PrivacyCamera: Cooperative Privacy-Aware Photographing with Mobile Phones

Ang Li[†], Qinghua Li[†], Wei Gao[§]
[†]Department of Computer Science and Computer Engineering, University of Arkansas
[§]Department of Electrical Engineering and Computer Science, University of Tennessee
Email:[†]{angli,qinghual}@uark.edu, [§]weigao@utk.edu

***Abstract*—Nowadays, mobile phones are usually embedded with powerful cameras. Due to the convenience of carrying mobile phones, an increasing number of people use mobile phones to take photos anytime and anywhere. However, when a user takes a photo of a scenery, a building or a target person, sometimes an unexpected stranger is also included in the photo. Such photos reveal where the stranger has been and thus can breach his privacy. This problem has received little attention in the literature. In this paper, we propose PrivacyCamera, a cooperative system to protect the stranger's privacy in the above scenario. Through cooperation between the photographer and the stranger, the system can automatically blur the stranger's face in the photo upon the stranger's request when the photo is being taken. This paper describes the design, analysis, prototype implementation, and experimental evaluation of the system. Experiments show that PrivacyCamera can effectively protect stranger's privacy in an efficient way.**

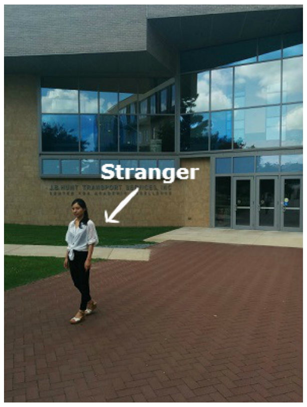
***Index Terms*—Mobile Phone, Photo, Privacy**

I. Introduction

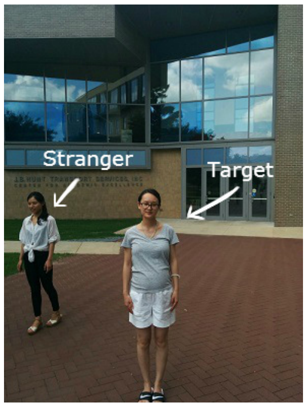
Mobile phones are usually embedded with powerful cameras today [1]. For example, iPhone 6 has an 8-megapixel camera. As mobile cameras in the pocket, mobile phones are increasingly used by people to take photos anywhere and anytime. However, there are some privacy issues associated with this convenience. When a user takes a photo of a beautiful view or a friend using his mobile phone, a stranger may be accidentally included in the photo, with the face clearly recognizable. Figure 1 shows two examples. In Figure 1(a), the photographer intends to take a photo of the building but a stranger appears; in Figure 1(b), the photographer intends to picture the target person but a stranger is also included. In these cases, the photo can reveal the stranger's location and even activity. For strangers who do not want to appear in the photo and get their location exposed, being accidentally included in a photo breaches privacy. Thus, this problem should be addressed.

With the development of image processing technology, there exist several softwares which can blur faces in a photo, such as Adobe Photoshop and ObscuraCam [2]. However, none of these commercial softwares can make the stranger in the photo know that he is included in the photo and give him the right to decide whether to blur his face or not. These solutions only allow the photographer to make decisions as to blurring the stranger's face or not.

A naive solution for protecting stranger's privacy is that each user stores a pool of familiar faces (e.g., self, family members and friends) in the phone and the phone simply blurs all other faces in the photo. However, this solution may cause unnecessary blurring. Some strangers may not care about whether they are included in the photo or not. Blurring their faces is not needed and can unnecessarily degrade the quality of the photo.



(a) A stranger is included when the photographer pictures a building



(b) A stranger is included when the photographer pictures a target person

Fig. 1. Privacy issues with photos taken by mobile phones

In this paper, we propose a mobile cooperative privacy protection system, called *PrivacyCamera*, to protect the privacy of a stranger who is accidentally included in a photo taken by mobile phones. PrivacyCemara can work as an App on both the photographer's and the stranger's mobile phone. At the time of taking a photo, it can automatically notify nearby strangers of the possible inclusion in the photo via peer-to-peer short-range wireless communications (e.g., Wifi Direct [3]). If a stranger does not want to appear in the photo, he can send a request to the photographer. The photographer will determine if the requesting stranger is in the photo or not. If so, the photographer will blur the stranger's face in the photo.

The contribution of this paper is summarized as follows:

- To the best of our knowledge, PrivacyCamera is the first mobile system which can notify nearby strangers of the possible inclusion in a photo when the photo is being taken, give them an option to opt out, and blur a stranger's face upon his request.
- We design a location-based stranger determination scheme to determine if a stranger is in the photo or not based on his relative location to the photographer and the heading direction of the camera, and theoretically analyze its effectiveness.
- We design a Gaussian Blur-based face blurring scheme that can smoothly blur a stranger's face with minimal negative effect on the quality of a photo.
- We implement a prototype system on Nexus 5 phones, and evaluate the system's performance and cost using experiments.

The rest of the paper is organized as follows. Section II presents the design of PrivacyCamera and theoretical analysis of its performance. Section III describes the prototype implementation. Section IV shows evaluation results. Section V reviews related work. Section VI concludes the paper.

II. System Design

This section describes the design of PrivacyCamera and analyzes its performance.

A. System Overview

Three types of entities are involved in the system: the *photographer* who takes photos using a mobile phone, the *target* that the photographer intends to picture, and the *stranger* who is near the photographer and might be unintentionally included in the photo. The target can be a building, a natural scenery, a person, etc. The system is designed for outdoor usage.

The system aims to protect the stranger's privacy through providing a method for the stranger to opt out from the photo. The basic idea is that the photographer notifies nearby strangers of the possible inclusion in a photo at the time of taking the photo, and blurs a stranger's face in the photo upon the stranger's request. Note that the system does not intend to simply blur every stranger's face in the photo. This is because blurring inevitably affects the quality of the photo, even though our design adopts an advanced blurring technique to minimize such effect. To minimize the quality degradation brought to the photo, the system only blurs a stranger's face if he requests.

To make the system work, both the photographer and the stranger are required to install PrivacyCamera (in the form of an App) on their mobile phone. PrivacyCamera relies on the cooperation between photographers and strangers to protect privacy. Since each mobile phone user can sometimes be a photographer and sometimes a stranger, PrivacyCamera users essentially cooperatively protect each other's privacy. It is worthwhile to note that the paradigm of inter-user cooperation has been successfully adopted in many real-world systems such as peer-to-peer file downloading systems [4] and online recommender systems [5]. This success has also motivated our system design. The more users adopt this system, the better privacy can be protected for each other.

As the first work in this direction, this paper starts with considering two relatively simple photographing scenarios:

- **Scenario 1:** The target of a photo is not a person but something else such as a building. One stranger is accidentally included in the photo, and he may or may not want his face to be blurred.
- **Scenario 2:** The target is a person. One stranger accidentally appears in the photo, and he may or may not want to blur his face.

Based on our observations, these two scenarios represent a significant portion of photographing cases. Thus, our scheme can enhance privacy in many real-world scenarios. We will address more complex scenarios in future work.

Even under these two relatively simple scenarios, the problem is still challenging. First, there might be multiple strangers nearby who can receive the notification of possible inclusion in the photo. Some of them may request their faces to be blurred but others may not request so. Although only one stranger is included in the photo in the two scenarios, we still need to determine if the stranger in the photo is requesting for face blurring or not, which is not easy. Second, in Scenario 2, if the stranger in the photo requires face blurring, we need to make sure that the stranger's face, not the target's, is blurred.

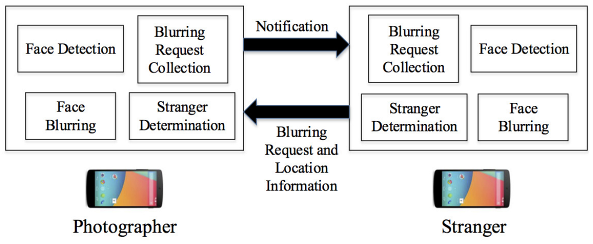


Fig. 2. The architecture of PrivacyCamera

For simplicity, photographer is also used to denote the photographer's mobile phone when the context is clear. The same applies to stranger.

B. The Architecture and Workflow of PrivacyCamera

As Figure 2 shows, the system consists of four major modules: *face detection*, *blurring request collection*, *stranger determination* and *face blurring*. When a photographer takes a photo, the face detection module will run on the captured image. If no face is detected, no further processing is needed. If any face is detected, the blurring request collection module sends notifications to nearby strangers using peer-to-peer short-range wireless communications. If a stranger receiving the notification does not want to be included in the photo, he sends a blurring request to the photographer. Since this stranger may or may not be included in the photo, to help the photographer determine if this stranger is the one in the photo, this stranger puts his location (i.e., GPS coordinates) in the request. To avoid linking attacks based on the IP and MAC addresses exposed in communications, the photographer and each stranger can frequently change their IP and MAC addresses using standard mechanisms [6], [7]. Then, the stranger determination module of the photographer will check if the requesting stranger is in the photo or not based on their relative location and the heading direction of the camera. If the stranger is in the photo, the face blurring module of the photographer will smoothly blur his face; otherwise, the request is ignored.

The design of PrivacyCamera is based on several technologies available in commercial-off-the-shelf mobile phones. Face detection can be done using APIs provided by mobile phones, e.g., the *FaceDetector* [8] APIs in Android SDK. Peer-to-Peer communications between the photographer and nearby strangers can be supported by short-range wireless technologies such as WiFi Direct [3] and Bluetooth which are available on most mobile phones today, e.g., Nexus 5. We will introduce how these two modules can be implemented in our prototype system in Section III. Next, we describe how to determine if a stranger is in the photo and how to blur faces.

C. Stranger Determination

This module detect if a stranger who requests face blurring is included in the photo or not. This is done through checking if the stranger is in the field of view of the photographer's camera or not.

This process is illustrated in Figure 3. First, the camera's heading direction is determined using the orientation sensor embedded in mobile phones. Note that the heading direction read from compass is in degrees east of Magnetic North instead of True North (there is a declination angle between the two), and it should be converted in degrees east of True North (i.e., β in the figure) so as to be in the same coordinate system with GPS coordinates. Next, the stranger's relative direction to the photographer (i.e., α in the figure) is obtained using the GPS coordinates of the stranger and the photographer. Then, the relative angle from the stranger to the camera (denoted by δ) is calculated as $\delta = |\beta - \alpha|$. Lastly, we determine if the stranger is in the field of view of the camera or not. The horizontal view angle of the camera (denoted by γ) which specifies the effective horizontal scope of the camera can be obtained using the API of mobile OS (e.g., GetHorizontalViewAngle() on Android OS). For example, it is 60°under default focal length for the Nexus 5 phone. If $\delta \leq \gamma/2$, the stranger is in the photo; otherwise, he is not in the photo.

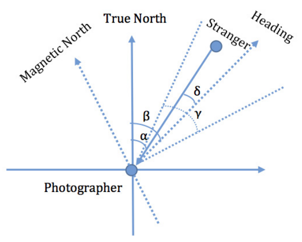


Fig. 3. Detecting if a stranger is in field of view of a camera

For Scenario 2, it is not enough to determine that the stranger is in the photo. We also need to tell which face is the target and which is the stranger. To achieve this goal, we adopt a heuristic approach. We observe from real-life experiences that when we take a photo of a target person, we usually intentionally make the target's face larger than anyone else accidentally included into the photo. For example, if a stranger is too close to the camera which makes his face larger than the target's, the photographer will probably change a facing direction or ask the target to move a little so that the target is better captured into the photo than the stranger. Thus in the photo the stranger's face should be smaller than the target's. Based on this, the smaller face will be determined as the stranger in Scenario 2.

D. Face Blurring

The goal of face blurring is to mask the identifiable features of a face without reducing the quality of the photo much.

As a preparation step for face blurring, we first need to determine a blurring area in the face which encloses the main identifiable features of the face. In this paper, a square area is used as the blurring area. Specifically, the square is drawn by setting the middle point between eyes as the center of the square, and setting the length of a side as 2.4 times of the distance between eyes. Our tests show that the square drawn in this way can cover the main identifiable features of a face with the minimum area (see Figure 5). Next, we describe how to blur this square area.

To blur faces smoothly, the Gaussian Blur algorithm [9] is used in this paper. The effect of Gaussian Blur is like viewing an image through a translucent screen. The basic idea of Gaussian Blur is to adjust the color value of each target pixel (under the RGB color model) as the weighted average of the color value of itself and other nearby pixels. The weights are calculated based on Gaussian function such that closer pixels have higher weights. Since the pixels closer to the target pixel usually have more similar colors with the target pixel than the pixels farther away, this blurring method can achieve smooth blurring with minimal effect on photo quality.

Given a target pixel to blur, all the pixels that will be used to blur the target pixel are enclosed in a circle with radius R and centered in the target pixel. Let C denote this circle. Let the target pixel be the origin of a two-dimensional coordinate system whose coordinates are $[0, 0]$. Then for pixel $[x, y]$ in the circle, where x and y represent the abscissa and the ordinate of this pixel, its weight is calculated as follows:

$$w(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{1}$$

where σ is the standard deviation of Gaussian distribution. Let $v_{[x,y]}(R, G, B)$ denote the color values of a pixel $[x, y]$. Then the blurred color values of the target pixel is computed as follows:

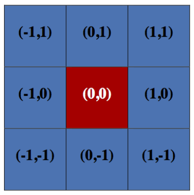
$$v_{[0,0]}(R,G,B) = \sum_{[x,y]\in C} v_{[x,y]}(R,G,B) \frac{w(x,y)}{\sum_{[x,y]\in C} w(x,y)} \tag{2}$$

Here, we use a simple example in Figure 4(a) to illustrate how a target pixel is blurred. In this example, there are 9 pixels in an image, and the central pixel is the target pixel which is marked in red color. The blur radius is set such that only the target pixel and its direct neighbors are in the circle. The coordinates of the 9 pixels are shown in Figure 4(a). The original RGB color values of each pixel is shown in Figure 4(b). To conduct Gaussian Blur, the weight for each pixel is calculated by Equation 1. Suppose σ in Equation 1 is 1.5. Then the original weight for each pixel computed by Equation 1 is shown in Figure 4(c). After that the new color values for the target pixel can be calculated using Equation 2, which are shown in Figure 4(d).

For the square blurring area, we can blur each pixel in this square from the leftmost pixel to the rightmost pixel in each row and row by row using Gaussian Blur. Given a certain σ, the effect of Gaussian Blur is more intense as the blur radius increases. To fully hide the identifiable features in the face, a big enough blur radius should be set. Our tests show that the effect of Gaussian Blur with $\sigma = 3$ and $R = 30$ is good enough. Figure 5 shows the blur effects under different radius values and fixed $\sigma = 3$. It can be seen that when the blur radius is 30 the face cannot be identified.

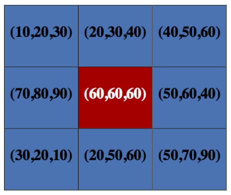
E. Analysis on the Effectiveness of Protection

Suppose a stranger is included in a photo and he requests to blur his face. For the stranger's face to be really blurred, a precondition is that the stranger determination module successfully detects the stranger as being in the photo based on the stranger's claimed location measured by GPS. Here,



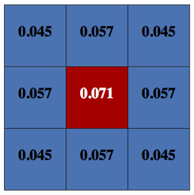
(-1,1)	(0,1)	(1,1)
(-1,0)	(0,0)	(1,0)
(-1,-1)	(0,-1)	(1,-1)

(a) Pixel coordinate in an image



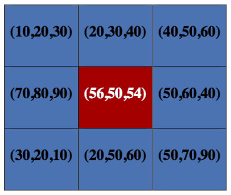
(10,20,30)	(20,30,40)	(40,50,60)
(70,80,90)	(60,60,60)	(50,60,40)
(30,20,10)	(20,50,60)	(50,70,90)

(b) Original color values for each pixel



0.045	0.057	0.045
0.057	0.071	0.057
0.045	0.057	0.045

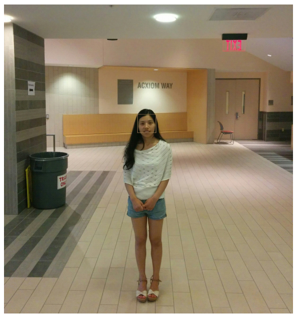
(c) Original weight for each pixel



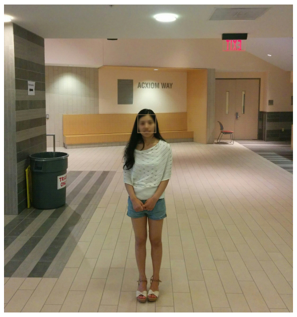
(10,20,30)	(20,30,40)	(40,50,60)
(70,80,90)	(56,50,54)	(50,60,40)
(30,20,10)	(20,50,60)	(50,70,90)

(d) Blurred color values for the central pixel

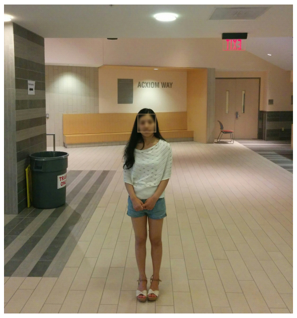
Fig. 4. An example process of Gaussian Blur for blurring the central pixel



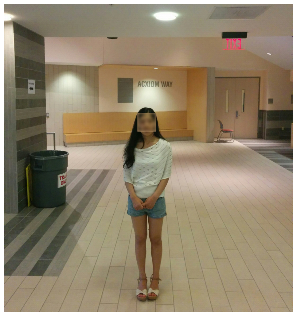
(a) without blurring



(b) radius=10



(c) radius=20



(d) radius=30

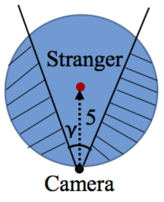
Fig. 5. Effect of Gaussian Blur with different radiuses

we analyze the *true protection rate* of PrivacyCamera, which is defined as the percentage of times when the system can successfully detect a stranger as being in the photo given that the stranger is really in the photo.

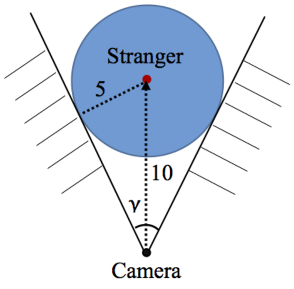
The true protection rate depends on a few factors: GPS accuracy r, the horizontal view angle of the camera γ, the real distance between the stranger and the camera d, and the real relative angle from the stranger to the camera δ. It actually equals to the probability that when the stranger is really in the camera's field of view, its GPS-measured location is also in the camera's field of view. If we draw a circle centered at the stranger's real location with radius r, a true protection happens when the GPS-measured location is within the intersection between the circle and the view of the camera. The probability is equal to the fraction of the circle in the intersection (assuming that it is equally likely for the GPS-measured location to be any point in the circle). Figure 6 shows two special cases when the stranger is directly facing the camera (i.e., $\delta = 0$) and the distances are 5 meters and 10 meters.

Since r and γ depend on the device, we can consider these two parameters as constants. In fact, we obtained r and γ on Nexus 5 phones (see Section IV), which are 5 meters and 60°, respectively. For simplicity, we use these values in our analysis. Next we analyze the true protection rate as a function of d and δ.

Generally speaking, given a certain δ, the true protection rate will be higher when d increases, since longer d can better tolerate the inaccuracy of GPS. Here, we want to find out the upper bound and lower bound of the true protection rate. Since the effective range of face detection is 10 meters as shown in Section IV-C, the case with $d = 10$ meters is considered as the upper bound. The worst case happens when the stranger and



(a) Case: d=5 meters



(b) Case: d=10 meters

Fig. 6. Two special cases where δ=0°and r=5 meters

the photographer are at the same location, i.e., $d = 0$. Besides, we also consider the case with $d = 5$ meters as a reference point in the middle, based on our experience that a stranger is more than 5 meters away from the camera in most cases.

The true protection rate for the lower bound case is pretty straightforward to derive. Since γ is 60°, the true protection rate equals to the probability that the GPS-measured location is within the view of the camera, which is $\frac{60}{360} = 16.7\%$.

The true protection rate for the other two cases $d = 5$ and $d = 10$ with changing δ can be deduced using geometry. We omit the detailed process due to the space limitation. For $d = 5$ meters, the true protection rate is given in Equation 3; for $d = 10$ meters, the true protection rate is given in Equation 4.

$$P_1 = \frac{1}{3} + \frac{h_1 \times \sqrt{r^2 - h_1^2} + h_2 \times \sqrt{r^2 - h_2^2}}{\pi r^2}, \tag{3}$$

where $h_1 = r \times \sin(30 - \delta)$, $h_2 = r \times \sin(30 + \delta)$ and $\delta \in [0°, 30°]$.

$$P_2 = \frac{2\arccos\frac{h}{r}}{360} + \frac{h \times \sqrt{r^2 - h^2}}{\pi r^2}, \qquad (4)$$

where $h = d \times \sin(30 - \delta)$ and $\delta \in [0°, 30°]$.

Based on these two equations, we can calculate the theoretical true protection rate for any specific relative angle δ in these two cases. Figure 7 shows the numerical results where the x-axis is the relative angle δ. The upper bound achieves to 100% when the stranger is directly facing to the camera (i.e., $\delta = 0$) with the distance of 10 meters. When the stranger and the photographer stand at the same spot, the true protection rate is 16.7%, which is the lower bound. For the case with $d = 5$ meters, the true protection rate decreases from 60% to 46% when the relative angle increases. When the real distance is between 5 meters and 10 meters, the true protection rate is expected to sit between the red dashed line and the green dotted line.

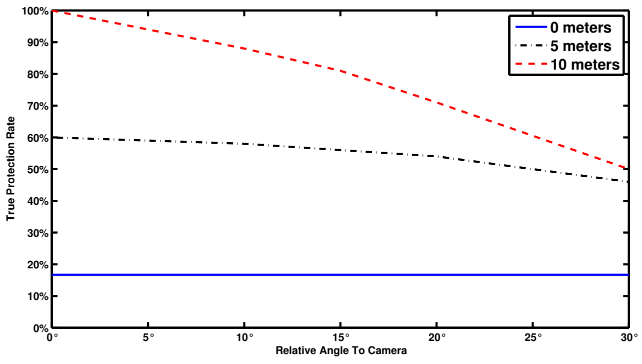


Fig. 7. True protection rate when the distance from the stranger to the camera is 0 meters, 5 meters, and 10 meters. X-axis is the relative angle δ.

III. Implementation

We implemented a prototype system on Nexus 5 phones. The system uses Android 5.1.1 OS and Android 4.3 APIs. This section describes the implementation of major modules. The face blurring module is implemented as described in Section II-D and thus not described in details here.

A. Face Detection

The face detection module is implemented based on the *FaceDetector* [8] class provided in Android SDK. Faces in an image can be detected by calling the *findFaces* method of *FaceDetector*. This method detects faces by finding pupils in the image. The *findFaces* method returns a number of detected faces in the image and populates them into an array of *FaceDetector.Faces* class [10].

From each instance of the *FaceDetector.Faces* class, we can obtain the distance between the two eyes of a face in pixels and the coordinates of the middle point between the two eyes. As introduced in Section II-D, the face blurring module uses these information to determine the blurring area for a face.

B. Blurring Request Collection

This module enables the photographer to send notifications to nearby strangers, and enables each stranger to send a blurring request as well as his location to the photographer. In our prototype, Wifi Direct [3], [11], [12] is used to implement the peer-to-peer communications between the photographer and the strangers. The photographer first discovers nearby peers by calling the *discoverPeers* method of *WifiP2pManager* system service. Next he sends a notification to each peer and collects the blurring request from peers. Then the communications end. Note that the peer discovery and communications between the photographer and strangers happen only when a photo is taken.

Location Acquisition The stranger gets his location using the *LocationManager* [13] service. Depending on the device, several technologies can be applied to determine current location, including GPS and cellular network. In the prototype, we check the availability of GPS and cellular network in turn. Then, we can find the current location by calling the corresponding method of GPS or cellular network. Finally, the longitude and latitude of the current location are obtained.

C. Stranger Determination

We introduced how to detect if a stranger is in the photo or not in Section II-C. Here, we describe how to obtain the parameters used in that approach (see Figure 3) on a phone.

The camera's heading direction (β in Figure 3) can be obtained by reading sensor data from gyroscope embedded in Android phones. However, the returned value is in degrees east of Magnetic North instead of True North. To be in the same coordinate system with GPS coordinates, we convert it to be in degrees east of True North by adding the declination angle between Magnetic North and True North. The declination angel can be obtained by calling the native method in Android APIs.

We can obtain the relative direction from the stranger to the photographer (α in Figure 3) by calling the *bearingTo* method of the *Location* object, passing in the stranger's current location. Since the original bearing value returned from *bearingTo* is within the range from negative 180°to positive 180°, we normalize it to be within the range from 0°to 360°.

With the camera's heading direction and the relative direction from the stranger to the photographer, we can easily calculate the relative angle δ in Figure 3 and normalize it to be within 0°and 180°.

The horizontal view angle of the camera γ in Figure 3 can be obtained by calling the *getHorizontalViewAngle* method of the *Camera.Parameters* object. In our current prototype, only the rear-facing camera without zooming in and zooming out has been considered. However, the value with zooming in and zooming out can be obtained similarly, and our general approach is applicable to those cases as well.

IV. Evaluations

A. Experimental Methodology

The experiments are conducted outdoors on the campus of the University of Arkansas under fine weather. Figure 8 shows two typical scenes of experiments. In the experiments, standard GPS instead of assisted GPS (A-GPS) [14] is used for location acquisition since A-GPS has lower accuracy. Before conducting each test, we wait around 30 seconds to make sure that the GPS receiver is able to get the latest location information. Each test is done at a different location. We do

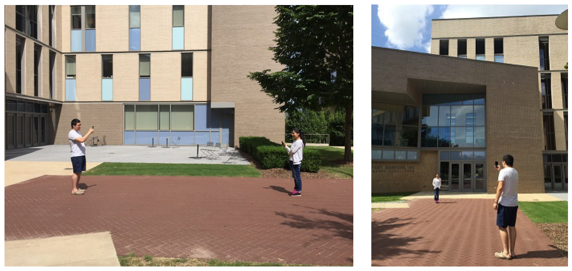


Fig. 8. Experiment scene

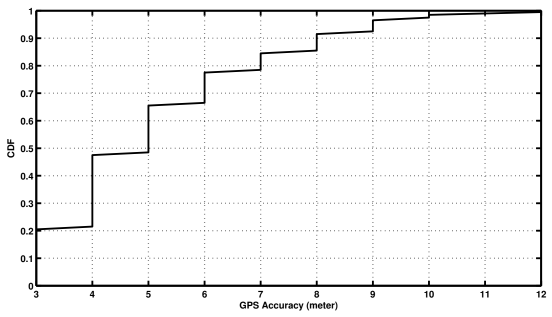


Fig. 9. Cumulative distribution of GPS accuracy

not zoom in and zoom out the rear-facing camera, and just use the default focal length.

B. GPS Accuracy Test

As GPS is used to determine the location of strangers, the accuracy of location obtained from the GPS receiver of a phone is an important factor that determines the performance of the system. The accuracy level of GPS may vary from tens of meters to millimeters [15]. The actual accuracy depends on many factors, such as sky blockage, receiver quality and atmosphere condition [16]. For high-quality consumer-grade GPS receivers, the accuracy can be within 5 meters under the open sky and 10 meters under closed canopies [17].

To examine the accuracy of GPS receivers on mobile phones, we conducted 100 tests at different locations. Each test calls the *getAccuracy* method of the *Location* object in Android to get an approximate accuracy at the current location in meters. The approximate accuracy is defined in the following way: if we draw a circle with the center at the current location and the radius equal to the accuracy, there is a 68% possibility that the true location is inside the circle. The test results are shown in Figure 9. The average accuracy is about 5 meters, and in 66% of the tests the accuracy is no more (i.e., not worse) than 5 meters.

C. Face Detection Test

This part evaluates the effectiveness of the face detection module in detecting faces in a photo. Considering that the strength of light might affect detection, we conducted tests in the morning, at noon and in the evening. As shown in Figure 10, even under dark lighting conditions, the face detection module can effectively detect the face in photos. Additionally, we changed the positions of the person to be detected within the field of view of the camera. Specifically, since the horizontal view angle of Nexus 5 is 60°, we did tests when the relative angel from the person to the camera (δ in Figure 3) is 0°, 10°, 20°and 30°. The results show that faces can be successfully detected when the distance between the person and the camera is within 10 meters at any relative angles, but cannot be detected when the distance is over 11 meters. When the distance is between 10 meters and 11 meters, faces can sometimes be detected.

TABLE I
TRUE PROTECTION RATE FOR SCENARIO 1 WHEN $d = 5$ METERS

Test	**Relative angle to camera**		
	0°	**15°**	**30°**
Test 1	20°	2°	15°
Test 2	8°	41°	20°
Test 3	17°	3°	22°
Test 4	12°	60°	16°
Test 5	40°	18°	54°
Test 6	34°	13°	41°
Test 7	37°	49°	39°
Test 8	18°	17°	29°
Test 9	20°	37°	44°
Test 10	23°	24°	61°
True Protection Rate	70%	60%	50%

D. Accuracy of Protection

This part evaluates the effectiveness of our system in protecting the stranger's privacy.

1) True Protection Rate for Scenario 1: This group of tests considers Scenario 1 where one stranger appears in the photo. Suppose the stranger wants to blur his face. We evaluate the true protection rate. In our tests, the stranger stands 5 meters and 10 meters away from the camera. The stranger and the photographer are positioned in ways such that the relative angle between the stranger and the camera's heading direction (δ in Figure 3) is 0°(i.e., the camera directly faces the stranger), 15°and 30°. The camera's heading direction is randomly set in each test. If the relative angle calculated by the stranger determination module is no more than 30°, the stranger is successfully detected as being in the image and his face is blurred. Figure 11(a) gives an example to show the effect of blurring.

Table I and Table II show the true protection rates when the distance is 5 meters and 10 meters respectively. In each table, we also show the relative angles calculated by the stranger determination module to provide more information. In both cases, the true protection rate decreases when the relative angle increases, i.e., when the stranger is closer to the edge of the camera's view. The true protection rate is higher when the distance is 10 meters than when it is 5 meters, because longer distance can better tolerate the inaccuracy of GPS location. Moreover, we can find that these test results are consistent with our theoretical analysis shown in Figure 7.

2) False Protection Rate For Scenario 1: Suppose in Scenario 1, the stranger in the photo (denoted by A) does not request to blur his face. However, another nearby stranger B who is not in the photo may submit a blurring request. In this case, we define *false protection rate* as the percentage of times when the stranger B not in the photo is mistakenly detected as being in the photo and stranger A's face is hence falsely

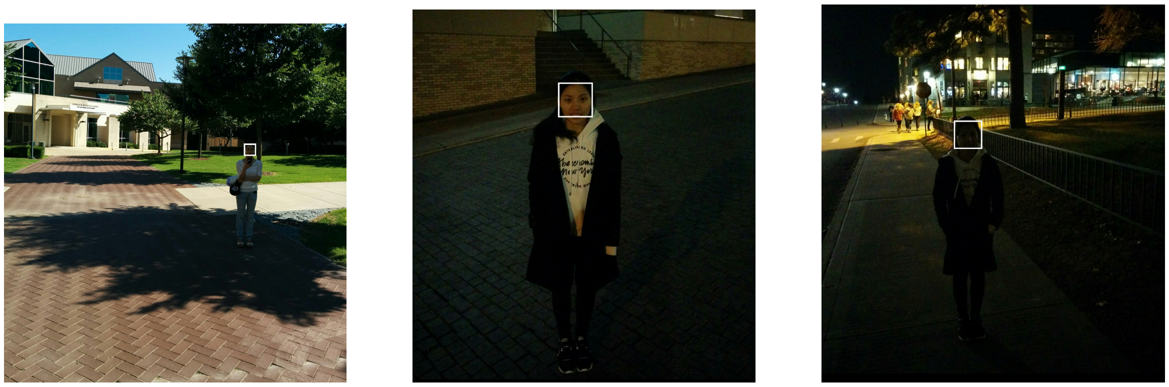


Fig. 10. Face detection under poor lighting conditions

TABLE II
TRUE PROTECTION RATE FOR SCENARIO 1 WHEN $d = 10$ METERS

Test	**Relative angle to camera**		
	0°	**15°**	**30°**
Test 1	23°	18°	29°
Test 2	3°	19°	2°
Test 3	8°	39°	44°
Test 4	7°	46°	38°
Test 5	5°	22°	35°
Test 6	0°	34°	21°
Test 7	22°	17°	46°
Test 8	21°	2°	25°
Test 9	19°	15°	29°
Test 10	31°	25°	17°
True Protection Rate	90%	70%	60%

TABLE III
FALSE PROTECTION RATE FOR SCENARIO 1 WHEN $d = 5$ METERS

Test	**Relative angle to camera**					
	30°	**60°**	**90°**	**120°**	**150°**	**180°**
Test 1	15°	28°	87°	82°	138°	169°
Test 2	20°	35°	72°	95°	179°	171°
Test 3	22°	42°	73°	83°	165°	178°
Test 4	16°	74°	63°	94°	147°	174°
Test 5	54°	77°	67°	147°	124°	159°
Test 6	41°	72°	58°	136°	178°	167°
Test 7	39°	63°	78°	127°	175°	192°
Test 8	29°	65°	79°	121°	179°	203°
Test 9	44°	82°	97°	123°	160°	163°
Test 10	61°	79°	50°	138°	158°	168°
False Protection Rate	50%	20%	0%	0%	0%	0%

blurred. In the tests, stranger B and the photographer are positioned in ways such that the relative angle from stranger B to the camera (δ in Figure 3) is 30°, 60°, 90°, 120°, 150°and 180°. If the relative angle calculated by the stranger determination module is no more than 30°, stranger B is falsely detected as being in the photo. Table III and Table IV show the results when the distance between B and the photographer is 5 meters and 10 meters respectively. Similar to the true protection rate case, the relative angles calculated by the stranger determination module are also shown. We can see that the false protection rate decreases when the relative angle increases and when the distance increases which is reasonable.

We further evaluate the false protection rate in a more noisy environment, where there are five strangers like stranger B in the above test who are not in the photo but submit a blurring request. In this case, we define *false protection rate* as the percentage of times when anyone of these five strangers not in the photo is mistakenly detected as being in the photo and stranger A's face is hence falsely blurred. In this group of tests, the distance between strangers and the photographer is between 5 and 10 meters, and the relative angle from strangers to the camera is within 30°to 90°. All the strangers' locations are randomly selected within these ranges. Over 20 independent tests, the false protection rate is as low as 10%.

3) True Protection Rate for Scenario 2: The true protection rate for Scenario 2 depends on two factors. One factor is the true protection rate for Scenario 1, and the other factor is the accuracy of correctly telling the stranger's face from the target's face. We first ran tests to evaluate the accuracy of correctly telling the stranger's face from the target's. In our tests, the stranger stands farther from the camera than the target as assumed in Section II-C. They do not stand in a line so that both of their faces appear in the photo. To make sure both faces can be detected, we keep them within 10 meters from the camera. Over 20 tests, we found that the system can always successfully tell the stranger's face. As a result, the true protection rate in Scenario 2 should be the same as that of Scenario 1. Figure 11(b) gives an example to show the effect of blurring.

TABLE IV
FALSE PROTECTION RATE FOR SCENARIO 1 WHEN $d = 10$ METERS

Test	**Relative angle to camera**					
	30°	**60°**	**90°**	**120°**	**150°**	**180°**
Test 1	29°	71°	110°	146°	127°	169°
Test 2	2°	83°	113°	135°	144°	171°
Test 3	44°	82°	118°	157°	142°	178°
Test 4	38°	75°	99°	126°	138°	174°
Test 5	35°	41°	102°	137°	129°	159°
Test 6	21°	65°	99°	140°	136°	167°
Test 7	46°	82°	97°	128°	131°	192°
Test 8	25°	56°	98°	156°	168°	203°
Test 9	29°	68°	86°	117°	152°	163°
Test 10	17°	76°	83°	105°	149°	168°
False Protection Rate	40%	0%	0%	0%	0%	0%

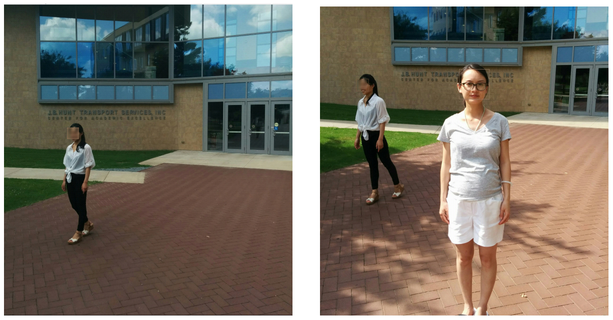

(a) Scenario 1 (b) Scenario 2

Fig. 11. The effect of blurring the stranger's face in Scenario 1 and Scenario 2

TABLE V
RUNNING TIME OF BLURRING FACES

Distance	**Blur Radius**		
	10	**20**	**30**
5 meters	4ms	7ms	13ms
10 meters	2ms	4ms	7ms

E. Cost Evaluation

1) Communication Delay: This part evaluates the round-trip delay from the time the photographer sends out a notification to the time he receives a request from the stranger. In the tests the stranger stands 5 meters and 10 meters away from the camera, and 10 tests were run for each distance. The average delays are 188ms and 193ms when the distances are 5 meters and 10 meters, respectively. Hence, the communication delay is short.

2) Running Time of Blurring Faces: This part evaluates the time needed to blur a face on a mobile phone. Two factors affect the time, the distance from the stranger to the camera and the blur radius of Gaussian Blur. The distance has an effect since it affects the size of the blurring area. If the blur radius is larger, more computations are needed for blurring each pixel of the blurring area. In these tests, we set the distance as 5 meters and 10 meters, and set the blur radius as 10, 20 and 30. Table V shows the results, where each data point is the average of 5 tests. When the distance increases, the running time decreases. This is because longer distance means smaller face in the photo and hence smaller blurring area. When the blur radius increases, the time increases due to the higher computation load. In all these cases, the running time of blurring faces is very short.

3) Power Consumption: To measure the power consumption of our system on the phone, we utilize a widely used App called PowerTutor [18] which can accurately monitor the power consumption of different Apps. We compare the consumption of our system with Google Maps and Chrome.

First, to evaluate our power consumption when no photos are taken, we tested these Apps running in the background for 5 minutes. Table VI shows their average power consumption. It can be seen that PrivacyCamera consumes much lower power than the other two Apps.

TABLE VI
POWER CONSUMPTION WHEN RUNNING IN THE BACKGROUND

	Google Maps	**Chrome**	**PrivacyCamera**
Average Power (mW)	25	32	10

Then we measure the power consumption of one conversation between the photographer and the stranger (including sending notification and receiving blurring request) and blurring one face on the phone. For comparison, we also measured the power consumption of visiting one web page in Chrome and searching for one location in Google Maps. Table VII shows the results, where each data point is the average result of 10 tests. In our system, each communication conversation only consume 0.12J, which is the lowest among the tested operations. The power consumption of blurring one face is 7.5J. Based on this number, a fully-charged battery (3.8V, 2300 mAh) of Nexus 5 phone can support the blurring of 4195 faces before being depleted. Thus, the power consumption is low. We noticed that the power consumption for blurring one face is higher than visiting one web page and searching one location. However, users usually do not take photos as often as they visit web pages and searching locations. Thus, we expect that the overall power consumption of PrivacyCamera should be lower than Chrome in practice.

TABLE VII
POWER CONSUMPTION WHEN RUNNING IN THE FOREGROUND

Test Application	**Operation**	**Average Energy Usage(J)**
Google Maps	Search for 1 location	2.4
Chrome	Visit 1 web page	1.4
PrivacyCamera	Conduct 1 conversation	0.12
PrivacyCamera	Blur 1 face	7.5

V. RELATED WORK

Jung and Philipose [19] propose a method to protect video privacy. The wearable camera will stop recording a person when it detects that the person is making certain gestures, e.g., waving hands. *MarkIt* [20] detects the sensitive objects predefined by users in videos and covers the sensitive objects with markers before releasing the video to third-party applications. Jana et al. [21] design an OS abstraction *Recognizer* to enforce fine-grained access control in augmented reality system. It can reduce the quality of raw sensor data when third-party applications request to access it. A similar system is SemaDroid [22]. Jana et al.[23] implement a privacy protection layer to restrict untrusted applications to access input data from perceptual sensors. For example, a person's face sketch can be transformed depending on different privacy levels. Privacy-preserving crowdsensing schemes [24], [25], [26] can preserve anonymity for participant-contributed data including photos, but they do not protect the privacy of strangers included in photos.

Schiff et al. [27] propose a system to detect persons that wear special tracking markers and block their faces from photos. However, people must wear special markers beforehand which does not apply to our considered problem. Bo et al. [28] design a protocol to protect the privacy of people being photographed based on people's privacy desires, which are contained in a physical tag. In their approach, however, people are required to wear clothes with QR-code as privacy tags.

Wang et al. [29] propose an approach to protect people's privacy based on the recognition of their visual fingerprints in images or videos, including motion patterns and visual appearance (e.g., clothing color). However, it requires people to upload their visual fingerprints to the server whenever their visual fingerprints such as clothing are changed, which needs intensive intervention by people. Moreover, their approach relies on a server to do the detection which does not fit our scenario. Templeman et al. [30] prevent private images from being shared with others based on attributes extracted from the image such as location and content. *PlaceAvoider* [31] is a system that can notify the photographer when an application is going to capture images in sensitive areas (e.g., bedroom). Pidcock et al. [32] propose a system to notify bystanders of nearby mobile sensing activities. Tan et al. [33] propose a system to protect photo privacy in Android phones. The system can recognize the photos that contain persons known to the phone owner, and denies third-party applications to access these photos. However, none of the above approaches can be applied to protect stranger's privacy in our scenarios.

VI. Conclusion and Future Work

This paper proposed a system PrivacyCamera to protect strangers' privacy who are accidentally included in a photo taken by mobile phones. The system can notify nearby strangers of the possible inclusion in a photo and allow them to decide if to blur their faces in the photo. We designed techniques to detect if a stranger requesting face blurring is in the photo or not based on GPS locations. We implemented a prototype system, and evaluated the system's performance and cost through experiments. Evaluations show that the system can accurately detect the stranger and blur his face to protect his privacy.

This work made an initial step towards protecting strangers' privacy when taking photos by mobile phones. Many open issues still remain. Future directions include considering more complex scenarios with multiple strangers in the photo and multiple strangers who are not in the photo but nearby.

Acknowledgment

Wei Gao is supported in part by NSF grants CNS-1456656 and CNS-1526769.

References

[1] N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell, "A survey of mobile phone sensing," *Communications Magazine, IEEE*, vol. 48, no. 9, pp. 140–150, 2010.

[2] T. G. Project, "Obscuracam: The privacy camera," https://play.google.com/store/apps/details?id=org.witness.sscphase1&hl=en.

[3] http://www.wi-fi.org/discover-wi-fi/wi-fi-direct.

[4] http://sourceforge.net/projects/gtk-gnutella/.

[5] A. S. Das, M. Datar, A. Garg, and S. Rajaram, "Google news personalization: scalable online collaborative filtering," in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 271–280.

[6] T. Jiang, H. J. Wang, and Y.-C. Hu, "Preserving location privacy in wireless lans," in *Proceedings of the 5th international conference on Mobile systems, applications and services*. ACM, 2007, pp. 246–257.

[7] M. Gruteser and D. Grunwald, "Enhancing location privacy in wireless lan through disposable interface identifiers: a quantitative analysis," *Mobile Networks and Applications*, vol. 10, no. 3, pp. 315–325, 2005.

[8] http://developer.android.com/reference/android/media/FaceDetector.html.

[9] L. Shapiro and G. Stockman, "Computer vision. chap. 12," *New Jersey: P rentice Hall*, pp. 137–150, 2001.

[10] http://developer.android.com/reference/android/media/FaceDetector.Face.html.

[11] A. Pyattaev, K. Johnsson, S. Andreev, and Y. Koucheryavy, "3gpp lte traffic offloading onto wifi direct," in *Wireless Communications and Networking Conference Workshops (WCNCW), 2013 IEEE*. IEEE, 2013, pp. 135–140.

[12] S. Trifunovic, B. Distl, D. Schatzmann, and F. Legendre, "Wifi-opp: ad-hoc-less opportunistic networking," in *Proceedings of the 6th ACM workshop on Challenged networks*. ACM, 2011, pp. 37–42.

[13] R. Meier, *Professional Android 4 application development*. John Wiley & Sons, 2012.

[14] G. M. Djuknic and R. E. Richton, "Geolocation and assisted gps," *Computer*, vol. 34, no. 2, pp. 123–125, 2001.

[15] P. Misra, B. P. Burke, and M. M. Pratt, "Gps performance in navigation," *Proceedings of the IEEE*, vol. 87, no. 1, pp. 65–85, 1999.

[16] http://www.gps.gov/systems/gps/performance/accuracy/.

[17] M. G. Wing, A. Eklund, and L. D. Kellogg, "Consumer-grade global positioning system (gps) accuracy and reliability," *Journal of forestry*, vol. 103, no. 4, pp. 169–173, 2005.

[18] L. Zhang, B. Tiwana, Z. Qian, Z. Wang, R. P. Dick, Z. M. Mao, and L. Yang, "Accurate online power estimation and automatic battery behavior based power model generation for smartphones," in *Proceedings of the eighth IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis*. ACM, 2010, pp. 105–114.

[19] J. Jung and M. Philipose, "Courteous glass," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*. ACM, 2014, pp. 1307–1312.

[20] N. Raval, L. Cox, A. Srivastava, A. Machanavajjhala, and K. Lebeck, "Markit: privacy markers for protecting visual secrets," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*. ACM, 2014, pp. 1289–1295.

[21] S. Jana, D. Molnar, A. Moshchuk, A. M. Dunn, B. Livshits, H. J. Wang, and E. Ofek, "Enabling fine-grained permissions for augmented reality applications with recognizers." in *USENIX Security*, 2013, pp. 415–430.

[22] Z. Xu and S. Zhu, "Semadroid: A privacy-aware sensor management framework for smartphones," in *Proceedings of the 5th ACM Conference on Data and Application Security and Privacy*. ACM, 2015, pp. 61–72.

[23] S. Jana, A. Narayanan, and V. Shmatikov, "A scanner darkly: Protecting user privacy from perceptual applications," in *Security and Privacy (SP), 2013 IEEE Symposium on*. IEEE, 2013, pp. 349–363.

[24] Q. Li and G. Cao, "Providing privacy-aware incentives in mobile sensing systems."

[25] ——, "Providing privacy-aware incentives for mobile sensing," in *Pervasive Computing and Communications (PerCom), 2013 IEEE International Conference on*. IEEE, 2013, pp. 76–84.

[26] ——, "Providing efficient privacy-aware incentives for mobile sensing," in *Distributed Computing Systems (ICDCS), 2014 IEEE 34th International Conference on*. IEEE, 2014, pp. 208–217.

[27] J. Schiff, M. Meingast, D. K. Mulligan, S. Sastry, and K. Goldberg, "Respectful cameras: Detecting visual markers in real-time to address privacy concerns," in *Protecting Privacy in Video Surveillance*. Springer, 2009, pp. 65–89.

[28] C. Bo, G. Shen, J. Liu, X.-Y. Li, Y. Zhang, and F. Zhao, "Privacy. tag: Privacy concern expressed and respected," in *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*. ACM, 2014, pp. 163–176.

[29] H. Wang, X. Bao, R. Roy Choudhury, and S. Nelakuditi, "Visually fingerprinting humans without face recognition," in *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 2015, pp. 345–358.

[30] R. Templeman, A. Kapadia, R. Hoyle, and D. Crandall, "Reactive security: Responding to visual stimuli from wearable cameras," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*. ACM, 2014, pp. 1297–1306.

[31] R. Templeman, M. Korayem, D. Crandall, and A. Kapadia, "Placeavoider: Steering first-person cameras away from sensitive spaces," in *Network and Distributed System Security Symposium (NDSS)*, 2014.

[32] S. Pidcock, R. Smits, U. Hengartner, and I. Goldberg, "Notisense: An urban sensing notification system to improve bystander privacy," in *in PhoneSense*, 2011.

[33] J. Tan, U. Drolia, R. Martins, R. Gandhi, and P. Narasimhan, "Short paper: Chips: content-based heuristics for improving photo privacy for smartphones," in *Proceedings of the 2014 ACM conference on Security and privacy in wireless & mobile networks*. ACM, 2014, pp. 213–218.