Pinto

Enabling Video Privacy

Information Systems Security
Abdul Mannan 16K-3620
Murtaza Multan 16K-3618

Sharing video evidences benefits the public but threatens **visual privacy**

Security cam

Dashcam

Releasing surveillance videos is strongly discouraged in some countries due to **visual privacy concerns**.

Visual privacy vs. Video forgery

The conventional **post-process blurring** opens the door for **posterior fabrication**.

Visual privacy

Video forgery

Attackers:

Video requester

Video owner

Desired Properties for Video Sharing

1. **Visual privacy**
   - Visual protection depending on situations
   - e.g., type of object to be blurred

2. **Video authenticity**
   - Time of recording
   - Data integrity

3. **Fine video quality**
   - High frame rates
   - Minimal blurring intensity & blurred areas

Existing image-processing techniques?

They produce **poor quality, low-frame-rate videos** due to the limited processing power of commodity cameras.

Desired properties

Method	Frame rate	Video quality	Video authenticity	Visual privacy
Raw recording	24.0 fps	✓	✓	✗
Realtime Blurring	2.3 fps	✗	✓	✓
Fingerprinting	1.1 fps	✗	✓	✓
Watermarking	1.2 fps	✗	✓	✓

Tests on Raspberry Pi 3 (1.2 GHz CPU)

Pinto: Our solution for commodity IoT cameras

Method	Frame rate	Video quality	Video authenticity	Visual privacy
Pinto	24.0 fps	✓	✓	✓

Software-based solution producing privacy-protected, forgery-proof, and high-frame-rate videos using low-end IoT cameras.

Pinto enables:

- Recording a realtime video stream at a fast rate
- Allowing post-processing for privacy protection prior to video sharing
- Keeping their original, realtime signatures valid even after post blurring

Key idea 1:
Deferring the CPU-intensive object detection

Our observation:
Object detection phase takes much more CPU time (x 10^4) than pixelation phase (only 0.05 ms per frame).

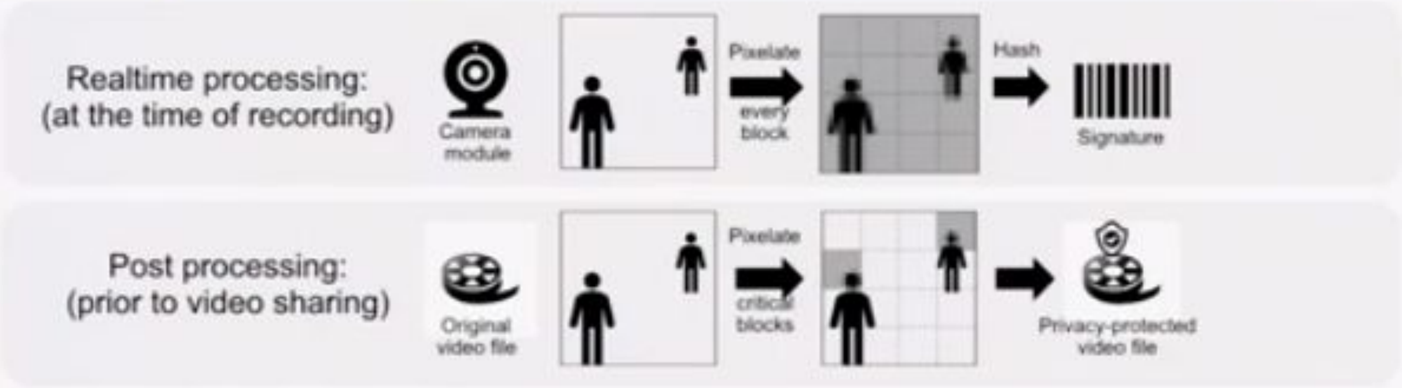
		Processing time taken per frame		
Object blurring	**Frame rate**	**Detect time**	**Pixelate**	**I/O time**
Human face	2.3 fps	431.5 ms	0.05 ms	47.4 ms
Car plate	5.1 fps	146.2 ms	0.05 ms	47.1 ms

Time taken in each step when running realtime blurring on Raspberry Pi

Pinto performs **fast pixelation of entire frames in real time** while deferring the CPU-intensive object detection until necessary for post processing.

Key idea 2: Block-level pixelation

Frame division into subimage blocks allows fine-grained visual privacy on each frame.

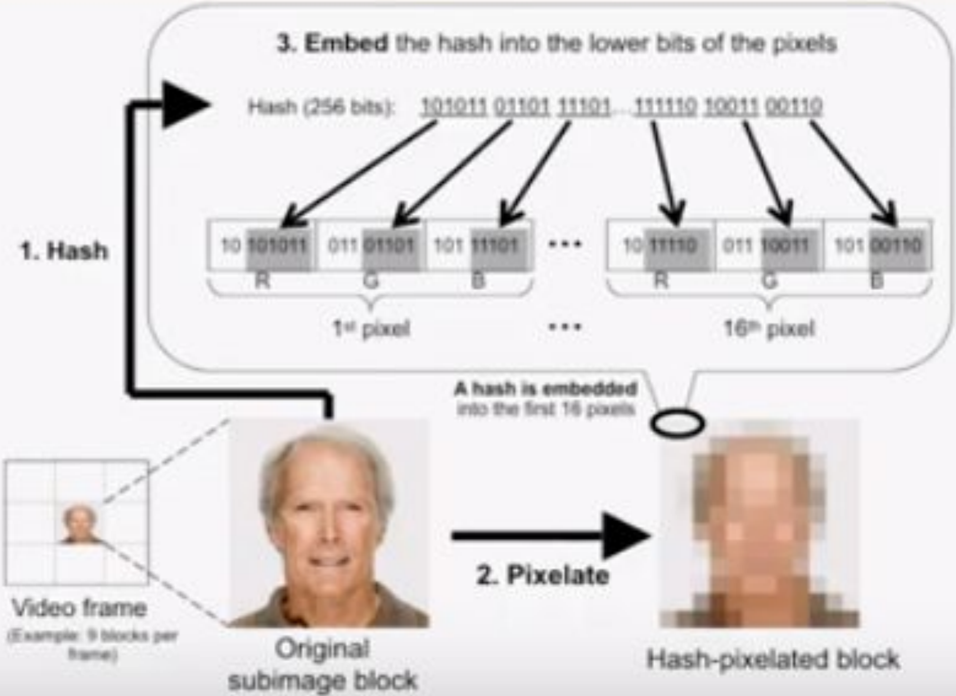


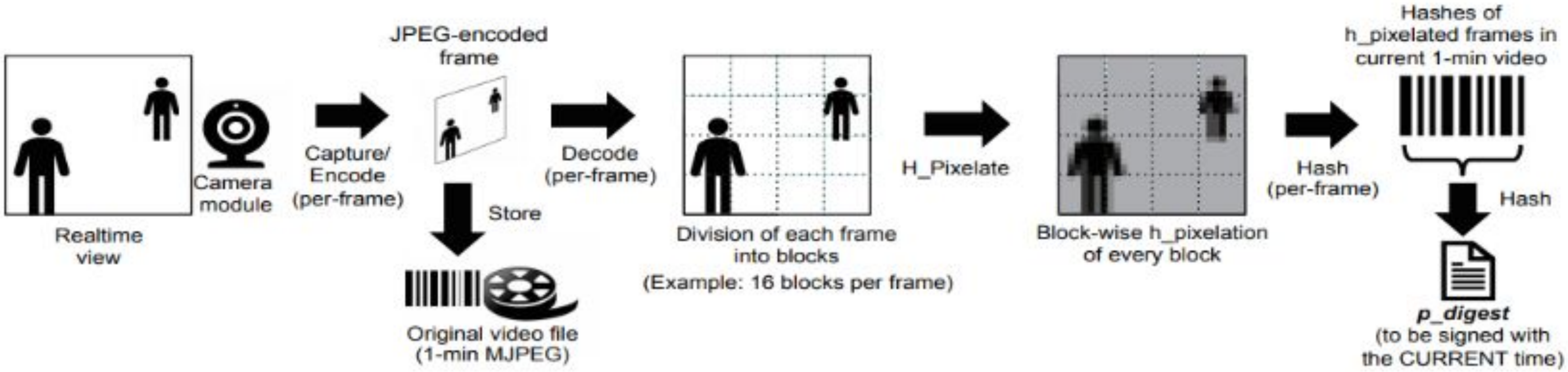
Pinto performs the subimage block-level pixelation on:

- every block (in real time)
- blocks of sensitive objects (post-processing for video sharing)
- blocks of non-sensitive objects (requester-side verification)

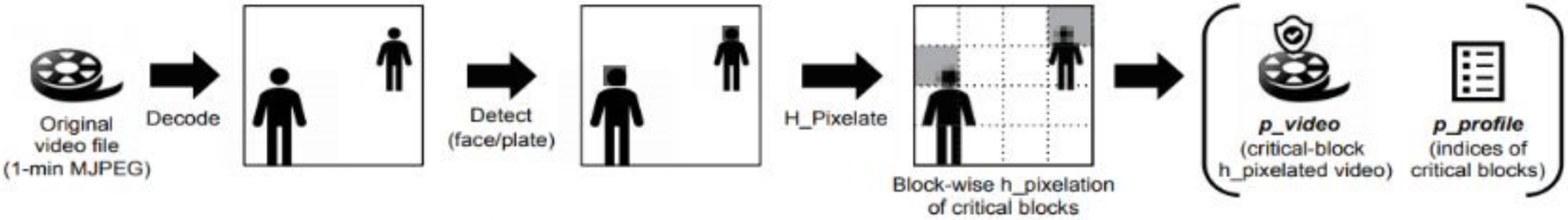
Key idea 3: Forgery-proof pixelation

We devise ***hash-pixelation* (or h_pixelation)** for
both visual privacy protection and forgery prevention.



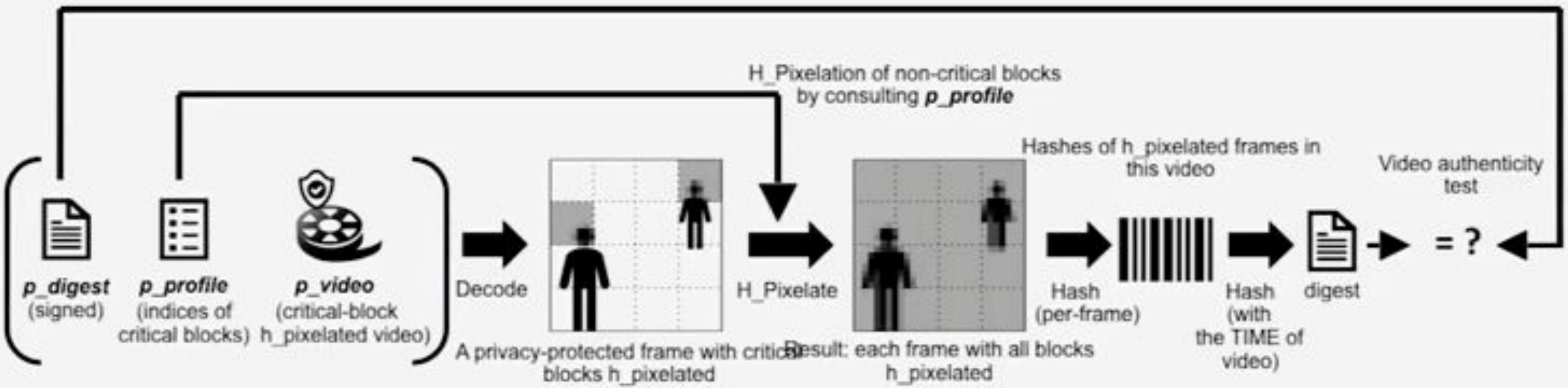


(a) Realtime processing in camera device (at the time of recording).



(b) Post processing (prior to sharing of a video).

Verification of video authenticity (at the requester-side)



Two design factors in Pinto

1. Pixelation intensity (*P-scale*)

- It affects the visual privacy and perceived frame quality.
- **Pixelation intensity should be minimal as possible while de-identifying sensitive objects**.

2. In-frame block count (*B-count*)

- It controls the processing speed and video quality.
- The block-count should be determined for **fine-grained block division while ensuring fast processing speed for producing high-frame-rate videos**.

Implementation and Evaluation

- Implementation of Pinto
  - 1.1K lines of Python and C++ code
  - Using OpenCV and libx264 libraries

- Evaluation checklist:
  1. How much **visual privacy** does Pinto provide?
  2. How well does it **prevent against content forgery**?
  3. How much **video frame rate** does it achieve?
  4. How does in-frame block count affect **human-perceived video quality**?

Sample Output

Acknowledgement

Yu, H., Lim, J., Kim, K., & Lee, S. B.

Reference:

Yu, H., Lim, J., Kim, K., & Lee, S. B. (2018, October). 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.

Thank You

Questions?