Pervasive and Mobile Computing 45 (2018) 35–54

Contents lists available at ScienceDirect

Pervasive and Mobile Computing

journal homepage: www.elsevier.com/locate/pmc

Counting via LED sensing: Inferring occupancy using lighting infrastructure

Yanbing Yang [a,b,*], Jun Luo [a], Jie Hao [c], Sinno Jialin Pan [a]

[a] School of Computer Science and Engineering, Nanyang Technological University, Singapore
[b] College of Computer Science, Sichuan University, China
[c] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China

ARTICLE INFO

Article history:
Available online 8 March 2018

Keywords:
Occupancy inference
Visible Light Sensing
Statistical learning

ABSTRACT

As a key component of building management and security, occupancy inference through smart sensing has attracted a lot of research attention for nearly two decades. Nevertheless, existing solutions mostly rely on either pre-deployed infrastructures or user device participation, thus hampering their wide adoption. This paper presents CeilingSee, a dedicated occupancy inference system free of heavy infrastructure deployments and user involvements. Building upon existing LED lighting systems, CeilingSee converts part of the ceiling-mounted LED luminaires to act as sensors, sensing the variances in diffuse reflection caused by occupants. In realizing CeilingSee, we first re-design the LED driver to leverage LED's photoelectric effect so as to transform a light emitter to a light sensor. In order to produce accurate occupancy inference, we then engineer efficient learning algorithms to fuse sensing information gathered by multiple LED luminaires. We build a testbed covering a 30 m^2 office area; extensive experiments show that CeilingSee is able to achieve very high accuracy in occupancy inference.

1. Introduction

The awareness of (indoor) occupancy is crucial to many aspects of smart building management; these include, among others, controlling the HVAC and lighting systems for the sake of energy conservation, choosing the right information service based on congestion level, as well as safely evacuating people under life-threatening circumstances. In the past two decades, various smart sensing technologies have been dedicated[1] to infer occupancy for indoor facilities, and most of them require deploying certain sensing infrastructure with mainly three typical sensors: passive infra-red (PIR) [4,5], acoustic/ultrasonic [6–8], and camera [9,10]. Other solutions attempt to infer occupancy indirectly by monitoring the usage of existing services (e.g., Wi-Fi [11] and power grid [12]). Whereas the former method requires installations of extra infrastructure and hence incurs both high cost for building management and potential infringement of user privacy, the latter approach can hardly be accurate: what if some occupants simply do not use any services?

Our key observation is that, in any human occupied indoor spaces, lighting is a necessity while the resulting diffuse reflection can be "disturbed" by the presence of occupants. In the meantime, *Visible Light Sensing* (VLS), as a variance of heavily studied *Visible Light Communication* (VLC) [13–17], has started to show its potential in many sensing-intensive applications [18]. Therefore, a natural question is: *can we apply VLS to build an occupancy inference system that is free of*

* Corresponding author at: School of Computer Science and Engineering, Nanyang Technological University, Singapore.
E-mail address: yyang017@ntu.edu.sg (Y. Yang).

[1] Though occupancy information can be derived from an indoor localization system (e.g., [1–3]), no practical indoor localization system has been widely adopted so far. Moreover, relying on user location tracking to "count" occupancy is highly inefficient and may infringe privacy.

https://doi.org/10.1016/j.pmcj.2018.01.003
1574-1192/

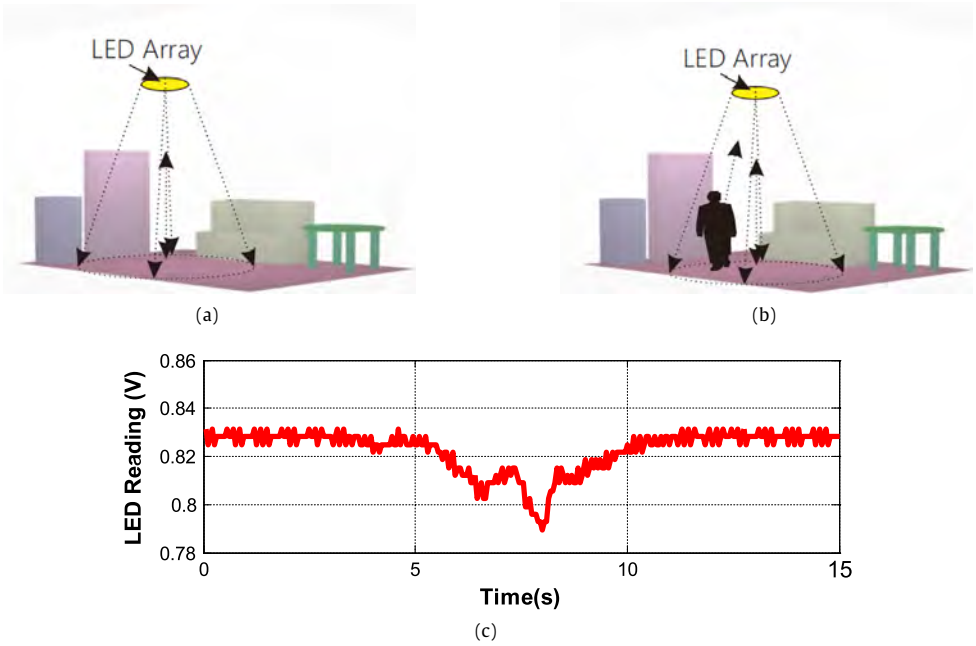


Fig. 1. Inferring occupancy by LED sensing. (a) The sensing coverage of an LED array. (b) An occupant enters the covered area. (c) The variance in LED readings due to the absence/presence of the occupant.

reliance on both heavy infrastructures and user involvements? In this paper, through a detailed exposition of our CeilingSee system, we intend to provide readers with a positive answer.

The first idea of CeilingSee is very intuitive: from the ceiling point of view, the diffuse reflection (consisting of main reflections from the floor and variously fixed furniture on the floor) is bounded to be affected by the presence of occupants. Therefore, sensing such perturbations could allow us to infer occupancy. While simply installing an array of light sensors on the ceiling could be a solution, it would introduce yet another infrastructure. Fortunately, the increasing popularity of LED lighting systems and the readily verifiable photoelectric effect of LED [19,14] have motivated our another novel idea: re-designing the driver of a *Commercial Off-The-Shelf* (COTS) LED could enable it to serve as both a light emitter and a light sensor. Consequently, CeilingSee simply leverages the existing LED lighting systems and borrows a fraction of LED chips to sense the variance in diffuse reflection. We illustrate these ideas in Fig. 1; it clearly demonstrates the potential and effectiveness of VLS-based occupancy inference.

The seemingly straightforward ideas of CeilingSee impose on us two major challenges.

Firstly, although conventional LED-to-LED communication [14] has already employed an LED as receiver (a special form of sensor), sensing the variance in diffuse reflection is much more challenging due to the very low SNR, hence it necessitates using the collective sensing ability of multiple LED chips. Existing LED receiver directly connects an LED chip to the I/O port of an MCU, thus relying on the controllable nature of the I/O port to toggle the states of the LED between *forward biased* (emitting or sending) and *reverse biased* (sensing or receiving). Unfortunately, this would not work when multiple LED chips are used together, as the voltage/current would exceed what an I/O can take (normally no more than 3.3V/20 mA) and one cannot afford to directly attach an MCU to each LED chip. As a result, we design a novel circuit for accommodating the collective photoelectric effects of an LED array.

Secondly, as the sensing coverage[2] of a single LED array (consisting of multiple "sensorized" LED chips) is limited, we have to use multiple arrays to cover a large indoor area, which happens to be in line with the lighting requirement. Moreover, CeilingSee needs to account for multiple occupants dispersed on the area, especially those not strictly under an LED array. Therefore, it is necessary that efficient inference algorithms are in place to utilize the collective sensing outcomes of all LED arrays. CeilingSee responds to this challenge by engineering a machine learning algorithm that maps the multi-dimensional sensing data to the demanded occupancy count.

To validate our design of CeilingSee, we build a testbed consisting of multiple LED arrays in order to cover a 30 m^2 office area. We implement the hardware part for controlling the LED arrays, as well as the software part for sensing data processing and hence occupancy inference. Our main contributions are as follows:

[2] It is also termed *Field of View* (FoV) in sensing nomenclature, so we use coverage and FoV interchangeably hereafter.

Table 1
Comparing CeilingSee with existing solutions for occupancy inference.

Sensor Type	Infrastructure reuse	Privacy concern	Accuracy	Cost	Limitation
Ultrasonic [8]	No	No	$\geq$ 90%	Medium	Need prior maximum and distribution of occupants
Thermal Sensor & PIR [5]	No	No	88.5%	Medium	Extra sensors
Camera [9]	Possible	Yes	80%	High	Heavy computation, re-calibration based on dimming
Camera & PIR [20]	Partial	Yes	94%	High	Heavy computation, re-calibration based on dimming
Photosensors [21]	No	No	N/A	Low	Extra sensor, re-calibration based on dimming
LED (CeilingSee)	Yes	No	$\geq$ 90%	Low	Re-calibration based on dimming

- We propose the novel idea of applying only ceiling-mounted LED lighting system for inferring occupancy, and we build CeilingSee to showcase the efficiency and effectiveness of such a lightweight occupancy inference approach.
- We re-design the driver of a COTS LED array so that CeilingSee can freely toggle the LED array between light emitting and light sensing modes and a group of LED arrays can be collectively used for sensing the variance in (indoor) diffuse reflection.
- We engineer data processing and machine learning algorithms to deduce the occupancy within the FoV of a single LED array, as well as to infer full area occupancy by fusing the multi-dimensional sensing outcomes from multiple LED arrays.
- We conducted extensive field experiments in the past six months to validate the effectiveness of CeilingSee, and the results strongly demonstrate its high accuracy in occupancy inference.

We are not expecting CeilingSee to fully replace the existing occupancy inference solutions. Instead, we deem the technical implication of CeilingSee as twofold: (i) it is an avatar of a VLS-based idea that minimizes the resources required for inferring occupancy and may thus inspire other applications of VLS, and (ii) it serves as a complement to other solutions for improving the efficiency and scalability of occupancy inference systems. We compare CeilingSee with typical existing solutions in Table 1. In the following, we first introduce the principle of LED sensing, along with the design and experience with a single sensing unit of CeilingSee in Section 2. Then we present our learning-based occupant inference algorithms in Section 3. We report the performance evaluation of CeilingSee in Section 4, and we discuss the limitations and potentials of CeilingSee in Section 5. A brief literature survey is provided in Section 6, before finally concluding the paper in Section 7.

2. Sensing reflection by LED

Lighting systems are pervasively deployed and used for indoor spaces due to the inadequate natural lighting from windows, especially when the space has limited access to day light. Such lighting systems are normally ceiling-mounted and hence cause diffuse reflections from the floor (including various furniture on it), which is biased towards the ceiling due to the blending with minor specular reflection. When (human) occupants move into this "reflection field", they can cause perturbations readily sensible by certain ceiling-mounted light sensors. To avoid introducing an extra sensing infrastructure, our idea is to re-use part of the existing LED lighting system to serve the sensing function.

It is well known that LED has photoelectric effect, i.e., light shining upon an LED can cause it to emit electrons, which is a reverse effect of LED's default functionality [22]. This effect has motivated a few proposals to use an LED not only as a sender but also as a receiver in VLC [19,14]. However, converting an LED receiver to an LED sensor is almost impossible as the light signal sent by an LED has a much higher *Signal-to-Noise-Ratio* (SNR) than the variance in reflection. Moreover, as we only "borrow" part of the LED lighting system for the sensing purpose, we want to toggle the LEDs between light emitting and light sensing modes when either is in need. Therefore, we present the details of CeilingSee's hardware implementation in this section, aiming to address the aforementioned issues.

2.1. From LED receiving to LED sensing

We briefly describe the conventional design of an LED receiver, and then we explain why and how the design of LED sensing for CeilingSee should be different.

2.1.1. Bidirectional setting of LED receiver

In a conventional design for light receiver in the LED–LED communication, a bidirectional interface to an LED is created by connecting the LED directly between the two I/O pins of a micro-controller (MCU) [19], as shown in Fig. 2. Fig. 2(a) shows that the LED emits light when its anode and cathode are connected to VCC and GND, respectively, via a simple I/O configuration. Reverting the I/O configuration sets the LED in reverse bias mode as in Fig. 2(b); it charges the inner stray capacitance of the LED and prepares the LED for light sensing. Fig. 2(c) further illustrates the actual measurement phase: MCU reads the voltage changes on LED's cathode and times how long it takes for the photocurrent to discharge the capacitance to the I/O pin's digital input threshold. Obviously, the discharging time is inversely proportional to the amount of incident light. The simplicity of this LED receiving circuit stems from the matching voltages between an MCU and an LED. Apparently, it does not work if one wishes to drive more LEDs with one MCU.

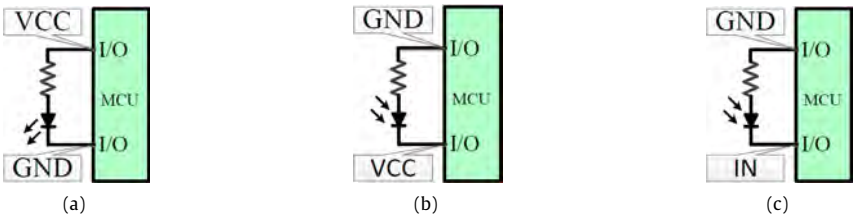


Fig. 2. Conventional bidirectional interface between LED and MCU. (a) Normal I/O configuration for light emitting. (b) Reverse bias for light sensing. (c) I/O as input for reading sensed signal.

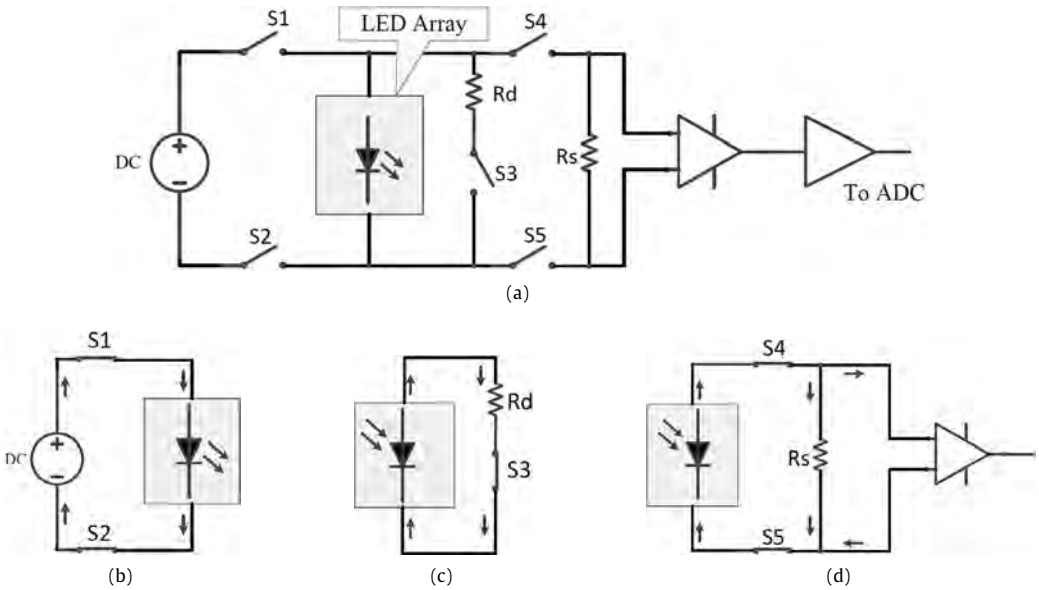


Fig. 3. Architecture of the LED array driver. (a) Circuit schematic of the driver. (b) The equivalent circuit during light emitting. (c) The equivalent circuit for discharging. (d) The equivalent circuit during light sensing.

2.1.2. Collective sensing with multiple LEDs

Compared with the LED–LED communication, the signal for our reflection sensing scenario is much weaker.[3] As a result, if we used the same circuit as described in Section 2.1.1, we would either fail to sense the signal due to the too weak photocurrent or experience a huge sensing delay thanks to the long discharging time. Whereas using multiple LEDs to collectively sense the weak signal appears to be a straightforward solution, the much higher voltage level caused by aggregating multiple LEDs renders the existing I/O-based circuit design invalid. Also, reverse biasing an array of LED is almost impossible for normal MCU due to the limit of its absolute maximum voltage ratings.[4] Therefore, our intention is to use an LED array as one light sensing unit without the need for reverse biasing it. According to the LED's equivalent circuit model [19], an LED can be deemed as a current source with a shunt capacitor, while the current source is driven by incident light to generate tiny photocurrent. Whereas the current produced by one LED is too weak to be measurable (even through an amplifier), the aggregated current of the LED array (with a sufficient amount of LEDs) would suffice for the sensing purpose.

Based on the aforementioned ideas, we re-design the driver of COTS LED luminaires to control the LED state toggling, so that CeilingSee can duty-cycle part of the luminaire between light emitting (for normal lighting) and light sensing (for occupancy inference). Fig. 3 shows the architecture of this driver circuit and also its functionalities under different modes. The five switches in Fig. 3(a) are the key components for the driver. The LED array emits light (lighting phase) if both S1 and S2 are ON and other switches are OFF, as shown by Fig. 3(b). Subsequently, Fig. 3(c) shows a short discharging window

[3] The signal could also be weaker in LED–LED communication if the transmission distance goes beyond the centimeter testing scenarios in [19,14]. As a practical sensing system, CeilingSee has to work under a "transmission distance" (that from floor to ceiling) of several meters.

[4] Though this could be made possible by using special high-power components, e.g. high reverse voltage MOSFET, the high cost would compromise our purpose of building a lightweight occupancy inference system.

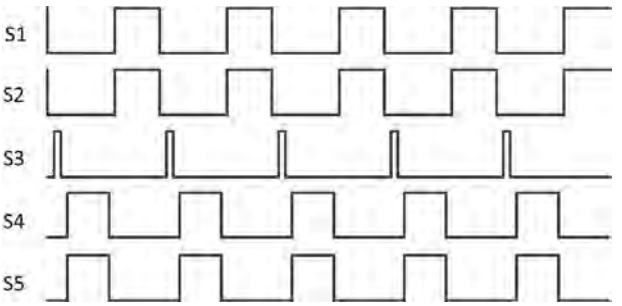


Fig. 4. Duty-cycled control signal sequence for toggling between lighting and sensing.

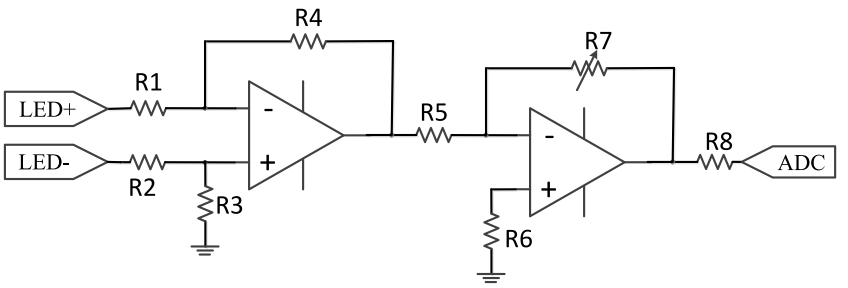


Fig. 5. Amplifier circuits. The voltage difference between the cathode and anode of the LED array is firstly extracted by a differential amplifier. Then the differential signal is further amplified before sampled by ADC.

that allows residual charges on the array to be cleared for preparing sensing (discharging phase), by putting S3 ON while others OFF. Finally, switching S4 and S5 ON and others OFF enables the array to act as a light sensor (sensing phase): the resistor Rs ($>10M\Omega$) converts weak photocurrents to voltage signals that drive amplifiers to produce sensing outcome. We also show the duty-cycled control sequence of different switches in Fig. 4: to prevent short circuit, a dead-time should be inserted between two consecutive phases. We will discuss more detailed implementations in Section 2.1.3.

2.1.3. Design details of the LED driver

We elaborate on two key components of the driver here, namely discharging circuit and amplifier circuit.

Discharging Circuit. During the lighting phase, the inner capacitance of the LED array can be partially charged and the charging level cannot be controlled; this could interfere the sensing phase as the photocurrent produced by incident light is tiny. To this end, the discharging circuit is particularly designed for preparing the sensing phase without the need for reverse biasing the LEDs. It consists of a switch and a bleeder resistor shown in Fig. 3(a). The discharging phase between the light and sensing phases can be very short (normally 100 μs); it quickly clears the residual charges so as to get a clean start for the sensing phase. The switching from the lighting phase to discharging phase has to be strictly controlled by MCU with a dead-time in-between (another 100 μs), otherwise the driver can be damaged due to short circuit.

Amplifier Circuit. During the sensing phase, the photocurrent excited by the incident light is converted to weak voltage signal by Rs, which needs to be amplified before being sampled by the ADC of MCU. Fig. 5 shows the detailed schematic of the amplifier circuit; it consists of a differential amplifier with fixed-gain and an inverting amplifier with adjustable gain. The differential amplifier is used to amplify the voltage (difference) between the cathode and anode of the LED array; its output can be approximated as:

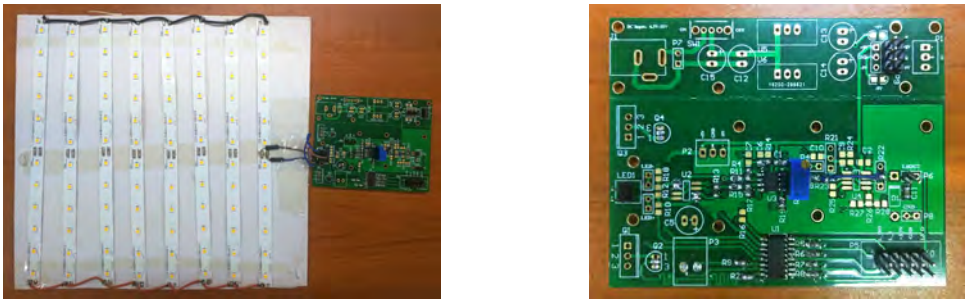
$$V_{\text{out1}} = -(V_{\text{LED+}} - V_{\text{LED-}}) \times \frac{\text{R4}}{\text{R1}}, \tag{1}$$

where $V_{\text{LED+}}$ and $V_{\text{LED-}}$ are the voltages of the LED array's cathode and anode, respectively, and V_{out1} is the output of the differential amplifier.

The adjustable gain amplifier circuit includes a high gain inverting amplifier and an adjustable resistor R7. As a result, we have an adjustable output V_{ADC} expressed as (2), which is in turn related to the light excited voltage by (3).

$$V_{\text{ADC}} = -V_{\text{out1}} \times \frac{\text{R7}}{\text{R5}}, \tag{2}$$

$$V_{\text{ADC}} = (V_{\text{LED+}} - V_{\text{LED-}}) \times \frac{\text{R4}}{\text{R1}} \times \frac{\text{R7}}{\text{R5}}. \tag{3}$$



(a) Sensing/lighting unit. (b) Re-designed driver.

Fig. 6. A sensing/lighting unit of CeilingSee, consisting of a LED array and a re-designed driver.

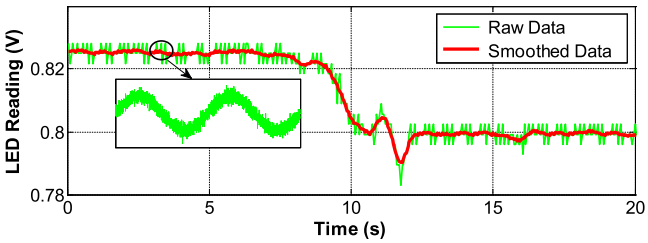


Fig. 7. Raw and smoothed ADC readings (V_{ADC}) that indicate reflection variance due to occupancy.

2.2. Experiencing single sensing unit

Fig. 6 shows one *sensing unit* of CeilingSee; it consists of a 8×12 LED chip array [23] and a PCB carrying our re-designed driver. We separate the chips into two groups and duty-cycle them in a complementary manner to perform both lighting and sensing simultaneously while avoiding flickering. Using this hardware platform, we first test the capability of single-unit occupancy inference, aiming to study the performance of such a unit under various parameter settings and situations. More hardware implementation details on the unit will be given in Section 4.1.

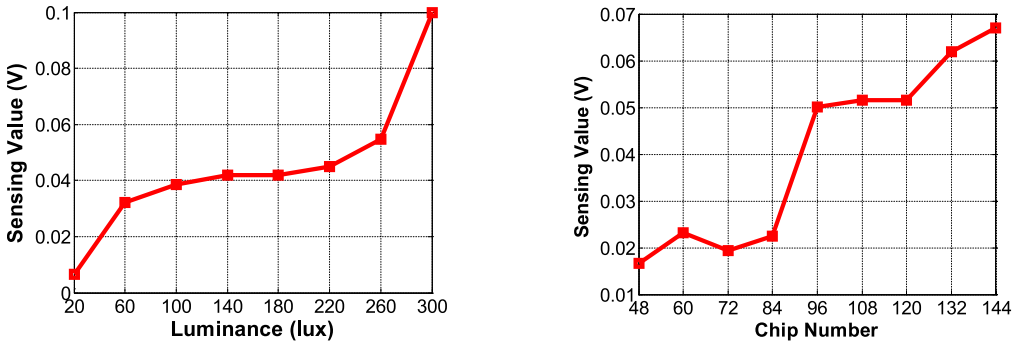
2.2.1. Raw reading and signal smoothing

We first demonstrate the effectiveness of using LED sensing to indicate occupancy: we record V_{ADC} when one occupant walks into the sensing coverage and stay there, and the V_{ADC} in Fig. 7 clearly shows the variance in reflection incurred by occupant. The rather unstable raw readings are mainly interfered by two sources: a minor random noise and a 50 Hz component.[5] Both interferences are very easy to be removed by applying a moving average with a window size of 100 ms. Therefore, we report only the smoothed readings hereafter. Also, we take the absolute difference between a V_{ADC} reading and the nominal reading (obtained in absence of any occupants) as the actual *sensing value*.

2.2.2. Impacts of illuminance and dimension

As a sensing unit is sensing the reflection, the outcome ought to be affected by the ambient illuminance. Also, according to Section 2.1.2, the sensitivity of the unit depends on the number of LED chips involved. We are here to quantitatively understand these impacts. Fig. 8(a) shows that the sensing values are significantly affected by the (average) illuminance, and an illuminance lower than 60lux would affect the performance of CeilingSee. Fortunately, normal office area has an illuminance of 500lux and other public facilities such as supermarkets or theaters can reach 1000lux [24]. In the following, we maintain the average illuminance to 150lux: the one offered by the laboratory where CeilingSee is deployed. In Fig. 8(b), we vary the number of chips and check the resulting sensitivity. The results show that the unit with 96 chips appears to be the most cost-effective choice, hence leading to our design in Fig. 6.

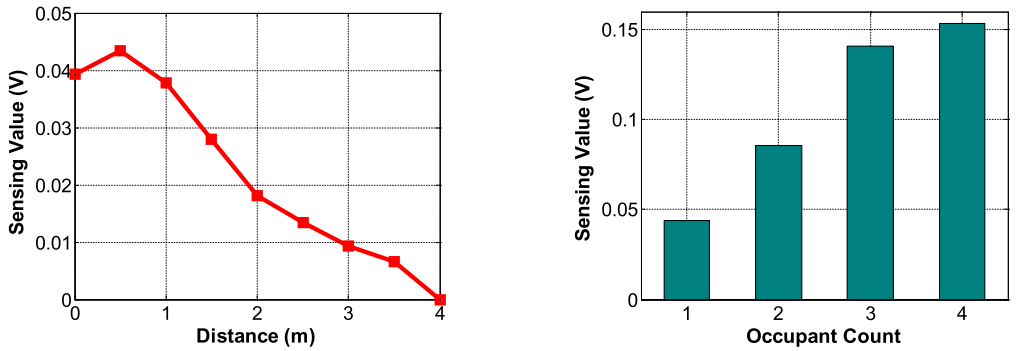
[5] As we deploy CeilingSee in a public research lab, the LED sensing units have to be co-located with existing fluorescent lights that cause the 50 Hz component. This happens to attest the compatibility of CeilingSee with legacy lighting systems.



(a) Sensing value vs. illuminance.

(b) Sensing value vs. LED chip number.

Fig. 8. Impact of ambient illuminance and chip number on sensitivity.



(a) Sensing value vs. distance.

(b) Sensing value vs. occupancy.

Fig. 9. Changes in sensing value caused by the occupant position and number.

2.2.3. Occupant position and number

In the previous experiments, we put an occupant right below a sensing unit. Now we let the occupant gradually move away from that center point, and record the corresponding sensing values in Fig. 9(a). Interestingly, whereas the values are generally decreasing in distance, it has a peak at 0.5 m. This stems from the fact that the perturbation in reflection incurred by the occupant depends on not only the distance but also occupant's cross section with respect to the unit; the tradeoff between this conflicting factors naturally lead to the peak. This is a nice property as it can effectively increase the FoV of one sensing unit.

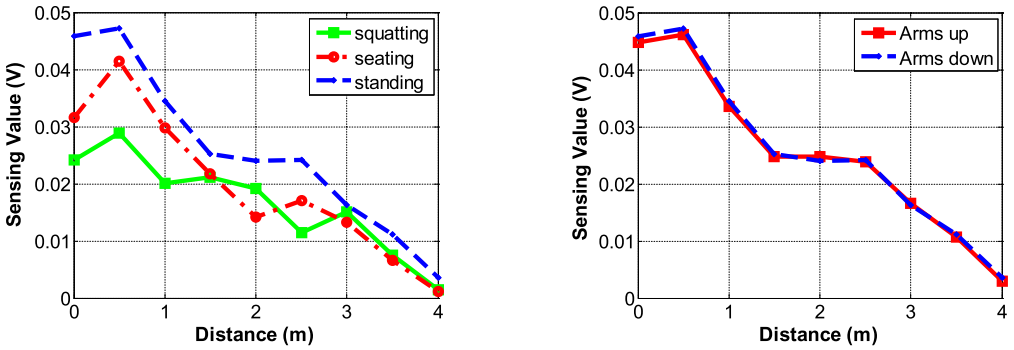
We also vary the number of occupants that stand within a circle of radius 1 m around the unit. Fig. 9(b) shows an almost linear increase in sensing values with the occupant number, though with some saturation at 4 occupants. Normally, we do not expect to have so crowded situations where more than 4 people standing upon a roughly 3 m^2 area, so the results demonstrate the ability of a single unit to count the number of occupants within its FOV.

2.2.4. Postures and gestures of an occupant

An occupant may have different postures and we consider three typical ones: standing, seating, and squatting (rare). We study the impact of these postures on the sensing values when the occupant is at various distances from a sensing unit. Fig. 10(a) shows that, though squatting does lead to rather low sensing values, standing and sitting cause very close sensing values. This is explained by the fact that the sensing value incurred by the occupant depends mainly on occupant's cross section. Though occupants may "hide" from CeilingSee by squatting, this is such a rare posture indoors that it would not cause much trouble to the overall inference performance. An occupant may also change gestures without leaving the monitored area, and such interference should not be responded by CeilingSee. In Fig. 10(b), we show that CeilingSee is virtually insensitive to gesture changes (e.g., waving arms) of an occupant, implying that the occupancy inference function is robust against such an interference.

2.2.5. Color and height of an occupant

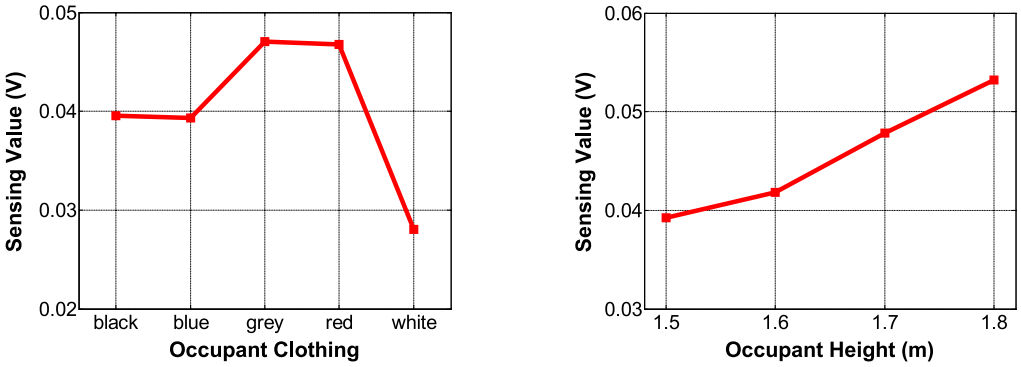
The dressing color and height of an occupant may also affect reflection, so we let one occupant dress in five different colors and also gather four occupants with heights varying from 1.5 m to 1.8 m. According to Fig. 11(a), CeilingSee is largely



(a) Effect of occupants' postures.

(b) Effect of occupants' gesture.

Fig. 10. Changes in sensing value caused by different postures and gestures.



(a) Effect of dressing color.

(b) Effect of occupant height.

Fig. 11. Changes in sensing value caused by different dressing colors and occupant heights.

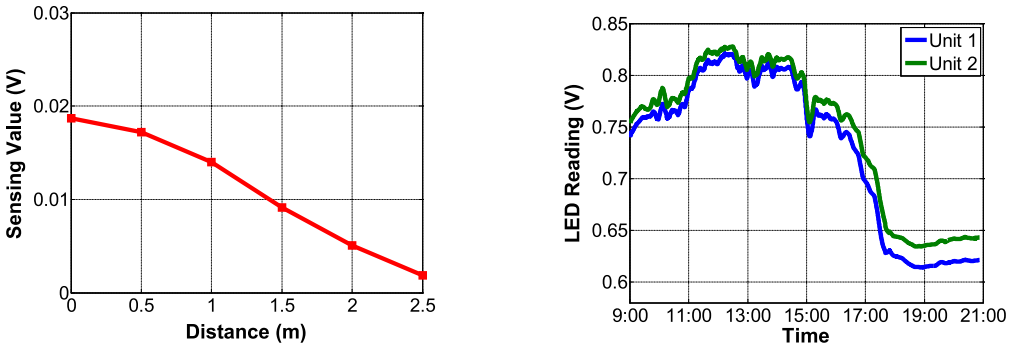
insensitive to dressing colors except white. Fortunately, as far as we do not have majority of the occupants dressing in white, the bias caused by white dressing can be offset by other dressing colors. Fig. 11(b) shows that CeilingSee is also quite insensitive to occupant height, which has somewhat been inferred by the posture results.

2.2.6. Background changes

The term sensing value (c.f. Section 2.2.1) is defined with respect to the nominal readings of individual sensing units. Apparently, these nominal readings can be affected by background changes, including furniture moving and natural light shedding in from windows. We study the impact of moving furniture by moving a chair away from the center of a unit, and we also choose two units very close to the window to check the variance of their nominal readings at different times of a day. According to Figs. 12(a) and 12(b), the impact of both factors are minor compared with the variance caused by occupants presence. Therefore, we conclude that CeilingSee is immune to minor background changes. In cases that indoor space being totally refurbished (e.g., repainted or rearranged), we may re-train CeilingSee based on new nominal readings.

3. Occupancy inference

We illustrate our system architecture in Fig. 13. The LED sensing measurements are first smoothed and sampled to obtain vectors of sensing values (termed *snapshots* hereafter) as described in Section 2.2. The snapshots are either directly taken or further differentiated to extract temporal features, and these inputs are fed to a fine-tuned regression module that is trained to perform occupancy inference. Details on how we train the regression module are presented in this section.



(a) Moving furniture.

(b) Varying natural light.

Fig. 12. Changes caused by moving furniture and varying natural light.

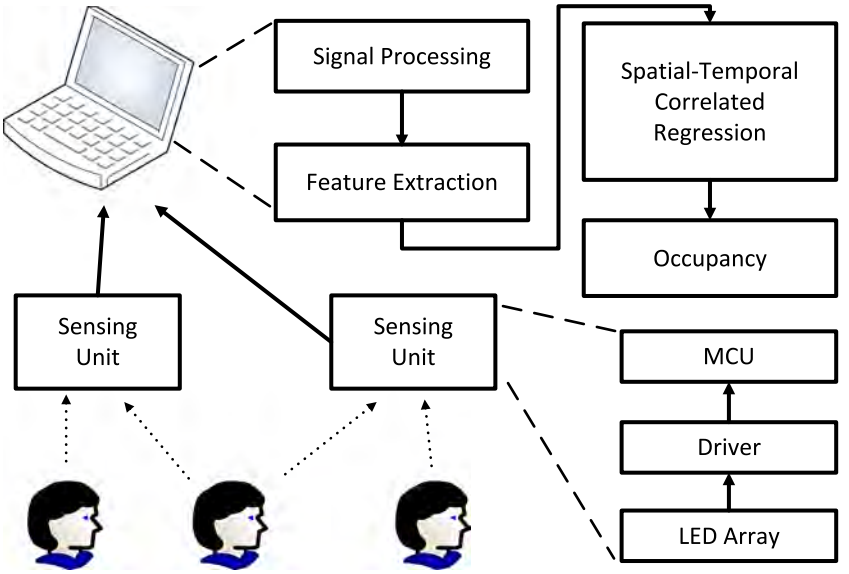


Fig. 13. System architecture. CeilingSee gathers data from all sensing units, pre-processes them to remove noise, and then passes them to the regression module for inferring occupancy.

3.1. Spatial distribution of sensing values

When multiple occupants exist in an area monitored by CeilingSee, each sensing unit will produce a sensing value depending on the spatial distribution of the occupants around it (as studied in Section 2.2). We denote the sensing value of a unit $x_\ell(t)$, where ℓ is the index of that unit and t is the time instant when the value is sampled. Combining all the values produced by the whole system, we end up with a time-varying vector $\mathbf{x}(t) = [x_1(t), x_2(t), \ldots, x_n(t)]$ where n is the number of units of CeilingSee. As we always sample $\mathbf{x}(t)$ in a discrete manner, we actually use $\mathbf{x}_i$ to denote a *snapshot* at the ith time slot, and the spatial distribution of the individual components of $\mathbf{x}_i$ is apparently correlated with that of the occupants during that time slot. We illustrate a few typical snapshots using our deployment shown in Fig. 15.

In Fig. 14, we show 8 snapshots captured when there are 4, 8, 12 occupants and each with two spatial distribution patterns, namely all standing under sensing units and away from those units. A direct observation is that, in general, more occupants usually yield higher overall sensing values and a higher sensing value indicates more occupants around that unit. These observations agree with the impact of distance/occupant number on sensing value shown in Fig. 9. More detailed inspections show that, with the same number of occupants at different positions, locating occupants right below the LED

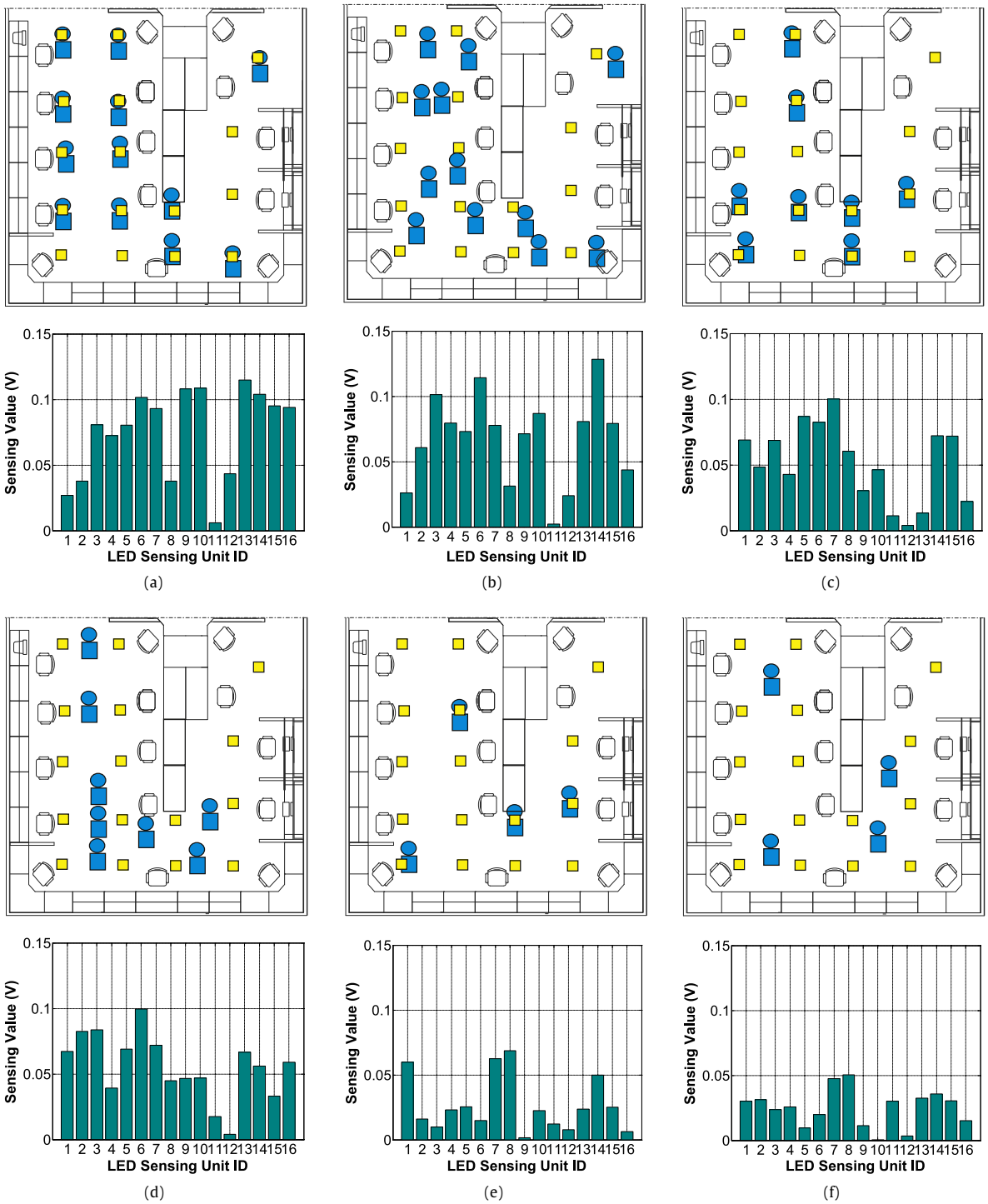


Fig. 14. Snapshots obtained by CeilingSee: different occupancy patterns lead to different snapshots. Upper figures show the occupancy patterns, with yellow squares represent the sensing units and blue human-shaped marks indicate occupants. Lower figures show the corresponding snapshots. (a) 12 occupants all under sensing units. (b) 12 occupants all away from sensing units. (c) 8 occupants all under sensing units. (d) 8 occupants all away from sensing units. (e) 4 occupants all under sensing units. (f) 4 occupants all away from sensing units.

sensing units actually leads to lower overall sensing values than moving them away from the units. This stems from our finding in Fig. 9(a). Nevertheless, directly inferring occupancy from individual snapshots through linear regression may yield rather coarse-grained estimations with a low accuracy.

3.2. Regularized regression

Given a training set $D = \{\mathbf{x}_i, y_i\}_{i=1,\ldots,m}$ where $\mathbf{x}_i \in \mathbb{R}^n$ denotes the input snapshots and $y_i \in \mathbb{R}$ denotes the labels (i.e. the occupancy count), we need to find a function $f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$ in the form of $f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b$ to fit the relationship between $\{\mathbf{x}_i\}$'s and $\{y_i\}$'s, where $\mathbf{w} \in \mathbb{R}^n$ is the vector of weight parameters and b is a bias factor (they are to be learned from the training set), and $\langle \cdot, \cdot \rangle$ is the inner product. To avoid the overfitting issue, we formulate the learning problem of $\mathbf{w}$ as a regularized regression problem by introducing a regularization term on $\mathbf{w}$ as follows,

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \left[\sum_{i=1}^{m} \mathcal{L}(f(\mathbf{x}_i), y_i) + \gamma \|\mathbf{w}\|_2^2 \right], \tag{4}$$

where $\mathcal{L}(\cdot)$ is a loss function and $\gamma \geq 0$ is a tradeoff parameter controlling the relative weight between the loss function and the regularization penalty $\|\mathbf{w}\|_2^2$. Different definitions of the loss function lead to specific regularized regression methods, and we adopt ε-insensitive loss that is described by

$$\mathcal{L}(f(\mathbf{x}), y) = |f(\mathbf{x}) - y|_\varepsilon = \begin{cases} 0 & \text{if } |f(\mathbf{x}) - y| \leq \varepsilon, \\ |f(\mathbf{x}) - y| - \varepsilon, & \text{otherwise.} \end{cases}$$

This leads to the well-known Support Vector Regression (SVR) [25]. Moreover, the function $f(\mathbf{x})$ learned by solving (4) can only capture the linear relationship between $\{\mathbf{x}_i\}$'s and $\{y_i\}$'s, whereas the intrinsic relationship between snapshots and the corresponding occupancy counts can be highly nonlinear. To this end, we further formulate the learning problem of $\mathbf{w}$ as a nonlinear regularized regression problem by introducing a nonlinear feature map $\phi(\mathbf{x})$ that maps $\mathbf{x}$ to a Reproducing Kernel Hilbert Space (RKHS),

$$f^*(\mathbf{x}) = \arg\min_{f} \left[\sum_{i=1}^{m} \mathcal{L}(f(\mathbf{x}_i), y_i) + \gamma \|f\|_{\mathcal{H}}^2 \right], \tag{5}$$

where $\|\cdot\|_{\mathcal{H}}$ is the norm in the RKHS. By using the kernel trick, i.e., $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$, we can obtain the minimizer of the optimization (5):

$$f^*(\mathbf{x}) = \sum_{i=1}^{m} u_i \mathcal{K}(\mathbf{x}_i, \mathbf{x}) + b, \tag{6}$$

where u_i is obtained by solving the dual problem of (5) [25]. Our design now focuses on choosing a proper kernel function $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)$ and pre-processing $\{\mathbf{x}_i\}$'s so as to improve the inference performance.

3.3. Handling spatial–temporal correlations

In order to achieve an accurate inference, we fine-tune the regression by taking into account three types of correlations among the training snapshots. As shown in Section 3.1, snapshots with similar labels are correlated, while the individual sensing values (collected by sensing units geographically distributed in a monitoring area) have spatial correlations. Moreover, the snapshots taken at consecutive time slots can be correlated depending on whether some occupants move or not.

The correlations among snapshots are handled by applying a Gaussian kernel $\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) = e^{-\nu \|\mathbf{x}_i - \mathbf{x}_j\|^2}$ with $\nu > 0$. Such a kernel aims to bring down the interferences between two rather "dissimilar" snapshots during the training process. The spatial correlation among sensing values can be handled by pre-processing the training data through the Geographically Weighted Regression (GWR), where each snapshot $\mathbf{x}$ is multiplied by a symmetric matrix W:

$$W(k, \ell) = \begin{cases} e^{-\mu(d_{k\ell}/h)^2} & \text{if } \ d_{k\ell} < h \\ 0 & \text{otherwise,} \end{cases} \tag{7}$$

where $d_{k\ell}$ denotes the Euclidean distance between the kth and ℓth sensing units and the "bandwidth" h is set according to the FoV of our sensing units.

The aforementioned approaches may work well when occupants are static. When occupants are moving, the individual snapshots can be rather unstable. Nevertheless, the occupant motion also provides more information, e.g., variance among

two consecutive snapshots. Apparently, minor temporal variances in the snapshots often indicate unchanged occupancy, while major ones may indicate otherwise. To take the advantage of this increased information dimension, we expand every snapshot $\mathbf{x}_i$ by further involving its variance with respect to the previous snapshot, so the new snapshot has the form of $[\mathbf{x}_i, \mathbf{x}_i - \mathbf{x}_{i-1}] \in \mathbb{R}^{2n}$.

3.4. Incremental inference

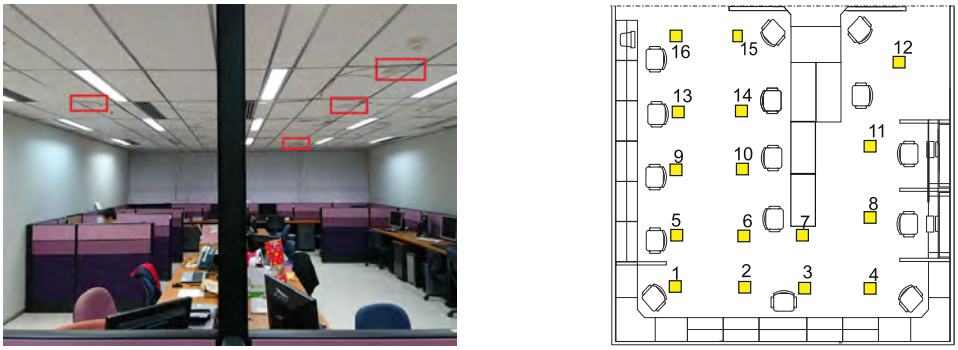
Due to the dynamics of a real-world environment, the learning-based system set up using training data pre-collected offline may be out-of-date, and may thus perform poorly online. To adapt the system to environmental dynamics, a few training data need to be collected online periodically (normally every week). We also implement an incremental or online algorithm to update the trained SVR model whenever a new training sample is collected in real time [26], which entails a very efficient adaptation to a dynamic environment without the need for retraining from scratch.

4. System evaluation

In this section, we evaluate the performance of CeilingSee in terms of inference accuracy, latency and power consumption through extensive field experiments.

4.1. Experimental setup

We deploy a CeilingSee testbed in one of our university laboratories; it covers an area of 5 m×6 m. The testbed consists of 16 LED lighting/sensing units mounted on the ceiling as shown in Fig. 15(a). We try to deploy the units in 1.25 m×1.25 m grid, but we have to slightly adjust the positions of some units adapting to the office layout. As briefly presented in Section 2.2, each sensing unit includes a 8 × 12 LED chips array and a re-designed driver. We connect all drivers to a Lenovo ThinkPad laptop computer via the serial port, so the laptop acts as a lightweight server to process the sensing data and in turn to deliver occupancy inferences.

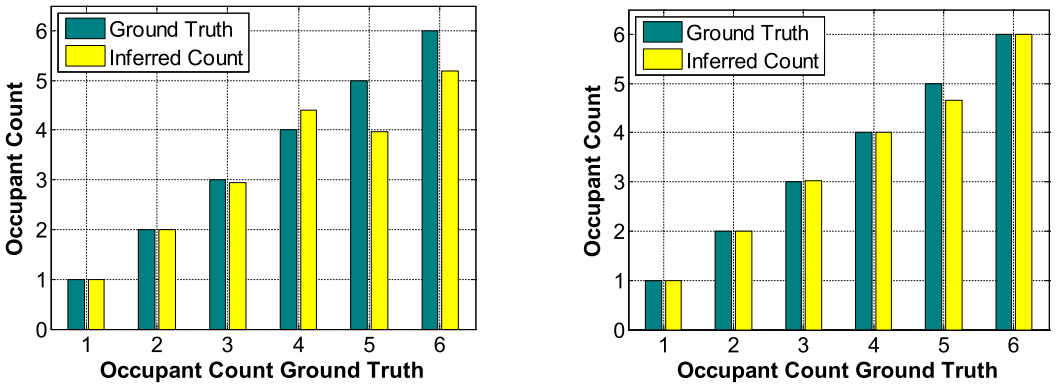


(a) Testbed in photo. (b) Sensing units layout.

Fig. 15. CeilingSee testbed overview. Sixteen sensing units are deployed on the ceiling covering a 5m×6m indoor area.

The MCU of our driver, CC2541, is a low-cost power-optimized system-on-chip module, running on up to 32 MHz and supporting 12-bit analog-to-digital conversions [27]. Most importantly, CC2541 is designed for Bluetooth Low Energy (BLE). Therefore, although we currently use the serial port to connect sensor units to the server, we plan to make CeilingSee a fully wireless system by using TI's BLE-STACK [28] to upload sensing data. In our testbed, we run the MCU at 16 MHz and let its ADC sample at 500 Hz, and the low-pass filter discussed in Section 2.2.1 is performed by the MCU so that the snapshots are uploaded to the server only at 10 Hz, given the low variation rate of the reflection.

This testbed has been running for about 6 months, during which we have kept monitoring the regular occupancy of the deployment site, and we have also invited groups with up to 20 volunteers to perform specific experiments on the system. All the experiments are performed based on two occupancy patterns: (i) *static pattern* where occupants stand or sit at arbitrary locations, and (ii) *dynamic pattern* where all occupants walk or even run freely in the lab. For each pattern, the number of occupants varies from 1 to 20, and we encourage them to change their postures between standing and sitting, as well as to perform other daily activities during the tests. We gather more than 10,000 snapshots for each pattern; they are all labeled manually.



(a) Single sensing unit.

(b) Multiple sensing units.

Fig. 16. Comparing the accuracy of occupancy inference between (a) single sensing unit and (b) multiple sensing units.

4.2. Single sensing vs. multiple sensing

We first evaluate the detection capacity of one sensing unit in CeilingSee. We arbitrarily select one of 16 sensing units (Unit 6 in Fig. 15(b)) to detect nearby occupants within its sensing range. We choose the radius of FoV as 1.5 m since it corresponds to a relatively robust sensing value as shown in Fig. 9(a). We assume the maximum occupants within a circle of radius 1.5 m is 6 as otherwise it would be too crowded. We collect the data when 0 to 6 occupants standing or sitting within the FoV of Unit 6. Applying the regularized regression on the sensing values obtained by Unit 6, Fig. 16(a) shows the estimated occupancy counts with corresponding ground truth values. An estimated occupancy count is calculated as the average of over 1000 inferences for the true occupancy counts, so the values are not integers anymore. We observe that the inference for fewer occupants are more accurate and the accuracy decreases slightly with an increasing occupancy count. This can be explained by the fact that the occupants' cross sections overlap with each other when the density increases. The overall accuracy of Unit 6 is only 77.9%, with the accuracy metric defined as the ratio between the number of correct inferences and that of all tests.

$$\frac{\sum_{i=1}^{c} \mathcal{I}_{f^*(\mathbf{x}_i)=y_i}}{c}, \tag{8}$$

where $\mathcal{I}$ is the indicator function and c is the cardinality.

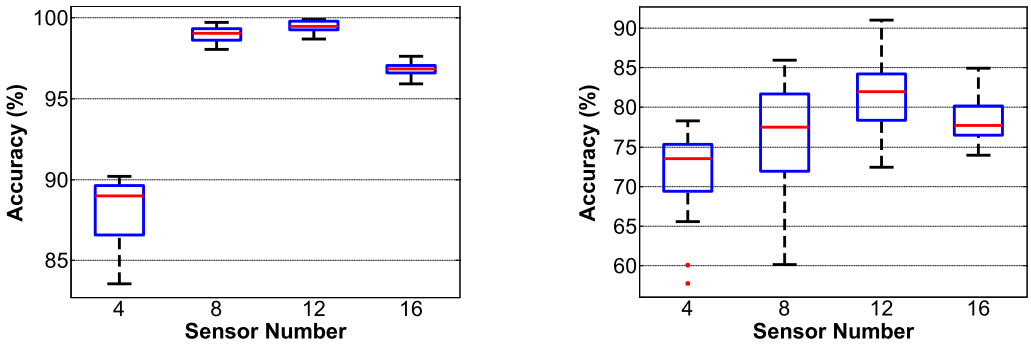
Intuitively, applying multiple sensing units would improve the inference accuracy as it yields a high dimension in sensing data. Now we intend to study to what extend the inference accuracy is improved when multiple sensing units are used collectively. Given the same occupant settings, we gather the sensing values from Units 1, 2, 3, 5, 6, 7, 9, 10 as the inputs for regression, which lead to the results shown in Fig. 16(b). It boosts the overall inference accuracy up to 96.8%, 19% higher than that of single sensing unit.

4.3. Impact of sensor density and training intensity on inference accuracy

Before evaluating the performance of CeilingSee, we firstly experimentally study the two design parameters of CeilingSee, namely how dense the sensing units should be deployed and how much training is needed. For the first aspect, we select 4, 8, and 12 readings out of each snapshot to derive new snapshots, so that we can evaluate the impact of sensor density. For the second aspect, we take a substantial fraction of the labeled data as testing data while using the remaining fraction for the training purpose. We vary the Test to Total Ratio (TTR) from 90% to 99%. To better understand the accuracy performance, we introduce a new accuracy metric that allows for a certain number of miscounts indicated by a non-negative integer ϵ. In particular, we define *accuracy with miscount tolerance* as:

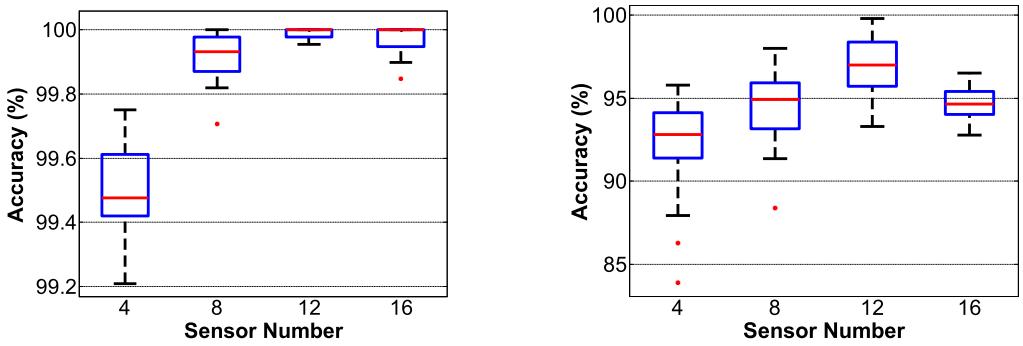
$$\frac{\sum_{i=1}^{c} \mathcal{I}_{|f^*(\mathbf{x}_i)-y_i|\leq\epsilon}}{c}. \tag{9}$$

Apparently, setting $\epsilon = 0$ would bring us back to the normal accuracy metric defined in (8). Fig. 17 shows the impact of the number of sensing units given $\epsilon = 0$. While 8 units appear to already offer very good accuracy for static patterns, 12 units perform the best in both cases and they would be necessary to cope with dynamic situations. Denser deployments beyond 12 units are clearly not beneficial. Fig. 18 further shows the statistics given $\epsilon = 1$. Apparently, the relatively low accuracy for dynamic patterns with $\epsilon = 0$ is mostly due to a single miscount, as raising ϵ to 1 would allow even 4 units to



(a) Static pattern. (b) Dynamic pattern.

Fig. 17. Accuracy vs. varying number of sensing units with $\epsilon = 0$.



(a) Static pattern. (b) Dynamic pattern.

Fig. 18. Accuracy vs. varying number of sensing units with $\epsilon = 1$.

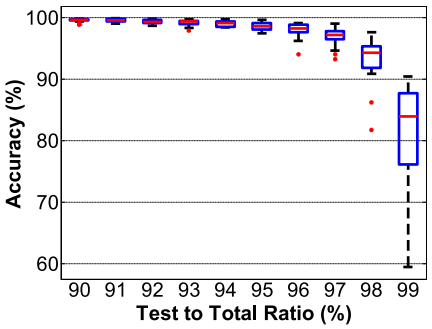
almost always achieve an accuracy beyond 90% under both patterns, although the better performance of static patterns and the superiority of 12 units still remain. We will continue observing the performance difference between these two patterns, which shall be further explained in Section 4.5. To get the highest accuracy, we will keep using 12 units for the remaining experiments.

Figs. 19 and 20 evaluate the impact of training intensity by varying TTR from 90% to 99%. As expected, the higher the TTR (hence less training), the worse the performance (in terms of the mean and variance) is, but the performance degrades in a rather graceful manner. In fact, if one miscount can be tolerated, only 2% training data would be needed for static pattern and 5% for dynamic one. Given our abundant training data, we stick to 90% TTR for the remaining experiments to obtain the best performance.

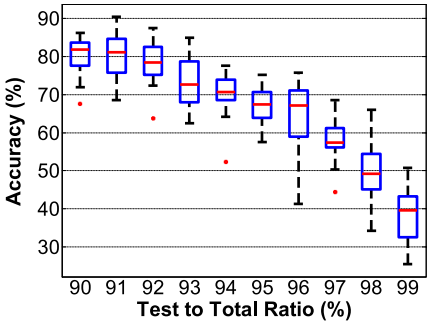
4.4. Breakdown of inference accuracy

We now study the occupancy inference accuracy with respect to varying occupancy counts; the statistics are reported for static pattern in Fig. 21 and dynamic pattern in Fig. 22. We observe that the accuracy based on static pattern is always higher than 97% for all occupancy counts, and tolerating one miscount brings almost all of them to 100%. Though the accuracy for dynamic pattern appears to be rather disappointing, the majority of the miscounting cases involve only one miscount, because setting $\epsilon = 1$ causes drastic improvements to all occupancy counts.

Under static patterns, the inference accuracy is relatively stable with various occupancy counts, while it is generally degrading as the occupancy count increases for both $\epsilon = 0$ and $\epsilon = 1$ under dynamic patterns. This can be largely attributed to the drastic increase in transient states when more occupants are constantly moving, as we shall elaborate in Section 4.5. All in all, if we simply allow one miscount, the inference accuracy based on both static and dynamic patterns can be maintained above 90% for all tested occupancy counts. Therefore, we can confidently conclude that CeilingSee achieves a very promising inference accuracy by using only existing lighting infrastructure. Note that we refrain from comparing CeilingSee with existing proposals, because, on one hand, it is unfair to do so as CeilingSee does not depend on extra infrastructure, and on the other hand, CeilingSee is not meant to replace other systems, but rather acts as a lightweight complementary solution.

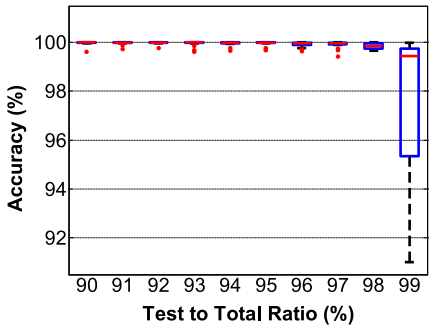


(a) Static pattern.

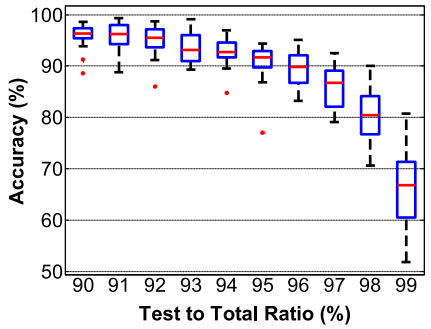


(b) Dynamic pattern.

Fig. 19. Accuracy with varying TTR given $\epsilon = 0$.



(a) Static pattern.



(b) Dynamic pattern.

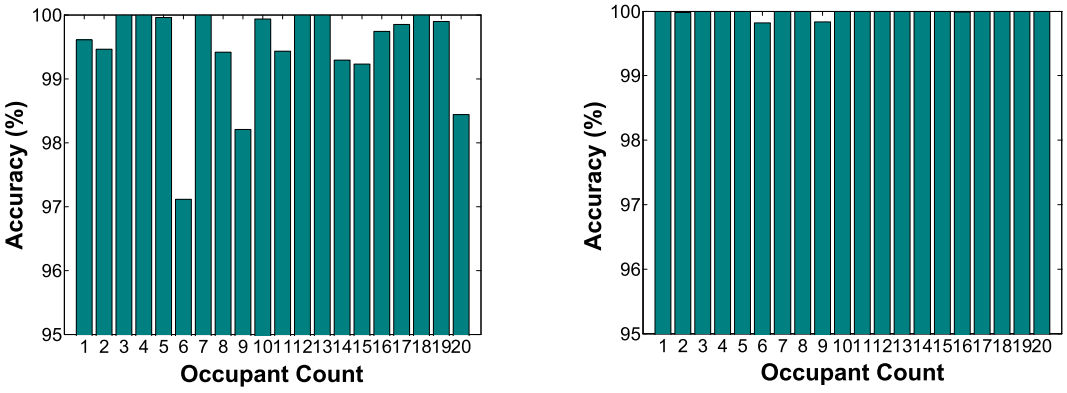
Fig. 20. Accuracy with varying TTR given $\epsilon = 1$.

4.5. Responsiveness and real life scenarios

As CeilingSee may serve as input to indoor energy management systems (e.g., for HVAC), its responsiveness is a concern and hence entails the need for evaluating how quickly CeilingSee can respond to changes in occupancy count. CeilingSee's sensing units configure their sample rates of ADC as 500 Hz. A moving average filter in the MCU processes the output of ADC and produces snapshots at 10 Hz, which causes at most a 100 ms delay. The transmissions of snapshots to the CeilingSee's server via serial port are done at a baud rate of 115,200, consuming less than 1 ms. The server takes less than 1 ms on average to figure out the occupancy counts by operating our regression algorithm. Therefore, inference latency is about 100 ms in total which is mainly consumed by sampling and preliminary data processing on MCU. We could further reduce this delay by running the filter with a smaller window size, but it does not appear to be necessary, because the variations in LED readings take a much slower pace (in seconds), as shown by Fig. 23(a) when an occupant passes by 4 units and stops at the last. In fact, the current high level of responsiveness is one of the main reasons that cause miscounts under dynamic pattern. As shown in Fig. 23(b), the occupancy counts reported by CeilingSee may oscillate during the time period between two actual count changes if occupants keep moving. This apparently causes miscounts as we have experienced earlier, but it is the price the system has to pay for instantaneously detecting count changes.

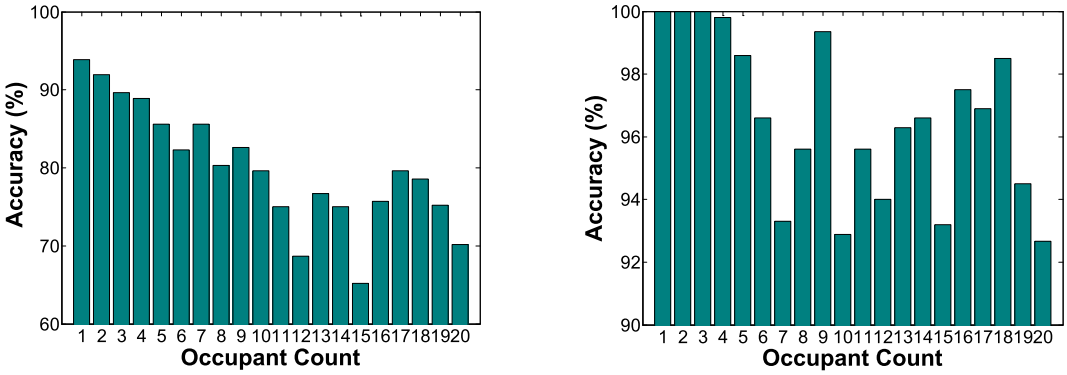
In order to put the performance of CeilingSee into practical perspective, we choose to report a whole day monitoring data (for 18 February, 2016) among several months operation of the system. In Fig. 24(a), we plot the inferred occupancy count (without ground truth) from 9am to 22pm. The reported data exhibit a rather plausible pattern: occupants start to arrive in the early morning, they leave for lunch during the noon but come back after lunch; more occupants show up in the afternoon, but most of them leave around 6pm, leaving only a few working till late evening. Between two occupancy count changes, the actual daily activity pattern is a mixture of both static and dynamic patterns: it can be rather stable while most of the occupants are sitting (presumably working), while it fluctuates from time to time due to the activities such as taking a short break, mingling with other colleagues, and so on.

We also arbitrarily choose three sensing units, unit 8, 10, 16, to plot their readings in Fig. 24(b). The trends of these readings are consistent with that of occupancy count. In particular, sensor readings vary only slightly when there are fewer occupants in the morning, lunch time and off work time, whereas it changes more actively when there are relatively more occupants in the afternoon (as the chance of their moving becomes higher). If we further look into these readings, for example



(a) $\epsilon = 0$. (b) $\epsilon = 1$.

Fig. 21. Accuracy with varying number of occupants under static pattern.



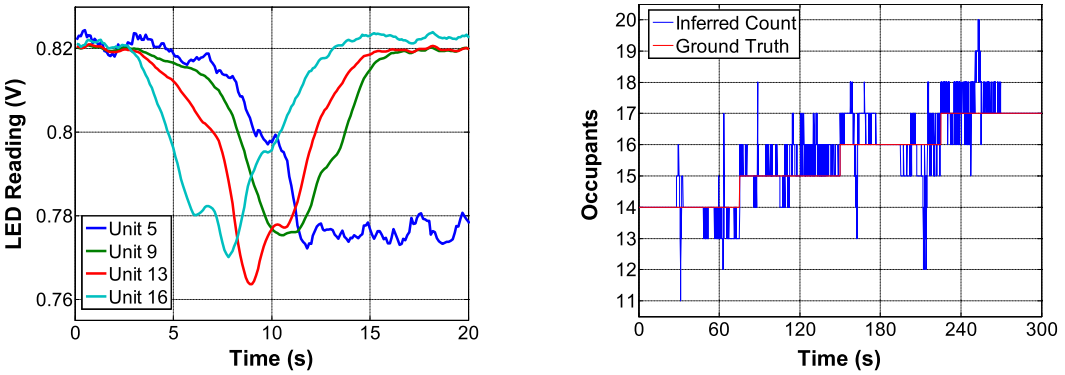
(a) $\epsilon = 0$. (b) $\epsilon = 1$.

Fig. 22. Accuracy with varying number of occupants under dynamic pattern.

those around 14:00 and 15:00, fluctuations are so intense that the sampled snapshots can vary rapidly. The resulting large variations in snapshots reduce the effectiveness of regression and hence the inference accuracy. This explains why CeilingSee performs less accurately under dynamic pattern as reported in Sections 4.3 and 4.4. Fortunately, as we have shown, the miscounts caused by such fluctuations are largely negligible. We could apply Hidden Markov Model to maintain the temporal consistency of occupancy count so as to improve the inference accuracy under dynamic pattern. Nevertheless, it would certainly retard the response of CeilingSee to actual count changes.

4.6. Energy consumption

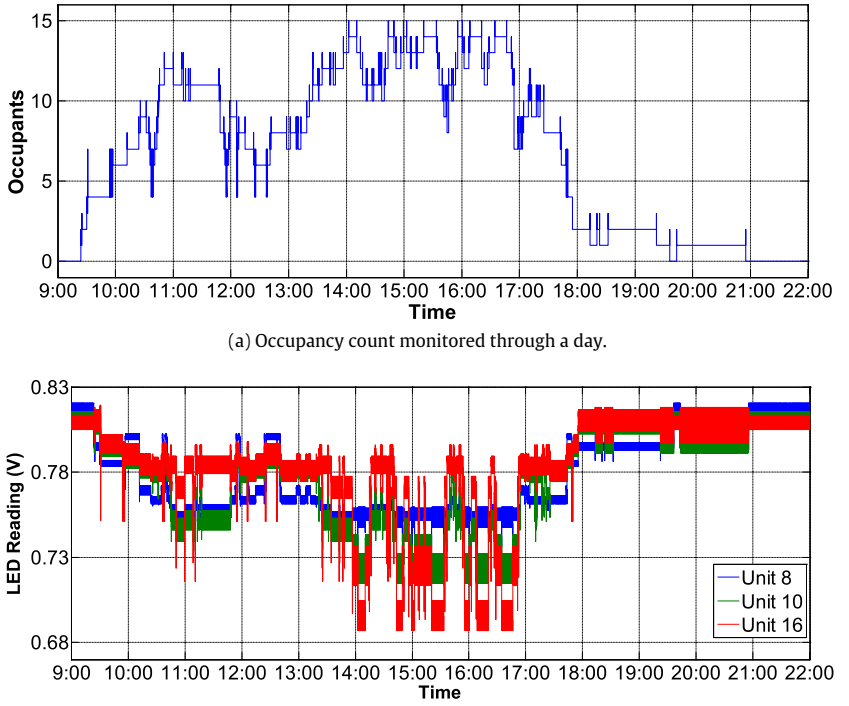
Since CeilingSee utilizes existing lighting infrastructure as light sources, we do not count in the energy consumption for illumination. Consequently, the energy consumption of CeilingSee mainly involves the energy consumed by the driver circuits and microcontrollers of the LED sensing units. The driver circuit of one sensing unit works at a DC voltage of $\pm$ 5 V. It consumes 10.1 mW when it works in the sensing state, and at most 50 mW when working in illuminating state. The energy consumption of the MCU is 24 mW. Therefore, the total energy consumption is 34.1 mW for a typical CeilingSee sensing unit under its sensing state. As MCUs and drivers are default components in general LED illumination devices, the extra power consumption of our CeilingSee sensing unit is only 10.1 mW. This extra power consumption is incurred by the amplifier circuits on re-designed driver. Moreover, since data transmission of sensed data can also be integrated into VLC or Power Line Communication (PLC) [29], the power consumption for data transmission can be neglected. The energy consumption caused by computations on the laptop can also be ignored given that the laptop is working on many other tasks at the same time. All in all, we can conclude our CeilingSee is a truly low-cost energy-efficient system for occupancy inference. We summarize the power consumption in Table 2.



(a) Variations in LED readings.

(b) Oscillations under dynamics.

Fig. 23. Dynamic response of CeilingSee.



(a) Occupancy count monitored through a day.

(b) Selected LED readings during the same period.

Fig. 24. A real life monitoring scenario for one day.

Table 2
Power Consumption of One Sensing Unit.

	Sensing	Sensing & illuminating
Driver	10.1 mW	50.0 mW
MCU	24.0 mW	24.1 mW
Total	34.1 mW	74.1 mW

5. Discussions

Though CeilingSee exhibits promising performance in occupancy inference, we have also recognized some of its limitations and potential extensions as well.

5.1. Light dimming

As CeilingSee senses the variance in diffuse reflection caused by occupants to infer the occupancy, the light dimming requirement by certain indoor facilities will certainly affect its performance. For CeilingSee to effectively work, our current design requires an illuminance level around 150 lux. While such a level is normal in office area, it will not be the case if we extend CeilingSee to monitor the occupancy in areas such the basement of a building. For such difficult applications, our current make-shift is to increase the gain of amplifier circuits, but we are certainly on the way to look for more effective solutions.

5.2. Improved learning methodology

Currently, we train only one regression module for overall inference. In fact, the model may be time-variant. Therefore, one potential improvement of CeilingSee in terms of its inference accuracy is to train different regression models for different time slots. Also, our current training is fully supervised, which limits the ability of taking un-labeled data into account. So we also plan to employ a related semi-supervised learning method in order to improve the scalability of CeilingSee in handling large indoor deployments such as auditoriums.

5.3. User tracking and identification

According to our experiments, CeilingSee can not only infer the presence of occupants, but also deduce possible positions of the occupants [30]. As shown in Fig. 23(a), an occupant passing by 4 units can be clearly inferred by the sequence of sensor readings, and in particular, the time difference between two consecutive "valleys" can even allow us to compute the moving speed of that occupant. Moreover, the results shown in Fig. 9(a) can be applied to refined the position of an occupant relative to a certain sensing unit. Of course, detecting individual moving traces when multiple occupants are moving arbitrarily can be very challenging, especially if their moving traces cross each other. Nevertheless, we believe our experience with CeilingSee has shed light on device-free localization and tracking by using existing lighting infrastructure, and it has the potential to be extended to a full-fledged tracking system by, for example, incorporation with a location based VLC.

5.4. Occupancy inference with natural light

Although our current design focuses on only indoor environment, CeilingSee does have the potential to be applied to outdoor occupancy inference for both human users or vehicles. For example, we may deploy such a sensing system on street lights, which would cover streets lying below them. Such an extension can greatly expand CeilingSee's application scenarios, but the difficulty there is that the sensing range has to be significantly boosted beyond what the current design can manage, so there could be a long way to go towards such extensions.

6. Related work

We survey two major bodies of literature that are closely related to CeilingSee in this section, namely occupancy inference and VLC/VLS. Some preliminary work on CeilingSee has been reported in [31], yet this journal version delivers a holistic study on CeilingSee, by providing further details on the system design, as well as the system properties and limitations.

6.1. Occupancy inference

PIR sensors have been heavily used for the purpose of occupancy inference since the very beginning [32,4,5]. Conventional PIR sensors have a wide coverage but individual sensors only deliver binary indicators [9], so sophisticated statistical learning method has to be in place to properly infer occupancy [4]. ThermoSense [5] complements PIR sensors with a ceiling-mounted infrared thermal array to "count" individual figures so that more accurate sensing outcome can be obtained. Our CeilingSee adopts the same ceiling-mounted configuration, but it piggybacks on existing lighting system without entailing an extra sensing infrastructure.

Unlike PIR sensor, acoustic or ultrasonic sensors require active sound/ultrasound emissions to detect the variation in reflections, which is in turn used to indicate the presence of occupants [32]. While these sensors are more sensitive than PIR sensors, they are more prone to false triggering as door/window opening and even duty-cycling of HVAC systems may cause false indications (these inferences were not accounted for in a recent implementation of ultrasonic occupancy inference system [8]). To confine the weakness of ultrasonic sensors, Doorjamb [7] mounts ultrasonic range finders only above each doorway, so that it not only counts people entering/leaving a room but also identify (thus track) individual

persons by measuring their heights. Such a system should work well in residences but may have trouble in tracking multiple entering/leaving in public facilities (e.g., office or airport). Using light rather than sound as the media, CeilingSee is totally immune to the all the aforementioned drawbacks.

Camera-based occupancy inference has been intensively studied in the computer vision community for more than a decade [33,9,10]. All of these approaches rely on rather sophisticated (yet not very reliable) image process algorithms to extract the human silhouettes from the background [33,9], while most recent literature applies certain cutting-edge machine learning algorithms to enhance the detection reliability (hence accuracy) [10]. In fact, camera-based approaches appear to be the worst choice from a practical point of view: it entails a heavy infrastructure (several cameras have to be strategically placed and the data processing complexity grows drastically with the deployment scale), while its infringement to user privacy cannot be overlooked. As a result, later developments tend to complement cameras with PIR sensors for better scalability [20].

6.2. Visible light communication (VLC)

Since the seminal work by Komine and Nakagawa [13], VLC has been approached from both physical layer design (e.g., [34,35]) and application/system development (e.g., [14,18]). We only discuss the latter given the system nature of our work.

The most direct application of VLC is to make use its communication ability for providing data service to users. Though the high speed communication techniques (e.g., [35]) are still in its experimental phase, practical systems have been developed based on three major mechanisms: (1) LED–LED [19,14], (2) screen-camera [36–38,16], and (3) LED-camera [15,39,40]. Whereas LED–LED communication has a long history, the potential is low due to the limited frequency response ability of LED that leads to the lowest data rate among the three ($<$1 kb/s [14]). Nevertheless, it does serve as a motivation of using LED as sensors by our CeilingSee. Screen-camera communication appears to be the most popular approach as it may obtain the highest data rate among the three (up to several tens of kb/s [37]); this is due to the large screen area that allows for various coding and modulation schemes to be used. Although LED-camera communication only provides a moderate 1–2 kb/s data rate, it can be readily set up thanks to the pervasive adoption of LED lighting systems.

The easy deployment LED-camera communication has inspired a new application of VLC, namely *Visible Light Position* (VLP) [1–3]. The idea is to use the moderate data rate of LED-camera links to transmit only location indicators, so that a user's smartphone can pick-up a few such indicators to locate itself. Extending from this idea, Zhou and Campbell [41] propose a more general concept of *Visible Light Sensing* (VLS), which goes beyond VLP by not only locate but also identify users and their postures [18,42]. In particular, Li et al. [18] propose to deploy light sensors (photodiodes) on floors to sense the shadows of a person so as to deduce his/her postures. Whereas this proposal can be used for occupancy inference, the sensing floor entails yet another infrastructure to deploy. Our CeilingSee utilizes LED lighting systems to sense the variance in diffuse reflection, removing the reliance on any extra infrastructure and hence pushing VLS to a new frontier.

7. Conclusions

In this paper, to automatically estimate indoor occupancy, we have developed a device-free system, CeilingSee, that piggybacks on existing LED lighting infrastructure. The system consists of two main components: (1) a re-designed LED driver, which leverages LED's photoelectric effect to transform a light emitter to a light sensor, so as to obtain sensing values in the form of snapshots at any time, and (2) a machine-learning-based algorithm to infer indoor occupancy using the snapshots as input in real time. To verify the efficiency and effectiveness of the developed system, we have conducted extensive experiments in a testbed covering a 30 m^2 office area. The experiment results have shown very promising performance of CeilingSee and hence demonstrated its great potentials to be applied to many smart building applications. In our future work, we plan to develop various intelligent systems to provide personalized services or security monitoring on top of this system, as well as to extend its application to outdoor environments.

Acknowledgments

The authors would like to thank the anonymous reviewers for their constructive feedback and valuable input. Special thanks to IEEE PerCom 2017 team and attendees, especially Christine Julien and Marco Zuniga, for their time and valuable discussions. This work was mainly supported by the AcRF Tier 2 Grant MOE2016-T2-2-022, the DSAIR Center at NTU, and National Natural Science Foundation of China under Grant No. 61373091. The research of Jie Hao was also supported by National Natural Science Foundation of China under Grant No. 61602242 and Natural Science Foundation of Jiangsu Province under Grant No. BK20160807.

References

[1] L. Li, P. Hu, C. Peng, G. Shen, F. Zhao, Epsilon: A visible light based positioning system, in: Proc. of 11th USENIX NSDI, 2014, pp. 331–343.
[2] Y. Kuo, P. Pannuto, K. Hsiao, P. Dutta, Luxapose: Indoor positioning with mobile phones and visible light, in: Proc. of 20th ACM MobiCom, 2014, pp. 447–458.
[3] Z. Yang, Z. Wang, J. Zhang, C. Huang, Q. Zhang, Wearables can afford: Light-weight indoor positioning with visible light, in: Proc. of 13th ACM MobiSys, 2015, pp. 317–330.
[4] R. Dodier, G. Henze, D. Tiller, X. Guo, Building occupancy detection through sensor belief networks, Elsevier Energy Build. 38 (9) (2006) 1033–1043.
[5] A. Beltran, V.L. Erickson, A.E. Cerpa, ThermoSense: Occupancy thermal based sensing for HVAC control, in: Proc. of the 5th ACM BuildSys, 2013, pp. 11:1–11:8.
[6] S.P. Tarzia, R.P. Dick, P.A. Dinda, G. Memik, Sonar-based measurement of user presence and attention, in: Proc. of the 11th ACM UbiComp 2009, 2009, pp. 89–92.
[7] T.W. Hnat, E. Griffiths, R. Dawson, K. Whitehouse, Doorjamb: Unobtrusive room-level tracking of people in homes using doorway sensors, in: Proc. of the 10th ACM SenSys, 2012, pp. 309–322.
[8] O. Shih, A. Rowe, Occupancy estimation using ultrasonic chirps, in: Proc. of the 6th ACM/IEEE ICCPS, 2015, pp. 149–158.
[9] V. Erickson, M. Carreira-Perpinan, A. Cerpa, OBSERVE: Occupancy-based system for efficient reduction of HVAC energy, in: Proc. of the 10th ACM/IEEE IPSN, 2011, pp. 258–269.
[10] C.L. Chen, S. Gong, T. Xiang, From semi-supervised to transfer counting of crowds, in: Proc. of the 14th IEEE ICCV, 2013, pp. 2256–2263.
[11] B. Balaji, J. Xu, A. Nwokafor, R. Gupta, Y. Agarwal, Sentinel: Occupancy based HVAC actuation using existing WiFi infrastructure within commercial buildings, in: Proc. of the 11th ACM SenSys, 2013, pp. 17:1–17:14.
[12] L. Yang, K. Ting, M. Srivastava, Inferring occupancy from opportunistically available sensor data, in: Proc. of the 12th IEEE PerCom, 2014, pp. 60–68.
[13] T. Komine, M. Nakagawa, Fundamental analysis for visible-light communication system using LED lights, IEEE Trans. Consum. Electron. 50 (1) (2004) 100–107.
[14] S. Schmid, G. Corbellini, S. Mangold, T.R. Gross, LED-to-LED visible light communication networks, Proc. of the 14th ACM MobiHoc, 2013, pp. 1–10.
[15] H.-Y. Lee, H.-M. Lin, Y.-L. Wei, H.-I. Wu, H.-M. Tsai, K.C.-J. Lin, RollingLight: Enabling line-of-sight light-to-camera communications, in: Proc. of the 13th ACM MobiSys, 2015, pp. 167–180.
[16] T. Li, C. An, X. Xiao, A.T. Campbell, X. Zhou, Real-time screen-camera communication behind any scene, in: Proc. of the 13th ACM MobiSys, 2015, pp. 197–211.
[17] Y. Yang, J. Hao, J. Luo, CeilingTalk: Lightweight indoor broadcast through LED-camera communication, IEEE Trans. Mobile Comput. 16 (12) (2017) 3308–3319.
[18] T. Li, C. An, Z. Tian, A.T. Campbell, X. Zhou, Human sensing using visible light communication, in: Proc. of the 21st ACM MobiCom, 2015, pp. 331–344.
[19] P. Dietz, W. Yerazunis, D. Leigh, Very low-cost sensing and communication using bidirectional LEDs, in: Proc. of the 5th ACM UbiComp 2003, 2003, pp. 175–191.
[20] V.L. Erickson, S. Achleitner, A.E. Cerpa, POEM: Power-efficient occupancy-based energy management system, in: Proc. of the 12th ACM/IEEE IPSN, 2013, pp. 203–216.
[21] M. Ibrahim, V. Nguyen, S. Rupavatharam, M. Jawahar, M. Gruteser, R. Howard, Visible light based activity sensing using ceiling photosensors, in: Proc. of the 3rd ACM VLCS Workshop, 2016, pp. 43–48.
[22] F.M. Mims, Siliconnections: Coming of Age in the Electronic Era, McGraw-Hill, New York, 1986.
[23] LUXEON XF-3535L, http://www.lumileds.com/products/matrix-platform/luxeon-xf-3535l.
[24] Illuminance, http://www.engineeringtoolbox.com/light-level-rooms-d_708.html.
[25] A.J. Smola, B. Schölkopf, A tutorial on support vector regression, Stat. Comput. 14 (3) (2004) 199–222.
[26] J. Ma, J. Theiler, S. Perkins, Accurate on-line support vector regression, Neural Comput. 15 (11) (2003) 2683–2703.
[27] CC2541, http://www.ti.com/lit/ds/symlink/cc2541.pdf.
[28] BLE-STACK, http://www.ti.com/tool/ble-stack.
[29] H. Ma, L. Lampe, S. Hranilovic, Integration of indoor visible light and power line communication systems, in: Proc. of 17th IEEE ISPLC, 2013, pp. 291–296.
[30] Y. Yang, S. Ding, J. Hao, J. Luo, Roaming in connecting light: Practical visible light communication leveraging LED sensing, in: Proc. of the 4th ACM VLCS, 2017, pp. 9–14.
[31] Y. Yang, J. Hao, J. Luo, S.J. Pan, CeilingSee: Device-free occupancy inference through lighting infrastructure based LED sensing, in: Proc. of the 15th IEEE PerCom, 2017, pp. 247–256.
[32] L. Audin, Occupancy sensors: Promise and pitfalls, Tech Update, Tu-93-8, E Source, Inc., Boulder, CO, 1999.
[33] D. Yang, H. Gonzalez-Banos, L. Guibas, Counting people in crowds with a real-time network of simple image sensors, in: Proc. of the 9th IEEE ICCV, 2003, pp. 122–129.
[34] S. Rajagopal, R.D. Roberts, S.-K. Lim, IEEE 802.15. 7 visible light communication: Modulation schemes and dimming support, IEEE Commun. Mag. 50 (3) (2012) 72–82.
[35] D. Tsonev, H. Chun, S. Rajbhandari, J. McKendry, S. Videv, E. Gu, M. Haji, S. Watson, A. Kelly, G. Faulkner, M. Dawson, H. Haas, D. O'Brien, A 3-Gb/s single-LED OFDM-based wireless VLC link using a gallium nitride μLED, IEEE Photonics Technol. Lett. 26 (7) (2014) 637–640.
[36] W. Hu, H. Gu, Q. Pu, LightSync: Unsynchronized visual communication over screen-camera links, in: Proc. of 19th ACM MobiCom, 2013, pp. 15–26.
[37] A. Wang, S. Ma, C. Hu, J. Huai, C. Peng, G. Shen, Enhancing reliability to boost the throughput over screen-camera links, in: Proc. of 20th ACM MobiCom, 2014, pp. 41–52.
[38] W. Hu, J. Mao, Z. Huang, Y. Xue, J. She, K. Bian, G. Shen, Strata: Layered coding for scalable visual communication, in: Proc. of the 20th ACM MobiCom, 2014, pp. 79–90.
[39] J. Hao, Y. Yang, J. Luo, CeilingCast: Energy efficient and location-bound broadcast through led-camera communication, in: Proc. of the 35th IEEE INFOCOM, 2016, pp. 1–9.
[40] Y. Yang, J. Nie, J. Luo, ReflexCode: Coding with superposed reflection light for LED-camera communication, in: Proc. of the 23rd ACM MobiCom, 2017, pp. 193–205.
[41] X. Zhou, A.T. Campbell, Visible light networking and sensing, in: Proc. of the 1st ACM HotWireless, 2014, pp. 55–60.
[42] C. An, T. Li, Z. Tian, A.T. Campbell, X. Zhou, Visible light knows who you are, in: Proc. of the 2nd ACM VLCS Workshop, 2015, pp. 39–44.