Journal of Image Processing Theory and Applications (2021) Vol. 4: 1-12 **DOI: 10.23977/jipta.2021.41001**
Clausius Scientific Press, Canada **ISSN 2560-6239**

Privacy Preserving Smartphone Camera Tracking Using Support Vector Machines

Q. Memon, K. Al Shanqiti, A. Al Falasi, A. Al Jaberi, Y. Amer

UAE University, Al Ain, qurban.memon@uaeu.ac.ae

***Keywords*:** Camera detection; Camera tracking; Camera blocking; Privacy, Security; Machine Learning

Abstract: Due to easy access to recent hand-held technologies, privacy concern has also increased. One most common and prevalent form of privacy and intellectual property breach nowadays is use of smartphone camera. This work reflects an attempt for privacy protection in smartphone environment. The potential contribution from this study is to disable smartphone camera temporarily by tracking it. The key parts of the approach are camera lens detection and tracking in upper body area, and then disabling it. The detection stage involves camera calibration followed by image processing steps conducted on captured environment frame for camera lens detection using either by database search or support vector machines applied on published data of smartphone camera lenses. The disabling process is enabled using safe-to-environment laser(s) controlled by a hardware interface. For tests, an experimental system is built, and results show improvement in tracking using support vector machines with performance error less than 0.5%.

1. Introduction

With recent technology evolution, smartphone camera use is around each part of our lives. From a technical perspective, as cameras develop with upgrades in designs, resolution, and lower sizes with affordable price, so are its effects on society. The use of camera has grown exponentially with the advent of smartphone in the last twenty years, but it has put a huge impact on private, commercial, and public domain at a wider scale. While it is positive use can be appreciated as we consider technology for a brighter future, however, its negative use (especially without seeking permission) has produced a dark impact on invading privacy of people especially in social gatherings since maintaining privacy is considered a loved concept globally. With current environment that is full of technologies available either in software or hardware format, intruding one's private life has turned out to be easier and common form of activity. One of the worst breaches of privacy now-a-days is taking pictures or videos in social gatherings or photography-forbidden environment like (intellectual property) paintings in exhibitions, etc.

In public places, it has been noticed that each of us feel free as one pleases to take pictures and vides using smart phones. In sensitive areas like airports, nuclear facilities and military bases, this activity can be deemed as detrimental to national security. In private parties, though privacy is deemed of higher consideration, yet people feel free to make unauthorized videos and take pictures in events

such as weddings and similar gathering. These concerns lead to a bigger question. How can this threat be eliminated or at least reduced to an acceptable level? Is it possible at all? Shall we leave it to human beings, who will change themselves once they collectively realize that privacy is the basic right of each and every individual?

The idea in this study to find a solution that addresses this concern by developing a technology to minimize the damage to privacy, which can be built using a smartphone camera tracking and a means to disable it. This excludes verbally asking people not to take pictures in order to maintain mood of the environment. Rather, image-processing techniques around well-known camera specifications (such as size, shape, radii, etc.) can be devised to track the position of the smartphone in order it to disable it temporarily with a blurring or jamming approach.

Jamming approach has been discussed in literature at many places like [1], where smartphones are used for jamming wireless devices selectively using arbitrary waveforms in the 2.4 and 5 GHz bands. However, a malicious attacker can misuse a jamming source to disturb wireless networks within a set area or environment. In this method, pictures or videos once recorded cannot be transmitted, but there is no way to stop any one from taking unsolicited videos or pictures. For security and tracking in other fields, related works can also be found in [2-3].

Learning algorithms, nowadays, find remarkably preferential place in numerous demanding computer vision tasks. Face or object recognition is one of the important topics in current user-based or industrial products. In [4], the authors use deep learning methods namely local receptive fields (LRF) and convolutional neural networks to detect face liveliness in digital forensic environment. In another research, the authors [5] propose multi-camera people detector using machine learning to address multi-view people occupancy map in a several synchronized cameras with overlapping fields of view. Another example, where authors [6] use a model based on machine learning to identify camera source using interpretation of learning curves, for purpose of accuracy and better prediction. Due to recent interest in drone technology in daily lives, machine learning has also been investigated for safety and security issues due to loss of control, collision or in invading secured properties. In [7], the authors have introduced drone detection scheme using machine learning onboard drone camera system to infer location on image and vendor model.

In literature, a number of technological solutions have been reported [8-9], but they lack perfection, precision and are partial to handle challenges, as addressed in this work. Some of the recent research works [10-11] have reported limited success in disabling camera. The work in [10], for example, uses Near Infrared labelling dependent on camera, label recognition policy and an enforcement plan based on Android. The objects, who wear this label do not notice it, and the system enforces privacy based on label recognition, and a Gaussian blur. In [11], the authors demonstrate on-off lighting of LED in indoor environment to produce a pattern that interferes with smartphone camera sensor to build a vertical stripped effect on phone/video being taken by the camera. Obviously, as pointed by authors [11], multiple LED’s are needed to cover an entire area. The synchronization across LED’s is still an issue to be addressed.

In the next section, an approach is presented, which discusses calibration, upper human body detection, camera lens detection, tracking, and disabling. The section three discusses support vector machine model to learn camera lens specifications for camera lens detection. In section four, the experimental results are stated, followed by conclusions.

2. Proposed approach

In this section, an approach is proposed to detect camera lens using processing of the image frame and then detecting it using smartphone camera lens database. After camera lens detection, laser light is beamed at coordinates of detected camera lens. Each step requires a number of functions to be done

by related components. The logical flow of the proposed approach is shown in Figure 1, and various corresponding functions carried by respective components are discussed below.

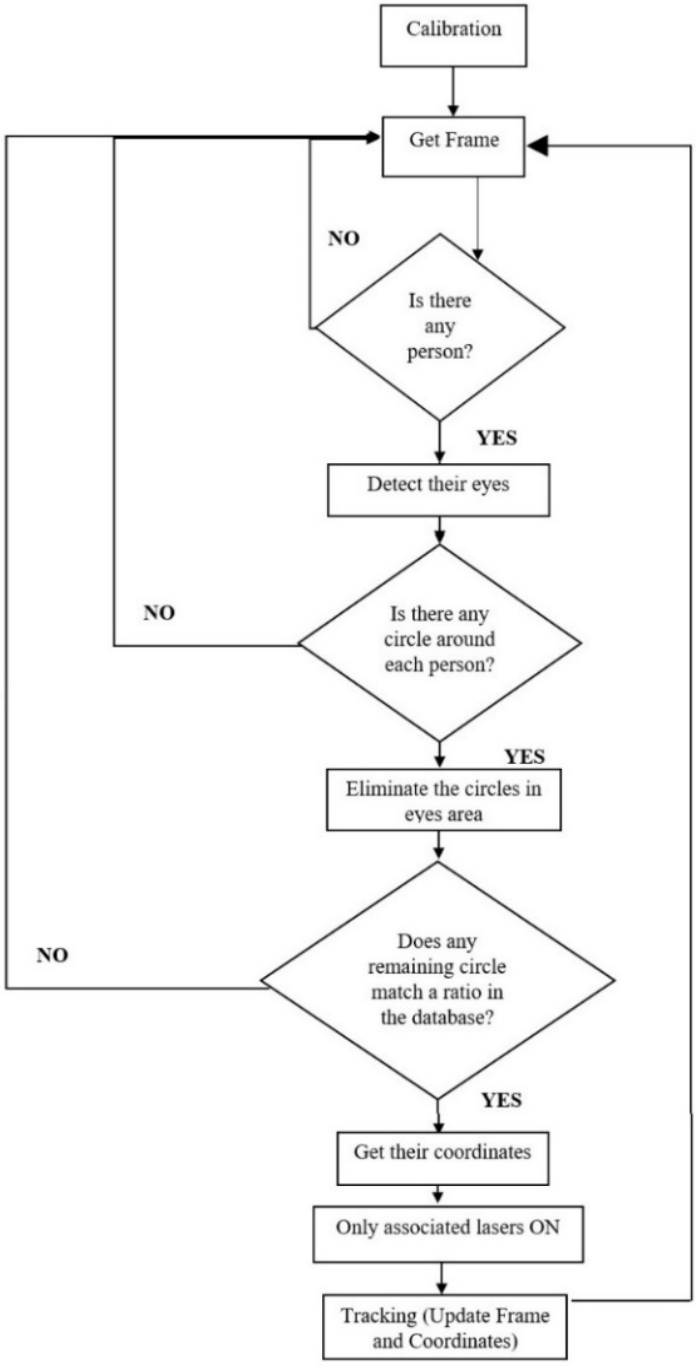

Figure 1: The system Flowchart

A. System specifications

Calibration: In order to get laser coordinates within captured frame, calibration step needs to be completed. These coordinates are set to be compared with circle coordinates (related to camera lens) for object detection. Since lasers used are red, a red object detection method was developed. The calibration using red object detection method implemented on non-red background. Once laser turns on, the camera takes a video of about 50 frames (as set in the program). The code has two loops: the

outer one detects red laser, and the inner loop bounds it for center coordinate calculation. Once set to variables, these coordinates are later converted to have ranges. This whole process repeats until all laser center coordinates are determined.

Camera: The camera used is a webcam that captures video up to 1280x720 pixels, with up to 5 Megapixels resolution. It is connected to the processor and continuously captures the frames for camera lens detection. The required resolution was estimated by using height of a person (set to 2m) and surroundings (set at $2m^2$) related to a camera lens ($1mm^2$), and calculated as:

$$Resolution = \frac{Object\ Size}{Size\ of\ the\ detail\ to\ be\ inspected} = \frac{2m^2}{1mm^2} = 2000px \quad (1)$$

Detection Process: A laptop computer with i7 processor was employed to get and process images to determine targeted coordinates related to those of smartphone camera. After a frame is captured, three parameters are determined to complete detection of smartphone camera lens. The first parameter is face, which involves identification of people in images or videos. The Viola-Jones-classifier [13], a very commonly known approach for computational speed, was used for face detection. This algorithm uses key concepts like Haar-like features, Adaboost machine-learning method, and cascade classifier for efficiency. Once face is identified, the area around face is enlarged to include upper body, as most of the time people hold camera up to that level in order to take a picture or video. After face detection, eyes are detected using the same function [13]. For this, the Hough transform [14] is used to detect circles in the upper body frame:

$$x = a - R\,cos(t); y = b - R\sin(t) \quad (2)$$

Where (x, y) are coordinates of the center. The intersection of (x, y) points in parameter space points to a circle of radius R centered at (a, b). The approach detects both dark and bright circles, but only bright ones are buffered in a variable '[cen, rad]', where 'cen' stands for array for detected circle centers and 'rad' for array for circle radii, respectively. Thus, in case 'x' circles are detected, the 'cen1' and 'rad1' will have 'x' rows for center and radii arrays, respectively. Next, any circles outside the bounding box covering upper body need to be removed, as these fall outside the area within which smartphones are typically held by people while taking videos. The second step involves eliminating circles in the eyes area to ensure safety of eyes. The third step removes circles outside the range of published ratio limits. This is accomplished by matching coordinates of the laser (obtained from calibration) with the remaining circles after second elimination step. The results are displayed in Figure 2. As discussed before, camera calibration is done once during initial stage, but continuous tracking requires repetitive detection.

Database Development: For purpose of eliminating circles outside the range of published ratio limits, the tables are made for a range of smartphones that the system is deemed to track. The Table 1 displays camera area of well-known smartphones. This information is publicly available on Internet published by manufacturers. The Table 2 displays ratios of these camera areas to corresponding bounding box areas determined in this work.

(a)  (b) 

Figure 2: The detected circles (a) Before elimination (b) After elimination

Table 1: Camera area of well-known smartphones

	User 1			User 2			
	Area of box	Area of Lens	Ratio	Area of box	Area of Lens	Ratio	Average of ratios
iPhone3	$(405)^2$	$\Pi\ (9.8)^2$	543.9	$(416)^2$	$\Pi\ (9.9)^2$	559.7	551.84
iPhone4	$(389)^2$	$\Pi\ (8.3)^2$	695.8	$(363)^2$	$\Pi\ (7.8)^2$	680.6	688.24
iPhone5	$(375)^2$	$\Pi\ (7.9)^2$	702.9	$(411)^2$	$\Pi\ (9.8)^2$	559.8	620.78
iPhone6	$(430)^2$	$\Pi\ (9.9)^2$	592.1	$(420)^2$	$\Pi\ (10.2)^2$	538.6	565.37
iPhone7	$(405)^2$	$\Pi\ (9.2)^2$	619.5	$(325)^2$	$\Pi\ (6.9)^2$	700.1	659.81

Table 2: Ratios (Area to Box Ratios)

	Height (mm)	Width (mm)	Camera Area (mm)	Average of ratios
iPhone 7 plus	158.2	77.9	28	440.135
iPhone7	138.3	67.1	28	331.43
iPhone6s plus	158.2	77.9	28	440.135
iPhone 6s	138.3	67.1	28	331.426
iPhone SE	123.8	58.6	28	277.93
iPhone 4	115.2	58.6	28	241.097
iPhone 3	115.5	62.1	28	256.162
Samsung Galaxy-S4	136.6	69.8	31	307.57
Samsung Galaxy-S5	142	72.5	31	332.096
Samsung Galaxy-S6	142.1	70.1	28	355.76
Samsung Galaxy-S7	143	70.5	26	387.75

Architecture and Interfacing: Based on this previous discussion, the prototype was developed as shown in the Figure 3*b*. For prototype testing, a matrix of two lasers was built and interfaced to the processor through a microcontroller board, while webcam was interfaced through a USB port, as shown in Figure 3*a*. Once frame with detected circles is available, the lasers of corresponding (calibrated) coordinates switch 'ON' through interface.

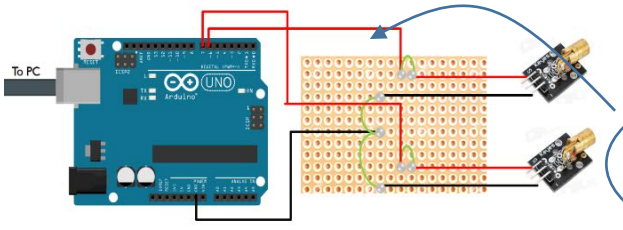

Figure 3*a*: Detailed laser interface

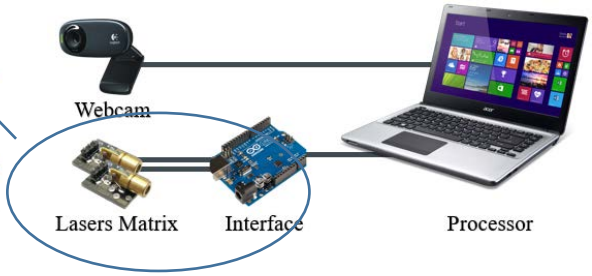

Figure 3*b*: Connections to lasers through PCB

Laser Matrix: As a prototype, a 2x1 matrix corresponding to two coordinates was used to jam the source. Based on safety considerations, class 2 lasers were used since the maximum output power (allowed in GCC region countries) does not exceed 1mW [12]. The wavelength of the class 2 lasers varies for Red (620-780 nm) and for Green (492-577 nm), but their use depends on success rate of blurring photos, safe use, cost, and path visibility. For laser positioning, there are some constraints. The lasers are to be placed perpendicular to frame surface i.e., parallel to the ground, as it is required

to be independent of the depth. This is so because a very small laser angle deviation is quite noticeable at a larger distance from the laser, and this will cause frame coordinates vary with depths. For further details on facial feature recognition and tracking, reader is referred to works reported in [15-17].

3. Machine Learning for Camera Detection

The presented data, in Table 2, is that of a particular specification of various smartphone cameras that depict the area ratio of the cameras with respect to the larger pictures taken to detect the smart phone cameras in a larger picture. Each of the data vector consists of only one feature, the areas ratio. The labels for each of the data vector can be assigned ranging from one to eleven. Though data is limited to eleven (11) different camera lens specifications, but for practical purposes, the data size can be increased to accommodate any number of cameras, and a supervised learning such as multilayer perceptron, radial basis function or support vector machine algorithm [18], etc. can be applied to train on such a large database.

This input data is also the training data used for training of the support vector machine built for classification in this project. The training algorithm used here is primarily the one-vs.-one multiclass support vector machine (SVM) algorithm. It uses binary learner. Since we have eleven (11) classes, which means that we have 55 binary learners corresponding to ((K−1)/2=) 55 different combinations, where each cell of the binary learner in its own is a classifier. Mathematically, it is defined as [19]:

$$Q = \frac{1}{2}\|w\|^2 - \sum_i \propto_i [y_i(\bar{w}.x_i + b) - 1] \quad (3)$$

where $\propto_i$, x_i, y_i, b, and w represent multiplier constants, input, output class (+1/-1), bias, and resulting weights once function Q is minimized such that [19]:

$$\sum_i \propto_i y_i = 0, \; where \; 0 \leq \propto_i \leq C; \quad C \; is \; a \; constant \quad (4)$$

are satisfied. The resulting weights and the decision are computed as [18]:

$$\bar{w} = \sum_i \propto_i [y_i(x_i)]; \quad \left(\sum_i \propto_i y_i \bar{x}_i.\bar{u} + b\right) \geq 0, \; then \; + class \quad (5)$$

The coding matrix is a $K \times L$ matrix, where K is the number of classes and L is the number of binary learners. The algorithm functions in a way such that the data of each class is compared with one another two at a time till eventually each class becomes unique. Its classifying architecture is displayed in Figure 4. The bias in set to zero by default by the classification learner tool. The SVM network has 1 input node, 11 hidden nodes and 1 output node, total of 2 layers (since input layer is not considered as no computations takes place there).

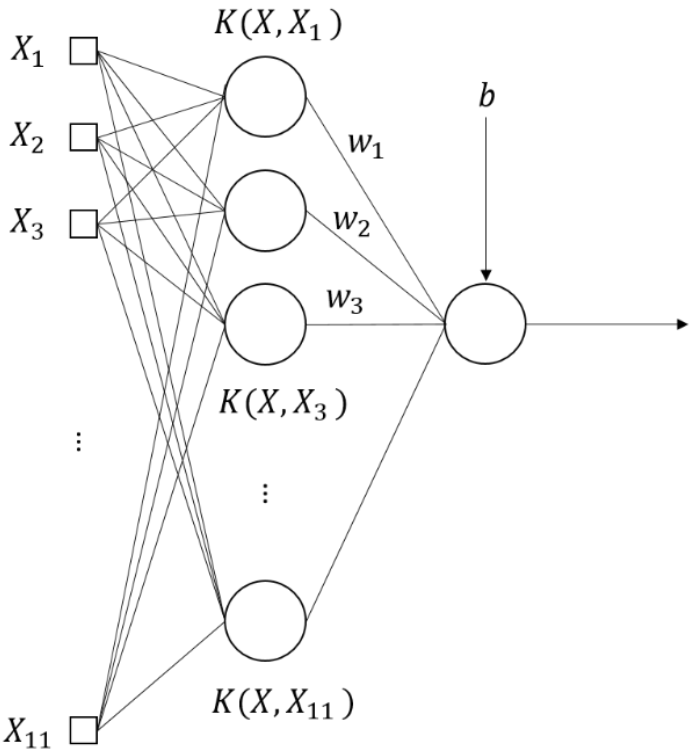

Figure 4: The architecture of the SVM used in this case

The model used were Gaussian, linear, and polynomial kernel functions. For purpose of training, we generated 100 random samples that range from 200-450 area ratios. The samples were ±3% of the originally values to account for rounding and measurement errors. The number set was generated using 'rand' function, converted to a table, and then inputted to the model. After the Classification Learner Tool was launched and the training table was inputted, the training results showed 100% accuracy for Gaussian, polynomial order 2, 3, as well as for 4, but poor results for linear function, as shown in Table 3. The resulting weights, support vectors, and alphas are shown in Table 4, Figures 5, and 6, respectively.

Table 3: Error loss versus different kernel functions

Kernel	**Error loss**
Gaussian	0%
Linear	12.05%
Polynomial (order 2)	0%
Polynomial (order 3)	0%
Polynomial (order 4)	0%

Table 4: The weights for the 55 binary learners

1	0.166667	12	0.166667	23	0.166667	34	0.25	45	0.25
2	0.166667	13	0.166667	24	0.166667	35	0.166667	46	0.166667
3	0.166667	14	0.166667	25	0.166667	36	0.166667	47	0.166667
4	0.166667	15	0.166667	26	0.166667	37	0.166667	48	0.166667
5	0.166667	16	0.166667	27	0.25	38	0.166667	49	0.25
6	0.166667	17	0.166667	28	0.166667	39	0.166667	50	0.166667
7	0.166667	18	0.166667	29	0.166667	40	0.25	51	0.166667
8	0.166667	19	0.25	30	0.166667	41	0.166667	52	0.25
9	0.166667	20	0.166667	31	0.166667	42	0.166667	53	0.166667
10	0.25	21	0.166667	32	0.166667	43	0.166667	54	0.25
11	0.166667	22	0.166667	33	0.166667	44	0.166667	55	0.25

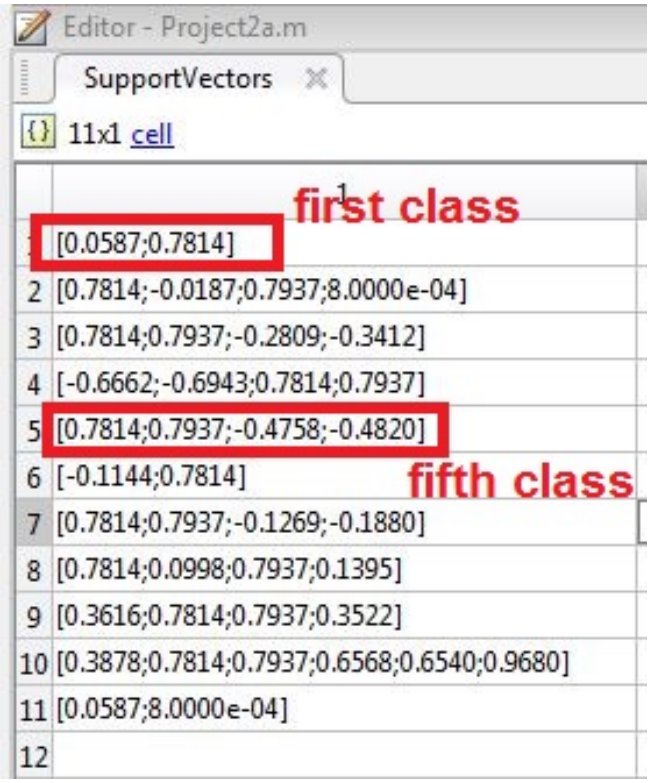

Figure 5: Resulting support vectors

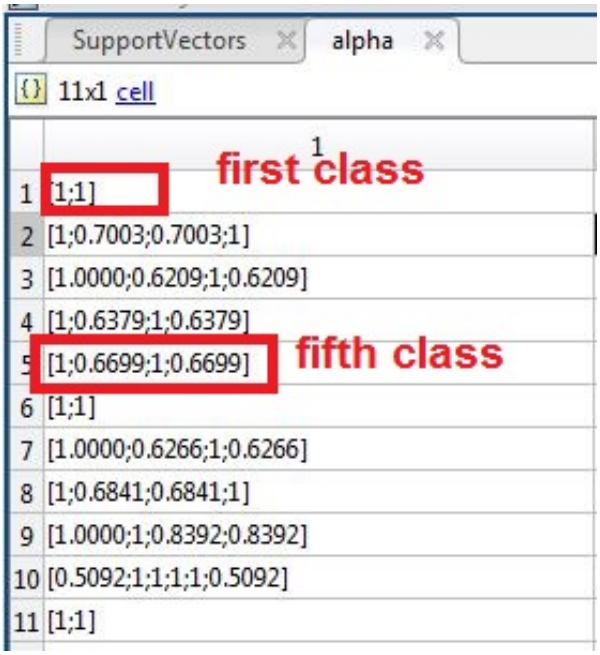

Figure 6: Resulting alphas ‘α’

For purpose of verification, the resulting classification due to Gaussian kernel is shown in Table 5. For testing purposes, samples outside the training set, but within the same range were used. Effectively, all binary SVM classifiers were predicted, and then the one with highest number of votes was chosen as the classifier for that data sample. The results matched to those of training, which shows 100% accuracy for testing range selected.

Table 5: The samples after classification for Gaussian kernel

1	2	3	4	5	6	8	9	10	11
234.6561	255.953	269.6246	305.4403	309.6861	322.4411	332.1333	346.316	376.5115	428.3343
235.4716	257.2442	271.4598	305.4403	309.6861	322.4411	332.1333	346.3169	377.3412	428.344
237.3235	260.8812	277.8038		311.3966	322.4411	332.1333	350.4955	385.7831	428.9339
238.0945	262.771	279.2749		311.3966	324.591	332.6994	354.0112	387.0379	429.2984
239.4033	263.5705	284.2807		312.6354	324.591	332.6994	358.0898	387.8168	432.3159
240.5456	263.7738	285.0964		312.6354	324.591	332.6994	361.5783	388.4323	433.4983
240.6529				317.3477	326.4893	334.5856	363.5198	388.6717	433.5027
241.4122				318.3222	326.4893	334.5856	363.7745	389.3001	437.5555
242.7967					326.4893	334.5856	363.8695	389.435	439.2917
246.7182						336.7204	363.9352	391.3792	439.3767

249.1488						336.7204		394.7918	439.8229
						336.7204		398.0518	439.8731
						336.8039		398.5711	439.936
						336.8039		398.8	441.2221
						336.8039			442.6482
						337.4309			
						337.4309			
						337.4309			
						341.9554			

Table 6: Percentage errors

		Detection and Database match		Detection and Machine Learning	
Test No.	Circles Detected (before filters applied)	Circles (excluding camera lens) after elimination	% Error	No. of circles (excluding camera lens) left after elimination	%Error
1	256	4	1.56	1	0.44
2	174	1	0.57	1	0.57
3	287	3	1.05	1	0.35
4	429	5	1.16	1	0.23
5	99	0	0.00	0	0
6	126	2	1.58	0	0
7	328	0	0.00	0	0
8	189	1	0.53	1	0.53
9	64	0	0.00	0	0
10	211	2	0.95	1	0.47
Average % Error			0.74		

4. Results and Discussions

For calibration, a program was coded in Matlab and tested independently. The calibration was conducted during various lighting environments ranging from normal to dimmed lighting. For testing, various trials were conducted on different subjects holding the smartphone camera. For circle detection, ten (10) consecutive tests on frames were run to calculate the smartphone camera coordinates and are shown in Table 6. As a final test, microcontroller was integrated with laser matrix and connected to the main detection code. Once the setup runs, the camera coordinates are compared with the lasers' coordinates, and the corresponding matched laser turns ON. The test was repeated many times with results similar, as shown in Table 6. It should be noted that error percentage and detection duration were found to be about 0.74% and two seconds. However, this error was of type

false positive. In all testing cases, it never was a false negative, which means that undetected smartphone cameras were there. Using machine learning on the database, this error was reduced to 0.25%. As for as two-seconds delay time for detection is considered, one can always improve on coding process. At the moment, we are targeting duration to be less than a second. But since a person taking a picture does not move enough, a delay between 1 and 2 seconds may be still acceptable in certain environments.

5. Conclusions

The main contribution was to discourage unsolicited use of smartphones in a private environment. This was accomplished using processing steps that included upper body detection, circle detection and its corresponding coordinates in captured frames. The processing code interfaced with laser matrix through microcontroller to point laser beam(s) at detected circle coordinates. The testing generated an error rate of less than 1%.
The work presented in this work can be enhanced in many aspects. For example: (i) having a denser matrix will have lasers closer to each other to eliminate blind spots. The coordinates used in the testing were continuous but can be made discrete and then a rotating motor can be deployed to point at detected circles. (ii) using a high resolution and wide range camera can help to have precise radii (and hence the coordinates) of the detected circles (iii) the code can be modified to include night vision to account for dark operation environment.

References

[1] *M. Schulz, F. Gringoli, D. Steinmetzer, M. Koch, and M. Hollick, "Massive reactive smartphone-based jamming using arbitrary waveforms and adaptive power control," ACM Conference on Security and Privacy in Wireless and Mobile Networks, pp. 111-121, doi: 10.1145/3098243.3098253*

[2] *Q. Memon, "A New Approach to Video Security over Networks," International Journal of Computer Applications in Technology, 25 (1), 72-83, 2006.*

[3] *Memon, Q., Khoja, S., "RFID–based Patient Tracking for Regional Collaborative Healthcare," International Journal of Computer Applications in Technology, 45 (4), pp. 231-244, 2012*

[4] *Y. Akbulut, A. Şengür, Ü. Budak and S. Ekici, "Deep learning-based face liveness detection in videos," International Artificial Intelligence and Data Processing Symposium, Malatya, 2017, pp. 1-4, doi: 10.1109/IDAP.2017.8090202.*

[5] *T. Chavdarova and F. Fleuret, "Deep Multi-camera People Detection," IEEE International Conference on Machine Learning and Applications, Cancun, 2017, pp. 848-853, doi: 10.1109/ICMLA.2017.00-50*

[6] *V. U. Sameer, A. Sarkar and R. Naskar, "Source camera identification model: Classifier learning, role of learning curves and their interpretation," International Conference on Wireless Communications, Signal Processing and Networking, Chennai, 2017, pp. 2660-2666.*

[7] *D. Lee, W. Gyu La and H. Kim, "Drone Detection and Identification System using Artificial Intelligence, "International Conference on Information and Communication Technology Convergence," Jeju, 2018, pp. 1131-1133., doi: 10.1109/ICTC.2018.8539442*

[8] *The Pirate Eye Anti-Piracy Solution. (n.d.). Retrieved from http://www.pirateeye.com/pirateeye/*

[9] *Kate Greene, (June 22, 2006). MIT Technology review, a prototype device seeks out cameras and blocks them from taking pictures and video. Retriever From https://www.technologyreview.com/s/405968/lights-camera-jamming/*

[10] *D. Li, D. Guo, W. Han, H. Chen, C. Cao and X. S. Wang, "Camera-Recognizable and Human-Invisible Labelling for Privacy Protection," International Conference on Mobile Ad-Hoc and Sensor Networks, Hefei, 2016, pp. 365-369., doi: 10.1109/MSN.2016.066*

[11] *J. Hsu, "LiShield Can Block Smartphone Cameras for Privacy's Sake," IEEE Spectrum, Oct. 2017.*

[12] *T. Wheatley, "Laser pointer prohibition: improving safety or driving misclassification," International Laser Safety Conference, 2013, pp. 48-54*

[13] *Viola, P., Jones, N., "Robust Real Time Object Detection," International Journal of Computer Vision, pp. 137-154, 2004.*
[14] *Hough, P.V.C. Method and means for recognizing complex patterns, U.S. Patent 3,069,654, 1962*
[15] *Q Memon, "On assisted living of paralyzed persons through real-time eye features tracking and classification using Support Vector Machines," Medical Technologies Journal, 3 (1), 316-333, 2019.*
[16] *R. Kumar, M. Geetha, "Deep Learning Model: Emotion Recognition from Continuous Action Video," In Data Science: Theory, Analysis, and Applications, 305-322, 2019*
[17] *S Pal, S Shaw, T Saurabh, Y Kumar, S Chakraborty, "A Study and Analysis of an Emotion Classification and State Transition System in Brain Computer Interfacing," In Data Science: Theory, Analysis, and Applications, 225-248, 2019*
[18] *Al-Kassim, Z., Memon, Q., "Investigating camera calibration for eye tracking of the physically challenged," ICFST 2016, MATEC Web of Conferences, DOI: 10.1051/matecconf/2016590800.*
[19] *Cortes, C.; Vapnik, V., "Support-vector networks", Machine Learning, 1995, 20 (3): pp. 273–297. doi:10.1007/BF00994018.*