A Zone-Level Occupancy Counting System for Commercial Office Spaces Using Low-Resolution Time-of-Flight Sensors

Hao Lu[a], Arunas Tuzikas[a], Richard J. Radke[a]

[a]*Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, 110 8th St, Troy, NY, 12180, USA*

Abstract

Understanding the locations of occupants in a commercial built environment is critical for realizing energy savings by delivering lighting, heating, and cooling only where it is needed. In this paper, we present an indoor occupancy counting system using a sparse array of inexpensive, low-resolution, and privacy-preserving time-of-flight sensors. We develop and validate an algorithm for zonal occupancy counting that can deal with multiple people walking underneath the sensors in arbitrary directions, and evaluate the system both in realistic simulations of office spaces and in a real-world installation. We found that our system has an error rate of around 0.4%, resulting in highly accurate person localization and zone counting using only a few sensors per space.

Keywords: Indoor Occupancy Counting, Time-of-Flight Sensors

1. Introduction

Building energy consumption has become the main source of urban energy usage [1]. The major contributors to energy consumption in commercial buildings are lighting and Heating, Ventilating and Air-Conditioning (HVAC). Finding methods to minimize these costs will not only create significant energy savings for businesses, but also reduce emissions attributed to power generation. However, thermal comfort and air quality should not be neglected. Many studies [2, 3] show that understanding where and when spaces are occupied is the key to balancing thermal comfort and energy savings.

Email addresses: `luh6@rpi.edu` (Hao Lu), `tuzika@rpi.edu` (Arunas Tuzikas), `rjradke@ecse.rpi.edu` (Richard J. Radke)

Some studies [4] used fixed occupancy schedules to control building operations. However, as more office occupants adopt flexible working hours, demand-driven control strategies that enable lighting/HVAC systems to adapt to dynamic changes of occupancy become more efficient [5, 6]. Hence, accurate real-time zone-level occupancy counting information plays an important role in smart rooms and energy-efficient building management systems (BMS).

Deploying sensor networks to monitor occupancy information in buildings is a common approach. Designers must make cost and practicality tradeoffs when choosing the sensor modalities for an occupancy counting system. Low-cost sensors that can easily cover a whole room such as CO_2 sensors, acoustic sensors, and RF sensors have limited resolution and counting accuracy. High counting accuracy sensors like standard cameras and depth sensors have limited fields of view (FoV) and may be more expensive to install and maintain. Fisheye cameras are widely applied in public surveillance due to their wide FoV and high resolution, though they are generally more expensive. However, privacy considerations limit the use of any type of cameras in non-public areas of commercial office spaces.

In this paper, we propose a zone level counting system using a sparse deployment of low-resolution, low-cost time-of-flight sensors. These are installed at zone boundaries and count the entries and exits for each zone, allowing accurate occupant counting in a large office space using very few sensors. Since the sensors are small, low-cost, and self-contained, the installation and maintenance costs are low. Since each sensor only returns the distance to a small number of locations, the system naturally protects occupant privacy. Therefore, our system is suitable for environments with low crowd density and high privacy requirements, such as commercial office or healthcare spaces. Unlike other doorway counting systems that can only detect people moving in one of two directions, our counting system works well in complex environments where people can walk under the sensor in any direction. We report the results from both a real deployment and simulated environments using the same underlying algorithms, demonstrating that our system is fast and accurate.

2. Related Work

Generally, zone-level occupancy counting systems are either sparse (i.e., detecting motion in a narrow cone angle below a small number of sensors) or dense (i.e., trying to sense an entire environment with either a large number of sensors or wide field-of-view sensors). Liu et al. [7] categorized occupancy counting methods by sensor type into 5 categories: motion sensors, environmental sensors, radio frequency (RF) sensors, contextual sensors, and cameras.

Passive infrared (PIR) sensors are the most common motion sensor. Raykov et al. [8] used a single PIR sensor to detect occupants in a meeting room, but it requires a long time (20 min) to train. Wahl et al. [9] used pairs of PIR sensors installed in doorways to count entries and exits in an indoor environment, but did not discuss multi-person scenarios. PIR sensors can only detect the motion of heat sources, which limits their recognition of stationary humans. Andrews et al. [10] mounted a PIR sensor on a moving platform to artificially induce the motion that is necessary for stationary occupant counting, but did not study dynamic scenarios, which are very common in commercial spaces. Even though moving the PIR sensor is a solution, it will increase the sensors' energy cost and have limited counting accuracy in crowded or rapidly changing scenarios due to the one-dimensional output and low sampling rate of PIR sensors.

Environmental sensors infer the occupancy count from sound, light, temperature, or CO_2 measurements of an indoor environment. Wolf et al. [11] used stochastic differential equations to infer occupancy counts from CO_2 data. Szczurek et al. [12] fused CO_2, humidity, and temperature data to infer occupancy counts based on k-nearest neighbor (KNN) and linear discriminant (LDA) models. Li et al. [13] used indoor environmental sensor data (e.g., air temperature, air humidity ratio, air CO_2 concentration, and so on) to infer occupant count based on inverse physics based models. Valle [14] used Gaussian Mixtures and Hidden Markov Models to estimate room occupancy from audio measurements. Like PIR sensors, environmental sensors generally have low counting accuracy, because the occupant count is not directly related to these environmental factors, and they usually respond slowly.

Radio frequency sensors include WiFi, Bluetooth, ultra-wideband (UWB), and ZigBee. Many RF methods require the occupants to carry their mobile devices to be detected, which limits their application and has privacy implications, but some recent studies try to avoid this problem. Depatla et al. [15] used only signal strength indicator (RSSI) measurements to estimate occupancy numbers in a given zone. Zou et al. [16] proposed a device-free occupancy counting method based on the channel state information (CSI) transmission between two WiFi routers. Yang et al. [17] demonstrated a door counting method using COTS WiFi devices. They use one transmitter and multiple receivers to monitor the number and motion of people walking through a door. They recover time-frequency information from the WiFi signal and train a convolutional neural network for counting. Many RF methods are sensitive to non-human background movement like a door opening that would affect their performance in a complex environment.

Contextual sensors monitor chairs, computers, and even floors. Labeodan et al. [18] installed strain and vibration sensors on chairs to infer occupancy in a room.

Kaddoura et al. [19] placed pressure sensors on the floor to track occupancy in an indoor environment. Razavi et al. [20] collected 5000 households' electricity meter data for more than 18 months and used this data to predict the present and future occupancy information.

Compared to all other sensors, camera-based methods using computer vision provide the most accurate and fine-grained occupancy information. These can not only obtain occupant count [21, 22], but also occupant identity [23], location [24] and even behavior [25]. However interference between similar-looking objects and occlusions from items like furniture cause measurement errors, and issues of privacy are always a serious concern.

Some studies [26, 27] installed cameras looking down at the room entrance. The overhead installation reduces occlusions from other objects and partially mitigates the privacy issue, since the camera can only detect occupants' heads and shoulders and recognition is more difficult. To further address the privacy issue, some researchers [28, 29, 30] used depth cameras (especially time-of-flight (ToF) cameras) to detect occupancy information. The depth image contains only distances, not colors, so it can protect occupants' privacy and reduce computational complexity. However, depth cameras are usually more expensive than surveillance cameras.

To balance installation cost, privacy issues and measurement accuracy, many researchers have proposed to use low resolution sensors to detect occupancy information. Bhattacharya et al. [31] proposed a method for real-time person tracking and coarse pose estimation using arrays of single-pixel ToF sensors. Compared to this work, our ToF sensors are more sparsely installed, hence reducing the number of necessary sensors and requiring a more complicated algorithm. Mikkilineni et al. [32] used an 8×8 thermal camera above a doorway to count the number of people entering or leaving the room. They used a per-pixel adaptive threshold and region growing to extract human profiles and used logical boundaries to obtain the motion vectors of occupants, but did not provide numerical counting results. Cokbas et al. [33] proposed a low-resolution overhead thermal tripwire for door counting. Their work follows a similar process flow to ours: background subtraction, event detection, and event classification.

Compared to these approaches, our work has three main contributions.

1. We use low-resolution ToF sensors (in 8×8 and 4×4 modes) to detect occupant profiles, which are less noisy and susceptible to environmental heat sources than thermal sensors.
2. Instead of simply using a region growing method, we propose an agglomerative clustering method to distinguish different people more precisely, which is robust to multiple people moving under the sensor simultaneously.

3. While many algorithms are limited to bi-directional motion detection, our algorithm considers multiple arbitrary directions of motion, which is more suitable for open-plan offices.

3. Methods

3.1. Testbed Environments

We validate our algorithms both in a real-world physical commercial office suite and in detailed simulated environments created using the Unity game engine[1]. Unity and other 3D game engines (e.g., Epic's Unreal engine[2]) are widely used in designing algorithms for occupancy counting [30, 34, 35, 36, 37, 38]. The advantage of the simulation environment is that many hours of realistic, controllable human and sensor behavior can be recorded and compared to algorithmic results. On the other hand, running long-term experiments in real-world environments is challenging due to the difficulty of collecting independent ground truth and, in light of the pandemic, maintaining social distancing.

We carry out our experiments in two environments. The first is a physical office space that is $3.0m \times 10.7m$, denoted Office 1, and illustrated in Figure 1(a). We report results from both physical sensors installed in this office space and an accurate "digital twin" we built in Unity. We also created a second, more complex environment, denoted Office 2 and illustrated in Figure 2(a), that only exists in Unity simulation. This space is roughly $14.6m \times 18.6m$ and has an open layout and many potential human paths.

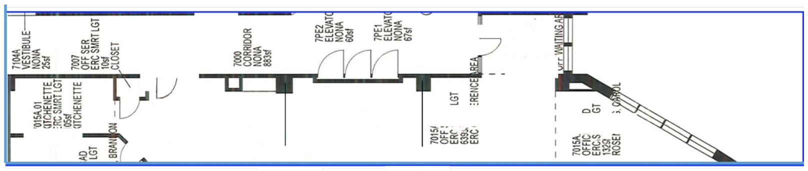
We divide each space into HVAC/lighting control zones indicated by the walls and red lines in Figures 1(b) and 2(b). There are 3 zones in Office 1 and 10 zones in Office 2. The goal of this paper is to accurately count the occupants in each of these zones at all times, using a sparse array of sensors.

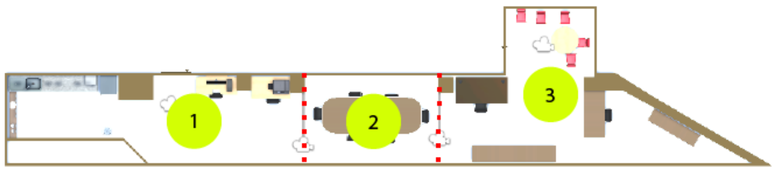
3.2. Time of Flight Sensors

Time-of-flight (ToF) sensors determine distances by measuring the elapsed time t between the emission of a light pulse, its reflection off of an object, and its return to the ToF sensor, as illustrated in Figure 3. Since the speed of light c is constant, the distance to an object d can be determined as $d = ct/2$.

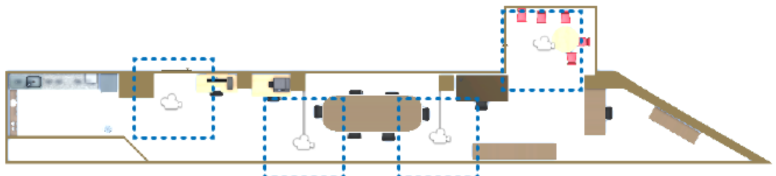
The sensor we consider in this paper is the recently announced VL53L5 time-of-flight sensor from ST Microelectronics, shown in Figure 4(a). We designed a custom

[1]`https://unity.com`
[2]`https://www.unrealengine.com`


(a) Office 1 floor plan



(b) Office 1 zone division


(c) Office 1 sensor setup

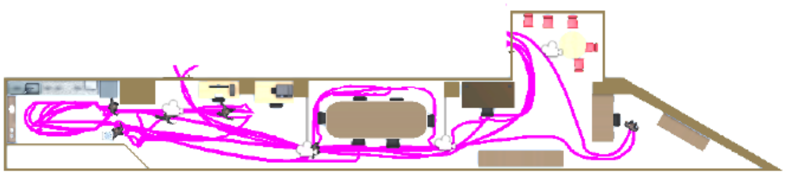
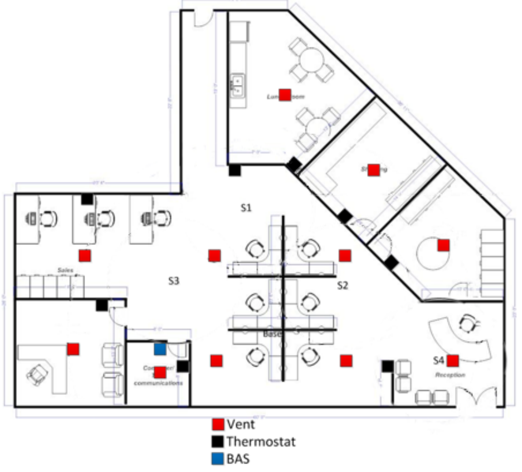
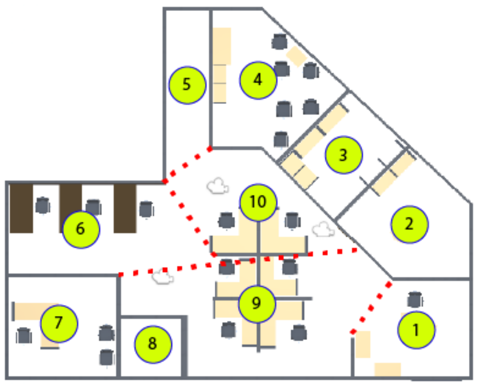

(d) Office 1 worker routes

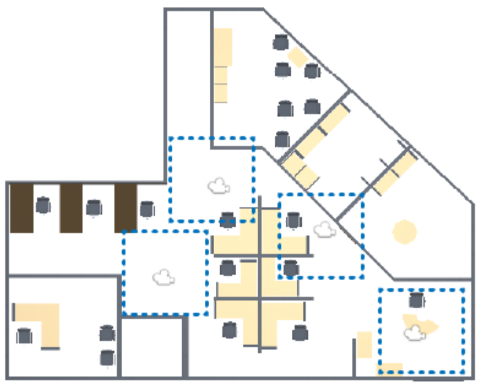
Figure 1: Simulation configuration for Office 1.



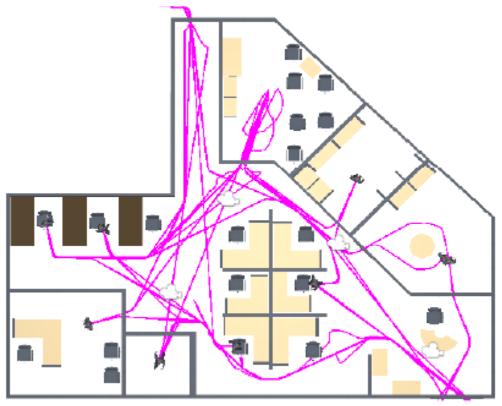
(a) Office 2 floor plan



(b) Office 2 zone division



(c) Office 2 sensor setup



(d) Office 2 worker routes

Figure 2: Simulation configuration for Office 2.

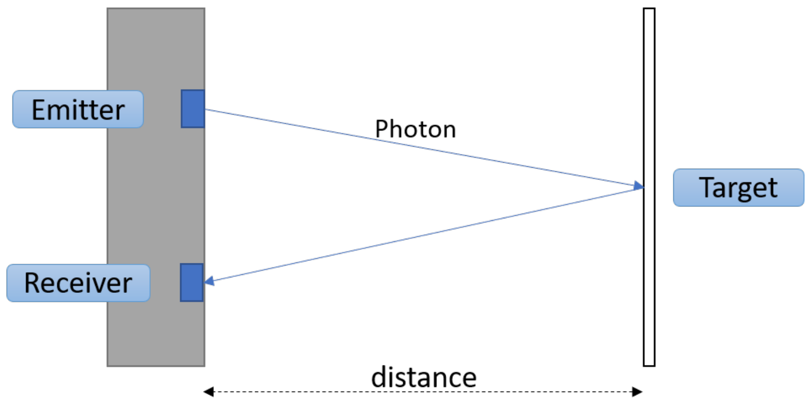


Figure 3: The time-of-flight principle. In practice the emitter and receiver are so closely spaced that the light paths can be treated as parallel.

sensor pod with a Raspberry Pi Zero microcomputer and a VL53L5 ToF sensor, shown in Figure 4(b). The Raspberry Pi Zero is used to control the VL53L5 sensor and to collect and transmit range data. Our sensor pods are AC powered, requiring a 5.1V supply with a microUSB power connector.

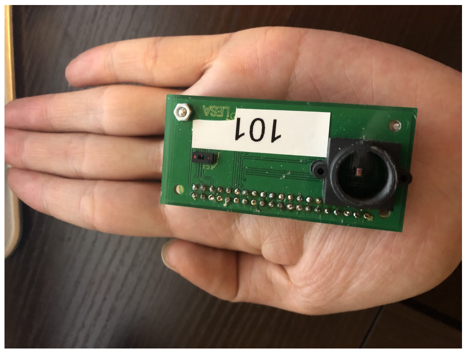
We mounted each sensor pod on the ceiling pointed directly downwards, as illustrated in Figures 4(c) and 4(d). The sensor has a detection range of approximately 5cm to 400cm and a diagonal angle of 60 degrees, as illustrated in Figure 5, where h is the ceiling height (2.7m in our environments). The sensor can return an 8×8 array of distance measurements (shown as black dashed lines in Figure 5) at 15 frames per second, or a 4×4 array of distance measurements (shown as red dashed lines in Figure 5) at 60 frames per second. Based on our tests, in an artificially lit environment (no infrared interference), the sensor measurements have zero-mean noise with standard deviation 2.5 cm.

We accurately model the sensor so that the simulated sensor characteristics are as close as possible to the real sensor, made possible by comparing the real and simulated measurements in Office 1. For example, Figure 6 illustrates the simulated sensor reading for a person under it, using the correct cone angle/resolution from the physical sensor and the real-world noise level.

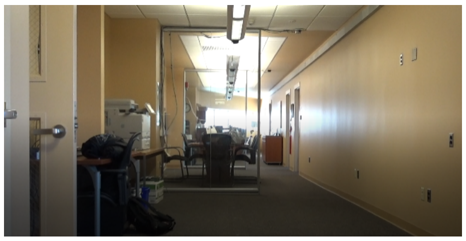
3.3. Sensor Networking

We use WiFi signals and the Message Queuing Telemetry Transport (MQTT) communication protocol to build a sensor network that collects data from the sensors, transmits data to a processor, and downloads the processed data to a computer or outputs it to a BMS controller. MQTT is a common messaging protocol for the

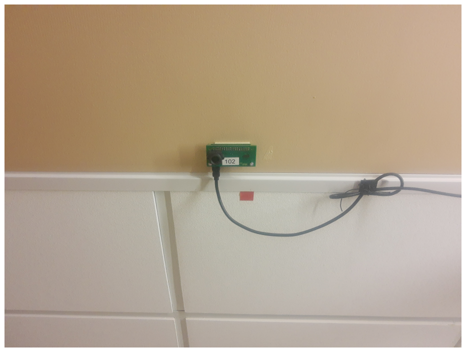
(a) STM VL53L5 sensor



(b) Custom-designed sensor pod

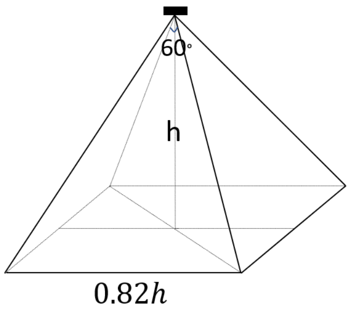


(c) Office 1 environment

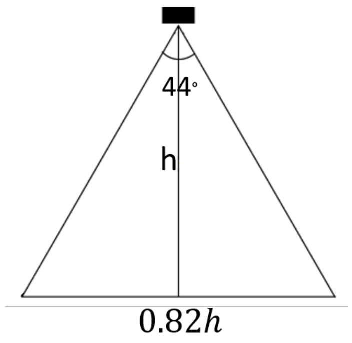


(d) Sensor pod mounted on ceiling

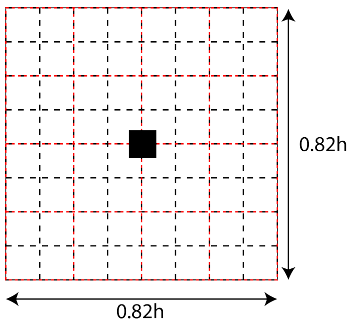
Figure 4: The ToF sensor, our custom pod, and its physical installation.



(a) 3D view

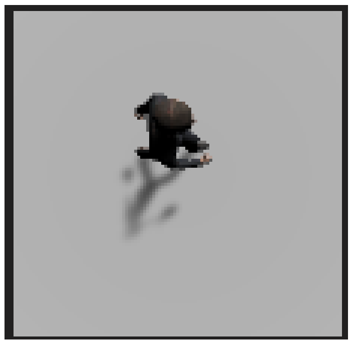


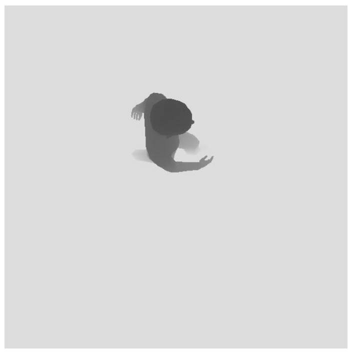
(b) Side view

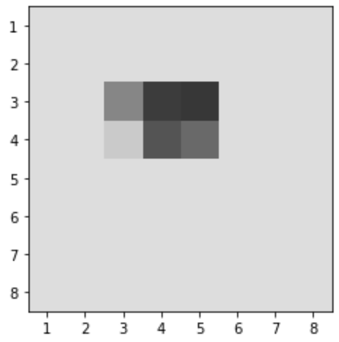


(c) Top view

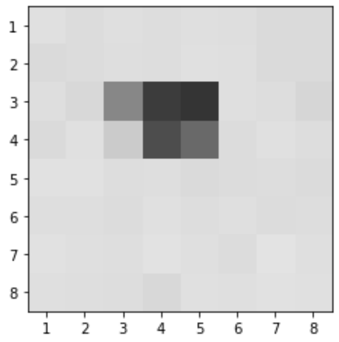
Figure 5: The VL53L5 sensors' field of view.


(a) Overhead view


(b) Depth image



(c) Noiseless sensor output



(d) Noisy sensor output

Figure 6: VL53L5 sensor simulation procedure in Unity.

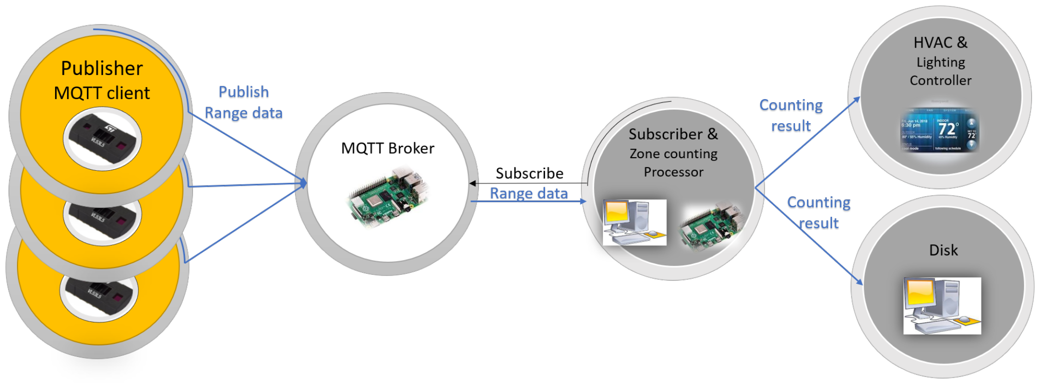


Figure 7: The MQTT publish-subscribe configuration for our sensor network.

Internet of Things, known for its light weight and efficiency. It consists of two parts: brokers and clients. Clients can publish messages to, or subscribe to messages from, brokers, but are not directly connected to each other.

In our implementation, we use a central controller (either a PC or Raspberry Pi 4) as the broker. All pods publish their distance measurements to the central controller, which subscribes to the measurements and performs the zone counting algorithm described in Section 4. The zone counting results can then be sent to an HVAC or lighting controller, or downloaded to a computer for further analysis. Figure 7 illustrates the MQTT configuration for our sensor network.

We want to use as few sensors as possible to reduce installation and maintenance costs. In the examples described in this paper, we manually determined the placement of sensors based on the desired zone divisions (shown in Figures 1(b) and 2(b)) and the workers' routes (shown in Figures 1(d) and 2(d)). We placed the sensors at the boundaries of each zone to make sure that every person that walks through a boundary will be detected by a sensor. In our experiments (both real and simulated), we required only 4 strategically placed sensors in each space. Figures 1(c) and 2(c) illustrate the fields of view of the sensors in each environment projected onto the ground plane. We can see that much of each space is not sensed directly, only the critical regions near zone boundaries.

3.4. Occupant Simulation

The final component is the realistic simulation of occupant behavior in the spaces. Animated human workers can walk, sit, and open doors in the space based on randomized paths in the context of a general 9-hour work schedule. Workers travel between their assigned desk, the communal kitchen, and the space outside the office.

The time each worker spends in each location is modeled as an independent exponential distribution, with expected value 2h, 15min, and 15min, respectively. There are 5 simulated workers in Office 1 and 8 simulated workers in Office 2. Additionally, visitors randomly arrive in each office following a Poisson distribution, head to a random location, and stay there for a random amount of time before leaving the office. Figures 1(d) and 2(d) illustrate examples of the random routes the simulated characters take in each office.

4. Algorithm Summary

In this section we discuss how we determine the occupancy counts in each zone from the raw data collected by our time-of-flight sensors. The procedure has the following steps:

1. 3D reconstruction
2. Background subtraction and filtering
3. Point clustering and person localization
4. Zone counting

We now discuss each step in detail.

4.1. 3D Reconstruction

Recovering the 3D position corresponding to the sensor reading at pixel (i, j) involves straightforward trigonometry. The pixel (i, j) defines an angled ray from the center of the sensor, and the distance $d = S(i, j)$ specifies a distance to travel along this ray, resulting in a 3D point (x, y, z) for each reading (i, j, d). The 3D point (x, y, z) is used in a clustering process in a later step.

4.2. Background Subtraction

Since the indoor environment is relatively stable, we can assume the background is fixed. We store a background image $B(i, j)$ when the room isn't occupied, and use a simple differencing method for background subtraction: pixel (i, j) is foreground at time t if

$$B(i, j) - S(i, j, t) > \tau \tag{1}$$

and background otherwise. Here we set the threshold τ proportional to the standard deviation of the measured noise of the sensor (i.e., $\tau = 2\sigma = 0.05m$). We note that this computation only makes sense in the (i, j) space since the floor-plane (x, y) position varies with the measured distance d. The z values of background points are set to 0, and the z values of foreground points are subtracted from the ceiling height to obtain a distance from the floor.

We further define a cut-off height to disregard foreground points lower than the possible height of persons/objects of interest, and also set these z values to 0. However, since the time-of-flight rays at the edges of the sensor's field of view may intersect a person low on the body, we must be careful to not set the cut-off height too low to avoid missed detections in situations where the sensors are widely spaced. We chose cut-off heights of 60cm for Office 1 (where the gaps are narrower) and 20cm for Office 2 (where the floor plan is more open).

4.3. Point clustering and person localization

At each point in time, the reading at each sensor is an array of heights (i, j, d) as illustrated in Figure 8c. The next step is to decide the number of people under the sensor. This can be more difficult than it seems since the area covered by each "pixel" is nontrivial (e.g., a 37 cm $\times$ 37 cm square on the ground for Office 1). Readings from multiple people (e.g., 2 people passing each other in a doorway) can be combined into a single pixel or merge into a connected "blob".

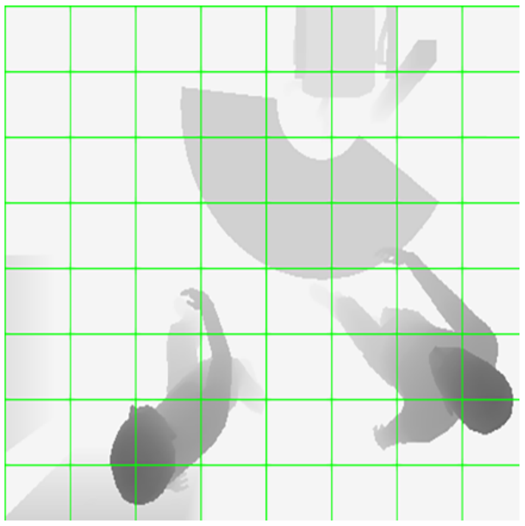
Our approach is an agglomerative clustering scheme aided by the accurate simulation discussed in Section 3. Beforehand, we generate a large database of sensor readings by positioning a synthetic human at each of 30$\times$30 positions under the sensor, at 36 possible orientations, using 10 possible human models with different heights (1.6m to 1.9m) and widths (0.5m to 1m), resulting in 324000 samples. For each sample, we save the shape (not the actual depth values) of the resulting blob, i.e., the footprint of non-zero sensor readings after background subtraction. In the 8$\times$8 sensor mode, we found there to be 1110 unique blob shapes that can be generated by a single person, each of which can fit into a 5$\times$5 square of (i, j) pixels.

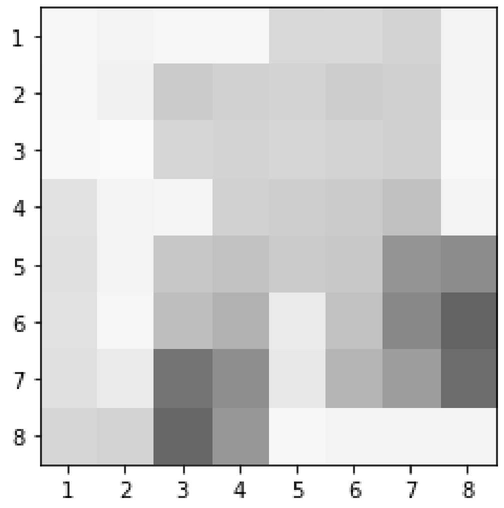
Then our clustering scheme is:

1. Start from the pixel with the highest z value under the sensor (i.e., a potential head location).
2. Add the nearest non-clustered point into the current cluster.
3. If the cluster points form one of the 1110 shapes found in the library, define this to be a person, remove them from the sensor array, and go to Step 1. Otherwise, go to Step 2.

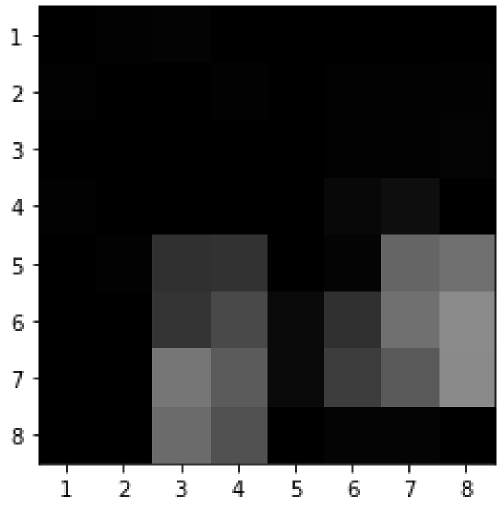
The process terminates when all points have been clustered (the final cluster may not match one of the 1110 shapes). Figure 8d illustrates an example result of the process for a sensor reading corresponding to two nearby people.

To accurately determine the location of each person under the sensor, we take the reading corresponding to the persons after clustering (which fit in a 5$\times$5 window)

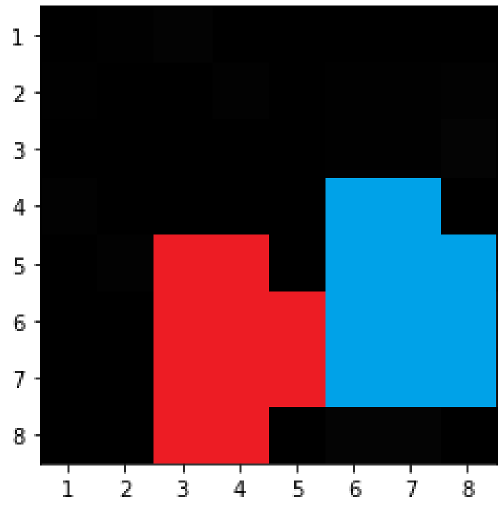

(a) Original scene.



(b) Sensor reading before background subtraction.



(c) Sensor reading after background subtraction.



(d) Clustering result

Figure 8: Point clustering process.

and feed this 5×5 array of z values into a random forest regressor trained on the simulated dataset. The output of the regressor is a continuous (i.e., non-quantized) (i, j) location of the person.

We next estimate the motion vector (equivalently, the velocity) of each detected person, based on the data from the previous 5 frames. This requires associating the N_c person blobs in the current frame with the N_p person blobs in the previous frame. There are three possibilities:

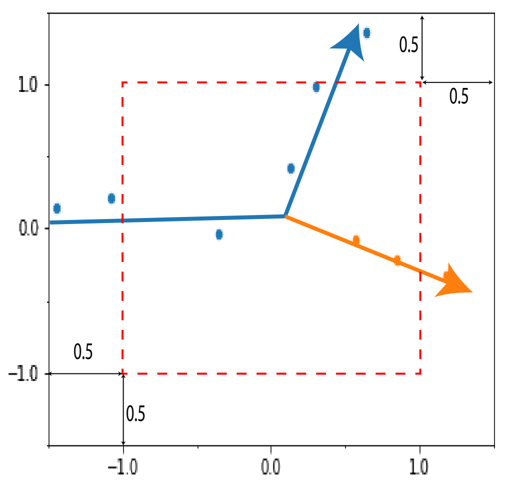
- $N_c = N_p \neq 0$. In this case, we match blobs in the current frame to the previous frame using the Hungarian algorithm [31] (i.e., minimizing the sum of distances between matched pairs).
- $N_c > N_p$. New blobs have appeared. N_p of the blobs in the current frame are associated with the blobs in the previous frame using the Hungarian algorithm, and the new $N_c - N_p$ blobs are given new labels.
- $N_p > N_c$. Old blobs have disappeared. N_c of the blobs in the current frame are associated with the blobs in the previous frame using the Hungarian algorithm, and the old $N_p - N_c$ blobs are removed from the list.

In each case, the motion vectors of all associated blobs are computed as the difference in their positions between the current and previous frames.

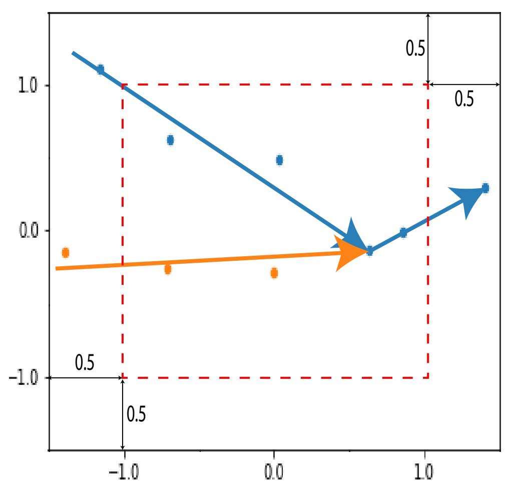
As illustrated in Figure 9, the main source of error in our algorithm is the situation where two people walking closely together enter or exit the sensor field of view and are detected as one person at some point. These split and merge events could cause over/undercounts and accumulated errors could make an unoccupied space appear to be occupied or vice versa. We use a simple heuristic that persons cannot spontaneously appear or disappear in the middle of the sensor field of view (illustrated by the central red square in Figure 9, which in practice we set to be 50cm from the edge of the sensor field of view). If a split occurs in this region, then we know the incoming single blob actually contained two people. If a merge occurs in this region, we know the outgoing blob actually contains two people. In either case, we can update the zone count accordingly as discussed in the next section.

4.4. Zone Counting

As described in Section 3, we assume that the sensors are placed along walking paths to capture all possible transitions between zones. The motion vectors of the person blobs in the previous section provide the evidence for making decisions about zone transitions. As illustrated in Figure 10(a), a zone count change from Zone A



(a) split



(b) merge

Figure 9: Split and merge cases (ground plane units are in meters).

to Zone B is triggered when a new blob appears at the A side and disappears at the B side. In general, our method handles more complex transitions as illustrated in Figure 10(b).

5. Experimental Results

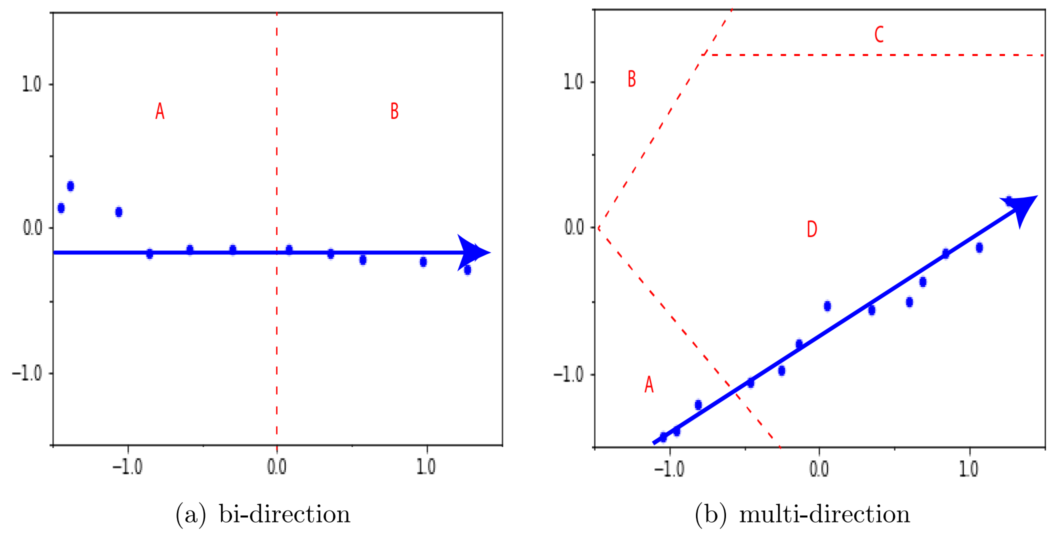
In this section we discuss the performance of each part of our algorithm and analyze the overall zone counting accuracy in the two office spaces we considered.

5.1. Localization Accuracy

Figure 11 illustrates the cumulative distribution function (CDF) of localization error. As we discussed in Section 4.3, we generate 324000 samples with known positions. We use 80% of the samples to train the localization model and use the remaining 20% of the samples to test the localization results. The indicated point shows that the probability of localization error less than 20cm is about 99%. The average localization error is 2.9cm. This error ensures that we can easily distinguish two paths that are more than 20cm apart and recognize the correct zones in complex environments.

5.2. Clustering Accuracy

By design of the clustering algorithm, the simulated training examples discussed in Section 4.3 are all correctly detected as single-person clusters. To test the cluster-



(a) bi-direction (b) multi-direction

Figure 10: Zone transitions (ground plane units are in meters). Blue dots show position estimates and blue arrows show estimated motion vectors.

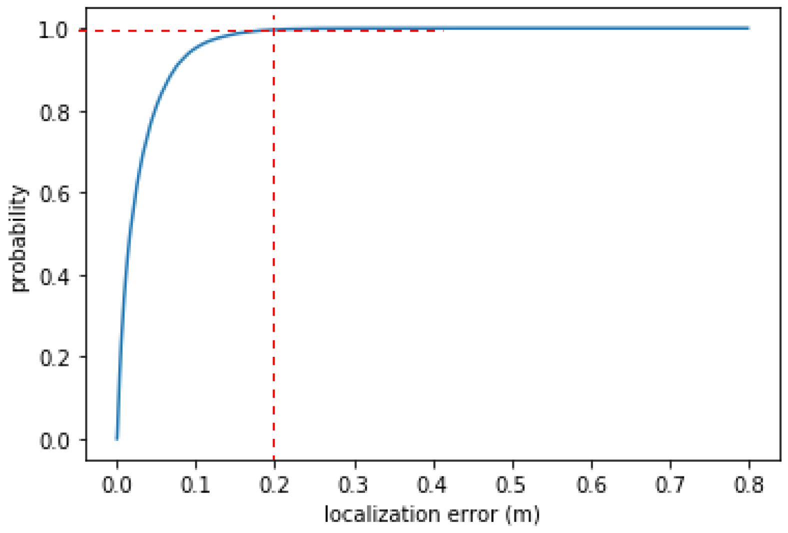


Figure 11: CDF of localization error for 8×8 sensing mode.

ing accuracy in the multiple person case, we generated 89000 samples of two office workers under the field of view of the sensor who have random heights, locations, and orientations. We then plot in Figure 12 the probability that two clusters are detected as a function of inter-person separation. From this experiment we found that two people standing more than 2m apart are correctly distinguished more than 90% of the time. Here, the distance between two people is defined as the Euclidean distance between their heads, not the outer contours of their bodies (e.g., if two people stand shoulder to shoulder, their distance is around 1m).

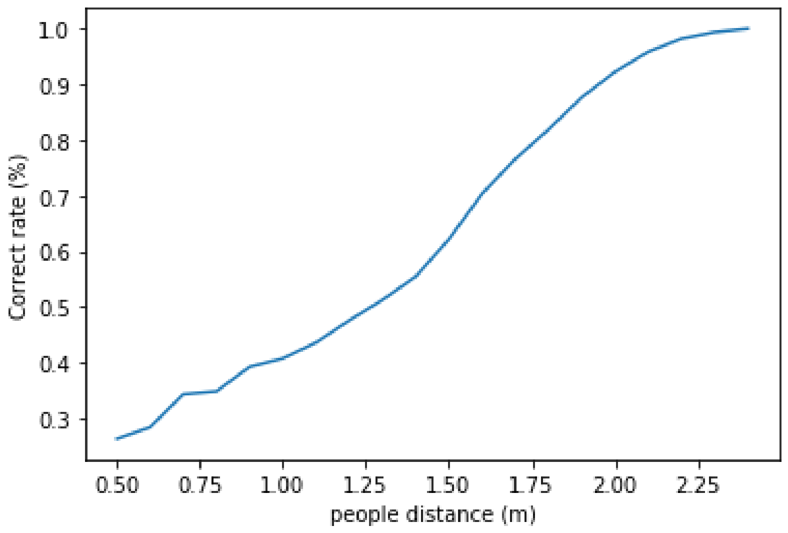


Figure 12: Correct clustering probability vs. inter-person distance (8×8 mode).

Since persons are viewed at 15 frames per second as they move under the sensor, many frames are available to make the clustering decision, mitigating the possibility of under/overcounting. For example, if two people are separated by only 1m, the probability of distinguishing them from one frame of data is 40%. However, if we observe 10 consecutive frames, the probability that at least one frame correctly distinguishes them is 99.98%, which is all we need to apply the split/merge logic discussed in Section 4.3.

5.3. Zone Counting Performance (Simulated Environment)

We now report zone counting performance results in the simulated Office 1 and Office 2 environments. Of these, Office 1 is easier in that there are only two directions of possible motion under each sensor, which are mounted over doorways and narrow gaps. Office 2 is more challenging since there are several entrance/exit zones observed by each sensor, and the possible walking paths are not bidirectional.

Tables 1 and 2 report zone counting results for Office 1 and Office 2, respectively, corresponding to 5 simulated 9-hour days in each space, as described in Section 3. In addition to total error rate, we break out the sensor activations based on the number of people under the sensor in each interaction. As expected, most of the activations (82% in Office 1 and 93% in Office 2) only involve one person, and errors in this case are very rare. The vast majority of the remaining activations involve two people under the sensor, with a handful of situations involving three people under the sensor. The results indicate that the algorithm makes very few mistakes.

Table 1: Counting performance in Office 1

Errors/total activations				
Days	***total error rate***	***1 person***	***2 persons***	***3 persons***
Day 1	1/392	0/327	1/63	0/2
Day 2	1/434	0/350	1/82	0/2
Day 3	3/410	0/317	2/89	1/4
Day 4	0/352	0/316	0/36	0/0
Day 5	3/438	0/359	3/75	0/4
Total	8/2026	0/1669	7/345	1/12
Percentage	0.40%	0%	1.73%	8.3%

Table 2: Counting performance in Office 2

Errors/total activations				
Days	***total error rate***	***1 person***	***2 persons***	***3 persons***
Day 1	1/497	0/479	1/18	0/0
Day 2	2/585	2/539	0/44	0/2
Day 3	0/426	0/409	0/17	0/2
Day 4	5/563	3/512	2/51	0/0
Day 5	4/602	1/557	3/41	0/4
Total	12/2673	6/2496	6/171	0/8
Percentage	0.45%	0.24%	3.50%	0

We identified two situations that caused errors in our experiment. The first, which we call localization error, is due to a bad location estimate resulting in an inaccurate motion vector and an incorrect zone transition. This case is illustrated in Figure 13, in which the person actually walks from Zone D to Zone A, but because of the incorrect estimate of their last location, we estimate that this person walks from Zone D to Zone B.

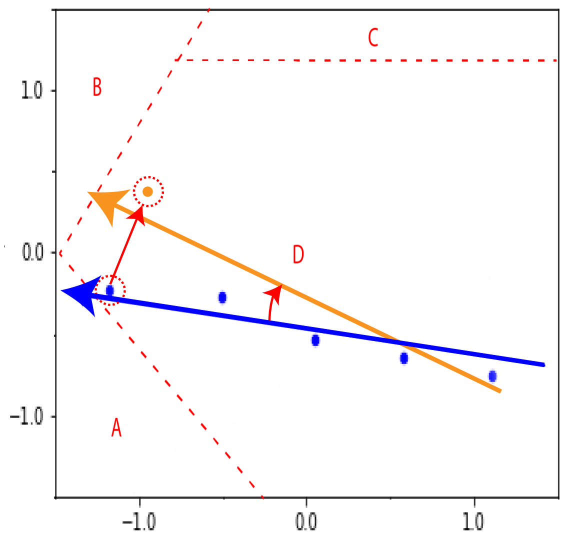


Figure 13: Counting error due to localization error. The blue line and points are the ground truth. The orange line and orange points are the estimated results. The result is that the algorithm registers a $D \to B$ transition instead of the correct $D \to A$ transition.

The other error-causing scenario is due to two or more people who walk underneath the sensor close together and are not detected by our split/merge logic, resulting in zone under- or over-counts. We call this crowding error. Table 3 shows the types of errors in the Office 1 and Office 2 experiments. We can see that in the small and simple environment, crowding accounts for most of the errors, while localization error is more likely in the large and complex environment.

5.4. Zone Counting Performance (Real Environment)

We also report results from a short experiment in the real-world version of Office 1 using the physical sensor in 4×4 mode and the same algorithms described in Section 3. Due to the pandemic, it was not possible to conduct long-term experiments with a fully staffed office of normally moving workers. In our first experiment, we collected 1.5 hours of data involving one- and two-person interactions under the sensor. During

Table 3: Breakdown of Error Types in Simulated Experiments

	localization error	**crowding error**	**total**
Office 1	1	7	8
Office 2	6	6	12
total	7	13	20

this experiment, the distance between two people was always more than $1m$. Table 4 reports the results. We observed no errors in this experiment, which shows that our algorithm works well for typical behavior.

Table 4: Typical Performance in Real-World Office 1

activity	**Errors/total activations**
1 person	0/106
2 persons same direction	0/18
2 persons crossing	0/15
Total	0/139

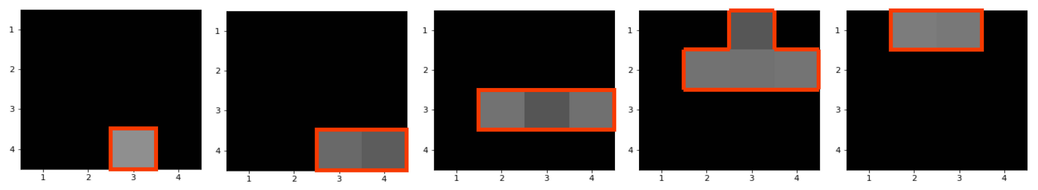
Figure 14 shows time series of real sensor readings for the cases of a single person, two persons walking in the same direction, and two persons crossing under the sensor walking in different directions, respectively.

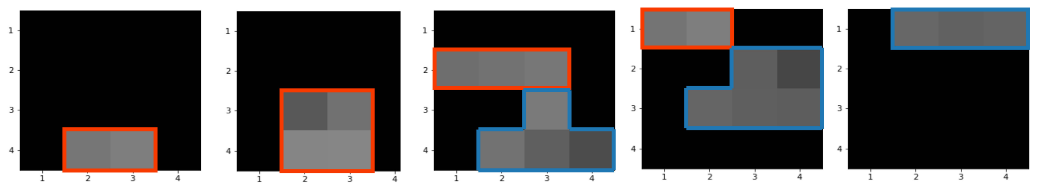
To test the counting performance in more crowded situations, we conducted a second experiment involving situations when two people walk closely (shoulder to shoulder) under the sensor, or three people are visible in the sensor FoV at the same time (with no distance requirement). In real scenarios, we do not expect more than three people to simultaneously appear under a single sensor.

Table 5: Crowded Performance in Real-World Office 1

activity	**Errors/total activations (%)**
2 persons (less than 1m)	9/44 (20.5%)
3 persons	3/39 (7.7%)

While three-person performance is fairly good, as expected, two-person perfor-


(a) 1 person walking under the sensor


(b) 2 people walking in the same direction

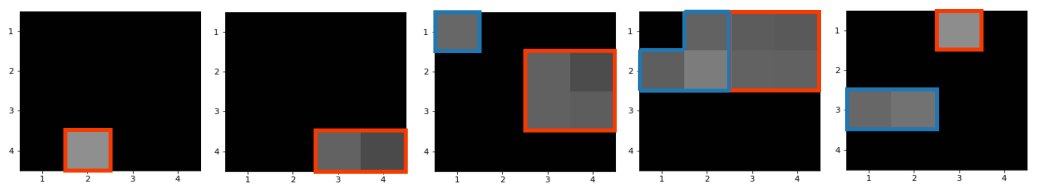

(c) 2 people crossing under the sensor

Figure 14: Data from a real sensor mounted in the real Office 1. Algorithmic clustering results are shown as red and blue outlines around raw sensor readings.

mance suffers when the pair walks very closely under the sensor. All of the errors are under/overcounts resulting from inaccurate separation of people, which is a limitation of the sensor's extremely low resolution. Hence, we judge our sensors to be suitable for commercial office spaces with low crowding levels (e.g., a typical office space vs. a subway turnstile).

5.5. HVAC energy usage simulation

We estimated the potential energy savings of occupancy-based HVAC control using the simulated Office 2 space. These savings are based on a ventilation control scheme that uses the occupancy count compared with a baseline scheme that uses constant air volume control. This analysis considers the entire space in aggregate, without considering potential minor improvements from individual zone control. The cooling case is considered, where the supply air temperature is 13° C. The control principle is that CO_2 levels must be managed by ventilating fresh air into the space, and the outside fresh air and the re-circulated zone return air that is mixed and supplied to the zones must be cooled to offset the internal heat generation. If a daily

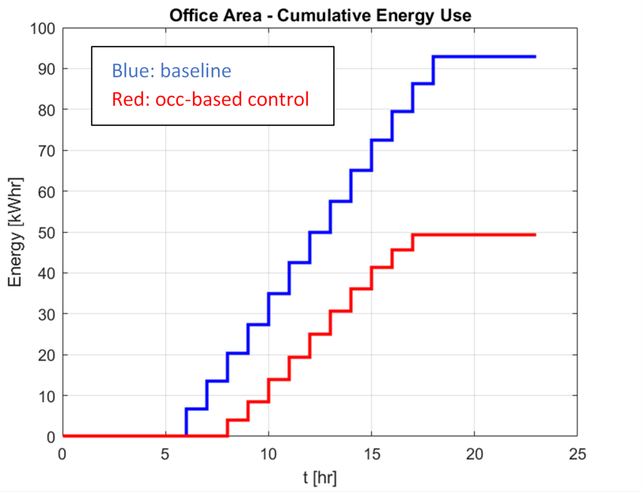


Figure 15: Cumulative thermal energy shown with/without occupant counting for the simulated Office 2.

occupancy schedule is known, then the total energy savings over a 24-hour period can be calculated. Here, in the occupancy-based control case, the instantaneous power savings as a function of occupancy in each hour is integrated over the hour to get the hourly energy savings. Figure 15 shows the result, from which we estimate an energy savings of 45% over a 24-hour simulation. This is in line with previous research reporting that energy savings of about 30% could be achieved in HVAC cooling by using an occupancy-controlled thermostat that switches the temperature set point to a higher threshold for the unoccupied zones [39], among other studies [40, 41, 42].

5.6. Longer-Term Issues

Even though our probability of making an error for a single activation is low, these errors can accrue in the longer-time-scale problem of zone counting. For example, an off-by-one miscount at the beginning of the work day could propagate for the whole day, possibly making an occupied zone appear to be unoccupied or vice versa. Table 6 reports the instantaneous probability that the zone counts are in an incorrect state (for the simulated environments discussed initially). We can see that approximately 76% of the time (Office 1) and 86% of the time (Office 2), the zone count is exactly correct, and the vast majority of the remaining time the zone count is at most off by one person.

If we wanted the occupant counts to be exactly correct more than 95% of the time, we would need the algorithm error rate to be less than 0.02% (our current error rate is 0.40–0.45% according to Tables 1 and 2). Clearly this puts very stringent

Table 6: The instantaneous probability of different miscounts in each simulated environment. The miscount is defined as the absolute difference between the true and estimated numbers of occupants.

miscount	**probability**	
	Office 1	Office 2
0	75.5%	85.5%
1	24.5%	11.8%
2	0	0
3	0	2.6%
4	0	0

requirements on any occupancy counting technology. One possibility is to "reset" the system at periodic intervals to avoid error propagation. Table 7 shows that the required error rate varies with the "reset" time interval.

Table 7: Algorithmic error rate that would be required to obtain correct counting rate more than 95%, as a function of reset interval.

reset interval (h)	**required miscounting rate**
0.5	0.43%
1	0.21%
2	0.11%
4	0.05%
9	0.02%

The latency of our system is only 0.33s (5 frames), which is quite small compared to environmental sensors like CO_2 sensors. While HVAC systems take some time to bring spaces to temperature, this low latency can enable lighting systems to deliver both energy efficiency and a better occupant experience (i.e., delivering light only where and when it is needed).

6. Conclusions

We introduced a zone-counting system to accurately monitor an occupied environment using low-cost, privacy preserving sensors, and analyzed the results of our system in both real and simulated environments. We believe that time-of-flight sensors present an attractive modality for real-world occupancy counting in commercial office spaces since they could be inexpensively and unobtrusively designed into ceiling tiles and door frames.

Compared to other sensors, the strengths of our ToF sensors include their cost, low energy consumption and data storage/processing requirements, and privacy preserving nature. On the other hand, their low resolution makes it difficult to discriminate people moving in crowds. Performance would improve if a similarly low-cost ToF sensor with about twice the resolution were available in the same form factor. Such sensors would be able to discriminate occupants' heads from their bodies while preserving their privacy.

To achieve true net energy savings using this or any occupant counting system, the costs of installing, operating, and maintaining the system must be taken into account. For example, a wireless, battery-powered sensor array is likely cheaper and easier to maintain than a sensor array that must be wired into a building management system by a trained electrician. Furthermore, the sensors themselves need not be operated at full power all the time, but should be designed to "wake up" when an occupant is in their vicinity. Finally, as we investigate the use of these sensor arrays at a larger scale (e.g., whole floors of open-plan offices), it would be useful to have an algorithm that can automatically use the floor plan of the built environment and the known lighting and HVAC zones to automatically determine the necessary number and position of occupancy sensors. We plan to adapt methods for automatic placement from the sensor networking literature, e.g., [43].

7. Acknowledgments

Thanks to Martin Krucinski and Alexander Brissette of ABB Ltd. for providing the analysis of the potential energy savings of occupancy-based HVAC control in Section 5.5. Thanks to Yunshi Wen (Rensselaer Polytechnic Institute) for helping design the characters' movement in the Unity simulation. The information, data, or work presented herein was funded in part by the Advanced Research Projects Agency-Energy (ARPA-E), U.S. Department of Energy, under Award Number DE-AR0000942. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

References

[1] D. B. Crawley, J. W. Hand, M. Kummert, B. T. Griffith, Contrasting the capabilities of building energy performance simulation programs, Building and

environment 43 (2008) 661–673.

[2] K. Sun, Q. Zhao, J. Zou, A review of building occupancy measurement systems, Energy and Buildings 216 (2020) 109965.

[3] E. O'Dwyer, I. Pan, S. Acha, N. Shah, Smart energy systems for sustainable smart cities: Current developments, trends and future directions, Applied Energy 237 (2019) 581–597.

[4] Z. Yang, B. Becerik-Gerber, The coupled effects of personalized occupancy profile based HVAC schedules and room reassignment on building energy use, Energy and Buildings 78 (2014) 113–122.

[5] K. C. Jagadeesh Simma, A. Mammoli, S. M. Bogus, Real-Time Occupancy Estimation Using WiFi Network to Optimize HVAC Operation, Procedia Computer Science 155 (2019) 495–502.

[6] J. Shi, N. Yu, W. Yao, Energy Efficient Building HVAC Control Algorithm with Real-time Occupancy Prediction, Energy Procedia 111 (2017) 267–276.

[7] D. Liu, X. Guan, Y. Du, Q. Zhao, Measuring indoor occupancy in intelligent buildings using the fusion of vision sensors, Measurement Science and Technology 24 (2013) 074023.

[8] Y. P. Raykov, E. Ozer, G. Dasika, A. Boukouvalas, M. A. Little, Predicting room occupancy with a single passive infrared (pir) sensor through behavior extraction, in: Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, 2016, pp. 1016–1027.

[9] F. Wahl, M. Milenkovic, O. Amft, A distributed pir-based approach for estimating people count in office environments, in: 2012 IEEE 15th International Conference on Computational Science and Engineering, IEEE, 2012, pp. 640–647.

[10] J. Andrews, M. Kowsika, A. Vakil, J. Li, A motion induced passive infrared (PIR) sensor for stationary human occupancy detection, in: 2020 IEEE/ION Position, Location and Navigation Symposium (PLANS), 2020-04, pp. 1295–1304. ISSN: 2153-3598.

[11] S. Wolf, D. Calı, J. Krogstie, H. Madsen, Carbon dioxide-based occupancy estimation using stochastic differential equations, Applied Energy 236 (2019) 32–41.

[12] A. Szczurek, M. Maciejewska, T. Pietrucha, Occupancy determination based on time series of co2 concentration, temperature and relative humidity, Energy and Buildings 147 (2017) 142–154.

[13] H. Li, T. Hong, M. Sofos, An inverse approach to solving zone air infiltration rate and people count using indoor environmental sensor data, Energy and Buildings 198 (2019) 228–242.

[14] R. Valle, ABROA: Audio-based room-occupancy analysis using Gaussian mixtures and Hidden Markov models, in: 2016 Future Technologies Conference (FTC), 2016, pp. 1270–1273.

[15] S. Depatla, A. Muralidharan, Y. Mostofi, Occupancy estimation using only wifi power measurements, IEEE Journal on Selected Areas in Communications 33 (2015) 1381–1393.

[16] H. Zou, Y. Zhou, J. Yang, C. J. Spanos, Device-free occupancy detection and crowd counting in smart buildings with wifi-enabled iot, Energy and Buildings 174 (2018) 309–322.

[17] Y. Yang, J. Cao, X. Liu, X. Liu, Door-monitor: Counting in-and-out visitors with cots wifi devices, IEEE Internet of Things Journal 7 (2019) 1704–1717.

[18] T. Labeodan, K. Aduda, W. Zeiler, F. Hoving, Experimental evaluation of the performance of chair sensors in an office space for occupancy detection and occupancy-driven control, Energy and Buildings 111 (2016) 195–206.

[19] Y. Kaddoura, J. King, A. Helal, Cost-precision tradeoffs in unencumbered floor-based indoor location tracking, in: Proceedings of the third International Conference On Smart homes and health Telematic (ICOST), Citeseer, 2005.

[20] R. Razavi, A. Gharipour, M. Fleury, I. J. Akpan, Occupancy detection of residential buildings using smart meter data: A large-scale study, Energy and Buildings 183 (2019) 195–208.

[21] J. Zou, Q. Zhao, W. Yang, F. Wang, Occupancy detection in the office by analyzing surveillance videos and its application to building energy conservation, Energy and Buildings 152 (2017) 385–398.

[22] A. K. Chandran, W. Wong, Pedestrian crowd level estimation by head detection using bio-inspired retina model, in: 2016 IEEE Region 10 Conference (TENCON), 2016, pp. 3153–3156. ISSN: 2159-3450.

[23] A. K. Chandran, L. A. Poh, P. Vadakkepat, Real-time identification of pedestrian meeting and split events from surveillance videos using motion similarity and its applications, J Real-Time Image Proc 16 (2019) 971–987.

[24] Y. Benezeth, H. Laurent, B. Emile, C. Rosenberger, Towards a sensor for detecting human presence and characterizing activity, Energy and Buildings 43 (2011) 305–314.

[25] P. W. Tien, S. Wei, J. K. Calautit, J. Darkwa, C. Wood, A vision-based deep learning approach for the detection and prediction of occupancy heat emissions for demand-driven control solutions, Energy and Buildings 226 (2020) 110386.

[26] I. Ahmed, A. Ahmad, F. Piccialli, A. K. Sangaiah, G. Jeon, Jeon, A robust features-based person tracker for overhead views in industrial environment, IEEE Internet of Things Journal 5 (2018-06) 1598–1605. Conference Name: IEEE Internet of Things Journal.

[27] J. Perng, T. Wang, Y. Hsu, B. Wu, The design and implementation of a vision-based people counting system in buses, in: 2016 International Conference on System Science and Engineering (ICSSE), 2016-07, pp. 1–3. ISSN: 2325-0925.

[28] S. Petersen, T. H. Pedersen, K. U. Nielsen, M. D. Knudsen, Establishing an image-based ground truth for validation of sensor data-based room occupancy detection, Energy and Buildings 130 (2016-10-15) 787–793.

[29] B. S. Totada, S. D. Cabrera, Detection of people from time-of-flight depth images using a cell-tracking methodology., in: 2018 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), 2018-12, pp. 084–089. ISSN: 2162-7843.

[30] G. Diraco, A. Leone, P. Siciliano, People occupancy detection and profiling with 3d depth sensors for building energy management, Energy and Buildings 92 (2015-04) 246–266.

[31] I. Bhattacharya, R. J. Radke, Arrays of single pixel time-of-flight sensors for privacy preserving tracking and coarse pose estimation, in: 2016 IEEE Winter Conference on Applications of Computer Vision (WACV), IEEE, 2016, pp. 1–9.

[32] A. K. Mikkilineni, J. Dong, T. Kuruganti, D. Fugate, A novel occupancy detection solution using low-power IR-FPA based wireless occupancy sensor, Energy and Buildings 192 (2019-06-01) 63–74.

[33] M. Cokbas, P. Ishwar, J. Konrad, Low-resolution overhead thermal tripwire for occupancy estimation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2020, pp. 88–89.

[34] J. Dai, J. Wu, B. Saghafi, J. Konrad, P. Ishwar, Towards privacy-preserving activity recognition using extremely low temporal and spatial resolution cameras, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2015, pp. 68–76.

[35] L. Jia, R. J. Radke, Using time-of-flight measurements for privacy-preserving tracking in a smart room, IEEE Transactions on Industrial Informatics 10 (2013) 689–696.

[36] D. Ioannidis, P. Tropios, S. Krinidis, G. Stavropoulos, D. Tzovaras, S. Likothanasis, Occupancy driven building performance assessment, Journal of Innovation in Digital Ecosystems 3 (2016) 57–69.

[37] Unity, Unity game engine., `https://unity.com/`, [Online; accessed 28-June-2021].

[38] Epic Games, Unreal engine., `https://www.unrealengine.com/en-US/`, [Online; accessed 28-June-2021].

[39] J. Yang, M. Santamouris, S. E. Lee, Review of occupancy sensing systems and occupancy modeling methodologies for the application in institutional buildings, Energy and Buildings 121 (2016) 344–349.

[40] J. Li, K. Panchabikesan, Z. Yu, F. Haghighat, M. El Mankibi, D. Corgier, Systematic data mining-based framework to discover potential energy waste patterns in residential buildings, Energy and Buildings 199 (2019) 562–578.

[41] D. F. M. Cabrera, H. Zareipour, Data association mining for identifying lighting energy waste patterns in educational institutes, Energy and Buildings 62 (2013) 210–216.

[42] C. Turley, M. Jacoby, G. Pavlak, G. Henze, Development and evaluation of occupancy-aware hvac control for residential building energy efficiency and occupant comfort, Energies 13 (2020) 5396.

[43] I. Vlasenko, I. Nikolaidis, E. Stroulia, The smart-condo: Optimizing sensor placement for indoor localization, IEEE Transactions on Systems, Man, and Cybernetics: Systems 45 (2014) 436–453.