Computer Vision-Based Intelligent Classroom Systems for Efficient Power Management in Large Educational Institutions

by

Muhtadi Haque (180021227)
Sadman Mahmud Faisal (180021230)
Md. Tahsinul Islam (180021231)
Tareq Hasan Akash (180021240)

BACHELOR OF SCIENCE
IN
ELECTRICAL AND ELECTRONIC ENGINEERING

Department of Electrical and Electronic Engineering
Islamic University of Technology (IUT)
Board Bazar, Gazipur-1704, Bangladesh.
February 2021

CERTIFICATE OF APPROVAL

The thesis titled, **"Computer Vision-Based Intelligent Classroom Systems for Efficient Power Management in Large Educational Institutions**" is accepted as partial fulfillment of the requirement for the Degree of BACHELOR OF SCIENCE IN ELECTRICAL AND ELECTRONIC ENGINEERING of Islamic University of Technology (IUT).

Approved by:

_____

Prof. Dr. Khondokar Habibul Kabir
(Supervisor) Professor,
Electrical and Electronic Engineering Department,
Islamic University of Technology (IUT), Gazipur.

Declaration of Candidate

It is hereby declared that this thesis report is only submitted to the Electrical Engineering Department. Any part of it has not been submitted elsewhere for the award of any Degree or Diploma.

Muhtadi Haque
Student Id.: 180021227

Sadman Mahmud Faisal
Student Id.: 180021230

Md. Tahsinul Islam
Student Id.: 180021231

Tareq Hasan Akash
Student Id.: 180021240

Table of Contents

List of Figures

List of Tables

List of Abbreviations

MOG2	Mixture Of Gaussian 2
OPENCV	Open Source Computer Vision Library
GPIO	General Purpose Input/Output
MCU	MicroController Unit
USB	Universal Serial Bus
IP	Internet Protocol
TCP	Transmission Control Protocol **SAD** Sum of Absolute Difference **CT** Current Transformer
AC	Alternating Current
DC	Direct Current
SD card	Secure Digital card
PSU	Power Supply Unit
DSI	Display Serial Interface
CSI	Camera Serial Interface
HDMI	High Definition Multimedia Interface **MIPI** Mobile Industry Processor Interface **ES** Embedded System
KDE	K Desktop Environment
GPU	Graphics Processing Unit
ARM	Advanced RISC Machines
SDRAM	Synchronous Dynamic Random-Access Memory
BLE	Bluetooth Low Energy
CCD	Charge-Coupled Device **HAT** Host Access Transformation **PoE** Power over Ethernet **NOOBS** New Out of Box Software **OS** Operating System
ISO	International Organization for Standardization
IoT	Internet of Things
Tk	Taka
KWH	Kilo Watt Hour
CCTV	Close Circuit Tele-Vision **RFID** Radio-frequency Identification **SDK** Software Development Kit
AI	Artificial Intelligence
RGB	Red Green Blue

Acknowledgements

We would like to express our deepest gratitude to all those who have supported and contributed to the completion of this thesis. Firstly, we extend our sincere appreciation to our supervisor Dr. Khondokar Habibul Kabir for his guidance, expertise, and continuous encouragement throughout this research. His valuable insights and constructive feedback have immensely contributed to the success of this work. Additionally, we are grateful to our friends and family for their unwavering support and understanding during this challenging journey. Their encouragement and belief in our abilities have been instrumental in our perseverance. Finally, we acknowledge the participants and individuals who generously shared their time and knowledge for the data collection process. Without their cooperation, this research would not have been possible.

Abstract

The increasing demand for electricity coupled with depleting resources necessitates urgent action to reduce wastage and address the pressing need for efficient energy utilization. In large educational institutions a huge amount of energy is being wasted as the electrical appliances of a classroom are often switched on even though there are no occupants in the room. This thesis explores the development of an intelligent classroom system to enhance energy efficiency and reduce electric bills in traditional classroom settings. By leveraging automation and Internet of Things (IoT) technologies, the research aims to optimize energy consumption by tracking real-time occupancy and controlling lighting and fans accordingly. Various occupant detection models, including the MOG2 algorithm, OpenCV, and YOLOv3, are compared to identify the most effective approach for accurately detecting human occupants. The study presents a comprehensive methodology that includes the design of a hardware setup using Raspberry Pi, Pi cameras, and other components for occupant detection and appliance control. The key finding underscores the superior performance of YOLOv3 in accurately identifying human occupants while minimizing false detections. The proposed intelligent classroom system offers a user-friendly and customizable solution, enabling efficient energy management and cost reduction. This research contributes to the advancement of smart classroom technologies, promoting sustainability and improved resource utilization in educational institutions.

Chapter 1

Introduction

The increasing demand for electrical energy has become a pressing issue in modern society. Electricity has become an essential component of our daily lives, and our dependence on it continues to grow. However, this heightened consumption often leads to substantial electric bills. Negligence and forgetfulness contribute to wasteful energy usage, particularly when lights and fans are left on unnecessarily. In recent years, the pursuit of sustainable and energy-efficient solutions has become a paramount concern in various fields, including the design of educational environments. Integrating IoT based intelligent technologies, such as image processing, can help detect occupancy and optimize energy usage, providing an effective solution to minimize unnecessary energy consumption [1].

1.1 Problem statement

Traditional classrooms often suffer from inefficient energy usage due to static control systems that are not responsive to real-time occupancy. Unnecessary use of lights and fans when students leave the classroom contributes to energy waste and subsequently high electric bills [2]. Automation can provide better control over electrical equipment. IoT offers benefits in energy savings and can be utilized to develop a classroom automation system that tracks occupants and controls lighting and fans accordingly.

1.2 Research gap

Despite the potential benefits of employing an IoT-based system to address energy consumption issues, there remains a research gap concerning its specific implementation in intelligent classrooms. While IoT devices have demonstrated their capability to enhance daily life through interconnectivity and data analysis, further investigation is required to explore their effective utilization for energy reduction purposes [3]. Moreover, although IoT finds applications in various domains, the area of classroom automation emerges as a prominent research focus. However, there is a need to delve into the integration of object detection and motion detection within the context of classroom automation. Additionally, the existing literature lacks a comprehensive analysis and comparison of different occupant

detection models within an IoT-driven intelligent classroom system. This research endeavors to bridge these research gaps by proposing an IoT-based intelligent classroom system that leverages the YOLOv3 algorithm. Furthermore, it aims to conduct a comparative study of diverse occupant detection models. The primary emphasis of this research lies in reducing electricity costs through effective energy management strategies, providing multiple avenues for achieving this objective.

1.3 Problem identification

The identified problem stems from the escalating energy consumption and insufficient energy production, leading to substantial wastage. Negligent behavior and unnecessary usage of electrical appliances, even in unoccupied rooms, contribute to this issue. Recognizing the potential of IoT for energy savings, there is a need to develop a system that effectively reduces energy extravagance and minimizes human intervention. The core objective is to save energy by implementing an intelligent system that utilizes occupant detection through image processing to control electrical appliances efficiently. Existing approaches often overlook the potential of utilizing visual data and intelligent control mechanisms to optimize energy consumption.

1.4 Research motivation

The motivation behind this research stems from the urgent need to address the growing energy consumption and wastage in the large educational institutes. As energy demands continue to rise, it is crucial to find effective solutions to conserve energy and reduce unnecessary usage. The research is focused on implementing the energy-efficient intelligent classroom system in an educational institution, which consists of numerous classrooms and experiences high energy consumption.

Additionally, lights and fans are often left on unnecessarily when students forget to turn them off. The developed model aims to significantly decrease the institution's energy consumption, resulting in substantial energy savings and cost reduction. Implementing this cost-effective, user-friendly, and customized system becomes crucial in reducing energy consumption and lowering the institution's electric bill [4].

1.5 Scope of the research

The application scope of the energy-efficient intelligent system extends beyond classrooms to encompass a wide range of environments, including libraries, houses, banks, hospitals, laboratories, offices, industries, and any other location where occupants are present and frequent activation and deactivation of appliances is required. The system's versatility lies in its ability to leverage the existing electrical wiring infrastructure, eliminating the need for extensive rewiring processes. This flexible and adaptable solution enables optimized energy management in various settings, making it a viable option for enhancing energy efficiency and reducing wastage across diverse contexts.

1.6 Research objectives

To develop an efficient and intelligent classroom system this research will explore and compare occupant detection models, select the most suitable one, and implement a cost-effective and user-friendly solution that optimizes energy consumption based on occupancy. The main objectives are as follows:

1. **Compare different occupant detection models:** Conduct a comparative analysis of various occupant detection models to identify the most effective and accurate approach for detecting and tracking occupants in the classroom environment.
2. **Develop an energy-efficient intelligent classroom system:** Design and implement a system that utilizes the chosen occupant detection model to automatically control lighting and fan systems based on occupancy. The system should be capable of seamlessly scanning individuals, tracking their movements, and activating the corresponding lights and fans in their specific positions.
3. **Develop a system that is economically viable, user-friendly, and adaptable to individual preferences and requirements:** Ensure that the developed system is cost-effective, making it accessible for implementation in educational institutions. Focus on creating a user-friendly interface that is easy to operate and customize according to specific classroom requirements and preferences.

1.7 Research outcome

The implementation of the proposed system will lead to the following research outcomes:

1. **Creation of an automated smart classroom environment:** The research will result in the creation of a smart classroom where all appliances are controlled automatically based on the presence of occupants. This outcome will enhance the overall efficiency and convenience of the classroom environment.
2. **Comparative analysis of occupant detection models for intelligent classroom systems:** The research will involve implementing and evaluating various occupant detection models. This comparison will provide insights into the performance, accuracy, and reliability of different models, assisting in the selection of the most effective approach for occupant detection in the intelligent classroom system.
3. **Creation of a user-centric, economically viable and customizable system:** The developed intelligent classroom system will prioritize cost-effectiveness, user-friendliness, and customization. This outcome will ensure that the system is affordable, easy to operate, and adaptable to the specific requirements and preferences of different classrooms.

1.8 Novelty of the research

This research presents a novel contribution in the form of an intelligent classroom system that leverages IoT technologies and advanced occupant detection models to optimize energy usage. The distinguishing factor of this study is the utilization of real-time video feed from an operational classroom for accurate occupant detection, setting it apart from previous research that relied on test models. The incorporation of state-of-the-art image processing techniques, including the MOG2 algorithm, OpenCV, and YOLOv3, enables precise and real-time monitoring of classroom occupancy. Furthermore, the comparative analysis and selection of the most effective occupant detection model further enhance the originality of this study.

1.9 Research methodology

We have studied 3 occupation detection models in this thesis book. The efficiency of these models is compared. These models are:

1. Background Subtraction (MOG2 algorithm)
2. OpenCV
3. YOLOv3 (Real time object detection algorithm)

By comparing the occupation detection models, the most suitable algorithm will be chosen for this model. We have also proposed a blueprint for the design of the hardware setup. This model can be built using Raspberry Pi model 3B, Pi cameras and other necessary equipment which makes this model economically viable and customizable. By incorporating the most suitable algorithm with this setup, our model will be able to detect occupants in a particular zone and control the electrical appliances of that zone resulting in creation of an automated smart classroom environment.

1.10 Organization of the thesis

The thesis paper has been divided into the 7 following chapters-

Chapter 1 presents a short introduction about the use of IoT in order to save the electrical energy. It represents the research gap, problem identification, research motivation, scope of the research, research objectives, research outcomes and so on.

Chapter 2 demonstrates a detailed description of the related work.

Chapter 3 provides detail information about the proposed model.

Chapter 4 describes the comparison of three image processed occupant detection models.

Chapter 5 describes the hardware and software setup.

Chapter 6 is devoted to represent the calculated results, graphs and explanation of the results obtained.

Chapter 7 contains the conclusions of the whole study.

At the end of each chapter a complete list of references corresponding to that chapter has been given.

Chapter 2

Related Works

Recent research indicates that "the automation system" can be utilized in various disciplines, which is an inspiration for further development.

Darshan Ganiger et al. [5] proposed an automated system using image processing for controlling the power supply in classrooms. As a criterion for deciding whether to switch on or off the power supply, the system detects the motion of objects, thereby reducing energy consumption. But the proposed model has its limitations. If the classroom benches are rearranged, the motion detection method may not work. The bench orientation can be adjusted without changing the code by taking a fresh reference image which is a hassle itself. Moreover, the paper does not include the cost of implementing the proposed method, which may limit educational institutions with low finances. Lastly, proposed system's environmental impact, such as reduced power consumption and carbon emissions, is not discussed in the paper. In that case, this study is more dynamic and efficient in motion detection than that of the aforementioned paper.

Ms. Tanuja Sali et al. [6] proposed a schema for an intelligent classroom automation system. This paper examines the application of the Internet of Things in automation and analysis, with a particular emphasis on household and classroom automation systems. The suggested approach has not been tried in a real-world classroom, therefore more research and testing may be needed to establish its efficacy and usability. Moreover, a onetime sign up is necessary to initiate the model which may be problematic. In our research, no such type of sign up from the students is required.

Ms. Anisha Gupta et al. [7] suggested an Ethernet-based intelligent automated system for electrical energy minimization utilizing an INTEL GALILEO 2nd generation development board, which could be implemented in large organizations such as a university or office. The design suggested is automated, permitting remote control and monitoring of electrical appliances and controls without human intervention. It should be noted that installing and

implementing the proposed system may need upfront investment and setup charges. To perform properly, the system may need regular maintenance and upgrades which is also applicable in our proposed model too. One thing that should be noted is that our proposed model does not require that much upfront investment.

Rathy G. A et al. [8] proposed a design of a Class Rooms Automation System (CRAS) utilizing internet-connected smart devices in classrooms. The entire classroom automation system is designed and controlled with Zigbee and Cortex M3. One thing that has to be noted is that the paper does not discuss the potential security risks associated with using IoT devices in a classroom setting, which could be a concern for some institutions. In that case, our research is very much promising and does not have any kind of security risks.

Mrityunjaya Patted et al. [9] proposed a PIC microcontroller-based intelligent classroom automation system that incorporates attendance monitoring, message transmission, and control of appliances such as fans and lights in order to conserve energy and maintain the classroom system's smooth operation. The primary objective of the system is to reduce energy consumption and enhance teaching methods by removing the need for manual intervention from teachers, students, and floor attendance.

Shruthi et al. [10] proposed a motion detection model using the background subtraction technique, a popular method for identifying movement in video. The paper describes a frame-difference method for classifying and detecting video motion. Coming to the shortcoming of this paper, the research indicates that comparing the current frame to a fixed backdrop frame works best with a static background and no motionless objects in the following frames. All videos experience this. The approach performs poorly in all non-stationary backgrounds, which is why we are also implementing two other methods, apart from MoG2, in this research to contrast and compare amongst the three methods.

S. S. Shazali et al. [11] proposed a method for detecting motion by subtracting periodic background estimations. Using the region as a blob, the method generates bounding frames and centroid coordinate data for each blob. The system cannot detect moving items when they stop moving and become part of the background. The approach may also fail in abrupt

lighting changes. In comparison to that, our proposed model does not have that kind of limitations.

S. Khedkar et al. [12] proposed the use of Raspberry Pi and GSM in the development of a home automation application. The objective is to control various household activities, such as illumination, HVAC, and security locks, via a centralized controller to enhance comfort, convenience, security, and energy efficiency. Our research has taken inspiration from home automation techniques in order to apply them to classrooms.

Vaishnavi Kulkarni et al. [13] proposed a Natural Language Processing (NLP), Wi-Fi, and wireless sensor-based IoT-based classroom automation system. Voice-activated and sensor-based, the system enables users to control appliances and receive notifications regarding their condition, while reducing energy consumption and enhancing security. The paper does not discuss the system's scalability or its potential limitations in larger environments. But in this case, this research has gone to an in-depth length about the scalability of the proposed model.

C. Paul et al. [14] proposed an IoT-based smart classroom system that uses the MQTT protocol to automate electronic classroom appliances. The system architecture consists of wireless nodes, middleware, and a user interface for middleware interaction. It is essential to note that implementing and maintaining the proposed system architecture may necessitate a high level of technical expertise. In addition, the system may encounter difficulties in terms of scalability and compatibility with various electronic devices. In case of our research, difficulties in terms of scalability and compatibility with various electronic devices is not an issue.

Chapter 3

Overview of the Proposed Model

Here, we are utilizing classrooms for the implementation of our system, as we are making classrooms energy-efficient. As an example, for the implementation of the proposed model, we are utilizing a classroom from our university as an illustration.

3.1 Model Classroom Representation

We are taking the classroom of our institution, Islamic University of Technology as the model classroom. We will be dividing the whole classroom into 9 zones which will aid us to detect the total participants in the classroom as well as monitor the motion of the participants present in the classroom

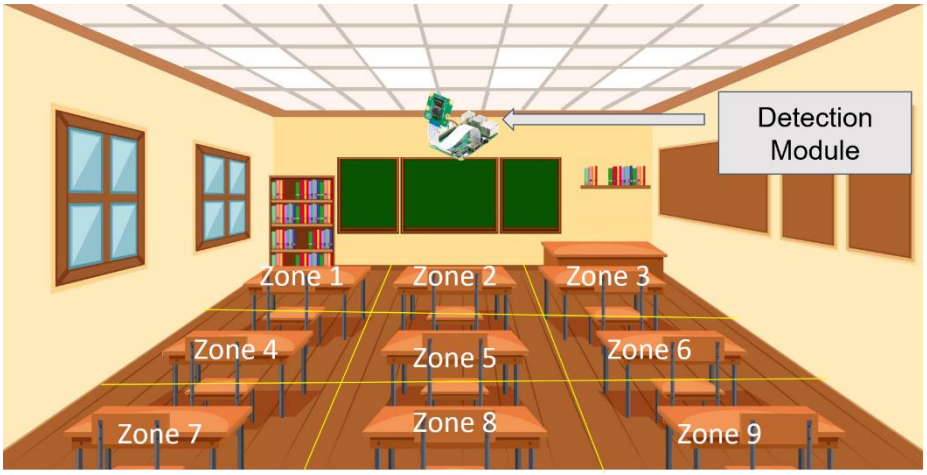

Fig 3.1: Model Classroom Representation

There is a total of forty-five students in the classroom. Each classroom is divided into three columns where every column has six benches. Every bench can accommodate a total of three students. So, the total number of chairs presents in the classroom is fifty-four.

3.2 Targeted amount of energy saving in a year

A brief calculation of energy saving in a year in accordance to that of IUT classrooms in represented below-

Table 3.1: Description of the classroom

1. The institution has 100 classrooms.
2. Each class consists of 45 students.
3. One light of 50 Watt for every 3 students and 2 lights for the teacher
4. One fan of 100 Watt for every 5 students and 2 fans for the teacher
5. 1 AC of 2000 Watt for every 15 students
6. Daily there are 6 periods
7. Each period consist of 1.25 hours
8. There is a recess break of 1.5 hours
9. The institution has 200 working days

Total lights:

$$(45 \div 3) + 2 = 17$$

Total fans:

$$(45 \div 5) + 2 = 11$$

Total ACs:

$$45 \div 15 = 3$$

Now, if we consider that the lights, fans and ACs were kept round clock as well as the recess period, the total energy consumption of one classroom for one day in KWh will be-

$$((17 \times 50 + 11 \times 100 + 3 \times 2000) \times 9) \div 1000 = 71.55$$

Total energy consumption of the institution for one year in KWh will be-

$$71.55 \times 100 \times 200 = 547200 = 1431000$$

So, the energy consumption of the institution is calculated as 1431 MWh for a year. According to Bangladesh Energy Regulatory Commission (BERC) per unit energy consumption bill is 11.46 taka for consumers over 600 units. So, we can see that, it costs 16.39 million taka per year as the electric bill.

After applying our developed model in the system, the working hour of one classroom for one day becomes 7.5 hours as we can easily neglect the recess period. Then the average power consumption of one classroom in KWh becomes-

$$((17 \times 50 + 11 \times 100 + 3 \times 2000) \times 7.5) \div 1000 = 59.625$$

Total energy consumption of the institution for one year in KWh will be-

$$59.625 \times 100 \times 200 = 547200 = 1192500$$

So, the energy consumption comes down to 1192.5 MWh per year.

Saving of the electric bill becomes-

$$(1431000\text{-}1192500) \times 11.46 = 2.73 \text{ million taka}$$

Now, if we properly apply our proposed model, the running time of the electrical appliances comes down to 6.53 hrs. The proper break down of this calculation is presented in Chapter 6. Then the average power consumption of one classroom in KWh becomes-

$$((17 \times 50 + 11 \times 100 + 3 \times 2000) \times 6.53) \div 1000 = 51.9135$$

Total energy consumption of the institution for one year in KWh will be-

$$51.9135 \times 100 \times 200 = 547200 = 1038270$$

So, the energy consumption comes down to 1038.27 MWh per year.

Saving of the electric bill becomes-

$$(1431000\text{-}1038270) \times 11.46 = 4.5 \text{ million taka}$$

So, we can easily say that the proper implementation of our proposed model can save the institution upto 4.5 million taka per year.

3.3 Technique

The system initially employs image processing to determine the presence of occupants. It then analyzes movement by comparing the current frame to the previous frame. Based on the results of the image processing, the system determines whether or not to power on the classroom's appliances. The system is comprised of the ESP32 and Raspberry Pi, which are linked via Wi-Fi.

The Raspberry Pi single-board computer series is selected due to its minimal cost, modularity, and open architecture. It is a potent computer that supports the most recent web innovations and data sets, providing numerous opportunities for the project's expansion.

The ESP32 is an inexpensive Wi-Fi microprocessor that combines a complete TCP/IP system with a microcontroller. In this system, a relay module is used as a discrete hardware device for remote device switching, facilitating network or internet-based control.

When the Raspberry Pi starts up, the system runs the proposed algorithm. This includes comparing the current frame to the last frame taken by the camera to see if people are moving. If the system sees movement in a certain part of the classroom, it sends a signal to the Raspberry Pi to let it know that someone is in that spot. So, the Raspberry Pi checks to see if the person is standing or sitting and sends a signal to the ESP32 based on what it finds. The ESP32 then talks to the relay module and tell to toggle on the devices in that part of the classroom.

Here the environment is Python and the module is OpenCV. The code is done in Python language using Spyder. Thonny Python is used to run the Python script for Raspberry Pi.

3.4 Input

The inputs are synchronized according to the following four categories:

1) The classroom's reference image functions as the premise or benchmark.
2) A depiction of the room's coordinate system, which determines four coordinate points at the room's extremities to provide a precise bird's-eye view.
3) Identification and monitoring of the coordinates of room appliances.
4) A continuously active live video transmission that is regarded as the captured frame.

3.5 Dependency

The proposed model is dependent on the occupants that are present in the room. Moreover, the placement of the camera is very much important for the proper efficiency of the model.

3.6 Parameter

There are two types of parameters in our proposed model, viz- constant and variable.

3.6.1 Constant parameters:

1) Reference frame
2) Position of the camera
3) Coordinate system

3.6.2 Variable Parameter:

Live video feed of the presence of students in different parts of the room

Chapter 4

Algorithm Developments

In this chapter, three algorithms are compared to find the best and most efficient algorithms to detect human movement in a classroom environment. The algorithms are

1. Background subtraction (MOG 2 algorithm)
2. Computer Vision (using different algorithms)
3. You Only Look Once (YOLO V3)

4.1 Mixture of Gaussians 2 (MOG2) Algorithm

Background removal is a crucial preprocessing measure in video processing systems that rely on this method. One possibility involves using a visitor counter in a public location, where a stationary camera records the number of people entering and exiting the area. Another example is employing a traffic camera to collect vehicle data [15]. In these instances, the human or car must be extracted first.

Technically, it is required to separate the moving foreground objects from the fixed background. When faced with an image that only depicts a background, such as an empty room or a road devoid of vehicles, the process of analysis is quite simple. To assist in the extraction of the foreground objects, remove the newly presented image from the backdrop. However, the availability of such an image may be limited in the majority of circumstances, necessitating the extraction of the background from the available frames.

The use of the Gaussian Mixture-based Background/Foreground Segmentation Algorithm aids in the achievement of this goal. The technique used comprises the use of a technique in which each backdrop pixel is represented by a combination of K Gaussian distributions, with K ranging from 3 to 5. The weights applied to the pixel mixture indicate the length and scope of those specific colors' existence in the given scene. Background hues that display increased endurance and stability are considered viable possibilities [16].

4.1.1 Basics of MOG2 Algorithm

The MOG2 algorithm is a Background/Foreground Segmentation Algorithm based on Gaussian Mixtures [17]. It is designed to handle variations in lighting and other factors, making it more adaptable to changing scenes [17]. In this algorithm, each background pixel is modeled by combining K Gaussian distributions, where K typically ranges from 3 to 5 [18]. The mixture weights represent the proportions of time that certain colors remain in the scene [18]. This flexibility in selecting the appropriate number of Gaussian distributions for each pixel contributes to its improved adaptability in dealing with changing scenes [19].

One advantage of the MOG2 Subtractor is that it is preinstalled in OpenCV, making it easier to use compared to the Manual-mode [19]. The MOG2 Subtractor also incorporates the concept of dealing with a framed history, where it considers a specified number of past frames (e.g., the last 120 frames) [19]. This approach allows the algorithm to handle scenarios where significant changes occur over time, such as the transition from day to night. By considering a history of frames, the MOG2 Subtractor avoids issues that may arise when comparing the current frame to a single reference frame taken hours ago [20].

In summary, the MOG2 algorithm is a powerful Background/Foreground Segmentation Algorithm that leverages Gaussian Mixtures and adapts well to changing scenes [17]. Its ability to model background pixels with multiple Gaussian distributions and its consideration of a framed history contribute to its effectiveness in handling variations in lighting and other environmental factors [18] [20]. The MOG2 Subtractor, which is readily available in OpenCV, simplifies the usage of this algorithm [19].

4.1.2 Construction of MOG2 Algorithm

The equation for the Gaussian mixture model in the MOG2 algorithm is as follows:

$$P(x) = \Sigma w_i * N(x; \mu_i, \sigma_i)$$

In this equation:
P(x) represents the probability of a pixel value x.
Σ denotes the summation operation.

w_i represents the weight or proportion of the i-th Gaussian distribution.
N(x; μ_i, σ_i) represents the Gaussian distribution with mean μ_i and standard deviation σ_i [29].

The MOG2 algorithm is created in the coding process using the "cv.BackgroundSubtractorMOG2" constructor with preset parameters [21]. A group of variables or elements that define the boundaries or limits of a specific system, process, or phenomenon are referred to as parameters.

1. History, the length of the historical period under investigation is an important aspect in determining the number of preceding frames that are considered.
2. The varThreshold parameter is used to determine if a pixel is close to a reference sample based on the squared distance between the two.
3. By setting the detectShadows option to true, the algorithm can detect and label shadows.

The implementation of cv.BackgroundSubtractorMOG2 is in charge of creating an object that implements the MOG2 algorithm, as described in reference [21].
The algorithm can be implemented using the following steps, as shown sequentially for a better understanding [22].

1. **Load the databases and video, and initialize the subtractor:**
    a. Load the databases and video.
    b. Instantiate the subtractor using the cv.BackgroundSubtractorMOG2 constructor with appropriate parameter values. The history parameter determines the number of recent frames considered, while the varThreshold and detectShadows parameters control the distance threshold for pixel classification and shadow detection, respectively.
2. **Extract frames from the video using a while loop:**
    a. Use a while loop to continuously extract frames from the video source.
    b. Store the current frame in a variable, such as "frame".
3. **Apply the subtractor to process the frames:**

a. Use the subtractor's "apply" method to process the frame and obtain a foreground mask.
b. Store the resulting mask in a variable, such as "mask".

4. **Display the output:**
    a. Show the frame and the mask on a projector or suitable display device to visualize the results.

These steps illustrate the basic construction and implementation of the MOG2 algorithm for background subtraction and motion detection.

4.1.3 Real-Life Example of MOG2

For a better understanding of the entire process, a real-life video is used as an example.

Original Video:

A snapshot of a frame input into the algorithm

Fig 4.1: Input for the MOG Algorithm

MOG Algorithm Output:

Here the static parts of the frame are grey or black colored and the moving part of the frame is highlighted as white.

Fig 4.2: Output the MOG Algorithm

4.1.4 Occupant Detection Process for MOG2 Algorithm

Occupant detection is a crucial task in various domains such as surveillance, smart homes, and occupancy monitoring systems. The MOG2 algorithm, known for its robust background subtraction and motion detection capabilities, can be enhanced to incorporate occupant detection. This integration involves additional processing steps to identify and track occupants based on the foreground mask generated by the MOG2 algorithm [23].

The first step in the occupant detection process is to apply morphological operations to the foreground mask. Techniques like erosion and dilation are utilized to remove noise and refine the shapes of detected objects. This helps to smoothen the mask and fill in gaps or holes, leading to more accurate object representation [24].

Once the refined mask is obtained, contour detection techniques are employed to identify individual objects or regions within the foreground. Contours are extracted from the connected components in the mask, providing a basis for further analysis. To ensure reliable occupant detection, contour filtering is applied to eliminate small or irrelevant objects. This is achieved by setting appropriate thresholds based on size, aspect ratio, or other criteria, thereby retaining only the contours representing potential occupants [25].

To track the movement of occupants across frames, object-tracking algorithms are employed. Popular approaches such as the Kalman filter or centroid tracking are utilized to maintain object continuity and assign unique identifiers to tracked objects. Object tracking enables the

monitoring of occupants' positions and trajectories over time, facilitating further analysis and inference [26].

Occupant classification is another crucial aspect of the detection process. Machine learning or computer vision techniques can be employed to classify the tracked objects as occupants or non-occupants. By training a model using labeled data, it becomes possible to distinguish between different classes, such as humans, furniture, or pets. This classification step provides valuable information for various applications, including security, energy management, and personalized services [27].

Post-processing techniques play a vital role in refining occupant detection results. Methods like non-maximum suppression or temporal filtering can be utilized to eliminate duplicate or false detections, improving the accuracy of occupancy information. Additionally, occupancy validation algorithms can be applied to verify the presence of occupants based on duration or movement patterns, ensuring more reliable and trustworthy occupant detection [28].

4.1.5 Issues with the MOG2 Algorithm process

One of the limitations of the MOG2 algorithm for occupant detection is its dependency on detecting movement to identify occupants. The algorithm is designed to detect occupants only when they are in motion, and it fails to detect them when they are stationary. This compromises the usability of the algorithm in scenarios where occupants may not be continuously moving. Continuous movement is often impractical for human occupants, making the algorithm less effective in real-world applications.

In Figure 4.1, the system considers the initial frame as the previous frame. After a certain period, the system captures another frame as the present frame. The algorithm then compares the present frame with the last frame to detect any movement from the previous position. However, if none of the occupants have moved significantly between the frames, the algorithm fails to detect their presence. Consequently, the system is marked as deactivated, indicating a lack of occupant detection. As a result, even if there are two students present on the scene, the appliances remain turned off.

To address this limitation, alternative approaches can be explored. One possible solution is to use computer vision instead of MOG 2 Algorithm. We will see details about that in the next part.

4.2 Computer Vision Algorithm

The second method which was tested to detect motion is Computer Vision namely OpenCV. OpenCV (Open Source Computer Vision) is an open-source library that provides a wide range of computer vision and image processing functions. It is written in C++ and has bindings for various programming languages, including Python. OpenCV offers a comprehensive set of tools and algorithms that enable developers to perform tasks such as image and video processing, object detection and tracking, feature extraction, camera calibration, and more.

Originally developed by Intel in 1999, OpenCV has gained immense popularity due to its versatility, efficiency, and extensive community support. It is widely used in various domains, including robotics, augmented reality, medical imaging, surveillance, and automotive applications.

OpenCV provides a vast collection of functions that cover a broad spectrum of computer vision tasks. These functions include image filtering, geometric transformations, feature detection and matching, camera calibration, optical flow, object recognition, and machine learning integration. With its rich functionality, OpenCV empowers developers to build robust computer vision applications with ease.

The library is designed to be cross-platform, making it compatible with different operating systems such as Windows, macOS, Linux, and even mobile platforms like Android and iOS. OpenCV's easy-to-use API and extensive documentation make it accessible to both beginners and experienced developers.

OpenCV is continuously evolving and incorporates the latest advancements in computer vision research. It provides a flexible and efficient framework for implementing computer

vision algorithms, enabling developers to leverage cutting-edge techniques and achieve accurate and reliable results. [30]

4.2.1 Basics of OpenCV

OpenCV (Open Source Computer Vision) is a widely used open-source library for computer vision and image processing tasks. It provides a vast collection of functions and algorithms that enable developers to work with images and videos, perform various operations, and extract valuable information from visual data. Here are some key basics of OpenCV:

1. **Image Input and Output:**
    a. Loading and saving images in various formats (e.g., JPEG, PNG, BMP).
    b. Reading and writing videos from files or capturing them from cameras.
2. **Image Processing:**
    a. Image resizing, cropping, and rotation.
    b. Adjusting brightness, contrast, and color balance.
    c. Applying filters for blurring, sharpening, and edge detection.
    d. Converting images between color spaces (e.g., RGB, grayscale, HSV).
3. **Image Analysis:**
    a. Feature detection and description for identifying key points and extracting descriptors.
    b. Object detection and recognition using pre-trained models or custom classifiers.
    c. Image segmentation to separate objects or regions of interest.
    d. Contour detection and shape analysis.
    e. Histogram computation and image statistics.
4. **Camera Calibration:**
    a. Estimating intrinsic and extrinsic camera parameters.
    b. Correcting lens distortion.
5. **Video Processing:**
    a. Reading and processing video frames.
    b. Background subtraction for detecting moving objects.
    c. Optical flow estimation for motion analysis.

d. Video stabilization.

6. **Machine Learning Integration:**
    a. Training and using machine learning models for tasks like classification and regression.
    b. Utilizing pre-trained deep learning models for image recognition and object detection.
7. **Graphical User Interface (GUI) Support:**
    a. Displaying images and videos in Windows.
    b. Handling user input through mouse and keyboard events.
    c. Creating interactive graphical applications.

OpenCV supports various programming languages, including C++, Python, Java, and more. It is cross-platform, allowing developers to use it on different operating systems such as Windows, macOS, Linux, and mobile platforms like Android and iOS.
The extensive documentation and active community support make OpenCV a valuable resource for beginners and experienced developers working on computer vision projects.

4.2.2 Construction of OpenCV Algorithm

OpenCV is basically a tool for image processing and related work. To detect movement using OpenCV, we had to come up with our own set of sequences of algorithms. To do that we have to revert to the original understanding of how the computer detects any kind of motion. Our input source i.e., camera records in RGB or color format, but for movement detection we don't need that. Also, a camera might have a lot of resolution which might affect the processing power of the whole system. Then we have to take the difference between the two frames and detect motion. So, we have come up with 4 sequential algorithms to do the said task. They are:

1. Grayscale conversion
2. Difference Detection
3. Threshold calculation
4. Contour Generation

4.2.2.1 Grayscale conversion

First, in Grayscale conversion two types of algorithms are handled.

A. Converting color image to gray_scale image

B. Converting grayscale image to GaussianBlur

Converting color image to grayscale image:

This is done by using the function *cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)*.
The code snippet is an example of using the cvtColor function from the OpenCV library (cv2) to convert an image frame from the BGR color space to grayscale.

In computer vision and image processing, images are typically represented in different color spaces. BGR (Blue-Green-Red) is the default color space used by OpenCV for reading and displaying images. However, for our detection task or tasks like image analysis or feature extraction, it is often beneficial to convert the image to grayscale.

The cvtColor function is used to perform color space conversions. Here, frame refers to the input image frame. By passing cv2.COLOR_BGR2GRAY as the second argument, it specifies the conversion from BGR to grayscale.

The function cv2.cvtColor takes the input image and the color conversion flag as arguments, and it returns the converted grayscale image. The grayscale image has a single channel, where the pixel values represent the intensity or brightness of the original image.

This conversion is useful for reducing the dimensionality of the image, simplifying subsequent processing steps, and focusing on the image's structural information rather than color. Grayscale images are often used in tasks such as edge detection, image thresholding, and feature extraction.

Converting grayscale image to GaussianBlur:

This is done by using the function *cv2.GaussianBlur(gray, (21, 21), 0)*
The code snippet demonstrates the usage of the GaussianBlur function from the OpenCV library (cv2) to apply a Gaussian blur filter to a grayscale image.

In image processing, a Gaussian blur is a widely used technique for reducing noise and smoothing images. It applies a weighted average of nearby pixels to each pixel in the image, resulting in a blurring effect. The Gaussian blur is named after the Gaussian distribution, as the weights assigned to neighboring pixels follow a Gaussian kernel.

In the given example, gray refers to the grayscale image on which the Gaussian blur is applied. The (21, 21) argument specifies the size of the Gaussian kernel, which determines the extent of blurring. A larger kernel size results in a stronger blur effect. The last argument, 0, denotes the standard deviation of the Gaussian distribution. Setting it to 0 instructs OpenCV to automatically calculate the standard deviation based on the kernel size.

Fig 4.3: Grayscale conversion and noise removal with Gaussian blur

The cv2.GaussianBlur function takes the input image and the kernel size as arguments and returns the blurred image. The function convolves the image with the Gaussian kernel, replacing each pixel's value with the weighted average of its neighbors.

Gaussian blur is commonly employed in various image processing tasks, such as noise reduction, image smoothing, and pre-processing steps before performing more complex operations like edge detection or feature extraction.

4.2.2.2 Difference Detection

Difference detection is done using the function *cv2.absdiff(static_back, gray)* It demonstrates the usage of the absdiff function from the OpenCV library (cv2) to compute the absolute difference between two images.

In computer vision and image processing, calculating the absolute difference between two images is a common operation used for tasks like motion detection, change detection, and background subtraction.
In the given example, static_back and gray are the input images for which the absolute difference is computed. The static_back image is typically a reference or background image, while gray is usually the current frame or an image captured at a different time.

The cv2.absdiff function takes these two images as arguments and returns the absolute difference image, which represents the pixel-wise absolute intensity differences between corresponding pixels in the input images.

Each pixel in the resulting absolute difference image represents the absolute value of the intensity difference between the corresponding pixels in the input images. The larger the difference between pixel intensities, the brighter the corresponding pixel in the absolute difference image.

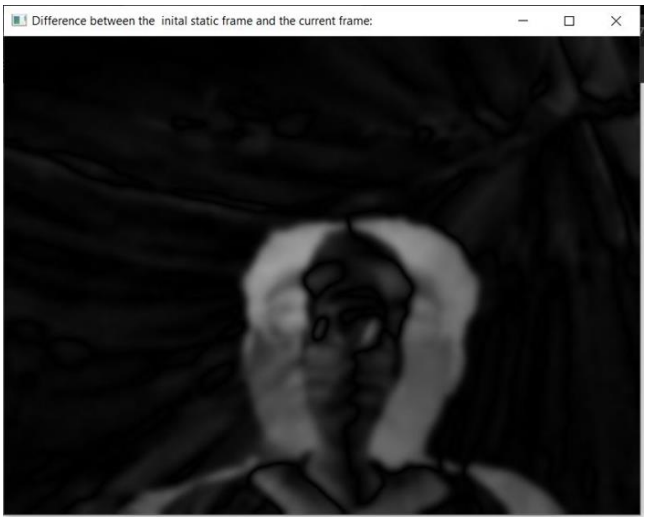

Fig 4.4: Difference Frame

This technique is often used in motion detection applications, where the absolute difference image is thresholded to identify regions where significant changes have occurred between frames. These regions can then be further processed or analyzed to detect and track moving objects.

4.2.2.3 Threshold calculation

To implement the threshold detection algorithm two functions are used
thresh_frame = cv2.threshold(diff_frame, 30, 255, cv2.THRESH_BINARY) [1]
thresh_frame = cv2.dilate(thresh_frame, None, iterations = 2)
The code thresh_frame = cv2.threshold(diff_frame, 30, 255, cv2.THRESH_BINARY)[1] followed by thresh_frame = cv2.dilate(thresh_frame, None, iterations = 2) demonstrates the usage of thresholding and dilation operations using the OpenCV library (cv2).
In image processing, thresholding is a common technique used to separate objects or regions of interest from the background based on pixel intensity values. It converts a grayscale image into a binary image, where pixels above a certain threshold value are set to a maximum value (e.g., 255), and pixels below the threshold are set to zero.
In the given example, diff_frame represents the input image or frame on which thresholding is applied. The 30 argument specifies the threshold value, which determines the intensity level at which the pixel values are separated into foreground and background. Pixel values above the threshold are set to 255 (white) in the binary image, while values below the threshold are set to 0 (black).

The cv2.threshold function takes the input image, threshold value, maximum value, and thresholding method as arguments. It returns a tuple of two elements: the threshold value used and the thresholded image. In this case, the [1] indexing is used to extract the second element of the tuple, which is the thresholded image.

After thresholding, the cv2.dilate function is applied to the thresh_frame image to enhance and expand the foreground regions. Dilation is a morphological operation that enlarges the white regions in a binary image by adding pixels to their boundaries. This helps to fill in gaps and smooth the segmented regions.

Fig 4.5: Threshold Frame

The cv2.dilate function takes the thresholded image, a structuring element (here, None is used to create a default rectangular kernel), and the number of iterations as arguments. It returns the dilated image. If the intensity difference for a particular pixel is more than 30 in our case then that pixel will be white and if the difference is less than 30 that pixel will be black

4.2.2.4 Contour Generation

Finally, to visualize the result we have to determine where the movement has been detected. This is done by the following function

cnts,_ =

cv2.findContours(thresh_frame.copy(),cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)

This demonstrates the usage of the findContours function from the OpenCV library (cv2) to find and extract contours from a binary image.

In image processing, contours are the boundaries of objects or regions in an image. The findContours function is used to detect and extract these contours from a binary image.

In the given example, thresh_frame is the binary image from which contours are to be extracted. The copy() method is used to create a copy of the thresholded image since the findContours function modifies the input image.

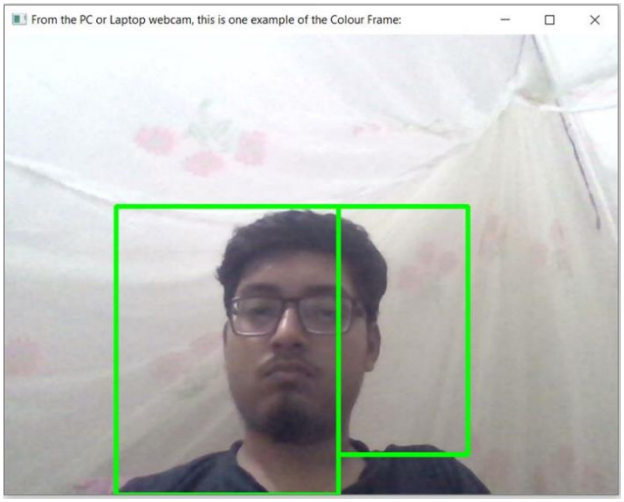

Fig 4.6: Output result frame with bounding rectangular box showing movement

The cv2.findContours function takes the input image, contour retrieval mode, and contour approximation method as arguments. The contour retrieval mode is specified as cv2.RETR_EXTERNAL, retrieves only the external (outermost) contours of the connected components in the image. The contour approximation method is specified as cv2.CHAIN_APPROX_SIMPLE, approximates the contours using a simple algorithm that compresses horizontal, vertical, and diagonal segments.

The function returns two values: the extracted contours, represented as a list of contour points, and the hierarchy, which provides the relationship between different contours (not used in this example, denoted by _).

The cnts variable stores the extracted contours, which can be further processed or analyzed for various tasks such as object detection, shape analysis, or object tracking.

4.2.3 Real-Life Example of OpenCV

For a better understanding of the entire process, a real-life video is used as an example.

Original Video:

A snapshot of a frame input into the algorithm

Fig 4.7: Input for the OpenCV Algorithm

OpenCV Algorithm Output:

Here the moving part of the frame is highlighted as a green contour.

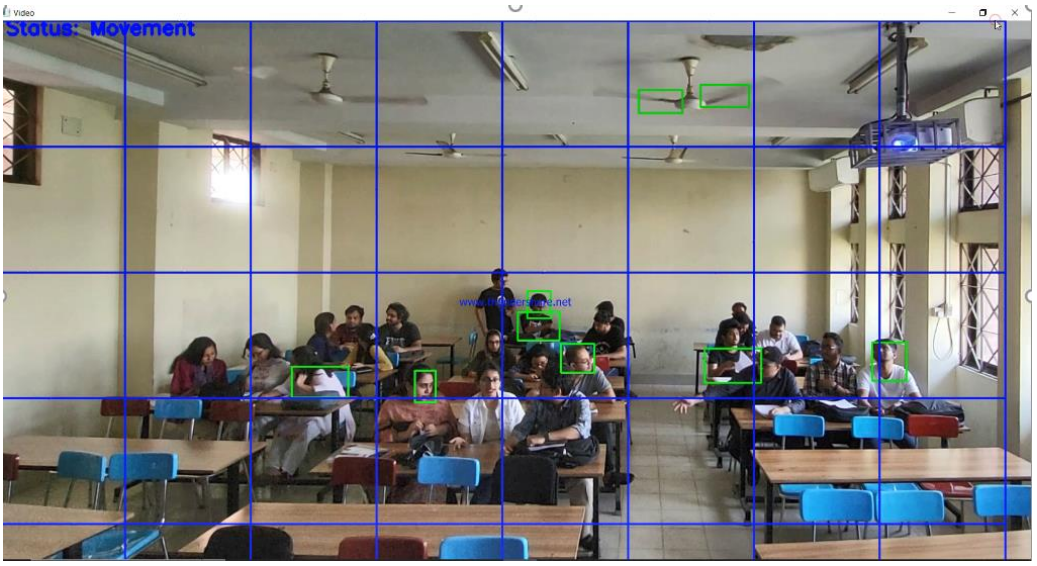

Fig 4.8: Output the MOG Algorithm

4.2.4 Issues with the OpenCV Algorithm process

The built-in OpenCV algorithm for motion detection has certain limitations. It detects any kind of motion without considering the specific objects or elements involved. In a classroom environment, various factors contribute to motion, not just humans. For example, ceiling fans are constantly moving and the algorithm tends to put a contour around them as well. This shows that the algorithm alone is not sufficient for accurate motion detection.

Another issue is the limited processing power of the algorithm. It fails to detect all the moving objects in a classroom environment due to this limitation. This undermines the purpose of the entire system, as it is essential to accurately detect and track human occupants to perform power management.

Therefore, additional improvements or modifications are necessary to overcome these challenges and make the motion detection system more reliable and effective in a classroom setting.

4.3 You Only Look Once (YOLO V3)

The artificial intelligence area is built on object detection techniques. You Only Look Once (YOLO) is a popular and well-known algorithm [31]. YOLO is well-known for its ability to identify objects. The first YOLO version was introduced in 2015 by Redmon et al. [32]. Using a sliding window technique, traditional object identification approaches required running a classification algorithm on several sections of a picture. This method was time-consuming and computationally costly. YOLO, on the other hand, altered the game by offering a single neural network architecture capable of predicting bounding boxes and class probabilities for many objects in a single image. The main idea behind YOLO is to divide the input picture into grid cells and assign bounding box predictions and class probabilities to each grid cell. Each grid cell predicts a given number of bounding boxes, as well as confidence ratings indicating the existence of an item and class probabilities for the objects anticipated. These predictions are then adjusted using non-maximum suppression to remove superfluous bounding boxes and increase final output accuracy.

A study shows that YOLO v3 has advantages in detection speed while maintaining certain MAP and thus can be applied for real-time pill identification in a hospital pharmacy environment [33]. YOLO offers an excellent real-time performance by concurrently optimizing the object identification job and the underlying neural network architecture, making it appropriate for applications requiring speedy and precise object recognition, such as autonomous driving, video surveillance, and robotics.

4.3.1 Architecture of YOLO V3

YOLO is a Convolutional Neural Network (CNN)-based model with a single stage. Objects may be recognized in the single-stage model without the requirement for a prior stage. YOLO divides the image into grids and applies a single Convolutional Neural Network model to it [34]. Each grid calculates the bounding boxes and the accompanying confidence score. The class of the item in the bounding box is established using the predicted confidence score. The method "only looks once" at the picture, which means it only needs one forward propagation to pass through the network to generate predictions. It gives the name of the recognized after non-max suppression [35].

YOLO (v3) is influenced by The Feature Pyramid Network (FPN). YOLO(v3), like FPN, contains heuristics such as residual blocks, skip connections, and up-sampling. It starts with Darknet53 as a basic network, then adds 53 extra layers to make object detection easier. YOLO (v3), like FPN, detects objects using (1 1) convolution on feature maps. It creates three distinct scales of feature maps. with the use of a stride of 16. A few convolutions are added to the 79th layer, which is then concatenated with the 61st layer on 2x up-sampling to produce a 26 26 feature map. Finally, the detection is performed by the 106th layer, which has a 52-52 feature map and a stride of 8. Following the same procedure of adding a few convolutions to the 91st layer and combining it with the 36th layer using the (1 1) kernel, the down-sampled feature maps are concatenated to the up-sampled feature maps at various points to extract fine-grained features for detecting smaller objects of various dimensions. Different feature maps, namely 52 52, 13 13, and 26 26, are used to identify big, smaller, and medium-sized objects, accordingly [36]. In this proffered study we used the coco dataset which only identifies Humans to detect movements in the classroom.

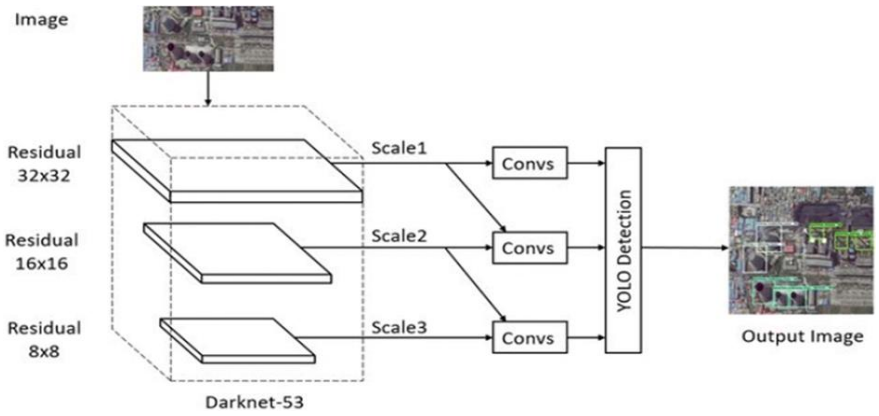

Fig 4.9: YOLO v3 Architecture

4.3.2 Movement Detection Process of YOLO V3

The YOLOv3 method is suggested in this work for human movement detection because of its benefits of high accuracy and cheap computing complexity [37]. The human movement detection technique is based on the YOLOv3 algorithm's network structure, and the specific detection procedure is as follows:

1. **Image Pre-processing:** Image pre-processing methods are used to handle the training set data for human movement detection. To guarantee uniform input across the dataset, the photos are transformed by scaling, normalization, and other processes. The training network then receives input from the pre-processed pictures.
2. **Darknet-53 Feature Extraction:** The pre-processed photos are sent across the Darknet-53 network to extract features. Convolutional neural network Darknet-53 collects distinct characteristics from the input photos at various levels.
3. **First Feature Extraction:** The output of Darknet-53's 77th layer is used as the first feature to be extracted. Convolution and down sampling (sampling) are then used for this feature to minimize its spatial dimensions while preserving crucial data.
4. **Second Feature Extraction:** Extraction of the second feature in Darknet-53 involves combining (splicing) the outputs of the 83rd and 61st layers. This second feature also passes via convolution and down sampling, much like the first feature.

5. **Third Feature Extraction:** In Darknet-53, the third feature is created by combining the outputs of the 93rd and 36th layers. This feature does not undergo any more down sampling or convolution.
6. **Training using the YOLO Layer:** The YOLO (You Only Look Once) layer is supplied with the three retrieved characteristics for training. The YOLO layer is in charge of spotting items in the input photos, including people moving about. The features and related ground truth annotations are used to train the network. The network is trained repeatedly until it performs at the desired level.
7. **Final Weight Model:** The network creates the final weight model when training is finished, which incorporates the learned parameters and is capable of recognizing human movement.
8. **Testing:** The test set photos are fed into the same network architecture to recognize the human movement in fresh images. The detecting technique makes use of the weight model that was previously constructed. The network examines the test photographs and produces detection findings that show whether or not there is human movement present in the pictures.

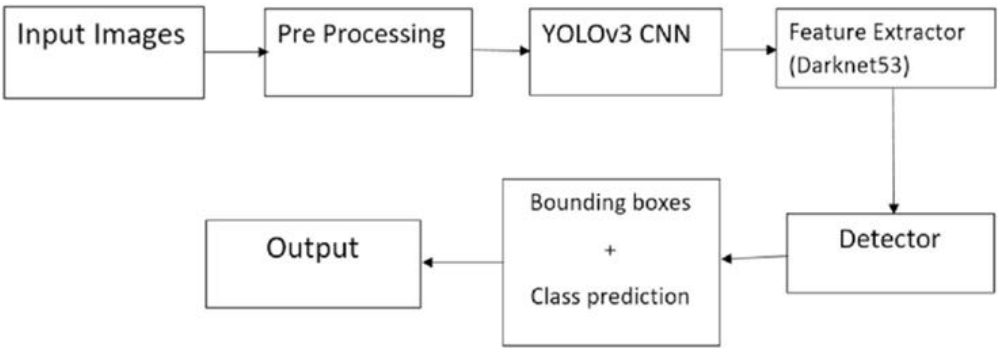

Fig 4.10: YOLO v3 Movement detection process

4.3.3 Real-Life Example of YOLO V3

For a better understanding of the entire process, a real-life video is used as an example.

Original Video:

A snapshot of a frame input into the algorithm

Fig 4.11: Input for the OpenCV Algorithm

YOLO V3 Algorithm Output:

Here the moving part of the frame is highlighted as a green contour.

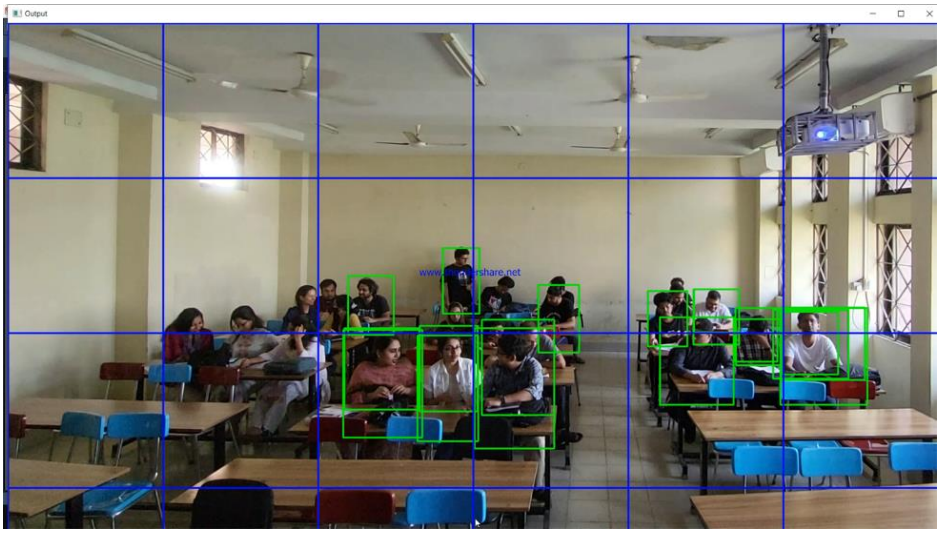

Fig 4.12: Output the YOLO V3 Algorithm

4.3.4 Issues with the YOLO V3 Algorithm process

The primary issue encountered with YOLO V3, a popular object detection algorithm, was its high processing demand. The algorithm took considerable time to process each frame, making it unsuitable for real-time applications. Additionally, the low quality of the camera and the significant distance between the camera and the occupants posed challenges for accurate human detection. The algorithm struggled to detect all individuals in such scenarios. Moreover, YOLO V3 had difficulty recognizing individuals who were not facing forward, limiting its effectiveness in identifying occupants who were not facing the camera directly. Addressing these limitations requires improvements in processing efficiency, camera quality, and robustness in recognizing individuals from various orientations.

4.3.5 Benefits of using the YOLO V3 Algorithm instead of other ones

Compared to other algorithms and processes, YOLO V3 works differently. Whereas other algorithms just detect only movement calculating the difference between the first or the previous and the current frame, YOLO V3 uses Coco’s dataset to identify the object which we need (in this case it was humans). Using YOLO V3 we were able to detect only Humans in the classroom. Just like the OpenCV algorithm, there were a lot of moving objects in the classroom but where OpenCV detected all the moving objects, YOLO V3 only detected human beings here. So, we can conclude that YOLO V3 will be the best of the compared algorithms to detect humans in a classroom environment.

Chapter 5

Our Model Setup: Hardware & Software Configuration

5.1 Hardware setup

We wanted to build a complete physical setup to implement in real classrooms. However, due to international chip shortage leading to heightened price of Raspberry Pi modules we couldn't do so. Despite the unavailability of a complete physical system, the research outcomes remain valuable in terms of conceptual design, theoretical analysis, and the formulation of recommendations for future implementations. The proposed hardware configuration serves as a blueprint for building an intelligent classroom system, and its specification provides a foundation for further exploration and development in subsequent research endeavours. Although the physical realisation was not achieved within the scope of this research, the insights gained and the proposed hardware configuration will contribute to the advancement of intelligent classroom technologies and serve as a valuable resource for future implementation efforts. The proposed hardware model consists of:

1. **Raspberry Pi:** The main control unit of the system will be a Raspberry Pi model 3B single-board computer. It will give the processing capacity and connectivity needed to run the system.
2. **Pi Camera:** The system will include a Raspberry Pi-compatible camera module (Raspberry Pi Camera Module V2) for taking photos and video footage of the classroom setting. It will be possible to analyse the presence of occupants using image processing and occupancy detection techniques.
3. **ESP32:** The ESP32 MCU, a programmable microcontroller with built-in Wi-Fi and Bluetooth capabilities, will be used in the system. It will act as a bridge between the intelligent system and the electrical appliances, allowing for automated control based on occupancy identification.
4. **Relay Modules:** These modules function as electronic switches, allowing the intelligent classroom system to control electrical appliances such as lights and fans.

The relays allow the system to turn on and off these appliances based on occupancy sensing.

5. **Micro SD Card:** A storage card with enhanced memory capacity for storing data, software, and intelligent classroom system configurations.
6. **Power Supplies:** To ensure steady and reliable operation, the ESP32 microcontroller and relay modules require dedicated power supply. These power sources supply the appropriate voltage and current to suit the needs of the various components.
7. **Power Cables:** Cables and connectors are required for connecting power supply to the ESP32 and relay modules, guaranteeing proper power distribution.
8. **Display:** A display device, such as an LCD or LED screen, can be integrated into the system to provide users with visual feedback and status information. It improves the user experience and makes monitoring and controlling the intelligent classroom system easier.
9. **Stand:** A solid stand or mounting solution is required to firmly position the Raspberry Pi, Pi camera, and other components within the classroom environment. It provides stability and excellent location for occupancy detection.

The intelligent classroom system is built on the combination of various hardware components, which allows it to detect people, manage appliances, and improve energy efficiency. The hardware configuration described here will be used as a guide for the actual implementation of the intelligent classroom system in future research and practical applications.

5.2 Raspberry Pi configurations

The Raspberry Pi Model 3B is a popular single-board computer with a variety of features and capabilities that are appropriate for a variety of applications, including the building of an intelligent classroom system. The Raspberry Pi Model 3B provides an appropriate blend of processing power, memory capacity, and networking options for establishing the intelligent classroom system. Because of its adaptability, community support, and low cost, it is a popular platform for educational and IoT applications. The Raspberry Pi Model 3B's configuration and technical features are as follows:

1. **Processor:** The Model 3B has a 1.2GHz 64-bit quad-core ARM Cortex-A53 processor, which provides enough computing capacity for data processing and system control.
2. **Memory:** It contains 1GB of LPDDR2 SDRAM, which allows for smooth multitasking and efficient data-intensive task processing.
3. **Storage:** The Model 3B lacks built-in storage but accepts microSD cards for capacity expansion. There is a micro-SD card slot for installing the operating system and storing data.
4. **Connectivity:** It includes built-in Wi-Fi (802.11n) and Bluetooth 4.2, allowing for wireless data transfer, remote control, and networking with other devices.
5. **USB Ports:** The Model 3B has four USB 2.0 ports for connecting external devices such as cameras, keyboards, and storage drives.
6. **Video Output:** It supports HDMI output, which allows for high-definition video display on compatible monitors or TVs. There is also a composite video jack for connecting to analog screens.
7. **GPIO Pins**: The Model 3B has a 40-pin GPIO (General Purpose Input/Output) header for connecting additional components and expansion boards to expand the system's capabilities.
8. **Power source:** A 5V micro-USB power source is required for operation, providing enough power to support the system's performance and attached peripherals.
9. **Dimensions:** The Model 3B has a small form factor, measuring 85mm x 56mm x 17mm, making it ideal for a variety of space-constrained applications.

5.3 Pi Camera configuration

The Raspberry Pi Camera Module V2 is a small camera module designed for use with Raspberry Pi devices. It offers a simple and effective method for recording images and videos in a variety of applications, including the intelligent classroom system. With its resolution and sensor characteristics, the Raspberry Pi Camera Module V2 provides high-quality image capabilities. Because of its seamless interaction with Raspberry Pi boards, it is an excellent choice for recording visual data in the intelligent classroom system. Its small size and simple interface allow for the development of novel applications as well as real-time

monitoring and analysis of the classroom environment. The Raspberry Pi Camera Module V2's configuration and technical features are as follows:

1. **Resolution:** The camera module can capture still photographs with up to 8 megapixels (3280 x 2464 pixels) and record videos at a maximum quality of 1080p at 30 frames per second.
2. **Sensor:** It has a Sony IMX219 1/4-inch CMOS sensor, which provides better image quality, sensitivity, and low-light performance.
3. **Lens:** The camera module has a fixed-focus lens with a diagonal field of view of approximately 62.2 degrees.
4. **Connectivity:** It connects to the Raspberry Pi board directly through a specific CSI (Camera Serial Interface) connector, ensuring rapid and reliable data transfer.
5. **Control:** The camera module is managed by the Raspberry Pi's camera interface, which allows for simple configuration and changes.
6. **Power Source:** It gets power straight from the Raspberry Pi board, removing the need for an external power supply.
7. **Dimensions:** With dimensions of roughly 25mm x 23mm x 9mm, the Raspberry Pi Camera Module V2 is suited for integration into small systems.
8. **Software Support:** The camera module is supported by the official Raspberry Pi operating system (Raspbian) as well as a variety of third-party software libraries, allowing for a wide range of image and video processing possibilities.

5.4 ESP 32 configuration

The ESP32 is a powerful microcontroller module that is frequently utilised in embedded systems and Internet of Things applications. It has advanced functionality and networking possibilities, making it ideal for deploying the intelligent classroom system. The ESP32 module provides a flexible and feature-rich foundation for developing the intelligent classroom system. Its powerful microprocessor, wireless networking possibilities, plenty of GPIO pins, and support for many interfaces allow for simple interaction with other components and sensors. Because of the ESP32's wide programming capabilities and compatibility with common development environments, it is suitable for both novice and professional developers. The ESP32's configuration and technical characteristics are as follows:

1. **Microcontroller:** The ESP32 module is based on the ESP32 microcontroller, which has a dual-core Tensilica LX6 CPU that runs at up to 240MHz. It provides a good blend of computing power and energy efficiency.
2. **Memory:** It has many memory configurations, including flash memory and RAM options. The capacity of flash memory can range from 4MB to 16MB, while RAM capacity can range from 520KB to 8MB.
3. **Wireless connectivity:** Wireless connectivity is provided by the ESP32 module, which supports Wi-Fi 802.11 b/g/n, enabling for seamless integration into wireless networks. Bluetooth 4.2 and BLE (Bluetooth Low Energy) are also included for effective connectivity with other devices.
4. **GPIO Pins:** The module includes an adequate number of GPIO (General Purpose Input/Output) pins, which normally range from 15 to 36 pins depending on the model. These pins can be used for a variety of functions, including digital input/output, analogue input, PWM (Pulse Width Modulation), I2C, SPI, and more.
5. **Interfaces:** It has a variety of interfaces for connecting external devices and sensors, such as I2C (Inter-Integrated Circuit), SPI (Serial Peripheral Interface), UART (Universal Asynchronous Receiver-Transmitter), and others. These interfaces allow for simple integration with a wide variety of peripherals.
6. **Power Supply:** The ESP32 module requires a conventional 3.3V power supply voltage and enables low-power modes for energy-efficient operation. It can be charged through USB or by an external power source.
7. **Programming:** Programming Languages and Development Environments: The ESP32 may be programmed using a variety of programming languages and development environments, including the Arduino IDE, Micro Python, and ESP-IDF (ESP32 IoT Development Framework).
8. **Dimensions:** The module is available in a variety of form factors, including surface-mount modules and development boards of variable sizes. These solutions give versatility for a variety of project requirements.

5.5 Installation of Raspbian

To install Raspbian on a Raspberry Pi, we can follow these steps:

1. **Download Raspbian:** Go to the official Raspberry Pi website (www.raspberrypi.org) and navigate to the "Downloads" section. Select the latest version of Raspbian (now known as Raspberry Pi OS) and choose the recommended version for most users, usually the "Raspberry Pi OS with desktop and recommended software" option.
2. **Prepare the microSD card:** Insert the microSD card into your computer and use a disk imaging tool like Etcher (www.balena.io/etcher) to write the Raspbian image to the microSD card. Select the downloaded Raspbian image file and the target microSD card, and then click "Flash" to start the process. This will create a bootable microSD card with Raspbian.
3. **Configure Wi-Fi (optional):** To connect your Raspberry Pi to a Wi-Fi network, create a file named "wpa_supplicant.conf" in the microSD card's root directory. Open the file and replace "SSID" with your network name and "PASSWORD" with your Wi-Fi password with the following content:

```
country=US
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1

network={
   ssid="SSID"
   psk="PASSWORD"
}
```

4. Enable SSH (optional): To enable SSH access to your Raspberry Pi, create a file named "ssh" (no extension) on the microSD card's root directory. This will enable SSH by default on the Raspberry Pi.
5. Insert the microSD card as follows: Insert the microSD card from your computer into the Raspberry Pi's microSD card port.
6. Start the Raspberry Pi: Using a micro USB cable, connect the Raspberry Pi to a power source. The Raspberry Pi will reboot and begin the installation procedure.
7. Initial configuration: To complete the initial setup of Raspbian, follow the on-screen prompts, including selecting your desired language, keyboard layout, and changing the default password for the "pi" user.

5.6 Setting up the camera

To configure the Raspberry Pi Camera Module V2, we can follow these steps:

1. **Hardware setup:** Before connecting the camera module, make sure your Raspberry Pi is turned off. Locate the Raspberry Pi board's camera connector (it's a little, flat ribbon cable connector near the HDMI port). Lift the camera connector's plastic clip gently, enter the camera cable with the metal contacts facing away from the HDMI port, and then slide the plastic clip back down to secure the cable.
2. **Enable the camera interface:** Log in to the Raspbian operating system and turn on your Raspberry Pi. Launch the command line interface or terminal.
3. **Run the configuration tool:** To launch the Raspberry Pi Configuration tool, type the following command into the terminal: raspi-config sudo
4. **Navigate to camera settings:** Press Enter after navigating to "Interfacing Options" with the arrow keys. Select "Camera" from the "Interfacing Options" menu and click Enter.
5. **Enable the camera:** To enable the camera interface, select "Yes" on the camera interface screen and then press Enter. A notice will appear verifying that the camera interface has been activated.
6. **Reboot the Raspberry Pi:** To restart the Raspberry Pi, use Tab to select "Finish" and Enter. You will be asked to restart the Raspberry Pi. Select "Yes" to restart the computer and implement the modifications.
7. **Test the camera:** After rebooting the Raspberry Pi, use the command-line program "raspistill" to test the camera. Return to the terminal and type the following command: raspistill -o test.jpg. This command uses the camera to capture an image and saves it as "test.jpg" in the current directory. If the camera is functioning properly, you should be able to see a preview of the camera feed, and the captured image will be saved.

Chapter 6

Result Analysis

6.1 Comparison amongst the three algorithms

We have used three algorithms in our proposed model. They are compared to find the best and most efficient algorithms to detect human movement in a classroom environment. The algorithms are:

1. Background subtraction (MOG 2 algorithm)
2. Computer Vision (using different algorithms)
3. You Only Look Once (YOLO V3)

We have analyzed all three algorithms and found that the first two, viz., MOG2 and Computer Vision, although promising, do not show the anticipated results when it comes to motion detection in a classroom. Motion detection is very important in our proposed model, and in this case, YOLO v3 excelled very well. For this reason, we have chosen YOLO v3 over the other two algorithms and prepared a well-documented study of the results that we have found.

We are therefore attaching a comparison table for all three approaches we proposed here:

Table 6.1: Comparison of Three Models

Methods	**MOG2**	**Computer Vision**	**YOLO v3**
Proper detection	No	No	Yes
Detection of the only move-able object	Yes	Yes	Yes
No. Of dependency on the fixed frame	No	Yes	Yes

Can comply with unexpected changes	No	Yes	Yes
Stability	No	No	Yes
Period control	No	No	Yes

We have found that YOLO v3 showed the most promising results. In the case of our proposed model, as mentioned in Chapters 3 and 4, the whole input video is divided into nine sections, which provides us with easy detection of motion in that specified area. A Python code was also developed to tabulate the motion detection of a certain section. It tabulates when motion has started and when it has stopped. Not only that, but the height and width of the screen where motion detection has been observed are also recorded. This information is simultaneously saved in a CSV file. Then another Python script is run based on the data in the CSV file. Using the starting and ending times of motion, we can detect for how long movement has occurred, and by using the width and height, we can detect in which region of the room movement has occurred. Lastly, by using the above two pieces of information, we regulate the electrical appliances accordingly.

Here we are attaching a part of all the data which we used for motion and region detection. The first table depicts the region where motion has occurred and the second one depicts where there is no motion.

Table depicting movement:

Table 6.2: Movement detected

Start Time	End Time	X	Y	Width	Height	Type
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.165382PM	4	-41	815	891	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.165382PM	34	-24	880	849	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.165382PM	38	-25	913	861	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.166379PM	-21	-20	862	904	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.166379PM	-15	-20	878	920	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.166379PM	16	38	908	796	Movement

2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.166379PM	6	21	963	836	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.166379PM	21	29	1235	835	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.166379PM	646	526	117	255	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.166379PM	622	508	161	298	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.166379PM	472	542	215	297	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.166379PM	572	544	122	298	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.166379PM	522	540	175	300	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.166379PM	652	531	118	292	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.166379PM	622	518	159	315	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.166379PM	514	569	181	281	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.166379PM	536	476	55	74	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.166379PM	729	478	35	61	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.166379PM	532	475	58	111	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.166379PM	533	492	59	106	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.166379PM	816	503	52	91	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.166379PM	865	519	66	64	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.166379PM	751	507	67	133	Movement
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.167377PM	938	540	63	57	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.693004PM	4	-41	815	891	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.693004PM	33	-24	881	849	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694001PM	38	-25	913	860	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694001PM	-21	-20	862	904	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694001PM	-15	-20	878	920	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694001PM	16	38	908	796	Movement

2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694001PM	6	21	963	836	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694001PM	21	29	1235	835	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694001PM	646	526	117	255	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694001PM	622	508	161	297	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694001PM	472	542	215	297	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694001PM	572	544	122	298	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694001PM	522	540	175	300	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694001PM	652	531	118	292	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694001PM	622	518	159	315	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694001PM	514	569	181	281	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694001PM	536	476	55	74	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694001PM	729	478	35	61	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694001PM	532	475	58	111	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694998PM	533	492	59	106	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694998PM	816	503	52	91	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694998PM	865	519	66	64	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694998PM	751	507	67	133	Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694998PM	938	540	63	57	Movement
2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.103905PM	41	-9	916	831	Movement
2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.103905PM	-24	-17	863	903	Movement
2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.103905PM	-21	-9	888	904	Movement
2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.103905PM	4	35	928	804	Movement
2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.103905PM	6	22	971	840	Movement
2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.103905PM	23	33	1252	829	Movement

2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.103905PM	645	526	119	255	Movement
2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.103905PM	622	510	162	294	Movement
2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.103905PM	472	544	218	295	Movement
2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.103905PM	571	545	124	298	Movement
2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.103905PM	523	541	175	299	Movement
2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.104903PM	650	530	119	295	Movement
2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.104903PM	623	519	157	315	Movement
2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.104903PM	515	569	179	281	Movement
2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.104903PM	533	476	60	73	Movement
2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.104903PM	523	477	54	112	Movement
2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.104903PM	530	475	62	112	Movement
2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.104903PM	533	491	60	108	Movement
2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.104903PM	815	503	54	94	Movement
2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.104903PM	866	520	63	63	Movement
2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.104903PM	939	540	62	58	Movement
2023-06-17 13:41:43.508823PM	2023-06-17 13:41:43.508823PM	-19	-18	825	912	Movement
2023-06-17 13:41:43.508823PM	2023-06-17 13:41:43.508823PM	11	42	916	794	Movement
2023-06-17 13:41:43.508823PM	2023-06-17 13:41:43.508823PM	9	20	963	847	Movement
2023-06-17 13:41:43.508823PM	2023-06-17 13:41:43.508823PM	644	530	119	246	Movement
2023-06-17 13:41:43.508823PM	2023-06-17 13:41:43.508823PM	618	510	165	294	Movement
2023-06-17 13:41:43.508823PM	2023-06-17 13:41:43.508823PM	468	541	224	306	Movement
2023-06-17 13:41:43.508823PM	2023-06-17 13:41:43.508823PM	518	541	182	304	Movement
2023-06-17 13:41:43.508823PM	2023-06-17 13:41:43.508823PM	650	530	119	299	Movement
2023-06-17 13:41:43.508823PM	2023-06-17 13:41:43.508823PM	623	519	157	319	Movement

2023-06-17 13:41:43.508823PM	2023-06-17 13:41:43.508823PM	513	568	181	284	Movement
2023-06-17 13:41:43.508823PM	2023-06-17 13:41:43.508823PM	530	477	63	75	Movement
2023-06-17 13:41:43.508823PM	2023-06-17 13:41:43.508823PM	528	477	64	108	Movement
2023-06-17 13:41:43.508823PM	2023-06-17 13:41:43.508823PM	532	490	59	110	Movement
2023-06-17 13:41:43.508823PM	2023-06-17 13:41:43.508823PM	813	504	53	91	Movement
2023-06-17 13:41:43.508823PM	2023-06-17 13:41:43.508823PM	866	522	63	60	Movement
2023-06-17 13:41:43.508823PM	2023-06-17 13:41:43.508823PM	940	541	62	59	Movement
2023-06-17 13:41:43.508823PM	2023-06-17 13:41:43.508823PM	747	519	69	141	Movement
2023-06-17 13:41:44.022451PM	2023-06-17 13:41:44.022451PM	-23	-34	802	944	Movement
2023-06-17 13:41:44.022451PM	2023-06-17 13:41:44.023448PM	-15	-25	868	943	Movement
2023-06-17 13:41:44.022451PM	2023-06-17 13:41:44.023448PM	-11	35	956	806	Movement
2023-06-17 13:41:44.022451PM	2023-06-17 13:41:44.023448PM	9	0	953	887	Movement
2023-06-17 13:41:44.022451PM	2023-06-17 13:41:44.023448PM	643	533	120	234	Movement
2023-06-17 13:41:44.022451PM	2023-06-17 13:41:44.023448PM	468	545	223	299	Movement
2023-06-17 13:41:44.022451PM	2023-06-17 13:41:44.023448PM	522	539	179	307	Movement
2023-06-17 13:41:44.022451PM	2023-06-17 13:41:44.023448PM	650	536	122	293	Movement
2023-06-17 13:41:44.022451PM	2023-06-17 13:41:44.023448PM	623	524	158	318	Movement
2023-06-17 13:41:44.022451PM	2023-06-17 13:41:44.023448PM	351	638	138	168	Movement

Table depicting no movement:

Table 6.3: No movement detected

Start Time	End Time	X	Y	Width	Height	Type
2023-06-17 13:41:42.165382PM	2023-06-17 13:41:42.167377PM					No Movement
2023-06-17 13:41:42.693004PM	2023-06-17 13:41:42.694998PM					No Movement

2023-06-17 13:41:43.103905PM	2023-06-17 13:41:43.104903PM	No Movement
2023-06-17 13:41:43.508823PM	2023-06-17 13:41:43.508823PM	No Movement
2023-06-17 13:41:44.022451PM	2023-06-17 13:41:44.024445PM	No Movement
2023-06-17 13:41:44.500173PM	2023-06-17 13:41:44.500173PM	No Movement
2023-06-17 13:41:44.938003PM	2023-06-17 13:41:44.939000PM	No Movement
2023-06-17 13:41:45.422708PM	2023-06-17 13:41:45.423705PM	No Movement
2023-06-17 13:41:45.834633PM	2023-06-17 13:41:45.834633PM	No Movement
2023-06-17 13:41:46.322810PM	2023-06-17 13:41:46.323808PM	No Movement
2023-06-17 13:41:46.735707PM	2023-06-17 13:41:46.736704PM	No Movement
2023-06-17 13:41:47.145613PM	2023-06-17 13:41:47.145613PM	No Movement
2023-06-17 13:41:47.554518PM	2023-06-17 13:41:47.554518PM	No Movement
2023-06-17 13:41:47.966417PM	2023-06-17 13:41:47.966417PM	No Movement
2023-06-17 13:41:48.381814PM	2023-06-17 13:41:48.381814PM	No Movement
2023-06-17 13:41:48.794742PM	2023-06-17 13:41:48.795735PM	No Movement
2023-06-17 13:41:49.206610PM	2023-06-17 13:41:49.207608PM	No Movement
2023-06-17 13:41:49.621502PM	2023-06-17 13:41:49.621502PM	No Movement
2023-06-17 13:41:50.036392PM	2023-06-17 13:41:50.037390PM	No Movement
2023-06-17 13:41:50.455299PM	2023-06-17 13:41:50.455299PM	No Movement
2023-06-17 13:41:50.865177PM	2023-06-17 13:41:50.865177PM	No Movement
2023-06-17 13:41:51.410719PM	2023-06-17 13:41:51.411716PM	No Movement
2023-06-17 13:41:51.955263PM	2023-06-17 13:41:51.956261PM	No Movement
2023-06-17 13:41:52.749237PM	2023-06-17 13:41:52.750235PM	No Movement
2023-06-17 13:41:53.291791PM	2023-06-17 13:41:53.292803PM	No Movement
2023-06-17 13:41:53.838327PM	2023-06-17 13:41:53.839324PM	No Movement

2023-06-17 13:41:54.282140PM	2023-06-17 13:41:54.283138PM	No Movement
2023-06-17 13:41:54.729942PM	2023-06-17 13:41:54.729942PM	No Movement
2023-06-17 13:41:55.154807PM	2023-06-17 13:41:55.154807PM	No Movement
2023-06-17 13:41:55.576680PM	2023-06-17 13:41:55.576680PM	No Movement
2023-06-17 13:41:55.988579PM	2023-06-17 13:41:55.989576PM	No Movement
2023-06-17 13:41:56.406462PM	2023-06-17 13:41:56.407460PM	No Movement
2023-06-17 13:41:56.824345PM	2023-06-17 13:41:56.825342PM	No Movement
2023-06-17 13:41:57.249209PM	2023-06-17 13:41:57.250206PM	No Movement
2023-06-17 13:41:57.669087PM	2023-06-17 13:41:57.669087PM	No Movement
2023-06-17 13:41:58.091956PM	2023-06-17 13:41:58.091956PM	No Movement
2023-06-17 13:41:58.515850PM	2023-06-17 13:41:58.516848PM	No Movement
2023-06-17 13:41:58.937696PM	2023-06-17 13:41:58.937696PM	No Movement
2023-06-17 13:41:59.360602PM	2023-06-17 13:41:59.360602PM	No Movement
2023-06-17 13:41:59.791445PM	2023-06-17 13:41:59.791445PM	No Movement
2023-06-17 13:42:00.212290PM	2023-06-17 13:42:00.212290PM	No Movement
2023-06-17 13:42:00.635159PM	2023-06-17 13:42:00.636156PM	No Movement
2023-06-17 13:42:01.056034PM	2023-06-17 13:42:01.056034PM	No Movement
2023-06-17 13:42:01.478904PM	2023-06-17 13:42:01.478904PM	No Movement
2023-06-17 13:42:01.902770PM	2023-06-17 13:42:01.902770PM	No Movement
2023-06-17 13:42:02.327635PM	2023-06-17 13:42:02.328633PM	No Movement
2023-06-17 13:42:02.765464PM	2023-06-17 13:42:02.765464PM	No Movement
2023-06-17 13:42:03.189332PM	2023-06-17 13:42:03.189332PM	No Movement
2023-06-17 13:42:33.775579PM	2023-06-17 13:42:33.775579PM	No Movement
2023-06-17 13:42:34.211414PM	2023-06-17 13:42:34.211414PM	No Movement

2023-06-17 13:42:34.638273PM	2023-06-17 13:42:34.638273PM	No Movement
2023-06-17 13:42:35.071116PM	2023-06-17 13:42:35.071116PM	No Movement
2023-06-17 13:42:35.508975PM	2023-06-17 13:42:35.508975PM	No Movement
2023-06-17 13:42:35.945779PM	2023-06-17 13:42:35.945779PM	No Movement
2023-06-17 13:42:36.382611PM	2023-06-17 13:42:36.382611PM	No Movement
2023-06-17 13:42:36.813459PM	2023-06-17 13:42:36.813459PM	No Movement
2023-06-17 13:42:37.245305PM	2023-06-17 13:42:37.246302PM	No Movement
2023-06-17 13:42:37.682137PM	2023-06-17 13:42:37.682137PM	No Movement
2023-06-17 13:42:38.121961PM	2023-06-17 13:42:38.122959PM	No Movement
2023-06-17 13:42:38.564778PM	2023-06-17 13:42:38.565775PM	No Movement
2023-06-17 13:42:39.006597PM	2023-06-17 13:42:39.006597PM	No Movement
2023-06-17 13:42:39.445425PM	2023-06-17 13:42:39.445425PM	No Movement
2023-06-17 13:42:39.888241PM	2023-06-17 13:42:39.888241PM	No Movement
2023-06-17 13:42:40.342028PM	2023-06-17 13:42:40.343025PM	No Movement
2023-06-17 13:42:40.779857PM	2023-06-17 13:42:40.780854PM	No Movement
2023-06-17 13:42:41.214695PM	2023-06-17 13:42:41.215692PM	No Movement
2023-06-17 13:42:41.649534PM	2023-06-17 13:42:41.650531PM	No Movement
2023-06-17 13:42:42.091352PM	2023-06-17 13:42:42.091352PM	No Movement
2023-06-17 13:42:42.525224PM	2023-06-17 13:42:42.526217PM	No Movement

We wanted to calculate the time for which the electrical appliances will be turned off to limit the misuse of electrical energy. A simulation has been done using a video feed of 549 seconds of running time. It has been found out that the total non-movement duration is 71 seconds.

The table is displayed below:

Table 6.4: Total Non-movement duration

Zone	Total Non-Movement Duration (seconds)
Zone 1	0.061925
Zone 2	0.001
Zone 3	0
Zone 4	33.413895
Zone 5	35.999306
Zone 6	0
Zone 7	1.167021
Zone 8	0.694534
Zone 9	0
Total=	71.337681

Using the table, we calculated the total non-movement time with respect to 7.5 hours or (7.5 × 3600) seconds. The calculated value was 3491.8 seconds which is almost equal to 58.19 minutes or simply 0.969 hour.

So, the total running time of the electrical appliances when the proposed model is adopted will be (7.5 – 0.969) hours = 6.53 hours.

6.2 A Comparative analysis of three scenarios

A comparison of three scenarios is displayed below:

Table 6.5: Comparison table

Methods	**With system inactive**	**With system active but management is off**	**With system active and management is on**
Daily active class time (in hours)	9	7.5	6.53
Per day energy consumption for one classroom	71550	59625	51913.5
Per day energy consumption (unit) for one classroom	71.55	59.625	51.9135
Energy consumption for a year (200 days) for one classroom	14310	11925	10382.7
Electricity bill for a year for one classroom	163992.6	136660.5	118985.74
Per day savings in taka for one classroom	0	136.66	225.03
Per year savings in taka for one classroom	0	27332.1	45006.85
Per day savings in taka for the whole institution	0	13666	22503
Per year savings in taka for the whole institution	0	2733210	4500685

Here we are attaching the table which contains all the data of yearly Energy Consumption, Savings and Bills according to increased number of classrooms:

Table 6.6: Yearly consumption, saving and bill

Classroom number	Consumption (Unit)	Saving (in Taka)	Bill (in Taka)
1	10382.7	45006.85	118985.74
20	207654	900137	2379714.8
40	415308	1800274	4759429.6
60	622962	2700411	7139144.4
80	830616	3600548	9518859.2
100	1038270	4500685	11898574

6.3 Necessary Graphs

Consumption (Unit) vs Classroom Number graph

We implement the system in one classroom and increase the number of classrooms up to one hundred and collect the energy consumption in unit for a year according to Table 6.6.

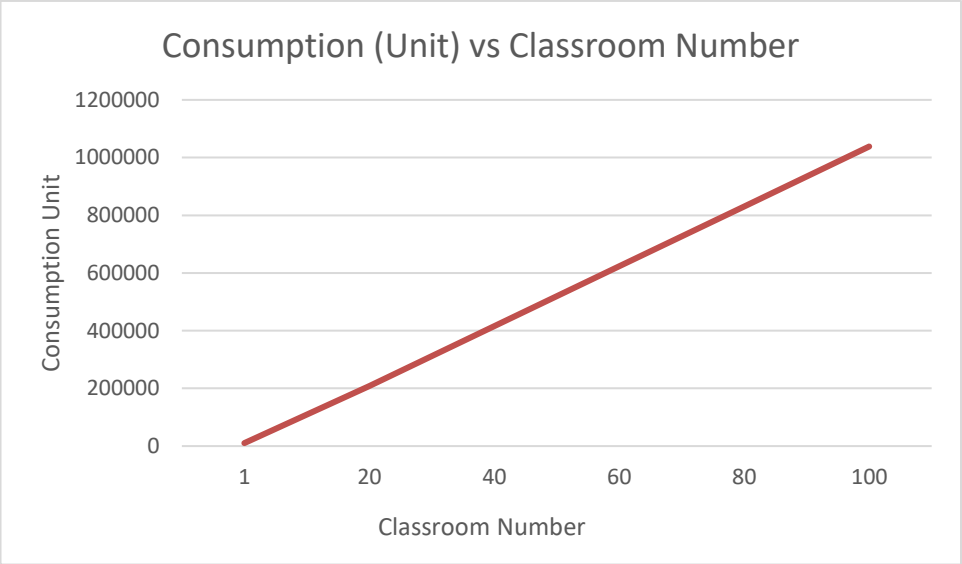

Figure 6.1: Unit vs Classroom Number graph

As observed in Figure 6.1, the energy consumption increases proportionally with increased classroom numbers. In the figure, the Y-axis represents energy consumption in Unit and X-axis represents the classroom number

So, after implementation of the system in one classroom and increase the number of classrooms up to one hundred that the institution has, we observe that the graph is linear and the unit consumption is proportional to the increasing number of classrooms.

Savings (Taka) vs Classroom Number graph

We implement the system in one classroom and increase the number of classrooms up to one hundred and collect the savings in taka for a year according to Table 6.6.

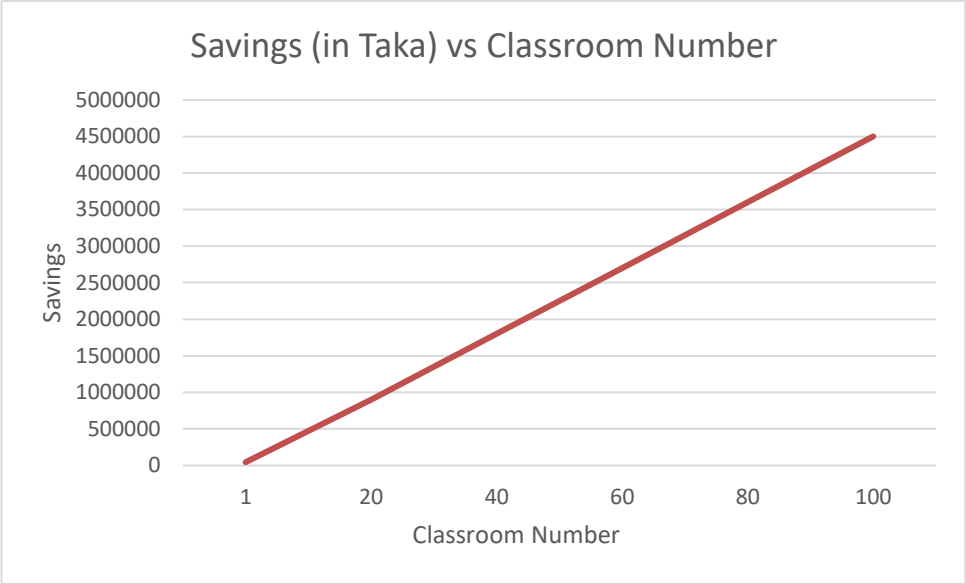

Figure 6.2: Savings vs Classroom Number graph

As observed in Figure 6.2, the savings increases proportionally with increased classroom numbers. In the figure, the Y-axis represents savings in Taka and X-axis represents the classroom number

So, after implementation of the system in one classroom and increase the number of classrooms up to one hundred that the institution has, we observe that the graph is linear and the savings is proportional to the increasing number of classrooms.

Bill (Taka) vs Classroom Number graph

We implement the system in one classroom and increase the number of classrooms up to one hundred and collect the bill in taka for a year according to Table 6.6.

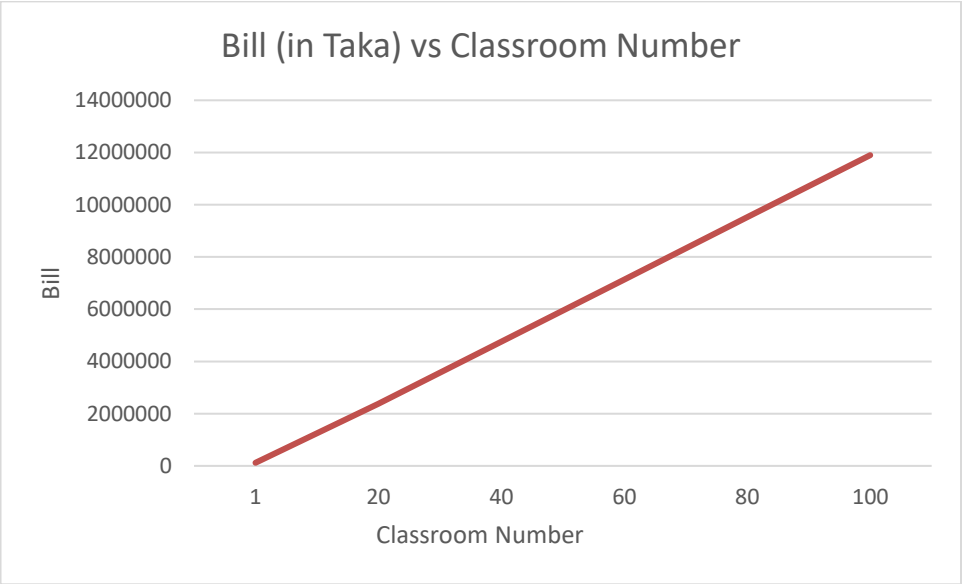

Figure 6.3: Bill vs Classroom Number graph

As observed in Figure 6.3, the savings increases proportionally with increased classroom numbers. In the figure, the Y-axis represents bill in Taka and X-axis represents the classroom number

So, after implementation of the system in one classroom and increase the number of classrooms up to one hundred that the institution has, we observe that the graph is linear and the bill is proportional to the increasing number of classrooms.

Bill & Savings (in Taka) vs Consumption (Unit) graph

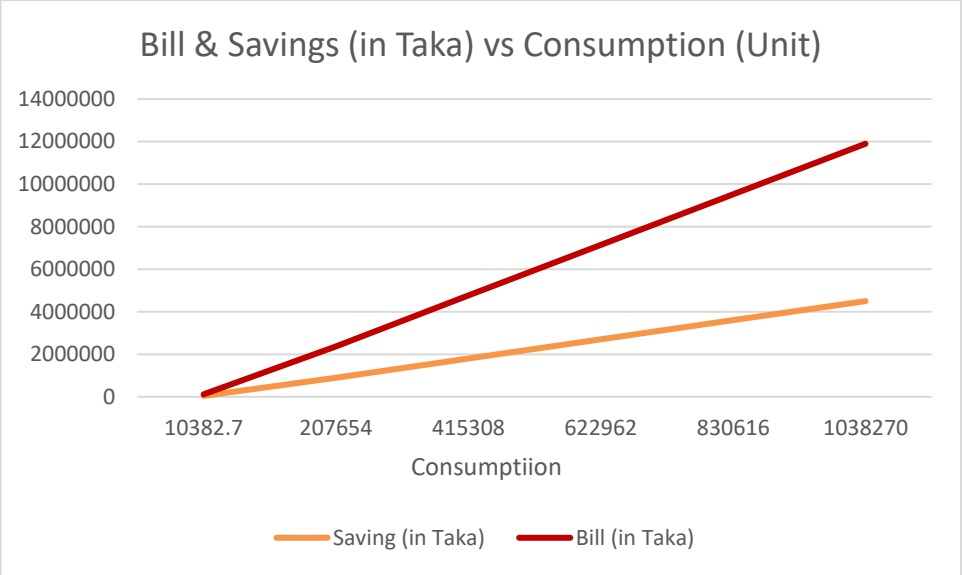

Figure 6.4: Bill & Savings vs Consumption graph

As observed in Figure 6.4, the savings increases proportionally with increased classroom numbers. In the figure, the Y-axis represents bill and savings in Taka and X-axis represents the unit consumption.

6.4 Cost Breakdown

A table is given here showing the approximate cost of one device for one classroom for applying the system:

Table 6.7: Approximate cost

Components	**Price (BDT)**
Raspberry pi 3B (1GB)	18,990
Pi cam	4,690
Sd card	500
Power supply	600
Esp32	700
Relay module	240
Miscellaneous	100
Total	25,820

As a device can run for infinite time and we can see the cost is very low compared to how much electric bill is charged every month for one classroom, so this system is very cost-friendly and user-friendly as we were expecting.

6.5 Discussion of Result

In our proposed thesis, we have explored and compared three existing algorithms to detect motion in a classroom. The result of one algorithm widely differs from that of the other. After proper scrutiny we can easily state that YOLOv3 gives satisfactory result when it comes to motion detection. The YOLOv3 detected just humans in the classroom whereas, the OpenCV algorithm recognized all moving objects in the classroom. This is quite problematic in our case as there are many moving objects in the classroom which do not require detection. And lastly, the MoG2 algorithm did not give satisfactory results. Hence, we can easily implement YOLOv3 to automate classrooms.

6.6 Fulfillment of the Objective and Expected Outcome

This study's research objectives were met with success, yielding substantial outcomes. The major goal was to create an effective and intelligent classroom system. To accomplish this, the researchers investigated and analyzed three occupant recognition models, taking into account their accuracy, dependability, and real-time monitoring capabilities. The most appropriate occupant detection model was chosen for implementation in the system after careful consideration.

Firstly, a completely automated smart classroom setting with occupancy-based energy consumption was established. The system accurately identifies and tracks occupants by incorporating the chosen occupancy detection model, allowing for dynamic modifications of power supply and effective energy management. This result directly addresses the goal of creating an intelligent classroom system.

Secondly, a comparison of occupant detection models for intelligent classroom systems was carried out. This research provides useful insights into the strengths and shortcomings of various models, allowing for informed decision-making for future implementations. By assessing and benchmarking occupant detection models in the context of classroom automation, it adds to the existing body of knowledge.

In addition, the research result comprises the development of a user-centric, economically viable, and adaptable system. The intelligent classroom solution designed prioritizes user experience and ease of use, guaranteeing that educators and students can engage with the system easily. Furthermore, the implementation strategy takes cost-effectiveness into account, making the solution accessible and practicable for educational institutions with various budgets. Because the system is configurable, it may be adapted to varied school settings and requirements.

Overall, the study objectives were met, resulting in the construction of an efficient and intelligent classroom system, as well as a comparative analysis of occupancy detection algorithms and the establishment of a user-centric and adaptable solution. These findings help to advance smart building technology in educational contexts, as well as providing practical insights for energy optimization and occupancy identification approaches.

Chapter 7

Conclusion & Future Work

7.1 Conclusion

In summary, the goal of our study was to develop an intelligent classroom that uses an image-processed occupant detection technique and is energy-efficient. We presented a solution that made use of image processing methods to track student mobility and activity in a classroom and automatically modify energy usage. Our study looked at and compared three distinct detection models to determine the best strategy that took into account a variety of uncontrolled variables in the classroom setting. We developed a reliable detection mechanism that successfully cut down on energy use after extensive testing. Beyond schools, the suggested system has the potential to be used in places like libraries, homes, banks, hospitals, labs, offices, and enterprises where inhabitants' presence and frequent appliance control are necessary. The system may be effortlessly linked with current security systems, obviating the need for extra wiring and making use of reasonably priced hardware Raspberry Pi. The expense of putting this system into place is kept to a minimum by making use of the current infrastructure. Additionally, it provides a wireless and adaptable solution to the energy consumption issue, making it a desirable and workable answer in the current environment. The objective of increasing knowledge about energy conservation is aligned with its cost and potential for energy savings. We may underline that our suggested approach helps to maintain a better learning environment in modern technology-dependent world. Reliable occupant recognition and effective appliance control are made possible by the integration of image processing and intelligent detection models. In conclusion, our energy-efficient intelligent classroom system's design and implementation provide a practical, affordable solution with a variety of uses. It has the ability to significantly reduce energy use in a variety of contexts by fostering energy conservation and awareness.

7.2 Future Works

Our main motive of the proposed model is to limit the misuse of electrical energy. Non renewable fuels like coals and gases are becoming more and more scare as time is

progressing. These fuels are necessary for the generation of electrical energy. By implementing our proposed model, excessive consumption of fuel can be limited. Now, if we look at a financial point of view, the generation and usage of electrical energy can cost a lot. This can also be prevented if we properly implement our proposed model.
Our primary objective was to implement this project in classrooms of educational institutions. We also aim to implement this model for offices, both governmental and non-governmental as these kinds of offices also misuse a huge amount of electrical energy.

Lastly, our proposed model is based on motion detection. We can also use this motion detection technique to completely automate classrooms which can limit cheating in exams as well as proper surveillance of classrooms.

Appendix A

Appendix A contains all the figures of hardware setup.

Figure A.1: Raspberry Pi Model

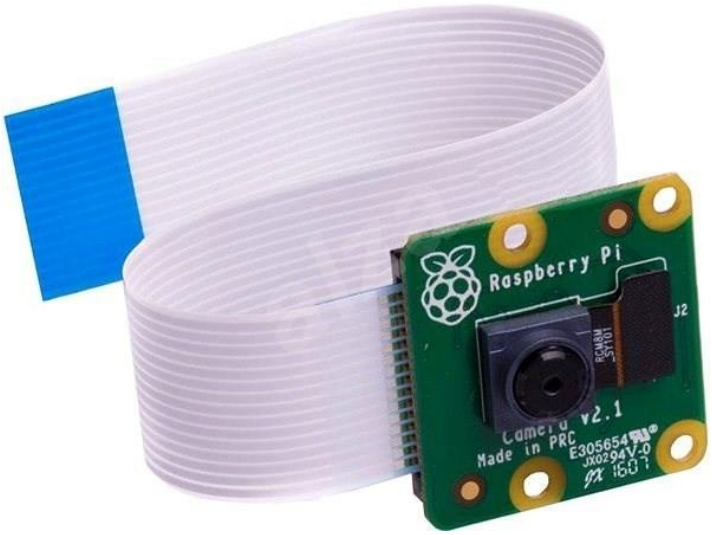

Figure A.2: Pi Cam 2.1

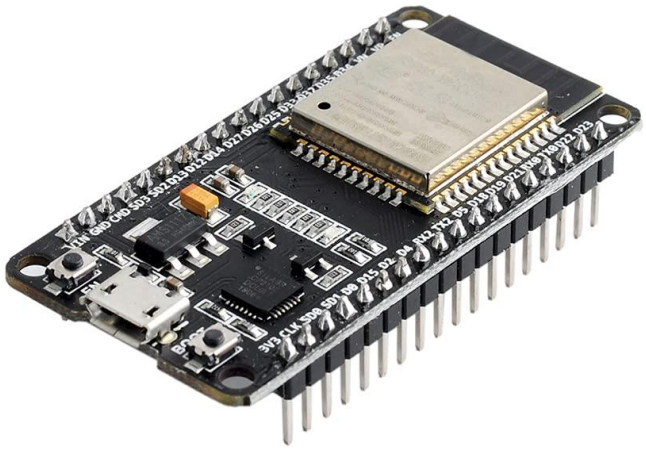

Figure A.3: ESP32 Node

```python
# -*- coding: utf-8 -*

import cv2
import numpy as np
import datetime
import csv

# Load YOLOv3 model
net = cv2.dnn.readNetFromDarknet('yolov3.cfg', 'yolov3.weights')

# Define the classes that the YOLOv3 model can detect
classes = ['person']

# Set the minimum confidence threshold for the predictions
conf_threshold = 0.5

# Initialize the video capture object
cap = cv2.VideoCapture("classroom EDITED V3.mp4")

# Set the size of the squares in the mesh
square_size = 255

# Create a CSV file to store the motion information
csv_file = open("motion_log.csv", "w", newline="")
csv_writer = csv.writer(csv_file)
csv_writer.writerow(['Start Time', 'End Time', 'X', 'Y', 'Width', 'Height', 'Type'])

start_time = None

while True:
    # Read the frame from the webcam
    ret, frame = cap.read()
```

Figure A.4: A YOLOv3 code snippet

```python
# -*- coding: utf-8 -*-
import csv
from datetime import datetime

# Define the zone coordinates
zones = {
    'Zone 1': [(0, 0), (640, 360)],
    'Zone 2': [(640, 0), (1280, 360)],
    'Zone 3': [(1280, 0), (1920, 360)],
    'Zone 4': [(0, 360), (640, 720)],
    'Zone 5': [(640, 360), (1280, 720)],
    'Zone 6': [(1280, 360), (1920, 720)],
    'Zone 7': [(0, 720), (640, 1080)],
    'Zone 8': [(640, 720), (1280, 1080)],
    'Zone 9': [(1280, 720), (1920, 1080)]
}

# Initialize a dictionary to store the total movement duration for each zone
zone_movement_duration = {zone: 0 for zone in zones}

# Read the motion log CSV file
with open('motion_log.csv', 'r') as file:
    reader = csv.DictReader(file)

    # Iterate over each row in the CSV file
    for row in reader:
        start_time = row['Start Time']
        end_time = row['End Time']
        x = int(row['X']) if row['X'] else None
        y = int(row['Y']) if row['Y'] else None

        # Iterate over each zone to check if the movement occurred in that zone
```

Figure A.5: A code snippet for motionlog after using YOLOv3

```python
# -*- coding: utf-8 -*-
import cv2 as cv
import numpy as np
import time
import datetime

# lenth is 640   width is 480


# Define the size of the image
img_width = 1500
img_height = 1500

# Define the number of bounding boxes to generate
num_boxes = 8

# Calculate the width and height of each bounding box
box_width = img_width // num_boxes
box_height = img_height // num_boxes

# Generate the bounding boxes
rectangles = []
for i in range(num_boxes):
    for j in range(num_boxes):
        # Calculate the x and y coordinates of the top-left corner of the bounding box
        x = i * box_width
        y = j * box_height
        # Append the (x,y,w,h) tuple to the list of boxes
        rectangles.append((x, y, box_width, box_height))


```

Figure A.6: An OpenCV code snippet

```python
# -*- coding: utf-8 -*-
"""
from __future__ import print_function
import cv2 as cv
from sys import exit
import argparse
parser = argparse.ArgumentParser(description='This program shows how to use background subtraction methods provided by \  OpenCV..')
#parser.add_argument('--input', type=str, help='D:\Downloads\3 Methods to Instant Playback The Recording from CCTV Camera via DVR NVR', defaul

#parser.add_argument('--input', type=str, help='D:\Downloads\3 Methods to Instant Playback The Recording from CCTV Camera via DVR NVR', defaul

#parser.add_argument('--input', type=str, help='D:\Downloads\3 Methods to Instant Playback The Recording from CCTV Camera via DVR NVR', defaul
parser.add_argument('--input', type=str, help='D:\Downloads\3 Methods to Instant Playback The Recording from CCTV Camera via DVR NVR', default


parser.add_argument('--algo', type=str, help='Background subtraction method (KNN, MOG2).', default='MOG2')
args = parser.parse_args()
if args.algo == 'MOG2':
    backSub = cv.createBackgroundSubtractorMOG2()
else:
    backSub = cv.createBackgroundSubtractorKNN()
capture = cv.VideoCapture(cv.samples.findFileOrKeep(args.input))
#capture = cv.VideoCapture(0)
if not capture.isOpened():
    print('Unable to open: ' + args.input)
    exit(0)
while True:
    ret, frame = capture.read()
    if frame is None:
        break

    fgMask = backSub.apply(frame)
```

Figure A.7: A MoG2 code snippet

REFERENCES

[1] Vijayan, D. S., Rose, A. L., Arvindan, S., Revathy, J., & Amuthadevi, C. (2020, November 19). *Automation systems in smart buildings: a review - Journal of Ambient Intelligence and Humanized Computing*. SpringerLink. https://doi.org/10.1007/s12652-020-02666-9

[2] *Study on Energy Saving Lighting of Classroom Based on Cirtopic*. (n.d.). Study on Energy Saving Lighting of Classroom Based on Cirtopic | IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/abstract/document/5701448

[3] Motlagh, N. H., Mohammadrezaei, M., Hunt, J., & Zakeri, B. (2020, January 19). *Internet of Things (IoT) and the Energy Sector*. MDPI. https://doi.org/10.3390/en13020494

[4] *Energy Efficiency in Smart Buildings: IoT Approaches*. (n.d.). Energy Efficiency in Smart Buildings: IoT Approaches | IEEE Journals & Magazine | IEEE Xplore. https://ieeexplore.ieee.org/abstract/document/9050775

[5] D. Ganiger, K. A. Patil, P. Patil, and M. Anandhalli, "Automatic control of power supply in classroom using image processing," in *2017 International Conference on Recent Advances in Electronics and Communication Technology (ICRAECT)*, 2017, pp. 230–234.

[6] T. Sali, C. Pardeshi, V. Malshette, A. Jadhav, and V. Thombare, "Classroom automation system," *International Journal of Innovations in Engineering and Technology*, vol. 8, no. 3, 2017.

[7] A. Gupta, P. Gupta, and J. Chhabra, "Iot based power efficient system design using automation for classrooms," in *2015 Third International Conference on Image Information Processing (ICIIP)*, 2015, pp. 285–289.

[8] G. Rathy and P. Sivasankar, "Iot based classroom automation using zigbee," *Inter- national Journal of Innovative Science, Engineering & Technology*, vol. 6, no. 3, pp. 98–102, 2019.

[9] P. Mrityunjaya, S. Muley, and D. Panda, "Intelligent classroom automation system using pic microcontroller," *International Journal of Research in Engineering and Technology*, vol. 05, no. 06, p. 154–160, 2016.

[10] M. Shruthi and G. Indumathi, “Motion tracking using pixel subtraction method,” in *2017 International Conference on Computing Methodologies and Communication (ICCMC)*, 2017, pp. 550–552.

[11] S. S. Shazali, W. L. Cheong, S. Mohamaddan, A. A. Kamaruddin, A. Yassin, and K. Case, “Motion detection using periodic background estimation subtraction method,” in *2011 7th International Conference on Information Technology in Asia*, 2011, pp. 1–4.

[12] S. Khedkar and G. M. Malwatkar, "Using raspberry Pi and GSM survey on home automation," 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), Chennai, India, 2016, pp. 758-761.

[13] K. W. V. Kulkarni and P. Wani, “Iot based secured classroom automation system using nlp,” *International journal of computer science and engineering*, vol. 5, no. 2, pp. 1–6, 2019.

[14] C. Paul, A. Ganesh, and C. Sunitha, “An iot-based smart classroom,” in *International Conference on Computer Networks and Communication Technologies Lecture Notes on Data Engineering and Communications Technologies*, 2018, p. 9–14.

[15] Medioni GG, Cohen I, Bremond F, Hongeng S, Nevatia R (2001) Event detection and analysis from video streams. IEEE Trans Pattern Anal Mach Intell 23(8):873–889

[16] A. Manju and P. Valarmathie, “Video analytics for semantic substance extraction using OpenCV in Python,” Journal of Ambient Intelligence and Humanized Computing, vol. 12, no. 3, pp. 4057–4066, 2021.

[17] OpenCV Documentation. (n.d.). BackgroundSubtractorMOG2 Class Reference. Retrieved from https://docs.opencv.org/3.4.9/d7/d7b/classcv_1_1BackgroundSubtractorMOG2.html

[18] *Zivkovic, Z. (2004). Improved adaptive Gaussian mixture model for background subtraction. In Proceedings of the 17th International Conference on Pattern Recognition, 2, 28-31.*

[19] OpenCV-Python Tutorials. (n.d.). Background Subtraction. Retrieved from https://docs.opencv.org/3.4/db/d5c/tutorial_py_bg_subtraction.html

[20] *Smaka, A., and Królikowski, J. (2019). A Comparison of Background Subtraction Algorithms for Static Camera Images. In Proceedings of the 9th International Conference on Digital Image Processing and Pattern Recognition (DPPR 2019), 102-110.*

[21] KaewTraKulPong, P., & Bowden, R. (2002). An improved adaptive background mixture model for real-time tracking with shadow detection. In Proceedings of the 2nd European Workshop on Advanced Video-Based Surveillance Systems (AVBS), 115-122.

[22] OpenCV-Python Tutorials. (n.d.). Background Subtraction. Retrieved from https://docs.opencv.org/3.4/db/d5c/tutorial_py_bg_subtraction.html

[23] KaewTraKulPong, P., & Bowden, R. (2002). An improved adaptive background mixture model for real-time tracking with shadow detection. In Proceedings of the 2nd European Workshop on Advanced Video-Based Surveillance Systems (AVBS), 115-122.

[24] OpenCV-Python Tutorials. (n.d.). Morphological Transformations. Retrieved from https://docs.opencv.org/3.4/db/df6/tutorial_erosion_dilatation.html

[25] OpenCV-Python Tutorials. (n.d.). Contour Features. Retrieved from https://docs.opencv.org/3.4/dd/d49/tutorial_py_contour_features.html

[26] Bradski, G., & Kaehler, A. (2008). Learning OpenCV: Computer vision with the OpenCV library. " O'Reilly Media, Inc.".

[27] *Smeureanu, I., & Sappa, A. D. (2019). Occupancy Detection in Smart Homes Using Convolutional Neural Networks. Sensors, 19(20), 4495.*

[28] Zhao, J., Wang, Y., & Li, Y. (2013). Real-time occupancy detection based on optimized sensor deployment in smart environments. IEEE Transactions on Industrial Informatics, 9(4), 2254-2263.

[29] *Stauffer, C., & Grimson, W. E. (1999). Adaptive background mixture models for real-time tracking. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2, 246-252.*

[30] OpenCV documentation: https://docs.opencv.org/

[31] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A Review of Yolo Algorithm Developments," *Procedia Comput. Sci.*, vol. 199, pp. 1066–1073, 2021, doi: 10.1016/j.procs.2022.01.135.

[32] W. Zhiqiang and L. Jun, “A Review of Object Detection Based on Convolutional Neural Network,” pp. 11104–11109, 2017.

[33] L. Tan, “Comparison of YOLO v3, Faster R-CNN, and SSD for Real-Time Pill Identi cation”.

[34] M. Ş. Gündüz and G. Işık, “A new YOLO-based method for real-time crowd detection from video and performance analysis of YOLO models,” *J. Real-Time Image Process.*, vol. 20, no. 1, 2023, doi: 10.1007/s11554-023-01276-w.

[35] O. Masurekar, O. Jadhav, P. Kulkarni, and S. Patil, “Real Time Object Detection Using YOLOv3,” pp. 3764–3768, 2020.

[36] *T. Diwan, G. Anirudh, and J. V. Tembhurne, “Object detection using YOLO: challenges, architectural successors, datasets and applications,” Multimed. Tools Appl., vol. 82, no. 6, pp. 9243–9275, 2023, doi: 10.1007/s11042-022-13644-y.*

[37] X. Wang and K. Jia, “Human Fall Detection Algorithm Based on YOLOv3,” *2020 IEEE 5th Int. Conf. Image, Vis. Comput. ICIVC 2020*, pp. 50–54, 2020, doi: 10.1109/ICIVC50857.2020.9177447.