LAPD: Hidden Spy Camera Detection using Smartphone Time-of-Flight Sensors

Sriram Sami
National University of Singapore
srirams@comp.nus.edu.sg

Sean Rui Xiang Tan
National University of Singapore
seantanr@comp.nus.edu.sg

Bangjie Sun
National University of Singapore
bangjie@comp.nus.edu.sg

Jun Han*
Yonsei Univerity
jun.han@yonsei.ac.kr

ABSTRACT

Tiny hidden spy cameras concealed in sensitive locations including hotels and bathrooms are becoming a significant threat worldwide. These hidden cameras are easily purchasable and are extremely difficult to find with the naked eye due to their small form factor. The state-of-the-art solutions that aim to detect these cameras are limited as they require specialized equipment and yield low detection rates. Recent academic works propose to analyze the wireless traffic that hidden cameras generate. These proposals, however, are also limited because they assume wireless video streaming, while only being able to detect the presence of the hidden cameras, and not their locations. To overcome these limitations, we present *LAPD*, a novel hidden camera detection and localization system that leverages the time-of-flight (ToF) sensor on commodity smartphones. We implement *LAPD* as a smartphone app that emits laser signals from the ToF sensor, and use computer vision and machine learning techniques to locate the unique reflections from hidden cameras. We evaluate *LAPD* through comprehensive real-world experiments by recruiting 379 participants and observe that *LAPD* achieves an 88.9% hidden camera detection rate, while using just the naked eye yields only a 46.0% hidden camera detection rate.

CCS CONCEPTS

• **Human-centered computing** → **Mobile computing**; • **Computer systems organization** → **Sensors and actuators**.

KEYWORDS

Time-of-Flight, Hidden Camera, Smartphone

ACM Reference Format:
Sriram Sami, Sean Rui Xiang Tan, Bangjie Sun, and Jun Han. 2021. LAPD: Hidden Spy Camera Detection using Smartphone Time-of-Flight Sensors. In *The 19th ACM Conference on Embedded Networked Sensor Systems (SenSys '21), November 15–17, 2021, Coimbra, Portugal.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3485730.3485941

*Corresponding author.

SenSys '21, November 15–17, 2021, Coimbra, Portugal

ACM ISBN 978-1-4503-9097-2/21/11.
https://doi.org/10.1145/3485730.3485941

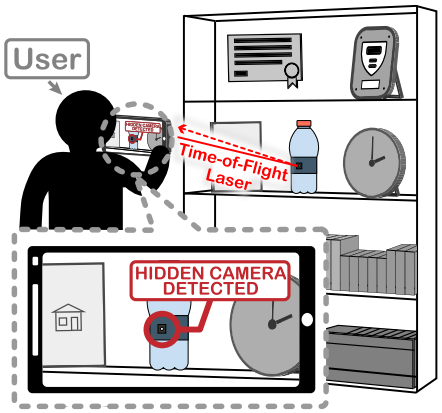


Figure 1: Figure depicts a user operating a smartphone running the *LAPD* app. *LAPD* emits laser pulses from the smartphone's time-of-flight (ToF) sensor, and detects reflections from a hidden camera concealed in a water bottle. *LAPD* annotates the location of the hidden camera on the screen.

1 INTRODUCTION

Tiny hidden spy cameras placed in sensitive locations such as hotel rooms and lavatories are increasingly a threat to individual privacy globally [31, 33, 53, 67, 81, 97]. For example, in South Korea alone, there were over 6,800 such reported cases in a single year [22, 65, 69]. These hidden cameras are difficult to detect due to their small form factors, with lens diameters as small as 1 – 2 millimeters [22, 69, 75, 87]. Consequently, the general public is left vulnerable and generally relies on the authorities to find these cameras [60].

The state-of-the-art solutions to assist authorities and the general public to detect and localize hidden cameras are commercial "hidden camera detectors" [61, 85]. A user looking through a detector's viewfinder observes bright reflections from nearby camera lenses due to the red light emitted from LEDs on the detector. While more effective than the naked eye, the users must carry such devices with them. In addition, the detectors exhibit high false positives from reflective surfaces [21].

In attempts to overcome the limitations of the state-of-the-art detectors, recent academic works propose to detect the presence of hidden cameras by analyzing the wireless traffic they generate [15, 16, 44, 48, 57, 78, 91, 92]. However, these techniques bear two

main limitations. First, they aim to detect only the *presence* of the hidden cameras, and not their exact *location*. Second, these techniques are only applicable to hidden cameras that wirelessly stream their recorded video. However, many small hidden cameras only passively record to a local memory card [3, 7, 32, 53] to save power and/or maintain a small form factor [2, 51].

In light of the above limitations, we ask the following question – can we propose a solution that only leverages an individual's **commodity smartphone** to **automatically detect** and **localize** hidden cameras? To answer this question, we propose *LAPD* (**L**aser-**A**ssisted **P**hotography **D**etection), that utilizes information from laser time-of-flight (ToF) depth sensors increasingly equipped in modern smartphones. These ToF sensors are originally intended to measure depth to aid augmented reality applications such as immersive gaming [66], contactless distance measurement [28], and more [42, 86, 95]. Figure 1 illustrates the process of using *LAPD* to detect hidden cameras. The user selects a suspicious object from the *LAPD* app on their smartphone, and is guided through a quick scan of the object. During the scan process, the ToF sensor emits laser pulses and receives the reflected light off of the object – such as the water bottle – and its surroundings. Specifically, the hidden camera embedded in the object reflects the incoming laser pulses at a higher intensity than its surroundings due to an effect called *lens-sensor retro-reflection*. This occurs when almost all light energy impacting an object is reflected *directly* back to the source (see Section 2.2). These unexpectedly high-intensity reflections from hidden cameras cause certain regions of the ToF sensor to be "saturated" and appear as black pixels. *LAPD* processes these saturated areas to automatically identify the hidden camera and its location and displays it on the user's smartphone screen.

However, detecting the hidden cameras with commodity smartphones comes with three inherent challenges that render this problem extremely difficult. First, different objects that embed the hidden cameras may cause varying reflectivity. Hence, it is critical that the smartphone be located at an *ideal distance* away from the object. If the phone is too close, the ToF sensor will oversaturate. Similarly, if the phone is moved further away, insufficient light will reach the ToF sensor and hinder the observations. To overcome this challenge, *LAPD* utilizes augmented reality to guide the user to move closer and further away from the object, calculating the ideal distance by determining the object's reflectivity at various distances.

Second, modern smartphones are significantly constrained by the equipped ToF sensor hardware – both in *spatial* (i.e., number of pixels) and *bit resolutions* (i.e., number of bits per pixel) – consequently limiting the amount of information for *LAPD*. This constraint leads to difficulties in distinguishing reflections from the hidden camera lens (as small as 1 – 2 millimeters) as opposed to other reflections from the surroundings as it is difficult to discern the shape, size, and precise intensity of the reflections. Hence, this challenge yields many false positives. To overcome this challenge, we design and implement a chain of filters including deep-learning-based filters that incorporate multi-modal information – including depth and reflection intensities – to eliminate false positives.

Third, the reflections are limited by their optical properties such that they are only observed within a constrained angle. Specifically, this is a *20° field-of-view (FoV)* cone projected from the hidden camera. Hence, this challenge exacerbates the difficulty of detecting

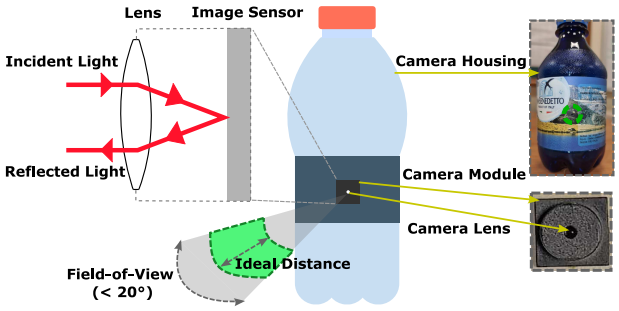


Figure 2: Figure depicts an example of a commercially available hidden camera product, namely a water bottle. We also illustrate the lens retro-reflection effect and its corresponding *ideal distance* and *maximum visible Field-of-View (FoV)*. Retro-reflection occurs when a lens-sensor combination reflects incoming light directly back towards the source.

hidden cameras as it restricts the observable angle. We exploit this limitation by implementing an FoV filter that eliminates remaining reflections – or candidate hidden cameras– which appear highly reflective outside the constrained angle.

We implement *LAPD* as a smartphone app that runs in real-time and evaluate it via comprehensive real-world experiments under varying conditions. We recruit 379 participants to compare the results with state-of-the-art commercial hidden camera detectors and using only the naked eye. From our comprehensive experiments, *LAPD* achieves an 88.9% hidden camera detection rate, compared to just using the naked eye which yields only a 46.0% hidden camera detection rate. Overall, we make the following contributions:

- We propose *LAPD* that uses time-of-flight (ToF) sensors on commodity smartphones to *automatically* detect and localize passively recording hidden cameras.
- We overcome inherent challenges with *LAPD*'s system design, using computer vision and deep-learning techniques.
- We implement *LAPD* as a fully functional end-to-end mobile application that is able to detect hidden cameras in real-time.
- We demonstrate through real-world experiments that *LAPD* is able to outperform the state-of-the-art hidden camera detectors and naked eye observations.

2 BACKGROUND AND FEASIBILITY STUDY

We now present the relevant background information and our feasibility study of *LAPD*.

2.1 Hidden Spy Cameras

There are growing security and privacy concerns over hidden spy cameras including hotels and vacation rentals [22, 31, 51, 53, 60, 65, 67, 69, 87, 97]. These hidden cameras are typically hidden in innocuous-looking items such as alarm clocks, pens, and even water bottles [2, 60, 61, 69, 85, 87]. Many hidden cameras can be easily purchased online for tens of dollars [3, 7, 32].

A hidden camera typically consists of an electronic *camera module* and *housing* that hides the module. The camera module contains

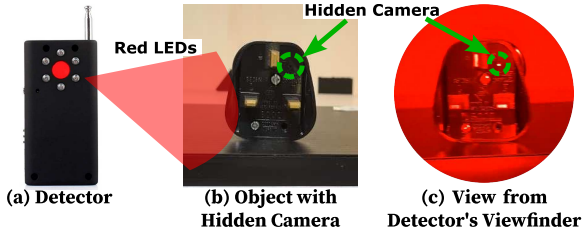


Figure 3: Figure depicts the operation of a commercial hidden camera detector device. The red-tinted image represents the view through the detector's viewfinder, indicating a bright spot at the location of the hidden camera.

two major components, namely an image sensor and a lens. Regardless of the housing's design, the camera module's lens is always exposed so that it can capture images or video [51, 65, 85]. Image sensors are mostly Charge Coupled Device (CCD) or Complementary Metal Oxide Silicon (CMOS) sensors. Pinhole lenses are mostly used in hidden cameras due to their small sizes of approximately 1 to 2 mm in diameter [22, 69, 75, 87]. Most lenses are circular as it is easier for manufacturers to grind and polish them into a circular shape (e.g., on a lathe) [96]. *LAPD* utilizes these properties, namely the small size and circularity of lenses, in Section 3.6.2 to better distinguish between hidden cameras and other reflections.

2.2 Lens-Sensor Retro-Reflection

Lens-sensor retro-reflection, also known as "cat-eye reflection", describes the effect where for certain optical systems, almost all incident light energy is reflected *directly* back to the source. Hidden cameras are one such system that causes retro-reflections. Figure 2 illustrates an example where retro-reflection occurs between a light source and a camera hidden in a water bottle. When a light ray passes through the lens in a hidden camera module and hits the image sensor, it is reflected in the parallel but opposite direction as the incident light. However, retro-reflection is only visible in a limited field-of-view (FoV). The FoV is a cone spanning approximately 20° originating from the hidden camera [54–56]. The visible distance is also limited because the intensity (I) of the reflected light decreases as the distance (d) increases due to the light energy spreading over a wider area according to the Inverse Square Law ($I \propto \frac{1}{d^2}$) [12]. In practice, the limited FoV and visible distance make hidden camera detection challenging. To address this issue, *LAPD* guides users to the optimal boundary: a region at an acceptable *distance* and *angle* from a suspicious object. *LAPD* also leverages the limited FoV to reject spurious reflections, thus turning the physical limitations of retro-reflection into a tool to remove false positives.

2.3 State-of-the-Art Hidden Camera Detectors

Commercial hidden camera detectors [4–6, 39, 80], such as the CC308+ [5] and K18 [6], are widely used as state-of-the-art solutions to assist the authorities and general public to detect hidden cameras [61, 85]. They use blinking or continuous red LEDs to generate retro-reflections from hidden cameras that a human operator attempts to interpret. Figure 3 depicts an example of a user scanning a suspicious object (e.g., a wall plug) using such a detector.

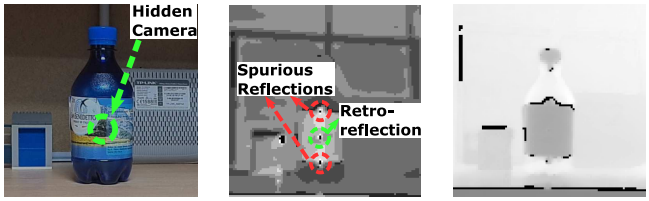


Figure 4: Figure 4(a) depicts the RGB image of a commercially available hidden camera product (i.e., a water bottle). Figure 4(b) depicts the intensity map of the scene extracted from a smartphone ToF sensor, with the retro-reflection from a hidden camera in the center, surrounded by other spurious reflections. Black areas indicate highly reflective regions. Figure 4(c) depicts the corresponding depth map.

A user looking through the detector's viewfinder observes a bright reflection from the camera's lens due to the red light emitted from LEDs on the detector. However, it is difficult and cumbersome for a user to correctly identify a hidden camera due to numerous other visible bright spots, rendering such a solution less effective in the real-world. In contrast, *LAPD* automatically removes most of the false positives, thus alleviating the burden on the user.

2.4 Smartphone Time-of-Flight Sensors and Feasibility Study

Smartphone ToF Sensors. Many modern smart devices [37, 38, 46, 77] have built-in Time-of-Flight (ToF) cameras for applications such as distance measurement, motion tracking, depth-of-field effects, and augmented reality [19, 23, 42, 86, 95]. A smartphone ToF sensor uses near-infrared (≈850 nm) light, which is invisible to the human eye, to obtain depth information. One of the methods to estimate the distance or depth (d) uses the speed of laser signal (C) and the time-delay of the reflected light (Δt) with a simple calculation ($d = \frac{C \times \Delta t}{2}$) [45]. A ToF sensor captures *depth* information by emitting light from a laser, receiving its reflections, and storing the data in a two-dimensional **depth map**. It also provides a *confidence* measure of the estimated depth to indicate its accuracy and validity [59], which is also stored in a two-dimensional confidence map. ToF sensors often store such information as 16 bits per pixel, where the first 13 bits refer to the depth data for each pixel (in mm), and the last 3 bits represent the confidence of depth estimation at the pixel. Note that nearby low confidence regions are typically associated with unexpectedly high-intensity reflections [59]. In this paper, refer to the confidence map as the **intensity map**. *LAPD* utilizes both depth and intensity maps to identify high-intensity retro-reflections from hidden cameras.

Feasibility Study. To test our hypothesis that the depth and intensity information from ToF sensors can be used to detect hidden cameras, To test our hypothesis to utilize the ToF sensors to detect hidden cameras, we conduct a feasibility study on a commercially available hidden camera in a water bottle. Figure 4(a) depicts the RGB image, Figure 4(c) the depth map, and Figure 4(b) the intensity map of the water bottle. Note that the high-intensity retro-reflection from the hidden camera is visible in the intensity map. However,

due to the limited spatial and bit resolution describing the intensity map, it is difficult to differentiate hidden camera retro-reflections from other high-intensity reflections. In fact, we indicate two *spurious reflections* on Figure 4(b) that do not originate from a hidden camera. There may be many such spurious reflections due to shiny or glossy objects. We address such challenges by applying several filters described in Section 3.

3 SYSTEM DESIGN AND IMPLEMENTATION

This section presents *LAPD*'s design and implementation details.

3.1 System Model

The goal of *LAPD* is to detect the presence and location of hidden cameras using only the information available from smartphones with ToF sensors. We design *LAPD* to satisfy the following requirements: (1) *accessibility:* operate on already-existing commodity smartphones with ToF sensors, (2) *accuracy:* correctly identify the presence and location of hidden cameras, and (3) *usability:* automatically detect these hidden cameras with minimal user intervention. We assume that the hidden camera lenses are exposed and oriented to the probable location of the victim. We also assume that *LAPD* runs on a smartphone equipped with a ToF sensor.

3.2 Design Overview

Figure 5 illustrates the components of *LAPD*'s design. *LAPD* requires accurate 3D localization as a prerequisite, which is described in *3D Localization* (§3.3). The user first interacts with the *LAPD* app in the *Suspicious Object Selection* (§3.4) phase to select an object to scan. The 3D location of the selected object is passed as input to the *Scan Distance Computation* (§3.5) phase. Here, *LAPD* guides the user to an appropriate distance from the object to maximize the probability of observing a hidden camera. The user maintains this distance during the *Object Scan* (§3.6) phase, where *LAPD* guides them to move the smartphone methodically to scan the surface of the object for hidden cameras. During this phase, *LAPD* processes and filters image frames from the ToF sensor to decide if any high-intensity reflections from the object are likely to originate from hidden cameras. If a hidden camera is detected, its location is annotated on the screen so that the user may take further action. We now present the details of *LAPD*'s design.

3.3 3D Localization

LAPD requires a 3D understanding of the space around the user to track potential hidden cameras even if the user is moving. To achieve this, we implement *LAPD* using Android's augmented reality framework (**ARCore**) [27] which includes 3D localization. As 3D localization is not our contribution, we assume the following abstractions are available from the ARCore framework: any point (u, v) on the smartphone's 2D screen can be transformed to a 3D coordinate (x, y, z) with a function $T_{2D\rightarrow 3D}(u, v)$, and the 3D coordinates of the smartphone's current position (x_p, y_p, z_p) are always accurately known. ARCore achieves this by using the smartphone's camera and inertial sensors to perform Simultaneous Localization and Mapping (SLAM) of the environment [64]. This generates a 3D representation of the environment, and tracks the smartphone's position and orientation within it.

3.4 Suspicious Object Selection

In this module, the user begins the hidden camera detection process by constraining *LAPD*'s search space. The user selects an object of interest on the 2D smartphone screen, where the goal is to take as input the selected 2D coordinates and transform them into the corresponding 3D coordinates of the object. The object's 3D coordinates are used to track the object in 3D space regardless of the phone's motion in the subsequent phases. The user first draws a 2D bounding box with a center point (u_c, v_c), around the object. This 2D center point is subsequently transformed into the 3D coordinates of the object (x_c, y_c, z_c) using the transform function $T_{2D\rightarrow 3D}(u_c, v_c)$. These 3D coordinates are passed to the next phase.

3.5 Scan Distance Computation

Recall from Section 2.2 that each object with a hidden camera may have a different *ideal distance* where a retro-reflection is visible. If the user is too close, the hidden camera reflection is obscured, and if they are too far, not enough light reaches the ToF sensor. The goal of this phase is to calculate the *ideal distance* (d_{ideal}) within the boundary that the user should stand from the object to maximize the probability of detecting any hidden cameras. This phase takes as input the 3D coordinates of the object (x_c, y_c, z_c) to scan, and guides the user's movement to compute d_{ideal}.

To illustrate the importance of computing d_{ideal}, Figure 6(a) depicts a water bottle that a user inspects with *LAPD*. Figures 6(b) to 6(d) illustrate the laser intensity values reflected from the water bottle at three distances. Black regions indicate high-intensity reflections that saturate the sensor. The hidden camera's expected location is circled in each image, and the other black areas indicate spurious reflections. The hidden camera's reflection is only clearly visible in Figure 6(c), when the user is standing near d_{ideal}. If the user is too close (Figure 6(b)) or too far (Figure 6(d)), the hidden camera reflection cannot be detected correctly. Therefore, it is crucial to compute d_{ideal} accurately.

Computing d_{ideal} is challenging as it *varies across objects*. Reflective objects return a larger fraction of the laser's intensity to the ToF sensor at a given distance than others [1]. Hence, we develop an algorithm to generically detect d_{ideal} for each object. We define $SatMax$ as the number of pixels in the largest black area in the intensity map. Observe from Figure 6 that as the user moves further from the object, $SatMax$ decreases as the sensor receives less light [12], eventually becoming 0. If a hidden camera reflection is present, for an empirically determined p, $0 < SatMax < p$ pixels. $(0 < SatMax)$ holds as a hidden camera reflection appears as at least one black pixel. $(SatMax < p)$ holds as a hidden camera reflection has a maximum size determined by its lens diameter. Assuming hidden cameras with a lens size of 1 – 2 mm, we set $p = 10$.

LAPD first guides the user to move to within 20 cm of the object to saturate the ToF sensor ($SatMax \gg p$), and then asks them to gradually move further from the object. *LAPD* uses the object's 3D coordinates (x_c, y_c, z_c) and the phone's location (x_p, y_p, z_p) to determine the distance from the object $d = \sqrt{(x_p - x_c)^2 + (y_p - y_c)^2 + (z_c - z_p)^2}$. As d increases, *LAPD* continually computes $SatMax$ using image analysis techniques (see *High-Intensity Reflection Region Extraction* (§3.6.1)). When $SatMax < p$, the algorithm returns the current distance as d_{ideal}.

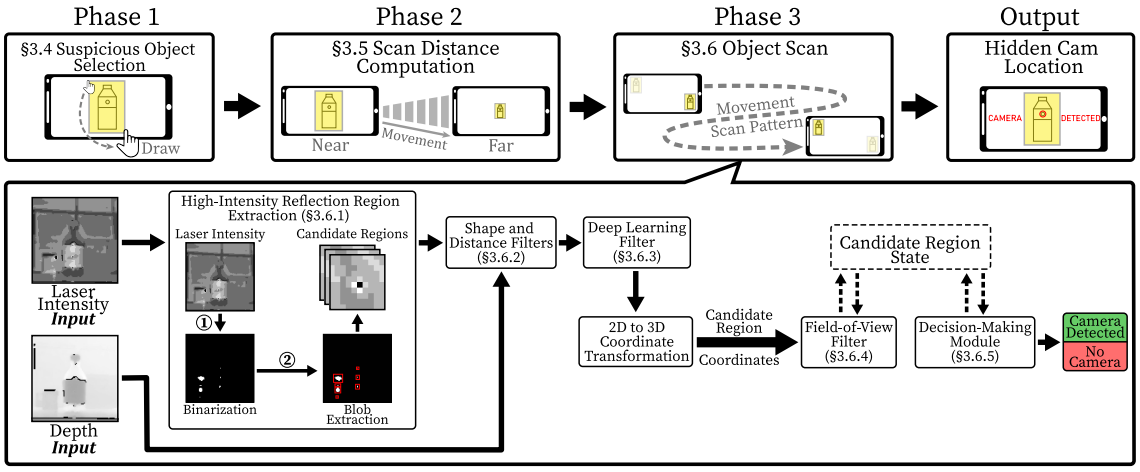
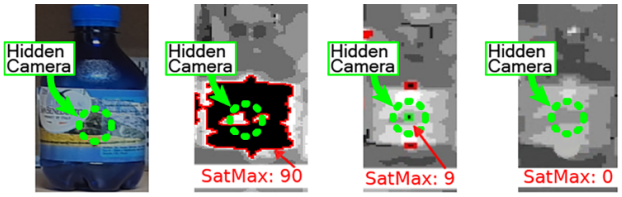


Figure 5: Figure depicts the flowchart of *LAPD*'s design. The user first uses *LAPD* to select a suspicious object. *LAPD* then guides them to determine the ideal scan distance, and scan the object for hidden cameras. During the scan, *LAPD* uses a computer vision and machine learning processing pipeline to detect reflections from hidden cameras while rejecting false positives.



(a) Reference (b) Too Close (c) Ideal Distance (d) Too Far

Figure 6: Figure depicts the effect of varying the smartphone's distance to the target object. The hidden camera's location is circled. When the phone is too close or far from the object, the hidden camera is not visible as the resulting reflections either oversaturate the sensor or are not detected.

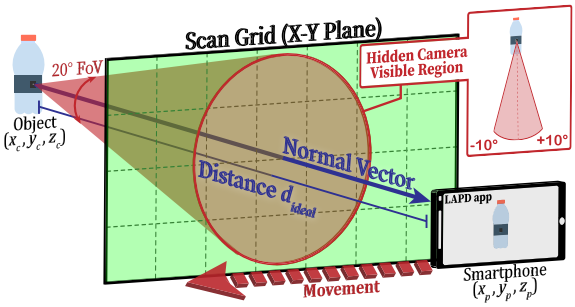


Figure 7: Figure depicts the scan grid that *LAPD* generates for the targeted water-bottle object. The user is guided to each square in the grid to increase the probability of observing a hidden camera. The scan grid is larger than the FoV of any potential cameras to trigger the FoV Filter on potential false positives. The grid is in the X-Y plane, parallel to the smartphone screen at the ideal distance away from the object, and the Z-axis lies on the normal vector of the plane.

3.6 Object Scan

The goal of this phase is to identify the location of hidden cameras within the selected object by observing it from multiple locations. It takes the object's location (x_c, y_c, z_c) and the ideal distance to stand from the object, d_{ideal}, as inputs and annotates the suspected locations of hidden cameras on the smartphone's screen. *LAPD* achieves this by instructing the user to move the smartphone in a grid pattern around the object while maintaining the distance d_{ideal} from the object. Figure 7 illustrates the scan process and coordinate system. *LAPD* generates a scan grid in the X-Y plane which is subdivided into multiple square areas. The scan grid is perpendicular to the normal vector from the object to the smartphone. *LAPD* guides the user to each square in the grid to maximize the probability of encountering hidden camera reflections. The entire scan takes approximately 30 seconds to one minute.

During the scan, *LAPD* processes in real-time a continuous stream of images from the smartphone's ToF sensor to identify any hidden cameras within the scanned object. The *High-Intensity Reflection Region Extraction* module (§3.6.1) initially identifies *all possible* high-intensity reflections (or **"blobs"**) that could correspond to a hidden camera. However, this also results in a large quantity of false positive blobs that require multiple filtering passes to remove. Subsequently, the *Shape and Distance Filters* (§3.6.2) remove false positives based on the known physical properties of hidden cameras. As many false positives have the same shape and size as hidden camera reflections, we then apply a *Deep Learning Filter* (§3.6.3) to remove these complex cases. However, we find there is still insufficient information within a *single ToF frame* to remove *all* false positives. *LAPD* then converts the remaining blobs' 2D locations to 3D coordinates to track them across frames. The *Field-of-View (FoV) Filter* (§3.6.4) compares the current and historical 3D blob locations to further remove false positives observed from an unexpectedly large field-of-view. Finally, the *Decision-Making*

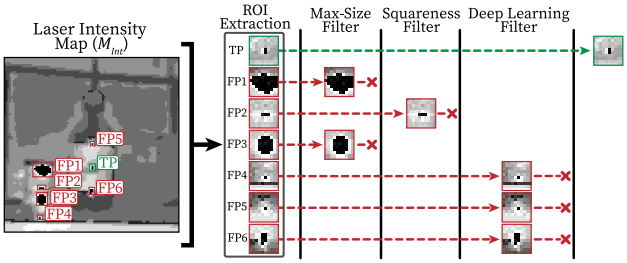


Figure 8: Figure depicts the process of *LAPD*'s *Object Scan* phase. The aim is to only select the laser intensity blob corresponding to a true hidden camera. Blobs that are too large or not square enough are removed first, and any remaining false positive blobs are removed by the deep learning filter.

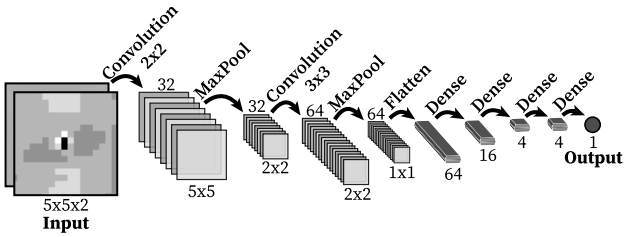


Figure 9: Figure depicts *LAPD*'s deep learning filter architecture, consisting of one input layer (stacked intensity and depth ROI), two convolutional layers with max-pooling, four dense layers, and an output node.

Module (§3.6.5) decides if a blob is a hidden camera when sufficient information is collected. We now explain each module in detail.

3.6.1 High-Intensity Reflection Region Extraction. This module aims to extract *all* high-intensity regions that may contain a hidden camera from its input of a 2D laser intensity map, M_{Int} as depicted in Figure 5. The module outputs a list of such candidate regions.

① Binarization. We first normalize the intensity map M_{Int} between 0 and 1, where 1 represents the highest (saturated) light intensity. Recall from Section 2.4 that if a hidden camera is present, we expect a high-intensity reflection from its location. Therefore, to identify the candidate regions, we first identify all saturated areas, M_{sat}, by *binarizing* the laser intensity map – i.e., for all 2D coordinates u, v in the map, $M_{sat}(u,v) = \begin{cases} \text{white} & \text{if } M_{Int}(u,v) < 1 \\ \text{black} & \text{if } M_{Int}(u,v) = 1 \end{cases}$. This produces an image where black pixels represent high intensity regions, and white pixels represent low intensity regions.

② Blob Extraction. As clusters of high-intensity pixels (blobs) may indicate a hidden camera, *LAPD* extracts them by applying a *connected components* extraction algorithm on M_{sat}. A connected component is a cluster of black pixels that are adjacent to each other by 4-way connectivity (no diagonals) [26]. We compute these components using the Scan plus Array-based Union-Find (SAUF) algorithm [93]. SAUF detects connected components by exploring each pixel and using a union-find data structure to store connectivity information. The SAUF algorithm is applied to the binarized map M_{sat} to produce a list of blobs. As many blobs may be detected for each frame due to spurious reflections, the following modules attempt to remove as many false positives as possible.

3.6.2 Shape and Distance Filters. This module takes as input all detected blobs from the previous module and reduces false positives by removing blobs that violate the expected physical properties of a hidden camera reflection. The *Shape* filters remove blobs if they are too large ("Max-Size" filter) or not circular enough ("Squareness" filter; see Section 2.1), while the *Distance* filter removes blobs that are unexpectedly close or far from the smartphone. Figure 8 illustrates an example of a list of blobs extracted from a laser intensity image of a water bottle, where shape filters remove blobs FP1, FP2, and FP3. The Max-Size filter removes blobs with an area larger than nine pixels (e.g., FP1, FP3), and the Squareness filter removes blobs with |blob width - blob height| > 1 (e.g., FP2). We set these values based on our empirical observations and prior work [34, 96].

The Distance filter considers the distance from the smartphone to the detected blob and removes blobs outside a minimum distance, d_{min}, and maximum distance, d_{max}. This is a very preliminary filter that aims to remove mostly random noise. d_{min} is where the ToF sensor is saturated regardless of the target object, and d_{max} is where there is insufficient reflected light from any hidden camera. We obtain the constants d_{min} and d_{max} through comprehensive experiments. We calculate the distance to the blob using the ToF *depth map* corresponding to the current intensity map. This distance is then used to remove blobs outside the expected range.

While the Shape and Distance filters remove some false positive blobs, in many cases, up to 50% – 80% of them remain. Many such blobs are sufficiently small, square, and at the right distance to pass the shape and distance filters (e.g., FP4, FP5, FP6). Therefore, we need additional filters to further remove false positives.

3.6.3 Deep Learning Filter. This module aims to use a *deep-learning model* to remove the remaining false positive blobs. The model takes as input the list of blobs that passed the *Shape and Distance Filters*, the intensity map M_{Int}, and the corresponding depth map, and removes blobs that are not likely to be hidden cameras. We rely on deep learning due to the limited information in each ToF frame, as more complex features are then required to extract sufficient information for an accurate filter. The model's goal is to use the values of depth and intensity pixels around each blob to learn patterns that uniquely define hidden cameras.

LAPD uses *on-device deep learning* with a Convolutional Neural Network (CNN) to remove false positive blobs. For each blob, the model outputs a value between 0 and 1, indicating its confidence that the blob represents a hidden camera. We use a novel approach of **combining** the **depth and intensity maps** in the input layer, as using only one modality yields low accuracy. Depth information alone only indicates the surface texture near the blob, while intensity information alone is susceptible to false positives. However, their combination provides mutual information to produce a discriminative model. We choose a CNN architecture as it consistently performs well in image classification tasks [54, 55]. Figure 9 depicts the architecture. For each detected blob, the input to the CNN is a $5 \times 5 \times 2$ image. *LAPD* crops a 5×5 region of interest (ROI) around the blob's center from both the intensity and depth maps, and stacks

them in the channel dimension. The images are only of size 5×5 to focus the CNN on the local features of the blob, and avoiding overfitting to the hidden camera's housing object. After the input layer, our architecture consists of two ReLu activated convolutional layers of size 2×2 with 32 filters and 3×3 with 64 filters, respectively, with max-pooling and batch-normalization layers between them. This is followed by four fully connected dense layers of sizes 64, 16, 4, and 4, respectively, with a significant dropout rate of 0.8 to control overfitting, and a final output node with sigmoid activation.

The depth ROI must be further processed to maximize the model's performance, as it can represent a large range of distances from 0 to over 5000 mm. As distance filters are already applied, the precise depth values are not as useful as the *minute surface depth variations* near the blob. For instance, these differences could indicate a hole containing a hidden camera. Hence, we subtract the *median depth value* of the ROI from every pixel to focus the model on depth variations near the reflection to improve the accuracy.

We implement our model using Keras [17] and Tensorflow Lite [25]. We train our model on a large dataset of reflections from hidden cameras in different housings totaling over 10 K manually labelled instances. We split the dataset into 80% for training and 20% for testing. The train and test sets do not share *any* hidden camera housings. This prevents the system from overfitting on the housings instead. As reflections from hidden cameras are rarer than spurious reflections, the training dataset has a class imbalance of 1 hidden camera : 6.27 non-camera samples. We correct this by applying class weightages during training with a binary crossentropy loss function and the Adam [41] optimizer. We train the model on two NVIDIA Titan V GPUs with a batch size of 32 for 400 epochs and save the model with the lowest loss. Finally, we convert our model to a Tensorflow Lite model, which is then used in our app. The model seamlessly removes blobs with a confidence score of less than 0.5, while maintaining a target of 30 frames per second. Subsequently, we convert the 2D coordinates of each remaining blob into 3D coordinates with the $T_{2D \rightarrow 3D}$ coordinate transform function from *3D Localization* (§3.3). This list of coordinates is passed to the subsequent module for further filtering.

3.6.4 Field-of-View (FoV) Filter. This module aims to further remove false positives from the remaining blobs by identifying those that are outside the maximum retro-reflection FoV of 20°. Such blobs cannot originate from a hidden camera with a 1 – 2 mm diameter lens (see Section 2.2). This is a novel approach that exploits the *limitations* of retro-reflection as *useful* information. While previous modules operate on each ToF camera frame independently, the FoV filter accumulates state information in the Candidate Region State hashmap across multiple frames. This is necessary as we need to observe a blob from at least two unique smartphone locations to calculate its observable FoV. For each observed blob at 3D location (x_c, y_c, z_c), we record a list of the smartphone's 3D locations (i.e., (x_i, y_i, z_i) at frame i) where the blob is visible. For instance, $(x_c, y_c, z_c) \rightarrow [(x_1, y_1, z_1), (x_4, y_4, z_4)]$ indicates a blob at (x_c, y_c, z_c) is visible at two different smartphone locations at frame 1 and 4. With the smartphone at a constant z distance from the object from *Scan Distance Computation* (§3.5), we can compute the *maximum acceptable xy* distance for a 20° FoV between any two observations as $2z\tan(10\,\text{deg})$. In our example, if $\sqrt{(x_1 - x_4)^2 + (y_1 - y_4)^2} > 2z\tan(10\,\text{deg})$, (x_c, y_c, z_c) is disregarded as a potential hidden camera location and removed from the hashmap. After the FoV filter is applied across the hashmap, it is passed as input to the *Decision-Making Module*. Recall from Figure 7 that the FoV Filter also defines the dimensions of our *scan grid*. For the filter to potentially remove any blobs, the user must move the smartphone *beyond* the 20° camera FoV of any hidden cameras. Hence, the scan grid dimensions are computed to be larger than the FoV region of any potential hidden cameras.

3.6.5 Decision-Making Module. This module analyzes the Candidate Region State hashmap to ultimately decide which blobs are hidden cameras. Upon scan completion, *LAPD* annotates blobs suspected to be hidden cameras. To conclude that a blob is a hidden camera and not a random error, *LAPD* observes it from n positions that are at least d_ϵ distance apart from each other, where n and d_ϵ are empirically determined values. We require n observations of each blob as noise in our processing pipeline may cause a few sparse false positives, but genuine hidden cameras are visible *multiple* times during the scan. To avoid collecting all n observations while the user is stationary, we require all observations to be d_ϵ apart from one another. Taking the state hashmap from the previous section of $(x_c, y_c, z_c) \rightarrow [(x_1, y_1, z_1), (x_4, y_4, z_4)]$, the observations (x_1, y_1, z_1) and (x_4, y_4, z_4) may only both be present if they are d_ϵ apart, i.e., $\sqrt{(x_1 - x_4)^2 + (y_1 - y_4)^2} > d_\epsilon$. In this example, if the observations are far enough apart, and only two observations are necessary ($n \leq 2$), we conclude that (x_c, y_c, z_c) contains a hidden camera. This module also provides users with more control by taking the *majority decision* across multiple scans of the same object. For example, if *LAPD* indicates a hidden camera blob in an identical position in two out of three trials, *LAPD* can report with high certainty that a hidden camera is present.

4 EVALUATION

We now evaluate *LAPD* to demonstrate its real-world performance.

4.1 Experimental Setup

Apparatus. We develop and test *LAPD* on three popular smartphones – Samsung Galaxy S20+, S20 Ultra 5G, and Note 10+ [13, 20, 89, 90]. These contain 480×640 resolution Sony IMX516 ToF sensor [72], but currently, the Android API only provides a 240 $\times$ 320 image. We develop the *LAPD* app with Android SDK v24, ARCore v1.23.0, and Tensorflow Lite v2.3.0.

Figure 10 illustrates our experimental procedures for evaluating *LAPD*. We compare *LAPD*'s performance with the existing state-of-the-art K18 detector [6], and using just the naked eye. We test both the *blinking* and *continuous* camera reflection detection modes of the K18 for completeness. These modes refer to the state of the K18's red illumination LEDs. In the *blinking* mode, users can observe the differences in a static scene under ambient and LED-lit conditions to spot the hidden cameras. Users that prefer more manual control and speed may use the *continuous* mode that powers the LEDs constantly. We evaluate *LAPD* against 30 randomly selected objects (see Section 4.3.1), where nine of them contain hidden cameras. Specifically, we modify nine objects by drilling 1.5 mm diameter holes, or using existing holes, to insert and expose the lens of an

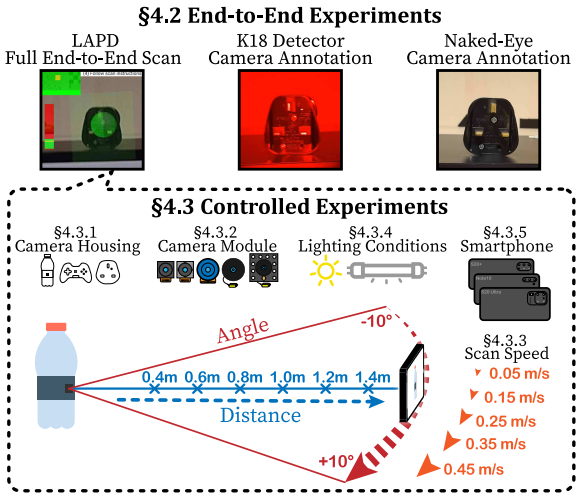


Figure 10: Figure depicts the setup of *LAPD*'s end-to-end and controlled experiments, which compares hidden camera detection rates and false positive rates for *LAPD* against other baselines. The controlled experiments vary environmental and system parameters to comprehensively evaluate *LAPD*.

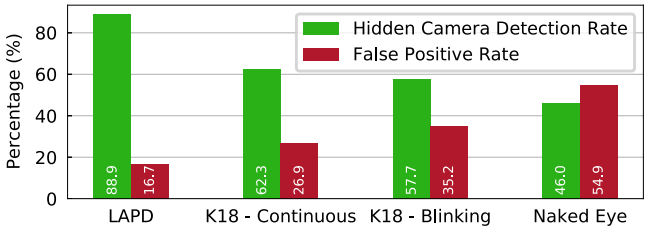


Figure 11: Overall hidden camera detection and false positive rates for *LAPD*, K18 baselines, and naked eye baseline.

OV2640 [68] camera. We choose the OV2640 as it is used extensively in surveillance systems [82], integrated into popular products [9, 94], and even DIY spy cameras [18, 74]. Note that we *do not use these 30 objects in the training set* of *LAPD*'s deep learning filter, and hence they are completely unseen test cases.

Terminology and Metrics. We define these terms and metrics to evaluate *LAPD*'s overall results. A *Hidden Camera Detection* occurs when a hidden camera's location is guessed within the acceptable ground truth region. A *False Positive* occurs when a hidden camera's location is guessed outside the ground truth region, or the guessed object does not contain a hidden camera. We note that the *Hidden Camera Detection Rate* and *False Positive Rate* may not sum to a value of *one* as the camera detection rate applies only to the nine objects containing a hidden camera.

4.2 End-to-End Experiments

Data Collection. To evaluate the overall performance of end-to-end experiments, we collect data for *LAPD* on a Samsung Galaxy S20+ and three *baseline* methods – the K18 detector in continuous mode, blinking mode, and the naked eye. We test these methods on the aforementioned 30 objects. For *LAPD*, one of this work's authors scans each object three times to obtain a total of 90 observations. We compute the detection and false positive rates using majority voting across the three trials (see Section 3.6.5). Due to COVID-19 restrictions, we recruit 379 participants from Amazon Mechanical Turk (MTurk) to evaluate the baseline methods as they are dependent on human judgment. We ask each participant to enroll in one of the three baseline methods and watch a set of pre-recorded videos (available at ***https://bit.ly/lapd-sensys***) of the 30 objects an unlimited number of times, simulating repeated trials per participant. Then, participants may choose to annotate the location of hidden cameras on images of the objects. Specifically, we use a Samsung S20+ color camera to capture videos of all 30 objects for each of the baselines, where each video inspects one object from 40 cm and 70 cm distances. For the K18 baselines, we record the videos through the K18 viewfinder and ensure that if an object contains a hidden camera, a retro-reflection must be clearly visible in the recorded video. As we are not directly testing the usability of each method, *LAPD* and the baselines are conducted in an "expert" manner for fairness. We obtain approval for these experiments through our university's Institutional Review Board.

Overall Results. We illustrate the end-to-end results in Figure 11, where we observe that *LAPD* correctly locates hidden cameras 88.9% of the time (eight out of nine objects across multiple trials), significantly outperforming the baseline methods. Even if majority voting is disabled, *LAPD* still achieves a 77.8% detection rate. On the contrary, the state-of-the-art K18 detector's continuous and blinking methods only exhibit 62.3% and 57.7% detection rates, respectively, despite ensuring that all hidden camera reflections are present in the K18 video recordings. The naked eye method further reduces the detection rate to 46.0%. As each hidden camera is recessed within its housing, there are limited visual cues for the naked eye. The K18 methods exhibit increased detection rates, likely assisted by our ensured presence of a bright retro-reflection for all objects with hidden cameras. We find that objects possessing only one obvious hole result in the highest true positive rates, as users could unambiguously observe retro-reflections from that hole while ignoring any spurious retro-reflections from the object's surface. However, we also find that such human judgment in the baseline methods is detrimental to the overall false positive rate.

LAPD yields the lowest overall false positive rate of 16.67%, while the two K18 methods and the naked eye yield 26.9%, 35.2%, and 54.9%, respectively. We can understand this effect by focusing on two otherwise *identical* objects: `wheelA` (contains a hidden camera) and `wheelB` (does not), which are illustrated in Figure 12. *LAPD* detects the hidden camera in `wheelA` while maintaining a 0% false positive rate for `wheelB`. We note that the K18 and naked eye methods have approximately 100% detection rates for `wheelA`, indicating that the retro-reflection is in fact visible for the K18 videos, and the hole in the object is obvious. However, they have false positive rates of 43.3%, 49.1%, and 98.1% for `wheelB`, respectively. Clearly, it is almost impossible to differentiate the two objects with the naked eye. The K18 methods improve differentiability, but spurious reflections cause users to incorrectly suspect hidden cameras in locations where none exist. Overall, *LAPD*'s processing pipeline and automated decision-making reduce false positives, while its user guidance ensures that most hidden camera reflections are captured.

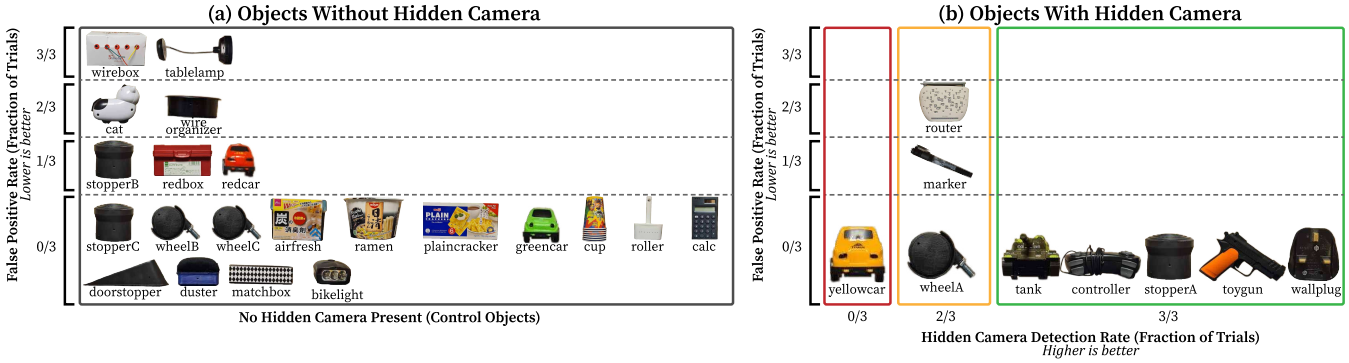


Figure 12: Figure depicts 30 objects tested. (a) depicts 21 objects without the hidden cameras. y-axis depicts the false positive rate (where lower is better). (b) depicts nine objects with the hidden cameras. Similarly, y-axis depicts the false positive rate, while x-axis depicts the hidden camera detection rate (where higher is better).

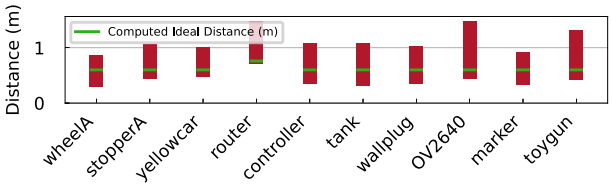
4.3 Controlled Experiments

We present the results of our comprehensive evaluation of *LAPD* under varying experimental and environmental conditions. These experiments are conducted using the same set of 30 objects and test environment as *LAPD* in the End-to-End experiments.

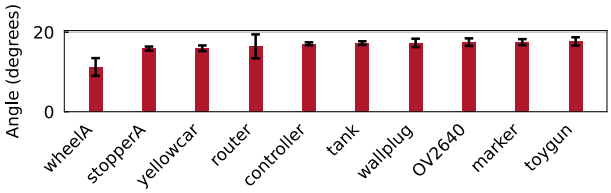
4.3.1 Varying Camera Housings. We explore the effect of varying hidden camera housings on *LAPD*'s performance.

Effect on Detection and False Positive Rates. Figure 12 illustrates the 30 objects selected for our evaluation and their detection rates with *LAPD* across three trials. Specifically, Figure 12(a) depicts 21 objects *without* the hidden cameras, while Figure 12(b) depicts nine objects that each *embed* a hidden camera. Certain objects have A/B/C suffixes – they represent identical objects where only the "A" object has a hidden camera. For both the figures, the y-axis indicates the false positive rate for each object. For Figure 12(b), the x-axis indicates the hidden camera detection rate. For instance, an object with a hidden camera detection rate of 2/3 indicates that *LAPD* detected the hidden camera in two out of three trials.

We obtain the previously discussed hidden camera detection rate of 88.9% by observing that eight out of nine objects with hidden cameras are detected in 2/3 (majority) trials or more. Similarly, the false positive rate of 16.7% is derived from the five out of 30 objects that cause false positives in over 2/3 trials. The `yellowcar` object is the only outlier in the hidden camera detection rates, as its hidden camera is acutely angled upwards, causing its visible FoV to be higher than the generated scan pattern. However, such a setting is impractical in the real world, as a hidden camera at such an angle is ineffective at capturing portions of the surrounding environment. We also note that the `router` generates a false positive reflection often due to its white and glossy surface with over 100 possible holes to hide a hidden camera. Even in this case, *LAPD* only generates one additional false positive. We observe also that most objects without hidden cameras do not cause *any* false positive reflections, even after repeated scans. Certain glossy objects with holes tend to produce false positives, such as `wirebox` and `tablelamp`. However, this is not always the case, as the white glossy `roller` did not result in any false positive reflections. Overall, we find that *LAPD* can detect hidden cameras across varying object types with high accuracy while yielding a low false positive rate.



(a) Varying target object on its *min* and *max detectable distance* along with *LAPD's computed scan distance*



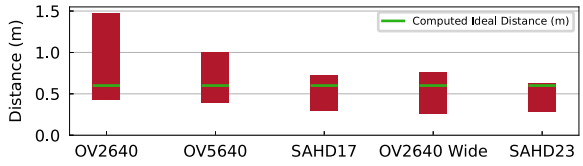
(b) Varying target object on its *max detectable angle*

Figure 13: Effects of varying camera housings.

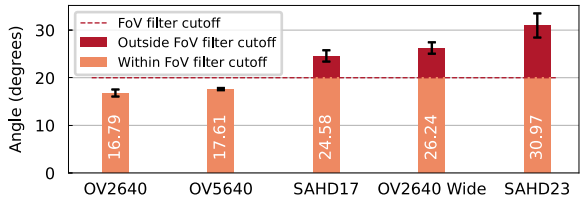
Effect on Ideal Distance and Maximum FoV. Recall that all objects with hidden cameras have an ideal distance and maximum FoV where the hidden camera can be observed. We first evaluate the effect of varying hidden camera housings on the distances where they can be detected, as depicted in Figure 13(a). As a reference, the figure indicates that the exposed OV2640 camera is visible from approximately 0.45 m to 1.5 m. As housings occlude part of the camera's FoV, all other objects are visible over a shorter total range.

Recall from *Object Scan* (§3.6) that *LAPD* computes the ideal scan distance for each object. Figure 13(a) illustrates that the computed scan distance for each object is *always* within its minimum and maximum visible distances. These results also prove the importance of computing the ideal scan distance for each object. The `router` is a glossy object that causes the ToF sensor to saturate excessively until the user stands at least 0.71 m away. The figure indicates that *LAPD* correctly instructs the user to stand 0.78 m away, ensuring that they detect the hidden camera.

(a) Different hidden camera modules



(b) Varying hidden camera module on its *min* and *max detectable distance*, and *LAPD*'s *computed scan distance*



(c) Varying hidden camera module on its *max detectable angle*

Figure 14: Effects of varying the hidden camera module.

We evaluate the effect of varying hidden camera housings on the camera module's detectable FoV. Figure 13(b) indicates that the exposed OV2640 camera has an average FoV of 17.5°. The FoVs of the other objects are marginally less than 17.5° due to occlusion caused by the camera housing. The `wheelA` object is an outlier with only an 11.2° FoV; a portion of the camera lens is occluded due to a misalignment of the lens and the drilled hole. Notably, all cases have an FoV less than of 20°, ensuring that the FoV filter module will not treat reflections from these objects as false positives.

4.3.2 Varying Hidden Camera Modules. We evaluate five hidden camera modules, illustrated in Figure 14(a), to understand the differences in their ideal distances and maximum FoVs. We conduct a controlled experiment that only exposes the lens of each camera during testing, while the rest is covered by matte black paper. Two camera modules (i.e., OV2640 and OV5640) satisfy assumptions of our system model with lens diameters of 1.5 mm, while the other cameras have larger lens diameters of 6, 8, and 10 mm, respectively, and are included for completeness. Figure 14(b) indicates that all five camera modules have different detectable distances, and *LAPD* guides the user to an appropriate distance for each. *LAPD*'s deep-learning filter is trained on the OV2640 camera due to its ubiquity [9, 18, 74, 82, 94], but it has *not encountered reflections from the other cameras* prior to this experiment. *LAPD* is capable of detecting and classifying these unseen reflections as hidden cameras.

Figure 14(c) indicates that the OV2640 and OV5640 camera modules have similar retro-reflection FoVs, and are accurately detected by *LAPD*. Conversely, the remaining cameras with larger diameter lenses exceed the 20° FoV and may be ignored by the filter. However, *LAPD*'s FoV acceptance angle can be dynamically increased to target other hidden cameras of interest.

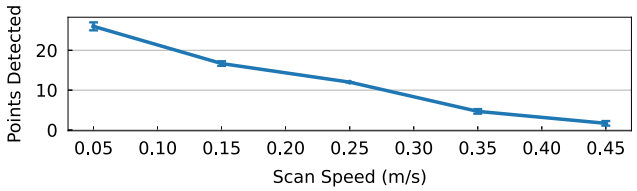
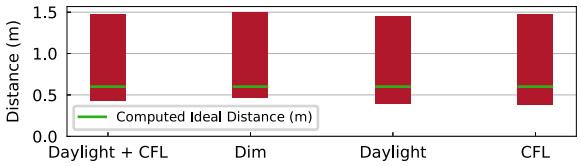
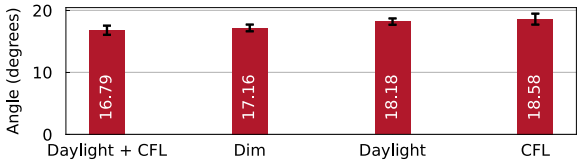


Figure 15: Figure depicts the effect of varying user's scan speed on the number of hidden camera detections.



(a) Varying ambient light conditions on *min* and *max detectable distance* of an OV2640 camera



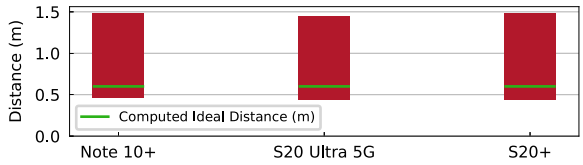
(b) Varying ambient light conditions on *maximum detectable angle* of an OV2640 camera

Figure 16: Effects when varying ambient lighting conditions.

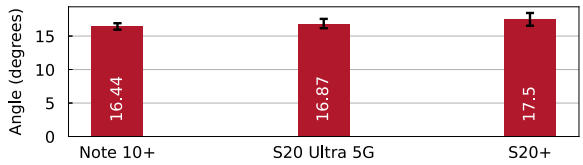
4.3.3 Varying Scan Speed. Recall from the *Object Scan* phase that *LAPD* guides the user to various positions to observe the object. We now present the effect of varying the user's scan speed on the average number of detected hidden camera reflections in Figure 15. We sweep the smartphone left-to-right three times across an OV2640 camera without a housing, initiating and completing the scan outside of its detectable FoV. We choose 0.05 m/s as the lowest average scan speed to approximate the pace of a methodical scan, whereas a 0.45 m/s speed approximates a careless scan. These speeds are derived from the phone's inertial measurement sensors.

We observe from Figure 15 that the number of hidden camera detections reduces approximately *linearly* as the scan speed increases. In our experiments, we empirically set six detections from one location as our decision-making threshold (see Section 3.6.5) for a hidden camera (i.e., $n = 6$). At scan speeds of 0.35 m/s and above, we receive fewer than six detections per sweep. However, a full object scan involves on average two to five sweeps depending on the object's size. Hence, it is probable that even at a scan speed of 0.45 m/s, *LAPD* will continue to detect hidden cameras.

4.3.4 Varying Lighting Conditions. As ToF sensors rely on receiving light in the infrared spectrum, we evaluate *LAPD*'s robustness to varying ambient lighting conditions in Figures 16(a) and 16(b). We collect detectable distance and angle data from the OV2640 camera without a housing, while using *LAPD* under different combinations of natural daylight and compact fluorescent light (CFL), as both light sources are known to contain infrared components. We find that *LAPD*'s performance is unaffected by ambient conditions, with



(a) Varying smartphone types on *min* and *max detectable distance* of an OV2640 camera



(b) Varying smartphone types on *max detectable angle* of an OV2640 camera

Figure 17: Effects when varying the smartphone types.

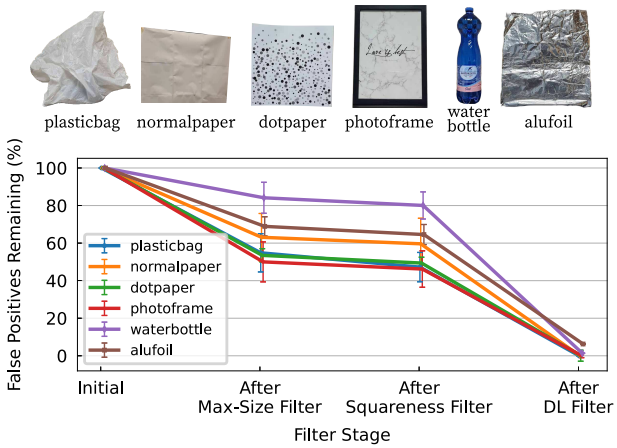


Figure 18: Figure depicts *LAPD*'s performance on rejecting false positives for the filters presented in Section 3.6. We select six objects that generate a large quantity of spurious reflections due to their glossy and brightly-colored texture.

all values within the error bars. This is likely due to the narrow-band light filter in the ToF sensor. We observe that even pointing two identical phones' ToF sensors at each other does not cause any cross-interference. We conclude that the ToF sensors and therefore *LAPD* is robust to changes in ambient lighting.

4.3.5 Varying Smartphone Types. Figures 17(a) and 17(b) illustrate the effect of varying the smartphone running *LAPD* on the detectable distance and angle data from the OV2640 camera without a housing. We test the Samsung Galaxy S20+, S20 Ultra 5G, and Note 10+ smartphones. Similar to varying lighting conditions, the choice of smartphone has no noticeable effect on the detectable distances or angles, indicating that *LAPD* is robust to such variations.

4.4 False Positive Rejection Performance

We now evaluate *LAPD*'s ability to reject false positive blobs. Recall that *LAPD*'s *Shape Filters* (§3.6.2) and *Deep Learning Filter* (§3.6.3)

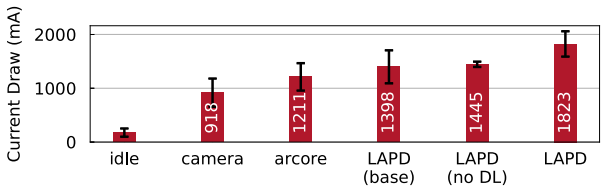


Figure 19: Figure depicts the cumulative impact of *LAPD*'s components on overall current draw.

aim to remove the significant number of false positive blobs that appear in each frame, as each blob forces the user to spend time investigating it. False positives may also reduce the user's willingness to believe true hidden camera detections.

Figure 18 illustrates the ability of our Max-Size, Squareness, and Deep Learning Filters to remove false positive *blobs*. We select six additional objects without hidden cameras that especially challenge LAPD by generating a large quantity of tiny spurious reflections. In contrast, glass or mirrors tend to generate one large reflection that is trivially filtered out. These six objects tend to be white and/or glossy (generating intense reflections), and possess many creases (generating multiple small reflections). *LAPD* is also *not trained* on these objects. For each object, we find three viewing angles that maximize the initial number of false positive blobs, and collect blob filtering statistics at each angle over ten trials. The objects each generate over 50 false positives *per frame* on average, and certain objects like `plasticbag` generate over 100 per frame. We then compute the average reduction in false positive blobs for each object and each filters' contribution.

The figure demonstrates that *LAPD* removes 98.38% of false positive blobs on average across the objects. The `plasticbag` and `normalpaper` objects have a 100% false positive removal rate. This rate is > 99.9% for the `dotpaper` and `photoframe`. Aluminium foil (`alufoil`) is challenging; it generates over 70 false positive reflections per frame, yet *LAPD* can remove 93.73% of these false positives. We note that the Max-Size and Squareness Shape Filters remove 20 to 50% of the false positives, forming an essential part of the filtering pipeline. We also observe that the deep-learning filter is consistently responsible for removing > 50% of false positives, and up to 78.68% for the `waterbottle`. As the deep-learning filter has not trained on any of these objects, this suggests that it is sufficiently generalizable to remove false positives in varying conditions.

4.5 Analysis of Energy Usage

Figure 19 illustrates the current drawn by *LAPD* and its sub-components while operating. We calculate the current drawn when: the phone is idle with the screen on (`idle`), the RGB camera is turned on (`camera`), ARCore processing of the RGB feed is turned on (`arcore`), LAPD and the ToF sensor are active (`LAPD (base)`), LAPD is processing false positives without deep-learning (`LAPD (no DL)`), and LAPD is operating its full pipeline (`LAPD`). We compute these statistics from 15 to 30-minute battery consumption traces using Android Battery Historian [30] on a Samsung S20+. We observe that the RGB camera feed contributes to around 40% of the overall current draw (741 mA) of *LAPD*, whereas the ToF sensor accounts only for 187 mA. The deep-learning filter requires

twice as much current as the ToF sensor (378 mA) when processing false positive reflections. Further techniques to optimize the model for mobile use such as quantization and pruning are likely to reduce this further. However, *LAPD* can currently run on a Samsung S20+ with a 4500 mAh battery for $\frac{4500}{1823} \approx 2.47$ hours. This allows approximately 150 objects to be scanned at 1 minute per scan.

5 DISCUSSION

We discuss the deployment considerations for *LAPD* and limitations.

5.1 Deployment Considerations

Crowdsourcing *LAPD*. To improve *LAPD*'s performance, we envision crowdsourcing to engage multiple users. This could be realized using multi-user augmented reality (AR) features [8, 29], where users share information across AR sessions. These features could also enable *progressive refinement* of hidden camera locations over time, allowing authorities to intervene at hidden camera "hot spots".

Automating *LAPD*. We envision that *LAPD* could be deployed on increasingly common robotic platforms, which could automatically scan for hidden cameras during their scheduled tasks. For instance, cleaning robots in hotels [10, 52, 79] could scan for hidden cameras in guests' rooms as part of their daily routine.

Smartphone's Flashlight and RGB Cameras. We envision extending *LAPD* to also use flashlights on smartphones to induce *visible-light* retro-reflections and capture them with the smartphone RGB camera. Our preliminary experiments indicate that we may be able to fuse both types of retro-reflections for increased accuracy.

Latency. Although *LAPD* operates at a constant 30 fps on the smartphones tested, these are typically high-end flagship smartphones as ToF sensors are an emerging technology. Further latency analysis of *LAPD*'s processing pipeline could confirm its suitability for future inexpensive smartphones containing ToF sensors.

Complementing WiFi-based Solutions. Many existing works propose to detect the *presence* of hidden cameras by analyzing wireless traffic, but are unable to *localize* them [15, 16, 44, 48, 57, 78, 91, 92]. *LAPD* could rely on these techniques to initially identify suspicious regions and then localize the hidden cameras.

5.2 Limitations

Availability of ToF Sensors. While many smartphones contain ToF sensors [38, 77, 86], they are not yet universal. However, ToF sensors on smartphones are already used for diverse applications including authentication, camera effects, and augmented reality [19, 23, 42, 77, 86]. The ToF market is projected to grow significantly in the next five years [63], with ToF sensors expected to be ubiquitous.

Inaccurate 3D Localization. The accuracy of *LAPD*'s FoV Filter and Decision-Making Module depends on the accuracy of the underlying 3D localization system. For instance, errors in 3D localization could affect the 20° FoV filter calculation, resulting in a false positive. The expected improvement of smartphone-based 3D localization systems would inherently further improve *LAPD*'s accuracy.

Per-Object Scan Duration. *LAPD* requires users to scan only one object at a time, as current ToF sensor limitations cause objects of different reflectivities to have distinct ideal scan distances (see Section 3.5). This leads to a moderately lengthy scan duration when multiple objects are inspected. However, as ToF sensors become increasingly sensitive, *LAPD* can be improved to scan larger areas with increased accuracy, as retro-reflections could be detected from further away at a higher resolution.

Hidden Camera Placement. LAPD's current scan pattern approximates all camera housings as flat planes where any hidden cameras face the user. However, we could infer the shape of camera housings to generate accurate scan patterns, as ToF sensors can create accurate 3D models of nearby objects [11, 70].

6 RELATED WORK

We now present related work on hidden camera detection.

WiFi-based Solutions. Multiple prior works propose to detect the hidden cameras by correlating their wireless traffic patterns with nearby human activities or changes in environmental conditions [15, 16, 44, 48, 57, 78, 91, 92]. Additionally, mmWave sensing may also be used to detect electronic devices transmitting radio frequency (RF) signals [49]. While these techniques are promising, they have significant limitations as they (1) require the hidden cameras to transmit data wirelessly – while a significant number of hidden cameras passively record to a local memory card (e.g., a MicroSD Card) – and are (2) unable to localize them.

Camera Detection using Retro-reflections. Another body of research utilizes pulsed laser beams to detect retro-reflections from electronic devices, with image processing techniques to remove background noise [34, 47, 71, 73, 76, 83, 96]. Several recent works use high-precision laser arrays to recognize unique patterns of retro-reflection from hidden cameras, and use machine learning to differentiate them from other reflections [54–56]. These methods require expensive laboratory equipment or specific devices with high-precision sensors which are unavailable to the general public. In contrast, *LAPD* only utilizes a commodity smartphone to detect hidden cameras in real-world scenarios.

Depth-based Sensing. ToF sensors, such as those used in *LAPD*, are primarily used to determine distance (or *depth*) information. Generally, depth sensors present numerous novel sensing opportunities, such as depth-based object detection [58, 62, 84], pose/gesture/action recognition [40, 43, 50, 88], and RGB-depth-fused environment mapping [14, 24, 35, 36].

7 CONCLUSION

We propose *LAPD*, a novel hidden spy camera detection system utilizing time-of-flight (ToF) sensors on commodity smartphones. We design and implement *LAPD* as a smartphone app that emits laser pulses from the ToF sensor to automatically detect and localize hidden cameras in real-time by identifying unique high-intensity reflections from hidden camera lenses. To evaluate its feasibility, we conduct a set of comprehensive real-world experiments under varying conditions. Specifically, we recruit 379 participants and compare the results with baselines, including the state-of-the-art hidden camera detectors and the naked eye. We observe that *LAPD* achieves an 88.9% hidden camera detection rate, while the naked eye experiment yields only a 46.0% hidden camera detection rate.

ACKNOWLEDGMENTS

This research was partially supported by grants from the Singapore Ministry of Education Academic Research Fund Tier 1 (R-252-000-A26-133 and R-252-000-B48-114).

REFERENCES

[1] 3M. 2004. Reflectivity. https://multimedia.3m.com/mws/media/295767O/reflectivity-flyer.pdf

[2] Shaffiq Alkhatib. 2020. Man Used Secret Cameras Disguised as Pens to Record Nude Videos of Female Friends 'for Fun'. *The Straits Times* (Nov. 2020).

[3] Amazon. 2021. *Hidden Camera - Mini Spy Camera 1080p HD Spy Cam Pen 2.5 Hours Video Taking Battery Life with 32GB Memory for Business Conference and Security.* https://bit.ly/3z3ZOWf.

[4] Amazon. 2021. *Hidden Camera Detector - Anti Spy Finder with The Largest Infrared Viewer and Brightest LEDs. Travel Size Pro Security and Privacy for AirBnB, Hotels, Bathrooms. Searches 2X Faster Than Other Models.* https://bit.ly/3poPaF83.

[5] Amazon. 2021. *Hidden Camera Detector, RF Detector & Camera Finder, Bug Detector, Counter Surveillance ,Anti Spy Camera Detectors with Compass,Locates Hidden Device in Office,Hotel Rooms,Airbnb Excursions,Bathrooms.* https://bit.ly/3z4l6Di.

[6] Amazon. 2021. *Innoo Tech Anti Spy Detector & Camera Finder RF Signal Detector GPS Bug Detector Hidden Camera Detector for GSM Tracking Device GPS Radar Radio Frequency Detector.* https://bit.ly/2Sa33L2.

[7] Amazon. 2021. *Mini Spy Camera Wireless Hidden, MHDYT Full HD 1080P Portable Small Covert Home Nanny Cam with Motion Detection and Night Vision, Indoor/Outdoor Micro Security Surveillance Hidden Camera.* https://bit.ly/3pvXdQE.

[8] Apple. 2021. Creating a Multiuser AR Experience | Apple Developer Documentation. https://developer.apple.com/documentation/arkit/creating_a_multiuser_ar_experience

[9] Arducam. 2015. Amazon.Com: Arducam Mini Module Camera Shield with OV2640 2 Megapixels Lens for Arduino UNO Mega2560 Board and Raspberry Pi Pico: Industrial & Scientific. https://www.amazon.com/Arducam-Module-Megapixels-Arduino-Mega2560/dp/B012UXNDOY/ref=sr_1_8?dchild=1&keywords=OV2640&qid=1622714789&sr=8-8

[10] Avidbots Corp. 2021. Avidbots | Robotic Cleaning Machine - Auto Floor Scrubber. https://www.avidbots.com/

[11] Bellus3D. 2021. FaceApp. http://www.bellus3d.com/faceapp

[12] Jeffrey R.S. Brownson. 2014. Chapter 03 - Laws of Light. In *Solar Energy Conversion Systems*, Jeffrey R.S. Brownson (Ed.). Academic Press, Boston, 41–66. https://doi.org/10.1016/B978-0-12-397021-3.00003-X

[13] Business Wire. 2021. Strategy Analytics: Samsung Galaxy S20+ 5G is World's No.1 5G Smartphone Model by Revenue in H1 2020. https://www.businesswire.com/news/home/20201027006174/en/Strategy-Analytics-Samsung-Galaxy-S20-5G-is-Worlds-No.1-5G-Smartphone-Model-by-Revenue-in-H1-2020

[14] Kang Chen, Yu-Kun Lai, and Shi-Min Hu. 2015. 3D indoor scene modeling from RGB-D data: a survey. *Computational Visual Media* 1, 4 (2015), 267–278.

[15] Yushi Cheng, Xiaoyu Ji, Tianyang Lu, and Wenyuan Xu. 2018. DeWiCam: Detecting Hidden Wireless Cameras via Smartphones. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security (ASIACCS '18)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3196494.3196509

[16] Y. Cheng, Xiaoyu Ji, Tianyang Lu, and W. Xu. 2020. On Detecting Hidden Wireless Cameras: A Traffic Pattern-based Approach. *IEEE Transactions on Mobile Computing* 19 (2020), 907–921.

[17] François Chollet et al. 2015. Keras. https://keras.io.

[18] Christopher Mendez Martinez. 2020. A Super Easy Security Camera With the ESP32 CAM - Hackster.Io. https://www.hackster.io/mcmchris/a-super-easy-security-camera-with-the-esp32-cam-02ac43

[19] Peter Clarke. 2019. 3D ToF Sensor Aims at Mobile Authentication. https://www.eenewsanalog.com/news/3d-tof-sensor-aims-mobile-authentication

[20] Stephanie Condon. 2021. Samsung Galaxy S20 Plus dominates 5G phone market in the US. https://www.zdnet.com/article/samsung-galaxy-s20-plus-dominates-5g-phone-market-in-the-us/

[21] Derek Ward. 2019. Hidden Camera Detectors Tested. https://ipvm.com/reports/hidden-cameras-finder

[22] Stacey Dooley. 2020. Stacey Dooley Investigates: 'My Daughter Was Tormented by Spycam Sex Crime'. *BBC Three* (April 2020).

[23] Eliana Brzozowski, Stefan Vogel, and Jochen Penne. 2020. State-of-the-Art Photography and Immersive AR/MR Experiences: REAL3T 3D ToF Imager with Longest Range in the Market. https://www.infineon.com/dgdl/Infineon-eeTimes_Europe_ToF_coverstory-Article-v01_00-EN.pdf?fileId=5546d46276c4f5350176f64937681034

[24] Nikolas Engelhard, Felix Endres, Jürgen Hess, Jürgen Sturm, and Wolfram Burgard. 2011. Real-time 3D visual SLAM with a hand-held RGB-D camera. In *Proc. of the RGB-D Workshop on 3D Perception in Robotics at the European Robotics Forum, Vasteras, Sweden*, Vol. 180. 1–15.

[25] Martín Abadi et al. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. https://www.tensorflow.org/ Software available from tensorflow.org.

[26] R Fisher, S Perkins, A Walker, and E Wolfart. 2003. Glossary - Pixel Connectivity. https://homepages.inf.ed.ac.uk/rbf/HIPR2/connect.htm

[27] Google. 2018. ARCore. https://developers.google.com/ar

[28] Google. 2020. Measure - Apps on Google Play. https://play.google.com/store/apps/details?id=com.google.tango.measure&hl=en&gl=US

[29] Google. 2021. Cloud Anchors Overview for Android | ARCore. https://developers.google.com/ar/develop/java/cloud-anchors/overview-android

[30] Google. 2021. Google/Battery-Historian. Google. https://github.com/google/battery-historian

[31] Amy Gunia. 2019. Tourist Charged After Spycam Found in Bondi Beach Hostel. *Time* (March 2019).

[32] B & H. 2021. *Hidden & Spy Cameras.* https://www.bhphotovideo.com/c/buy/Hidden-Cameras/ci/18682/N/4045021092.

[33] Emily Haavik. 2019. 'Recording Devices' Reported by Students in Some Minneapolis Hyatt Regency Rooms. *kare11.com* (Dec. 2019).

[34] Sifeng He, Y. Meng, and M. Gong. 2018. Active laser detection system for recognizing surveillance devices. *Optics Communications* 426 (2018), 313–324.

[35] Peter Henry, Michael Krainin, Evan Herbst, Xiaofeng Ren, and Dieter Fox. 2014. RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments. In *Experimental robotics*. Springer, 477–491.

[36] Gibson Hu, Shoudong Huang, Liang Zhao, Alen Alempijevic, and Gamini Dissanayake. 2012. A robust rgb-d slam algorithm. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 1714–1719.

[37] Huawei. 2019. HUAWEI P30 Pro Specifications. https://consumer.huawei.com/sg/phones/p30-pro/specs/

[38] Tech Insights. 2021. *Sony d-ToF Sensor found in Apple's new LiDAR camera.* https://www.techinsights.com/blog/sony-d-tof-sensor-found-apples-new-lidar-camera.

[39] Adam Juniper. 2021. *Best hidden camera detector in 2021: hunt out bugs, trackers and spy cams.* https://www.digitalcameraworld.com/buying-guides/best-hidden-camera-detector.

[40] Cem Keskin, Furkan Kıraç, Yunus Emre Kara, and Lale Akarun. 2013. Real time hand pose estimation using depth sensors. In *Consumer depth cameras for computer vision*. Springer, 119–137.

[41] Diederik P. Kingma and Jimmy Ba. 2017. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]* (Jan. 2017). arXiv:1412.6980 [cs]

[42] Denis Koshelev. 2021. *What is a ToF camera and how modern smartphones use it.* https://root-nation.com/en/articles-en/tech-en/en-what-is-a-tof-camera/.

[43] Alexey Kurakin, Zhengyou Zhang, and Zicheng Liu. 2012. A real time system for dynamic hand gesture recognition with a depth sensor. In *2012 Proceedings of the 20th European signal processing conference (EUSIPCO)*. IEEE, 1975–1979.

[44] Brent Lagesse, K. Wu, Jaynie Shorb, and Zealous Zhu. 2018. Automated Hidden Sensor Detection in Sensor-Rich Spaces. *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)* (2018), 433–435.

[45] Larry Li. 2014. *Time-of-Flight Camera – An Introduction.* Technical Report. Texas Instruments. https://www.ti.com/lit/wp/sloa190b/sloa190b.pdf

[46] LG USA. 2020. LG V60 ThinQ™ 5G | T-Mobile (LMV600TMLATMOCB). https://www.lg.com/us/cell-phones/lg-lmv600tmlatmocb-tmobile-v60-thinq-5g

[47] L. Li, J. Ren, and Xingbin Wang. 2015. Fast cat-eye effect target recognition based on saliency extraction. *Optics Communications* 350 (2015), 33–39.

[48] Zhijing Li, Zhujun Xiao, Yanzi Zhu, Irene Pattarachanyakul, B. Zhao, and H. Zheng. 2018. Adversarial Localization against Wireless Cameras. *Proceedings of the 19th International Workshop on Mobile Computing Systems & Applications* (2018).

[49] Zhengxiong Li, Zhuolin Yang, C. Song, C. Li, Zhengyu Peng, and W. Xu. 2018. E-Eye: Hidden Electronics Recognition through mmWave Nonlinear Effects. *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems* (2018).

[50] Bin Liang and Lihong Zheng. 2015. A survey on human action recognition using depth sensors. In *2015 International conference on digital image computing: techniques and applications (DICTA)*. IEEE, 1–8.

[51] Jeong-Yeo Lim. 2017. Hidden Camera at Airbnb House in Japan Catches Korean Couple off Guard. *The Jakarta Post* (July 2017).

[52] LionsBot. 2018. LionsBot. https://www.lionsbot.com/about/

[53] Steve Litz. 2019. Tourist Says He Found Hidden Cameras in Miami Airbnb. *NBC 6 South Florida* (Jan. 2019).

[54] Chongxing Liu, Changming Zhao, Haiyang Zhang, Zilong Zhang, Z. Cai, and Zhi peng Li. 2019. Spectrum classification using convolutional neural networks for a mini-camera detection system. *Applied optics* 58 33 (2019), 9230–9239.

[55] Chongxing Liu, Changming Zhao, H. Zhang, Zilong Zhang, S. Gao, and Yunshi Wang. 2019. Analysis of Mini-Camera's Cat-Eye Retro-Reflection for Characterization of Diffraction Rings and Arrayed Spots. *IEEE Photonics Journal* 11 (2019), 1–12.

[56] C. Liu, Changming Zhao, H. Zhang, Zilong Zhang, Yanwang Zhai, and Yali Zhang. 2019. Design of an Active Laser Mini-Camera Detection System Using CNN. *IEEE Photonics Journal* 11 (2019), 1–12.

[57] Tian Liu, Ziyu Liu, Jun Huang, Rui Tan, and Zhen Tan. 2018. Detecting Wireless Spy Cameras Via Stimulating and Probing. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, Munich Germany, 243–255. https://doi.org/10.1145/3210240.3210332

[58] Mohd Yazid Abd Manap, Rohilah Sahak, Azlee Zabidi, Ihsan M. Yassin, and Nooritawati Md. Tahir. 2015. Object detection using depth information from Kinect sensor. *2015 IEEE 11th International Colloquium on Signal Processing & Its Applications (CSPA)* (2015), 160–163.

[59] Giulio Marin. 2013. Confidence Estimation of ToF and Stereo Data for 3D Data Fusion. Stima della Confidenza delle Misure Ottenute da Sensori ToF e da Sistemi Stereo per la Fusione di Dati 3D. (2013). https://core.ac.uk/display/18229006

[60] Tiffany May and Su-Hyun Lee. 2019. 1,600 Motel Guests Were Secretly Streamed Live in South Korea, Police Say (Published 2019). *The New York Times* (March 2019).

[61] T. J. McCue. 2019. How To Find A Hidden Camera In Your House Or Airbnb Or Anywhere. *Forbes* (Oct. 2019). https://www.forbes.com/sites/tjmccue/2019/10/30/how-to-find-a-hidden-camera-in-your-house-or-airbnb-or-anywhere/?sh=24f63b5a1860

[62] Niluthpol Chowdhury Mithun, Sirajum Munir, Karen Guo, and Charles Shelton. 2018. ODDS: Real-Time Object Detection Using Depth Sensors on Embedded GPUs. *2018 17th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)* (2018), 230–241.

[63] Mordor Intelligence. 2021. Time-of-Flight (TOF) Sensor Market | 2021-2026 | Industry Report | Covid Insights. https://www.mordorintelligence.com/industry-reports/time-of-flight-sensor-market

[64] Esha Nerurkar, Simon Lynen, and Sheng Zhao. 2017. System and Method for Concurrent Odometry and Mapping.

[65] BBC News. 2019. Goo Hara and the Trauma of South Korea's Spy Cam Victims. *BBC News* (Nov. 2019).

[66] Niantic. 2020. Pokémon GO. https://niantic.helpshift.com/a/pokemon-go/?s=accessories&f=catching-pokemon-in-ar-mode&l=en&p=web

[67] The Times of India. 2019. Tourists Find Hidden Camera in Uttarakhand Hotel Room Dehradun News - Times of India. *The Times of India* (May 2019).

[68] Omnivision. 2006. *OV2640 Datasheet.* Technical Report. 47 pages. https://www.uctronics.com/download/cam_module/OV2640DS.pdf

[69] South China Morning Post. 2019. As South Korea Confronts 'Spycam Porn', Pressure Builds on Retailers. *South China Morning Post* (March 2019).

[70] Lubos Vonasek Programmierung. 2021. 3D Live Scanner – Apps on Google Play. https://play.google.com/store/apps/details?id=com.lvonasek.arcore3dscanner&hl=en_SG&gl=US

[71] Feng Qian, Bao Zhang, Chuanli Yin, Mingyu Yang, and X. Li. 2015. Recognition of interior photoelectric devices by using dual criteria of shape and local texture. *Optical Engineering* 54 (2015).

[72] Ray Fontaine. 2019. Imaging + Sensing End-of-Year Highlights | TechInsights. https://www.techinsights.com/blog/imaging-sensing-end-year-highlights.

[73] Ximing Ren and L. Li. 2011. Recognizing cat-eye targets with dual criterions of shape and modulation frequency. *Chinese Optics Letters* 9 (2011), 041101–41104.

[74] Retia. 2021. How to Set Up a Wi-Fi Spy Camera with an ESP32-CAM « Null Byte :: WonderHowTo. https://null-byte.wonderhowto.com/how-to/set-up-wi-fi-spy-camera-with-esp32-cam-0246590/

[75] Marc Roessler. 2002. How to find hidden cameras.

[76] Laurel C. Sadler and T. Alexander. 2010. Mobile optical detection system for counter-surveillance. In *Defense + Commercial Sensing*.

[77] Samsung. 2020. What Is DepthVision Camera on Galaxy S20+ and S20 Ultra? - The Official Samsung Galaxy Site. https://www.samsung.com/global/galaxy/what-is/depthvision-camera/

[78] Akash Deep Singh, L. Garcia, Joseph Noor, and M. Srivastava. 2020. I Always Feel Like Somebody's Sensing Me! A Framework to Detect, Identify, and Localize Clandestine Wireless Sensors. *ArXiv* abs/2005.03068 (2020).

[79] SoftBank Robotics. 2021. Commercial Cleaning Robots | SoftBank Robotics America. https://us.softbankrobotics.com/solutions/commercial-cleaning

[80] Detective Store. 2021. *Camera detectors.* https://www.detective-store.com/cameras-detectors-224.

[81] Susannah Sudborough. 2019. Cape Cod Resort Employee Used Secret Camera to Film Women Showering: Police. *NBC Boston* (Feb. 2019).

[82] Susrutha Babu Sukhavasi, Suparshya Babu Sukhavasi, Khaled Elleithy, Shakour Abuzneid, and Abdelrahman Elleithy. 2021. CMOS Image Sensors in Surveillance System Applications. *Sensors* 21, 2 (Jan. 2021), 488. https://doi.org/10.3390/s21020488

[83] Daniel Svedbrand, L. Allard, Magnus Pettersson, Pontus Köhler, M. Henriksson, and Lars-Göran Sjökvist. 2019. Optics detection using an avalanche photo diode array and the scanning-slit-method. In *Security + Defence*.

[84] Shuai Tang, Xiaoyu Wang, Xutao Lv, Tony X. Han, James Keller, Zhihai He, Marjorie Skubic, and Shihong Lao. 2013. Histogram of Oriented Normal Vectors for Object Recognition with a Depth Sensor. In *Computer Vision – ACCV 2012*, Kyoung Mu Lee, Yasuyuki Matsushita, James M. Rehg, and Zhanyi Hu (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 525–538.

[85] Techlicious. 2019. The Secrets to Finding Hidden Cameras.

[86] Maggie Tillman. 2021. *What is a ToF camera? Time-of-flight sensor and photography explained.* https://www.pocket-lint.com/phones/news/147024-what-is-a-time-of-flight-camera-and-which-phones-have-it.

[87] The Straits Times. 2019. South Korea Spycam Crimes Put Hidden Camera Industry under Scrutiny. *The Straits Times* (March 2019).

[88] Yan Wen, Chuanyan Hu, Guanghui Yu, and Changbo Wang. 2012. A robust method of detecting hand gestures using depth sensors. In *2012 IEEE International Workshop on Haptic Audio Visual Environments and Games (HAVE 2012) Proceedings*. IEEE, 72–77.

[89] Damien Wilde. 2021. Unsurprisingly, the Samsung Galaxy S20+ is the top-selling 5G smartphone in the US. https://9to5google.com/2020/05/21/galaxy-s20-5g-sales-us/

[90] Robert Williams. 2020. Samsung Galaxy S20+ outsells all other 5G phones, study finds. https://www.marketingdive.com/news/samsung-galaxy-s20-outsells-all-other-5g-phones-study-finds/578560/

[91] K. Wu. 2018. Detecting Streaming Wireless Cameras with Timing Analysis.

[92] K. Wu and Brent Lagesse. 2019. Do You See What I See? Detecting Hidden Streaming Cameras Through Similarity of Simultaneous Observation. *2019 IEEE International Conference on Pervasive Computing and Communications (PerCom* (2019), 1–10.

[93] Kesheng Wu, Ekow Otoo, and Kenji Suzuki. 2005. Two Strategies to Speed up Connected Component LabelingAlgorithms. *Pattern Analysis Application* 0, 0 (11 2005). https://www.osti.gov/biblio/929013

[94] Xiuxin. 2019. Amazon.Com: 2 Pack ESP32-CAM WiFi Bluetooth Camera Module Development Board ESP32 with Camera Module OV2640: Computers & Accessories. https://www.amazon.com/ESP32-CAM-Bluetooth-Camera-Module-Development/dp/B07S5PVZKV

[95] Yida. 2020. What is a Time of Flight Sensor and How does a ToF Sensor work? https://www.seeedstudio.com/blog/2020/01/08/what-is-a-time-of-flight-sensor-and-how-does-a-tof-sensor-work/

[96] X. Zhang and Feng Qian. 2017. A Fast Recognition Algorithm for Photoelectric Peeping Equipment. *DEStech Transactions on Computer Science and Engineering* (2017).

[97] Mandy Zuo. 2019. Hidden Hotel Cameras Lead to Arrests in China. *South China Morning Post* (June 2019).