OCCUPANCY TRACKING IN A BUILDING USING CONVOLUTIONAL NEURAL NETWORKS

BY

Paul Ortiz, B.S.E.E.

A thesis submitted to the Graduate School

in partial fulfillment of the requirements

for the degree

Master of Science

Major: Electrical Engineering

NEW MEXICO STATE UNIVERSITY

LAS CRUCES, NEW MEXICO

May 2022

Paul Ortiz
Candidate

Electrical Engineering
Major

This Thesis is approved on behalf of the faculty of New Mexico State University, and it is acceptable in quality and form for publication:

Approved by the thesis Committee:
Dr. Laura E. Boucheron
Chairperson

Dr. Deva K. Borah
Committee Member

Dr. Huiping Cao
Committee Member

ACKNOWLEDGMENTS

I would like to express my gratitude for the support from NM EPSCoR through NSF Grant #1757207 "RII Track-1: The New Mexico SMART Grid Center: Sustainable Modular, Adaptive, Resilient, Transactive". I would also like to thank my advisor Dr Laura E. Boucheron for providing the opportunity to work on this project and the knowledge and expertise she provided throughout.

VITA

June 2, 1998	Born in New Port News, Virginia, United States
2016 - 2020	B.S., New Mexico State University, Las Cruces, New Mexico
2020 - 2022	M.S., New Mexico State University, Las Cruces, New Mexico
2020 - 2022	Research Assistant, Klipsch School of Electrical and Computer Engineering, New Mexico State University.

FIELD OF STUDY

Major Field: Electrical Engineering

Area of Specialty: Image Processing and Machine Learning

ABSTRACT

TRACKING BUILDING OCCUPANCY FROM SECURITY CAMERAS TO ESTIMATE POWER DEMAND

BY

PAUL ORTIZ, B.S.

Master of Science

New Mexico State University

Las Cruces, New Mexico, 2020

Dr. Laura E. Boucheron, Chair

Tracking building occupancy has become an important topic due to recent advances in smart devices. The ability to provide a reliable occupancy count in a building can improve smart device functionality, as well as improve the power efficiency of their operation. This thesis presents a solution to tracking building occupancy using the security cameras that are present in most commercial buildings. This analysis was performed using a convolutional neural network

that ascertains the total occupancy of a building using computer vision and trajectory calculation methods. The specific building this was tested in was a standard office and laboratory building on a college campus that experiences a moderate amount of traffic. The following thesis provides an analysis of what specific data was extracted from the video feed and how this data is translated into an occupancy count. A brief discussion on speeding up the network and how the convolutional neural network was configured and a method to take the extracted data from the video feed and visualize it over a map of the building is described. The resulting occupancy count was relatively accurate to the expected occupancy of that building size and the occupant trajectories overlaid on the floor plan showing decent approximations of what path the occupants took.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

As modern appliances have become more advanced through technological development, the need for supplemental data to improve functionality has become more relevant. These devices can benefit greatly from having information about their environment. A powerful point of data, and the focus of this thesis, is the total number of occupants in the location where a device presides. Occupancy data can be particularly useful for these devices, including such applications as saving energy by allowing devices to regulate their operation based on the amount of occupants in the building. The main purpose of this thesis is to provide this data to the smart grid so any device connected to a smart grid will be able to make better-automated decisions.

This work provides a study of building occupancy using already existing security camera systems in a building using a convolutional neural network on the camera security feeds as well as further development in the process of tracking individuals using trajectory-based methods. The trajectory data is then converted into usable statistics for the building. Contributions in visualizing subjects in frame from an overhead perspective have also been added.

The results from this thesis are used in order to justify the practicality of this approach. This thesis also justifies the network parameters and other processing

techniques used for calculating total occupancy totals. Techniques used to shorten the length of time the convolutional neural network needs to perform detections are also described along with their corresponding effects on the data.

The structure of this thesis is as follows: Chapter 2 provides information on previous methods, as well as a general description of convolutional neural networks and detail about the specific network used in this thesis. Chapter 3 describes the general methods used to infer occupancy from the security footage. In Chapter 4 the results from analyzing a week of security footage, the effects of chosen metrics had on the resulting occupancy counts, and the effect chosen parameters have on the data are described. Chapter 5 contains closing remarks and directions for future work.

2 BACKGROUND

This chapter opens with a description of building occupancy and what previous methods have been used to track occupancy. It is followed by an analysis of the differences between the methods used in this thesis and a comparison of a similar camera-based solution. This chapter? also details other approaches that have been taken to dynamically track occupancy. Next is a discussion of convolutional neural networks, what they are, and why they are preferred for this application over other methods used for image detection tasks. Finally, a description of the particular convolutional neural network that was used is provided, along with the reasoning for using it.

2.1 Building Occupancy

Due to the prevalence and utility of smart devices, the need for devices to have an estimate of the occupancy of a building has become a more relevant topic. Building occupancy can be a powerful point of data for making more informed decisions for smart appliances [1] . A smart device is a device that is connected to other devices or the internet with the purpose of being more interactive or to have some autonomy [2]. These devices are much more prevalent due to the ease of access to the internet and other wireless communication.

Occupancy data, for example, would help a smart heating, ventilation, and

air conditioning unit (HVAC) unit make better decisions about appropriate temperature settings in the building. This data could even be useful for smaller smart appliances, such as a smart coffee maker that is able to automatically brew coffee based on how many are present building. This data could thus be helpful for device functionality as well as energy savings.

2.2 Occupancy Tracking

While there are many potential techniques that can be used to track occupancy, the technique used specifically in this research is to infer occupancy through the use of preexisting security camera infrastructure in buildings. The main idea behind this technique is to track those who appear in frame and make decisions about the total occupancy based on their movement. The total occupancy of the building is the main focus of this thesis, and the location of the occupant inside the building is not considered. This method of using preexisting hardware was in order to ensure ease of implementation for other locations. This method comes with the downside of the accuracy of the resulting occupancy being dependent on the quality and configuration of the security cameras. The major upside is the ease of integration of this system, as no additional hardware would need to be implemented other than the computational hardware needed for occupancy analysis.

2.3 Occupancy Tracking Techniques

Previous research has resulted in a multitude of solutions to occupancy tracking. As there is no single answer to how occupancy can be tracked, there are a few approaches that have been taken. The first major approach is using preexisting hardware already incorporated as part of the building infrastructure, whether it be cameras, network devices to track internet traffic, or other sources of data from the building. Another type of approach is to develop a new set of sensors or data collection devices that are independent of preexisting infrastructure in the building. Both approaches come with advantages and disadvantages associated with the cost and ease of implementation.

2.3.1 Camera Based Solutions

The solution used in this thesis is based on preexisting security cameras installed in buildings. Camera systems are standard in most commercial buildings, so this implementation is practical to implement in most buildings. There are a few approaches that can be taken using preexisting security cameras, but the general approach is to extract occupancy data from the security feeds of the building using image processing techniques. How the videos are processed to extract occupancy information is where potential differences lie.

A paper that uses a similar approach to that of this thesis is proposed by Zou et al. [3] where attributes of specific occupants are extracted, such as height and other

features, and each occupant is tracked independently using those features. This is dissimilar to the approach used in this thesis, in that the approach proposed in this thesis treats occupants independently of their characteristics since this thesis is concerned more with the volume of occupants instead of knowing where a specific occupant is in the building or the identity of that specific occupant. While the approach in [3] and this thesis are fundamentally similar in that they both use security cameras, they net different final results. Where the output of the Zou et al. [3] method is specific individuals tracked based on their attributes, this thesis is more focused on the trajectory of an occupant to better ascertain events such as entry to and exit from the building or transition between hallways. Both the methods of this thesis and that of Zou et al. [3] are susceptible to issues with security footage such as data corruption and improper data storage, as well as the quality of the image from the security cameras.

A major concern Zou et al. [3] stated with using the type of analysis used in this thesis is when occupants obscure one another in frame. A lot of the reasoning behind characterizing occupant attributes in Zou et al. [3] is to ensure all occupants are accounted for even when obscured. This is also handled in this thesis with buffering to handle occupants being obscured, further detailed in Chapter 3. While occupants obscuring one another will be a persistent issue with the camera-based solutions in this thesis and that of Zou et al. [3], the effects of occupants obscuring each other can be reduced by using networks capable of

handling these overlapping objects, and careful handling of occupant trajectories. Zou et al. [3] suggested using an overhead camera for higher accuracy, but that would conflict with the intended ease of implementation that comes along with using preexisting security cameras.

2.3.2 Smart Phone Data Solutions

Smartphones and other devices that use the internet services from a building provide data that could be used to estimate the occupancy. An approach using this data from networking devices is described in Shen et al. [4], where the Bluetooth signal from smartphones is used to track occupancy by collecting a list of smartphones in the building. This solution has the benefit of not needing additional hardware to implement, and was shown in preliminary testing to be a promising solution. There are some disadvantages in that devices may be left in the building and still be counted as an occupant, as well as some smartphones disabling Bluetooth when they are at low battery, or not having Bluetooth capabilities.

2.3.3 Sensor Based Solutions

Another technique is to build a custom device that detects the presence of occupants using sensors that pick up some change that occurs when an occupant is present. CO2 levels and infrared sensors were used in Luppe et al. [5] and the

sensor data was fused together to calculate the occupancy. While this uses a custom sensor that needs to be installed in the building, the overall cost to manufacture the devices is not overly costly and shows promising results.

2.4 CNNs versus Classical Machine Learning

A Convolutional Neural Network (CNN) is a form of machine learning, but has a different approach than the classical approaches to machine learning. In classical machine learning, a set of features is defined and extracted from the data, such as color features, boundary features, and region-based features of an image or object in an image. These features are used to make decisions about the objects present in an image, e.g., image classification which indicates what class of object is present in an image. The machine learning algorithm learns feature values specific to a class by analyzing a set of training data in which objects are annotated/labeled with the correct class. This approach can be very application specific, as for some applications, definition of relevant features for objects in an image can be impractical based on the complexity of the data.

The term deep learning is given to any type of machine learning neural network that uses multiple hidden neural layers. The specific type of hidden layer used in the network chosen in this thesis are convolution layers. The features are extracted with the basic mathematical operation of convolution, where a filter kernel is scanned from left to right through the entire image to

compare it against the features in the image. These convolutional kernels can be thought of as filters that extract features, and the set of filtered images are called a feature map. As the network goes further through the layers, the level of abstraction increases. Pooling layers cause a spatial reduction in resolution, and the convolutional filtering continues. The pooling layers are included to reduce overfitting, as without them the network would be working with an overabundance of feature maps. This cycle of convolving and pooling continues until reaching the fully connected layers which are responsible for classifying the features extracted by the convolutional layers. Fully connected layers can be thought of as a one dimensional squashing of a feature map where each fully connected layer is connected to the previous and next layer with a weight to each path and these weights are trained and recalculated to decide what is present in the image. The layers reach the final output layer which decides what the image is by giving scores to each object the network was trained to detect.

2.5 ResNet

The type of network used in this thesis can be described as a Residual Network or ResNet [6]. A common problem in CNNs is that the accuracy of a network may start decreasing when adding more layers because of vanishing/exploding gradients. More layers should in theory increase accuracy, but the effect of these vanishing/exploding gradients means typical networks such as VGG stick to 16

to 30 layers [7]. A ResNet reduces this accuracy problem by taking the output of previous layers and summing them with the output of future layers to ensure that data from previous layers are not degraded, and thus more layers can be added. ResNet layers are split into "residual blocks" comprised of the convolutional layers, normalization layers, and activation layers. The output of the blocks is sent to a pooling layer and to the fully connected layer. These blocks can be appended to one another for as many segments as necessary. The chosen network in this thesis uses ResNet101, meaning it is comprised of 101 of these blocks.

2.6 Region Proposal Networks

The network used in this research is a region proposal network named Mask R-CNN [8]. This network is based on the Fast R-CNN [9], both being region proposal networks. Region proposal networks can be thought of as convolutional neural networks with an additional smaller network that takes the output from a convolutional layer, which would be the feature map output, and feeds it into the region proposal network. This region proposal network creates bounding boxes in the regions of interest of the image. These regions of interest are then searched for features of objects the network has defined, and is done by sliding a window across the feature maps to see if it contains a recognizable object. These regions of interests, or ROIS, are then extracted for features. This continues until a tightly fitting bounding box is decided around the objects in the frame. Faster R-CNN

speeds up this process even more by including a pooling stage in this region of interest, which reduces the total number of regions proposed which in turn reduces the total amount of computation required.

2.7 Mask RCNN

Mask RCNN takes the process of labeling objects in frame one step further and provides a mask rather than a bounding box specifically around the edges of the object in question [8]. This is done by including another stage that attempts to identify each pixel in an image with the corresponding class it has identified, known as semantic segmentation. This semantic segmentation is performed by using an additional network called a Fully Convolutional Network that is connected to the original network. This kind of network takes the feature maps extracted in the convolution layers, upscales them to their appropriate size in the image, and combines them to create the mask [10]. The output is a binary mask that is zero if the pixel does not contain the class and a 1 if the pixel does belong to the class. These results are further refined in instance segmentation, which labels each object in frame with an individual mask over the pixels that represent an object in frame. Mask RCNN has the benefit of higher accuracy than the original Faster RCNN, with only a slight computational overhead added.

Mask R-CNN was chosen over the faster Fast R-CNN its the improved accuracy. The ability to return the instance segmentation is also useful in

applications potentially more complex than this, as referencing specific points on a detection could prove useful for better network clarity. Through testing, the speed difference between Fast RCNN and Mask RCNN was not substantial enough for Fast RCNN to be used. The accuracy of the Mask RCNN network was instead more appealing.

3 METHODS AND DATA

The chapter describes the data and the methods used for building occupancy tracking. First, this chapter begins by describing the data set provided by New Mexico State University that was used to develop and test the tracking algorithm. Second, a description of the prepossessing stage is provided. Third, a description of how the image frame coordinates of those seen in frame are extracted using a neural network, along with the network's configuration follows. Fourth, the thought process behind why this specific network was chosen is described. Fifth, the methods for translating image frame coordinates into physical occupant trajectories is described. Sixth is a description of how the gathered occupancy image coordinates are translated into events and further organized into a total occupancy count at any given time. Finally, this chapter closes with a description of the visualization process used to return a map of all occupant positions and travel history. A block diagram of the occupancy tracking process is shown in Figure 1.

3.1 Data

The video data that was used in this work was provided by New Mexico State University. The data consists of the security videos from 2020-2022 of the NMSU building Goddard Annex, with calculations being performed on data from the beginning of 2022. The security videos are saved on the New Mexico

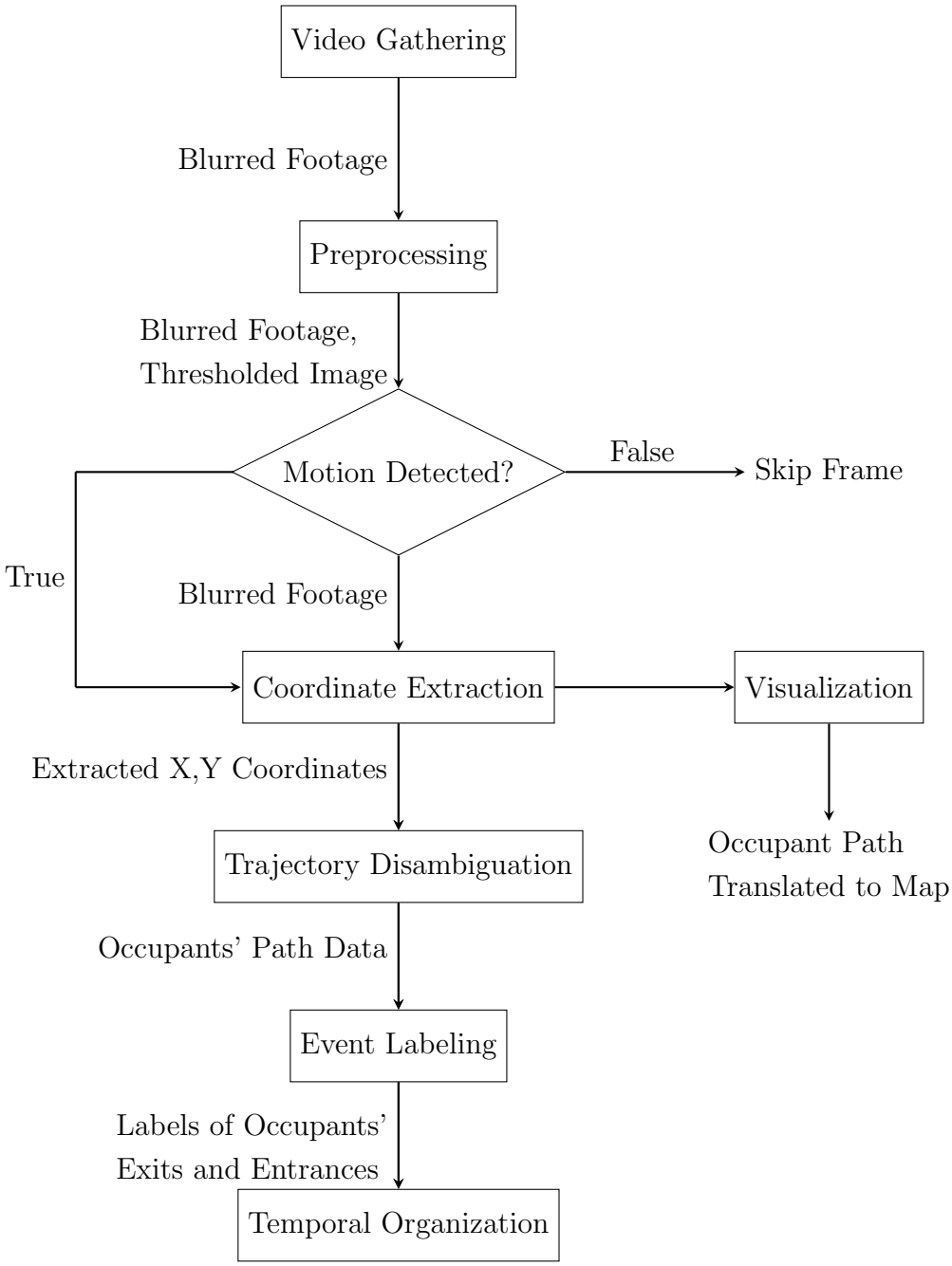


Figure 1: Stages describing the entire occupancy tracking process and inputs/outputs for each part of the occupancy tracking process.

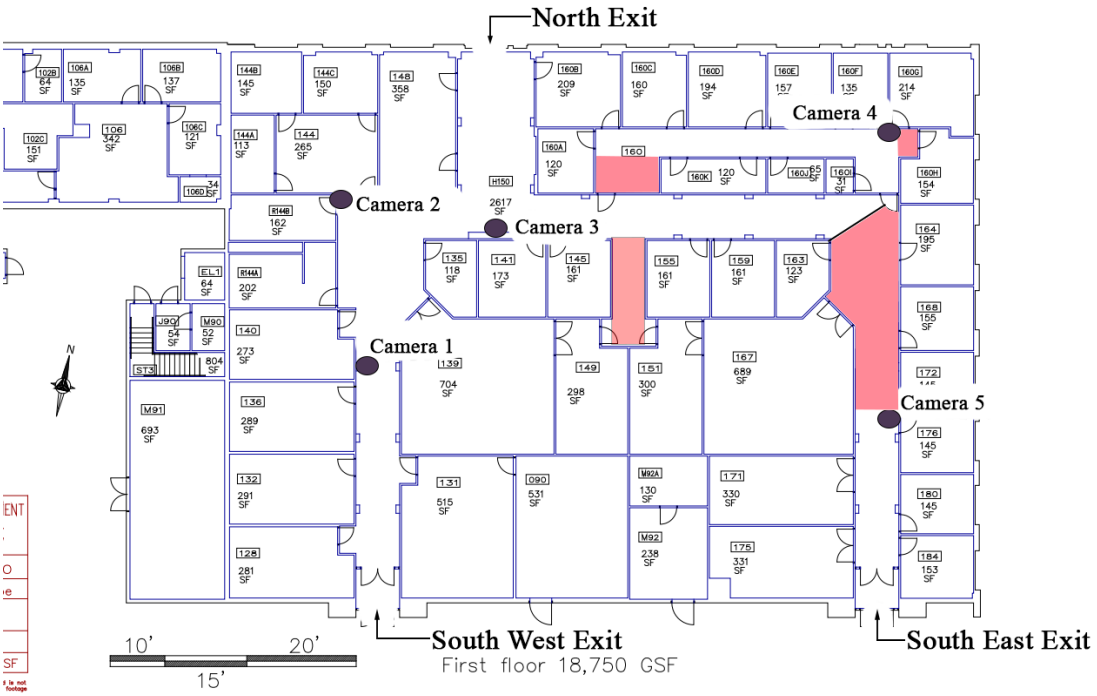


Figure 2: Goddard Annex floor plan. The locations of the five cameras are denoted by black ellipses and the exits have been labeled for future reference. The pink regions denote the blind spots of the cameras.

State University supercomputer cluster Discovery for subsequent processing. The data set contains video sequences from five cameras placed in Goddard Annex in the positions illustrated in Figure 2. For the purposes of this research, the videos provided have been blurred to obscure individual identities, specifically with a Gaussian blur of kernel size 11×11, and a standard deviation of 7. This kernel size and standard deviation were specifically chosen as it blurs the faces of occupants enough that the features of their face are not distinguishable. The use of the security videos for this research was approved by NMSU's Institutional Review Board (IRB).

The video cameras in each hall record in 1200 by 800 pixels at 10 frames per second. The footage is compressed before it is saved for the purposes of reducing the amount of storage needed to archive the footage (see Figure 4). The camera system has a motion detection process for further data size reduction. This means the cameras only record when they detect movement. The motion detection built into the camera system is indifferent to what is is the frame, but is reasonably sensitive to any changes in the frame. Overall, the final output videos for an 8 hour workday are cut down to a 10 to 60 minute video that contains only frames in which motion was detected, usually due to persons moving around the building. While the motion detection is reasonably reliable, it is possible that some events of persons moving throughout the building are missed or truncated in length. These videos are then saved to the camera's corresponding server and then sent to Discovery.

Goddard Annex is a building composed of faculty and student offices, as well as conference rooms. There is also a separate building to the west of Goddard Annex as seen in Figure 2, but there is not an internal connection between these two wings and the footage for the western wing of the buildings is not available for this research. Goddard Annex receives a moderate amount of foot traffic, with most of the foot traffic being composed of people traveling to their office and others walking to consult with peers. There are three entrances to the building and each has a corresponding camera set to watch the door (Cameras 1, 3, and

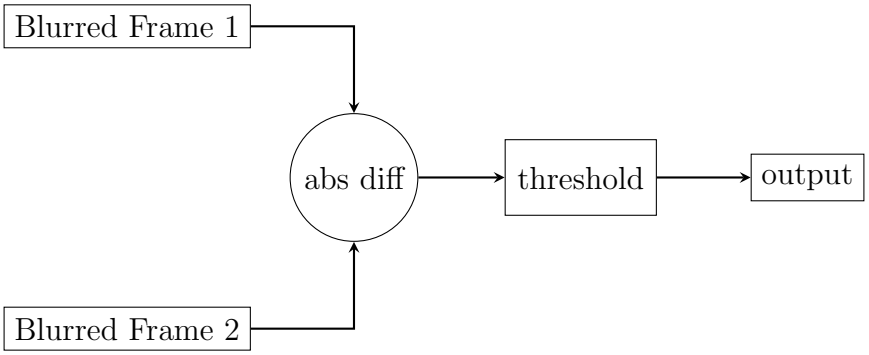


Figure 3: Stages describing the motion detection algorithm.

5 in Figure 2). There are two other cameras set up in the hallways to expand coverage to other portions of the building. There are noticeable blind spots in some of the hallways which are shown in pink in Figure 2. All other rooms are implied to be blind spots as the camera system does not monitor them.

3.2 Data Preprocessing

Before extracting the coordinates from a video source, preprocessing steps are included in order to improve processing time. The time required to run detections on a second of video ranges from 3 to 5 seconds depending on the complexity of the image. In order to cut down on processing time, a secondary motion detection algorithm is performed before subsequent processing in order to process only frames that have movement in them. This algorithm is illustrated in Figure 3. The reason for a secondary motion detection process is that the motion detection inside the camera is broad in what it accepts as movement. The portions of the videos of most import are the portions that contains the movement of individuals,

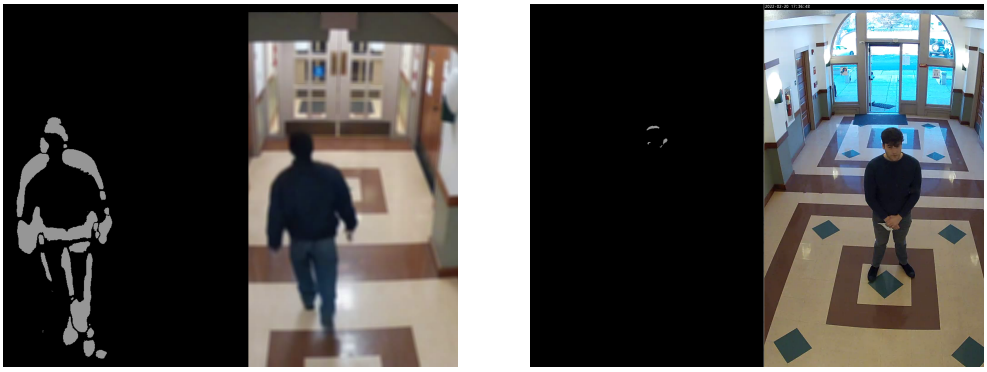
and the camera system is prone to superfluous recordings of an empty hallway triggered by either something unrelated in frame, and also to start recording when nothing is in frame for unknown reasons. As an example, the North Hallway has a window in the background which has the constant motion of cars driving behind it, causing the camera's motion detection to be activated constantly. The resulting video is often bloated with this footage making it three to four times as long as the other halls. The extra footage in question is irrelevant to the occupancy count. Placing an extra layer of detection removes these anomalies to cut down on the total processing time of the network by reducing the number of frames necessary to process.

The first step in the secondary motion detection process is to take the absolute difference of the current frame from the previous frame. This results in an image with pixels representing the magnitude of change between the two frames, where zero-valued pixels represent no change. While this initial image gives a decent approximation of the change in the frame, it still contains noise that does not represent actual movement in the frame. The general source of that noise is from the image artifacts caused by compression as well as noise from the sensor in the camera. An example of how the noise affects the overall quality of camera feeds is shown in Figure 4. These noise sources are lessened in the process of blurring due to the Gaussian kernel smoothing out these hardware and compression artifacts, which was performed in the initial step. The second step in the motion detection

Figure 4: Image showing the effects of noise from the video compression and camera noise. The red sign label 'EXIT' is difficult to read due to these effects on the video.

is to use a pixel value threshold to remove pixel difference magnitudes that are too small to represent real movement. This threshold is set so only pixels with an intensity between 15 and 150 out of 255 are allowed to pass through. The threshold filters out values great than 150 because pixels in the threshold image that are above 150 tend to be from compression artifacts causing a large difference in pixel values.

In order to decide if the frame has any movement, a sum of the binary threshold image can be taken, which represents a pixel area of movement. If this algorithm results in an image with zero to very few white pixels there is an insufficient



(a) Subject A: motion detected (b) Subject B: no motion detected

Figure 5: Left images: thresholded pixel differences where gray pixels are those with movement detected. Right images: image frames. Subject A moves enough to trigger motion and subject B does not move enough to trigger the motion detection.

amount of movement between the previous and current frame as the frame has not changed a significant amount. A threshold can be placed that further tightens the threshold for what is considered movement by increasing the number of pixels that represent movement in the binary image. The threshold is best decided by the size of a person in the frame, as in some applications the subjects in frame only take up a small portion of the image, meaning the sum constituting actual movement should be lower to account for this. An example of an occupant moving enough to trigger detection and one not moving enough to trigger detection is shown in Figure 5. It is important to note that the threshold image is only used for motion detection. Once it is determined there is sufficient movement in the frame, the footage that has only been blurred and not processed in any other way is what the network uses to detect occupancy.

Figure 5 shows two images of different situations, where the black and gray

portion of each image is the threshold image resulting from preprocessing. Figure 5(a) depicts an occupant in motion, which is shown by a high count of white-gray pixels in the threshold image on the left. Figure 5(b) shows an occupant standing still and causing very few white-gray pixels in the threshold image. The occupant in Figure 5(b) is not completely still, as there are white-gray pixels around their head indicating that they slightly moved their head, but this very slight movement is not considered enough to trigger a detection.

In the case where no movement is detected despite someone being in frame, such as the occupant standing still in Figure 5(b), the network can be forced to perform a detection. This is done by regularly pinging the network to perform a detection even when it does not detect motion, and is done by bypassing the motion detection step on a regular interval. In the calculations performed, the buffer set in the path processing algorithm is 40 frames, where if an occupant does not receive another coordinate for 40 frames, their path data is dropped. To prevent an occupant from being dropped incorrectly, overwriting the detection could be performed at regular intervals to ensure that an occupant standing still in frame gets their coordinates extracted within a 40 frame window so that their coordinates are continually extracted and their path is not dropped.

This motion detection can be falsely triggered by a scene with a consistent source of noise such as a window with cars and people passing behind it outside of the building, or an object that produces constant movement like a clock in the

frame. These false triggers can be further reduced by masking parts of the video frames known to contain objects susceptible to these false triggers. The mask must take into account the perspective in the scene. A simple mask laid on top of the frame would obscure valid regions where occupants can travel causing the network to not be able to detect it.

3.3 Coordinate Extraction

The process of detection begins with interpreting where the subjects are in the image. There are a few techniques for extraction of a person in the frame, such as difference images, classical machine learning, and convolutional neural networks. This research favors convolutional neural networks due to their accuracy and adaptability. The specific network used is Mask RCNN [11]. The network labels the subjects in frame with two different indicators, a mask and a box. Since the coordinate extractor should be indifferent to the size and shape of someone in the frame, the use of a mask or box corresponding to a person is not essential to extracting coordinates. The only information necessary to specify a person and their location in this application is a single (x, y) coordinate inside of the box that the network returns.

The specific pixel chosen as the location of a person is an important decision, and the simplest choice is the center of the box. While this may be an adequate choice of pixel for some applications, and would work for the major steps in this

Figure 6: Image of what pixel is extracted in an image to represent subject location, seen in the magenta circle at the bottom of the occupant's feet.

experiment, the best choice for a pixel is in the middle of the lowest point of the box due to later visualization steps. This point represents the middle of the person's feet. By doing this, the (x, y) coordinate that is returned can be interpreted as a point moving along the plane of the floor if the scene was observed from overhead. An example of this coordinate choice is shown in Figure 6. This is essential for later visualization steps that involve affine transformations. This is later expanded upon in Section 3.7.

3.3.1 Mask RCNN Configuration

The network chosen for this work is the Mask R-CNN developed by Facebook AI Research (FAIR), and the specific configuration of this network was developed by Matterport Inc. [11]. The reason this network was chosen is because of its

accuracy compared to its speed. In this specific application, a fast network is preferred to a slower more accurate network due to the sheer amount of frames that need to be processed. Mask R-CNN is a good middle ground between speed and accuracy. While not real-time, the network still provided decent results in a reasonable amount of time with consistent accuracy.

The network itself was pre-trained on a large COCO data set [12] that contained an adequate training set for human detection ("person" is a class included in the COCO dataset). A custom data set built around this experiment would further optimize the network's accuracy, but due to the intended general purpose nature of this process, the more general COCO data set provided to be representative enough for a wide scale of applications.

3.3.2 Network Confidence

With default parameters, the network was overly sensitive to human-like objects and provided coordinates of objects that were not people. These anomalies had a pattern in their confidence scores. The confidence scores of these objects that were not people tended to score lower by about 10 percent than actual people. In order to counter this, a threshold on the network's confidence scores provided a more accurate analysis of the people in frame. Filtering by the network's confidences allows for more consistent detections at the cost of false negatives. As shown in later sections, a missing detection is less costly for overall accuracy than a

superfluous detection. A threshold of confidence of around 97 percent removed these anomalies and the dropped frames (false negatives) were handled by the buffering system expanded on in Section 3.5.

3.3.3 Timestamp Extraction

In addition to the spatial coordinates of detected persons, the timestamp of the detection must be included to have a time-dependent analysis of occupancy. The footage provided includes a timestamp in the top left. This timestamp can be extracted using optical character recognition (OCR). This was achieved using the pytesseract [13] OCR library, and was included in the coordinate extraction phase of the process. To extract the timestamp, the portion of the image with the timestamp is separated from the rest of the image. The pytesseract library assumes black text on a white background, which is opposite of the timestamps' white text on black. The colors were inverted, and a dilation filter of size 2 by 2 was convolved through the timestamp image to increase the thickness of the text, (cv2.dilate) was run in order to make the text larger which is easier for the pytesseract library to handle. From there, the timestamp is in a refined enough state that pytesseract is able to have its detection function called and the timestamp is returned in a string. This string is then written into the first row of the matrix described below in Equation (1).

3.4 Trajectory Disambiguation

For frames in which motion was detected, the Mask RCNN network is used to detect any persons in the original (blurred but otherwise unprocessed) frame and the timestamp is extracted from the upper left corner of the same frame. The coordinates returned by the Mask RCNN network and the timestamps returned by the OCR are written to a comma separated variable (csv) file for later data processing. The format in which they are saved is

$$\begin{array}{llllll} t_1, & (x_{1,1}, y_{1,1}), & (x_{1,2}, y_{1,2}), & \cdots & (x_{1,n}, y_{1,n}) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ t_m & (x_{m,1}, y_{m,1}), & (x_{m,2}, y_{m,2}), & \cdots & (x_{m,n}, y_{m,n}) \end{array}, \tag{1}$$

where n is the number of persons detected in the frame, m is the number of frames in the video, t_i is the timestamp for the i-th frame, and $(x_{i,j}, y_{i,j})$ are the coordinates for the j-th person in the i-th frame. Inconsistencies in time-stamping are handled later when the coordinates are processed. If a frame does not contain a person, the row is left entirely empty, including the timestamp due to the timestamp extraction taking a somewhat significant amount of time to perform. The row is left blank to ensure that the frame number of the video corresponds to the row number in the csv. For future reference, we define the coordinate matrices $\mathbf{X}$ and $\mathbf{Y}$ to consist of the x and y coordinates, respectively, from the csv file:

$$\mathbf{X} = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m,1} & x_{m,2} & \cdots & x_{m,n} \end{bmatrix}, \tag{2}$$

$$\mathbf{Y} = \begin{bmatrix} y_{1,1} & y_{1,2} & \cdots & y_{1,n} \\ \vdots & \vdots & \ddots & \vdots \\ y_{m,1} & y_{m,2} & \cdots & y_{m,n} \end{bmatrix}, \tag{3}$$

where $x_{i,j}$, $y_{i,j}$ are the x-coordinate, y-coordinate for the i-th time and j-th person.

The method of pulling coordinates from pre-recorded footage is used due to the limitations of the data. A live feed of buildings was not available for this research. This means the data from an earlier date is referenced and used for statistical analysis. The disadvantage of this method not being real-time means decisions about the actual occupancy is decided after the event takes place, meaning real-time changes to smart devices will be based on a statistical model instead of decisions being made on the fly. This system, however, is designed so it can be adapted to a real-time system if given a live feed and with implementation on hardware capable of processing both the Mask RCNN and path processing in real-time.

The coordinates are fed into a script one frame at a time with the corresponding footage playing beside it to visualize what is happening in the current frame. The total number of occupants that are in the total footage is unknown to the algorithm as it reads the csv file, as the algorithm is programmed to accept one frame (one row) at a time instead of reading the entire csv file at once. This choice

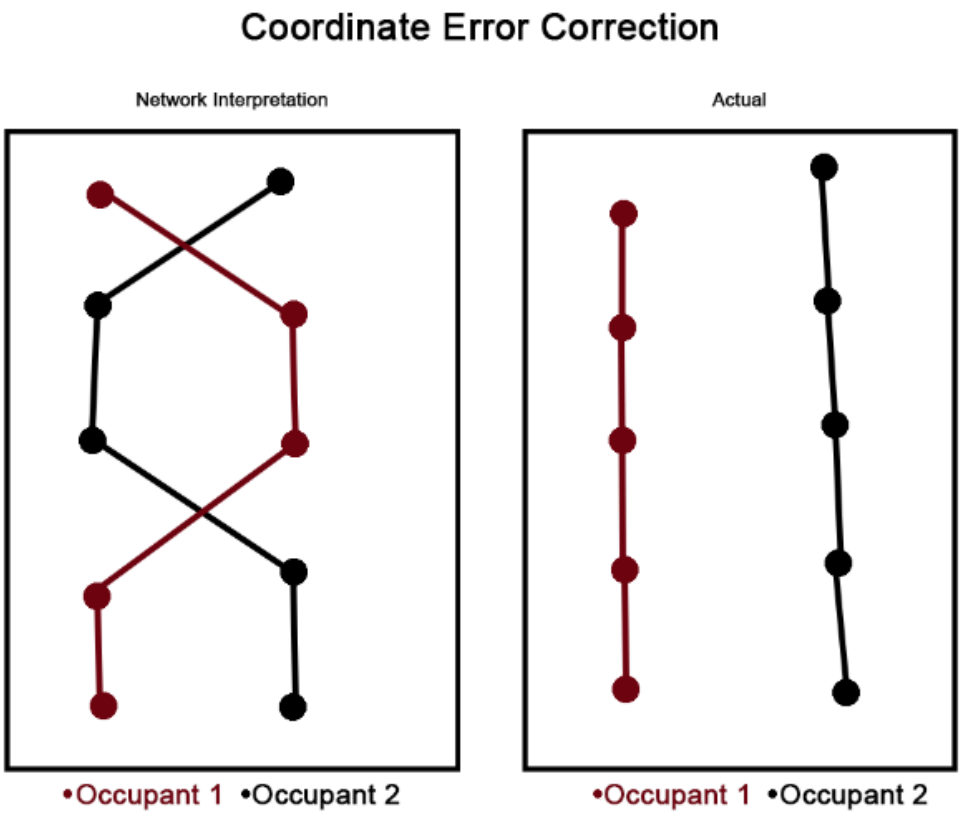


Figure 7: Example of how the closest coordinate fixes the error of incorrect coordinate assignment.

is consistent with the ability to use these methods on a live video stream. The coordinates for the subjects in the frame can contain some anomalies, which can be further exacerbated when two or more subjects are in frame. These anomalies are still given a tag as an occupant at this time.

The network is inconsistent in its assignment of a person to a coordinate, meaning an occupant can have a coordinate assigned to them that belongs to another occupant (see Figure 7). In order to make consistent paths, before being associated to an occupant's data, the coordinate is assigned to the occupant whose

last known coordinate is the closest Euclidean distance. Equation (4) describes this calculation.

$$L_t = \underset{\forall x_{t,i}, y_{t,i} \in \mathbf{X}, \mathbf{Y}}{\operatorname{argmin}} \sqrt{(x_{t,i} - x_{t-1,L_{t-1}})^2 + (y_{t,i} - y_{t-1,L_{t-1}})^2},\ i = 1, \ldots, n \qquad (4)$$

where L_t is the trajectory assignment (an arbitrary integer label) for time t, and $x_{t,i}$, $y_{t,i}$, $\mathbf{X}$, and $\mathbf{Y}$ are defined as in Equations (2) and (3). This process is done for each occupant's last known coordinate, and once the coordinate is chosen it is appended to the occupant's previous coordinates and the process continues for the remaining occupants. Occupants are processed in arbitrary order and once a coordinate has been assigned to the closest occupant, that coordinate is no longer available for assignment. After all occupants have their closest coordinate, the algorithm moves on to the next frame of the video.

If an occupant does not have a closest detection, such as when a person leaves the frame before another, the closest detection belongs to another occupant, or the network drops their position, nothing is appended to their data. If no occupants are detected in frame, an empty frame buffer is incremented. Once this empty frame buffer reaches 40 frames (4 seconds), the data is handed off to the next stage of the process. A visual representation of the closest coordinate algorithm is shown in Figure 7.

After this empty frame buffer threshold has been reached, the script takes all of the occupants and their paths to filter out anomalous data. Anomalous data is

described as data that is not of an appropriate length and does not have a valid trajectory. The filtering is to remove any occupant path that does not have more than 15 points in its path. This filter is designed to remove short detection errors that are common in human-like objects as the network will likely only detect these anomalies for a few frames and return to normal. This is a different buffer than the empty frame buffer, as it is intended to only remove small paths. The empty frame buffer is used to decide when all occupants have exited the frame.

3.5 Event Labeling

Paths are interpreted by a labeling system. Once the algorithm has decided that all occupants have left the frame it starts to process the coordinate data. From the occupant data, the initial and final points of when they were seen in frame are given labels. The labels are decided by boundaries in the frame of the video. There are two types of boundaries: Hallway and Exit. These bounds are set by drawing a polygon in frame labeled according to where the hall and offices are located in the image, and are set on a per camera basis. The initial and final points of an occupant's path are checked to see if they are contained in the boundary polygons and labeled accordingly. The boundaries can be further tweaked to refine what area of the frame constitutes the given region. A sample image is given in Figure 8.

This system works on an event b-ed system. An event is defined by the labels

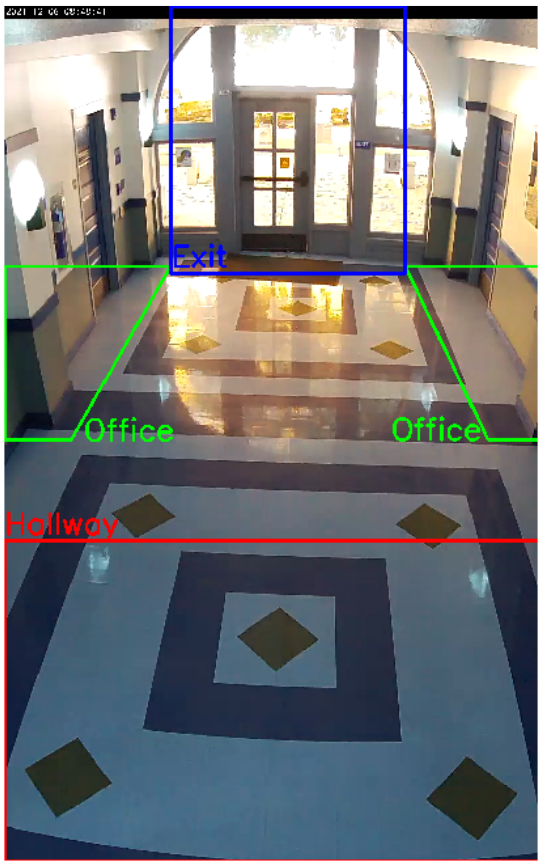


Figure 8: Sample of what the boundaries in an image look like.

given to the occupant's position history, the timestamp, and which feed they appeared in. These labels are then plugged into the lookup table described in Table 1. There are three major types of movements that are described by this system: Entry, Exit, and Transition. An entry is defined by someone entering the building into a hallway from outside the building. An entry increases the total global occupancy count of the building. The next type, exit, is an occupant leaving the building from a hallway, which would reduce the total global occupancy count. The final type of movement is a transition. A transition has two subparts, hallway

Table 1: This lookup table shows all the descriptions of all the events given the label of the initial and final position coordinates

Event	Final Location	Global Change
Exit to Hall	Hallway	+1
Exit to Exit	Exit	+0
Hall to Hall	Hallway	+0
Hall to Exit	Exit	−1

to hallway and exit to exit. In a hallway to hallway transition, the occupant in frame entered from a hallway and left in a hallway. This implies an occupant is already inside the building, but changed their mind about leaving and reentered the hallway outside the frame resulting in a global change of 0 due to them simply moving through the building instead of exiting. An exit to exit transition is similar in that an occupant enters in through an exit and doubles back into the exit. This also results in a global change of 0. The details of the transition data can be viewed in Table 1.

Each event is written to a csv file and labeled according to the location of the camera in the building. The event can be represented by a vector

$$\mathbf{e} = [T_0,\ T_f,\ G,\ C] \tag{5}$$

where T_0 is the initial time of event detection, T_f is the final time of event detection, G is the change in global occupancy, i.e., $+1$, -1, or 0, and C is a label of the camera feed in which the event took place. These event vectors $\mathbf{e}$ can be concatenated into an event matrix $\mathbf{E}$ were each row $\mathbf{e}_i,\ i = 1, \ldots, p$ is an

event vector for the p total events,

$$\mathbf{E} = \begin{bmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \\ \vdots \\ \mathbf{e}_p \end{bmatrix} \tag{6}$$

The data for each hallway is saved individually in a csv file with the event vectors each saved as a row in the file. This results in a csv containing every event from that feed in a given time period, which in the case of this research was a day of occupancy. Saving the event data independently for each of the camera locations has the advantage of not needing a set amount of cameras, and instead allowing for a complete occupancy count with cameras only pointed at the exits. If, for example, there was a missing camera feed at one of the building's exits, the count would be incomplete due to the events from this missing hallway not being included in the total.

3.6 Temporal Organization

With the event data and the timestamps calculated for each of the camera angles, the process of total occupancy tracking can begin. The time of the event is saved in the form of a time interval $[T_0\ T_f]$. The time interval allows for the specification of the time of entry T_0 and the time of exit T_f in the frame. This is used in the case of overlapping events. In the case of an overlapping event, the time of exit is used to decide which event completed first. This is to prevent out-of-order events when two occupants enter at the same time and one takes longer to exit. However,

due to the nature of the data set obtained from Goddard Annex, in practice the flow of occupants is not high volume enough to cause this precaution to come into effect due to occupants trickling at a slow rate. Instead, the events are organized only by the initial time T_0 that they enter the frame.

In the case of two cameras that overlap regions of a building, the simplest solution to stop this from happening is to mask out the region in which the cameras overlap. This, however, is an incomplete solution because masking out regions will lose information in the frame. As explained before, masking over a video feed can destroy valid portions of the image that are necessary for an accurate count. This loss would lose the data of an occupant as they cannot be detected in this state. Instead, a coordinate filter can be placed on one of the cameras so that a person arriving in that region of the frame will be ignored, and the camera that is not filtered that covers that area will instead handle that person's data. In this experiment, there are no examples of overlapping cameras when only considering the exits of the building, so no coordinate masking was necessary.

The total occupancy at any given time t_2 is defined as the sum of all events (see Table 1) up to that point, or equivalently as a prior occupancy value plus the difference for a given interval $[t_1, t_2]$:

$$Occupancy(t_2) = \sum_{\substack{\forall \mathbf{e} \in \mathbf{E} \\ s.t. \\ T_0 < t_2}} G = Occupancy(t_1) + \sum_{\substack{\forall \mathbf{e} \in \mathbf{E} \\ s.t. \\ t_1 < T_0 < t_2}} G, \tag{7}$$

where $\mathbf{e}$, $\mathbf{E}$, T_0, and G are defined as in Equations (5) and (6). To calculate the total amount of occupants who have entered and exited in an interval, the total occupancy before the initial time in the interval would be subtracted. This calculation can be performed at any given time to return the occupancy at that moment. This data can saved and compared against other processed days for statistical analysis.

3.7 Visualization

To visualize the results, an affine transformation was performed. The transformation takes the original camera angle and warps the perspective to an overhead view that can more easily correspond to a floor plan. This is a lossy transformation as some of the information is lost (pixels that do not belong to the floor) and some pixels are enlarged and warped, distorting the information in the frame. As this process is for visualization only and not for the actual processing, these discrepancies are tolerated. An example of results from the affine transformation is illustrated in Figure 9.

The technique for an affine transformation is a matrix operation, meaning that any coordinate returned extracted from the network can be converted to a visualized coordinate via a simple matrix multiplication. The affine transformation process is described by

$$\begin{bmatrix} r_x \\ r_y \\ r_s \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & b_1 \\ a_3 & a_4 & b_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \tag{8}$$

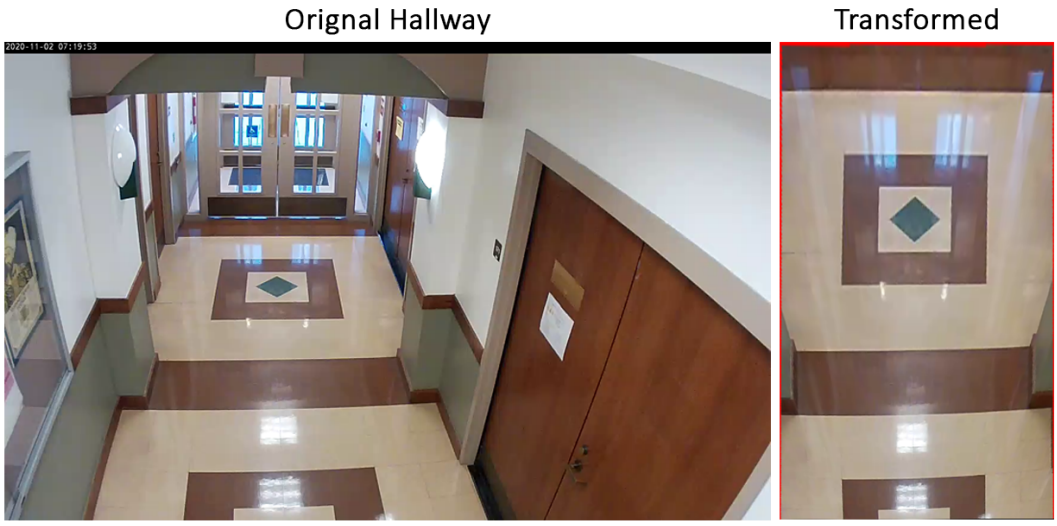


Figure 9: An example of the affine warp process for visualization

where r_x and r_y are the resulting affine-transformed coordinates, r_s is the scaling value for the affine transformation, x and y are the input coordinates (returned by the Mask RCNN), and the matrix contains factors that translate values from the original frame to the affine projected frame. The a components control scaling and rotation, the b components control translation, and the values in the bottom row are zeros and ones due to it being an affine transformation [14]. The values for the a and b components are calculated by inputting four pairs of coordinates, matched between the video and the floorplan to the function cv2.get-perspective [15]. With this transformation, any coordinate of an occupant can be translated into this new image. These can be further translated onto a map of the building for a total

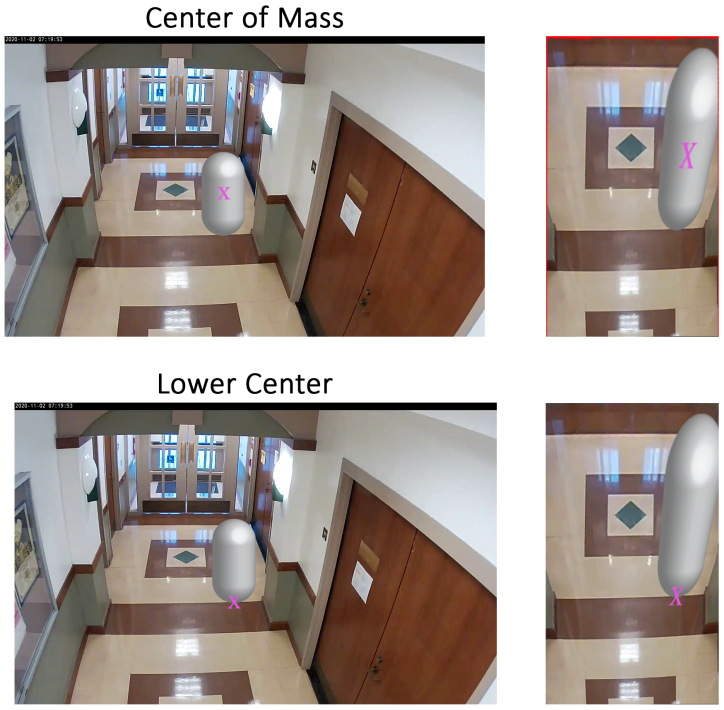


Figure 10: An example of the affine warp process for visualization

visualization of the entire building.

As described before, the lower middle point of the person was chosen in this step to prevent this visualized data to be further skewed. Figure 10 shows a representation of an occupant with different coordinate locations chosen. The first example, center of mass, shows that the resulting translated coordinate incorrectly describes the occupant's coordinate in the top half of the image. When a coordinate is chosen at the middle bottom of the person, the actual location of where they stand is a better approximation to where their feet would be from an overhead view.

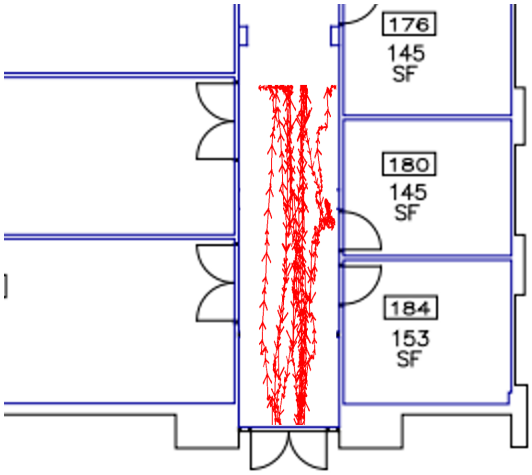


Figure 11: An example of path data being transposed onto the map of Goddard Annex.

With this transformation, the path of the person can be saved and transposed onto the floor plan of the building. Some operations such as flipping and resizing are necessary in order to correctly orient the paths to the corresponding camera angle. Figure 11 shows an example of this near the South East exit of Goddard Annex. A person entering their office can be seen as the coordinates gather by room 180 while they are opening the door. The blind spot of the camera can also be seen where the paths abruptly end and the occupant travels into an area not covered by the cameras.

4 RESULTS

This chapter begins with a presentation and discussion of data gathered over a week. The shortcomings in the collected data are described, as well as the shortcomings in the methods of processing occupancy data. This is followed by occupancy data from each of the hallways and what patterns emerge when examining the data on a per exit basis. Next is a description of the effects that the preprocessing had on the accuracy of the coordinate extraction. After this, a description of the effect of different confidence values in the Mask RCNN network is analyzed. The chapter closes with an example visualization of an overhead view of occupant paths as well as the results of some test cases.

4.1 Week of Occupancy Description

This week of data chosen for analysis was the week of January 24, 2022 to January 30, 2022 for advantages it has over other potential weeks of data. The first consideration is that it occurred during a typical full week of the school year with no holiday breaks. The building is accessible during holidays for those with an office and outer door keys, however, the occupancy would be skewed lower than normal due to most choosing to take to the holiday off. The next point of consideration is when this week took place with respect to the school year. The chosen week occurs during the first full week of school, where the volume of

occupants would be expected to be higher due to the higher demands of work that occur during the beginning of the year, with professors taking more time to set up coursework, and a higher frequency of students discussing coursework with professors in their offices. The final point of consideration is the consistency in the data set. The camera system is susceptible to footage corruption and improper saving of data. These corruption events can be somewhat frequent depending on the state of the cameras, meaning some weeks that would also be valid choices are rendered useless by footage corruption during important times of the day. This week in particular only had two such corruptions happen to the video. The specifics of the corruption in the data is further elaborated on in Section 4.5.

4.2 Occupancy Analysis for a Week

The total occupancy as shown in Figure 12 was computed every 15 minutes for the period of 5:00 am to 12:00 am. This sample rate and time range were chosen for visualization purposes, as the actual process itself does not have to restrict itself to calculate occupancy over this period. The method can return the estimated occupancy down to a one-second interval. There are also no entries or exits before the interval chosen, so the 5:00 am choice was also made to better focus the analysis.

Interesting behavior regarding the habits of occupants in this particular

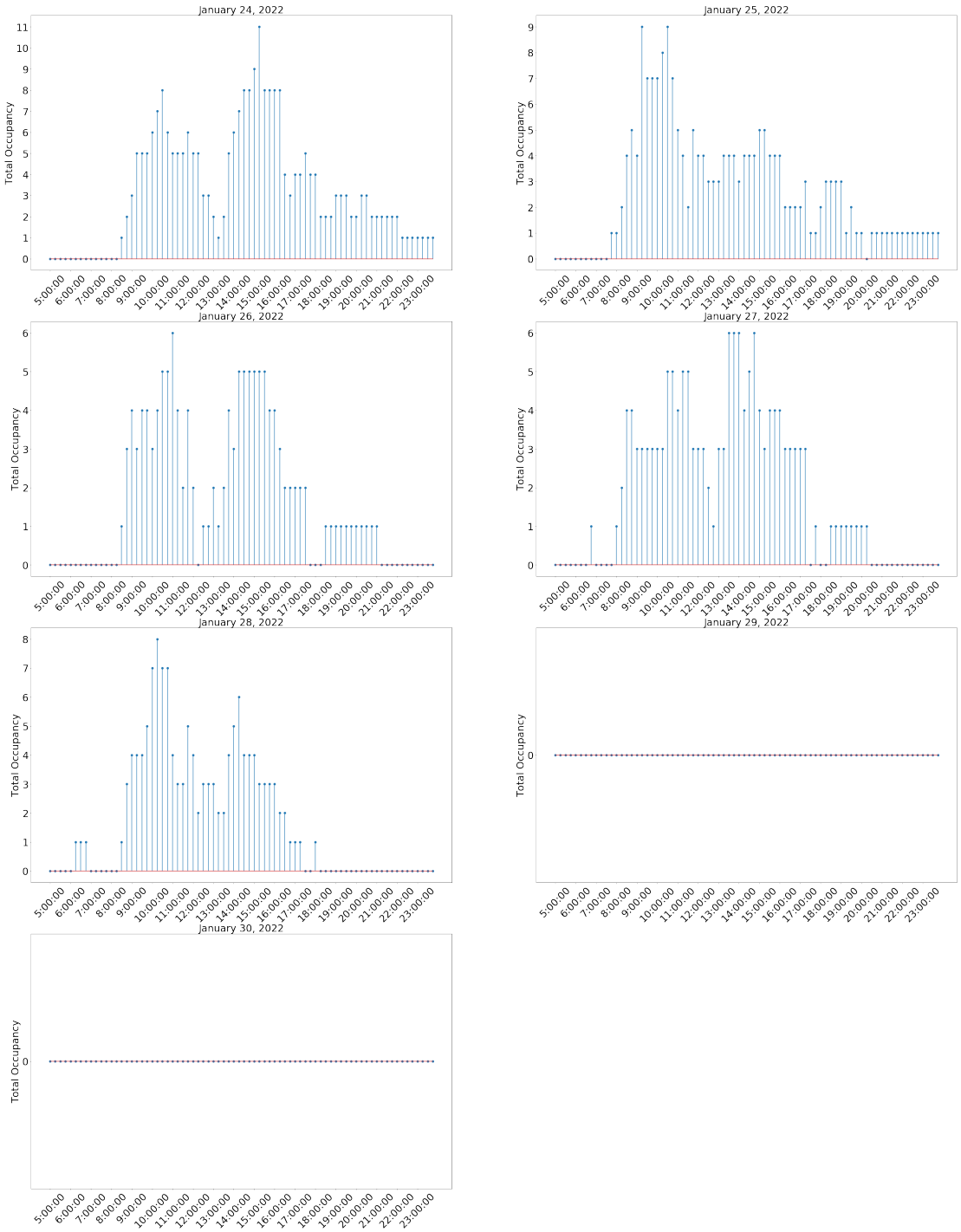


Figure 12: Week of occupancy data from January 24, 2022 to January 30, 2022.

building can be gathered from these plots. The first point of interest is the upward trend of occupancy corresponding with the start of the workday around 8:00 am. The next point of interest is the apparent dip in occupancy that occurs around midday. This dip corresponds with a standard lunchtime, as well as the point at which many classes the department holds occur. After this dip, a second wave occurs when occupants return from lunch or class and trickle in. The overall volume of the occupants starts off at a higher volume and slowly decreases over the span of the week with the exception of Friday trending upward in total occupancy compared to the trend of the previous two days. The weekends appear to have no occupancy at all, however, this is only true for the 30th, as the 29th did have an occupant who entered and left in the 15-minute sample rate so the plot itself does not show the event. If plotted at a higher cadence, there would be a point indicating an entrance 7:49:40 am and a point at 7:55:50 am indicating an exit. This is not apparent in the graph, as the total occupancy is tallied at 7:45:00am and again at 8:00:00am, and the occupant who entered left within this window meaning no occupancy is returned.

4.3 Errors in Occupancy Analysis

There are some notable errors that occur in the processing that can be seen in Figure 12. The first major indicator of an incorrect occupancy count is if the occupancy total is not zero after around 10:00 pm to 11:00 pm when all occupants

are likely to have departed. This is not the case in the plot for January 24 and January 25. This is due to a difficulty that occurs when the processing is handling two or more occupants entering the frame at once and those occupants obscure one another. This is where most of the issues with the occupancy analysis are found and is further elaborated on in Section 4.7, where it is shown that a person's coordinates are not handled appropriately in such scenarios. The issue of handling two occupants' coordinates is further exacerbated by the blurring of the image which averages pixels of the occupants together resulting in a more uniform area in the image. In camera feeds where the occupant takes up less of the frame, such as the exit being located far away from the cameras, the network has less data to analyze what is in the frame. This is what happened in the case of January 24 in the North hallway. Specifically on the 24th, three occupants left through the North exit at the same time, and since their time and spatial coordinates overlapped, one of the occupants' paths was not calculated correctly causing the occupant's exit to not be counted.

Another issue is in how the paths are decided. When using only the initial and final positions of a person's coordinate path, there is a chance those pixels end up in the wrong boundary of an image. This is what happened on the 25th. An occupant was in the exit of an image, but the network did not place their coordinate in the decided boundary for the exit due to a dropped detection or an error in the network's detection that misplaces the coordinate. Due to this,

the network cannot make a decision about what their path was and drops the data from the occupant's path, despite containing a coordinate in their path that was inbound of the exit. While the method described in this thesis works for the majority of cases, this flaw managed to create a miscount in exits showing an extra occupant that was not actually in the building.

Overall, the total occupancy was quite accurate with the expected amount of traffic in the building that was tested. The margin of error on most of the days appear to be no more than 1 to 2 inaccurate counts, as seen in the hanging extra occupants seen in Figure 12 on the day of January 24th and January 25th. A way to counter this effect is to have the count toward the end of the day reset to reflect that often no occupant is in the building. This is an incomplete solution in the case of an occupant staying past the time when all occupants have typically left, which in Figure 12 shows to be around 10:00 pm. An occupant staying the night, however, is less likely to happen as opposed to the standard case where all occupants have left by 10:00 pm.

4.4 Data from Each Hallway

The total inflow and outflow of each hallway is dependent on the day. Figure 13 gives a visual representation of the flow of exits and entrances where the radius of the circle increases with the volume of entrances and exits. In a situation where all occupants have left the building by midnight, the total area of the circles labeled

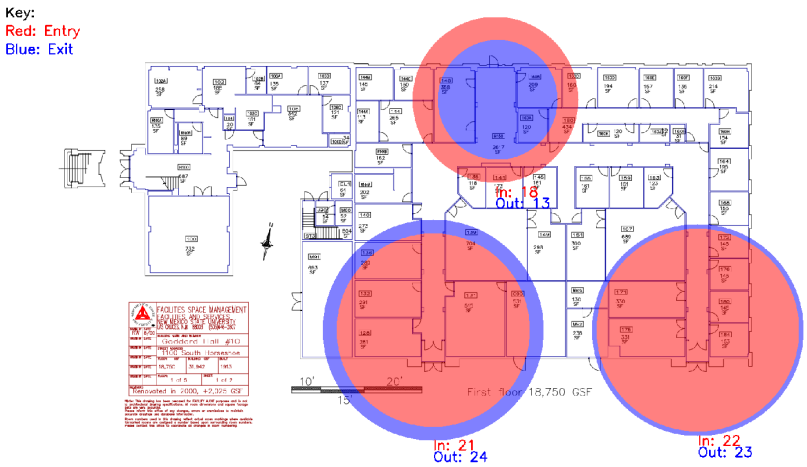


Figure 13: Heat map showing the amount each exit is used on January 25, 2022.

IN in Figure 13 should be the same as the area of the circles labeled OUT. This is not the case in the figure shown, as there was a missed exit in the occupancy count that cause the occupancy to remain at one after all occupants have left.

This analysis gives some idea as to what the most preferred entrance/exit of the building is. Before running this analysis, it was expected that the most trafficked entrance/exit would be the North exit, as it is the door closest to a large parking lot and is located near the majority of the commonly used offices. However, after running the analysis, this is shown not to be the case. It appears that both southern exits experience a higher volume of traffic. Upon inspection of the footage, the likely reason for this behavior can be observed. While many occupants do choose to start their day by entering through the North exit, the southern exits lead to the neighboring building of Thomas and Brown where most of the occupants travel to teach or work. The southern exits also open up to

a more trafficked part of campus. This behavior can also be seen in Figure 13, where the North entrance has a higher volume of occupants entering than exiting, where the southern exits have a higher volume of exits. Other days of the week also show the same behavior.

4.5 Shortcomings in the Data

While the data for the week was consistent with the expectation of a building of this size, it does not account for some of the shortcomings in the data set. The most obvious shortcoming is that the initial motion detection performed by the camera system is not guaranteed to record all the events that happened within a day. This is an inherent flaw in using preexisting camera systems to track occupancy in general. Cameras are susceptible to failures, which can be further compounded by including motion detection built into the camera. The motion detection specifically can be a problem for this kind of data as it adds another layer of potential data loss beyond just footage being corrupted, as a motion detection system can drop an occupant mid traversal, or simply not record them at all. An example of this is frequent in the cameras that were excluded from this process. The camera feeds from the North West and North East hallways suffered from frequent dropping of recordings while occupants were still traveling through the building. The North West hallway in particular covered a large area of the building, which in turn causes the occupants in frame to appear much smaller

when compared to the total frame. When occupants passed a certain part of the hallway, the motion detection built into the cameras often dropped them. The data gathered from these hallways is useful for determining where in the building the occupants are located, but for a total occupancy count, it is not necessary as it can be calculated with only the exits. For this reason and the inconstancy in the footage they, are not used.

On January 28, two camera feeds experienced a corruption of footage between 7:35:06am to 8:26:31am in the South East exit, and between 6:42:12am to 8:39:06am in the South West exit. The total occupancy during these times is unknown, as without every exit covered the total occupancy cannot be correctly calculated. This can be seen in Figure 12 on the plot for January 28, there is a lack of any data in this time span. There is also corrupted data in the South West hallway preventing footage from being saved on January 30th and was unable to be analyzed. In this sense, the count on that day cannot be guaranteed to actually have a zero occupancy, as there could potentially be many entry and exit events that could have occurred during this outage in the footage. While unlikely that all occupants decided to enter and leave in a high volume during this time, it reveals a major flaw in this process, as a missing camera will under-report the total occupancy. A good implementation of this solution to occupancy is to ensure a solid feed between the camera and the script handling the occupancy.

4.6 Optimization of Performance

To see what effect different values had on the system's ability to decide total occupancy, two important parameters of the process were changed. The first parameter that was changed was the threshold value in the preprocessing stage. This value determines what total amount of movement that is necessary in frame to have the network perform a detection. The second parameter that was changed was the minimum confidence required to have the object in frame be recorded as an occupant. The Mask RCNN network gives a confidence score stating how confident it is that the object is detected is the class it is. The higher the confidence the more sure the network is the object is the class it says it is. An extra threshold can be added that prevents a detection below a certain confidence to not have its coordinates recorded.

4.6.1 Metrics

To decide how the preprocessed data differs from the unprocessed data, the true positive and false-negative rates are calculated. The true positive rate in this context is defined as where the preprocessed and unprocessed data agree on a detection, i.e., the unprocessed data is used as the baseline ground truth in this context. The false negative rate is the rate at which the preprocessed data misses detections that exist in the unprocessed data due to the secondary motion detection filter not allowing the network to run a detection. The true

negative rate is the rate at which the preprocessed data agrees that an occupant does not exist in frame. For calculations that involve changing the confidence, there is a chance for a false positive. A false positive, in this case, would be an extra detection in one set of data that is not in the other. This can occur when the confidence value of the network is lower in one data set than the other. The dataset used as the ground truth for threshold-based statistics is the detections run without any type of thresholding. The dataset used as ground truth for the confidence-based statistics was 97 percent. These rates are defined in Equation (9),

$$\begin{aligned} TPR &= \frac{TP}{TP+FN} \\ TNR &= \frac{TN}{TN+FP} \\ FPR &= \frac{FP}{FP+TN} \\ FNR &= \frac{FN}{FN+TP}, \end{aligned} \tag{9}$$

where TPR is the true positive rate, FNR is the false negative rate, TNR is the true negative rate, FPR is the false positive rate, TP is a true positive, FP is a false positive, TN is a true negative, and FN is a false negative.

4.6.2 Preprocessing Optimization

The preprocessing step described in Section 3.2 is meant to shorten the length of time it takes to detect occupants. This process is destructive in nature, meaning data will always be lost. This preprocessing step makes the network skip frames,

some of which may contain occupants, so the preprocessed data should never contain more detections than the unprocessed. In a statistical sense, when comparing preprocessed and unpreprocessed data, there can only be true positives, true negatives, and false negatives. A false positive implies that, in the data that was preprocessed, there was an additional detection on footage with less data than the unprocessed version of the data. This is not possible since the preprocessed data will only remove detections by skipping frames. In this way, the valuable metrics for testing how much the preprocessing affects the data are the true positive rate and the false-negative rate.

An important parameter of the preprocessing step is specifying the threshold of the motion detection. The threshold in this case is defined as the amount of pixels in the thresholded difference image illustrated in Figure 5 that must be present to denote motion and thus the need for a detection by the Mask RCNN. Thresholding the data in this way means that some data is lost. In order to quantify the amount of data lost, various thresholds were chosen and tested against the data that was not preprocessed. For the day of January 24, 2022 the results are outlined in Table 2. The Mask RCNN network is configured with a minimum confidence score of 0.97 and a threshold value that varies as specified in the table.

There are a few interesting behaviors that arise from the analysis summarized in Table 2. The first general pattern is that as the threshold increases, the length of time to process decreases, but the accuracy also decreases. This is to be expected

Table 2: The amount of data lost and detection performance for various motion detection threshold values for footage from January 24, 2022, where l_u is the total footage length (in mm:ss) for the unprocessed data and l_p is the total footage length (in mm:ss) for the preprocessed data. Recall that the TPR and FNR are defined relative to the detections on the unprocessed data.

Hallway	Threshold	TPR(%)	FNR(%)	l_u	l_p	Time Reduction(%)
North	50,000	95.37	4.63	29:17	17:07	41.55
North	100,000	94.15	5.85	29:17	14:40	49.89
North	200,000	89.42	10.58	29:17	11:27	60.88
South East	50,000	99.93	0.27	12:19	08:55	27.56
South East	100,000	99.73	0.27	12:19	08:35	30.25
South East	200,000	99.32	0.68	12:19	07:55	35.67
South West	50,000	98.74	1.26	08:33	04:32	46.87
South West	100,000	97.81	2.19	08:33	04:13	50.54
South West	200,000	95.93	4.07	08:33	03:46	55.81

as the network is extracting fewer coordinates the higher the threshold is due to more activity needing to occur between frames to trigger a detection. The North hallway suffered the most from the preprocessing in regards to accuracy against the unprocessed file. The reason for this is that the North hallway has the farthest camera placement for the exit it observes, meaning that occupants in frame take up a smaller portion of the frame when near the exit. This means that more frames that do contain a subject will be skipped due to the sum of the threshold image not being enough to trigger the detection. The next worst performer is the South West hallway. The accuracy for this hallway is similar to that of the South East hallway, but the camera is a little further from the exit than the South East hallway. The time reduction follows an expected pattern of decreasing as the threshold is increased.

The next consideration is how a change in the motion detection threshold changes the total accuracy of occupancy calculations. When coordinate data is lost, it can change the characteristics of an occupant's path, especially if this drop occurs in the entrance or exit part of the image. Due to the high TPR rate for the exits, the overall occupancy for the day analyzed was not changed even with the highest threshold. This does not mean that the final output data was not completely unchanged, as there are some extra events or missing events. However, these events were often the path being cut in the hallway, meaning it is treated as a hallway-to-hallway transition resulting in no change in occupancy. After this path is dropped, the rest of the path data is then picked up from that point in the hallway to whatever the final destination actually was, meaning that the overall net occupancy change was zero. Despite this, for all full week results included, the total occupancy was calculated using the smallest threshold of 50,000, as the potential 50% reduction in time was less desirable than the potential of lost data when using higher threshold values.

4.6.3 Optimizing Network Detection

The confidence of the Mask RCNN is the amount that the network is confident that a given portion of the image is a certain class. Without putting a minimum on confidence score, any object that the network thinks is most likely a human out of all the classes it was trained on is decided to be a person and their coordinate

is recorded. This is not desirable since, if a minimum confidence limit is not chosen, many objects that are not occupants have coordinates extracted. The system does have a buffer built in to handle anomalous detections that ignores paths shorter than 15 points, but a large number of anomalous detections tend to build off one another and create invalid paths too long for the buffer to take effect. To combat this, a minimum confidence score can be chosen so only subjects the network is sure about to a certain percent are allowed to have their coordinates extracted. An initial minimum confidence score of 0.97 was chosen and used to run the week of occupancy analysis in Section 4.2. This score was chosen by inspecting the confidence value that was associated with a valid extraction of a person's coordinate in some sample footage. Values were between 0.95 and 0.98, so the value 0.97 was chosen and used in the analysis of an entire week. Different confidence values, however, will result in substantial changes to the results.

To see what the effect the confidence had on the system as a whole, the data for January 24, 2022 was processed using no restriction on confidence as well as a strict confidence of 0.99. The results for these two confidence scores were compared to the results for a confidence score of 0.97 as shown in Table 3. A true positive in this sense would be both the 0.97 confidence data set agreeing that the same detection data is present in the lower or high confidence values in a frame. A true negative is both data sets agreeing that there is no occupant data present in a frame. A false positive would be the lower or higher confidence data detecting

Table 3: Effects of Mask RCNN confidence c on detection performance. The performance metrics (TPR, FNR, TNR, FPR) are computed relative to the detections with a confidence of $c = 0.97$.

Hallway	Confidence c	TPR(%)	FNR(%)	TNR(%)	FPR(%)
North	No Restriction	95.37	4.63	99.2	0.8
North	0.99	84.79	15.21	100.0	0.0
South East	No Restriction	99.93	0.07	99.2	0.8
South East	0.99	93.73	6.27	100.0	0.0
South West	No Restriction	98.74	1.26	96.3	3.7
South West	0.99	81.78	18.22	100.0	0.0

another occupant that is not present in the 0.97 confidence data, which cannot happen in the case of the stricter confidence value since it only removes data, but is frequent in the unrestricted confidence due to more the network being less strict about what can be considered human. A false negative, in this case, would be the network detecting an occupant in one data set, and the occupant does not exist in another data set. This happens more commonly in the stricter 0.99 confidence data where the network will drop a detection that the 0.97 confidence data set has. This is shown in the FNR rate in Table 3 for 0.99 confidence where the FNR is much higher than the no restriction data. A false negative can occur in the unrestricted confidence case, and this happens due to the network deciding that object in frame is not part of the human class and therefore a coordinate is not extracted.

Table 3 shows that changing the confidence greatly impacts the performance of the network, in terms of what is a valid choice for a person in frame. An unrestricted confidence yields a lot of extra detections not present in the 0.97

confidence results. While some of these coordinates could be valid actual coordinates, running the analysis on data with unrestricted confidence, it is clear that some objects in frame have a vaguely human shape that the network thinks is most likely to be a person than any of the other classes it was trained on. The stricter threshold of 0.99 has the opposite effect, as the network needs to be very confident that the object in frame is indeed human. This means that event occupants in frame may not be counted due to their confidence value being to low.

It is again important to consider how this affects the ultimate occupancy results. Unrestricted confidence had a large effect on the final occupancy analysis due to the large amount of superfluous defections from non-occupant coordinates that were extracted. Although the data may suggest there was little impact due to high TPR, the FPR, in this case, is very much a determent to the use of an unrestricted confidence. The problem lies with the potential harm that a false detection can cause. Figure 14 shows these anomalies, particularly in the second half of the day where there are many incorrectly counted occupants. This major issue is due to the coordinates of anomalous detections being close enough to a valid occupant such that an incorrect coordinate will be associated with a valid occupant. There could also be more coordinates than occupants, and thus a new anomalous occupant is created and given coordinates. With enough of these anomalies, the path of a person could be ended preemptively or assigned

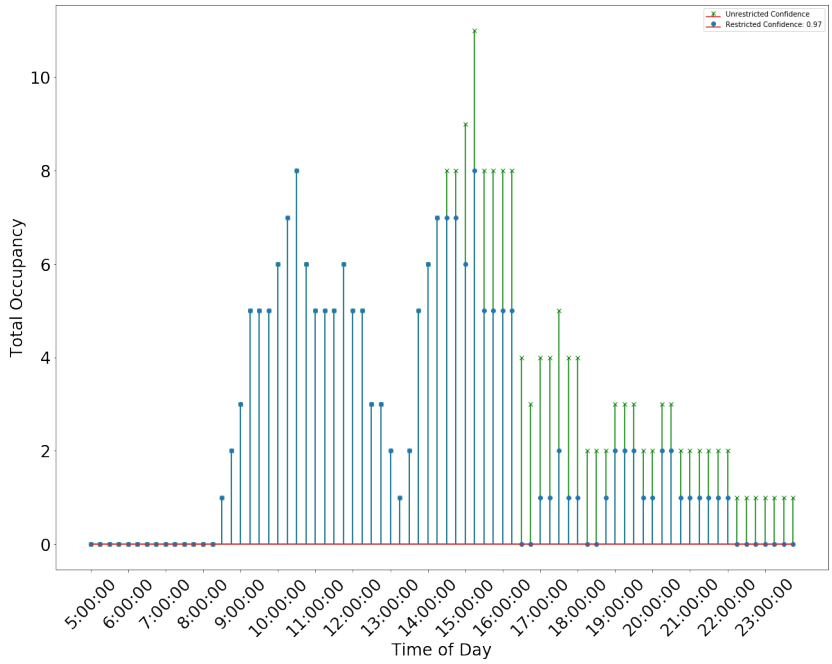


Figure 14: The difference in occupancy data when the network confidence is completely unrestricted. Many superfluous detections are present that cause a major shift in occupancy data.

to the wrong location. With stricter confidence, there are substantially fewer anomalies, so a higher confidence threshold is much more desirable than a lower confidence.

4.7 Scenarios and Visualization

The week of data chosen had events that caused the algorithm to have unexpected issues in processing. Some of these issues were minor and did not affect the occupancy results. Some other issues contributed to miscounts causing errors in the data.

(a) Visualisation of occupant's path. (b) Occupant obscured in frame.

Figure 15: An occupant is obscured in frame causing inconsistencies in the visualized path.

4.7.1 Occupant Obscured in Frame

A frequent occurrence that is worth considering is a person being obscured by an object in frame. This situation is frequent with janitorial staff, as they frequently carry equipment large enough to obscure parts of their body. They are a consistent source of data as they make a round trip in the building every morning around the same time.

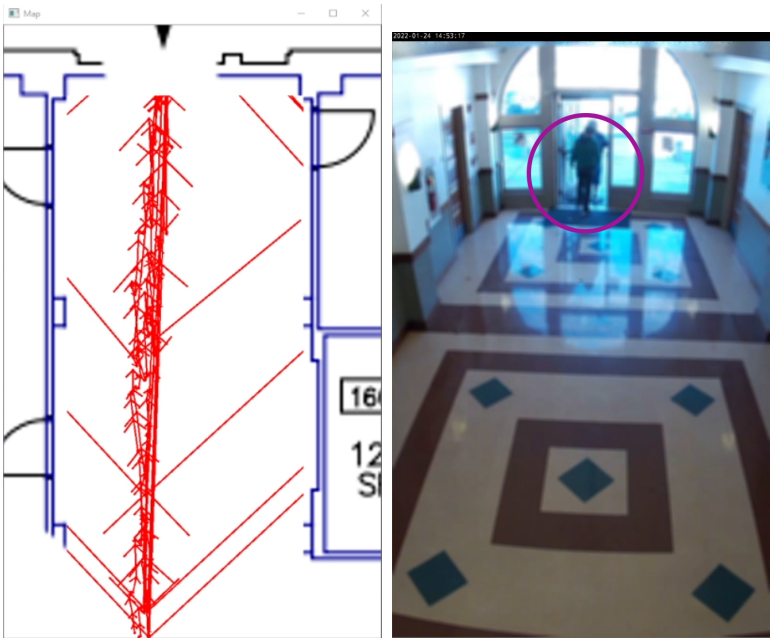
Figure 15 shows a member of the janitorial staff cleaning out trash bins in the office areas. Highlighted in the purple circle of the path view in Figure 15(a) is an area of particular interest because it is when the janitor is standing still with the trash bin covering their lower portion. If the path was entirely accurate to real life, the path in this area would not deviate as far up the hall as it appears to, as the occupant's motion is mostly side to side in frame. This is due to the

lower coordinates of the detection being extracted; as the garbage bin obscures the actual lowest point of the occupant, the bottom coordinate is chosen somewhere on their waistline instead of their actual feet that are obscured by the trash bin. Since their bottom coordinate is picked at waist level instead of foot level, the affine transformation then records them as being further up the hall. This is similar to the effect seen in Figure 10. This did not affect the overall occupancy count at this moment, however, as these minor deviations are not substantial enough to cause them to be miscounted.

4.7.2 Occupants Overlapping

Unlike the scenario with the janitorial staff in Section 4.7.1, there was an event with overlapping occupancy that did cause a miscount. Occupants who walk in a line normal to the camera sensor will cause problems with the processing. When more than one occupant walks in a straight line, they obscure the occupant or occupants ahead or behind them depending on what direction they walk with respect to the camera. The effect of this is most prevalent when the occupant behind another is taller or larger in stature and completely obscures the other occupant. If an occupant is obscured for the initial or last part of their path, their final position has the potential of being missed and causing an error in occupancy count.

Figure 16 shows this exact situation. The purple circle in Figure 16(b) shows



(a) Path of obscured occupants (b) Occupants appearing in frame

Figure 16: Occupants walk in single file towards the exit and cause a miscount

the moment that the miscount occurred due to one occupant being behind the other and being completely obscured. The technique used to normally prevent this is having the region which constitutes the exit extend further in front of the exit; see Figure 8. This method did not prevent this from happening in this situation, however, because the occupant was obscured by the other for a prolonged period of time. This type of obscuring is infrequent as there have to be two occupants walking single file within the exit region. These factors and the added challenge of this occurring in a hallway with where the occupants appear very small in frame caused this particular miscount.

Figure 16(a) shows lines coming out from the path data. These lines are the arrowheads from the arrows used to draw the path of the occupants. These large

branching lines are the arrowheads and represent a large magnitude change in an occupant's path. These lines show up frequently in this figure due to the occupant ahead being obscured and therefore missing coordinates that would normally be assigned to their path, and when they are seen again in frame their coordinate is much farther than previous causing this jump and the large arrowhead to appear.

4.7.3 Three Occupants

A scenario that could cause potential problems is when a hall experiences a high amount of traffic at once. While relatively uncommon for this data set due to the overall traffic of Goddard Annex, there are times a large amount of occupants enter the building simultaneously. This has the most detrimental effect of overlapping in frame, as well as problems in the distance-based selection of coordinates (Section 3.4). When many occupants walk close to each other, there is a chance that the next upcoming coordinate is swapped with another occupant.

Figure 17 shows the scenario where coordinates are swapped between occupant paths. This image data was taken in the middle of an exit event so the occupants have not quite completed their transition. Figure 17(a) shows many breaks in the paths that occurred. This is due to the close proximity of the occupants causing the closest coordinate to be associated with the wrong occupant. This is further compounded by the overlapping that occurs with the two occupants on the left.

(a) Three occupant path data (b) Occupants in frame

Figure 17: Three occupants enter the frame and begin overlapping and occasionally coordinates are swapped between occupant paths.

In contrast to Figure 16, the occupants appear larger in frame which does aid in this specific event correctly labeling that all occupants have left. This overall larger frame was much more conducive to the network correctly assigning the right event. This event was counted correctly despite the complications that were caused by the paths.

4.7.4 Minor Inconsistencies

There are a few inconsistencies that arise from processing paths on a coordinate system based on the upcoming coordinate that is closest to the occupant. The most prevalent issue is that of occupants switching coordinate paths with another. This is due to the process not giving any particular attribute besides coordinates to an occupant detection. An occupant switches path data with another occupant when the occupants are close enough that the distance between their last coordinate and the other person's incoming coordinate is

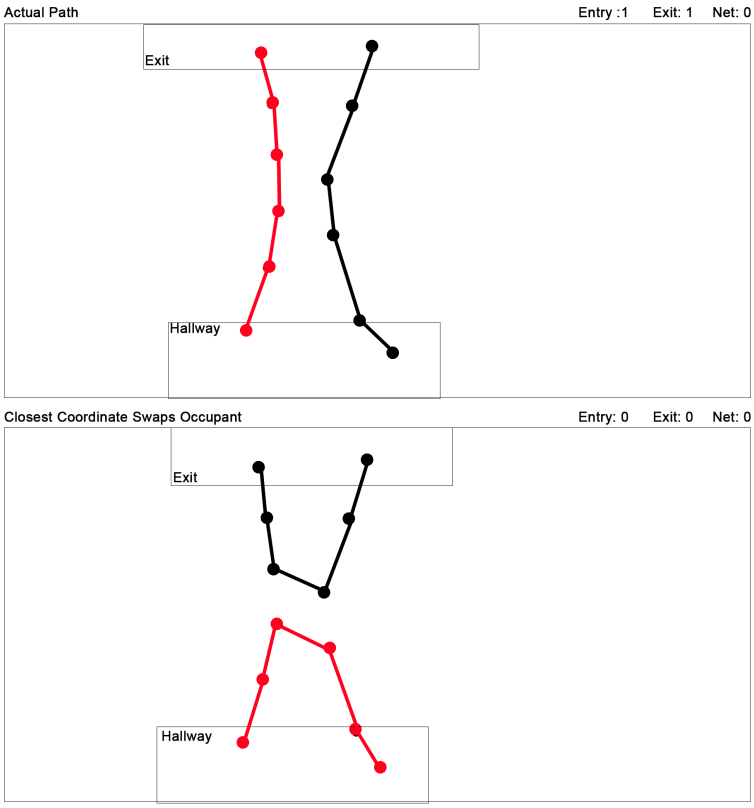


Figure 18: Diagram of swapped coordinates causing inconsistency

closer that person's next actual coordinate is to their last. This switch means that the true coordinates of the other person are assigned to the wrong occupant. Since the occupancy is calculated after all occupants left the frame, the total sum does not change if the actual correct paths were calculated. This is not exclusive to occupants walking in opposite directions, as occupants walking side by side with one another can also suffer from this inconsistency. The difference, in this case, is less obvious since the coordinates pass back and forth over a small distance, and the end result and net occupancy remains the same.

Figure 18 shows two occupants approaching each other and when they reach a

point closer to one another they swap paths. This path swap causes the occupant entering from the hall to switch with the occupant entering from outside the building. This results in the processing seeing an exit to exit transition and a hall to hall transition. The net result of this is the same as if they did not swap paths, as a hall to hall and an exit to exit transition both result in a 0, while an exit to hall and hall to exit result in a +1 and a -1, which is also net 0.

5 CONCLUSIONS AND FUTURE WORK

This thesis provides an occupancy tracking technique using preexisting security cameras in a building. It details what methods were used, as well as what optimizations were made to make the computation time more reasonable. It also describes how the data used during calculation can be used to visualize the path of occupants in the building. This thesis also elaborated on the issues associated with using camera-based solutions, such as subjects obscuring each other and handling issues that arise from video footage and data corruption.

The analysis of a week of footage resulted in interesting patterns that emerged from occupants. These patterns showed common times of entry and exits, as well as typical times when the building was at its most populated and when the occupancy dwindled. There was also an analysis of how the process performed with different network parameters, as well as with different preprocessing parameters. Finally, the results of the visualizations and analysis of special cases were performed.

Overall, the methods proposed in this thesis provided decent estimates for occupancy, however, the data itself can never be guaranteed to be entirely accurate. The main weak point of the system was occupants obscuring one another. This problem in particular caused most of the incorrect occupancy calculations. The best-case scenario for this method is a camera with decent

quality, occupants taking up a large portion of the frame, and occupants walking side by side instead of in a single file line. This cannot be guaranteed, but the methods used in this thesis were shown to be robust enough to handle cases outside of this ideal scenario.

Further application of Mask R-CNN's instance segmentation abilities could provide more accurate occupancy data by refining specific portions of an occupant. These applications include using the pixels that accurately encompass their feet for more accurate visualizations, using the properties of the mask to decide features of an occupant, or using the instance segmentation functionality to see where two occupants are overlapping in frame. The process could also be adapted to real-time building occupancy calculation with sufficiently capable hardware. Future applications of this method could make decisions about boundaries automatically by identifying objects in frame such as doorways. It is also possible to include a process that takes camera feeds from non-exit areas to give some idea of the locations of occupants. A prototype of this process was included initially but was dropped due to issues with the cameras that cover this part of the building not correctly saving data and the occupants being too small in the frame. Future applications may also include more involved processing steps to handle camera feeds with less than desirable image quality, as well as better methods for handling occupants that appear small in the frame.

REFERENCES

[1] T. Kenneth, "Personalization of smart-devices: Between users, operators, and prime-operators," *Operators, and Prime-Operators (January 6, 2021)*, vol. 70, 2021.

[2] C. A. Valhouli, "The internet of things: Networked objects and smart devices," *The hammersmith group research report*, vol. 20, 2010.

[3] J. Zou, Q. Zhao, and R. Cong, "Occupancy measurement by object tracking at building entrances," in *2017 36th Chinese Control Conference (CCC)*, 2017, pp. 11 086–11 091.

[4] W. Shen and G. Newsham, "Smart phone based occupancy detection in office buildings," in *2016 IEEE 20th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, 2016, pp. 632–636.

[5] C. Luppe and A. Shabani, "Towards reliable intelligent occupancy detection for smart building applications," in *2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE)*, 2017, pp. 1–4.

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[7] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[8] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2961–2969.

[9] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in Neural Information Processing Systems*, vol. 28, 2015.

[10] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[11] W. Abdulla, "Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow," https://github.com/matterport/Mask_RCNN, 2017.

[12] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *European Conference on Computer Vision*. Springer, 2014, pp. 740–755.

[13] "Python Tesseract repository," https://github.com/madmaze/pytesseract, accessed: 2020-03-04.

[14] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 4th ed. Pearson, 2018.

[15] Atul and Kang, "Cv2.warpperspective()," Nov 2020. [Online]. Available: https://theailearner.com/tag/cv2-warpperspective/