Gesture Recognition Using Ambient Light

RAGHAV H. VENKATNARAYAN, North Carolina State University, USA, rhampap@ncsu.edu
MUHAMMAD SHAHZAD, North Carolina State University, USA, mshahza@ncsu.edu

There is a growing interest in the scientific community to develop techniques for humans to communicate with the computing that is embedding into our environments. Researchers are already exploring ubiquitous modalities, such as radio frequency signals, to develop gesture recognition systems. In this paper, we explore another such modality, namely ambient light, and develop LiGest, an ambient light based gesture recognition system. The key property of LiGest is that it is agnostic to lighting conditions, position and orientation of user, and who performs the gestures. The general idea behind LiGest is that when a user performs different gestures, the shadows of the user move in unique patterns. LiGest first learns these patterns using training samples and then recognizes unknown samples by matching them with the learnt patterns. To capture these patterns, LiGest uses a grid of light sensors deployed on floor. While the general idea behind LiGest seems straightforward, it is actually very challenging to put it into practice because the intensity, size, and number of shadows of a user are not fixed and depend highly on the position and orientation of a user as well as on the intensity, position, and number of light sources. We developed a prototype of LiGest using commercially available light sensors and extensively evaluated it with the help of 20 volunteers. Our results show that LiGest achieves an *average* accuracy of 96.36% across all volunteers.

CCS Concepts: • **Human-centered computing → Gestural input**;

Additional Key Words and Phrases: Gesture recognition, Ambient light

ACM Reference Format:
Raghav H. Venkatnarayan and Muhammad Shahzad. 2018. Gesture Recognition Using Ambient Light. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 1, Article 40 (March 2018), 28 pages. https://doi.org/10.1145/3191772

1 INTRODUCTION

1.1 Motivation

With the advent of the Internet of Things, we are witnessing the infusion of computing into our everyday environments in various forms such as intelligent thermostats [9, 12, 13, 17], smart appliances [16, 21], and remotely controllable household equipment such as smart lights [14, 18, 19, 22]. Consequently, we need new ways to seamlessly communicate and interact with such always-available computing and always-connected devices. A natural choice for such communication and interaction is human gestures because gestures are an integral part of the way humans communicate and interact with each other in their daily lives. Indeed, there has been a rising trend in the integration of gesture recognition systems into various consumer electronics including gaming consoles, smart phones, and navigation devices. Xbox Kinect [23] and Leap [15] are the leading examples of commercially available gesture recognition systems. Gesture recognition systems further hold promise as a supplemental interface to voice-assistants, such as Amazon Echo, which remain out of access to the speech-impaired.

Authors' addresses: Raghav H. Venkatnarayan, North Carolina State University, 890 Oval Drive, Raleigh, NC, 27606, USA, rhampap@ncsu.edu; Muhammad Shahzad, North Carolina State University, 890 Oval Drive, Raleigh, NC, 27606, USA, mshahza@ncsu.edu.

2474-9567/2018/3-ART40 $15.00
https://doi.org/10.1145/3191772

Researchers are also exploring radio frequency (RF) based modalities, such as WiFi and RFID, to recognize gestures by monitoring changes in the RF signals due to human movements [24, 47, 50, 52, 78, 91, 92, 102–104, 106, 109, 113]. While the work on RF based gesture recognition holds promise, there are several other modalities, such as ambient light, that can enable gesture recognition (*e.g.*, by monitoring changes in the intensity of light due to human movements) and deserve exploration, but have not yet received enough attention from the research community. Each modality also has its own unique set of limitations and cannot enable gesture recognition in every practical scenario. Therefore, we envision that a truly practical gesture recognition system can be developed only by integrating multiple gesture recognition systems that complement each other and each of which is based on a unique modality such as ambient light, WiFi, ultrasonics, or backscatter.

1.2 Problem Statement

As a step towards realizing this vision, in this paper, our objective is to design, implement, and extensively evaluate an ambient light based gesture recognition system that has the following three properties:

1) Lighting Condition Agnostic: the gesture recognition system should not be affected by changes in the intensity, number, and/or position of light sources in the environment. The system should also not assume any control over the lighting infrastructure and should work with all types of ambient light sources.

2) Position and Orientation Agnostic: the system should recognize gestures of user irrespective of the location of the user in the given indoor environment and the direction the user is facing in, *i.e.*, user's orientation.

3) User Agnostic: the system should recognize gestures of any arbitrary user in the given indoor environment and not only of a specific set of users.

1.3 Relationship with the State of the Art

Two published schemes most related to our work are LiSense [58] and StarLight [60]. LiSense reconstructs a stationary user's posture in the form of a stick figure using a set of photodiodes deployed on the floor and a set of LEDs on the ceiling, which transmit encoded information to the photodiodes. StarLight uses a similar platform as LiSense and extends LiSense from stationary users to the users that move around and continuously reconstructs the posture of such mobile users. LiSense and StarLight are promising first steps towards using ambient light for human sensing. However, our work is different from these two schemes in two important ways.

The first difference is in the type of light sources: LiSense and StarLight both require the light sources to be only LEDs because they require the light sources on the ceiling to transmit encoded information to photodiodes on the floor. Our work, on the other hand, aims at doing gesture recognition *using ambient light* generated by any type of light sources, such as LEDs, fluorescent lights, and even sunlight, and *without requiring any control over the light sources*. The motivation behind this aim is twofold. First, while the adoption of LED based lights is steadily increasing, they are not yet ubiquitously deployed and constitute only 3% of all installed light sources [8]. Therefore, to enable light based gesture recognition in all buildings, it is imperative to enable it using ambient light generated by any type of light sources. Second, control over light sources may not always be possible, especially in existing buildings, as that requires a significant amount of prohibitively costly electrical rewiring.

The second difference is in objective: the primary objective of LiSense and StarLight is to reconstruct the posture of a user in the form of a stick figure, whereas the primary objective of our work is to recognize gestures of a user. While one can extend LiSense and StarLight to enable gesture recognition by applying appropriate image processing and machine learning techniques that track the limbs in the stick figure, this is not trivial. Furthermore, LiSense and StarLight also require body measurements of the users, such as height and size of body parts, which may not always be available apriori. We show in this paper that one can do gesture recognition without going through the process of posture reconstruction at all.

1.4 Proposed Approach

In this paper, we propose LiGest, an ambient light based gesture recognition system that has all of the three properties mentioned in Section 1.2. Similar to LiSense [58] and StarLight [60], LiGest uses a grid of light sensors deployed on the floor, but unlike these schemes, LiGest neither requires the light sources to be only LEDs nor requires any control over them. Note that LiGest's (and also LiSense's and StarLight's) approach of installing a grid of sensors on the floor is neither cost prohibitive nor impractical; in fact, floors with embedded sensors, albeit not light sensors, are already commercially available [10, 11]. The commercial availability of such floors establishes that embedding light sensors into floors is practically feasible.

The principle behind LiGest is that as a user performs a gesture in an indoor environment that is lit with light, the shadows of the user falling on the sensors on the floor move, resulting in changes to the intensity of light arriving at different sensors. As different gestures result in different motions of the body, the patterns of change in the intensity of light are different for different gestures. LiGest leverages this principle to first learn these patterns for each predefined gesture using machine learning based classification techniques and then recognizes them as the user performs the gestures at runtime.

To build classification models for a given set of gestures, for each gesture, LiGest first acquires training samples. As LiGest is agnostic to lighting conditions, position and orientation of user, and the user itself, it requires the training samples to be collected under only one lighting condition, at only one position and orientation, and from only one user. Each gesture sample essentially consists of N time-series of sensor values, where N represents the number of light sensors deployed on the floor. We call each time-series an S-stream. Next, LiGest applies several filtering and transformation operations on these training samples, which make LiGest agnostic to lighting conditions, position and orientation of user, and the user itself. These operations include denoising, signal standardization, wavelet transformation, rasterization, and principal component analysis. We will describe these operations in detail later in the paper. Finally, LiGest extracts features from the filtered and transformed training samples and generates support vector machine based classification models for the given set of gestures.

To recognize gestures of a user at runtime, as soon as the user performs a gesture, LiGest first automatically detects that a gesture has been performed and then evaluates the gesture against the classification models. Finally, note that, just like several existing RF based gesture recognition systems [52, 78, 103], LiGest currently recognizes macro gestures that involve hand, limb, or body movements, such as clap, jump, step, *etc.* It does not yet recognize micro gestures that involve small finger movements, such as typing.

1.5 Technical Challenges

In designing LiGest, we faced several technical challenges, out of which, we describe four here.

Change in the Light Intensity: The first technical challenge is to handle change in the intensity of ambient light in the given indoor environment, which can happen due to reasons such as dimming a light. Change in the intensity of ambient light alters the magnitude of the patterns of change in the S-streams, which can render the classification models unable to recognize gestures. To address this challenge, for each S-stream, LiGest first calculates its first order difference, which we call a dS-stream, and then executes a standardization operation in which it scales the dS-stream in proportion to the standard deviation in its values using Z-score transform [88]. The magnitudes of the patterns of change in the resulting standardized dS-streams are approximately the same for different samples of a given gesture, irrespective of the intensity of the ambient light in which those samples were collected. This standardization operation makes LiGest agnostic to the intensity of light as long as there is enough light ($\sim$ 100 Lux) in the room.

Change in the Position of Light Sources: The second technical challenge is to handle the change in the position of light sources in the given indoor environment, which can happen due to reasons such as moving the light source or turning off one light and turning on another. Change in the position of a light source causes the length and/or direction of a shadow to change, which in turn causes the shadow to be incident on a different set

of light sensors. To address this challenge, LiGest executes a rasterization operation in which it essentially maps the values in the standardized *d*S-streams on to pixels in a raster such that the location of the pixel is determined by the magnitude of each value. Although change in the position of a light source changes the set of sensors on which the shadow falls, the patterns of change in the intensity of light stay the same. Consequently, the data from this new set of sensors maps to the same pixels on the raster as the data from the previous set of sensors, and the resulting raster stays the same. This rasterization operation makes LiGest agnostic to the position of light sources.

Change in the Number of Light Sources: The third technical challenge is to handle the change in the number of light sources in the given environment, which can happen due to reasons such as turning on or off a light source. Change in the number of light sources changes the number of shadows. To distinguish between the shadows caused by different light sources, LiSense [58] and StarLight [60] modulate each LED light source at a different frequency, and determine the amount of light arriving from each light source at any given photodiode by applying fourier transform on the values reported by that sensor. If a user's presence blocks the light of any particular source from reaching the photodiode, its corresponding frequency component diminishes, and LiSense and StarLight can determine that the shadow due to that light source is falling on this photodiode.

Unfortunately, such a shadow-separation approach is not feasible in our setting due to two reasons: 1) we do not have any control over light sources; 2) most ambient light sources, such as fluorescent lights, cannot be modulated. To address this challenge, we leverage the observation that while each light source creates a unique shadow that falls on a unique set of sensors, the patterns of change seen by each set of sensors stay the same because these patterns result from the same gesture. Thus, if we map the values in the standardized *d*S-streams from each sensor on to a raster such that the mapping depends on the magnitudes of the values, the values from each set of sensors map to almost the same set of pixels on the raster. Consequently, the resulting raster always stays the same irrespective of the number of the light source. In this way, the rasterization operation makes LiGest independent of the number of shadows, and thus, agnostic to the number of light sources.

Change in the Position/Orientation of User: The fourth technical challenge is to enable LiGest to recognize gestures irrespective of the position and orientation of the user. While shadows can drastically change with change in the position and/or orientation of the user with respect to the light sources, their net effect is similar to change in the position of light sources, *i.e.*, a change in position and/or orientation of user causes each shadow to fall on a different set of light sensors but the patterns of change in the intensity of light seen by the new set of sensors stay the same. The rasterization operation, therefore, handles the effects of changes in user position and orientation as well, and makes LiGest agnostic to position and orientation of the user.

1.6 Key Contributions

In this paper, we make following four key contributions.

(1) We have proposed a gesture recognition system that works using ambient light generated from any type of light sources, including LEDs and fluorescent lamps, and without requiring any control over the light sources.
(2) We have developed techniques that make ambient light based gesture recognition systems agnostic to lighting conditions as well as to the position and orientation of user.
(3) We have collected a comprehensive real world dataset of 15175 samples from 20 users in 9 positions, 4 orientations, 11 lighting conditions, and 2 different environments. The dataset will be made available to the research community after publication.
(4) We have implemented and extensively evaluated LiGest using commodity light sensors. Our results from an extensive evaluation on our dataset show that LiGest achieves an *average* accuracy of 96.36%, which is similar to (and often higher than) the average accuracies achieved by RF based gesture recognition systems [24, 52, 78].

2 RELATED WORK

While a significant amount of work has been done on light based communication [28, 29, 33, 42, 45, 46, 54, 55, 57, 59, 62, 63, 74, 79, 83, 84, 101, 112], light based gesture recognition has received little attention. Next, we first describe existing light based human sensing schemes and then present other gesture recognition systems that use cameras, wearable sensors, and RF signals.

Light based Human Sensing: The work closest to ours is LiSense [58] and StarLight [60], which reconstruct user's posture in the form of stick figures using visible light, as discussed in detail in Section 1.3. CeilingSee [108] proposed to estimate the number of occupants in a room using ceiling-mounted LEDs as sensors. The fundamental difference between CeilingSee and LiGest is that they are designed to perform two very different tasks: CeilingSee estimates the number of occupants, while LiGest recognizes the gestures of occupants. A recent (but *not* peer-reviewed) technical report proposed GestureLite [51], which leverages ambient light to recognize hand gestures in "an environment with a reasonably consistent lighting scheme". It uses a small 25in^2 sensing platform comprising a grid of 9 photodiodes to largely classify wrist and hand movements of a user based on the shadows. The objective of GestureLite by design is very different from that of LiGest. GestureLite was proposed to recognize hand gestures when the hand is close to (< 11 inches from) a densely packed grid of photodiodes, while LiGest has been designed to recognize whole body based gestures at much farther distance (in the order of feet) from the photodiodes. By requiring the user's hand to be this close to the sensing grid, GestureLite achieves a fundamental advantage of not having to deal with multiple shadows because at distances as small as 11 inches from the sensing grid, ambient light rays from multiple sources largely converge, and thus the shadow patterns of a body part blocking the light within such distances are almost equivalent to the shadow patterns observed if the body part were to block a single light source, as demonstrated by Segen and Kumar in [86]. At larger distances for which LiGest has been designed for, multiple shadows are unavoidable due to the multiplicity of light sources in indoor environments, and thus, the methods proposed in GestureLite cannot be used. If we set aside the fundamental difference in the objectives of GestureLite and LiGest, and try to use GestureLite to recognize whole body based gestures, GestureLite suffers from four other important limitations. First, GestureLite assumes that the lighting condition stays fixed, *i.e.*, the position, number, and intensity of ambient light sources never changes. The authors acknowledge that even when classifying hand gestures at distances of less than 11in from the sensing grid, if lighting conditions change, the accuracy of GestureLite reduces to 74%. In comparison, LiGest handles changes in the lighting conditions very well. Second, GestureLite requires the user orientation to be always the same and the gestures to be always performed at similar speeds. In contrast, LiGest does not impose any such restrictions, and handles changes in user position, orientation, and gesture speed very well. Third, as acknowledged by the authors, GestureLite works with the reported accuracy only if the training and testing is performed on the same user. LiGest, on the other hand, can handle variations across users. Last, GestureLite uses over an order of magnitude greater density of photodiodes compared to LiGest: GestureLite uses one photodiode per 2.7in^2 while LiGest uses one photodiode per 36in^2.

Camera based Gesture Recognition: Camera and depth sensor based systems capture video frames and apply computer vision algorithms to track human limbs to recognize gestures [27, 31, 39, 43, 44, 48, 61, 68, 72, 76, 81, 90, 96, 99, 111]. Such systems often require exhaustive training to achieve good accuracy. For example, the tracking algorithm used in Kinect relies on hundreds of thousands of training images to build the classifiers [90]. Camera based systems have two major drawbacks. First, capturing and storing the raw camera data raises privacy concerns [82, 105]. Second, it requires users to be in the field of vision of camera, which limits its pervasiveness.

Wearable Sensors based Gesture Recognition: Wearable sensors based gesture recognition systems require the user to wear one or more sensors (such as accelerometers) on one or more limbs. These systems use the values reported by these sensors to recognize gestures [30, 35, 56, 69, 71, 73, 94, 95, 110]. With the increase in the adoption of smart phones, researchers have proposed schemes that utilize the motion sensors in smart phones to

recognize gestures [64–66, 77, 80, 107]. As the sensors are positioned at the site of motion in wearable sensors based gesture recognition systems, these systems usually achieve high accuracy. Unfortunately, wearable sensors based approaches are inconvenient because the users have to wear sensors continuously or hold a smart phone in their hand while performing gestures. Both camera and wearable sensor based systems are intrusive in the sense that they either do not preserve user privacy or require users to wear sensors.

Sound based Gesture Recognition: Sound based sensing and gesture recognition systems leverage changes in the arrival times of reflections of sound waves or changes in the Doppler shift in the reflected sound waves caused by the changes in human posture to recognize gestures [40, 75, 98]. While sound is a promising modality for use in gesture recognition, sound based gesture recognition has three limitations. First, sound based methods usually work when the user is at a short distance from the transmitter and receiver. Second, while they can use inaudible high-frequency tones that the ears of adults do not detect, these tones can be disturbing for pets and young children. Third, the shifted frequency depends upon the speed of sound in air, which in turn depends on the temperature, and thus making such systems quite sensitive to the environment. Nonetheless, as mentioned above, sound is a promising modality and researchers are actively working in this area to find solutions to such limitations. We envision that in future, it will be used in conjunction with other modalities, such as light, and help make gesture recognition systems more robust.

RF based Gesture Recognition: Recently, several gesture recognition systems have been proposed that use RF signals to recognize gestures. RF channels are characterized by received signal strength (RSS) and channel state indicator (CSI). When an object, such as a human arm, moves in an environment, the characteristics of the radio channels in that environment change, resulting in changes in the RSS and CSI values. Most RF based gesture recognition systems utilize the patterns of these changes in RSS values [92, 93] and CSI values [41, 87, 100, 102–104, 106, 109, 113] due to human motion to recognize human gestures and activities. The remaining RF based gesture recognition systems utilize software defined radios and measure other fine-grained signal characteristics such as angle of arrival and doppler shifts [25, 26, 78] or utilize special hardware [52]. While RF is a promising modality and we expect to see significant advancements in this direction, it has its own unique set of limitations, most important of which is its sensitivity to channel noise and environmental changes. The key advantage of RF and other similar modalities, such as ambient light and ultrasonics, is that they are non-intrusive, *i.e.*, they neither capture user's video nor require the user to wear any sensors.

3 SENSING PLATFORM

To collect gesture samples from our volunteers, we developed a sensing platform that comprised of a portable and sturdy 6ft × 6ft mat and $N = 36$ light sensors placed on it in the form of a square grid with an even spacing of 1ft between adjacent sensors. Each light sensor measures changes in the intensity of light due to the moving shadows as a user performs gestures, and reports the values to a central server. To keep the cost of our sensing platform low, we used cheap off-the-shelf light sensors, namely TSL237 [20]. TSL237 is a light-to-frequency converter, *i.e.*, it converts light intensity into a series of square-wave pulses: the lower the light intensity, the slower the pulses.

To connect the sensors to the central server, we interfaced the 36 sensors to 18 Arduino Uno boards, *i.e.*, 2 sensors per board. Figure 1 shows our sensing platform. Figure 2 shows the schematic diagram of the connection of two TSL237 sensors with one Arduino board. Each board acquires the light intensity values at a rate of 100 samples/sec from each of its two attached sensors and sends them to the central server via its USB port. While we can increase the sampling rate beyond 100 samples/sec, we found that this rate results in acceptable accuracy. We connected the 18 Arduino boards to the central server through a USB hub. The server runs a multi-threaded data acquisition program in Java to receive the light-intensity values sent by the boards.

To collect gesture samples, we placed the mat on floor in different environments and under various lighting conditions and asked our volunteers to stand on it at different positions and in different orientations and perform gestures. We will provide more details on the data collection process in Section 8.1. Note that while our sensing

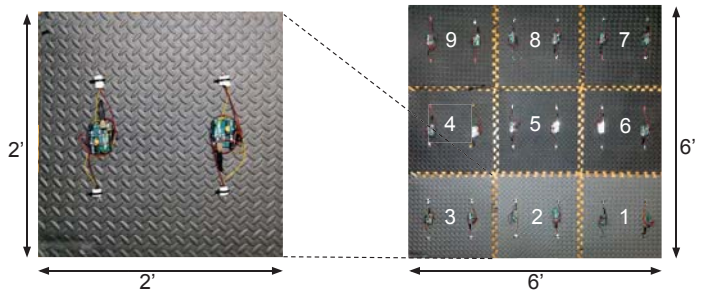


Fig. 1. Prototype sensing platform

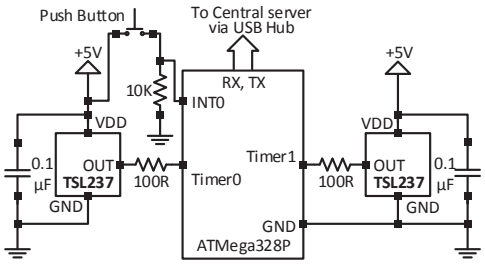


Fig. 2. Schematics for connecting a pair of sensors to the server

platform is 6ft × 6ft in size, we envision that when deployed as a production system, the platform will cover the entire floor so that the gestures of users are recognized everywhere on the floor. The motivation behind placing the mat on the floor is that the light sources are usually mounted higher than the typical heights of users, due to which the shadow(s) of the users fall primarily on the floor. Next, we describe the cost of our sensing platform and the anticipated cost when manufactured as a production system. The TSL237 sensors and the Arduino Uno boards that we have are priced at USD 2.7 per sensor [1] and USD 14.9 per Arduino [2]), respectively, at the time of writing this paper. The net cost of building our experimental sensing platform was approximately $36 \times 2.7 + 18 \times 14.9 \approx 365$ US dollars, which translates to about USD 10 per ft^2. When developed as a production system in large quantities, the components and micro-controllers are available at much lower prices. For example, very high quality photodiodes and powerful enough micro-controllers are available for purchase at just USD 0.1 [3] and 2.4 per piece [4], respectively. The sewable conducting threads required to make connections cost just USD 0.04/ft[5]. At these rates, the materials cost of installing the sensing platform in a 100ft^2 room is roughly USD 20 (100 photodiodes and four 25-input micro-controllers), which translates to just 20 cents per ft^2. Fortunately, this cost is very little when compared to the materials cost of, for example, carpeting, which starts at USD 1 per ft^2 [6], and averages above USD 3 per ft^2 [7]. As another example, consider a 2600 ft^2 house in which gesture recognition is desired in 80% of the area. The cost of deploying our system in this home would be $2600 \times 0.8 \times 0.2 =$ USD 416. Compared to the average cost of homes in the US, which stands at USD 188,900 at the time of publication of this paper, the cost of deploying our system is just 0.22%.

4 OVERVIEW

In this section, we provide an overview of LiGest. To recognize a given set of gestures in a given indoor environment, LiGest needs classification models for those gestures in that environment. For this, LiGest first acquires training samples of the gestures in that environment from at least one user in at least one position and orientation under at least one lighting condition. It then builds the classification models for the given set of gestures in the following three steps.

4.1 Preprocessing

In this step, LiGest takes the N raw S-streams from all N sensors and preprocesses them to make them usable for generating good classification models. During this step, LiGest performs following three operations.

1) Denoising: The S-streams can contain two types of noise: stray shadow noise, which occurs due to the shadows cast by the static objects in the environment, and hardware noise, which occurs due to the 60Hz flicker of fluorescent lights and minor imperfections in sensor and Arduino's hardware. LiGest removes the stray shadow noise from a given S-stream by taking its first order difference. We name the resulting stream dS-stream. Next, it removes the hardware and environment noise from the dS-stream by performing hard wavelet thresholding on it using DWT. We name the resulting stream denoised dS-stream.

2) Gesture Detection: After denoising, LiGest automatically detects the start and end times of gestures from the denoised *d*S-streams by using a supervised thresholding scheme and uses the values in the denoised *d*S-streams contained between the start and end times of each gesture sample for further processing.
3) Standardization: After detecting gestures, LiGest performs standardization operation on the denoised *d*S-streams of each sample during which it applies Z-score transform temporally on the denoised *d*S-streams. We name the resulting streams standardized *d*S-streams. The objective of this operation is to make LiGest independent of changes in the intensity of light sources.

4.2 Position, Orientation, Lighting, and User Agnostic Feature Extraction

To generate classification models for the gestures, LiGest needs features that satisfy following three requirements: 1) the features should have high classification potential; 2) the features should be agnostic to the position and number of light sources as well as to the position and orientation of the user; 3) the number of features should be minimal to keep the computational complexity low. Keeping these requirements in view, LiGest extracts features by applying the following three operations on standardized *d*S-streams of each gesture.
1) Wavelet Transformation: While the values in the standardized *d*S-streams can be directly used as features, the resulting number of features would be very large. To reduce the number of features, LiGest applies stationary wavelet transform on the standardized *d*S-streams and calculates detail-coefficients from a level that produces minimum number of coefficients without losing any useful information. LiGest uses these detail-coefficients for further processing.
2) Rasterization: On obtaining the detail-coefficients, LiGest applies a transformation operation on them, which we call rasterization. The objective of rasterization is to make LiGest independent of the number and position of light sources as well as the position and orientation of the user. LiGest performs rasterization by mapping the values of the coefficients obtained from the wavelet transformation onto a two dimensional binary raster image, such that the mapping of each coefficient depends on the value of the coefficient. It then performs a JPEG2000 inspired DWT based compression on the resulting raster to reduce the size of the raster further.
3) Principal Component Analysis: While the values of the binary pixels in the compressed raster can be directly used as features, the compressed raster still contains some redundant information. To remove the features containing redundant information, we apply principal component analysis (PCA) on this compressed raster and get an even smaller set of features that has a very high classification potential. LiGest uses this set of features to generate classification models.

4.3 Classifier Training

After extracting values of features from all training samples of a given set of gestures, LiGest builds support vector machine (SVM) based classification models for that set of gestures. LiGest uses these classification models to recognize gestures of a user at runtime.

5 PREPROCESSING

Next, we describe the three operations of denoising, gesture detection, and standardization, which LiGest uses to prepare the S-streams for feature extraction.

5.1 Denoising

The time-series of each sensor, *i.e.*, each S-stream, contains noise that must be removed to achieve a high gesture recognition accuracy. The S-streams can contain two types of noises: 1) stray shadow noise, and 2) hardware noise. Next, we describe each of these two types of noises and show how LiGest removes them.

5.1.1 Stray Shadow Noise. Static objects, such as a chair, can cast shadows on the light sensors. We call such shadows stray shadows. If one or more shadows of the user overlap with a stray shadow, the net light intensity measured by the sensors under the stray shadow will have a constant offset compared to the light intensity measured by those sensors in the absence of the stray shadow. Such offsets can potentially deteriorate LiGest's accuracy. To remove the effect of stray shadows from the S-streams, LiGest takes the first order difference of each S-stream. We call the resulting stream a *d*S-stream, which does not contain any offsets due to stray shadows. LiGest uses the *d*S-streams for further processing.

5.1.2 Hardware Noise. This type of noise can be further categorized into two subtypes: 1) time-localized, and 2) frequency localized. The time-localized noise appears in the form of spikes in the *d*S-streams, which occur due to the minor imperfections in Arduino's oscillator hardware. More specifically, due to the 100 samples/sec sampling rate, every 10ms, each Arduino board resets its counters through an interrupt. The interrupt sometimes gets delayed due to these imperfections and the counters are not reset in time. Consequently, Arduino board sends inflated sensor values to the central server, which appear as spikes in the *d*S-streams. The frequency-localized noise appears due to the flicker of lights powered by the 60Hz AC. The top figure in Figure 3 plots the magnitude of the fourier transform of the *d*S-streams from all N sensors. The figure does not plot the magnitudes of frequencies < 5Hz due to their large amplitudes. We observe a peak at 40Hz in this figure for all *d*S-streams (a 60Hz signal shows a peak at 40Hz when sampled at 100Hz due to spectral folding resulting from sampling rate less than Nyquist criteria). We also observe that apart from the 40Hz frequency, the *d*S-streams contain other frequencies, 20Hz and higher, with notable amplitudes. Though the reason behind the presence of these frequencies is unclear, they need to be removed from the *d*S-streams to clean them for further processing.

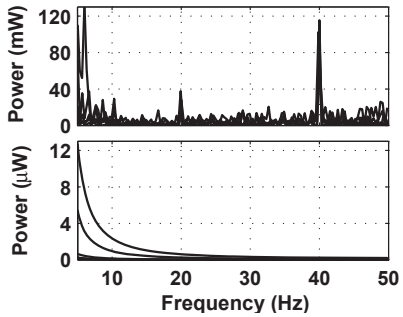


Fig. 3. Power spectrum of the 36 sensors

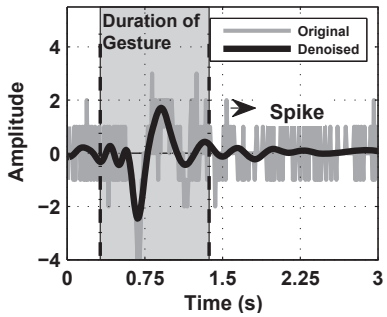


Fig. 4. Original and denoised *d*S-streams

As *d*S-streams contain both time-localized and frequency-localized noise, a natural choice is to use discrete wavelet transform (DWT) to remove such noise because DWT provides good resolution in both time and frequency. DWT can be computed efficiently using Mallat's Algorithm [89] when the length of the signal is an exact power of 2. We observed from our dataset that volunteers always took less than 5 seconds to perform any gesture. Consequently, we take chunks of consecutive 512 values from *d*S-stream and process each chunk individually to remove hardware noise from it.

DWT is a hierarchical transform with multiple levels, where each level processes a localized section of the signal and the frequency span at any given level is half of the frequency span at the level before it. At each level, DWT gives detail-coefficients, which correspond to the high frequencies in the signal, and approximation-coefficients, which correspond to the low frequencies in the signal. As seen in Figure 3, the variations introduced by noise in the *d*S-streams have higher frequencies and lower amplitudes compared to the variations introduced by gestures. Consequently, the noise can be removed from the *d*S-streams by setting detail-coefficients with values below a certain threshold to 0.

Based on this intuition, LiGest removes the hardware noise from each *d*S-stream in the following four steps. First, it applies DWT on the given *d*S-stream to compute level 4 detail-coefficients. LiGest does not calculate coefficients beyond level 4 because the length of each chunk is 512 and the highest frequency component in the *d*S-stream is limited to 50Hz due to our sampling rate of 100Hz. As the frequency span halves every DWT level, level 4 represents coefficients in frequency range $[0, 512/2^{4-1}]$ Hz, which completely accommodates the frequency range of [0, 50] Hz of our *d*S-streams. Any level greater than 4 does not completely accommodate all frequencies, whereas any level below level 4 just produces larger number of coefficients, which increases the computational complexity when training the classifiers.

Second, LiGest calculates the threshold below which it should set the detail-coefficients of level 4 to 0. To calculate this threshold, LiGest uses the method proposed by Donoho *et al.* [34], which is based on Stein's unbiased risk estimate (SURE) [38]. Third, LiGest compares the detail-coefficients with the threshold and sets all detail-coefficients with values less than the threshold to 0. This step is known as hard wavelet thresholding [49]. Last, LiGest applies inverse DWT on the resulting detail-coefficients along with the unmodified approximation-coefficients to generate the denoised chunk of 512 values. LiGest concatenates all denoised chunks to obtain the denoised *d*S-stream. The bottom figure in Figure 3 plots the magnitude of the fourier transform of the denoised *d*S-streams from all N sensors. We clearly observe that the denoised *d*S-streams do not contain high frequency noise anymore. Figure 4 plots a sample of an original and its corresponding denoised *d*S-stream. We clearly see in this figure that the denoised *d*S-stream is very smooth and contains no abrupt spikes.

5.2 Gesture Detection

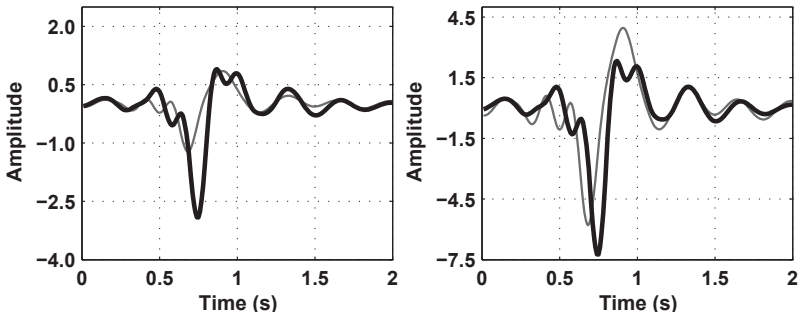
To detect the start and end times of a gesture, LiGest uses a simple thresholding scheme. When a user is not performing a gesture, theoretically, all values in the denoised *d*S-streams should be equal to 0. Practically, we observe that the values are equal to few tenths of a decimal, which is indeed close to 0. When a user performs a gesture, the values in the denoised *d*S-streams are much larger than 0, as can also be seen in Figure 4. Thus, by comparing the absolute values in the denoised *d*S-streams to a threshold slightly greater than 0, LiGest detects the start of a gesture. Next, it detects the end of the gesture as soon as it sees that 50 consecutive values in each denoised *d*S-stream are less than the threshold. LiGest checks 50 consecutive values before declaring the end of gesture to ascertain that the user has indeed stopped. To set the threshold, LiGest observes all *d*S-streams over a period of time when the user is not performing any gesture and sets the threshold to a value that is 10% larger than the largest absolute value observed among all *d*S-streams during that period of time. Using this approach, LiGest was successfully able to detect the start and end times of 99% samples that we collected from our volunteers.

After identifying the start and end times of the gesture, if the number of values in each denoised *d*S-stream during the start and end times is less than 512, LiGest appends 0s at the end of each denoised *d*S-stream to make the length equal to 512. The reason for appending the 0s is twofold: 1) make the length of every gesture sample equal so that our feature extraction module can extract equal number of features from every sample, and 2) make the length of every gesture sample an exact power 2 so that our feature extraction module can compute the DWT. The reason for choosing the length of gesture samples to be 512 is the same as given earlier: the duration of every gesture in our dataset is always less than 5s and 512 is the nearest exact power of 2. One can choose a larger or smaller value for the length of gesture samples based on the duration of the gestures one intends to perform. Onward, we will use L to represent the length of the gesture samples, where L can be any number that is an exact power of 2.

5.3 Standardization

Due to differences in the intensities of different light sources or due to differences in the distances of a user from different light sources, one shadow of the user falling on one sensor may be darker compared to another falling

on another sensor. Consequently, one sensor sees larger changes in the intensity of light compared to the other as the user performs a gesture. Similarly, due to changes in the intensities of light sources or the distance of user from the light sources across samples, the darkness of a shadow may vary across samples. Consequently, a sensor may see different magnitudes of changes in the intensity of light across different samples of the same gesture. All these observations deteriorate LiGest's accuracy. Figure 5(a) plots two denoised *d*S-streams from a sensor for two samples of a gesture collected at two different distances from a light source. The figure shows that while the shapes of the two denoised *d*S-streams are similar, their magnitudes are not.



(a) Denoised *d*S-streams (b) Standardized *d*S-streams

Fig. 5. Effect of standardization operation

Our approach towards solving this problem is to apply a suitable transform on the denoised *d*S-streams such that the resulting streams have similar magnitudes. It is straightforward to visualize that the values in the denoised *d*S-stream of a sensor that sees a darker shadow have a larger standard deviation compared to the values in the denoised *d*S-stream of a sensor that sees a lighter shadow. Thus, if we scale the values in each denoised *d*S-stream by the deviation across all streams, the resulting streams will have similar magnitudes.

Z-score transform is a well known transform that expresses data in terms of its deviation [88]. As soon as LiGest detects a gesture using the method described in Section 5.2, it applies the Z-score transform temporally on the N denoised *d*S-streams of that gesture sample. We call the resulting set of N streams standardized *d*S-streams. To facilitate the formal treatment of gesture samples in the rest of the paper, we represent each gesture sample by a matrix $\mathbf{G}$, where $\mathbf{G}_{i,j}$ is the i^{th} value in the j^{th} standardized *d*S-stream and $i \in [1, L]$, $j \in [1, N]$. Figure 5(b) plots two standardized *d*S-streams corresponding to the two denoised *d*S-streams in Figure 5(a). We see in this figure that the standardized *d*S-streams indeed have similar magnitudes. Finally, note that this standardization operation makes LiGest agnostic to changes in the intensity of light sources.

6 POSITION, ORIENTATION, LIGHTING, AND USER AGNOSTIC FEATURE EXTRACTION

While the values in a gesture matrix $\mathbf{G}$ can be directly used as features to generate classification models, the resulting number of features would be very large. For example, for our sensing platform $L = 512$ and $N = 36$, which results in $512 \times 36 = 18432$ features per sample. Unfortunately, such a large number of features not only increases the computational complexity of training classifiers but also decreases the accuracy due to the curse of dimensionality [53]. Thus, our objective is to develop a feature extraction scheme that reduces the number of features per sample without losing any useful information in the standardized *d*S-streams. For this, LiGest applies the following three operations on the gesture matrix $\mathbf{G}$: wavelet transformation, rasterization, and PCA. Next, we describe these three operations in detail.

6.1 Wavelet Transformation

Recall from Section 5.1.2 that DWT calculates approximation-coefficients and detail-coefficients for any given signal. These coefficients collectively characterize the shape of the signal and can, therefore, also be used as features. While the number of coefficients that DWT calculates at the first level is equal to the length of the signal, the number of coefficients halve as we go from one DWT level to next, *i.e.*, at level l, the number of coefficients reduce by a factor of 2^{l-1} compared to the number of coefficients at level 1. If one can justify to use coefficients from a level $l > 1$, one essentially reduces the number of features per sample.

The choice of using a level $l > 1$ for a given signal is justified only when the level l completely accommodates all frequencies present in the signal. Let $F_{\max}$ represent the highest frequency component in the standardized dS-streams across all training samples. LiGest automatically calculates $F_{\max}$ for a given environment by simply identifying the highest frequency component with notable magnitude across the fourier transforms of all standardized dS-streams of the training samples. If the dimension of the gesture matrix $\mathbf{G}$ for a given environment is $L \times N$ and the highest frequency component in the training samples is $F_{\max}$, the coefficients of level $l^{\psi} = \lfloor \log_2 \{L/F_{\max}\} \rfloor + 1$ completely accommodate all frequencies in the standardized dS-streams of the gestures in that environment. Thus, by using coefficients of level l^{ψ} as features, LiGest can reduce the number of features per sample by a factor of $2^{l^{\psi}-1}$. Consequently, for a given gesture matrix $\mathbf{G}$ with dimension $L \times N$, LiGest obtains a new matrix $\mathbf{G}^{\psi}$ with dimension $L^{\psi} \times N$, where $L^{\psi} = L/2^{l^{\psi}-1}$ and the j^{th} column of $\mathbf{G}^{\psi}$ is comprised of the level l^{ψ} coefficients of the j^{th} column of $\mathbf{G}$. In the samples that we collected from volunteers, $F_{\max} = 31$Hz. Thus, for our dataset, $l^{\psi} = \lfloor \log_2 \{512/31\} \rfloor + 1 = 5$. Thus, using DWT, LiGest reduces the number of features per sample by an order of magnitude from $512 \times 36 = 18432$ to $32 \times 36 = 1152$.

Unfortunately, standard DWT is not well suited to calculate coefficients for gesture samples because the sets of coefficients that standard DWT produces for two versions of the same signal that are shifted in time are different [70]. To overcome this problem, we use a variant of standard DWT, called stationary wavelet transform (SWT) [70], which produces same sets of coefficients for signals that are shifted in time but are otherwise the same. In applying SWT, we tested four wavelets, *i.e.*, Daubechies2, Coiflet2, Symlet2 and Haar, and chose the Haar wavelet due to its highest accuracy.

6.2 Rasterization

6.2.1 Motivation. A change in the position of a light source or the position or orientation of a person with respect to the light source causes the length and/or direction of the user's shadow to change, which in turn causes the shadow to fall on a different set and/or different number of light sensors. If this change in position only changes the direction of the shadow, then the new sensors on which the shadow falls see the same pattern for a given gesture as the previous sensors. If we can combine the time-series of all sensors in such a way that the resulting time-series is always the same for multiple samples of the same gesture irrespective of the set of sensors on which the shadow falls, then the change in the position of the light source or user does not have any impact.

If this change in position also changes the size of shadow, then the number of sensors on which the shadow falls also changes proportionately. For example, if the size of shadow doubles, the number of sensors on which the shadow falls also approximately doubles. In this case, the pattern that a sensor was seeing earlier for a given gesture is now seen by two sensors, but the pattern seen by the two sensors remains the same. In other words, we get redundant time-series. Similarly, a change in the number of light sources changes the number of shadows of the user. Although each shadow falls on a distinct set of sensors, the patterns of change in the intensity of light seen by each set of sensors are the same. Consequently, change in the number of light sources only changes the number of redundant time-series among the N time-series. If we can combine the time-series of all sensors in such a way that the resulting time-series contains only a single contribution from every set of redundant time-series, then the change in the position and/or number of light sources and position and/or orientation of user does not have any impact.

6.2.2 Intuition. Motivated by the discussion above, in this section, we design an operation named *rasterization* that achieves following two objectives. It combines the time-series of all sensors in such a way that the resulting time-series 1) is always the same for multiple samples of the same gesture irrespective of the position of the light source or position or orientation of user, and 2) contains only a single contribution from every set of redundant time-series. To understand the intuition behind our rasterization operation, consider the $L^{\psi} \times N$ values in the matrix $\mathbf{G}^{\psi}$ to be points in 3D space, where the x-axis represents the index of coefficient and goes from 0 to L^{ψ}, y-axis represents the index of sensor and goes from 0 to N and the z-axis represents the coefficient values and can lie in the range $(-\infty, \infty)$. Figure 6 shows a 3D plot containing 32×36 values of the matrix $\mathbf{G}^{\psi}$ of a clap gesture sample. Each line in this figure represents the coefficients corresponding to an individual sensor. We observe from this figure that the three black lines are very similar in shape to each other because these lines correspond to three sensors that see three redundant shadows. If we take a projection of this 3D space onto the 2D x, z-plane, the projection of each of these three lines will almost exactly lie on the same points on the x, z-plane, *i.e.*, the projection essentially contains a single contribution from these three lines. We call the x, z-plane a raster. Thus, by taking this projection, we achieve the second objective. Note that when the set of sensors that sees a shadow changes, only the order of the lines in Figure 6 changes, not their patterns. Consequently, the projection of the lines onto the x, z-plane stays the same. Thus, by taking the projection, we also achieve the first objective. LiGest can use the values of this x, z-plane (*i.e.*, the raster) as features, which are now agnostic to the position and number of light sources and the position and orientation of user.

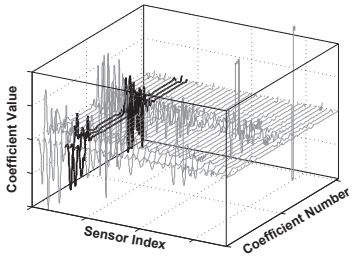


Fig. 6. 3D plot of $\mathbf{G}^{\psi}$ for clap gesture

Fig. 7. Raster of gesture in Figure 6

6.2.3 Method. Having explained the intuition behind rasterization, next, we formally describe how LiGest performs the rasterization operation on any given matrix $\mathbf{G}^{\psi}$ and obtains the corresponding raster. LiGest performs the rasterization operation in following three steps. First, it initializes a raster matrix $\mathbf{R}$ of dimensions $P \times Q$ to 0. This raster matrix is a binary matrix. The dimensions P and Q of this raster correspond to the z-axis and x-axis of Figure 6, respectively. Second, LiGest calculates the values of P and Q as follows. Let $\max_c$ and $\min_c$ represent the maximum and minimum values, respectively, of SWT coefficients among the matrices $\mathbf{G}^{\psi}$ of all training samples. LiGest sets $P = (\max_c - \min_c) \times \Delta_1$ and $Q = L^{\psi} \times \Delta_2$, where Δ_1 and Δ_2 set the granularity of the raster and should be set to the smallest values that provide acceptable accuracy. LiGest empirically determines the values of these parameters using grid search and 10-fold cross validation on the training samples. In our implementation $\max_c = 20$, $\min_c = -20$, $\Delta_1 = 2.5$ and $\Delta_2 = 16$. Third, $\forall i \in [1, L^{\psi}]$ and $j \in [1, N]$, LiGest maps the value of the element G^{ψ}_{ij} of the matrix $\mathbf{G}^{\psi}$ to an element R_{mn} of the raster matrix $\mathbf{R}$ and sets the element R_{mn} to 1, where $m \in [1, P]$ and $n \in [1, Q]$. LiGest does this mapping using the following two equations.

$$n = \left\lceil i \times \frac{Q}{L^{\psi}} \right\rceil, \quad m = \left\lceil \left(\frac{G^{\psi}_{ij} - \min_c}{\max_c - \min_c} \right) \times (P-1) + 1 \right\rceil$$

Figure 7 plots the raster matrix corresponding to the 3D plot in Figure 6. The white regions correspond to the elements of the raster matrix that are 1 and the black regions correspond to the elements that are 0.

6.2.4 Compression. A large part of Figure 7 is black, which means that the raster matrix is sparse. By applying an appropriate compression scheme on the raster matrix, we can reduce its size significantly by reducing the futile black region, which in turn will reduce the number of features that LiGest extracts. To compress the raster matrix, LiGest uses an approach inspired by a widely used image compression technique, JPEG2000 [67], during which it applies a two dimensional DWT on the raster matrix and chooses the coefficients at level 2. This results in a compressed raster matrix $\mathbf{R_c}$ of dimension $P_c \times Q_c$. In our implementation, as $P = 100$ and $Q = 512$, this dimension turns out to be 50×256. In computing DWT, LiGest uses Haar wavelet and level 2 because these provide the highest accuracies.

6.3 Principal Component Analysis

While LiGest can use the $P_c \times Q_c$ values in the compressed raster as features, the number of features is still quite large. In this section, we use principal component analysis (PCA) to reduce the number of features further while preserving majority of the useful information contained in the compressed raster. PCA is an orthogonal transformation that converts a set of samples containing correlated features into a corresponding set of samples containing linearly uncorrelated features. These linearly uncorrelated features are called principal components and are equal to the number of features in the original samples. The values of the first principal component, *i.e.*, the values of the first uncorrelated feature have the largest variance. The variance reduces monotonically for subsequent principal components. As the values of each of the first few principal components show very high variance, these first few components contain majority of the information contained in the data. Thus, by using only the first few principal components as features, we can achieve an accuracy just as good as the accuracy that would result from using all principal components.

LiGest applies PCA to reduce the number of features in the following five steps. First, as each value of the compressed raster matrix can be used as a feature, LiGest constructs a feature vector by appending all P_c rows of the raster matrix one after the other. The resulting feature vector has a length of $U = P_c \times Q_c$. Second, let T represent the total number of training samples. LiGest generates a training set matrix **TS** by stacking feature vectors of all training samples vertically. The dimension of this matrix **TS** is $T \times U$. Third, LiGest calculates the covariance matrix **C** of the matrix **TS**, which has a dimension of $U \times U$. Fourth, LiGest performs the eigen decomposition of the covariance matrix **C** to obtain U eigen vectors, where the dimension of each eigen vector is $U \times 1$ and the i^{th} eigen vector, $\mathbf{v}_i$, has the i^{th} largest eigen value, where $1 \leq i \leq U$. Last, for any given sample, LiGest calculates its i^{th} principal component, where $1 \leq i \leq U$, by projecting the feature vector of this sample on the i^{th} eigen vector, *i.e.*, taking the dot product of the feature vector with $\mathbf{v}_i$.

Figure 8 plots the classification accuracy of LiGest on a subset of our dataset when using first i principal component values for each sample as features. We see that as we increase the number of principal components used as features beyond 1, the accuracy of LiGest increases rapidly because subsequent principal components add new information. However, beyond using 50 principal component values, the rate of improvement of accuracy reduces significantly because the majority of information is contained within the first few principal components, and subsequent principal components do not add much new information. Thus, it is not useful to use components beyond 50, which only increases the computational complexity in training classifiers. Thus, LiGest uses the first 50 principal component values as features to build classification models. With this method, we reduced the number of features of compressed raster by two orders of magnitude from $50 \times 256 = 12800$ to just 50.

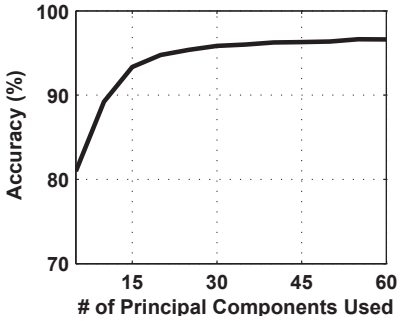


Fig. 8. Impact of the number of principal components on accuracy

7 CLASSIFIER TRAINING

After extracting feature values, LiGest has one feature vector of length 50 per gesture sample. LiGest uses feature vectors from all samples of all gestures in the training set and builds classification models using support vector machine (SVM). While there are several other choices to build classification models such as decision trees, naive bayes, and k-nearest neighbor, we chose SVM because a compressed raster (from which the final feature vector was obtained) is essentially a compressed black and white image and SVM has been shown to be very effective for image classification [36]. Next, we describe how LiGest uses SVM and refer interested readers to [32, 97] for more details on SVM.

To generate classification models to recognize a set of n different gestures, LiGest uses SVM in 1-vs-all configuration, where it generates n two-class classification models. Each two-class classification model is generated by treating samples of one gesture to belong to the positive class and samples of the remaining $n-1$ gestures to belong to the negative class. Each gesture is treated as positive class in exactly one of the n two-class classification models. In generating each two-class classification model LiGest uses the standard radial basis function (RBF) kernel and obtains the optimal values of the two tunable parameters, C and γ, using grid search [32, 85].

Recognizing Gestures at Runtime: To recognize a gesture at runtime, LiGest continuously denoises the dS-streams using the approach described in Section 5.1. It continuously monitors the denoised dS-streams to detect a gesture using the approach described in Section 5.2. To recognize successive gestures, LiGest requires the users to take a short pause of about 1 second between successive gestures. If a user does not take the pause, LiGest will treat the successive gestures as a single gesture, resulting in an error. Note that many recent gesture recognition systems, such as those proposed in [40, 52, 75, 100], also impose a similar requirement. As soon as it detects a gesture, it first standardizes the denoised dS-streams of this unknown gesture sample using the approach described in Section 5.3. Next, it extracts the feature vector by applying the three operations of wavelet transformation, rasterization, and principal component analysis on the standardized dS-streams, as described in Sections 6.1, 6.2, and 6.3, respectively. Note that in applying these three operations, LiGest uses the values of parameters (such as DWT and SWT levels, eigen vectors, and the number of PCA components) calculated at the time of building classification models from the training samples. Last, it evaluates the feature vector against all n two-class classification models and declares the unknown sample to be that gesture whose two-class classification model returns the highest likelihood of this unknown sample belonging to its positive class. Note that LiGest's speed in recognizing gestures at runtime is very fast (130 ms per gesture in our implementation) because to recognize a gesture, LiGest only has to extract features and check how close that set of features lies to the classification boundary of each of the n two-class classification models.

8 PERFORMANCE EVALUATION

In this section, we evaluate the performance of LiGest using data collected from real volunteers. We quantify the performance of LiGest in terms of accuracy, which is defined as the percentage of samples that LiGest recognizes correctly in a given set that contains unseen samples of the predefined gestures. We emphasize that we cannot compare LiGest with LiSense [58] and StarLight [60] due to the fundamental differences in their setups and objectives, as described in Section 1.3. Next, we first describe how we collected the data that we used in evaluating LiGest. After that, we evaluate the overall accuracy of LiGest followed by the evaluation of accuracy on unseen user positions, unseen user orientations, unseen lighting conditions, unseen users, unscripted positions/orientation, changing surroundings, changing sunlight and then provide a comparison with GestureLite [51]. We then study the effect of the four key operations of LiGest on the accuracy of LiGest, viz., denoising, standardization, wavelet transformation, and rasterization. Note that we already presented the effect of the number of PCA components on the accuracy of LiGest in Figure 8. Last, we test the limits of LiGest against obstructions, variations in illuminance and lighting density.

8.1 Data Collection

We collected a total of 15175 samples for five different gestures (punch, hug, clap, step, and jump) from 20 volunteers (14 males and 6 females) with ages ranging from 22 to 28 years and heights ranging from 150cm to 180cm. We carefully chose the five gestures so that the set of gestures we use to evaluate LiGest incorporates a good mix of motion from arms (punch and hug), hands (clap), feet (step), and entire body (jump). Before collecting samples from each volunteer, we gave a video demonstration of each gesture to each volunteer. We collected these samples with prior IRB approval. Next, we describe how we collected samples to evaluate the accuracy of LiGest in various scenarios.

Data collection for evaluating overall accuracy, accuracy on unseen user positions and orientations, unseen lighting conditions, and unseen users: To evaluate the accuracy of LiGest on *unseen user positions and orientations*, we placed our 6ft × 6ft sensing platform in the center of our 25ft × 16ft lab, and collected samples at 9 different positions on the platform, shown as triangles (P1 to P9) in Figure 9, and in 4 different orientations at each of these 9 position, shown as four arrows at P5 in this figure. Note that the sensing platform and the room are drawn at different scales in Figure 9. Figure 10 draws the sensing platform and the room at the same scale. The dimensions inside the sensing platform, however, are at the same scale as the platform. We emphasize here that when used in practice, the user is not confined to only these 9 positions or 4 orientations to perform gestures. In a production level deployment, as long as sensors cover the entire floor, the user can perform gestures anywhere on the floor. To make our setup more clear, we indicate the 9 positions relative to the sensors in Figure 1. To evaluate the accuracy of LiGest on *unseen lighting conditions*, we collected samples in two arbitrarily chosen lighting conditions. We created the first lighting condition using four 45-inch fluorescent tube-lights, three of which were situated along a line (the length of the line is approximately 45 × 3 inches) directly above the imaginary line connecting P4, P5, and P6 and the fourth was situated 7 feet towards the right of this line from P6 towards P1. We created the second lighting condition using six 45-inch fluorescent tube-lights, three of which were again situated along a line directly above the imaginary line connecting P4, P5, and P6 and the remaining three were situated along another parallel line 7 feet towards the right of the first line. All tube-lights were at a height of 7.2ft from the floor. The two rows of lights when viewed from the camera perspective in Figure 10 appear as shown in Figure 11. To evaluate the accuracy of LiGest on *unseen users*, we collected samples from 20 different volunteers. In each of the 4 orientations at each of the 9 positions under each of the two lighting conditions, from each of the 20 volunteers, we collected two samples for each gesture. We name the set of all these samples dataset-1. To incorporate the effects of stray shadows, we placed a chair around the mat and randomly moved it between samples such that its shadow fell on the sensing platform.

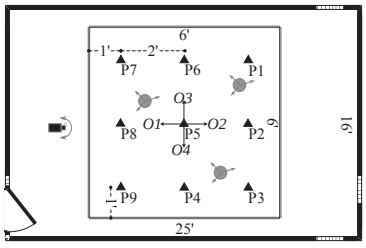


Fig. 9. Layout of sensing platform in lab, not drawn to scale

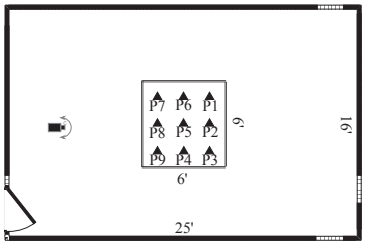


Fig. 10. Layout of sensing platform in lab, drawn to scale

Fig. 11. Data collection setup: lab

Data collection for evaluating accuracy with unscripted positions, unscripted orientations, and changing surroundings: To evaluate the accuracy of LiGest with unscripted changes in surroundings, we collected samples in three unscripted scenarios in the same lab where we collected dataset-1. In the first scenario, we

turned on four ceiling lights and opened the room door from where the outside light entered the room while people walked in front of the door randomly at different speeds. Due to the people walking, not only the amount of outside light entering the room changed, the shadows of people falling on parts of the sensing platform also changed randomly. In the second scenario, we additionally placed a chair in front of the door that casted a constant stray shadow on the platform in addition to the shadows of the people walking in front of the door. In the third scenario, we asked a volunteer to arbitrarily choose three positions on the sensing platform and provide samples for all gestures at each of those three positions in three orientations of volunteer's choosing. We name the set of all samples collected in these three scenarios dataset-2. We acknowledge that there are potentially many possible unscripted scenarios; we just arbitrarily chose the three described above.

Data collection for evaluating accuracy with changing sunlight: To evaluate the accuracy of LiGest with changing amount of sunlight in an environment, we placed our sensing platform in a 26ft × 28ft living room right next to a large patio window from where sunlight entered the living room, as shown in Figure 12 and as seen in Figure 13. The living room is illuminated by a single LED based lamp (which we do not control in any way, except that we turned it on). We collected samples at different positions and orientations at 9 randomly chosen times throughout the day. We name the set of all these samples dataset-3. The ambient light intensity at these 9 times ranged from 162 Lux to 2584 Lux (measured using a light meter).

Figure 14 plots the CDFs of the time taken by the volunteers to perform the samples of each gesture in our datasets. As an example of how to read this figure, consider the line with star markers that corresponds to the "clap" gesture. This line tells us that out of all the samples that we have for the "clap" gesture, the amount of time it took the volunteers to complete a single clap gesture was less than 1 second in about 62% of the samples, less than 2 seconds in about 78% of the samples, and less than 3 seconds in almost all 100% of the samples. We observe from this figure that the times taken to perform different samples of each gesture vary from less than a second to over three seconds. This shows that for each gesture, our dataset has a good mix of samples with speeds ranging from very slow to very fast. We further observe that the jump and step gestures take longer time compared to the other gestures. This happens due to the additional hand/arm movements, such as swaying, caused by inertia at the end of the gesture.

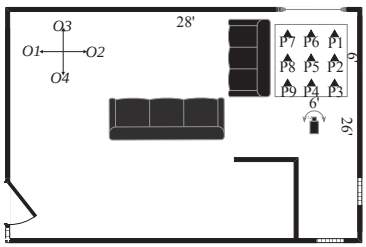


Fig. 12. Layout of the living room environment

Fig. 13. Data collection setup: living room

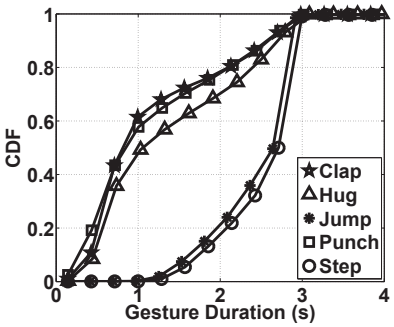


Fig. 14. CDFs of gesture durations

8.2 Gesture Recognition Accuracy

In this section, we evaluate the overall accuracy of LiGest as well as its accuracy on unseen user positions, unseen user orientations, unseen lighting conditions, unseen users, changing surroundings, and changing sunlight. Table 1 summarizes the accuracies of LiGest in different scenarios that we will evaluate it in.

8.2.1 Accuracy – Overall. To calculate the average accuracy of LiGest, we used 10-fold cross validation on the samples of the gestures in dataset-1. For the experiments that we will present in the subsequent sections, we will not use cross validation because for each of those experiments, we can divide the samples in our dataset into well-defined training and testing sets.

Table 1. Summary of the accuracies achieved by LiGest in different scenarios

Scenario	Average Accuracy
Overall	96.36 %
Unseen Positions	95.70 %
Unseen User Orientations	94.50 %
Unseen Lighting Conditions	93.00 %
Unseen Users	94.64 %
Unscripted Positions & Orientations	95.56 %
Unscripted surrounding changes	93.58 %
Changing Sunlight	95.90 %

LiGest achieves an average accuracy of 96.36%. Figure 15 shows the resulting confusion matrix for all gestures. We observe from this figure that LiGest achieves the highest average accuracy of 99% for the jump and step gestures. We also observe from this figure that the accuracy is fairly stable around all the gestures, with Clap having the least accuracy of 92%.

8.2.2 Accuracy – Unseen User Position. LiGest achieves an average accuracy of 95.7% *for gestures performed at positions it has not been trained on.* To measure the accuracy of LiGest at unseen positions, for each position in dataset-1, we tested all samples in dataset-1 collected at that position using classification models generated from the samples in dataset-1 collected at all other positions. Figure 16 plots LiGest's aggregate accuracy across all gestures at each position using a black bar and accuracy for individual gestures at each position using patterned and gray-scale bars. The gesture names are abbreviated in the legend as 'C' for clap, 'H' for hug, 'J' for jump, 'P' for punch, and 'S' for step. In the legend, 'A' stands for aggregate accuracy across all gestures. We observe from this figure that the positions 7, 8, and 9, which are the farthest from all light sources, see slightly lower accuracies due to lighter shadows, while positions 4, 5, and 6, which are closest to the light sources, see relatively higher accuracies due to darker shadows.

Fig. 15. Confusion matrix for dataset-1

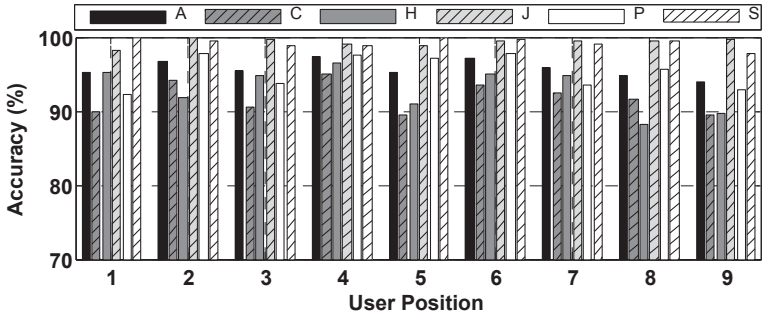


Fig. 16. Accuracy of LiGest at unseen positions

8.2.3 Accuracy – Unseen User Orientation. LiGest achieves an average accuracy of 94.5% *for gestures performed in orientations it has not been trained on.* To measure the accuracy of LiGest in unseen user orientations, for each orientation in our dataset-1, we tested all samples collected in that orientation using classification models generated from the samples in dataset-1 collected in all other orientations. Figure 17 plots LiGest's aggregate accuracy across all gestures and its accuracy for individual gestures at each orientation. The numbers on the x-axis correspond to the numbers assigned to the orientations shown by the arrows originating from P5 in Figure 9. We observe from this figure that the aggregate accuracies span a range of only 3.86%, indicating LiGest's

consistent performance across orientations. We also observe from this figure that in a couple of orientations, LiGest got confused between the clap and hug gestures, which caused the accuracy of LiGest in recognizing these two gestures to fall slightly below 90%. This happened because when performing these gestures, due to the direction the users were facing (see Figure 9), their bodies blocked parts of the shadow for the clap gesture, and the clap gesture appeared very similar to hug gesture.

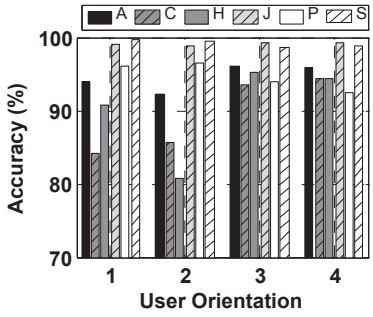


Fig. 17. Accuracy in unseen orientations

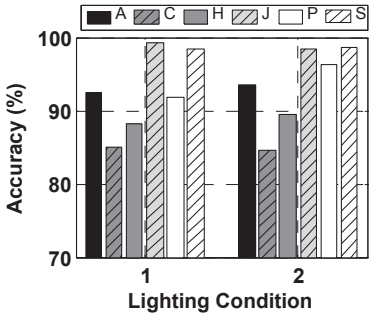


Fig. 18. Accuracy in unseen lighting cond.

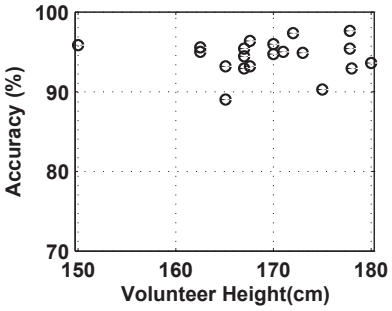


Fig. 19. Accuracy on unseen users

8.2.4 Accuracy – Unseen Lighting Conditions. LiGest achieves an average accuracy of 93% *for lighting conditions it has not been trained on.* To measure the accuracy of LiGest under unseen lighting conditions, for each lighting condition, we tested all samples in set-1 collected under that lighting condition using classification models generated from the samples in dataset-1 collected under the other lighting condition. Figure 18 plots LiGest's accuracy under each lighting condition. We see that the aggregate accuracy is slightly lower compared to the accuracies we have seen until now. The reason is that for this set of experiments, as the entire dataset is divided into two equal parts for training and testing, we have far fewer samples to generate classification models compared to the number of samples we had in the previous experiments.

8.2.5 Accuracy – Unseen Users. LiGest achieves an average accuracy of 94.64% *for gestures performed by user it has not been trained on.* To measure the accuracy of LiGest on unseen users, for each volunteer in our dataset, we tested all samples in dataset-1 collected from that volunteer using classification models generated from the samples in dataset-1 collected from all other volunteers. Figure 19 shows the resulting average accuracy of LiGest across all gestures for each volunteer in our dataset. Note that the accuracy on the right is for the tallest volunteer and on the left is for the shortest. We observe from this figure that the accuracies for *unseen* volunteers are always above 89% and have only small deviations across volunteers. Further, we do not see any trends in accuracies with changing heights, which shows that LiGest is unaffected by user's height.

8.2.6 Accuracy – Unscripted Positions and Orientations. LiGest achieved an average accuracy of 95.56% when tested at unscripted positions and orientations. To measure the accuracy of LiGest at unscripted positions and orientations, we tested the samples in dataset-2 that we collected in each of the three arbitrary positions and orientations of the volunteer's choosing by using classification models generated from the samples in dataset-1. The three positions and orientations chosen by the user are shown by the gray-colored circles and arrows in Figure 9. Figures 20 and 21 plot LiGest's average accuracy at each of the three positions and three orientations, respectively. The positions are numbered from top to bottom as seen in Figure 9 and the orientations are numbered clockwise starting from top. The average accuracy of 95.56% in this experiment is very close to the average accuracy of 95.7% that we saw in Section 8.2.2 for the unseen positions and orientations experiment. This again demonstrates that LiGest is very robust to changes in user position and orientation.

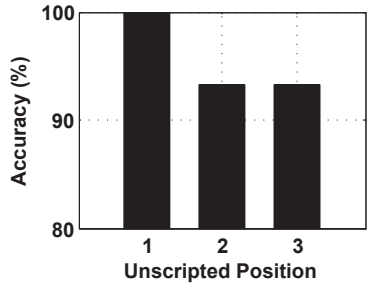


Fig. 20. Accuracy with unscripted positions

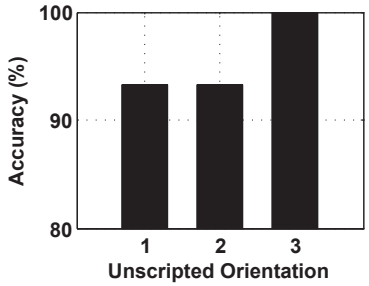


Fig. 21. Accuracy with unscripted orientations

8.2.7 Accuracy – Unscripted Surrounding Changes. The average accuracy of LiGest reduces by less than 3% with unscripted surrounding changes. To measure the accuracy of LiGest with unscripted surrounding changes, we tested all samples in dataset-2 collected in each of the two unscripted scenarios using classification models generated from the samples in dataset-1, which were collected with the lab door closed. Figure 22 plots the loss in LiGest's average accuracy in the two unscripted scenarios compared to the average accuracy reported in Section 8.2.1. We observe from this figure that the loss in accuracy is similar in both scenarios. This shows that the stray shadow casted by the furniture has little effect on the accuracy of LiGest.

8.2.8 Accuracy – Changing Sunlight. The average deviation in the accuracy of LiGest is less than 2.76% with changing sunlight. To measure the accuracy of LiGest at each of the 9 different times of the day with different amounts of sunlight, we tested all samples in dataset-3 collected at that time of day using classification models generated from the samples in dataset-3 collected at all other times of the day. Figure 23 plots the average accuracy of LiGest across all gestures at each time of day. We see that the average accuracies lie in the range 90% to 99%.

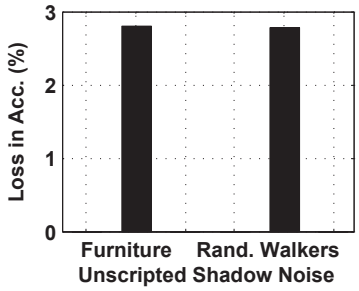


Fig. 22. Accuracy: changing surroundings

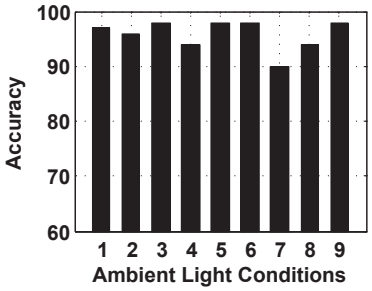


Fig. 23. Accuracy at different times of day

8.2.9 Accuracy – Comparison with GestureLite [51]. LiGest achieves 30.87% higher average gesture recognition accuracy compared to GestureLite. To do this comparison, we implemented the methods employed by GestureLite and repeated the experiments described in Section 8.2.1 using GestureLite. More specifically, we used the methods proposed in GestureLite to recognize gestures and performed 10-fold cross validation on the samples of the gestures in dataset-1. Figure 24 shows the resulting confusion matrix for all gestures. We observe from this figure that GestureLite achieves an overall average accuracy of 65.49%, which is 30.87% lower than the overall accuracy of 96.36% achieved by LiGest on the same data set. The reason for this low accuracy is the set of limitations of GestureLite described in Section 2.

8.3 Effect of Internal Operations on Accuracy

Next, we evaluate how much accuracy LiGest loses if we skip one of the four key operations of LiGest, namely denoising, standardization, wavelet transformation, and rasterization. To calculate the loss in accuracy due to

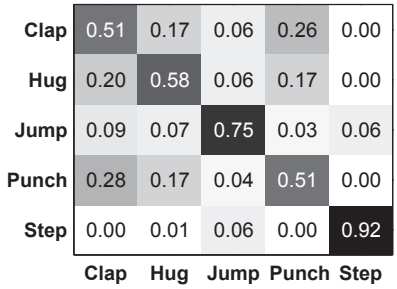


Fig. 24. Confusion matrix of GestureLite for dataset-1

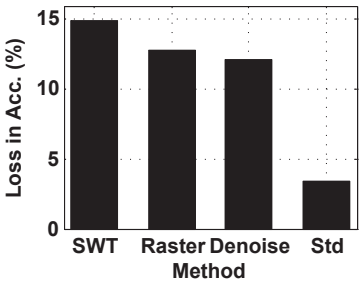


Fig. 25. Effect of operations on accuracy

not performing a given operation, we do 10-fold cross validation on all samples in dataset-1 just like we did for the results reported in Section 8.2.1, except that, now, we skip that operation. To skip the denoising operation, we directly apply standardization operation on raw S-streams instead of applying it on denoised *d*S-streams. To skip the standardization operation, we directly apply SWT on denoised *d*S-streams instead of applying it on standardized *d*S-streams. To skip the wavelet transformation operation, we directly apply rasterization on the matrix $\mathbf{G}$ of each sample instead of applying it on $\mathbf{G}^{\psi}$. To skip the rasterization operation, we directly apply PCA on matrix $\mathbf{G}^{\psi}$ instead of applying it on compressed raster. Figure 25 plots the loss in accuracy corresponding to skipping each of these operations. We observe that while lack of any operation causes a loss in LiGest's accuracy, the maximum loss in accuracy occurs when skipping the wavelet transformation operation.

8.4 Limitation Analysis

In this section, we characterize the impact of various adverse real-world situations on the accuracy of LiGest. More specifically, we study the impacts of obstructions from furniture, varying illumination levels and distances from light sources, and the density of light sources on the accuracy of LiGest. To calculate the accuracy of LiGest in each scenario under consideration in this section, we tested all samples collected in that scenario using classification models generated from the samples in dataset-1. Next, we describe how we collected data for each scenario and present the accuracy of LiGest.

8.4.1 Impact of Obstructions. As LiGest uses a floor-based sensing platform, user shadows can be blocked by furniture, such as a table or a chair, which in turn can impact the accuracy of LiGest. To characterize the performance of LiGest under such obstructions, we chose to place a chair of height 96cm near the user such that it blocked a desired percentage of the user's shadow when the user performed gestures, as shown in Figure 26. As we know the locations of light sources, the position of our sensing platform, and the height of the users, we used these values to calculate exactly which direction will the shadows of the user fall and what will be the length of the shadows when the user stands at each of the nine positions P1 through P9 to perform gestures. Using these calculations, for each of the 9 positions, we determined exactly where to put the chair on the platform such that when the user performs gestures at that position, the chair obstructs a user's shadow by a desired percentage (we used three values for this desired percentage of obstruction: 25%, 50%, and 75%). Next, for each value of the desired obstruction, we collected samples at each of the 9 positions for all gestures. More specifically, for each position, we first placed the chair such that it will block s% of user's shadow, where $s \in \{25, 50, 75\}$, and then asked the user to stand at that position and provide samples for all gestures.

The accuracy of LiGest stays largely unaffected even when 50% of the shadow of a user is obstructed. Figure 27 plots the average accuracy of LiGest across all 9 positions for the four percentages of obstruction. We see that the average accuracy reduces by just 1.78% and 2.22% when 25% and 50% of a user's shadow is obstructed, respectively. This stability in this accuracy of LiGest is attributed to two things: 1) the presence of multiple lights, which create

redundant shadows on the platform, and 2) our rasterization operation, which leverages this redundancy and mitigates the effects of obstructions of different parts in different shadows of the user. Our rasterization operation essentially takes the useful parts of all shadows and "stitches" them together, which enables LiGest to achieve high accuracy despite obstruction. We also observe from this figure that when the percentage of the obstruction of shadows is really large, such as 75%, LiGest starts to experience more significant drops in accuracy due to the lack of enough useful parts in redundant shadows that the rasterization operation can "stitch" together to obtain a useful raster.

We also conducted an experiment, where instead of having multiple light sources, we used only a single light source, and thus no redundant shadows. Figure 28 shows the average accuracies achieved by LiGest for 50% obstruction when there are two light sources and when there is only one light source. As expected, in the absence of redundant shadows, *i.e.*, with a single light source, the obstruction impacts LiGest's accuracy significantly. Fortunately, in almost all modern buildings, multiple light sources are used to illuminate the environment. The reason behind using multiple light sources instead of a single light source is that if a single light source is used to achieve the minimum recommended illumination of 400 lux, the light source will have to be very bright. Such a bright light source is discouraged because if an occupant accidentally looks at it, it can cause temporary or permanent damage to the occupant's sight.

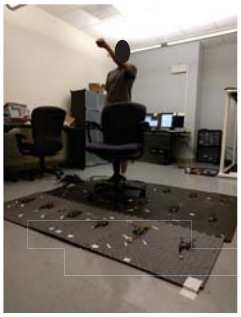


Fig. 26. Setup for creating obstructions using a chair

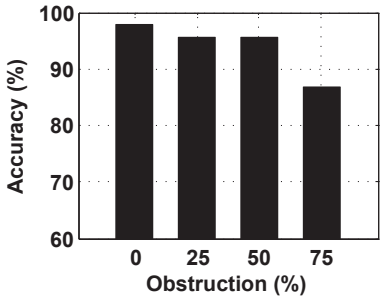


Fig. 27. Impact of obstructions on LiGest's accuracy

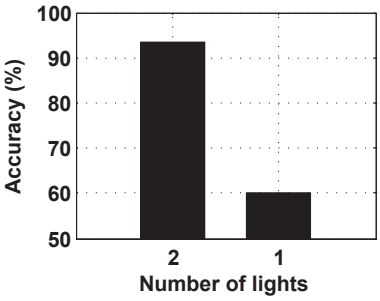
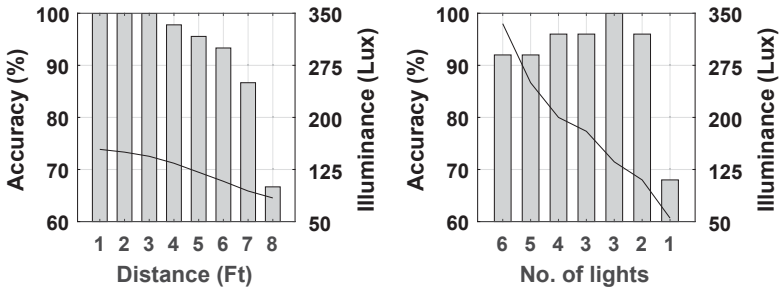


Fig. 28. Impact of the number of light sources along with obstructions

8.4.2 Impact of Illumination and Distance from Light Source. As LiGest uses a 6ft × 6ft sensing platform, which is much smaller in size compared to the size of the room where it is placed, it is imperative to characterize the performance of LiGest when this platform is placed at different distances from the light sources. To do this, we turned on the three fluorescent lights directly above the imaginary line connecting the positions P4, P5, and P6, when the sensing platform is in its initial position shown in Figures 9 and 10. Next, we moved the entire 6ft × 6ft sensing platform left towards the wall (that has the door) in steps of 1ft and collected samples for all gestures at the center of the platform. This movement of the platform away from the light sources exposes LiGest to two challenges: increasing lengths of shadows and decreasing ambient light intensity.

The accuracy of LiGest experiences a notable drop only when the illumination level falls below 100 lux. Note that the illumination of 100 lux is much smaller than the minimum recommended uniform illuminance of 400 lux for indoor environments [37]. This can be observed from Figure 29(a), which plots the average accuracy of LiGest on the samples that were collected at different distances of the sensing platform from its initial location. The solid line in this figure indicates the average illuminance at these distances. We observe from this figure that the accuracy of LiGest remains unchanged when the platform is moved by up to 3ft from its initial location. We see only a slight drop in accuracy when the platform is moved beyond 3ft and up to 6ft from its initial location. However, the accuracy drops sharply beyond 6 ft because the field of view of our photodiodes is 72° and at these

distances, the light source has moved out of the field of view of most photodiodes. Consequently, the photodiodes see an average illumination of just 84 lux at 8 ft, which is a very low level of illumination. This makes the shadows weak and LiGest cannot distinguish between the hand based gestures, which results in the loss in accuracy. This limitation is fundamental, and establishes a lower bound on the required light intensity. As the illuminance under 100 lux is below the recommended lower limit of 400 lux for indoor environments [37], it is safe to assume that in active environments, where such gesture recognition systems are useful, the illuminance level will be at least 400 lux, at which LiGest performs very well.



(a) Accuracy vs. distance/illuminance (b) Accuracy vs. light density

Fig. 29. Impact of distance/illumination from light sources and density of light sources on LiGest's accuracy

8.4.3 Impact of the Density of Light Sources. An increase in the density of light sources can weaken the contrast of shadows, and thus impact LiGest's accuracy. To study the impact of the density of light sources, we placed our sensing platform under the two rows of lights such that the center of the platform was situated exactly in the middle of the imaginary perpendicular line connecting the two rows (see Section 8.1 to recall the positions of the lights). Next, we collected samples for all gestures at the center of the platform under eight different combinations by toggling different lights in the two rows on and off. We controlled the six lights in the two rows in such a way that we could turn on or off one, two, and/or three lights simultaneously. More specifically, all three lights in the left row were operated using one switch, two lights in the right row were operated using another switch, and the remaining one light in the right row was operated using a third switch. As we have three switches, we get a total of $2^3 = 8$ combinations. Note that these 8 combinations cover all possible numbers of simultaneously turned on light sources, *i.e.*, 0 through 6. We did not collect any samples for the combination where none of the light sources were turned on because LiGest needs light to work.

The increase in the density of light sources does not significantly impact the accuracy of LiGest. This can be seen in Figure 29(b), which plots the accuracy of LiGest under the seven different lighting combinations plotted in the decreasing order of the number of turned-on lights (and thus the illumination). We observe from this figure that barring the only combination in which only a single light source is turned on producing a very low illumination of just 55 lux, the accuracy of LiGest is high and stable across all the other combinations. We also note that when the density of light sources is the highest (*i.e.*, combination 7 with all six lights turned on), the average accuracy of LiGest drops slightly but is still over 92%, despite the highest dilution of shadow contrast arising from the six concentrated tube-lights. With more sensitive photodiodes, we can easily mitigate this slight drop in accuracy with increasing density of light sources.

9 CONCLUSION AND FUTURE WORK

In this paper, we proposed LiGest, an ambient light based gesture recognition system. The key novelty of LiGest is in its high robustness against changing lighting conditions, changing user positions and orientations, and even changing users. LiGest works with all type of light sources and does not require any control over them. The key technical depth of LiGest lies in its internal operations of denoising, standardization, wavelet transformation, and rasterization. We implemented LiGest using cheap commercially available light sensors and Arduino Uno boards and extensively evaluated it in several different scenarios. LiGest achieves an average accuracy of 96.36%, which is comparable to (and often higher than) the average accuracies of existing RF based gesture recognition systems. In future, we plan to extend LiGest on five fronts. First, we plan to extend it to recognize gestures of multiple users simultaneously. Second, we plan to extend it to walls by deploying and experimenting with sensors on walls. Third, we plan to extend LiGest to support very low illuminance levels. Fourth, we plan to extensively study the impact of sensor placement on the accuracy of LiGest. Last, we plan to stress test LiGest by increasing the number of gestures by an order of magnitude.

ACKNOWLEDGMENTS

This work is supported in part by the National Science Foundation, under grant # CNS 1565609. The authors would also like to thank the anonymous reviewers for their valuable comments and helpful suggestions.

REFERENCES

[1] [n. d.]. https://www.mouser.com/ProductDetail/ams/TSL237S-LF/?qs=sGAEpiMZZMuhmhJ32%2fprP004vccwnQtD. ([n. d.]).
[2] [n. d.]. https://www.amazon.com/dp/B01N4LP86I/ref=sspa_dk_detail_2?psc=1&pd_rd_i=B01N4LP86I&pd_rd_wg=XYxWq&pd_rd_r=AF1VN3GKWB063X9FR0J8&pd_rd_w=2IO1M. ([n. d.]).
[3] [n. d.]. https://www.digikey.com/product-detail/en/everlight-electronics-co-ltd/PD15-22C-TR8/1080-1367-2-ND/2675858. ([n. d.]).
[4] [n. d.]. https://www.digikey.com/product-detail/en/microchip-technology/PIC18F45K50T-I-PT/PIC18F45K50T-I-PTTR-ND/3674584. ([n. d.]).
[5] [n. d.]. https://www.digikey.com/product-detail/en/sparkfun-electronics/DEV-11791/DEV-11791-ND/7393714. ([n. d.]).
[6] [n. d.]. https://www.homeadvisor.com/cost/flooring/install-carpeting/. ([n. d.]).
[7] [n. d.]. https://www.homedepot.com/p/Platinum-Plus-Broadway-Color-Hammerhead-Pattern-12-ft-Carpet-HDE0310552/300368944. ([n. d.]).
[8] [n. d.]. Adoption of Light-Emitting Diodes in Common Lighting Applications. http://energy.gov/sites/prod/files/2015/07/f24/led-adoption-report_2015.pdf. ([n. d.]).
[9] [n. d.]. Allure Smart Thermostat. https://www.allure-energy.com/. ([n. d.]).
[10] [n. d.]. Fidbox: a monitor for temperature and relative humidity. http://www.floorprotector.at/fidbox/EN/fidbox.html. ([n. d.]).
[11] [n. d.]. Future Shape SensFloor. http://www.future-shape.com/en/technologies/23. ([n. d.]).
[12] [n. d.]. Honeywell Lyric Thermostat. http://wifithermostat.com/Products/Lyric/. ([n. d.]).
[13] [n. d.]. Honeywell Smart Thermostat. http://wifithermostat.com/Products/WiFiSmartThermostat/. ([n. d.]).
[14] [n. d.]. Insteon LED Bulbs. http://www.insteon.com/led-bulbs/. ([n. d.]).
[15] [n. d.]. Leap Motion. https://www.leapmotion.com/. ([n. d.]).
[16] [n. d.]. LG Smart Appliances. http://www.lg.com/us/discover/smartthinq/thinq. ([n. d.]).
[17] [n. d.]. Nest Thermostat. https://nest.com/thermostat/meet-nest-thermostat/. ([n. d.]).
[18] [n. d.]. Philips Hue. http://www2.meethue.com/en-us/. ([n. d.]).
[19] [n. d.]. Smart Home. http://www.smarthome.com/. ([n. d.]).
[20] [n. d.]. TSL237 Datasheet. https://ams.com/chi/content/download/250250/975933/file/TSL237_Datasheet_EN_v1.pdf. ([n. d.]).
[21] [n. d.]. Whirlpool Smart Appliances. http://www.whirlpool.com/smart-appliances/. ([n. d.]).
[22] [n. d.]. WiFi Plug. http://www.wifiplug.co.uk/. ([n. d.]).
[23] [n. d.]. Xbox Kinect. http://www.xbox.com/en-US/xbox-one/accessories/kinect-for-xbox-one. ([n. d.]).
[24] Heba Abdelnasser, Moustafa Youssef, and Khaled A Harras. 2015. WiGest: A ubiquitous WiFi-based gesture recognition system. In *Proceedings of IEEE INFOCOM*.
[25] Fadel Adib, Zachary Kabelac, and Dina Katabi. 2015. Multi-Person Motion Tracking via RF Body Reflections. In *Proceedings of Usenix NSDI*.

[26] Fadel Adib, Zach Kabelac, Dina Katabi, and Robert C Miller. 2013. 3D tracking via body radio reflections. In *Proceedings of Usenix NSDI*.
[27] Jake K. Aggarwal and S. Ryoo Michael. 2011. Human activity analysis: A review. *Comput. Surveys* 43, 3 (2011).
[28] Shintaro Arai, Shohei Mase, Takaya Yamazato, Tomohiro Endo, Toshiaki Fujii, Masayuki Tanimoto, Kiyosumi Kidono, Yoshikatsu Kimura, and Yoshiki Ninomiya. 2007. Experimental on hierarchical transmission scheme for visible light communication using led traffic light and high-speed camera. In *Vehicular Technology Conference, 2007. VTC-2007 Fall. 2007 IEEE 66th*. IEEE, 2174–2178.
[29] Ashwin Ashok, Marco Gruteser, Narayan Mandayam, Jayant Silva, Michael Varga, and Kristin Dana. 2010. Challenge: Mobile optical networks through visual MIMO. In *Proceedings of the sixteenth annual international conference on Mobile computing and networking*. ACM, 105–112.
[30] Manuel Caputo, Klaus Denker, Benjamin Dums, and Georg Umlauf. 2012. 3D Hand Gesture Recognition Based on Sensor Fusion of Commodity Hardware.. In *mensch & Computer*, Vol. 2012. 293–302.
[31] German KM Cheung, Takeo Kanade, Jean-Yves Bouguet, and Mark Holler. 2000. A real time system for robust 3D voxel reconstruction of human motions. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, Vol. 2. IEEE, 714–720.
[32] Nello Cristianini and John Shawe-Taylor. 2000. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press.
[33] Paul Dietz, William Yerazunis, and Darren Leigh. 2003. Very low-cost sensing and communication using bidirectional LEDs. In *UbiComp 2003: Ubiquitous Computing*. Springer, 175–191.
[34] David L Donoho and Iain M Johnstone. 1995. Adapting to unknown smoothness via wavelet shrinkage. *Journal of the american statistical association* 90, 432 (1995), 1200–1224.
[35] Emre Ertin, Nathan Stohs, Santosh Kumar, Andrew Raij, Mustafa al'Absi, and Siddharth Shah. 2011. AutoSense: unobtrusively wearable sensor suite for inferring the onset, causality, and consequences of stress in the field. In *Proceedings of ACM Sensys*.
[36] G. M. Foody and A. Mathur. 2004. A relative evaluation of multiclass image classification by support vector machines. *IEEE Transactions on Geoscience and Remote Sensing* 42, 6 (June 2004), 1335–1343. https://doi.org/10.1109/tgrs.2004.827257
[37] Steve Fotios. 2011. Lighting in offices: lamp spectrum and brightness. *Coloration technology* 127, 2 (2011), 114–120.
[38] Dawid Gradolewski and Grzegorz Redlarski. 2013. The use of wavelet analysis to denoising of electrocardiography signal. In *XV International PhD Workshop OWD 2013, 19*, Vol. 22.
[39] Ye Gu, Ha Do, Yongsheng Ou, and Weihua Sheng. 2012. Human gesture recognition through a kinect sensor. In *Robotics and Biomimetics (ROBIO), 2012 IEEE International Conference on*. IEEE, 1379–1384.
[40] Sidhant Gupta, Daniel Morris, Shwetak Patel, and Desney Tan. 2012. Soundwave: using the doppler effect to sense gestures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1911–1914.
[41] Chunmei Han, Kaishun Wu, Yuxi Wang, and Lionel M Ni. 2014. WiFall: Device-free fall detection by wireless networks. In *Proceedings of IEEE INFOCOM*. 271–279.
[42] Tian Hao, Ruogu Zhou, and Guoliang Xing. 2012. Cobra: color barcode streaming for smartphone systems. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*. ACM, 85–98.
[43] Lorna Herda, Pascal Fua, Ralf Plänkers, Ronan Boulic, and Daniel Thalmann. 2000. Skeleton-based motion capture for robust reconstruction of human motion. In *Computer Animation 2000. Proceedings*. IEEE, 77–83.
[44] Nicholas R Howe, Michael E Leventon, and William T Freeman. 1999. Bayesian Reconstruction of 3D Human Motion from Single-Camera Video.. In *NIPS*, Vol. 99. 820–6.
[45] Wenjun Hu, Hao Gu, and Qifan Pu. 2013. Lightsync: Unsynchronized visual communication over screen-camera links. In *Proceedings of the 19th annual international conference on Mobile computing & networking*. ACM, 15–26.
[46] Wenjun Hu, Jingshu Mao, Zihui Huang, Yiqing Xue, Junfeng She, Kaigui Bian, and Guobin Shen. 2014. Strata: layered coding for scalable visual communication. In *Proceedings of the 20th annual international conference on Mobile computing and networking*. ACM, 79–90.
[47] Donny Huang, Rajalakshmi Nandakumar, and Shyamnath Gollakota. 2014. Feasibility and limits of Wi-Fi imaging. In *Proceedings of ACM SenSys*. 266–279.
[48] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, et al. 2011. KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. ACM, 559–568.
[49] Maarten Jansen. 2012. *Noise reduction by wavelet thresholding*. Vol. 161. Springer Science & Business Media.
[50] Kiran Raj Joshi, Steven Siying Hong, and Sachin Katti. 2013. PinPoint: Localizing Interfering Radios.. In *Proceedings of Usenix NSDI*. 241–253.
[51] M Kaholokula. 2016. *Reusing Ambient Light to Recognize Hand Gestures*. Technical Report. Dartmouth College, Tech. Rep.
[52] Bryce Kellogg, Vamsi Talla, and Shyamnath Gollakota. 2014. Bringing Gesture Recognition to All Devices. In *Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation (NSDI'14)*. USENIX Association, Berkeley, CA, USA, 303–316. http://dl.acm.org/citation.cfm?id=2616448.2616477
[53] Eamonn Keogh and Abdullah Mueen. 2011. Curse of dimensionality. In *Encyclopedia of Machine Learning*. Springer, 257–258.

[54] Hyun-Seung Kim, Deok-Rae Kim, Se-Hoon Yang, Yong-Hwan Son, and Sang-Kook Han. 2013. An indoor visible light communication positioning system using a RF carrier allocation technique. *Lightwave Technology, Journal of* 31, 1 (2013), 134–144.

[55] Ye-Sheng Kuo, Pat Pannuto, Ko-Jen Hsiao, and Prabal Dutta. 2014. Luxapose: Indoor positioning with mobile phones and visible light. In *Proceedings of the 20th annual international conference on Mobile computing and networking*. ACM, 447–458.

[56] Tara Lemmey, Stanislav Vonog, and Nikolay Surin. 2011. System architecture and methods for distributed multi-sensor gesture processing. (Aug. 15 2011). US Patent App. 13/210,370.

[57] Liqun Li, Pan Hu, Chunyi Peng, Guobin Shen, and Feng Zhao. 2014. Epsilon: A visible light based positioning system. In *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*. USENIX Association, 331–343.

[58] Tianxing Li, Chuankai An, Zhao Tian, Andrew T Campbell, and Xia Zhou. 2015. Human sensing using visible light communication. In *Proceedings of the Twenty First Annual International Conference on Mobile Computing and Networking, ser. MobiCom*, Vol. 15.

[59] Tianxing Li, Chuankai An, Xinran Xiao, Andrew T Campbell, and Xia Zhou. 2015. Real-time screen-camera communication behind any scene. In *Proc. of MobiSys*.

[60] Tianxing Li, Qiang Liu, and Xia Zhou. 2016. Practical human sensing in the light. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 71–84.

[61] Yi Li. 2012. Hand gesture recognition using Kinect. In *Software Engineering and Service Science (ICSESS), 2012 IEEE 3rd International Conference on*. IEEE, 196–199.

[62] Thomas DC Little, Peter Dib, Kandarp Shah, Nick Barraford, and Beth Gallagher. 2008. Using LED lighting for ubiquitous indoor wireless networking. In *Networking and Communications, 2008. WIMOB'08. IEEE International Conference on Wireless and Mobile Computing,*. IEEE, 373–378.

[63] Cen B Liu, Bahareh Sadeghi, and Edward W Knightly. 2011. Enabling vehicular visible light communication (V2LC) networks. In *Proceedings of the Eighth ACM international workshop on Vehicular inter-networking*. ACM, 41–50.

[64] Zhihan Lu, Muhammad Sikandar Lal Khan, and Shafiq Ur Réhman. 2013. Hand and foot gesture interaction for handheld devices. In *Proceedings of the 21st ACM international conference on Multimedia*. ACM, 621–624.

[65] Zhihan Lv. 2013. Wearable smartphone: Wearable hybrid framework for hand and foot gesture interaction on smartphone. In *Computer Vision Workshops (ICCVW), 2013 IEEE International Conference on*. IEEE, 436–443.

[66] Zhihan Lv, Shengzhong Feng, Muhammad Sikandar Lal Khan, Shafiq Ur Réhman, and Haibo Li. 2014. Foot motion sensing: Augmented game interface based on foot interaction for smartphone. In *CHI'14 Extended Abstracts on Human Factors in Computing Systems*. ACM, 293–296.

[67] M. W. Marcellin, M. J. Gormish, A. Bilgin, and M. P. Boliek. 2000. An overview of JPEG-2000. In *Proceedings DCC 2000. Data Compression Conference*. 523–541. https://doi.org/10.1109/DCC.2000.838192

[68] Thomas B Moeslund, Adrian Hilton, and Volker Krüger. 2006. A survey of advances in vision-based human motion capture and analysis. *Computer vision and image understanding* 104, 2 (2006), 90–126.

[69] Pavlo Molchanov, Shalini Gupta, Kihwan Kim, and Kari Pulli. 2015. Multi-sensor system for driverâĂŹs hand-gesture recognition. In *IEEE Conference on Automatic Face and Gesture Recognition*. 1–8.

[70] Guy P Nason and Bernard W Silverman. 1995. The stationary wavelet transform and some statistical applications. In *Wavelets and statistics*. Springer, 281–299.

[71] Andrew Nelson, Jackson Schmandt, P Shyamkumar, William Wilkins, D Lachut, Nabaneeta Banerjee, Sami Rollins, J Parkerson, and Vinay Varadan. 2013. Wearable multi-sensor gesture recognition for paralysis patients. In *SENSORS, 2013 IEEE*. IEEE, 1–4.

[72] Serban Oprisescu, Christoph Rasche, and Bochao Su. 2012. Automatic static hand gesture recognition using tof cameras. In *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*. IEEE, 2748–2751.

[73] Taiwoo Park, Jinwon Lee, Inseok Hwang, Chungkuk Yoo, Lama Nachman, and Junehwa Song. 2011. E-gesture: a collaborative architecture for energy-efficient gesture recognition with hand-worn sensor and mobile devices. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*. ACM, 260–273.

[74] Samuel David Perli, Nabeel Ahmed, and Dina Katabi. 2010. Pixnet: interference-free wireless links using lcd-camera pairs. In *Proceedings of the sixteenth annual international conference on Mobile computing and networking*. ACM, 137–148.

[75] Corey Pittman, Pamela Wisniewski, Conner Brooks, and Joseph J. LaViola, Jr. 2016. Multiwave: Doppler Effect Based Gesture Recognition in Multiple Dimensions. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '16)*. ACM, New York, NY, USA, 1729–1736. https://doi.org/10.1145/2851581.2892286

[76] Thomas Plötz, Chen Chen, Nils Y Hammerla, and Gregory D Abowd. 2012. Automatic Synchronization of Wearable Sensors and Video-Cameras for Ground Truth Annotation–A Practical Approach. In *Wearable Computers (ISWC), 2012 16th International Symposium on*. IEEE, 100–103.

[77] Lorenzo Porzi, Stefano Messelodi, Carla Mara Modena, and Elisa Ricci. 2013. A smart watch-based gesture recognition system for assisting people with visual impairments. In *Proceedings of the 3rd ACM international workshop on Interactive multimedia on mobile & portable devices*. ACM, 19–24.

[78] Qifan Pu, Sidhant Gupta, Shyamnath Gollakota, and Shwetak Patel. 2013. Whole-home gesture recognition using wireless signals. In *Proceedings of ACM MobiCom*.

[79] Crisanto Quintana, Vasco Guerra, J Rufo, J Rabadan, and R Perez-Jimenez. 2013. Reading lamp-based visible light communication system for in-flight entertainment. *Consumer Electronics, IEEE Transactions on* 59, 1 (2013), 31–37.

[80] Jaime Ruiz, Yang Li, and Edward Lank. 2011. User-defined motion gestures for mobile interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 197–206.

[81] Elliot N Saba, Eric C Larson, and Shwetak N Patel. 2012. Dante vision: In-air and touch gesture sensing for natural surface interaction with combined depth and thermal cameras. In *Emerging Signal Processing Applications (ESPA), 2012 IEEE International Conference on*. IEEE, 167–170.

[82] Dragos Sbîrlea, Michael G Burke, Salvatore Guarnieri, Marco Pistoia, and Vivek Sarkar. 2013. Automatic detection of inter-application permission leaks in Android applications. *IBM Journal of Research and Development* 57, 6 (2013), 10–1.

[83] Felix Schill, Uwe R Zimmer, and Jochen Trumpf. 2004. Visible spectrum optical communication and distance sensing for underwater applications. In *Proceedings of ACRA*, Vol. 2004. 1–8.

[84] Stefan Schmid, Giorgio Corbellini, Stefan Mangold, and Thomas R Gross. 2013. LED-to-LED visible light communication networks. In *Proceedings of the fourteenth ACM international symposium on Mobile ad hoc networking and computing*. ACM, 1–10.

[85] Bernhard Scholkopf, Kah-Kay Sung, Christopher JC Burges, Federico Girosi, Partha Niyogi, Tomaso Poggio, and Vladimir Vapnik. 1997. Comparing support vector machines with Gaussian kernels to radial basis function classifiers. *IEEE transactions on Signal Processing* 45, 11 (1997), 2758–2765.

[86] J. Segen and S. Kumar. 1999. Shadow gestures: 3D hand pose estimation using a single camera. In *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, Vol. 1. 485 Vol. 1. https://doi.org/10.1109/CVPR.1999.786981

[87] Souvik Sen, Jeongkeun Lee, Kyu-Han Kim, and Paul Congdon. 2013. Avoiding multipath to revive inbuilding WiFi localization. In *Proceeding of ACM MobiSys*. 249–262.

[88] L. A. Shalabi and Z. Shaaban. 2006. Normalization as a Preprocessing Engine for Data Mining and the Approach of Preference Matrix. In *2006 International Conference on Dependability of Computer Systems*. 207–214. https://doi.org/10.1109/DEPCOS-RELCOMEX.2006.38

[89] Mark J Shensa. 1992. The discrete wavelet transform: wedding the a trous and Mallat algorithms. *IEEE Transactions on signal processing* 40, 10 (1992), 2464–2482.

[90] Jamie Shotton, Toby Sharp, Alex Kipman, Andrew Fitzgibbon, Mark Finocchio, Andrew Blake, Mat Cook, and Richard Moore. 2013. Real-time human pose recognition in parts from single depth images. *Commun. ACM* 56, 1 (2013), 116–124.

[91] Stephan Sigg, Ulf Blanke, and Gerhard Troster. 2014. The telepathic phone: Frictionless activity recognition from WiFi-RSSI. In *IEEE PerCom*. 148–155.

[92] Stephan Sigg, Markus Scholz, Shuyu Shi, Yusheng Ji, and Michael Beigl. 2014. RF-sensing of activities from non-cooperative subjects in device-free recognition systems using ambient and local signals. *IEEE Transactions on Mobile Computing* 13, 4 (2014), 907–920.

[93] Stephan Sigg, Shuyu Shi, Felix Buesching, Yusheng Ji, and Lars Wolf. 2013. Leveraging RF-channel Fluctuation for Activity Recognition: Active and Passive Systems, Continuous and RSSI-based Signal Features. In *Proceedings of ACM MoMM*.

[94] Gurashish Singh, Alexander Nelson, Ryan Robucci, Chintan Patel, and Nilanjan Banerjee. 2015. Inviz: Low-power personalized gesture recognition using wearable textile capacitive sensor arrays. In *Pervasive Computing and Communications (PerCom), 2015 IEEE International Conference on*. IEEE, 198–206.

[95] Kostas Stravoskoufos, Stelios Sotiriadis, Alexandros Preventis, and Euripides GM Petrakis. 2014. Motion sensor driven gesture recognition for future internet application development. In *Information, Intelligence, Systems and Applications, IISA 2014, The 5th International Conference on*. IEEE, 372–377.

[96] Jesus Suarez and Robin R Murphy. 2012. Hand gesture recognition with depth images: A review. In *RO-MAN, 2012 IEEE*. IEEE, 411–417.

[97] Johan AK Suykens and Joos Vandewalle. 1999. Least squares support vector machine classifiers. *Neural processing letters* 9, 3 (1999), 293–300.

[98] Stephen P Tarzia, Robert P Dick, Peter A Dinda, and Gokhan Memik. 2009. Sonar-based measurement of user presence and attention. In *Proceedings of the 11th international conference on Ubiquitous computing*. ACM, 89–92.

[99] Michael Van den Bergh, Daniel Carton, Roderick De Nijs, Nikos Mitsou, Christian Landsiedel, Kolja Kuehnlenz, Dirk Wollherr, Luc Van Gool, and Martin Buss. 2011. Real-time 3D hand gesture interaction with a robot for understanding directions from humans. In *RO-MAN, 2011 IEEE*. IEEE, 357–362.

[100] Aditya Virmani and Muhammad Shahzad. 2017. Position and Orientation Agnostic Gesture Recognition Using WiFi. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 252–264.

[101] Anran Wang, Shuai Ma, Chunming Hu, Jinpeng Huai, Chunyi Peng, and Guobin Shen. 2014. Enhancing reliability to boost the throughput over screen-camera links. In *Proceedings of the 20th annual international conference on Mobile computing and networking*. ACM, 41–52.

[102] Guanhua Wang, Yongpan Zou, Zimu Zhou, Kaishun Wu, and Lionel M. Ni. 2014. We Can Hear You with Wi-Fi!. In *Proceedings of ACM MobiCom*.

[103] Wei Wang, Alex X Liu, Muhammad Shahzad, Kang Ling, and Sanglu Lu. 2015. Understanding and Modeling of WiFi Signal Based Human Activity Recognition. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*. ACM, 65–76.

[104] Yan Wang, Jian Liu, Yingying Chen, Marco Gruteser, Jie Yang, and Hongbo Liu. 2014. E-eyes: In-home Device-free Activity Identification Using Fine-grained WiFi Signatures. In *Proceedings of ACM MobiCom*.

[105] Zachary Weinberg, Eric Y Chen, Pavithra Ramesh Jayaraman, and Collin Jackson. 2011. I still know what you visited last summer: Leaking browsing history via user interaction and side channel attacks. In *Security and Privacy (SP), 2011 IEEE Symposium on*. IEEE, 147–161.

[106] Wei Xi, Jizhong Zhao, Xiang-Yang Li, Kun Zhao, Shaojie Tang, Xue Liu, and Zhiping Jiang. 2014. Electronic Frog Eye: Counting Crowd Using WiFi. In *Proceedings of IEEE INFOCOM*.

[107] Zhi Xu, Kun Bai, and Sencun Zhu. 2012. Taplogger: Inferring user inputs on smartphone touchscreens using on-board motion sensors. In *Proceedings of the fifth ACM conference on Security and Privacy in Wireless and Mobile Networks*. ACM, 113–124.

[108] Yanbing Yang, Jie Hao, Jun Luo, and Sinno Jialin Pan. 2017. CeilingSee: Device-free occupancy inference through lighting infrastructure based LED sensing. In *Pervasive Computing and Communications (PerCom), 2017 IEEE International Conference on*. IEEE, 247–256.

[109] Zheng Yang, Zimu Zhou, and Yunhao Liu. 2013. From RSSI to CSI: Indoor localization via channel response. *Comput. Surveys* 46, 2 (2013), 25.

[110] Koji Yatani and Khai N Truong. 2012. Bodyscope: a wearable acoustic sensor for activity recognition. In *Proceedings of ACM UbiComp*. 341–350.

[111] Bobo Zeng, Guijin Wang, and Xinggang Lin. 2012. A hand gesture based interactive presentation system utilizing heterogeneous cameras. *Tsinghua Science and Technology* 17, 3 (2012), 329–336.

[112] W Zhang and M Kavehrad. 2013. Comparison of VLC-based indoor positioning techniques. In *SPIE OPTO*. International Society for Optics and Photonics, 86450M–86450M.

[113] Zimu Zhou, Zheng Yang, Chenshu Wu, Longfei Shangguan, and Yunhao Liu. 2013. Towards omnidirectional passive human detection. In *Proceedings of IEEE INFOCOM*. 3057–3065.

Received: August 2017; revised: November 2017; and accepted: January 2018