERATURE REVIEW

In this chapter, a literature review of relevant work to the topic of demand side management, occupancy methodologies and machine learning is provided. Moreover, comparisons between DSM classifications namely, Demand Response and Energy Efficiency measures as well as IoT-based DSM schemes are also presented. Further emphasis is given to hardware used for motion detection that could be used for occupancy detection. Finally, more details on machine learning and deep learning approaches along with recent advances in the field is provided.

2.1 DSM Overview

The DSM approach generally deals with two sub-problems, the customer local power consumption and the aggregation of multiple users into usable coordinated demand-side energy resources [6]. DSM programs are classified in the literature into Energy Efficiency (EE) and Demand Response (DR) [7]. Developed DSM programs aim for peak clipping, load shifting, and valley filling. Different DSM programs differ in the type of optimization implementation methods, user interaction (cooperative or individual level), problem formulation, and time scale (real time or day ahead) [8]. In addition, some other factors make differences in DSM programs such as renewable energy resources involvement, energy storage system, occupancy, pricing scheme, multi-carrier energy system.

Demand Response (DR) is defined as the changes in electric usage by customers from their normal consumption patterns in response to changes in the price of electricity [9]. References [10]-[14] introduced DR management schemes where different algorithms and optimization techniques have been used. An energy management technique based on game theory approaches has been introduced and the multicarrier systems have been considered [10]. However, such an approach is complex and expensive to be deployed. A simpler DR scheme has been presented [11], [12], where users can subscribe to several engagement plans based on their preferences. Such approaches are simple but do not reach the optimum energy saving.

Other DR techniques utilized the game theory approaches in solving the optimization problem to find the best strategy for the optimal scheduling and consumption. [15] presented an energy management technique using the concept of smart energy hub or multicarrier systems. In their approach they have relied on the game theory approach to model the DSM problem. They formulated the problem as a non-cooperative game and finds the Nash equilibrium to reduce the peak to average ratio index and energy cost. However, the implementation of such techniques is challenging. [16] introduced an implementation of the DSM program that takes the operation cycle of the devices in consideration. They have formulated their problem as multi-objective optimization which achieves the least economic and environmental cost of the required daily energy of a group of low voltage customers. They also incorporated the user's lifestyle and the comfort levels into account. Implementing such an algorithm does not take in consideration the variety and the number of user's devices. [17] presented a DRM scheme that captures the uncertainties in the user demand and renewable power generation. The scheme includes

the utility, the service operators, and the end user where RTP used to reduce the effect of user's consumption uncertainty & TOU pricing are considered. Game-theoretic models of users' behavior have been used so each individual must reason about the consumption of others. [18] developed a day-ahead multi-objective optimization model on building level for building energy management systems under TOU price-based demand response which integrate storage devices, building integrated PV with other generations to optimize the economy and occupants' comfort -visual, thermal and air quality-. [19] presented a demand response mechanism for aggregators based on a combination of the RTP and the incentives to manage the charging and discharging of the EVs to maximize consumer's welfare in response to real time prices, accommodate their needs for transportation, and keep the distribution network within its limits operating limits. The previous mentioned techniques are complex in implementation and computationally expensive. [64] introduced a DSM scheme for smart microgrids (SMG), incorporating demand response and energy theft detection, implemented in a MATLAB-simulated campus SMG environment. This DSM utilizes Grey Wolf Optimization (GWO) for minimizing electricity costs, enhancing user comfort, and optimizing load patterns, encouraging consumer participation in dynamic energy consumption behaviors aligned with various pricing strategies. Comparative analysis showcases the GWO technique's superior effectiveness in reducing electricity bills and utility resource conservation, outperforming alternatives like the firefly algorithm (FFA) and bacteria foraging algorithm (BFA). [65] presented a blockchain technology and deep learning approach to efficient energy utilization when it comes to the surge in Electric Vehicle (EV) usage, contributing to grid economic reliability. Smart charging of EVs, facilitated by blockchain technology and deep learning, enhances demand side

management by enabling precise real-time control and accurate determination of charging systems. [66] showed a system utilizing the Internet of Things (IoT) for efficient power distribution in response to the challenges faced by utilities in developing countries, particularly the imbalance between supply and demand leading to load shedding and reduced power quality. This system, equipped with an ESP32 controller, nRF communication network, current sensors, real-time clock, and monitoring devices, aims to optimize load scheduling during peak hours. The objectives include achieving peak clipping in demand side management, ensuring efficient energy distribution between nodes, and extending the lifespan of distribution transformers. The system allows electricity boards to monitor node status through a webpage and communicate alerts to customers, ultimately contributing to reduced energy consumption and enhanced overall efficiency. [67] addressed the increasing significance of Local Energy Systems (LESs) or microgrids in future power systems, particularly in supporting the integration of renewable energy sources (RESs) for decarbonization. It presents a reinforcement learning-based demand-side management (DSM) strategy for multiple vector LESs, encompassing components like EV chargers, EV battery swap stations, electrical heating systems, and load point voltage controllers. This strategy aims for economical operation and efficient peak load shaving by coordinating multiple vector devices, considering constraints like thermal demand and real-time power consumption. Utilizing MATLAB/Simulink for modeling and performance assessment. [68] introduced a distributed sample-based event-triggered method for coordinating thermostatically controlled loads (TCLs) in response to photovoltaic and load variations from the demand side. This method employs a control scheme where TCLs communicate with neighboring TCLs only under certain event-

triggered conditions, efficiently reducing unnecessary communications and conserving limited network resources. It incorporates an initial power state updating scheme for TCLs, aligned with the output of PVs and loads, to maintain supply and demand balance. It took in consideration the resident comfort within specified temperature ranges and addresses the common issue of Zeno behavior in event-triggered control. The paper further solidifies its findings by applying Lyapunov stability theory to establish rigorous criteria for the proposed method's effectiveness. [69] addressed the shift in the power industry towards decentralized networks and the growing emphasis on reducing greenhouse gas emissions through the adoption of green energy resources. It introduces an effective load management approach for end-users, employing a Teaching-Learning optimization-based demand-side management technique combined with the integration of photovoltaic (PV) generation. It focuses on developing a secure and flexible trading platform between prosumers and utilities. [70] introduced a user interaction model based on a distributed demand response interaction using adjustable load demand response blockchain interoperability model complemented by a strategy designed to enhance reputation and trustworthiness, facing the challenges in traditional demand-supply interaction platforms, particularly the issues of trust and reliability in the connections between demand response aggregators and users. [71] presented a comprehensive set of protective measures to mitigate risks associated with the growing emphasis on demand side response (DSR) by the state as a key measure to balance electricity supply and demand. It identifies and analyzes the potential security risks associated with DSR, particularly concerning data and information leakage. it examines the various risks users may face during power grid regulation response. [72] introduced a priority-based load shifting demand response (PBLSDR) approach designed to bridge the

demand-supply gap, particularly with the rising incorporation of renewable energy sources. The research includes case studies that shed light on the practicality of load shifting and curtailments for residential customers with access to renewable energy. These studies offer valuable insights into the potential of demand response in residential markets, especially when coupled with automated demand response systems, highlighting their significance in reducing peak demand and dependence on high-cost, high-emission energy units. [73] presented a two-layer DR scheduling strategy tailored to user equipment levels, adaptable to various scenarios including residential users' varying response willingness and the uncertainty of electricity adjustment in different equipment. The strategy consists of two interconnected and mutually dependent components: the upper layer, which handles equipment-level DR scheduling, and the lower layer, which focuses on customer satisfaction feedback. [74] presents a methodology utilizes smart meter consumption data to analyze customer load patterns, identifying optimal periods and customer groups for incentivization. The key focus is on the selection of clustering algorithms, based on the compactness of clusters, and integrating both supervised and unsupervised Machine Learning (ML) techniques. It utilizes smart meter consumption data to analyze customer load patterns, identifying optimal periods and customer groups for incentivization.

Energy Efficiency (EE) measures can vary from the building material, design architecture, efficiency of the machine and appliances, load forecasting, occupancy estimates, and optimization techniques to schedule operation of devices in the buildings [20]-[31]. References [20]-[22] developed an energy management scheme for hybrid grids to reduce the AC-DC conversions and optimize the energy usage. An appliance scheduling solution taking in consideration the device operation and renewable generation and storage

devices has been introduced [23], [24]. Data science and machine learning have been integrated with demand side management and energy efficiency to achieve tasks such as prediction, classification, clustering, etc. Artificial neural networks, support vector machines and other algorithms are used to learn previous data and produce a certain output [25], [26].

Other EE measures have taken in consideration the user experience into account in the optimization and scheduling. [28] presented a Quality of Experience based energy management technique that schedules user's appliances to reduce the energy cost taking in consideration the level of the user's inconvenience when changing the device operation and renewable power generation. However, this approach is subjective to type of user and does not take care of multi-user usage. [29] introduced a scalable cooperative model predictive control scheme for urban districts consisting of multiple micro-grids with shared distributed energy resources DERs. [30] developed a multi-objective day-ahead DSM method that minimizes the energy cost, and the user discomfort levels. In their approach the user will submit a main plan and two alternative plans, and the algorithm will aggregate all the users plans and will choose the optimum cost-effective combination of all plans. Such approaches may lead to customer dissatisfaction because it involves the customer heavily in the process. [31] presented a smart energy efficient middleware system that can be communicated and integrated with heterogeneous technologies utilizing the advances in information communication technology ICT. HVAC, lighting, sensors, and actuators can be connected and integrated to the middleware to assure interoperability and future developments.

In addition, researchers have utilized the internet of things (IoT) as well in their solutions. IoT capabilities give a wide scalable, secure, private communication platform in requiring data, processing the information, and controlling mechanisms. This gives flexibility of designing a demand response, or energy efficiency schemes. Most of the previous mentioned optimization techniques or scheduling mechanisms can be utilized with much greater scalability and communication. An IoT-based solution that utilizes the new cloud-based technologies, Wi-Fi, and 3G/4G technologies has been developed [32]-[37]. However, such solutions are complex and costly.

2.2 Hardware Occupancy Methodologies

In all the previous presented solutions, building occupancy models suffer from inaccurate indication of the occupancy status of a building. Generally, several techniques have been presented in the literature with many technologies to address the occupancy issue. For instance, [38], [39] have utilized the motion sensors to monitor the occupancy status. Although motion sensors are widely used, they suffer from detecting occupancy in shared spaces [3]. Additionally, references [40], [41] presented a Radio Frequency Identification (RFID) based system to determine the occupant location in the indoor environments and to control the operation of the Heating, ventilation, and air conditioning (HVAC) systems. However, utilizing RFID technologies did not reach high accuracies in detecting moving objects [3]. Because of the drawbacks in RFID, researchers have investigated other technologies such as Wi-Fi. For instance, [42], [43] utilized a Wi-Fi based indoor positioning system and created occupancy profiles based on the measured Wi-Fi device number and locations in the university library building. Beside the lack of

accuracy in most of the previous techniques, implementing these hardware sets into complete systems is complex and needs a deep understanding of building management system protocols to be able to extract the occupancy information from these hardware detectors.

2.3 Deep Learning Methodologies

The field of Machine Learning and deep learning has gone through great improvements in recent years, mainly due to advances in deep learning. Such advances can be linked to several factors including the increasing computation power using Graphics Processing Units and distributed computing, methodological breakthroughs in deep learning, and development of software ecosystems [45]. In the field of machine learning [44], [45] relied on a statistical occupancy modelling, where [46], [47] have utilized a stochastic occupancy modelling, and [48], [49] exploited the machine learning occupancy modelling techniques in order to analyze the data collected during the occupancy monitoring periods. Machine learning and artificial intelligence techniques have been used to solve a variety of problems. Data classification, data understanding, data visualization, and many more approaches are used to solve the occupancy modeling problem.

On the other hand, deep learning is one subcategory of machine learning, and it is further divided into many subcategories such as computer vision which relies heavily on the convolutional neural network (CNN), natural language processing which relies on the recurrent neural network (RNN) and its variations like long, short term memory LSTM,

deep reinforcement learning, and ...etc. Computer vision is the field that processes images and videos to do certain operations such as face recognition, object detection, pose estimation, image segmentation, image classification, object tracking and many more analyses. These analyses could be used for different applications like security, real time monitoring, and ...etc.

In this thesis a computer vision system is built and utilized to process video frames from the environment aiming to do tasks like object detection, tracking and counting. The proposed approach will rely heavily on the people detection and localization algorithms moving in a sequence of images to help in counting the number of people that enters and exits a certain place.

Deep learning provides several advantages in the context of computer vision. CNN is well suited for all complex problems that are related to image processing video analytics since it can handle very high dimensional inputs. Also, they can be trained and fine-tuned incrementally and learn progressively as more data is provided.

The literature has presented a lot of object detection approaches with a trade-off between the speed of computation and accuracy of the detections and localization. Where the mobileNet [50] family is the fastest however they show lower accuracy compared to the other family. SSD & YOLO [51],[52] models come in the middle with a bit lower accuracy compared to the R-CNN [53] family which is the slowest family in terms of speed. In this thesis SSD algorithms will be utilized to do the people detection and localization task.

It can be added all advantages of the most popular networks in object detections and tracking each for two three lines.

In this thesis, SSD is considered to solve the real time occupancy problem. In addition, other image processing techniques are implemented in order to improve the overall performance. This shall be explained further in the next chapter.

2.4 Summery

This chapter provided an overall literature survey of the main DSM programs. Specifically, DR and EE schemes along with a brief review of IoT-based DSM techniques. Additionally, an overview of deep learning methodologies is also presented.

In summary, the following gapes are observed:

- Lack of occupancy accuracy in DSM methodologies, which is one of the most important factors that contributes to the increase in the energy consumption.
- Implementation complexity in occupancy mechanisms in different building types, which leads to scalability issues.
- Integration difficulty of the occupancy mechanism either needs civil work to install hardware or complicated models to deploy to the building.
- The tradeoff between complexity of the occupancy models and overall performance including the accuracy aspect.

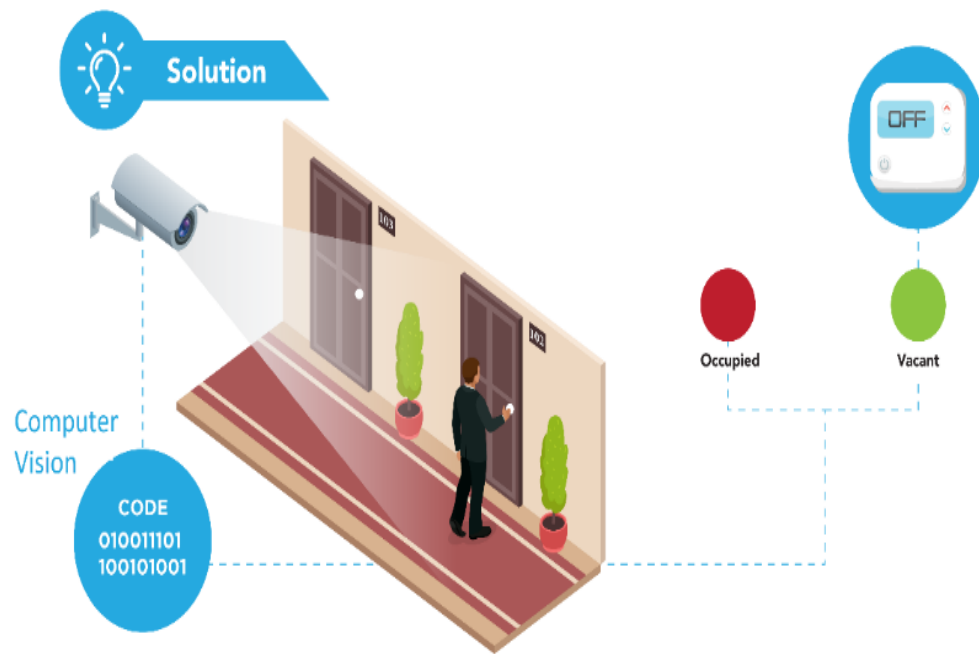


Figure 2.4.1 Description of the proposed system components

CHAPTER 3

PROPOSED METHODOLOGY

In this chapter, the proposed methodology to solve the occupancy status in buildings using the computer vision technique is presented. First, the human detection system is provided. Then, the tracking process is explained. Finally, the counting mechanism will take place where all people will be counted and as a result a real time occupancy status for each part of the building will be determined.

3.1 Deep Learning

It is a sub-field of machine learning. Deep learning has supervised, unsupervised and reinforcement learning in it similar to machine learning as well. Deep learning was initially inspired by the artificial neural networks commonly known as ANN.

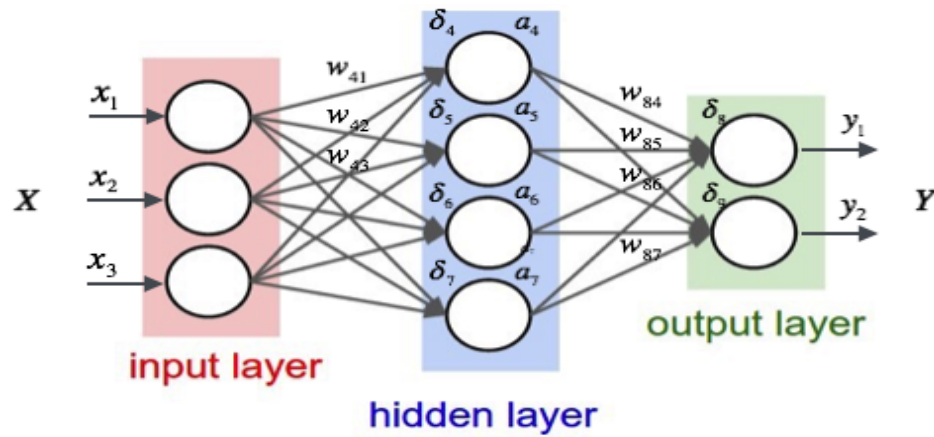


Figure 3.1.1 Shallow neural network

As shown in Fig. 3.1. a shallow neural network which consists of:

- Input layer: It can be image pixels aggregated in vectors or a time series data or any other type of data.
- Hidden layer: These layers are composed of learning weights that are trained in the training process.
- Output layer: This is the final layer where your desired output when a certain input is fed into the network.

Generally, the neural network is an approximation function between the input and the desired output as follows:

$$y = f(x; w) \tag{3.1}$$

where $w \in R^{n_0}$

- The network tries to learn the hidden layers weights that when multiplied by the inputs gives the best predicted output. Furthermore, the hidden transform the results by a nonlinear function that contributes to learn and identify very complex patterns in the input data and achieve better results and higher accuracies as shown in Fig 3.2.

Some examples of the popular activation functions:

- Sigmoid: A sigmoid function is a non-linear function that takes any real number as input and outputs a value between 0 and 1. This makes it well-suited for classification tasks, where the output of the neural network is a probability.

- Tanh: A tanh function is similar to a sigmoid function, but the output range is between -1 and 1. This makes it a good choice for tasks where the output of the neural network should be bounded, such as regression tasks.
- ReLU: A ReLU function is a non-linear function that takes any real number as input and outputs 0 if the input is negative, or the input itself if the input is positive. This makes it a very efficient function to compute, and it has been shown to be very effective in deep learning models.
- Leaky ReLU: A leaky ReLU function is a modification of the ReLU function that allows for a small non-zero output when the input is negative. This can help to prevent the vanishing gradient problem, which can occur in deep neural networks.
- ELU: An ELU function is another modification of the ReLU function that has been shown to be effective in deep learning models. ELU functions have a steeper slope than ReLU functions, which can help to improve the learning rate of the neural network.
- Softmax: A softmax function is a non-linear function that takes a vector of real numbers as input and outputs a vector of probabilities that sum to 1. This makes it well-suited for classification tasks where the output of the neural network is a probability distribution over multiple classes.

The choice of activation function depends on the specific task that the neural network is being used for. However, the functions listed above are a good starting point for most applications. In this paradigm deep learning is nothing but stacking many layers between the input and the output layers, hence the terminology deep learning.

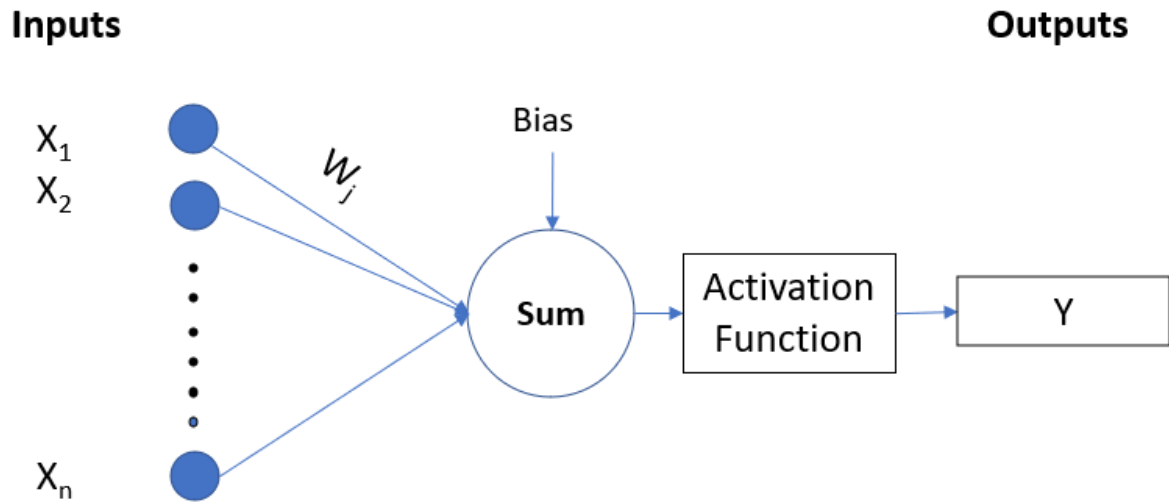


Figure 3.1.2 The operation of the neuron where the inputs are multiplied by the weights and summed together. After that, the output will be fed to the activation function to produce the output.

There are different algorithms designed to apply deep learning. Each algorithm has a specific use case like the type of data used and the task required to be performed out of learning this data. Considering these factors, the selected method can vary as follows:

- Convolutional Neural Network (CNN)
- Recurrent Neural Network (RNN)
- Long short-term memory (LSTM)
- Auto Encoders
 - Denoising autoencoder
 - Stacked autoencoder

- Variational autoencoder
- Sparse autoencoder
- Generative Adversarial network (GAN)

The application of deep learning is very wide such as:

- Classification: whether it is a task of predicting a cat or a dog and even multi-class classification,
- Feature extraction for visualizing the data in a lower dimensional space,
- Data preprocessing,
- Time series prediction: stock prediction, weather prediction, etc.,
- Regression tasks like localization,
- Object recognition which is both a classification & a regression task,
- Robotics.

In this thesis, the task required is object detection & tracking. Based on this task the selection of the deep learning techniques is determined. CNN is considered in this work since it is the best technique in deep learning that processes the videos and frames data types which will help in the counting process to identify occupancy status of the building. Industry wise this field is referred to as video analytics (VA) or computer vision (CV). In the next section more details on computer vision are presented.

3.2 Computer Vision

In the computer vision field CNN is used to process and analyze images and videos. It was firstly introduced by [54]. CNN was developed to overcome the challenges faced by NN. For example, when dealing with images with very size like 20×20 . If NN is used, then you need the input layer of the NN containing 400 weights to be trained. If the size of the input image increases, then the weights will increase significantly. On the other hand, CNN operates irrespective of the input image size with much less parameters to train. CNN structure varies from one to another depending on the complexity of the problem. Many scientists have developed different structures like Alex-Net [55], VGG-16 [56], and inception [57], etc. CNN structures have basic building blocks such as: convolutional layers, pooling layers, and fully connected networks. Each convolutional layer contains a different number of kernels to be trained for the task at hand. Kernels are the same concept of the NN weights. Fig. 3.4 shows how the kernel works. The number and size of kernels differs for each layer based on the CNN architecture and the task it performs. Kernels in most cases take size 3×3 but in some cases it can 5×5 depends on the application and what gives a better result. As shown in the same figure the kernel slides over the input for each sliding step the kernel is multiplied by the corresponding input added to form a point in the output. The sliding step is called stride which is the number of steps the kernel moves. This value is another hyperparameter that can be tuned for each convolutional layer to achieve a better result.

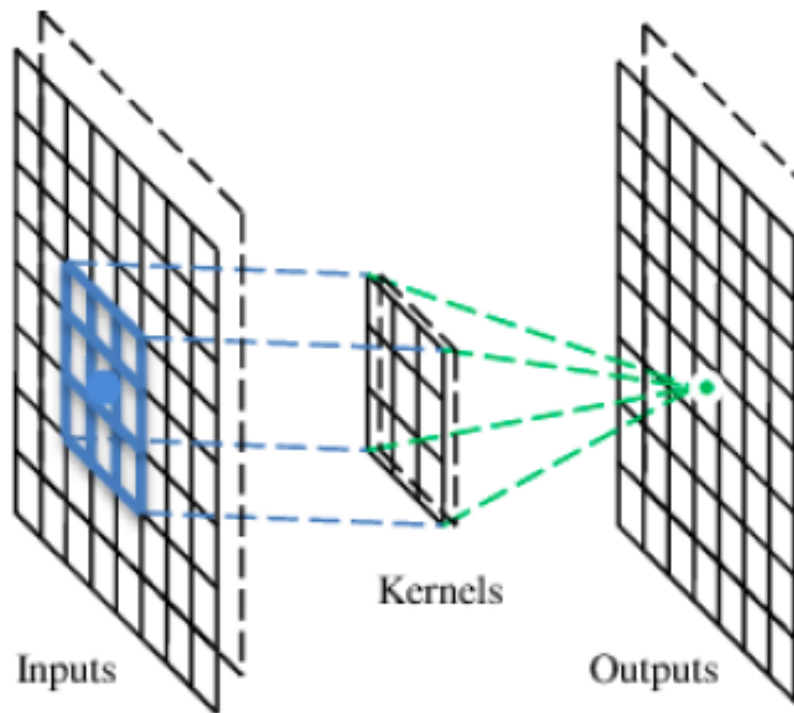


Figure 3.3 Each kernel in the convolutional layer slides on the input, multiplying the values of the kernel by the portion that is passed through the input to produce the output.

After the convolutional layers usually comes the pooling layers which used to reduce the size of the input by sliding a window of very small step size or stride as mentioned earlier. The size of these windows can take different values like 2,3, 4...etc. The window size and the stride in the pooling layer selection are other hyperparameters tuned to achieve better results. This window performs different operations depending on the structure of the network. The operations of the pooling layers are max-pooling or average pooling. In max-pooling the sliding window picks the maximum value and passes it to the output. Whereas in the average pooling the average is taken and passed to the next layer. Fig. 3.4 shows the max-pooling operation. The fully connected layer is a normal NN which was described earlier in this section.

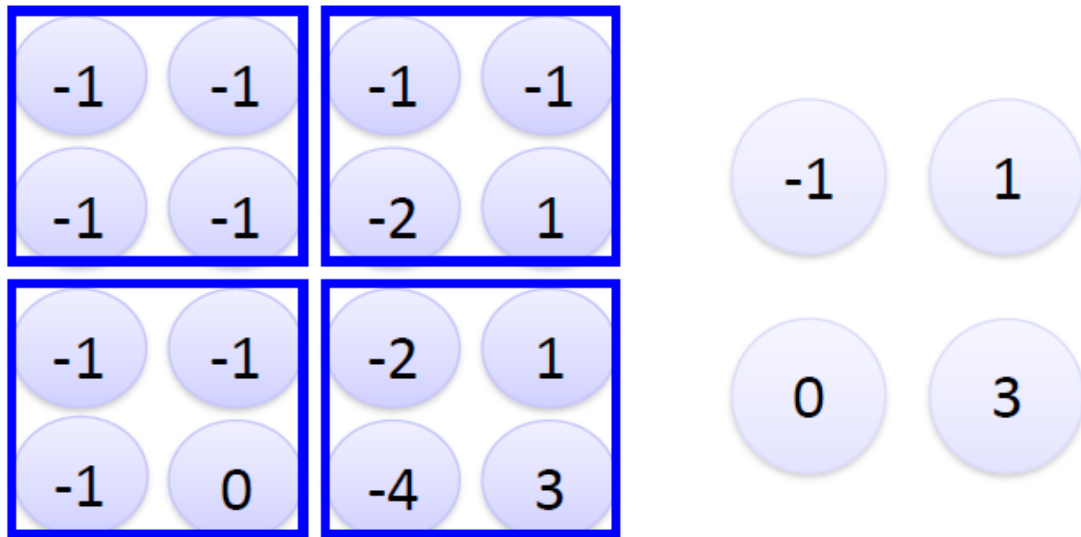


Figure 3.4 The max-pooling operation for an input image with size of 4*4.

For each CNN architecture that is dedicated for a specific task, the number of convolutional layers, pooling layers differs in heavy tasks like object detection, pose estimations and others. More layers are needed to be able to capture the features and achieve the desired output. Whereas, in simpler tasks like handwriting recognition less layer is needed. This thesis focuses on the object detection tasks since this is the core component in which humans are detected and all other processes depend on this step.

Finally, after a series of convolutional, and pooling layers comes the fully connected layers which are in simple terms NN layers which will be responsible for outputs. The outputs could be for example a bounding box in object detection task, a probability of object class in an image, as shown in Fig. 3.5.

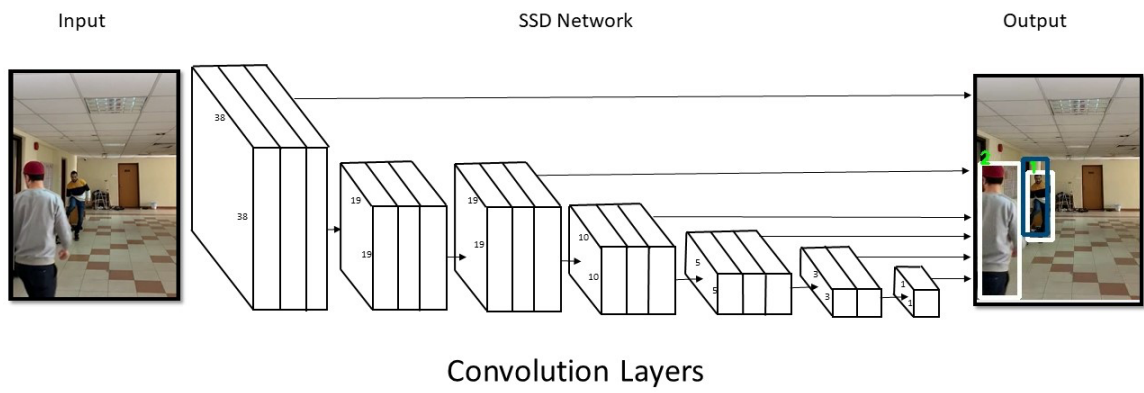


Figure 3.5 Convolution Neural Network (CNN) to classify objects like human [58]

3.3 Proposed System Architecture

The proposed computer vision process is divided into three main functions, which are detecting, tracking, and counting. The following subsections will describe each function in more detail.

3.3.1 Object Detection & Localization/Detector

Object detection is a critical branch of computer vision, involves the identification and localization of objects within an image or video. Unlike image classification, which categorizes an entire image, object detection pinpoints multiple objects and classifies them, often delineating each with a bounding box. This technology underpins a myriad of applications, ranging from self-driving cars, where it's crucial for recognizing pedestrians and obstacles, to security systems, industrial automation, and even healthcare, where it aids in identifying anomalies in medical imaging. The importance of object detection cannot be overstated. It's a cornerstone for systems requiring situational awareness and decision-making based on visual inputs. In retail, for example, object detection assists in inventory management and customer behavior analysis. In agriculture, it aids in crop analysis and pest detection, contributing significantly to precision farming. Its versatility and broad applicability make it a field of intense research and rapid development. The journey of object detection as a field has been intertwined with the evolution of computer vision and machine learning. Initially, object detection relied on simple pattern recognition and feature detection techniques. Early methods, like the Viola-Jones algorithm [75], introduced in 2001, were revolutionary for their time, enabling real-time face detection using features called Haar cascades. With the advent of machine learning, object detection experienced

significant advancements. Techniques such as Histogram of Oriented Gradients (HOG) combined with Support Vector Machines (SVM) became popular for tasks like pedestrian detection. However, these methods had limitations in terms of accuracy and versatility, struggling with variations in object appearance, scale, and orientation. The breakthrough came with the rise of deep learning, particularly Convolutional Neural Networks (CNNs). In 2014, the introduction of the R-CNN (Regions with CNN features) [76] marked a paradigm shift. It used selective search to propose regions and a CNN to classify them, significantly improving detection accuracy. This was followed by faster and more efficient models like Fast R-CNN and Faster R-CNN [53], which integrated region proposal and classification into a single, unified network. The field continued to evolve with the development of single-shot detectors like YOLO (You Only Look Once) [52] and SSD (Single Shot MultiBox Detector) [51]. These models could detect objects in real-time, offering a balance between speed and accuracy, making them suitable for applications requiring instant decision-making, like autonomous vehicles. Today, object detection stands at the forefront of AI research, with continual improvements in speed, accuracy, and adaptability. It's an area ripe with challenges and opportunities, continually pushing the boundaries of what machines can perceive and understand about the visual world.

Next some of the computer vision terminologies and concepts will be explained as it is importance to the proposed approach.

A. Object Detection Technical Terminologies

- Bounding Boxes

A bounding box is a rectangle drawn around an object of interest in an image, representing the object's location and extent. In object detection, bounding boxes are defined by

coordinates (usually the top-left corner, width, and height) and are critical for both training and evaluating models. They enable the algorithm to learn not only what objects look like but also where they are located in the visual space.

- **Precision and Recall**

Precision and recall are fundamental metrics used to evaluate the performance of an object detection model.

- **Precision** measures the accuracy of the predictions, indicating the proportion of positive identifications that were actually correct. It's calculated as the number of true positives (correct detections) divided by the total number of positive predictions (true positives plus false positives).

$$precision = True\ Positives / (True\ Positives + False\ Positives)$$

- **Recall**, also known as sensitivity, measures the model's ability to detect all relevant instances. It's the ratio of true positives to the sum of true positives and false negatives (instances that were not detected but should have been).

$$Recall = True\ Positives / (True\ Positives + False\ Negatives)$$

- **Intersection over Union (IoU)**

IoU is a crucial metric in object detection, used to quantify the accuracy of a bounding box prediction. It is the ratio of the area of overlap between the predicted bounding box and the ground truth (actual) bounding box to the area of their union.

$$IOU = Area\ of\ the\ overlap / (Area\ of\ the\ union)$$

- **Non-Maximum Suppression (NMS)**

NMS is a technique used to resolve the issue of multiple bounding boxes being predicted for the same object. It involves selecting the bounding box with the highest confidence score and eliminating all other boxes that have a high IoU with this box, assuming they are multiple detections of the same object.

- **Confidence Score**

A confidence score, in the context of object detection, is the probability that a predicted bounding box contains an object of interest. It reflects the model's certainty in its prediction. During the post-processing stage, bounding boxes with confidence scores below a certain threshold are usually discarded to reduce false positives.

- **Anchor Boxes**

Anchor boxes are predefined bounding boxes of various shapes and sizes that are used in object detection algorithms like SSD and YOLO. They serve as starting points for the model to adjust and predict the actual bounding boxes for objects. The use of anchor boxes helps the model detect objects of different shapes and sizes more effectively.

These concepts form the foundation of object detection, providing the tools and metrics necessary to develop, evaluate, and refine models. Understanding these basics is essential for delving deeper into the more complex aspects of object detection algorithms and their applications.

B. Overview of Object Detection Techniques

Object detection has evolved significantly, from traditional techniques reliant on handcrafted features to sophisticated deep learning models. This evolution reflects broader trends in computer vision and artificial intelligence.

- Traditional Techniques

- **Haar Cascades:** Introduced by Viola and Jones in 2001, this method is renowned for its efficiency in real-time face detection. It uses Haar-like features, which are simple contrast features that can be computed rapidly using an integral image. A cascade function is then applied, which enables the algorithm to discard non-faces quickly and focus computational resources on promising face-like regions. While fast and effective for some applications, Haar cascades are limited in their ability to handle variations in lighting, pose, and facial expressions.
- **Histogram of Oriented Gradients (HOG) + Support Vector Machines (SVM):** This technique, popularized for pedestrian detection, involves extracting HOG features from images—essentially, gradients or edge directions that are representative of shapes. These features are then fed into an SVM classifier to distinguish between object classes (e.g., pedestrian vs. non-pedestrian). HOG+SVM was a breakthrough for its time, offering robustness against variations in clothing and background. However, it struggled with occlusion and different viewing angles.

- **Transition to Deep Learning-Based Methods**

The limitations of traditional methods in handling variability in real-world data led to the adoption of deep learning, particularly Convolutional Neural Networks (CNNs), which automatically learn hierarchical feature representations from data.

- **R-CNN (Regions with CNN features):** Introduced in 2014, R-CNN was a major step forward. It used a region proposal algorithm to generate potential bounding boxes in an image and then ran a CNN on each of these regions to classify the object within. R-CNN significantly outperformed traditional methods but was computationally expensive due to the need to run a CNN separately for each region proposal.
- **Fast R-CNN:** Building on R-CNN, Fast R-CNN introduced the concept of RoI (Region of Interest) pooling, allowing the network to share computations. Instead of running a CNN on each region proposal, Fast R-CNN runs the CNN just once per image and then extracts features for each region from the resulting feature map. This approach was faster and more efficient than R-CNN but still relied on external region proposal methods.
- **Faster R-CNN:** To overcome the dependency on external region proposal algorithms, Faster R-CNN integrated the region proposal network (RPN) into the architecture, enabling the simultaneous prediction of object bounds and objectness scores at each position. Faster R-CNN achieved real-time performance and high accuracy, cementing deep learning's status as the state-of-the-art in object detection.

C. Detailed Explanation of SSD Architecture

The Single Shot MultiBox Detector (SSD) is a breakthrough in object detection for its balance of speed and accuracy. Developed by Wei Liu et al. in 2016, SSD diverges from previous methods like R-CNN by eliminating the need for a separate region proposal network.

- **Architecture Overview:** SSD's architecture is based on a feed-forward convolutional network that produces a fixed-size collection of bounding boxes and scores for the presence of object class instances in those boxes. It applies convolutional filters of different sizes to the feature maps at multiple scales, enabling the detection of objects at various resolutions.
- **Multi-Scale Feature Maps:** One of the key innovations of SSD is its use of multi-scale feature maps for detection. Unlike previous methods, which used a single feature map, SSD utilizes several feature maps from later stages of a base network (like VGG16), each responsible for detecting objects at different scales. This approach improves the detection of smaller objects, a challenge in many object detection tasks.
- **Anchor Boxes:** SSD utilizes anchor boxes (also called default boxes) of various aspect ratios at each location in these feature maps. During prediction, the model adjusts the sizes and aspect ratios of these boxes to best fit the objects in the image.
- **Convolutional Predictors:** For each feature map cell, a set of convolutional filters predicts the offset for the bounding boxes and the class probabilities. This process is repeated for every cell in each feature map, resulting in a large set of boxes, which are filtered through a non-maximum suppression step to produce the final detections.

D. Comparison between SSD & YOLO

- **SSD vs. YOLO:** YOLO (You Only Look Once) is another popular single-shot detector. Both SSD and YOLO are fast, but they differ in their approaches. YOLO divides the image into a grid and predicts bounding boxes and probabilities for each grid cell. SSD, on the other hand, skips the grid-based approach and directly predicts the bounding box offsets over different feature maps.
- **SSD Pros and Cons:** Better at detecting small objects due to multi-scale feature maps. Generally, more accurate than YOLO. On the other hand, SSD is slower than YOLO. Its performance heavily depends on the size of the objects.
- **YOLO Pros and Cons:** Extremely fast, suitable for real-time applications. Simpler architecture. On the other hand, it struggles with small objects and objects that appear in groups.
- **Mathematical Underpinnings**
 - **Loss Functions:** SSD's loss function is a combination of a localization loss (for bounding box prediction) and a confidence loss (for object class prediction). The localization loss is typically a Smooth L1 loss between the predicted box (offsets) and the ground truth box parameters. The confidence loss is a softmax loss over multiple classes.
 - **Bounding Box Calculations:** The model outputs offsets for each anchor box relative to the default box shapes in the cell. These offsets are then translated into actual bounding box coordinates for object detection.

- **Non-Maximum Suppression:** To handle multiple detections for the same object, SSD applies non-maximum suppression, which selects the box with the highest confidence and removes boxes that have a high IoU with it.

E. Proposed Methodology Architecture

As we discussed previously the detector consists of a Convolutional Neural Network (CNN) model trained to detect and localize a specific group of objects (people, chairs, birds, etc). CNNs are the best among deep learning models for analyzing images and pictures. These models are robust and work for most object detection applications [51]. The trained network takes a sequence of frames from the security cameras as input and then outputs three types of information. The first one is bounding boxes around the detected object. The second one is the confidence scores, which represent the probability of each detected object existence. The last type of information is the class of each detected object. Each bounding box is catheterized by x, y, w, h , the top left x-axis coordinate, the top left y-axis coordinate, the width of the bounding box, and the bounding box height, respectively. The CNN model, which is adopted in the proposed study, is the Single Shot Detector (SSD). The SSD model provides enormous speed gains compared to other CNN object detection models [51]. Fig. 3.6. shows the SSD architecture, which consists of a series of convolutional layers, leading to extract several sets of feature maps at different scales. Each CNN layer is responsible for estimating existing objects at a specific scale. After that, all objects are combined, and non-max suppression is applied to eliminate repeated objects and keep only the one with the highest score.

The SSD loss function used for training is weighted sum of the localization loss (loc) and the confidence loss (conf) [51]:

$$Loss(x, c, l, g) = \frac{1}{N} * (L_{conf}(x, c) + \alpha L_{loc}(x, l, g)) \quad 3.2$$

where N is the number of matched detections. If N = 0 the loss function will be 0, and α will be set to 1. Let $x_{ij}^p = \{0,1\}$ be an indicator for matching i-th default box to the j-th ground truth box of category p, then the localization loss between the predicted box (l) and the ground truth (g) is the smooth L1 loss function and given by:

$$\begin{aligned} L_{loc}(x, l, g) &= \sum_{i \in Pos}^N \sum_{m \in \{cx, cy, w, h\}} x_{ij}^p smoth_{L1}(l_i^m - g_j'^m) \\ g_j'^{cx} &= (g_j^{cx} - d_i^{cx})/d_i^w \\ g_j'^{cy} &= (g_j^{cy} - d_i^{cy})/d_i^h \\ g_j'^w &= \log(g_i^w/d_i^w) \\ g_j'^h &= \log(g_i^h/d_i^h) \end{aligned} \quad 3.3$$

where (d) is the default bounding box with width (w) and height (h).

The confidence loss is the SoftMax loss over multiple classes confidences (c) and given by:

$$\begin{aligned} L_{conf}(x, c) &= -\sum_{i \in Pos}^N x_{ij}^p \log(c_i'^p) - \sum_{i \in Neg} \log(c_i'^0) \\ c_i'^p &= (\exp(c_i^p)) / \sum_p \exp(c_i^p) \end{aligned} \quad 3.4$$

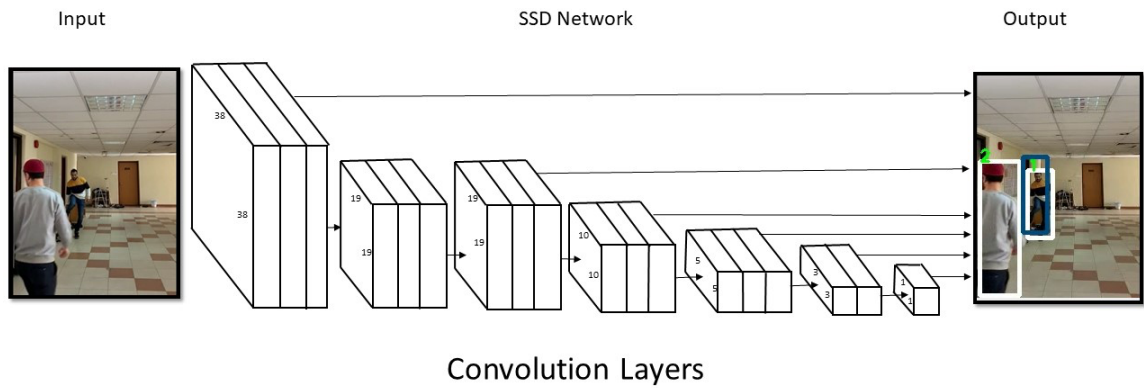


Figure 3.6 SSD model architecture composed of several convolutional layers to detect objects [51].

3.3.2 Object Tracking/Tracker

Object tracking in the realm of computer vision refers to the process of locating a moving object (or multiple objects) over time in a video sequence. While object detection identifies objects in individual frames, object tracking connects these detections across different frames to maintain the identity of each object. This distinction is crucial in understanding the dynamic behavior of objects, making object tracking a vital component in video analysis.

The significance of object tracking spans various domains, from security and surveillance, where it's used to track movements of people or vehicles, to sports analytics, autonomous vehicles, and even wildlife monitoring. In healthcare, it facilitates patient monitoring and

analysis of physical activities. For robotics and autonomous systems, object tracking is indispensable for navigation and interaction with dynamic environments.

A. Challenges in Object Tracking Compared to Object Detection

- **Temporal Consistency:** Unlike object detection, which operates on individual images, object tracking must maintain the identity of objects across frames. This involves dealing with temporal consistency and the challenges it brings, such as handling occlusions (when objects are temporarily blocked) or interactions between multiple objects.
- **Real-Time Processing:** Tracking often requires real-time processing to be effective, especially in applications like autonomous driving or real-time surveillance. This necessitates not only accurate but also fast algorithms that can keep up with the pace of the video stream.
- **Varying Conditions:** Object tracking must contend with changes in object appearance, scale, and orientation over time. Factors such as lighting changes, camera motion, and background clutter add complexity. The tracker must be robust to these variations to maintain accurate tracking.
- **Initial Detection Quality:** The quality of object tracking is heavily reliant on the accuracy of the initial object detection. Poor detection can lead to incorrect or lost tracks. This dependency highlights the importance of sophisticated and reliable detection algorithms.
- **Long-Term Tracking:** While object detection is a one-time process for each frame, object tracking must sustain over long periods. This long-term aspect raises issues like

handling re-entrance of objects into the scene after they have left or been occluded for a significant time.

- **Resource Constraints:** Object tracking algorithms need to balance accuracy with computational efficiency, especially in resource-constrained environments like mobile devices or embedded systems.

B. Basic Concepts in Object Tracking

Object tracking, a fundamental task in computer vision, involves continuously locating an object over time across video frames. Understanding the classifications of trackers and the metrics for evaluating them is crucial for developing and choosing the right tracking algorithms for specific applications.

- **Tracker Classification**
 - **Model-Based Trackers:** These trackers utilize predefined models of the objects they are tracking. The models often include shape, color, or motion characteristics of the object. Model-based trackers are particularly effective when the object's appearance is consistent and distinguishable. However, they can struggle in dynamic environments where the object's appearance may change drastically.
 - **Feature-Based Trackers:** Feature-based trackers focus on identifying and tracking distinctive features (like edges, corners, or textures) of objects. These features are tracked across frames to determine the object's motion. This approach is more robust to changes in appearance and scale but can be challenged by occlusion or a lack of distinguishable features.
 - **Deep Learning-Based Trackers:** Leveraging the power of neural networks, especially Convolutional Neural Networks (CNNs), these trackers automatically

learn to identify and track objects from large datasets. Deep learning-based trackers have shown remarkable performance improvements over traditional methods, particularly in handling complex tracking scenarios with occlusions, varying lighting conditions, and rapid movements. However, they require substantial computational resources and large annotated datasets for training.

- **Metrics for Evaluation**

- **Accuracy:** This metric measures how precisely the tracker locates the object. It's often quantified using metrics like Intersection over Union (IoU) or the center location error between the predicted and actual positions of the object.
- **Robustness:** Robustness refers to the tracker's ability to maintain consistent tracking under challenging conditions such as occlusion, illumination changes, scale variations, and fast movements. A robust tracker should be able to recover from temporary tracking losses.
- **Speed:** Tracking speed is critical, especially for real-time applications. It is measured in frames per second (fps). High-speed tracking is essential in scenarios like sports analytics or autonomous vehicle navigation, where quick response times are crucial.
- **Long-Term Tracking Ability:** This evaluates the tracker's ability to follow an object over extended periods, including handling temporary occlusions or when the object leaves and re-enters the frame.
- **Resource Efficiency:** For applications with limited computational resources, like mobile or embedded systems, the efficiency of the tracker in terms of memory and CPU usage is also an important consideration.

C. Traditional Object Tracking Methods

Traditional object tracking methods, developed prior to the widespread adoption of deep learning, rely on various algorithms and statistical models to follow objects across video frames. While these methods laid the groundwork for modern tracking, they also exhibited limitations that prompted the shift towards more advanced techniques.

- **Mean-Shift Algorithm [77]**

- **Overview:** The Mean-Shift algorithm is a non-parametric, iterative method used for locating the maximum density of a discrete sample (like pixels in a frame). In object tracking, it's employed to find the most likely target location by iteratively shifting a window towards the densest part of the probability distribution, typically defined by the color histogram of the target.
- **Process:** Initially, a kernel (window) is placed over the target. The algorithm then calculates the mean location of the target within the window, shifts the window towards this mean, and repeats the process until convergence.
- **Pros and Cons:** Simplicity and efficiency in computation. Performs well in simple scenarios where the target's appearance doesn't change drastically. On the other hand it is sensitive to the choice of window size. Struggles with scale changes, occlusions, and rotation. Limited in handling complex motion patterns.

- **Kalman Filters [78]**

- **Overview:** The Kalman Filter is a predictive filter that estimates the state of a linear dynamic system from a series of noisy measurements. In object tracking, it's used to predict the object's future location based on its previous states.

- **Process:** It operates in two steps: predict and update. The predict phase uses the state model to forecast the object's position. The update phase then adjusts this prediction based on the new measurement data.
- **Pros and Cons:** Effective in filtering out noise and making predictions under uncertainty. Well-suited for linear motion tracking. On the other hand, it Assumes a linear model, which is often inadequate for complex, real-world scenarios. Struggles with abrupt motion changes and non-linear dynamics.

- **Limitations Leading to the Adoption of Deep Learning Approaches**

- **Handling Complex Scenarios:** Traditional methods often falter in complex tracking environments characterized by rapid motion changes, occlusions, varying lighting conditions, and scale changes.
- **Dependence on Handcrafted Features:** Methods like Mean-Shift rely heavily on features like color histograms, which can be insufficient for distinguishing between objects with similar appearances or in dynamic backgrounds.
- **Limited Adaptability:** Traditional trackers lack the ability to adapt to new or unforeseen tracking scenarios. They are typically designed for specific situations and lack the generalizability that deep learning models offer.
- **Inadequacy in Real-Time Tracking:** While some traditional methods are computationally efficient, they often compromise accuracy, making them less viable for real-time applications that require both speed and precision.
- **Challenges with Non-Linearity:** Algorithms like Kalman Filters are constrained by their assumption of linearity, making them less effective for tracking objects with complex, non-linear movements.

D. Deep Learning in Object Tracking

The integration of deep learning into object tracking has revolutionized the field, offering enhanced accuracy, adaptability, and robustness in complex tracking scenarios. This shift to deep learning-based methods has largely been driven by the advancements in Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs).

- Overview of Deep Learning Methods in Tracking

- **CNN-Based Trackers:** CNNs, renowned for their capability in feature extraction and object recognition, have been widely adopted in object tracking. Trackers like GOTURN (Generic Object Tracking Using Regression Networks) use CNNs to learn generic object features from large video datasets, enabling them to track new objects without retraining. These networks can capture complex appearance changes, making them effective in tracking objects even under significant variations in scale, orientation, and illumination.
- **RNN-Based Trackers:** RNNs, particularly Long Short-Term Memory (LSTM) networks, are employed for their ability to model temporal dependencies in sequential data. In tracking, they are used to predict the motion and appearance of an object over time, providing an edge in scenarios where temporal coherence is critical.
- **Siamese Network-Based Trackers:** These trackers, like SiamFC (Siamese Fully Convolutional), utilize a Siamese neural network architecture to locate an object in a new frame by comparing it to a reference patch. They are known for their balance between accuracy and speed and are particularly effective in real-time tracking.

- **Combining Different Networks:** Some advanced tracking systems combine different types of networks, like CNNs for spatial feature extraction and RNNs for temporal modeling, to capitalize on the strengths of each.

E. Introduction to Deep SORT as an Advanced Technique

Deep SORT [59] (Simple Online and Realtime Tracking with a Deep Association Metric) is an extension of the SORT (Simple Online and Realtime Tracking) algorithm, integrating deep learning features to enhance tracking accuracy.

- **Deep Association Metric:** Deep SORT employs a deep neural network to generate high-dimensional feature vectors for each detected object, providing a robust association metric. This metric significantly improves the tracker's ability to associate detections over time, particularly in scenarios where objects are closely interacting or occluded.
- **Kalman Filter and Hungarian Algorithm:** Deep SORT combines the deep association metric with a Kalman filter for motion prediction and the Hungarian algorithm for frame-by-frame association. This integration allows for effective handling of temporary occlusion and smooth tracking even with erratic object motion.
- **Handling Occlusions and Re-identification:** One of the key strengths of Deep SORT is its ability to re-identify objects that have left and re-entered the scene or have been occluded for short periods. This is achieved by comparing the deep feature vectors of newly detected objects with those of previously tracked but currently unobserved objects.

- **Real-Time Performance:** Despite its complexity, Deep SORT is designed for real-time applications. It strikes a balance between tracking accuracy and computational efficiency, making it suitable for scenarios requiring both speed and reliability.

F. Deep Dive into Deep SORT

Deep SORT (Simple Online and Realtime Tracking with a Deep Association Metric) extends the SORT (Simple Online and Realtime Tracking) algorithm, incorporating deep learning to enhance tracking accuracy, especially in complex scenarios. This advanced method has become pivotal in applications where robust and efficient object tracking is crucial.

- **Detailed Explanation of the Deep SORT Algorithm**
 - **Integration of Deep Learning Features:** Deep SORT integrates deep learning for feature extraction, which is a significant enhancement over the original SORT algorithm. This integration involves using a Convolutional Neural Network (CNN) to generate high-dimensional feature vectors for each detected object, providing a robust appearance descriptor that aids in distinguishing between different objects, especially in crowded scenes.
 - **Data Association with Deep Features:** The algorithm employs these deep features for data association. This approach allows for more accurate matching of objects across frames, crucial for maintaining identity, particularly in cases of occlusion or interactions between objects.
 - **Motion Prediction with Kalman Filter:** Deep SORT uses a Kalman filter for predicting the motion of each tracked object. The Kalman filter, a linear quadratic

estimator, is effective in predicting the future state of an object based on its current state while accounting for the inherent uncertainty in motion.

- **Hungarian Algorithm for Frame-by-Frame Association:** The algorithm uses the Hungarian algorithm for solving the assignment problem that arises when matching detected objects to existing tracks. This approach ensures optimal matching based on the calculated distances between detected objects and predicted Kalman filter states.

- **Mathematical Explanation of its Components**

- **Kalman Filter:** The Kalman filter predicts the future state of an object based on a linear motion model. It operates in two phases: predict and update. The predict phase forecasts the next state of the object, while the update phase corrects the prediction based on new observation data. Mathematically, it involves updating the state estimate and covariance matrix using the process model and measurement model.
- **Hungarian Algorithm:** This algorithm is a combinatorial optimization algorithm that solves the assignment problem in polynomial time. It is used to find the optimal assignment between detected objects and existing tracks, minimizing the total cost, which in this case is the distance between predicted positions and observed positions.
- **Feature Extraction Using CNN:** Deep SORT utilizes a CNN to extract features from detected objects. These features are high-dimensional vectors representing the appearance of the object. The CNN is trained on a large dataset to learn

discriminative features, which are then used to compute similarity scores between objects across frames.

- **Advantages over SORT**

- **Improved Handling of Occlusions:** Deep SORT's use of deep features allows it to better handle occlusions. When an object is temporarily occluded and then reappears, its feature vector assists in correctly re-identifying and continuing to track it.
- **Enhanced Accuracy in Crowded Scenes:** The deep learning component significantly improves tracking accuracy in crowded scenes, where multiple objects are in close proximity, and there is a high risk of identity switches.
- **Robustness to Appearance Changes:** The CNN-based feature extraction provides robustness against changes in object appearance, such as varying lighting conditions or angles, making Deep SORT more versatile in different environments.
- **Balanced Speed and Accuracy:** While incorporating more sophisticated features, Deep SORT still maintains a balance between tracking accuracy and speed, making it suitable for real-time applications.

G. Summary of The Tracker Module

Since the proposed system is counting objects in a sequence of frames, trackers can avoid counting the same objects many times. As we discussed the tracker is responsible for taking an initial set of detected objects. Then, the tracker creates unique random identity numbers for each of the initially detected objects. After that, it tracks each object as it moves around the frames in a video, maintaining the unique IDs' assignment. Other benefits for trackers

are that even when the detection fails, it will identify and estimate the object location based on its history, and the system will be robust to object occlusion because it saves the features of all detected objects. When object occlusion happens, and some objects disappear from the scene when they come back into the scene the tracker will assign the same ID that was assigned to them before collision based on the feature of these objects. Most of the tracking algorithms rely on extracting certain features of the detected objects and comparing them across the frames. Any feature comparison algorithm can be used to compare features, such as Euclidean distance or cosine distance measures. Fig. 3.7. shows how the tracker gives a unique ID to each person. Fig. 3.8. shows how robust the system is even after one of the detected objects disappeared from the scene; the tracker could recover its identity. In the proposed approach, the Deep Sort Tracking algorithm is used. This tracker extracts a vector of 128 features from each detected object. More details on Deep Sort Tracking can be found in [59].

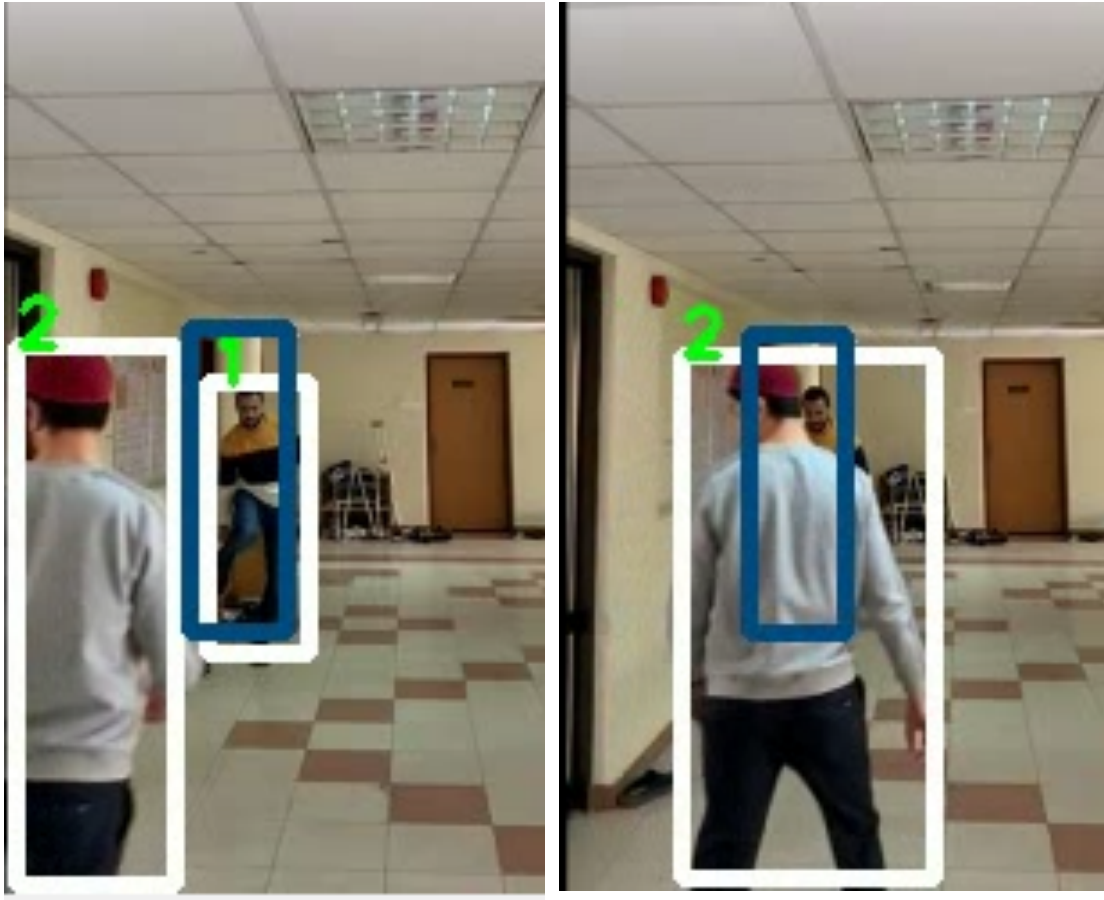


Figure 3.7 Object detection, identification, and tracking.

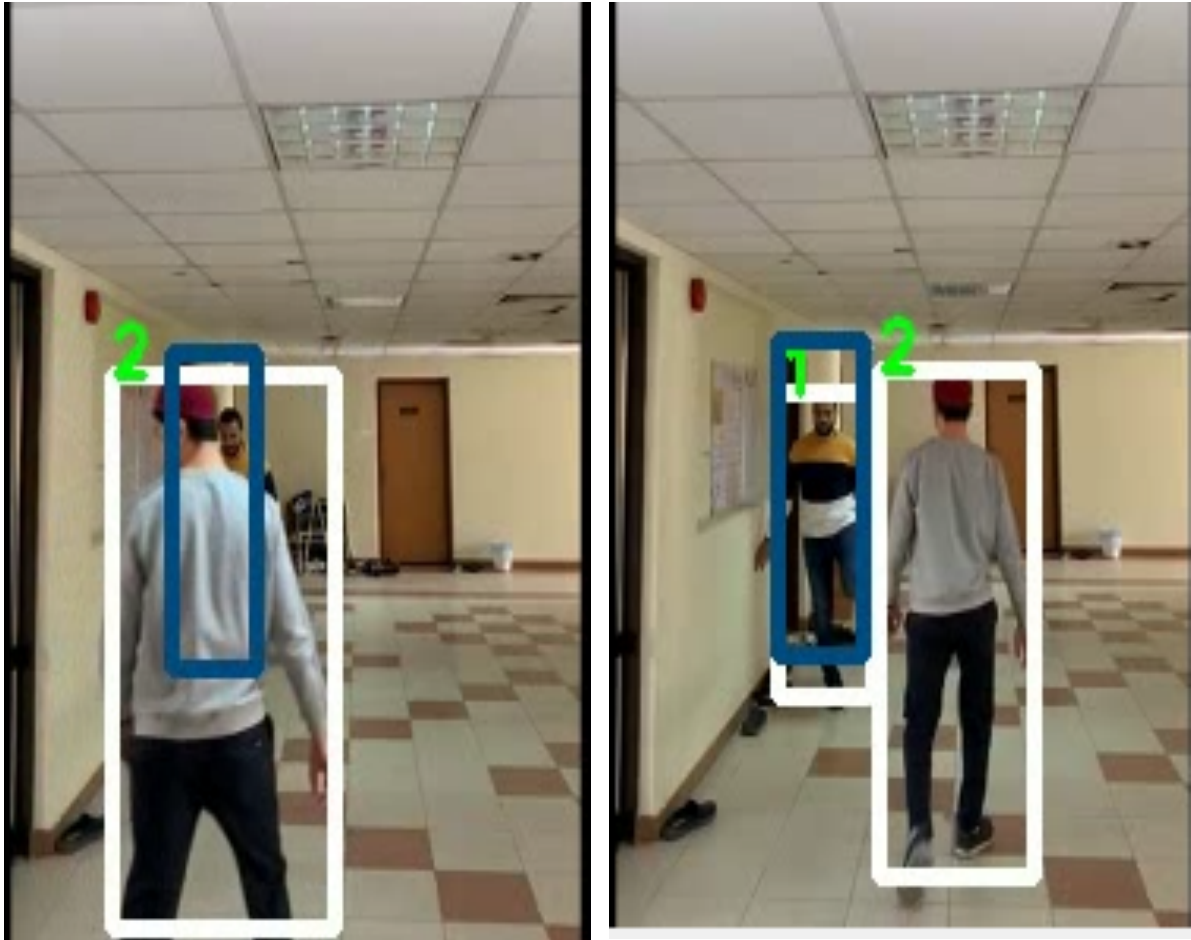


Figure 3.8 Tracking system occlusions robustness.

3.3.3 Object Counting/Counter

The counter counts the tracked objects based on certain conditions. When these conditions are satisfied, it decides whether to count the people going in or out. The counting algorithms start by receiving the tracked labeled objects from the tracker. Each object has its identity label. The counting is done by selecting the area of interest, as shown in Fig 3.9. For each object to be counted inside or outside a specific place, some rules must be satisfied. Firstly, the objects should intersect with the counting region with a minimum

intersection area. Secondly, the object has to have a negative or positive y-direction displacement depending on the object's motion.



Figure 3.9 Counting region in the workplace

A. Detailed Algorithm for The Counter

Initial Setup:

- Tracks of moving objects (presumably people) are being monitored.
- here are areas defined such as `door_box` (the area around a door) and a method to check if an object is on the left side of a door (`lef_side_box_check_out`).

- Functions IOU and area are used to calculate the Intersection over Union and the area of overlap between boxes, respectively.

For Each Track in the Tracks:

- **Remove Out-of-Bounds Tracks:**

If a track is marked as out of bounds but has been updated recently (`time_since_update == 0`), it is removed from the out-of-bounds list.

- **Counting People Exiting the Room:**

A person is considered exiting if:

They are confirmed by the tracker (`track.is_confirmed()`).

Their track overlaps significantly with the `door_box` ($\text{IOU} > 0.55$).

They pass the left side box check and have a significant area overlap.

They are not already counted as in the corridor (`not in self.people_in_cor`).

For such a person:

If their track age is between 5 and 10, they are added to `people_in_cor` and the door counter is decremented.

If they are already marked as outside the corridor or out of bounds but recently updated, they are similarly processed.

- **Counting People Entering the Room:**

A person is considered entering if:

The track hasn't been updated for 13 to 15 time units.

There is a significant overlap with the `door_box` ($\text{IOU} > 0.2$).

The person has a significant track age (> 40) and hits (> 40).

They are not already marked as outside the corridor.

For such a person:

They are added to `people_out_cor` and the door counter is incremented.

If they were previously counted as in the corridor, they are removed from `people_in_cor`.

- Marking Tracks as Out of Bounds:

If a track's bounding box is too close to the frame boundaries (< 5 or $> \text{frame width} - 7$) and hasn't been updated for more than 13 time units, it's added to the out-of-bounds list.

B. Key Points of the Algorithm:

- The algorithm uses spatial relations (like overlap with door area) and temporal information (like age and last update time of a track) to count people.
- It keeps track of people entering and exiting separately and adjusts a door counter accordingly.
- It considers tracks that are no longer being updated or are near the frame boundaries as special cases, likely as an exit or entry or to handle tracking errors.

C. Limitations and Considerations for Future Enhancements:

- The algorithm's accuracy heavily depends on the precision of the tracking system and the defined areas like `door_box`.
- The specific threshold values (like $\text{IOU} > 0.55$, $\text{age} > 40$) are crucial and may need tuning based on real-world testing.
- Handling of occlusions or interactions between multiple people is not explicitly mentioned but is vital for accurate counting.

3.4 Summery

In this chapter the proposed approach is clearly explained to solve the occupancy status problem in building. The proposed approach relies heavily on the deep learning and computer vision algorithms which is explained in this chapter. Finally, a complete computer vision flow is provided which is as follows:

- Object detector: which is responsible for detecting people in the camera feeds going in and out of each place/room in the building.
- Object tracker: which enhances the system robustness and reliability by avoiding collisions and frame drops due to networking issues.
- Object counter: which all the counting process happens. For each room/place it gives the current occupancy status.

CHAPTER 4

TECHNOLOGY ASSESSMENT

Computer vision algorithms have become increasingly popular and necessary in a variety of fields, including robotics, autonomous vehicles, and security systems. These algorithms require significant computational resources to process large amounts of image data in real time. As a result, the hardware implementation of computer vision algorithms is critical for their success. In this chapter, we review different hardware implementation techniques for computer vision algorithms, ranging from small size computational devices up to servers and cloud technologies. We also present a comparison between different hardware implementation architectures and propose a design for computation hardware specifications for a real-time occupancy profile detection system. Additionally, we list the libraries, frameworks, and software used in our implementation.

4.1 Computation Hardware

The computation hardware is the device that processes the camera feeds and hosts the computer vision and AI computation models. Generally, computer vision hardware computation devices can be categorized into three architectures:

4.1.1 Edge

Edge devices are becoming increasingly popular due to their ability to run machine learning models despite their small size in terms of computational power, memory, and RAM. These devices are capable of handling single camera feeds, and they are directly connected to the camera to ensure reliability and speed with no frames being dropped during processing shown in Fig. 4.1.

There are various types of IoT edge devices available on the market, including the Nvidia Jetson family, Raspberry Pi, and Microsoft Movidius. The decision to use an edge device largely depends on the specific situation and a number of factors, such as budget, environment, the number of places being considered, and scalability.

Edge devices are ideal for scenarios where only a single place or very few places are being considered. They are cost-effective and highly reliable, making them suitable for small-scale projects. They are also highly scalable, meaning that additional edge devices can be added as needed to handle more cameras or increase processing power.

However, there are some limitations to using edge devices. For instance, they have limited memory and computational power, which may make it difficult to handle large volumes of data. They may also require additional hardware components such as sensors and storage devices, which can increase the cost of the project.

Another important factor to consider when using edge devices is the choice of machine learning model. Not all machine learning models are suitable for edge devices due to their limited processing power and memory. As a result, it is essential to choose models that are lightweight and optimized for edge devices.

Overall, the use of edge devices is a situational decision that depends on a variety of factors. They are an ideal choice for small-scale projects where only a few places are being considered. However, they may not be suitable for large-scale projects that require more processing power and memory. Careful consideration must be given to the choice of machine learning models, hardware components, and scalability when using edge devices. In conclusion, edge devices are a powerful tool for running machine learning models in scenarios where computational power, memory, and RAM are limited. They are cost-effective, highly reliable, and scalable, making them ideal for small-scale projects. However, careful consideration must be given to the specific situation, including the choice of machine learning models, hardware components, and scalability, when using edge devices.

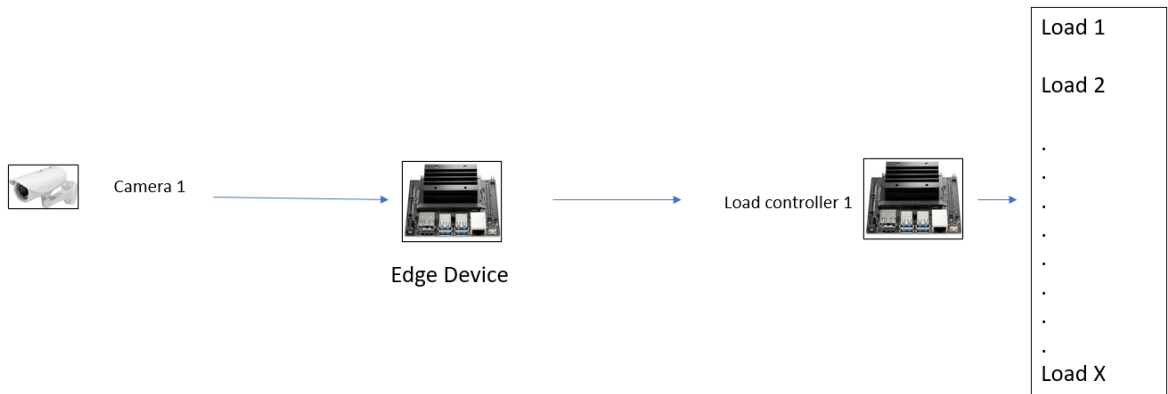


Figure 4.1.1.1 IoT edge device architecture

4.1.2 Servers

In the previous section, we discussed the use of edge devices for running machine learning models on single camera feeds. However, when many places or rooms need to be monitored, the use of computational servers becomes a more viable option.

In this architecture, the cameras are connected to a network and are accessible to the computational server as shown in Fig. 4.2. The server is responsible for processing the frames from the cameras and running machine learning models to detect occupancy or perform other computer vision tasks. This architecture provides greater scalability as the server can handle multiple camera feeds simultaneously.

The specification of the server can vary greatly depending on the number of cameras and the required processing power. For a small number of cameras, a standard desktop computer may suffice. However, for larger deployments, a dedicated server with multiple GPUs may be required.

One advantage of this architecture is that the computational server can be located in a separate room or even in a remote location. This allows for greater flexibility in deployment as the cameras can be located in areas where it may not be possible or practical to have a dedicated computing device.

Another advantage of this architecture is that the server can be designed to handle different types of computer vision tasks, such as object detection or facial recognition. This provides greater versatility and allows for the system to be adapted to different use cases.

In our occupancy detection system, we used a workstation with a high-end CPU and a single GPU. The CPU was an Intel Core i9-10900K, which provided the necessary processing power for handling multiple camera feeds. The GPU was an NVIDIA GeForce

RTX 3080, which provided the necessary compute power for running machine learning models.

In addition to the hardware specifications, the choice of software libraries and frameworks is also important. We used the TensorFlow framework for building our machine learning models, which provided a high-level API for building and training models. We also used OpenCV for processing the camera feeds and performing other computer vision tasks.

Overall, the use of computational servers for running machine learning models on camera feeds provides greater scalability and flexibility compared to edge devices. However, it is important to carefully consider the hardware specifications and software choices to ensure that the system can handle the required processing load and provide accurate results.

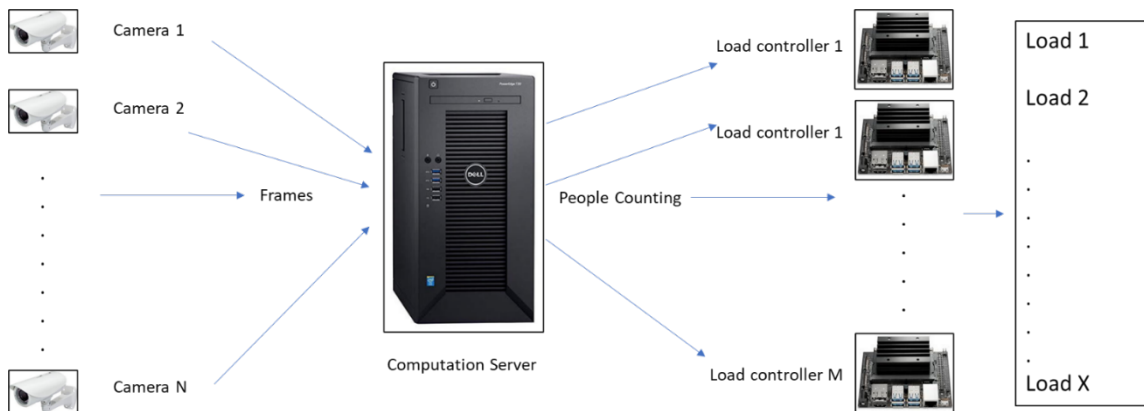


Figure 4.1.2.1 Workstation & server architecture

4.1.3 Cloud

In recent years, cloud computing has become an increasingly popular option for many applications, including Computer Vision. Cloud-based architectures offer a number of advantages over other architectures, but also present some challenges.

One of the main advantages of cloud-based architectures is their scalability. Cloud servers can be easily scaled up or down to handle varying workloads, making them an ideal choice for applications that require processing power beyond what is available on edge devices or local servers as shown in Fig. 4.3. Additionally, cloud-based architectures can provide access to a wide range of hardware and software resources, enabling developers to build and deploy more complex Computer Vision applications.

However, there are also some challenges associated with cloud-based architectures. One of the biggest challenges is the issue of reliability. Cloud-based architectures require an internet connection at all times, which means that any disruptions or outages can severely impact the performance of the system. Additionally, cloud-based architectures can be costly, particularly when large amounts of processing power are required.

Despite these challenges, many companies and organizations have found cloud-based architectures to be an effective solution for their Computer Vision applications. Companies like Google, Amazon, and Microsoft Azure offer cloud computing services that can be used to run machine learning models for Computer Vision applications.

One potential use case for a cloud-based Computer Vision architecture is in surveillance systems. Cloud-based architectures can be used to process feeds from multiple cameras in real time, enabling security personnel to monitor multiple locations from a single location. Additionally, cloud-based architectures can be used to provide analytics on the data

collected from these cameras, enabling businesses to better understand customer behavior and improve operational efficiency.

Another potential use case for cloud-based Computer Vision architectures is in autonomous driving systems. Cloud-based architectures can be used to process data from multiple sensors, including cameras, LiDAR, and radar. This data can then be used to train machine learning models that can help autonomous vehicles navigate safely and efficiently.

In terms of software and frameworks, there are a number of options available for building cloud-based Computer Vision applications. Some popular options include TensorFlow, PyTorch, and Caffe. Additionally, cloud computing providers like Google and Amazon offer their own machine learning frameworks that can be used to build and deploy Computer Vision applications.

In conclusion, cloud-based architectures offer a number of advantages for Computer Vision applications, including scalability and access to a wide range of hardware and software resources. However, they also present some challenges, including reliability and cost. Despite these challenges, cloud-based architectures have become an increasingly popular option for many companies and organizations looking to build and deploy complex Computer Vision applications.

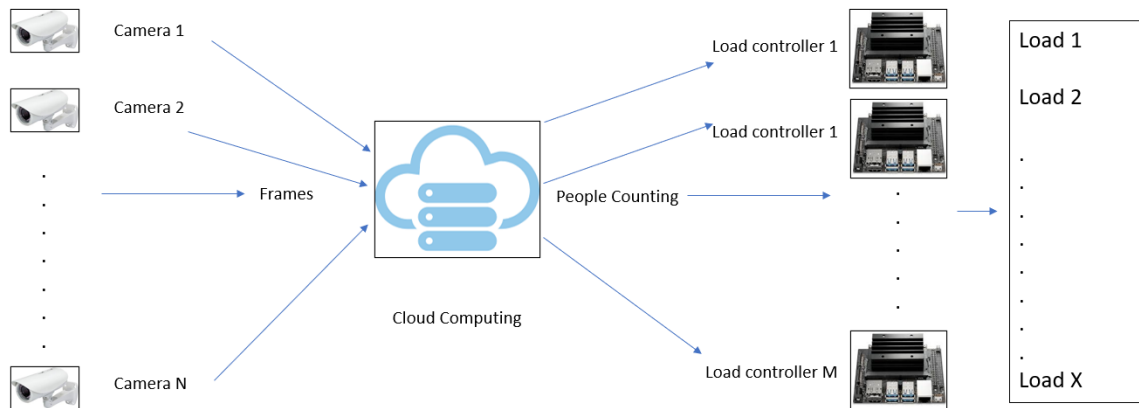


Figure 4.1.3.1 Cloud computing architecture

In this described experiment the computational hardware selected involves using a workstation to host the computation models and analyze camera feeds for occupancy detection in a building. The system is designed to process the feeds from multiple cameras and output the occupancy status for each room and place in the building. The analysis done on this server is object detection, tracking, and counting. The size of the computation server required depends on the number of camera feeds it should process.

To achieve high occupancy detection accuracy, each camera has to be processed at least with 6-8 Frames Per Second (FPS). This means that the workstation has to be powerful enough to process the camera feeds at this rate. Increasing the FPS beyond the saturation point contributes very little to the accuracy of the occupancy detection, but it consumes more hardware resources. Therefore, a balance value of FPS should be selected to achieve the best accuracy with minimum hardware required.

In this experiment, an FPS value of 8 is selected, which is found to be the optimal balance point for achieving the highest accuracy with minimum hardware required. However, this value can be subject to change based on the environment. For instance, a more crowded environment will require a higher value of FPS for the system to maintain high occupancy detection accuracy. It's important to note that the computational capabilities of the hardware, including the CPU, GPU, and RAM speeds and capacities, are the most important factors that contribute to the performance of the system. Therefore, the selected computation server device should satisfy the hardware requirements based on the number of cameras and the required FPS.

Increasing the number of cameras also increases the hardware resources required. For example, if a building has 12 cameras, a computation power of 96 FPS (12 cameras * 8 FPS) is required. Thus, the selected computation server device should satisfy such requirements. Table 4.1 shows the hardware specs and prices associated with the computation device used in the lab.

This experiment has important implications for building management systems that rely on occupancy detection to optimize energy usage and indoor air quality. By using a powerful workstation with the right hardware specifications, building managers can optimize the accuracy of occupancy detection while minimizing hardware requirements and costs. This can help reduce the carbon footprint of buildings and improve the quality of life for occupants.

Table 4.1 Hardware Specification

Specs	Price
• CPU core i7 • 8 Gbyte RAM	4,000 SAR

4.2 Cameras

The proposed system is an innovative solution for building managers who want to monitor occupancy and optimize energy usage without having to install additional infrastructure or do any civil works. One of the main advantages of this system is that it is integrated with the security cameras that are already installed in the building. This means that building managers can leverage existing resources to implement occupancy detection without incurring additional costs or disruptions.

To capture the camera feed, real-time streaming protocol (RTSP) is used. RTSP is a network control protocol that is used to establish and control media sessions between endpoints. In this case, RTSP is used to capture the camera feed and stream it to the occupancy detection system in real-time. This ensures that building managers have access to the most up-to-date occupancy data, which they can use to optimize energy usage and improve indoor air quality.

Once the camera feed has been captured using RTSP, the frames are sent to the occupancy detection system to be analyzed. The analysis is done in real-time using object detection, tracking, and counting algorithms. These algorithms use machine learning and computer vision techniques to detect the presence of people in the camera feed and track their movements. The system can also count the number of people in a room or area and generate reports based on this data.

RTSP is a network control protocol that is used to establish and control media sessions between endpoints. In the context of the occupancy detection system, RTSP is used to capture the camera feed and stream it to the system in real-time. This ensures that the occupancy data is current and accurate, allowing building managers to make data-driven decisions about building operations and energy usage.

The real-time nature of RTSP is critical for the effective operation of the occupancy detection system. Since the system relies on up-to-date occupancy data, any delay or lag in the RTSP feed could result in inaccurate occupancy data. By using RTSP to capture the camera feed and stream it to the system in real-time, the system can ensure that the occupancy data is accurate and reliable.

Moreover, RTSP enables the occupancy detection system to operate without the need for additional infrastructure or civil works. Since the system is integrated with the security cameras that are already installed in the building, there is no need to install additional cameras or other equipment. This significantly reduces the cost and disruption associated with implementing the occupancy detection system.

In summary, the real-time streaming protocol (RTSP) is a critical component of the proposed occupancy detection system. By leveraging RTSP to capture camera feeds in real-time and stream them to the occupancy detection system for analysis, building managers can optimize building operations, improve energy efficiency, and enhance building security. The real-time nature of RTSP ensures that the occupancy data is accurate and reliable, allowing building managers to make data-driven decisions about building operations and energy usage.

4.3 Actuators/ Thermostatically Controlled Loads

Actuators, in the context of building automation, are devices that are used to control and regulate various building systems, such as HVAC, lighting, and electrical appliances. These devices are an essential component of any smart building automation system, as they enable the system to take actions based on data collected from various sensors and other sources. The proposed system discussed in this section can be integrated with different types of actuators, including thermostats, switches, and external dashboards, to apply a specific controlling strategy based on the application and building.

One of the primary controlling strategies that can be applied using the proposed system is the control of the HVAC set point based on the number of occupants in each area of the building. This strategy is designed to optimize energy consumption and reduce costs

associated with heating, cooling, and ventilation. By dynamically adjusting the HVAC set point based on occupancy data, the system can achieve significant energy savings without compromising occupant comfort.

Initial experiments conducted on the proposed approach have shown that for each change in the set point temperature, there is a corresponding energy reduction of around 4%. This is a significant saving, given that HVAC systems typically account for a large portion of a building's energy consumption. The potential energy savings that can be achieved using the proposed system will depend on various factors, including the size of the building, the number of occupants, and the local climate.

In addition to controlling the HVAC system, the proposed system can also be used to control other electrical appliances and devices, such as lighting. The system can be integrated with smart switches and sensors to control the lighting in different areas of the building based on occupancy data, ambient light levels, and other factors. This can help reduce energy consumption and optimize the use of lighting in the building.

The experimental prototype of the proposed system shown in Fig. 4.4 consists of several components, including cameras, computation hardware, and a microcontroller to control the load operation simulated in the RTDS. The cameras are used to capture real-time occupancy data, which is then processed by the computation hardware to determine the optimal control strategy for the various actuators in the building. The microcontroller is responsible for controlling the operation of the loads, such as HVAC systems and lighting, based on the control strategy determined by the computation hardware.

One of the advantages of the proposed system is that it can be integrated with the existing infrastructure of a building without the need for significant modifications or civil works.

The system uses real-time streaming protocol to capture camera feed, which is then processed by the computation hardware. This means that the system can be easily installed in a building that already has security cameras in place, without the need for additional infrastructure modifications.

In summary, the proposed system discussed in this section can be integrated with various types of actuators, such as thermostats, switches, and external dashboards, to apply a specific controlling strategy based on the application and building. The system can be used to control HVAC systems, lighting, and other electrical appliances, with the potential to achieve significant energy savings. The experimental prototype of the proposed system consists of cameras, computation hardware, and a microcontroller to control the load operation simulated in the RTDS, and can be easily integrated with the existing infrastructure of a building.

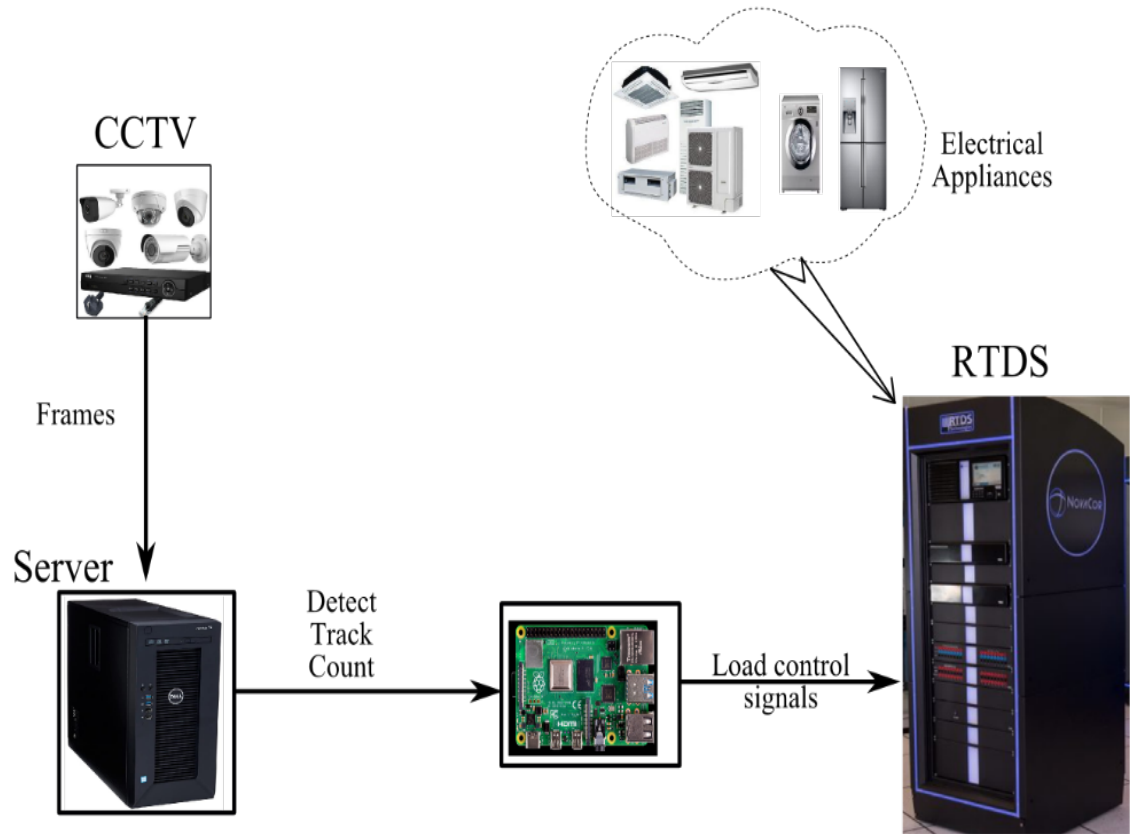


Figure 4.3.1 Targeted laboratory prototype for the proposed system.

4.4 Software Libraries

In the implementation of computer vision algorithms, Python is used as the primary programming language. The development of computer vision workflows requires the use of toolkits and libraries that provide efficient and optimized algorithms for processing and analyzing images. There are several libraries available that are considered state of the art in the field of computer vision.

In this section, we will discuss the different libraries and frameworks used for implementing the computer vision algorithms discussed in this chapter. These libraries are considered state-of-the-art and are maintained by big teams and organizations.

4.4.1 TensorFlow

TensorFlow is an end-to-end open-source platform for machine learning. It has a flexible ecosystem of tools and libraries that allows developers to easily build and deploy machine learning-powered applications. TensorFlow provides a wide range of APIs that are designed for different levels of abstraction, making it suitable for various use cases. Its high-level APIs allow developers to quickly prototype models, while its low-level APIs provide greater flexibility and control over model development.

One of the main features of TensorFlow is its ability to handle large datasets. It provides tools for distributed training and inference, making it possible to scale machine learning applications to handle massive amounts of data. TensorFlow also supports a variety of hardware, including GPUs and TPUs, making it possible to train models quickly and efficiently [60].

4.4.2 OpenCV

OpenCV is a popular and open-source computer vision and machine learning software library. It is used extensively for image processing techniques and image manipulation. OpenCV provides a comprehensive set of tools for various image processing tasks such as filtering, feature detection, object recognition, and more. It also provides a variety of functions for camera calibration, stereo vision, and optical flow.

OpenCV is widely used in both academia and industry. It has been used in a wide range of applications, including robotics, augmented reality, and self-driving cars. OpenCV is written in C++, but it also has APIs for Python, Java, and MATLAB, making it accessible to developers with different programming backgrounds [61].

4.4.3 Scikit-Learn

Scikit-learn is an open-source machine learning library that provides a range of tools for model fitting, data preprocessing, model selection, and evaluation. It supports both supervised and unsupervised learning algorithms, making it suitable for a wide range of applications. Scikit-learn provides a variety of machine learning algorithms, including linear models, decision trees, support vector machines, and more.

One of the main features of scikit-learn is its ease of use. It provides a simple and consistent API that makes it easy for developers to use different machine learning algorithms and evaluate their performance. Scikit-learn also provides tools for data preprocessing, including scaling, normalization, and encoding of categorical variables.

Scikit-learn has been widely used in various industries, including healthcare, finance, and marketing. It is also used extensively in research, where it is used for data analysis and exploration [62].

4.5 Summery

This chapter provided an overall comparison of different hardware techniques for implanting computer vision algorithms used in the proposed approach discussed in the previous chapter. Specifically, it looked at the Edge, Servers, and cloud computing architectures and comparison between these different implantation styles. Additionally, a specific computer vision hardware specifications design for the experimental setup is also described. Finally, an overview of the most used libraries in the proposed approach is also looked at such as Tensorflow, OpenCV, and Scikit-learn

CHAPTER 5

IMPLEMENTATIONS & RESULTS

This chapter provides details of environmental thermal models used, namely Environment 1-Hotel building- and environment 2 -Room-. Additionally, a simulation for each environment with different case studies is also presented. Finally, results of each case study and simulation are provided.

5.1 Power System

In order to conduct a power analysis of a hotel building, it is important to first identify the building's detailed properties and all of the base assumptions needed to develop a modeling method. This includes factors such as the building's size, orientation, materials used in construction, and occupancy levels. By understanding these details, it becomes possible to develop an accurate model that can be used to assess the energy consumption of the building and identify areas where improvements can be made.

Once the building properties have been identified, the next step is to develop a modeling method that takes into account all of the relevant factors that contribute to energy consumption. This typically involves using computer simulation software to create a detailed model of the building, including the layout of the rooms, the types of heating and cooling systems used, and the location and size of windows and doors, more on this in the following sections.

With the modeling method in place, it is then possible to perform actual calculations to determine the energy consumption of the building. This typically involves analyzing the energy usage data from the building's various systems, such as HVAC, lighting, and appliances, and calculating the energy requirements for each component of the building.

In addition to the building-wide power analysis, it is also important to conduct an experimental setup for the computer vision system in a specific room environment. This typically involves creating a thermal profile of the room, modeling the air conditioning system, and incorporating a thermostat to regulate the temperature.

By conducting this type of analysis, it becomes possible to identify areas where energy consumption can be reduced and to develop strategies for optimizing the energy efficiency of the building. This can include everything from upgrading HVAC systems and insulation to incorporating smart home technology and implementing energy-saving policies and procedures.

One key benefit of conducting a power analysis of a hotel building is that it can help to reduce the overall energy consumption of the building, which can have a positive impact on the environment as well as on the bottom line for hotel owners and operators. By understanding the specific factors that contribute to energy consumption in a hotel setting, it becomes possible to develop targeted strategies for reducing energy usage and improving overall efficiency.

In this section, the power analysis of the hotel is addressed. Firstly, the building detailed properties are identified along with all base assumptions needed. Secondly, the modeling method is developed along with the actual calculations. After that, an experimental setup

for the computer vision system is executed for the room environment including the thermal profile, Ac modeling and thermostat.

5.2 Environment Modeling

5.2.1 Environment 1 (Hotel Building Rooms)

A. Hotel Building Assumptions

The case study under consideration involves a hotel building with a specific layout and a set of parameters that have been identified for the purpose of power analysis. The building has N rooms, with each side having $N/4$ rooms, distributed over four levels. The AC rating of each room is 5.2 kWh, and the dimensions of each room are provided in terms of width, length, and height (**w**, **l**, & **h**) respectively. The longer side of each room is assumed to be facing the outside, and there is a glass window of A_{window} m² in each room.

To perform power analysis for the hotel building, several base assumptions were made, such as the indoor design temperature being assumed to be T_i C°, and the maximum outdoor design temperature being T_{out} C°. These assumptions are essential in developing an accurate model for the thermal behavior of the building.

Additionally, the values of parameters used for the AC power consumption calculation are listed in Table 5.1, which includes all parameters needed for the model.

B. Cooling Loads

The cooling loads refer to the amount of heat that must be removed from a building to maintain a comfortable indoor temperature. In general, there are two types of cooling loads: external cooling loads and internal cooling loads.

- External cooling loads are generated by heat transfer from external sources to the conditioned space through walls, roofs, ceilings, glass windows, and partition walls. These sources include outdoor air temperature, solar radiation, and heat transfer from adjacent buildings or surfaces.

The external cooling loads depend on various factors such as the orientation and size of the windows, the type and thickness of the building materials used, and the weather conditions. For example, a building with large windows facing west may experience higher external cooling loads during the afternoon, when the sun is at its highest point, compared to a building with small windows or windows facing east.

- Internal cooling loads are formed by the release of sensible and latent heat from internal sources like humans, electrical appliances, and electric lights. Sensible heat refers to the heat released by an object that can be sensed or measured by a thermometer, while latent heat refers to the heat released by an object due to moisture or water vapor.

The internal cooling loads are dependent on the occupancy level, the type and number of electrical appliances, the type and wattage of lighting fixtures, and the ventilation rate. For example, a room with a large number of occupants and several electrical appliances may experience higher internal cooling loads compared to a room with few occupants and no electrical appliances.

The total cooling load is the sum of the external and internal cooling loads. Accurately calculating the cooling loads is crucial for designing an efficient and effective cooling system. By understanding the sources of cooling loads, building designers and engineers

can make informed decisions regarding the building orientation, window size and placement, insulation, and HVAC system design.

C. Load Calculations

In the proposed model, the cooling loads are calculated using a comprehensive approach that takes into account several parameters. The model is based on the approach outlined in [63], which considers various factors such as the room structure, occupancy, and the characteristics of the heating unit.

To calculate the AC power needed for each room at a desired setpoint temperature and given outdoor temperatures, equation (4) is used. This equation takes into account both internal and external cooling loads. The external cooling loads are due to the heat gains from external sources such as walls, roofs, ceilings, glass windows, and partition walls, while the internal cooling loads are due to the release of sensible and latent heat from internal sources like humans, electrical appliances, and electric lights.

The equation for calculating the required AC power for each room is as follows:

$$G_i = \left(\frac{A_{wall}}{R_{wall}} + \frac{A_{window}}{R_{window}} + \frac{A_{ceiling}}{R_{ceiling}} \right) \times (T_{out} - T_i) + SHGC \times A_{window_{south}} \times H_{solar} \times \gamma + H_p \quad (4)$$

where A_{wall} , A_{window} , $A_{ceiling}$ are the area of the wall, window, and the ceiling of the room in m^2 , respectively. R_{wall} , R_{window} , $R_{ceiling}$, are the thermal resistances of the wall, window, ceiling of the room, respectively. $T_{out,i}$ is the outdoor temperature at instant i ($^{\circ}F$), $SHGC$ is the Solar Heat Gain Coefficient, H_{solar} , is the solar radiation thermal power measured in

Wh/m², and γ is a measurement constant. All the values of areas and resistances are available in TABLE 1. By substituting the values into 4), the heat needed to maintain the desired AC setpoint temperature could be obtained.

After calculating the cooling load for each room, the total cooling load for the entire building was obtained by summing up the cooling loads of all the rooms. This value was then used to calculate the power consumption of the AC system for the building. Once the total cooling load of all the rooms is calculated for each AC setpoint temperature, the total energy consumption can be determined in kWh for the entire duration that the room is unoccupied at each AC setpoint temperature. A comparative analysis of the energy consumption for all the AC setpoint temperatures is then carried out. To illustrate the process, an example of a hotel building is considered in this thesis, and its AC power analysis is discussed in detail. The system is designed to automatically assign a fixed AC setpoint temperature to the thermostat when a room or space is unoccupied. This ensures that energy is not wasted unnecessarily in cooling unoccupied rooms, thereby reducing energy costs and promoting energy efficiency.

D. Case Studies

The proposed approach for calculating AC power consumption using heat transfer equations and HVAC load calculations has been thoroughly tested through simulations. The simulations have been conducted for two cases - Case 1 and Case 2.

- In Case 1, the range allowed for the rooms to be empty is stochastic, which means that the rooms can be empty for any duration of time within a given 1range.
- In Case 2 however; the range is fixed and uniform across all time periods.

For each simulation, various assumptions and constraints were taken into consideration. For instance, in Case 1, the simulation assumed that the occupancy patterns of the rooms are stochastic, and the duration for which the rooms remain empty can be of varying lengths. Moreover, the simulation assumed that the outdoor temperature differs depends on the time of the day as discussed in the section below. On the other hand, in Case 2, the simulation assumed that the occupancy patterns are uniform, and the duration for which the rooms remain empty is fixed.

Table 5.1 Parameters Used to Calculate the Power Consumption

Parameters	Value	Parameters	Value
A_{Walls}	63 (m^2)	R_{Window}	$0.8 - 1 \left(\frac{^{\circ}F \times ft^2 \times h}{Btu} \right)$
A_{Ceiling}	30 (m^2)	SHGC	0.71
A_{Windows}	3 (m^2)	C_{HVAC}	140 ($\frac{Btu}{kWh}$)
R_{wall}	13-15 ($\frac{^{\circ}F \times ft^2 \times h}{Btu}$)	H_{Solar}	140 ($\frac{Wh}{m^2}$)
R_{Ceiling}	38 – 60 ($\frac{^{\circ}F \times ft^2 \times h}{Btu}$)	γ	$\frac{3.412}{10.76} \left(\frac{Btu.m^2}{Wh.ft^2} \right)$
T_i	20°	T_{out}	46°
N	80	W	5 m
L	6 m	H	3 m

A. Case 1: Stochastic Availability Times

The proposed approach involves simulating different occupancy scenarios to calculate AC power consumption. In Case 1, thirty stochastic scenarios are considered for each room to represent the occupancy status, whether a room is empty or occupied. Each scenario consists of 18-hour time slots representing the time from 6:00 AM to 12:00 midnight. Each slot can either be 0, which means the room is empty, or 1, which means the room is occupied.

To illustrate, Fig. 5.1 shows a single stochastic scenario, where the blue dots represent the occupied rooms at a specific time. In this scenario, the first room is occupied for the first 17 hours and not occupied for only one hour of the day. The assumption of considering only 18-hour periods is based on the fact that guests are usually in their rooms during the night and early morning hours.

For each scenario in Case 1, the algorithm considers variable constraints for the maximum and minimum hours allowed for the room to be empty. For instance, in one scenario during the morning hours, the maximum and minimum allowed hours for the room to be empty are [6,4]. This means that the algorithm will make a random choice between 4 and 6 hours for the room to be empty during that period. This assumption is based on the nature of hotels where people usually use the morning hours to be outside, either on vacation or on business days. Other scenarios during the morning, afternoon, and evening periods have different bound limits, such as [3,5] and [2,4], to simulate other behaviors.

In summary, Case 1 involves simulating 30 different occupancy scenarios for each room, where the occupancy status is represented by 18-hour time slots. The maximum and

minimum hours allowed for the room to be empty are variable and depend on the time of day and the specific scenario being considered.

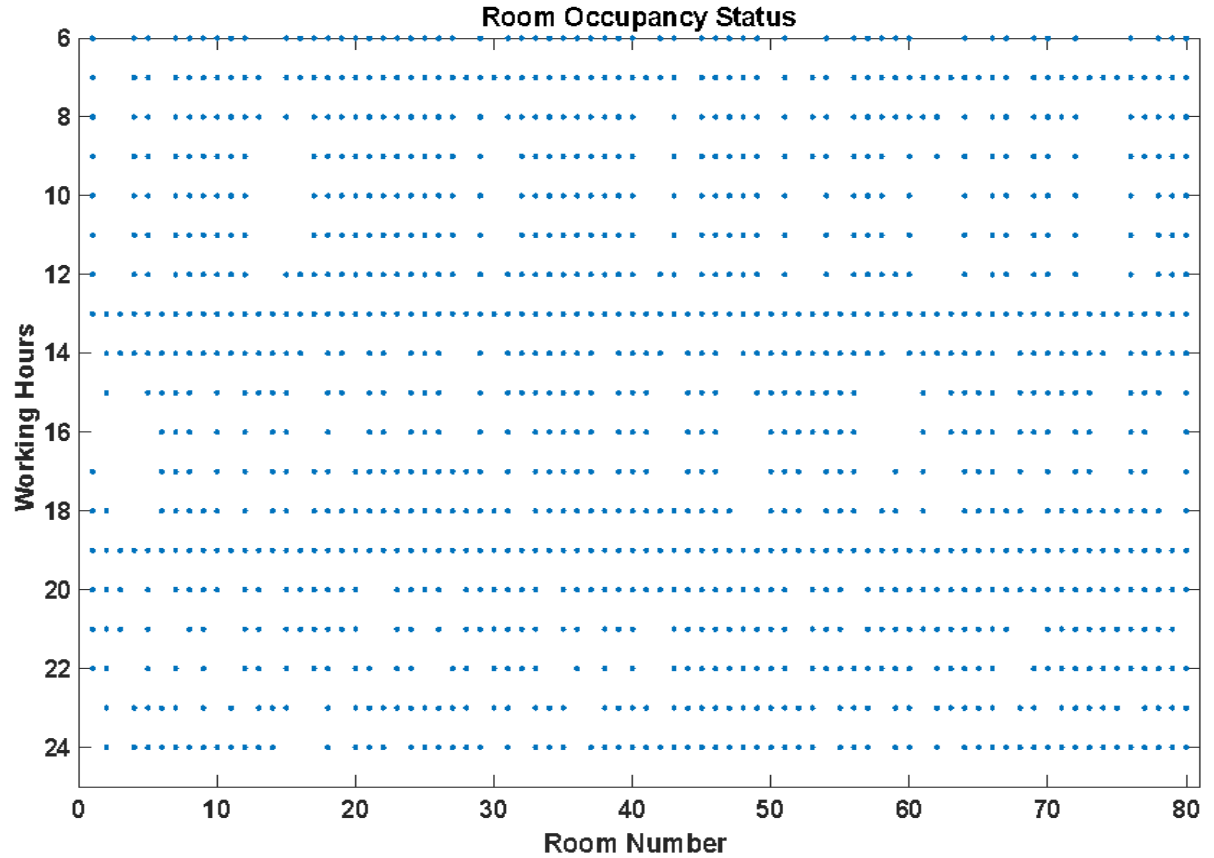


Figure A.1 Sample of single scenario of occupancy status for each room in the hotel building. Blue dot refers to occupied room.

B. Case 2: Fixed Availability Times

In Case 2, the same number of scenarios and test periods as in Case 1 were applied. The aim of Case 2 was to observe the change in energy savings resulting from varying the constraints. Unlike Case 1, the constraints for the maximum and minimum allowed time for the room to be empty were fixed and uniform across all scenarios and periods. In this case, the maximum and minimum allowed time for the room to be empty were set to [1,6].

To obtain the energy consumption data for each scenario in Case 2, the algorithm for each period, i.e., morning, afternoon, and evening, randomly selected a time between 1 and 6 hours for the room to remain empty. This assumption is based on the belief that most guests would only leave their rooms for a few hours during the day, with the majority of the time spent indoors. By fixing the constraints, the comparison of energy consumption between different AC setpoints can be carried out with greater accuracy.

E. Calculation and Comparison

The proposed approach calculates the AC power consumption based on each room's occupancy status, as shown in Fig. 5.1. The calculation is done only when the room is empty to reflect the energy that can be saved if the proposed approach is applied. The 18 hours of the day are divided into three sub-periods, each assumed to have a different outside reference temperature of 46°C, 42°C, and 39°C. During each sub-period, the energy is calculated for the total duration that the guests are outside of their rooms. TABLE 2 shows the energy consumption per hour for each sub-period, indicating that the energy consumption is higher in the morning and afternoon hours than in the evening hours.

In Case 1, thirty stochastic scenarios are considered for each room to represent the occupancy status, and the constraints for the maximum and minimum hours allowed for the room to be empty are variable. The guest preference setpoint energy consumption is calculated for each scenario, and Table 5.2 shows the results along with the AC energy consumption in kWh when changing the setpoint from 21°C to 25°C. The table reveals that a one-degree increase in the setpoint temperature can save up to 4% of the total power consumption in an empty room.

In Case 2, the same number of scenarios are applied with fixed and uniform constraints for all scenarios in all periods. The maximum and minimum allowed time for the room to be empty are [1,6], and the algorithm randomly selects a time between 1 and 6 for each period - morning, afternoon, and evening. The results of Case 2 are listed in Table 5.4, showing an increase in energy consumption compared to Case 1, but the saving percentage remains the same. This increase is due to the nature of the constraints in Case 2, where the selection of the room's empty bounds ranges between 1 and 6, while in Case 1, some scenarios had bounds that did not exceed a specific limit. Overall, the proposed approach can save up to 4% for each HVAC setpoint from 20oC, which is the most commonly selected setpoint by users [30].

Table 5.2 Energy Consumption In kWh For Different Setpoint Ac Temperature Concerning Three Different Outside Temperatures for Three Different Sub-Periods.

Sub-period		Morning	Afternoon	Evening
Outside Temp.		46 °C	40 °C	39 °C
Setpoints	18 °	1.64	1.4	1.3
	19 °	1.59	1.3	1.26
	20 °	1.54	1.26	1.21
	21 °	1.5	1.22	1.16
	22 °	1.45	1.17	1.12
	23 °	1.4	1.12	1.7
	24 °	1.36	1.07	1.02
	25 °	1.31	1.03	0.98

Table 5.3 Shows the Total Power Consumption in kWh for Each Scenario for Different Ac Setpoint Ranging from 20° to 25° In Case 1.

	AC Energy Consumption in kWh at different setpoints					
Scenario #	20°	21°	22°	23°	24°	25°
1	593	572	551	531	510	489
2	431	416	400	385	369	354
3	461	444	427	411	394	377
4	366	353	339	326	313	300
5	576	556	536	517	497	477
6	508	490	471	453	435	417
7	393	378	364	350	336	322
8	405	390	376	361	346	331
9	551	532	512	493	474	455
10	545	526	507	487	468	448
11	366	352	339	326	313	299
12	387	373	359	345	331	316
13	536	518	499	481	462	444
14	507	489	471	453	435	417
15	403	389	374	360	345	330
16	411	396	381	366	350	335
17	512	494	476	458	440	422
18	491	474	457	439	422	404
19	450	434	418	401	385	368
20	364	351	337	324	311	298
21	582	562	542	522	501	481
22	437	421	406	390	375	359
23	399	385	370	356	341	326
24	387	373	359	345	331	317
25	548	529	510	491	472	453

26	470	453	436	420	403	387
27	432	416	400	385	369	353
28	385	371	357	343	329	315
29	571	552	532	512	492	472
30	483	466	449	431	414	397
Total	13,951	13,454	12,957	12,459	11,962	11,465
% Saving with respect to 20°	0	4	7	11	14	18

Table 5.4 Shows the Total Energy Consumption in kWh for Each Scenario for Different Ac Setpoint Ranging from 20° to 25° In Case 2.

	AC Energy Consumption in kWh at different setpoints					
Scenario #	20 °	21 °	22 °	23 °	24 °	25 °
1	1,100	1,062	1,023	985	946	907
2	1,156	1,115	1,075	1,035	995	955
3	1,173	1,132	1,091	1,050	1,009	968
4	1,108	1,069	1,030	991	952	913
5	1,138	1,098	1,058	1,018	978	938
6	1,117	1,078	1,040	1,001	962	923
7	1,073	1,036	998	960	923	885
8	1,058	1,021	984	947	910	873
9	1,168	1,127	1,086	1,045	1,005	964
10	1,056	1,019	981	944	907	870
11	1,078	1,040	1,003	965	927	889
12	1,099	1,060	1,022	983	945	906
13	1,090	1,052	1,014	975	937	899
14	1,131	1,091	1,051	1,012	972	932
15	1,141	1,101	1,061	1,022	982	942
16	1,183	1,142	1,100	1,059	1,017	976
17	1,093	1,054	1,016	977	939	900

18	1,140	1,100	1,060	1,020	981	941
19	1,155	1,115	1,075	1,034	994	953
20	1,138	1,098	1,058	1,018	978	938
21	1,097	1,059	1,020	982	943	905
22	1,120	1,081	1,041	1,002	963	923
23	1,119	1,080	1,041	1,001	962	923
24	1,140	1,100	1,059	1,019	979	939
25	1,070	1,032	995	957	920	882
26	1,090	1,052	1,014	976	938	900
27	1,090	1,052	1,014	976	938	899
28	1,166	1,125	1,084	1,043	1,002	961
29	1,141	1,101	1,061	1,021	981	941
30	1,072	1,034	997	959	922	884
Total Energy [kWh]	33,500	32,326	31,152	29,978	28,804	27,630
% Saving with respect to 20°	0	4	7	11	14	18

5.2.2 Environment 2 (Individual Room Simulations)

In this section of the thesis, the aim is to create a model of the room environment using MATLAB Simulink. Fig. 5.2. shows the complete system diagram for the room environment. The room represents the physical space in the commercial building that needs to be controlled for its thermal comfort and energy efficiency. Different models should be taken in consideration while modeling the room model that can get too complicated or it could be simplified depends on the situation at hand. In the next section a detailed Model components is discussed providing wide range of parameters that can represent this environment. In the later section the experiment model parameters used for this setup is discussed with the mathematical models.

A. Room Environment Model

In this section, we delve into the comprehensive internal models that significantly influence the thermal profile of the room environment. These internal models play a crucial role in determining the overall temperature and energy dynamics within the room, considering various structural specifications and occupancy patterns.

a. Room Model.

The room model aims to capture the thermal behavior of the commercial building's interior space. This can be achieved by considering several important factors:

- Heat Transfer Equations: Formulate heat transfer equations based on conduction, convection, and radiation principles. These equations will account for heat exchange

between the room's interior surfaces and the surrounding environment, considering factors like thermal conductivity, surface area, and temperature gradients.

- Thermal Mass: Include the thermal mass of the room's structural elements and furnishings. Thermal mass affects the room's ability to store and release heat, influencing its response to temperature changes.
- Room Volume and Dimensions: The room's volume and dimensions directly impact the total heat capacity and the cooling load required to maintain the desired temperature.
- Windows and Doors: Consider the impact of windows and doors on heat transfer. Include factors like window area, glass properties (insulating or reflective), and whether doors are frequently opened or closed.
- Occupancy and Internal Heat Sources: Incorporate the heat generated by occupants, electronic equipment, and lighting. The number of occupants and their activities influence the internal heat load.
- External Heat Sources: If there are any external heat sources (e.g., sunlight entering through windows), account for their contribution to the room's heat gain.
- Transient Effects: Consider transient effects, as temperature changes in a room are not instantaneous. Thermal lag and time constants are relevant here.

b. Cooling Load Model

The cooling load model quantifies the amount of heat that needs to be removed from the room to maintain a comfortable temperature. This model can be developed by considering the following factors:

- Heat Gain Components: Sum up all the heat gain components, such as solar heat gain through windows, internal heat sources (occupants, equipment, lighting), and external environmental heat.
- Heat Loss Components: If relevant, include heat loss components like conduction through the walls and windows in colder climates.
- Climate Data: Consider the external environmental conditions, including outdoor temperature, solar radiation, and humidity levels. These will vary throughout the day and year.
- Thermal Properties of Materials: Account for the thermal properties of walls, windows, and other materials that influence heat transfer.
- Infiltration and Ventilation: Include the impact of air infiltration and ventilation on the cooling load. For instance, on hotter days, more outside air may need to be cooled, affecting the load.

c. AC Model

The AC model represents the air conditioning system responsible for cooling the room. It should consider the following aspects:

- Cooling Capacity: The AC system's cooling capacity determines how much heat it can remove from the room per unit of time. It is influenced by the AC system's design, size, and efficiency.
- Energy Consumption: Assess the energy consumption of the AC system based on its cooling capacity and operational characteristics. This could involve a relationship between cooling capacity and power consumption.

- **Thermostat Control Logic:** The AC system's control logic may involve on/off cycling, variable-speed operation, or other strategies to maintain the desired temperature setpoint efficiently.
- **Energy Efficiency Ratio (EER):** Evaluate the energy efficiency ratio of the AC system, which is the ratio of cooling capacity to power consumption. Higher EER values indicate more energy-efficient systems.

d. Thermostat System

The thermostat system plays a crucial role in maintaining the room's temperature at the desired setpoint. Consider the following elements:

- **Temperature Setpoint:** Determine the desired temperature setpoint based on occupant comfort and energy efficiency goals.
- **Hysteresis:** Incorporate hysteresis in the thermostat control logic to prevent frequent switching of the AC system, which can save energy.
- **Feedback Control:** Use feedback control to continuously monitor the room's temperature and adjust the AC system's operation to maintain the setpoint.
- **Occupancy Detection:** If available, consider integrating the occupancy detection information from the computer vision system to optimize the thermostat control strategy. For example, the AC system might reduce cooling when a room is unoccupied.
- **Adaptive Control:** Explore the possibility of adaptive control strategies that learn from past behavior and adjust the cooling system's operation accordingly.

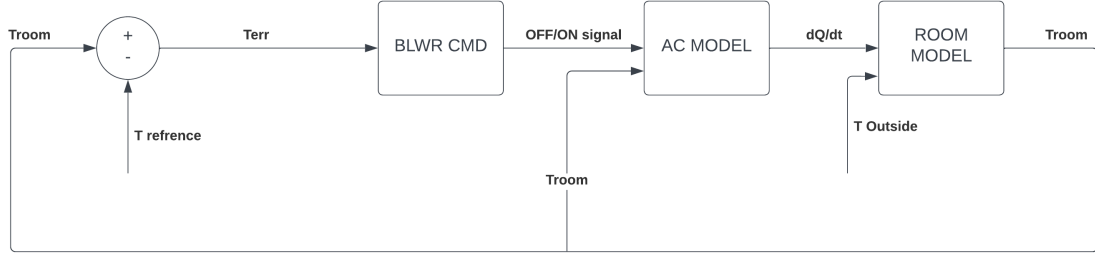


Figure B.1 Room Model Block Diagram

B. Experiment Model Components/Blocks

In this section, we delve into the comprehensive internal models that significantly influence the thermal profile of the room environment. These internal models play a crucial role in determining the overall temperature and energy dynamics within the room, considering various structural specifications and occupancy patterns.

a. Thermostat Block

The thermostat system plays a crucial role in maintaining the room's temperature at the desired setpoint. It acts as a control mechanism for the AC system. The thermostat continuously monitors the room's temperature and sends control signals to the AC system to adjust its operation based on the difference between the actual temperature and the setpoint. The equation of the thermostat block is as follows:

$$blower\ cmd = \{0, \quad Terr < -0.1, \quad Terr > 0.1 \quad (5)$$

Where **Terr** is the difference between the set point of the thermostat and the actual room temperature.

b. AC block

The cooling load refers to the amount of heat that needs to be removed from the room to maintain the desired temperature or setpoint. The cooling load is influenced by factors like external environmental conditions (such as outdoor temperature and solar radiation), internal heat sources (equipment, lighting, occupants), and the room's thermal characteristics.

The AC model represents the air conditioning system responsible for cooling the room. It could be a central air conditioning system or individual air conditioners depending on the building's configuration. The AC model should take into account the cooling capacity, energy efficiency, and control mechanisms of the cooling system.

As shown in the block diagram Above this block is responsible for simulating the AC behavior based on the following equation:

$$\frac{dQ}{dt} = [(T_{err} - 14) * \dot{M} * c] * blower\ cmd \quad (6)$$

Where **Mdot** is the Air flow rate $\dot{M} = 1 \text{ kg/sec} = 3600 \text{ kg/hr}$, **c** is the cp of the air = 1006 J/kg-K, **dQ/dt** is the amount of energy supplied by the AC.

c. Room model block

This model is responsible for simulating the room environment based on the following equation:

$$T_{in} = \int_0^t \left[(T_{out} - T_{in}) * \frac{1}{R_{eq}} - \frac{dQ}{dt} \right] * \frac{1}{M * c} \quad (7)$$

$$M = lenRoom * widRoom * htRoom * densAir$$

$$R_{eq} = (R_{wall} * R_{window}) / (R_{wall} + R_{window})$$

Where **Tout** is the outside temperature, **Tin** is the inside temperature, **Req** is the equivalent resistance of the wall and windows.

C. Calculation and Comparison

A series of experiments on the environmental setup is conducted, varying the predefined AC setpoint temperature from 20 to 40 degrees. Each experiment had a fixed time duration of 1 hour. The primary objective was to calculate the energy consumption for each setpoint and analyze the energy savings compared to the power consumption at the baseline setpoint of 20 degrees. The results of these experiments are summarized in Table 5.5 below. The energy consumption patterns for each setpoint and compared them to the energy consumed when the AC was set at 20 degrees is carefully observed. The impact of changing the AC setpoint on overall energy usage is being assessed. By systematically varying the setpoint temperature, valuable insights into the relationship between setpoint temperature and energy efficiency has been obtained. Table 5.5 presents the findings of these experiments, displaying the total power consumption in watt-hours (wh) for each setpoint temperature, as well as the percentage of energy savings compared to the power consumption at 20

degrees. The results reveal a clear trend: as the AC setpoint temperature increases, the total power consumption decreases. Higher setpoint temperatures require less cooling effort, leading to reduced energy consumption. Notably, as the setpoint temperature rises from 20 to 40 degrees, the energy savings percentage steadily increases. At a suitable setpoint of 25 degrees, the energy savings reach an impressive 50% compared to the baseline consumption at 20 degrees.

The power consumption Model is calculated based on the following equations:

$$\begin{aligned} power = & (blower\ cmd' \& Sw) * 100 \\ & + (blower\ cmd \& Sw) * 1500 \end{aligned} \quad (4)$$

Where **Sw** is the on/off switch of the AC, **Blower cmd** is the on/off thermostat signal.

This data indicates that setting the AC setpoint temperature strategically can significantly impact energy efficiency and reduce overall energy consumption. By selecting higher setpoint temperatures when possible, building occupants can enjoy comfortable conditions while promoting energy conservation and lowering operational costs. These findings emphasize the importance of informed decision-making when controlling HVAC systems in commercial buildings to achieve sustainable and environmentally responsible practices. The experimental approach and comprehensive analysis used in this study provide valuable insights for building managers and policymakers seeking to optimize energy usage and promote eco-friendly practices. Furthermore, the results underline the potential benefits of adopting intelligent and adaptive HVAC control systems that dynamically adjust setpoint temperatures based on occupancy patterns and environmental conditions. Implementing

such approaches can lead to substantial energy savings and contribute to a greener and more sustainable future for commercial buildings.

Table 5.5 Shows the Total Power Consumption in kWh for Each Scenario for Different Ac Setpoint Ranging from 20° to 25° In Case 1.

Setpoint	Power Consumption wh	Saving
20	1002.10	0.000%
21	842.22	15.954%
22	722.34	27.917%
23	629.12	37.220%
24	554.55	44.661%
25	493.56	50.748%
26	442.71	55.822%
27	399.69	60.114%
28	362.82	63.794%
29	330.87	66.982%
30	302.91	69.772%
31	278.24	72.234%
32	256.32	74.422%
33	236.70	76.380%
34	219.04	78.142%
35	203.06	79.736%
36	188.54	81.186%
37	175.28	82.509%
38	163.12	83.722%
39	151.94	84.838%
40	141.62	85.868%
41	132.06	86.822%

42	123.18	87.708%
43	114.91	88.533%
44	107.19	89.304%
45	100.00	90.021%

5.3 Summery

This chapter of the thesis focuses on the implementation of environmental thermal models and simulations for two distinct settings: a hotel building (Environment 1) and a specific room environment (Environment 2). The goal is to analyze the power consumption of the building and develop strategies to enhance energy efficiency by using computer vision techniques to control electrical appliances based on occupancy status.

In the first section, the power system analysis of the hotel building is presented. Detailed properties and base assumptions are identified to develop an accurate modeling method. The building's size, orientation, construction materials, and occupancy levels are considered in the simulations. A comprehensive approach is used to calculate cooling loads, which include both external (from outdoor sources) and internal (from occupants and appliances) cooling loads. The total cooling load is calculated for the entire building, and the power consumption of the AC system is determined for different AC setpoint temperatures. The analysis reveals that a one-degree increase in the setpoint temperature can save up to 4% of total power consumption in an empty room.

Next, Case Studies are conducted for both Environment 1 and Environment 2. In Case 1, thirty stochastic scenarios are considered for each room, where the occupancy status is represented by 18-hour time slots. The maximum and minimum hours allowed for the room

to be empty are variable and depend on the time of day. In Case 2, the constraints for the maximum and minimum allowed time for the room to be empty are fixed and uniform across all scenarios and periods. The results show that the proposed approach can save up to 4% of energy for each HVAC setpoint from 20°C, which is the most commonly selected setpoint by users.

The second section of the chapter focuses on Environment 2, where a room environment is simulated using MATLAB Simulink. The room's thermal profile is influenced by various structural specifications, such as dimensions, design, materials used, and the number of windows and doors. Three main model components are discussed: the thermostat block, the AC block, and the room model block. These components simulate the behavior of the thermostat system, the AC system, and the room environment, respectively.

The simulations are conducted for different AC setpoint temperatures ranging from 20°C to 45°C. The energy consumption for each setpoint is calculated, and the results show that as the setpoint increases, the power consumption decreases, with up to 15% energy savings for each setpoint change compared to the baseline setpoint at 20°C.

In conclusion, Chapter 5 provides valuable insights into utilizing computer vision in commercial buildings for energy-efficient control of electrical appliances based on occupancy status. The power system analysis and case studies in both Environment 1 and Environment 2 demonstrate the potential for significant energy savings by implementing smart control strategies. The research findings contribute to the promotion of energy efficiency in commercial buildings and offer practical applications for reducing energy consumption and its associated environmental impact.

CHAPTER 6

CONCLUSION

6.1 Conclusion

This thesis introduces a novel approach utilizing computer vision and image processing techniques to address the critical issue of building occupancy status, which plays a pivotal role in demand-side management schemes and directly influences building energy consumption. The methodology leverages existing CCTV camera systems installed in each building to process and analyze visual feeds through advanced computer vision techniques, effectively counting the number of occupants in each room or area to optimize energy consumption. Additionally, the thesis investigates the impact of occupancy status on energy usage by developing various environmental thermal models, subjecting them to diverse scenarios and constraints for comprehensive analysis.

The implementation of the proposed methodology employs industry-standard frameworks: Tensorflow for computer object detection and tracking, scikit-learn for sophisticated data analysis, and OpenCV for image processing and mathematical computations. The resulting system is designed as a retrofitting solution, capable of seamless integration into any building, regardless of its original purpose or layout.

The findings of this research demonstrate the effectiveness of the computer vision methodologies in real-time occupancy profiling, leading to significant reductions in building energy consumption. Furthermore, the versatility of the methodology allows its

integration with different Demand Side Management (DSM), Demand Response (DR), or Energy Efficiency (EE) programs to provide accurate building occupancy status without relying on uncertain models. Instead, it relies on direct visual information from the environment, resulting in improved energy management programs. So, what have been propose in this thesis can be summarized to the following:

- **Building Occupancy Status Problem:** The thesis identified the building occupancy status as a crucial factor in demand-side management schemes and its direct impact on building energy consumption. It aimed to address this problem using computer vision and image processing techniques.
- **Existing CCTV Camera Utilization:** The methodology utilized existing CCTV camera systems in buildings to capture visual feeds, and advanced computer vision techniques were applied to analyze these feeds. The goal was to count the number of people in each room or place within the building for more efficient energy consumption.
- **Environmental Thermal Models:** The thesis developed different environmental thermal models to study the effect of occupancy status on energy consumption. These models were tested under various scenarios and constraints to understand their performance.
- **Implementation in Frameworks:** The proposed methodology was implemented using various software frameworks. Tensorflow was used for computer object detection and tracking, scikit-learn for complicated data analysis, and OpenCV for image processing and mathematical computations.

- **Retrofitting System:** The developed system was designed as a retrofitting solution, making it compatible with any type of building, regardless of its original purpose or design.

While this thesis has made significant strides in utilizing computer vision techniques to improve building energy management through occupancy status analysis, there are several potential areas for future research and development to further enhance the system's effectiveness and versatility:

- 1- **Multi-Camera Integration:** Currently, the proposed methodology focuses on analyzing visual feeds from existing CCTV camera systems within each building. Future work could explore the integration of multiple camera feeds from different angles and locations to improve accuracy and coverage in large or complex buildings. This integration could involve advanced multi-camera calibration and fusion techniques to create a more comprehensive occupancy profile.
- 2- **Real-Time Energy Management System:** Building upon the real-time occupancy profile results obtained in this thesis, future work can focus on developing an integrated real-time energy management system. This system can dynamically adjust HVAC and lighting systems based on the detected occupancy status, optimizing energy consumption in response to changing occupancy patterns throughout the day.
- 3- **Enhanced Accuracy with Deep Learning Models:** While the current implementation utilizes Tensorflow for object detection and tracking, further advancements in deep learning models could be explored to achieve even higher accuracy in occupancy detection. Novel architectures, transfer learning, and more extensive training datasets could be investigated to enhance the model's performance.

- 4- **Energy Feedback System:** Future work could incorporate an energy feedback system that communicates real-time energy consumption data to building occupants. This feedback mechanism can create awareness among occupants about their energy usage patterns, encouraging energy-saving behaviors and promoting a culture of sustainability within the building.
- 5- **Integration with Smart Building Systems:** To achieve seamless and automated energy management, future work could focus on integrating the computer vision-based occupancy detection system with smart building technologies. Integration with Building Automation Systems (BAS) and Internet of Things (IoT) devices can facilitate real-time control and optimization of various building systems based on occupancy information.

Furthermore, exploring the feasibility of applying the computer vision-based occupancy detection system in different climatic regions and building designs would provide valuable insights into its applicability under diverse environmental conditions.

Finally, collaborating with building owners, managers, and energy utilities for pilot implementations and gathering feedback on user experience and system performance would aid in refining the methodologies and addressing any practical challenges that may arise during real-world deployments.

APPENDIX

Codes

```
import tensorflow as tf
import cv2
import numpy as np
import pandas as pd
import datetime
import time
import ipdb

from utils import define_rect
from deep_sort_ball_and_person.deep_sort.tracker import Tracker
from deep_sort_ball_and_person.deep_sort import nn_matching
from deep_sort_ball_and_person.deep_sort.detection import Detection
from deep_sort_ball_and_person import generate_detections
from deep_sort_ball_and_person.application_util import preprocessing as prep

def IOU(door_box, person_box):
    x1 = np.maximum(door_box[0], person_box[0])
    y1 = np.maximum(door_box[1], person_box[1])
    x11 = np.minimum(door_box[2], person_box[2])
    y11 = np.minimum(door_box[3], person_box[3])
    area_iou = (x11-x1)*(y11-y1)
    #area_door = (door_box[2]-door_box[0])*(door_box[3]-door_box[1])
    area_person = (person_box[2]-person_box[0])*(person_box[3]-person_box[1])
    iou = area_iou/(area_person)
    return iou

def area(door_box, person_box):

    area_door = (door_box[2]-door_box[0])*(door_box[3]-door_box[1])
    area_person = (person_box[2]-person_box[0])*(person_box[3]-person_box[1])
    if 1.5*area_door > area_person and area_door<9*area_person:
        return True
    return False

class Detector:
    def __init__(self, arghandler):

        tf.reset_default_graph()

        self.nms_max_overlap = 0.45
        self.image_out = []
```

```

self.FLAGS = arghandler

self.det_graph = self.get_detections_graph(self.FLAGS.model)
self.encoder = generate_detections.create_box_encoder(self.FLAGS.track_model,
batch_size = 1) # continue
self.metric = nn_matching.NearestNeighborDistanceMetric("cosine", 0.35,350) #
continue matching threshold, and nbudget.
self.tracker = Tracker(self.metric, max_iou_distance=0.75, max_age=100, n_init=4,
_std_weight_position = 2. / 20, _std_weight_velocity=1 /160)

#counter variables
self.door_box = self.FLAGS.door_box
self.people_in_cor = []
self.people_out_cor = []
self.count_in = []
self.count_out = []
self.potential_out = {}
self.door_counter = 2

self.ThemoAction = 0

self.out_of_bounds = [] # to help with the counting in problem (in, out,in) if the track
id went hout of the scene (not new , and not out to the room).

#switch problem variables
self.potential_switch_track = []

def get_detections_graph(self,path):

    detection_graph = tf.Graph()
    with detection_graph.as_default():
        od_graph_def = tf.GraphDef()
        with tf.gfile.GFile(path, 'rb') as fid:
            serialized_graph = fid.read()
            od_graph_def.ParseFromString(serialized_graph)
            tf.import_graph_def(od_graph_def, name='')
        return detection_graph
def detect_box(self,sess,img, class_id =1):
    h,w,_ = img.shape
    img_exp = np.expand_dims(img, axis = 0)
    image_tensor = self.det_graph.get_tensor_by_name('image_tensor:0')
    boxes = self.det_graph.get_tensor_by_name('detection_boxes:0')
    scores = self.det_graph.get_tensor_by_name('detection_scores:0')
    classes = self.det_graph.get_tensor_by_name('detection_classes:0')
    num_detections = self.det_graph.get_tensor_by_name('num_detections:0')
    (boxes, scores, classes, num_detections) = sess.run([boxes, scores, classes,
num_detections],
feed_dict={image_tensor:img_exp})

    filtered_boxes = []
    filtered_scores= []
    filtered_classes = []
    for i in range(len(boxes[0][:])): # we can use matrix instead of for loops

```

```

        if classes[0][i] == class_id: # or I can say == FLAGS.tr_Obj / or we can read
these id or interpret them from outside
            if scores[0][i] >= self.FLAGS.threshold:
                ymin = boxes[0][i][0]*h
                xmin = boxes[0][i][1]*w
                width = (boxes[0][i][3]-boxes[0][i][1])*w
                height = (boxes[0][i][2]-boxes[0][i][0])*h

                filtered_boxes.append([xmin, ymin, width,height])
                filtered_scores.append(scores[0][i])
                filtered_classes.append(classes[0][i])
            return filtered_boxes, filtered_scores, filtered_classes
def track_box(self,img, boxes, scores, classes):
    detection_dict = {}
    score_dict = {}
    detection_dict['person'] = []
    score_dict['person'] = []
    for i in range(len(boxes)):
        box = boxes[i]
        detection_dict['person'].append(np.array([box[0], box[1], box[2],
box[3]]).astype(np.float64))
        score_dict['person'].append(scores[i])

    detection_dict['person'] = np.array(detection_dict['person'])
    scores_a = np.array(score_dict['person'])
    features = self.encoder(img, detection_dict['person'])
    detections = [Detection(box,score,feature) for box,score,feature in
zip(detection_dict['person'],
        scores_a,features)]
    boxes_d = np.array([d.tlwh for d in detections])
    scores_d = np.array([d.confidence for d in detections])
    indices = prep.non_max_suppression(boxes_d, self.nms_max_overlap, scores_d)
    detections = [detections[i] for i in indices]
    self.tracker.predict()
    self.tracker.update(detections)
    tracks = self.tracker.tracks
    return tracks
#####
##### HELPER FUNCTION
def saveVideo(self):
    fourcc = cv2.VideoWriter_fourcc(*'XVID')
    out = cv2.VideoWriter(self.FLAGS.save_path, fourcc, 25.0,
(self.image_out[0].shape[1], self.image_out[0].shape[0]))
    for img in self.image_out:
        out.write(img)
    out.release()

def clean_counters(self,tracks):
    for track in tracks:
        if track.track_id in self.people_out_cor and track.time_since_update >= 70:
            self.people_out_cor.remove(track.track_id)
            self.potential_out.pop(track.track_id, None)

```

```

        if track.track_id in self.people_in_cor and track.age >= 70 and
        IOU(self.door_box, track.to_tlbr())<0.15: # above certain threshold cause of not counting the same id
        twice.
            self.people_in_cor.remove(track.track_id)

    def lef_side_box_check(self, track, door_box):
        if track.to_tlbr()[0] < door_box[0] + 5 :
            return True
        return False

    def lef_side_box_check_out(self, track, door_box):
        if track.to_tlbr()[0] < door_box[0]+20:
            return True
        return False

    def right_side_box_check(self, track, door_box):
        if track.to_tlbr()[0]+5 > door_box[0]:
            return True
        return False

    def camera(self):
        df = pd.DataFrame(columns = ["date", "time", "Room Occupancy Status"])
        cap = cv2.VideoCapture(self.FLAGS.demo)
        ret,img = cap.read()
        self.door_box = define_rect(img)
        gpu_options = tf.GPUOptions(per_process_gpu_memory_fraction=self.FLAGS.gpu)
        with self.det_graph.as_default():
            with tf.Session(graph = self.det_graph, config = tf.ConfigProto(gpu_options
= gpu_options)) as sess:
                start = time.time()
                prev_t = int(start)
                while ret:
                    ret, img = cap.read()
                    if img is None:
                        continue
                    frame = img
                    boxes, scores, classes = self.detect_box(sess,frame)
                    tracks = self.track_box(frame, boxes, scores,classes)

                    for track in tracks:
                        if (track.track_id in self.out_of_bounds and
track.time_since_update == 0):
                            self.out_of_bounds.remove(track.track_id)

                        # count the people out of the room
                        if track.is_confirmed() and IOU(self.door_box,
track.to_tlbr()) > 0.55 and self.lef_side_box_check_out(track, self.door_box) and
area(self.door_box,track.to_tlbr()) and track.track_id not in self.people_in_cor:
                            if track.age < 10 and track.age > 5 :

```

```

        self.people_in_cor.append(track.track_id)
        self.door_counter =
self.door_counter - 1

        elif (track.track_id in self.people_out_cor
or track.track_id in self.out_of_bounds) and track.time_since_update == 0:

        self.people_in_cor.append(track.track_id)
        self.door_counter =
self.door_counter -1
        self.people_out_cor:
        self.people_out_cor.remove(track.track_id)
        self.out_of_bounds:
        self.out_of_bounds.remove(track.track_id)

        # count the people in the room
        if track.time_since_update > 13 and
track.time_since_update < 15 and IOU(self.door_box, track.to_tlbr())> 0.2 and area(self.door_box,
track.to_tlbr()):# I should be adding hte IOU of the track # this condition has two meanings 1- that the
target has exited the coridor 2- the target has exited the scene 3- the target id has switched to tother id
        if track.track_id not in self.potential_out:
            self.potential_out[track.track_id]
= track.to_tlbr()
        if track.is_confirmed() and
IOU(self.door_box, self.potential_out[track.track_id]) > 0.2 and area(self.door_box,
self.potential_out[track.track_id]) and self.lef_side_box_check(track,self.door_box) and
track.time_since_update > 13 and track.time_since_update < 15 and track.age > 40 and track.hits > 40
and track.track_id not in self.people_out_cor:

        self.people_out_cor.append(track.track_id)
        self.door_counter =
self.door_counter + 1
        self.people_in_cor:
        self.people_in_cor.remove(track.track_id)

        # check the if the track has passed the boundary
        if (track.to_tlbr()[0] < 5 or track.to_tlbr()[2] >
(frame[0].shape[0]-7)) and track.time_since_update > 13:
            self.out_of_bounds.append(track.track_id)

        self.clean_counters(tracks)

        for track in tracks:
            if not track.is_confirmed or
track.time_since_update > 1:

```

```

        continue
        bbox = track.to_tlbr()
        cv2.rectangle(frame, (int(bbox[0]), int(bbox[1])),
(int(bbox[2]), int(bbox[3])),(255,255,255), 3)
        cv2.putText(frame, str(track.track_id),
(int(bbox[0]), int(bbox[1])), 0, 3e-3 * 200, (0, 255, 0), 2)
        cv2.rectangle(frame, (int(self.door_box[0]),
int(self.door_box[1])), (int(self.door_box[2]), int(self.door_box[3])),(128,75,0), 3)
        cv2.putText(frame, "the number of people in the room " +
str(self.door_counter), (int(20.), int(20.)), 0, 3e-3 * 200, (0, 255, 255), 2)
        self.image_out.append(frame)

        t = (int(time.time()) - int(start))
        if t != prev_t and t%10 == 0:
            now = datetime.datetime.now()
            dt_string = now.strftime("%m/%d/%Y
%H:%M:%S")

            if self.door_counter == 0:
                ta_prev = self.ThemoAction
                self.ThemoAction = 1
                if ta_prev == self.ThemoAction:
                    df =
df.append({"date":dt_string.split()[0],"time":dt_string.split()[1],"Room Occupancy Status": 0,
"Thermostat Action": 1}, ignore_index = True)
            else:
                df =
df.append({"date":dt_string.split()[0],"time":dt_string.split()[1],"Room Occupancy Status": 0,
"Thermostat Action": 0}, ignore_index = True)

            else:
                df =
df.append({"date":dt_string.split()[0],"time":dt_string.split()[1],"Room Occupancy Status": 1,
"Thermostat Action": 0}, ignore_index = True)

                ta_prev = self.ThemoAction
                self.ThemoAction = 0
                prev_t = t

        cv2.imshow("frame")
        choice = cv2.waitKey(1)
        if choice == 27:
            break
    cv2.destroyAllWindows()
    df.to_csv('r_new1.csv')
    if self.FLAGS.saveVideo:
        self.saveVideo()

```

References

- [1] P. Warren, " A review of demand-side management policy in the UK," *Renewable and Sustainable Energy Reviews*, vol. 29, pp. 941-951, 2014.
- [2] E. A. Abdelaziz, R. Saidur and S. Mekhilef, "A review on energy saving strategies in industrial sector," *Renewable and Sustainable Energy Reviews*, vol. 15, pp. 150-168, 2011.
- [3] S. Salimi and A. Hammad, "Critical review and research roadmap of office building energy management based on occupancy monitoring," *Energy & Buildings*, vol. 182, pp. 214-241, 2019.
- [4] Gneral Authority for Statistics Electricity energy Statistics- 2018. Available online: https://www.stats.gov.sa/sites/default/files/hst_ltq_lkhrbyy_-_rby_0.pdf (accessed on 10 September 2020).
- [5] Saudi Energy Efficiency Center Annual Report-2018. Available online: https://www.seec.gov.sa/sites/default/files/blog_files/annual-report-2018.pdf (accessed on 12 September 2020)
- [6] A. C. Chapman, G. Verbic and D. J. Hill, "Algorithmic and strategic aspect to integrating demand-side aggregation and energy management methods," *IEEE Transaction on Smart Grid*, vol. 7, no. 6, pp. 2748-2760, 2016.
- [7] P. Palensky and D. Dietrich, "Demand side management: demand response intelligent energy systems, and smart loads," *IEEE Trasaction on industrial informatics*, vol. 7, no. 3, pp. 381-388, 2011.
- [8] T. Malavazi, S. Perron, J. F. Fardin, D. Simonetti and C. Alvarez, "Demand-side energy management by cooperative combination of plans: A multi-objective method applicable to isolated communities," *Applied Energy*, vol. 240, pp. 453-472, 2019.
- [9] T. Kurniawan, M. Vasirani and K. Aberer, "When bias matters: an economic assessments of demand response baselines for residential customers," *IEEE transaction on smart grid*, vol. 5, no. 4, pp. 1755-1763, 2014.
- [10] A. Sheikhi, M. Rayati, S. Bahrami and A. Ranbar, "Integrated demand side management game in smart energy hubs," *IEEE transaction on smart grid*, vol. 6, no. 2, pp. 675-683, March 2015.
- [11] W. Li, N. Hassan, C. Wen and X. Liu, "Demand response management for residential smart grid: from theory to practice," *IEEE Access*, vol. 3, pp. 2431-2440, 2015.
- [12] A. Mnatsakanyan and S. W. Kennedy, "A Novel Demand Response Model with an Application for a Virtual Power Plant," *IEEE Transactions on Smart Grid*, vol. 6, no. 1, pp. 230-237, Jan 2015.

- [13] F. Luo, W. Kong, G. Ranzi and Z. Y. Dong, "Optimal Home Energy Management System With Demand Charge Tariff and Appliance Operational Dependencies," in *IEEE Transactions on Smart Grid*, vol. 11, no. 1, pp. 4-14, Jan. 2020, doi: 10.1109/TSG.2019.2915679.
- [14] F. Alfaverh, M. Denaï and Y. Sun, "Demand Response Strategy Based on Reinforcement Learning and Fuzzy Reasoning for Home Energy Management," in *IEEE Access*, vol. 8, pp. 39310-39321, 2020, doi: 10.1109/ACCESS.2020.2974286.
- [15] A. Sheikhi, M. Rayati, S. Bahrami, A. Ranbar, "Integrated demand side management game in smart energy hubs," *IEEE transaction on smart grid*, vol. 6, no.2, pp. 1-1, March 2015
- [16] G. Tsagrakis, C. Thomson, A. J. Collin, G. Harrison, A. Kiprakis. S. McLaughlin, "assessment of the cost and environmental impact of residential demand-side management," *IEEE transaction on industry application*, vol. 52, no. 3, May 2016
- [17] C. Eksin, H. Delic, A. Ribeiro, "Demand response management in smart grids with heterogeneous consumer preferences," *IEEE transaction on smart grid*, vol. 6, no. 6, November 2015.
- [18] F. Wang et al., "Multi-Objective Optimization Model of Source–Load–Storage Synergetic Dispatch for a Building Energy Management System Based on TOU Price Demand Response," in *IEEE Transactions on Industry Applications*, vol. 54, no. 2, pp. 1017-1028, March-April 2018, doi: 10.1109/TIA.2017.2781639.
- [19] M. R. Sarker, M. A. Ortega-Vazquez and D. S. Kirschen, "Optimal Coordination and Scheduling of Demand Response via Monetary Incentives," in *IEEE Transactions on Smart Grid*, vol. 6, no. 3, pp. 1341-1352, May 2015, doi: 10.1109/TSG.2014.2375067.
- [20] V. Pilloni, A. Floris, A. Meloni and L. Atzori, "Smart Home Energy Management Including Renewable Sources: A QoE-Driven Approach," *IEEE Transactions on Smart Grid* vol. 9, no. 3, pp. 2006-2018, May 2018., vol. 9, no. 3, pp. 2006-2018, May 2018.
- [21] J. Jin, Y. Xu, Y. Khalid and N. U. Hassan, "Optimal Operation of Energy Storage With Random Renewable Generation and AC/DC Loads," *IEEE Transactions on Smart Grid*, vol. 9, no. 3, pp. 2314-2326, May 2018.
- [22] B. L. e. al, "An AC–DC Hybrid Multi-Port Energy Router With Coordinated Control and Energy Management Strategies," *IEEE Access*, vol. 7, pp. 109069-109082, 2019.
- [23] Y. Lin and M. Tsai, "An Advanced Home Energy Management System Facilitated by Nonintrusive Load Monitoring With Automated Multiobjective Power Scheduling," *IEEE Transactions on Smart Grid*, vol. 6, no. 4, pp. 1839-1851, July 2015.

- [24] A. Pellegrino, V. R. M. L. Verso, L. Blaso, A. Acquaviva, E. Patti and A. Osello, "Lighting Control and Monitoring for Energy Efficiency: A Case Study Focused on the Interoperability of Building Management Systems," *IEEE Transactions on Industry Applications*, vol. 52, no. 3, pp. 2627-2637, May-June 2016.
- [25] M. M. Solana, M. Ros, M. D. Ruiz, J. G. Romero and M. J. Bautista, "Data science for building energy management: A review," *Renewable and Sustainable Energy Reviews*, vol. 70, pp. 598-609, April 2017.
- [26] S. Seyedzadeh, F. Rahimian, I. Glesk and e. al, "Machine learning for estimation of building energy consumption and performance: a review," *Visualization in Engineering* , vol. 6, no. 5, 2018.
- [27] A. Pallante, et al. "Decision support methodologies and day-ahead optimization for smart building energy management in a dynamic pricing scenario," *Energy and Buildings*, vol. 216, pp. 109963, 2020.
- [28] V. Pilloni, A. Floris, A. Meloni and L. Atzori, "Smart Home Energy Management Including Renewable Sources: A QoE-Driven Approach," in *IEEE Transactions on Smart Grid*, vol. 9, no. 3, pp. 2006-2018, May 2018, doi: 10.1109/TSG.2016.2605182.
- [29] A. Parisio, C. Wiezorek, T. Kytäjä, J. Elo, K. Strunz and K. H. Johansson, "Cooperative MPC-Based Energy Management for Networked Microgrids," in *IEEE Transactions on Smart Grid*, vol. 8, no. 6, pp. 3066-3074, Nov. 2017, doi: 10.1109/TSG.2017.2726941.
- [30] T. de Christo, et al. "Demand-side energy management by cooperative combination of plans: A multi-objective method applicable to isolated communities," *Applied Energy*, vol. 240, pp. 453-472, April 2019.
- [31] A. Pellegrino, V. R. M. Lo Verso, L. Blaso, A. Acquaviva, E. Patti and A. Osello, "Lighting Control and Monitoring for Energy Efficiency: A Case Study Focused on the Interoperability of Building Management Systems," in *IEEE Transactions on Industry Applications*, vol. 52, no. 3, pp. 2627-2637, May-June 2016, doi: 10.1109/TIA.2016.2526969.
- [32] J. Pan, R. Jain, S. Paul, T. Vu, A. Saifullah and M. Sha, "An Internet of Things Framework for Smart Energy in Buildings: Designs, Prototype, and Experiments," *IEEE Internet of Things Journal*, vol. 2, no. 6, pp. 527-537, Dec 2015.
- [33] R. J. Tom, S. Sankaranarayanan and J. J. P. C. Rodrigues, "Smart Energy Management and Demand Reduction by Consumers and Utilities in an IoT-Fog-Based Power Distribution System," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 7386-7394, Oct 2019.
- [34] M. H. Y. Moghaddam and A. Leon-Garcia, "A fog-based internet of energy architecture for transactive energy management systems," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1055-1069, April 2018.

- [35] B. Hussain, Q. U. Hasan, N. Javaid, M. Guizani, A. Almogren and A. Alamri, "An Innovative Heuristic Algorithm for IoT-Enabled Smart Homes for Developing Countries," in *IEEE Access*, vol. 6, pp. 15550-15575, 2018, doi: 10.1109/ACCESS.2018.2809778.
- [36] T. Chiu, Y. Shih, A. Pang and C. Pai, "Optimized Day-Ahead Pricing With Renewable Energy Demand-Side Management for Smart Grids," in *IEEE Internet of Things Journal*, vol. 4, no. 2, pp. 374-383, April 2017, doi: 10.1109/JIOT.2016.2556006.
- [37] S. Kazmi, N. Javaid, M. J. Mughal, M. Akbar, S. H. Ahmed and N. Alrajeh, "Towards Optimization of Metaheuristic Algorithms for IoT Enabled Smart Homes Targeting Balanced Demand and Supply of Energy," in *IEEE Access*, vol. 7, pp. 24267-24281, 2019, doi: 10.1109/ACCESS.2017.2763624.
- [38] T. Labeodan, W. Zeiler, G. Boxem and Y. Zhao, "Occupancy measurement in commercial office buildings for demand-driven control applications—A survey and detection system evaluation," *Energy and Buildings*, vol. 93, pp. 303-314, April 2015.
- [39] T. Labeodan, K. Aduda, W. Zeiler and F. Hoving, "Experimental evaluation of the performance of chair sensors in an office space for occupancy detection and occupancy-driven control," *Energy and Buildings*, vol. 111, pp. 195-206, Jan 2016.
- [40] Z. Zhen, Q. Jia, C. Song and X. Guan, "An Indoor Localization Algorithm for Lighting Control using RFID," in *2008 IEEE Energy 2030 Conference*, Atlanta, GA, 2008.
- [41] N. Li, G. Calis and B. B. Gerber, "Measuring and monitoring occupancy with an RFID based system for demand-driven HVAC operations," *Automation in Construction*, vol. 24, pp. 89-99, July 2012.
- [42] Y. Want and L. Shao, "Understanding occupancy pattern and improving building energy efficiency through Wi-Fi based indoor positioning," *Building and Environment*, vol. 114, pp. 106-117, March 2017.
- [43] M. B. Afuosi, and M. R. Zoghi, "Indoor positioning based on improved weighted KNN for energy management in smart buildings," *Energy and Buildings*, vol. 212, pp. 109754, 2020.
- [44] R. Goldstein, A. Tessier and A. Khan, "Space layout in occupant behavior simulation," in *Conference Proceedings: IBPSA-AIRAH Building Simulation Conference*, Sydney, 2011.
- [45] C. Harris and V. Cahill, "Exploiting user behaviour for context-aware power management," in *IEEE International Conference on Wireless And Mobile Computing, Networking And Communications*, Montreal, 2005.
- [46] J. Page, D. Robinson, N. Morel and L. Scartezzini, "A generalised stochastic model for the simulation of occupant presence," *Energy and Buildings*, vol. 40, no. 2, pp. 83-98, 2008.

- [47] J. Dobbs and B. M. Henecy, "Model predictive HVAC control with online occupancy model," *Energy and Buildings*, vol. 82, pp. 675-684, Oct 2014.
- [48] R. J. M. Jin and C. J. Spanos, "Virtual Occupancy Sensing: Using Smart Meters to Indicate Your Presence," *IEEE Transactions on Mobile Computing*, vol. 16, no. 11, pp. 3264-3277, Nov 2017.
- [49] G. Newsham, H. Xue, C. Arsenault, J. Valdes, G. Burns, E. Scarlett, S. Kruithof and W. Shen, "Testing the accuracy of low-cost data streams for determining single-person office occupancy and their use for energy reduction of building services," *Energy and Buildings*, vol. 135, pp. 137-147, Jan 2017.
- [50] Howard, Andrew G., et al. "Mobilenets: Efficient convolutional neural networks for mobile vision applications." arXiv preprint arXiv:1704.04861 (2017).
- [51] Liu, Wei, et al. "Ssd: Single shot multibox detector." European conference on computer vision. Springer, Cham, 2016.
- [52] YOLO:Real-Time Object Detection, Available online: <https://pjreddie.com/darknet/yolo>, (accessed on 5 September 2020)
- [53] Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." Advances in neural information processing systems. 2015.
- [54] Y. LeCun, Y. Bengio and others, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, p. 1995, 1995.
- [55] A. Krizhevsky, L. Sutskever, G. E. Hinton, "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems vol. 25, pp. 1097-1105, 2012.
- [56] K. Simonyan, and A. Zisserman, "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).
- [57] S. Christian, et al. "Going deeper with convolutions." Proceedings of the IEEE conference on computer vision and pattern recognition, Boston, MA, USA, 2015, pp. 1-9
- [58] K. Patel, "MNIST Handwritten Digits Classification using a Convolutional Neural Network (CNN)," Towards Data Science, 7 9 2019. [Online]. Available: <https://towardsdatascience.com/mnist-handwritten-digits-classification-using-a-convolutional-neural-network-cnn-af5fafbc35e9>. [Accessed 24 3 2021].
- [59] N. Wojke, A. Bewley and D. Paulus, "Simple Online and Realtime Tracking with a Deep Association Metric," in 2017 IEEE International Conference on Image Processing (ICIP), Beijing, 2017.
- [60] M. Abadi, B. Paul, C. Jianmin, C. Zhifeng, D. Andy, D. Jeffrey, D. Matthieu, et al. "TensorFlow: A System for Large-Scale Machine Learning." In 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16). pp. 265-283, 2016.

- [61] F. Pedregosa, et al. "Scikit-learn: Machine learning in Python." *Journal of machine learning research*, pp.2825–2830, 2011.
- [62] G. Bradski, "The OpenCV Library." *Dr. Dobb's Journal of Software Tools*, 2000.
- [63] S. Shao, M. Pipattanasomporn and S. Rahman, "Development of physical-based demand response-enabled residential load models," *IEEE Transactions on Power Systems*, vol. 28, no. 2, pp. 607-614, May 2013.
- [64] M. A. Hassan, A. A. Eladl, B. E. Sedhom and M. A. Saeed, "A New Optimal Centralized Demand Side Management for a Campus Smart Microgrid," *2023 IEEE 32nd International Symposium on Industrial Electronics (ISIE)*, Helsinki, Finland, 2023, pp. 1-6.
- [65] C. P. Shirley, S. V. Evangelin Sonia, V. Sathya, N. Manikandan, M. K. Vidhyalakshmi and C. V. K. Reddy, "Blockchain and Deep Learning Development of Smart Charging of Electric Vehicles to Meet the Demand Side Management," *2023 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS)*, Erode, India, 2023, pp. 1377-138.
- [66] M. Manoharan and P. S, "An Optimized Demand Side Management with Automatic Peak Clipping and Load Scheduling for Energy Conservation," *2023 Second International Conference on Augmented Intelligence and Sustainable Systems (ICAISS)*, Trichy, India, 2023, pp. 1710-1715.
- [67] J. Liu, S. Zhang and J. Wang, "Demand-Side Management Strategy for Local Energy System Supporting Renewable Energy Sources Integration," *2023 6th International Conference on Energy, Electrical and Power Engineering (CEEPE)*, Guangzhou, China, 2023, pp. 1332-1337.
- [68] C. Xu et al., "Sample-based Event-triggered Control of Thermostatically Controlled Loads for Demand Side Management*," *2022 IEEE International Conference on Cyborg and Bionic Systems (CBS)*, Wuhan, China, 2023, pp. 413-418.
- [69] U. Pati and K. D. Mistry, "Design and Implementation of an Enhanced Demand-Side Management Solution For Distribution End Prosumers," *2023 IEEE IAS Global Conference on Renewable Energy and Hydrogen Technologies (GlobConHT)*, Male, Maldives, 2023, pp. 1-7.
- [70] C. Songsong, G. Taorong, G. Feixiang, G. Haiqin and C. Kun, "Research on Interactive Mode and Strategy of Adjustable Load Demand Response Blockchain," *2022 IEEE 6th Conference on Energy Internet and Energy System Integration (EI2)*, Chengdu, China, 2022, pp. 2029-2033.
- [71] K. Shi, S. Chen, D. Li, M. Feng and Q. Zhou, "Evaluation of the Machine Learning Schemes for Information Security of Demand Response," *2022 IEEE 6th Conference on Energy Internet and Energy System Integration (EI2)*, Chengdu, China, 2022, pp. 1472-1476.
- [72] C. Arun, R. Aswinraj, M. T. Bijoy, M. Nidheesh and R. Rohikaa Micky, "Day Ahead Demand Response Using Load Shifting Technique in Presence of Increased Renewable Penetration," *2022*

- IEEE 7th International conference for Convergence in Technology (I2CT), Mumbai, India, 2022, pp. 1-6.
- [73] C. Songsong, G. Taorong, L. Lei and W. Chaoliang, "A two-tier demand response scheduling strategy based on users' equipment level," 2022 IEEE 5th International Electrical and Energy Conference (CIEEC), Nangjing, China, 2022, pp. 665-670.
 - [74] S. Srivastava, A. Kanungo and T. Jain, "An Integrated Approach based on Supervised and Unsupervised ML algorithms for Efficient Design of Demand Response Programs," 2023 International Conference on Intelligent and Innovative Technologies in Computing, Electrical and Electronics (IITCEE), Bengaluru, India, 2023, pp. 324-329.
 - [75] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, Kauai, HI, USA, 2001, pp. I-I, doi: 10.1109/CVPR.2001.990517.
 - [76] G. Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.
 - [77] D. Comaniciu and P. Meer, "Mean shift: a robust approach toward feature space analysis," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 5, pp. 603-619, May 2002, doi: 10.1109/34.1000236.
 - [78] Kalman, Rudolph Emil "A New Approach to Linear Filtering and Prediction Problems", in Transactions of the ASME--Journal of Basic Engineering, Vol. 82, no. Series D, Pp. 35-45, 1960

Vitae

Name :Muhammad Sobih

Nationality :Egypt

Date of Birth :11/11/1992

Email :Muhammadsobih11@gmail.com

Address :6658, Katheer Ibn Amrou Streat, West Naseem Dist,
Riyadh, Saudi Arabia

Academic Background:

- 1- BS in Electrical Engineering from KFUPM
- 2- MSc in Electrical Engineering from KFUPM

Professional Experience

Product Manager: Dawafast, July 2021 – Present

- Collaborated with the management team to implement growth strategies, leading the acquisition of customers and suppliers for Dawafast.
- Formulated and executed the technical product strategy, ensuring alignment with market demands.
- Maintained a continuous loop of user and market feedback, iterating on the product accordingly.
- Conducted thorough market research and analysis to improve our products.
- Improved internal operations by automating processes, resulting in a 10x increase in order capacity.
-

Computer Vision Engineer: Tbaud, May 2020 – May 2021

- Developed and implemented system architecture to maintain high-performance standards in computer vision applications.
- Conducted testing and fine-tuning of various deep learning models to align with specific business cases and optimize overall performance.
- Led the execution of thorough testing and system commissioning procedures before deployment, guaranteeing a smooth and successful implementation.
- Collaborated with cross-functional teams to integrate computer vision solutions into broader technological ecosystems.
- Contributed to the ongoing research and development efforts, staying abreast of the latest advancements in computer vision technology.
- Provided technical support and expertise during the troubleshooting and optimization phases of computer vision projects.

Data Scientist -Part Time-: PROXEC, May 2021 – March 2022

- Organized the refinement of the data pipeline, overseeing its transformation from raw data to a state of readiness for utilization in various models, ensuring the extraction of meaningful insights.
- Spearheaded the development of diverse deep learning and machine learning models, playing a pivotal role in empowering the research team with valuable insights.
- Expanded the scope of data analytics by implementing advanced statistical techniques and data mining methodologies, enhancing the depth and accuracy of information extracted.
- Collaborated cross-functionally to integrate data-driven insights into decision-making processes, fostering a culture of informed and strategic decision-making.
- Contributed to the enhancement of data quality and integrity, implementing data cleansing and validation processes to ensure the reliability of analytical results.
- Assisted in building and maintaining a robust data infrastructure, ensuring the seamless flow and accessibility of data across the organization.

Electrical & Machine Learning Engineer: MYCORR, Sep 2018 – Dec 2020

- Computer Vision Projects:
 - o Led the technical team in product developments.
 - o Developed system architecture to uphold high-performance standards.
 - o Conducted client interviews for comprehensive requirement gathering.
 - o Executed testing and system commissioning before the deployment phase.
- Smart Home Systems:
 - o Created engineering design schematics and KNX system line diagrams.
 - o Facilitated collaboration among project stakeholders, contractors, and authoritative bodies to secure drawing endorsements.

- Supervised electrical contractors to ensure accurate execution of design plans.
 - Engineered and installed intelligent control panels for KNX automation devices and relay systems.
- Solar PV System Design Projects:
 - Ensured adherence to standards set by the Royal Commission in all engineering activities.
 - Regularly engaged with clients to align design proposals with their requirements and schedule.
 - Oversaw project finances to minimize expenses.
 - Supervised the integrity of engineering design submissions.