Providing visual privacy and forgery prevention in the case of using low-end cameras

Rouzbeh Froozandeh, Nasrin Eshraghi Ivari
Faculty of Geodesy and Geomatics, Faculty of Computer Science
University of New Brunswick
nasrin.eshraghi@unb.ca, rforouza@unb.ca

***Abstract*— IoT cameras are devices which sharing evidence through video streaming in this era. With the lot of benefit for public, threatens the privacy of individuals. Although, one main challenge is to protecting visual privacy by try to keep video authenticity. post-process blurring would give a new view for posterior fabrication. while the real-time blurring results has a poor quality, low-frame-rate videos owing to the limited processing power. This work introduces Pinto, a software-based solution uses IoT cameras to generating privacy-protected, forgery-proof, and high-frame-rate videos.**

this framework, at the fast rate records stream of videos which these are realtime. Pinto lets post-processing to share videos for privacy protection. while keeping their primary, signatures valid astonishingly after the post blurring, guaranteeing no content forgery since the time of their recording. The software is implementable in cameras of today. The first type of this software is implemented on three different embedded devices, each deployed in application which is specific. vehicular, and aerial the production of privacy-protected, forgery-proof videos.

I. INTRODUCTION

personalized video cameras are widespread in these years because of inexpensive, network based and easy use for surveillance. these kind of video recording tools are very useful in real-time applications. figure 1 shows on-site security cameras to in-vehicle dashboard cameras (or dashcams), and aerial drone cameras. These types of cameras can record every moment in their sight , and make a valuable pictures. On the other hand, evidence can be captured and shared by the network-enable cameras[1].

Sharing of video evidences can create a powerful source for individual privacy. But many sensitive things like licence plate of the vehicle, face of the people and in some cases, home or work address can be captured. This issue cases concern about personal privacy in virtual world[**?**], and it is a barrier for sharing and capturing moments such as sharing or release of private surveillance videos.

(a) On-site security cam (b) In-car dashcam (c) Aerial drone cam

Fig. 1. Personalized video surveillance cameras

Decrease impact of the privacy in sharing videos have some requirements: first, videos have a good qualities, second, some parameters should authenticated like data integrity and recording time, third,videos have visual privacy protection and the degree of visual privacy depend on the circumstances of requests.

In the work "Pinto: Enabling Video Privacy for Commodity IoT Cameras" [2] which we have chosen it as an anchor paper, these above requirements are developed. These problems are not isolated but interconnect with one another under two limited device capabilities and content unaware hashing rendering existing solutions inadequate.

Video forgery is very easy with the software using for editing videos and this is a very big challenge because makes unreliable videos. On the other hand, real time blurring is another issue in some captured frame such as sky one and this kind of videos can possible with the special hardware. One important thing is the quality which it is depend on the capability of the device.

Pinto, a software-based framework for producing privacy-protected, forgery-proof, and high-quality

*This work is a report of the anchor paper :Pinto: Enabling Video Privacy for Commodity IoT Cameras

videos using low-end IoT cameras
cite.

Pinto make three things better. First, what is expensive in filming a real time is object detection, pixelation of a frame area is lightweight. Pinto performs fast pixelation of frames. Second, frame division into subimage blocks for visual privacy. Third, pixelation with hashing for forgery prevention usage.

On the other hand, Pinto has some advantages like : (1) real time and fast streaming video recording at a high frame rate. (2) the CPU-intensive object detection is executed only when video sharing is needed, saving the device power to do other things; (3) the post processing only permits pixelation for privacy protection while prohibiting any other modification to original videos; (4) the post pixelation enables post-decision on visual privacy upon request, flexibly determining the degree of visual privacy of stored videos at the time of their release; (5) video processing is done at the camera level, hence not requiring powerful back-end servers; and (6) it is a software-based solution immediately implementable in todays commodity IoT cameras.

Pinto was implemented in three different systems.the first one is 720 MHz1.2 GHz CPUs, and deploy them into the applications like : on-site security cam, in-car dashcam, and aerial drone cam. the measures show this platform provides: (i) privacy protection very strongly (ii) reliable authenticity verification and (iii) the high quality of videos to compare vwith HD ones.

II. DETAILED DESCRIPTION OF THE TOPIC

To achieve fine-grained visual privacy, a grid-based approach is applied which is referred to as Block-level operations. To do so, each frame is divided into sub-images (blocks). Pixilation process is applied to each block separately. The blocks can be divided into two separate categories: 1) blocks of sensitive objects and 2) blocks of non-sensitive objects.

The other key ingredient of Pinto is called Hash-Pixelation which is useful in both visual privacy protection and forgery prevention. Considering a sub-image block, H-Pixelation (Hach-Pixelation), starts with hashing the sub-image block. This procedure is followed by pixilation process in which the content is changed in the way that recognition of objects becomes more difficult (or impossible). As the final step, the created hash is embedded into the pixelated sub-image. To reduce the visual jarring, the initial 256-bit hash is distributed to the 16 bits of the first 16 pixels (16*16=256). This change is not perceptible by human eyes.

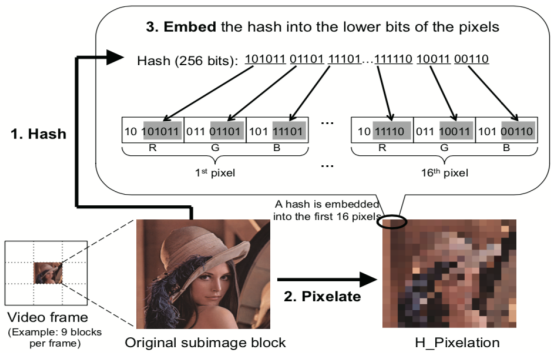


Fig. 2. h-pixelation process

Fig. 3. Block-level operation

The other key ingredient of Pinto is called Hash-Pixelation which is useful in both visual privacy protection and forgery prevention. Considering a sub-image block, H-Pixelation (Hach-Pixelation), starts with hashing the sub-image block. This procedure is followed by pixilation process in which the content is changed in the way that recognition of objects becomes more difficult (or impossible). As the final step, the created hash is embedded into the pixelated sub-image. To reduce the visual jarring, the initial 256-bit hash is distributed to the 16 bits of the first 16 pixels (16*16=256). This change is not perceptible by human eyes. the framework of the pinto is elaborated in the next section.

A. Framework

all the frames undergo real-time H-Pixelation. Block-level pixelation is done in real-time for each frame (while recording). The H-Pixelated video is then hashed - which is called p_digest- is sent to a server using a secure channel. P-digest is used in authenticity evaluating in the case that the video is shared. The sensitive blocks are also detected and pixelated through the server applications. At the same time, the original video (not pixelated) and the time-stamped p-digest file are stored in the local device.

Sensitive object detection and pixilation are done when videos need to be shared. While this is done as a

$$S \longrightarrow A: \quad T^u_{cur}, \{H(p\_digest_u | T^u_{cur})\}_{K_S^-}.$$

Fig. 4. p-digest

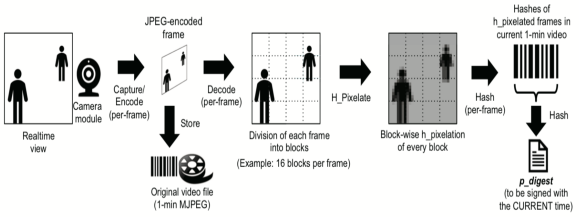


Fig. 5. Real-time processing in the camera

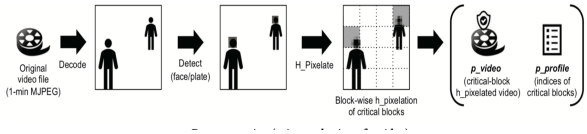


Fig. 6. Post processing procedure

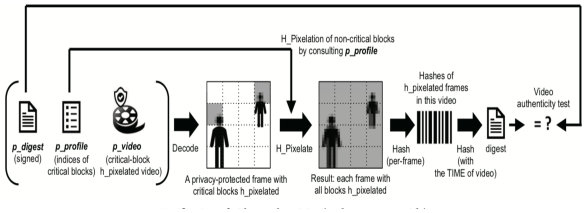


Fig. 7. Verification of video authenticity

post-processing step, the fast computational resource is not required and also the type object detection techniques are independent of Pinto. For each frame, the blocks that contain sensitive contents are called critical blocks. These blocks are kept in a compact format called p_profile. The privacy protected video that has undergone Block-level H-Pixelation is called p_video (Figure 1).

When someone requests the verification of a videos authenticity, along with the p_video, p_digest and p_profile are made available. If the h_pixelation of the non-critical blocks successfully restores the fully h-pixelated version, it indicates that no forgery has occurred. The total real-time processing time is reduced by minimizing the time required to generate p_digest. Although this process consists of h_pixelation of the whole blocks, generating the p_digest is still a lightweight process while the cumbersome process of object detection is not applied. The resulting h_pixelated frames are hashed and then the next frame is received from the camera and processed on.

B. Procedural Description

P_digest file is produced on the fly and in real-time. Figure 3 shows the operations that are done in real-time. As illustrated, the recorded video is fed to two different parallel processes. The frames are recorded as an original video file and at the same time, they are used in p_digest generation. When the video u (1-min) is recording, the local device (A in this case), generates p_digestu by hashing all the frames and sends it to a trusted server. The server S generates the p_digestu as figure 4:

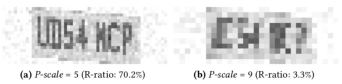
Where T_curû and K_S^{+} are the current time and S is the private key.

Post Despite p_digest, p_video, and p_profile are generated by accomplishing post-processing tasks. Critical blocks in each frame are h-pixelated to produce p_videou (Figure 4). The indices are also logged in the compact structure called p_profileu.

When a video is requested, h_pixelation is performed on the requester side for all non-critical blocks and also generates a per-frame hash of the resulting mentioned h-pixelated frames. The time and integrity of the video (called authenticity of the video) is verified if the video matches the p_digestu certified by the server with known public key K_S^{+}.

C. Design Decisions

Another consideration is determining the pixelation intensity which affects the frame quality and privacy at the same time. There should be a balance between frame quality and visual privacy, in the sense that increasing one would affect the other one. As an example, using almost the same pixel value for the whole pixels fulfills the privacy in all cases but questions the usability of the videos. As it is said by the authors The intenser the pixelation the poorer (/the stronger) the human-perceived video quality (/privacy protection). It is preferred that the privacy is achieved by applying the minimal changes in the frames.



(a) *P-scale* = 5 (R-ratio: 70.2%) **(b)** *P-scale* = 9 (R-ratio: 3.3%)

Fig. 8. pixelation of license plate

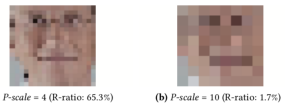


Fig. 9. Pixelation of a human face

The In-frame block count also needs to be determined. By limiting the block counts per frame, the real-time h-pixelation process is faster.

D. Evaluation

1) Protection of Visual Privacy : The pixelation intensity plays a key role in protecting the visual privacy. This measure is called P-scale which indicates the degree of resolution deduction in the frames. As an explanation, P-scale X indicates the resolution is changed in the sense that it is scaled downed by factor X2. In this way, pixel values are replaced by an average achieved from a square block of X2 pixels. This process is lossy since it is non-invertible. To investigate the effectiveness of different levels of pixelation, a deep-learning power attacker is used. It is assumed that the video requester is a potential attacker equipped with visual recognition power. +100K images of UK license plates, +334K facial photos and +50K pictures of address signs were collected for the tests. The sample images have undergone pixelation regarding different P-scale values. The datasets are categorized by their types (plate/face/sign) and P-scale. A deep neural network model is trained by using part of the datasets as training data. The recognition success ratio (R-ratio) -which indicates the probability that the attacker recognizes the pixelated object- was determined for the testing data. Figure 8 shows the results for different types of objects. The figures show the R-scale against P-scale for a different number of training datasets by object type (n). The R-scale increases with the increase in the volume of training data. However, we can also see the diminishing returns showing that the datasets are big enough to train the deep-learning models. The results show that the recognition decreases with P_scale and it drops to near zero when P_scale is around 10. This experience suggests that P-Scale should be considered higher than 11 so that the privacy of plates, signs, and faces are guaranteed.

The other important factor in object recognition and privacy is the object-size. The training and testing data are divided into three groups by size: large (larger than 100*100 pixels), medium (smaller than large objects but bigger larger than 25*25 pixels) and small. Figure 8 shows R-ratio by object size. It indicates that the larger the object size the higher the R-ratio. It can be interpreted that a large object by having more pixels-are easier to recognize. Thereafter, larger objects need a higher degree of pixelation to be unrecognizable.

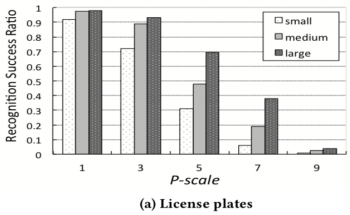


Fig. 10. Recognition success ratio (R-ratio) against P-scale

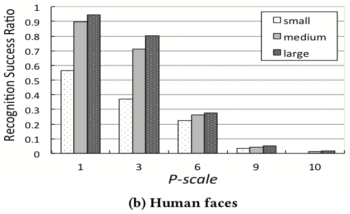


Fig. 11. Recognition success ratio (R-ratio) against P-scale

2) Prevention of Content Forgery: To check the originality of the video, the real-time signature of h_pixelation is used. Different kinds of forgery are considered in this study (some examples can be found in Figure 9) including copy-move, splicing, erasing, lighting, collision, and pixelation. Many video editing software is able to do mentioned forgery. OpenCV has been used in this study for automating forgery process.

Forgery success ratio (F-ratio) is defined as the probability of generating forgery that cannot be detected. Table 1 summarizes results for various types of forgery on HD frames. The table also compares the results of h_pixelation methods with the cases in which only regular pixelation or hash-only is applied. The results show the essence of using the combination of hashing and pixelation process.

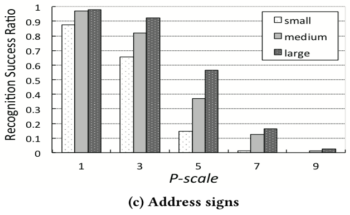


Fig. 12. Recognition success ratio (R-ratio) against P-scale

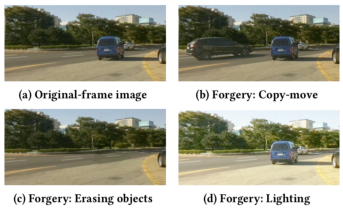


Fig. 13. Examples of forgery types

Forgery type	Forgery success ratio (F-ratio)		
	No Pixelation (Hash-only)	Pixelation (P-scale: 12)	H_Pixelation (P-scale: 12)
Copy-move	0%	0%	0%
Retouching	0%	0.3%	0%
Collision	0%	100%	0%
Pixelation	100%	0%	0%
Splicing	0%	0%	0%
Erasing	0%	0%	0%
Lighting	0%	0%	0%

Fig. 14. Forgery success ratio (F-ratio)

III. RELATED WORK IN THIS AREA

Lots of research is existed on a visual of privacy protection or forgery detection or prevention, but there is a little research can be founded on both.

Most prior works are based on visual privacy protection which rely on very powerful server or customized hardware for processing the streams of video from camera devices. In the papers [3], [4], [5] and [6] used FPGA based hardware to show high-accuracy face-detection.

Some Cameras[7] useMJPEG streams of Panasonics security cameras for transmitting servers for processing the videos in a real time manner. They also using object tracking.

Cardea[8] uses Samsung Galaxy Note camera as a client for real time recognition, and connects them via internet with a server.

Personalized video monitoring cameras have become important ones, and these devices are with the cost of low around $30 to $250 [9]. They have small and low functionality processors, and come on-board SD memory cards. These cameras continuously record around 1-min default time and store them inside the device. if the memory is full, the oldest replace with the new one and will delete the previous track. Now a days, many of these devices have a wireless interfaces which is located inside. this option is very useful in many IoT applications which need real time data.

The Existing techniques for forgery detection try to verify whether primary images have been deleted or not. these issue is categorized in two main group. The first one is digital watermarking[10] and fingerprinting.

In [11], showed that aerial vehicles have a lot of popularity among an ever growing community of amateurs as well as service providers. UAVs equipped with IoT devices, in delivering IoT services from great heights.

In the field of Iot, cloud based camera supervision systems are available for some environments like private one [12]. Although, surveillance is a very sensitive case which require privacy or confidentiality, authenticity, and availability of a systems.

In [13], the security of embedded systems are considered. However, based on the results of many recent analyses of individual firmware images, embedded systems acquired a reputation of being insecure. In this work the first public, large-scale analysis of firmware images were published.

based on the [14], using video coding in health care was an issue. In this work, an IoT surveillance system for health care monitoring is presented. this system consists of some important parts as sensors, actuators, and cameras. the topology used was mesh network. It was decided to be used as it provides important advantages. also,for the data compression and transferring the Constrained Application Protocol is used , and SHVC, the scalable High-Efficiency Video Coding is applied for video compression and transferring.

IV. THOUGHTS AND REMARKS

This study successfully demonstrates the advantages of using the proposed method and on the top of it h_pixelation. The advantage of using Pinto is that it can be used as a privacy solution when cheap cameras are in use and on-board computation is limited. With the growth up wearable cams, IoT technologies, dash-cams, and etc., a framework like Pinto can bridge the gaps in the privacy area. The proposed method depends on the fast real-time pixelation and postpones the resource-consuming object detection operations. Another innovation is block-wise processing which is used in combination with h_pixelation to prevent forgery and provide visual privacy.

In our opinion, the most important advantage of using pinto is that this method can be easily added to the current monitoring or IoT systems. it means that stakeholders do not need to change their current operating cameras and they only need to add the software features. This characteristic is important because in most cases, there are many economic and managerial obstacles in the way of hardware updating.

One disadvantage of the proposed method is the post-processing sharing design. Post-processing module can slow down the whole process of storing and sharing the videos. This limitation can be even more annoying in the case of live broadcasting. It seems that Pinto is most suitable for the cases that live monitoring or broadcasting is not required. The other disadvantage of Pinto is that it is limited to MJPEG format. To apply this method for other formats, some details in compression approaches need to be changed. Also, the last limitation that is worth mentioning is that the whole analysis has been done based on a fixed 1280*720 HD frame size. In this situation, more tests are required to demonstrate the applicability of Pinto for other resolutions and consequently different blocking strategies.

REFERENCES

[1] T. Zhang, A. Chowdhery, P. V. Bahl, K. Jamieson, and S. Banerjee, "The design and implementation of a wireless video surveillance system," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, pp. 426–438, ACM, 2015.

[2] H. Yu, J. Lim, K. Kim, and S.-B. Lee, "Pinto: Enabling video privacy for commodity iot cameras," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1089–1101, ACM, 2018.

[3] J. Cho, S. Mirzaei, J. Oberg, and R. Kastner, "Fpga-based face detection system using haar classifiers," in *Proceedings of the ACM/SIGDA international symposium on Field programmable gate arrays*, pp. 103–112, ACM, 2009.

[4] S. V. Dharan, M. Khalil-Hani, and N. Shaikh-Husin, "Hardware acceleration of a face detection system on fpga," in *2015 IEEE Student Conference on Research and Development (SCOReD)*, pp. 283–288, IEEE, 2015.

[5] D. Hefenbrock, J. Oberg, N. T. N. Thanh, R. Kastner, and S. B. Baden, "Accelerating viola-jones face detection to fpga-level using gpus," in *2010 18th IEEE annual international symposium on field-programmable custom computing machines*, pp. 11–18, IEEE, 2010.

[6] J. Matai, A. Irturk, and R. Kastner, "Design and implementation of an fpga-based real-time face recognition system," in *2011 IEEE 19th Annual International Symposium on Field-Programmable Custom Computing Machines*, pp. 97–100, IEEE, 2011.

[7] 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*, pp. 65–89, Springer, 2009.

[8] J. Shu, R. Zheng, and P. Hui, "Cardea: Context-aware visual privacy protection from pervasive cameras," *arXiv preprint arXiv:1610.00889*, 2016.

[9] S. Co, " The Best Indoor/Outdoor Surveillance Cameras of 2018." `/https://www.pcmag.com/article2/0,2817,2475954,00.as`, Accessed 2018. [Online; accessed].

[10] R. B. Wolfgang and E. J. Delp, "Fragile watermarking using the vw2d watermark," in *Security and Watermarking of Multimedia Contents*, vol. 3657, pp. 204–214, International Society for Optics and Photonics, 1999.

[11] N. H. Motlagh, M. Bagaa, and T. Taleb, "Uav-based iot platform: A crowd surveillance use case," *IEEE Communications Magazine*, vol. 55, no. 2, pp. 128–134, 2017.

[12] J. Obermaier and M. Hutle, "Analyzing the security and privacy of cloud-based video surveillance systems," in *Proceedings of the 2nd ACM International Workshop on IoT Privacy, Trust, and Security*, pp. 22–28, ACM, 2016.

[13] A. Costin, J. Zaddach, A. Francillon, and D. Balzarotti, "A large-scale analysis of the security of embedded firmwares," in *23rd {USENIX} Security Symposium ({USENIX} Security 14)*, pp. 95–110, 2014.

[14] A. P. Plageras, K. E. Psannis, Y. Ishibashi, and B.-G. Kim, "Iot-based surveillance system for ubiquitous healthcare," in *IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society*, pp. 6226–6230, IEEE, 2016.