Getting the Face Behind the Squares: Reconstructing Pixelized Video Streams

Ludovico Cavedon Luca Foschini Giovanni Vigna
University of California, Santa Barbara
{cavedon,foschini,vigna}@cs.ucsb.edu

Abstract

Pixelization is a technique to make parts of an image impossible to discern by the human eye by artificially decreasing the image resolution. Pixelization, as other forms of image censorship, is effective at hiding parts of an image that might be offensive to the viewer. However, pixelization is also often used also to achieve anonymity, for example to make the features of a person's face unrecognizable or the defining characteristics of cars and building unidentifiable. This use of pixelization is somewhat effective in the case of still images, even though it is open to dictionary attacks. However, when used in videos, pixelization might be vulnerable to full reconstruction attacks.

In this paper, we describe an attack against the anonymization of videos through pixelization. We develop an approach that, given a pixelized video, reconstructs the image being pixelized so that the human eye can clearly identify the object being protected. We implemented our approach and tested it against both artificial and real-world videos. The results of our experiments show that, in many cases, video pixelization does not provide sufficient guarantees of anonymity.

1 Introduction

Hiding or blurring an image, or parts of it, is performed to achieve different goals. For example, blurring is often used to prevent graphic, suggestive, or offensive content from being clearly identifiable by the viewer.

The most obvious way to remove content is to mask the area to be protected using a solid-colored shape (e.g., a black square). However, other ways of hiding or blurring an image that maintain the general appearance of the image (e.g., in terms of color distribution) might be preferred, as they produce a more "natural" result. For example, blurring with a Gaussian filter, and tessellation scrambling can be used for this purpose. However, if too much information is retained through the blurring process, it could be possible to invert the transformation and obtain the original image. For example, three years ago, a pedophile posted on the web a picture of himself where his face was scrambled with a whirl blur filter to be unrecognizable: nevertheless, the police was able to apply the inverse filter and reconstruct the image of the man's face (see Figure 1).

Pixelization is a widely-used technique that decreases the resolution of an area of an image to make the details of an image impossible to discern [4].[1] Unfortunately, pixelization is vulnerable to several reconstruction attacks and should only be used to hide the immediate appearance of offensive graphic content.

If pixelization is used to achieve privacy or anonymization of a still image, the transformation could be vulnerable to dictionary attacks [15]. Even worse, if pixelization is used to anonymize a video, under certain conditions it is possible to completely reverse the anonymization process and obtain the original, unpixelized image.

This is a known problem of pixelization [5]. However, to the best of our knowledge, there has never been a formal description of this attack, and, in addition, there is no tool that is available to analyze pixelized videos.

In this paper we present an approach to the reconstruction of an image from pixelized video. Our approach uses a number of novel techniques to optimize the recovery process. We have developed a prototype tool that implements our approach and we used it on several examples of pixelized videos. The results show that in many cases it is possible to completely reconstruct the graphic information that was protected by using pixelization.

For sake of simplicity, we limited our experiment to

[1] Note that the terms "pixelized" and "pixelated" are different. A pixelized image is an image that has undergone the pixelization process. A pixelated image is an image whose resolution allows one to identify the single pixels of the image and in general gives the impression of poor quality.

Figure 1: The reconstructed picture of a suspected pedophile (left) and the "anonymized" version that was posted on the Internet (`http://www.msnbc.msn.com/id/21190969/`; published on Oct 8, 2007; retrieved on Nov 18, 2009).

gray-scale videos. Working with color videos does not pose significant additional challenges, and the same technique we applied to our gray-scale video can be applied separately to each channel (red, green, and blue components) of a color video.

In summary, these are our contributions:

- We formalized the problem of reconstructing an image from a video that has been anonymized using pixelization. To the best of our knowledge, this is the first formalization of the problem in this context.
- We developed an approach for image reconstruction that uses several novel techniques to optimize the recovery of the graphic content that has been pixelized.
- We implemented our approach in a tool, and we tested its performance on a number of videos, showing that indeed content can be recovered in many cases.

The rest of this paper is structured as follows. In Section 2 we describe the de-identification approach that is associated with pixelization. In Section 3, we describe our approach to the reconstruction of the original image from a sequence of pixelized images. Then, in Section 4, we describe our tool and the experiments that we ran on both artificial and real-world video sequences. We highlight the limitations of our approach and potential areas for improvement in Section 5, while, in Section 6, we analyze related work. Finally, in Section 7 we briefly conclude.

2 Image De-identification via Pixelization

Pixelization [4] is a technique employed to hide details from an area of an image, by replacing the area with a very low resolution version of it. An appealing feature of this technique is that it preserves the color distribution of the scrambled area, and, therefore, it gives the viewer a milder distortion of the original image with respect to other techniques.

In pixelization, the area of the image to be anonymized is divided in a grid of squares of equal size,[2] and the color of each square is replaced with the mean of the color of the pixels underlying that square.

The results of pixelization can be seen, for example, in Figure 4(b), where the face of William Mark Felt, Sr., the man known as "Deep Throat" in the Watergate scandal, is scrambled to make him unrecognizable.

However, when this method is applied to video sequences depicting the same subject, it is highly probable that the pixelization squares change position with respect to the underlying image, therefore averaging different pixels at different times. While a human observer is not able to exploit this fact, it is not difficult to process the video frames to obtain a partial reconstruction of the original image. This process is call *image recovery*.

3 Image Recovery

For each frame, the pixelization filter is a linear operation, which maps multiple square regions of an image into their respective means.

Let x be a vector of size $C = n \cdot m$ representing the $m \times n$ image that we want to reconstruct. We introduce the pixelization matrix P, defined as the vertical concatenation of two matrices P_1 and P_2. P_1 has size $(R = (f \cdot px \cdot py), C)$, while P_2 has size (k, C), where:

- f is the number of frames available;
- px and py are the number of horizontal and vertical pixelization squares; and
- k is the number of pixels that are covered by pixelization only in a subset of the frames, whose value[3] can thus be retrieved exactly.

The total size of P is then $(R + k, C)$. As shown in Figure 3, each row of the first R rows of P represents the pixelization operation applied to *one* square of one frame. Therefore, supposing that the squares are of size

[2] Actually, it is not necessary for the squares to be all of equal size or to be squares at all. However, this is the most common way to perform pixelization. Hereinafter, without loss of generality, we will assume that the area to be pixelized is divided into equal-sized squares.

[3] In this paper, by pixel *value* we mean the 8-bit gray-scale value.

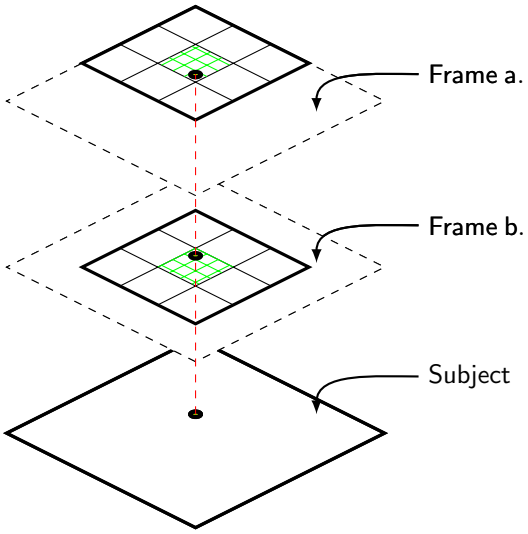


Figure 2: The black dot in the subject image is captured by two frames a and b. In each frame, the dot is captured by the same square (thin grid), but it is averaged with different portions of the original image, because of the relative offset in the pixelized area (thick grid) in a and b.

$$\begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1C} \\ \vdots & \vdots & \ddots & \vdots \\ p_{R1} & p_{R2} & \cdots & p_{RC} \\ p_{(R+1)1} & p_{(R+1)2} & \cdots & p_{(R+1)C} \\ \vdots & \vdots & \ddots & \vdots \\ p_{(R+k)1} & p_{(R+k)2} & \cdots & p_{(R+k)C} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_C \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ b_{R+k} \end{bmatrix}$$

$$P = P_1 | P_2 : (R+k, C) \qquad x \qquad b$$

Figure 3: The reconstruction procedure is cast as a linear system. The pixelization matrix P is the vertical concatenation of P_1, in which different rows account for different squares in different frames and P_2, which accounts for the pixels that were seen out of the pixelized grid at least once. x is the subject image to be reconstructed (expressed as a vector). b is the vector of known terms, coming either from the gray-level intensity of a square, or from the actual value recorded for pixels seen outside the pixelized grid at least once.

$g \times g$, every such row will have g^2 elements set to $1/g^2$ and the rest of them set to 0. Each row r, in fact, when multiplied by the image x, produces a single value given by the average of the pixel in x underlying the square represented by r. The vector b in Figure 3 stores such known terms, one for each square and each frame, $R = f \cdot px \cdot py$ in total.

The last k rows of the matrix will have just one element set to 1.

The vector b needs further explanation. Each element in b comes either from the average of a pixelization square in a frame, or from the actual value of a single pixel in the original image, which at some point in time have slid out the pixelized area and could be retrieved fully. We define the set of pixels that have emerged from the pixelized area at least once during the video the "uncovered pixels". For real video sequences, the color of the uncovered pixels will vary from frame to frame (hopefully of a small quantity), so the value stored in b will be the median of all the collected values from the video sequence. Similarly, if the video is compressed with a lossy algorithm, all the values of a pixelization square may not be the same; in this case, as well, we take the median value.

Differently from the majority of super-resolution approaches (e.g., [7] and [9]), we do not need to consider the point spread function (PSF) of the video camera, i.e., we are not interested in modeling how a real image is distorted though the lenses and the charge-coupled device (CCD). In fact, we are not trying to obtain an image of resolution higher than the video frames, but, instead, we are aiming to recover an image of the same quality.

The linear system $Px = b$ is very unlikely to have an exact solution; the system could be under-determined because the video frames do not cover all the possible offsets of the pixelization squares (which translates into not having enough rows in P linearly independent with each other). More precisely, if we have $g \times g$ pixelization squares, in order to have a full-rank P matrix we need the pixelized area during the video to cover g^2 adjacent offsets. Intuitively, this implies that each pixel of the original image appears in enough equations (rows of P), in each of them averaged with other different pixels, conveying enough information for the full reconstruction of the original pixel.

Moreover, this system is likely to have no solution, mainly for two reasons. The first one is the quantization approximation, that is, the color of the pixelization squares is not the exact mean of the original pixels, but, instead, it is rounded to the nearest integer when stored as a new frame; the second one is the fact that the images in the different frames of a real-world video are subject to change because of sensor noise, non discrete panning of the camera, change in light, atmospheric aberration, etc.

The naïve way of trying to get the reconstructed image $\hat{x}$ would be trying to compute $\hat{x} = P^+ b$, where P^+ is the pseudo-inverse matrix of P. In this way, we would try to find an $\hat{x}$ that minimizes the mean squared error of the

system. This solution, however, will not give satisfying results (see Figure 4(c)) even for the ideal case in which the P matrix has full rank and all the pixelized frames have been generate from identical shifted images. This is due to the sensitivity to the quantization of the value of the pixelization squares. The pseudo-inverse method, in fact, does not take into account that the values of the pixels are bounded and generates a mostly out-of-range solution featuring a chessboard like pattern.

3.1 Maximum A Posteriori (MAP) approach

Based on the experience provided by previous literature and our experiments, we propose the following method to obtain a good-quality reconstructed image in a reasonable time frame. The goal of our approach is to find an image $\hat{x}$ such that, if shifted and pixelized in multiple frames, will give an output sequence that is as close as possible to our input video.

We therefore want to minimize the error with respect to the $P\hat{x} = b + \varepsilon_q$ system, where ε_q is a vector of the same size of b and whose elements can take values in the range $[-0.5, 0.5]$. The role of ε_q is to take into account the quantization effect.

We also need a regularization function to be applied to the image in order to correct for the noise introduced by the quantization, the CCD sensor, the image registration error, etc. Such regularization function must smooth image artifacts, but, at the same time, preserve edges. Minimization of *total variation* (TV, or L_1 norm of image derivatives) [13] has proven to be superior in image de-noising when compared to L_2 norm minimization, because of its less strong penalization of image edges.

Our function to minimize, therefore, is:

$$f(x) = ||Px - b - \varepsilon||_2 + \alpha \mathrm{TV}(x) \qquad (1)$$

where $\mathrm{TV}(x) = ||\nabla x||_1$ is the Total Variation of image x and α is how much TV is weighted with respect to the system's mean squared error (MSE).

This approach is often called *Maximum a Posteriori* (MAP) method. This class of methods tries to find the original image x which, if pixelized, is most likely to have generated the frames in the video, given a probability distribution for x. In our case, the regularization function (i.e., the TV function) can be interpreted as the prior probability for x, favoring piecewise smooth images.

In order to solve this problem, we followed the efficient iterative approach proposed in [7], i.e., we adopted the steepest descent method:

$$\begin{aligned}
\hat{x}_{k+1} &= \hat{x}_k - \beta \nabla_f(\hat{x}_k) \\
\nabla_f(x) &= P'(Px - b) + \\
&+ \alpha \sum_{\substack{(l,m)\in \\ \{(0,1),(1,0)\}}} (S_{l,m}(x) - \mathrm{shift}_{-l,-m}(S_{l,m}(x))) \\
S_{l,m}(x) &= \mathrm{sign}(x - \mathrm{shift}_{0fill,l,m}(x))
\end{aligned}$$

where β is the descent speed. The $\mathrm{shift}_{0fill,l,m}(x)$ function shifts the image x of l rows and m columns, while filling the new border elements with zeros; the $\mathrm{shift}_{l,m}(x)$ function, is the same shifting operation, but the new elements will be the same as the previous ones in that position. These two distinct operations are needed to ensure proper handling of border pixels.

Although Farsiu et al. found that *Bilinear Total Variation* [14] regularization gives better results [7], in our case it is not beneficial because of the chess-board pattern that the minimum MSE solution tends to generate. Therefore, in contrast with Fariusa et al. [14], we limit the radius of the TV regularization operator for each pixel to 1. This means that for each pixel the TV is computed taking into account only its axis aligned adjacent pixels (North, South, East, West).

3.2 Image Registration

In pixelized video recovery, we assume that the same image has been recorded in different frames at different positions. One of the most important tasks is therefore *image registration*, i.e., the tracking of how the image moves across frames. Wrong or inaccurate image registration would lead to reconstruction with excessive blurring or artifacts (depending on the chosen weight for TV regularization).

Our reconstruction technique is extremely sensitive to image registration. At each frame, we require that the pixelized image is accurately tracked, which means to correctly recover the offset with respect its position in the previous frame.

To improve the accuracy of this step, we developed a *sub-pixel* image registration technique. This technique allows us to deal with, and recover correctly, fractional (sub-pixel) image offsets. A detailed description of how the sub-pixel registration is performed is out of the scope of this work.

3.2.1 Image registration with OpenCV

In order to perform image registration we used the *Intel OpenCV library* [1], which contains a collection of computer vision functions. In particular, the optical flow

tracking API allowed us to automatically identify interesting features on the input video and track their movement across multiple frames.

We used the

`cvCalcOpticalFlowPyrLK()`

method, implementing a pyramidal version of the Lucas-Kanade feature tracking algorithm [3]. Such algorithm proved to be faster that the other optical flow tracking functions implemented in OpenCV, yet very precise (providing fractional offsets) and convenient for our needs; in fact it allows us to provide an image mask that we used to exclude the pixelized area from being analyzed for movement tracking. The image mask also allows us to instruct the feature tracking algorithm to work on a restricted part of the image, containing feature moving together with the pixelized area of the image (e.g., a moving car or person).

3.3 Outliers removal

When dealing with real videos recoded by a camcorder, a longer video sequence does not necessarily mean improved reconstruction quality. In fact some variables of the video are likely to change over time (e.g., light source intensity and position, automatic white balance of the camcorder, position of the subject or camera).

Moreover, even for short video sequence, the image registration algorithm may give inaccurate results for some frames.

In both cases the effectiveness of our approach is negatively affected. We therefore introduced a method for recursively filtering out the outliers, i.e., the frames that mostly contribute to the error of the recovered solution in our linear system.

This can be achieved by observing the distribution of the squared errors $(b-P\hat{x})^2$ summed over each frame. In our experiments, as shown in Section 4.3, we observed that a small set of frames was giving a much higher contribution to the error than the other frames. We therefore removed them from the video and reprocessed it to recover the pixelized image again.

4 Experiments

In order to show the effectiveness of our approach we provide three examples, one from an artificial video sequence, and two from real-world videos. All the experiments have been run with $\beta = 0.3$ and $\alpha = 0.01$ for the steepest descent algorithm.

4.1 Ideal case

The artificial video has been created by using one single image (Figure 4(a)). Such image has been shifted into 10 different positions both horizontally and vertically, so as to generate 100 different frames, which have been combined sequentially into a video. Then, the video has been pixelized into 100 squares of size 10×10 (Figure 4(b)). After that, the pixelized video has been process with the algorithm described in Section 3 (Figure 4(d)).

Even though this case represents an optimal, unrealistic example, it represents an interesting baseline for our approach. In particular, it shows that even after a 10×10 pixelization it is possible to recover an impressive amount of image detail. It also allows a comparison between the extremely noisy solution provided by the pseudo-inverse method and the solution provided by the MAP approach with total variation prior.

We then tried to vary the size of the pixelization squares, while keeping fixed the size of the pixelized area. More specifically, we used the same image of the previous example and pixelized with pixelization squares of size 2, 3, 4, 5, 6, 10, 12, and 15. For every experiment, we wanted a 60×60 pixelized area, with no pixel getting uncovered in any frame. The mean square error per pixel of the reconstructed image proved to increase linearly with the size of the pixelization square. As such number increases, also the number of frames in the video sequence increases, and therefore one may expect that this additional information would compensate for the reduced number of equations per frame (as the number of pixelization squares decreases): in fact, the number of equations in the P_1 matrix remains constant. However, bigger pixelization squares imply that more information is lost during the quantization of the value of each square; in other words, every equation of P_1 has more degrees of freedom (pixels) in re-distributing the quantization error.

In the previous example, we generated a number of frames having different offsets in order to have a full-rank P matrix. For example, if the pixelization squares size was 10×10, we shifted the original image from 1 to 10 pixels horizontally and from 1 to 10 pixels vertically, therefore generating 100 different image offsets. However, we wanted to see how a reduced number of available frame offsets would impact the quality of the reconstructed image. For this purpose, we randomly selected a fraction of all the frames of the previous experiment and ran the same reconstruction algorithm on that. We repeated the experiment for 10 different percentages, from 10% to 90%. In Figure 5, we show the recovered images, while in Figure 6 we plotted the value of the mean squared error per pixel of the pixelized area. This shows that with our reconstruction method we do not actually need a full-rank P matrix (as it usually happens for a real video); the algorithm is therefore robust even if 50% of the frames are missing.

(a) $p = 10\%$ (b) $p = 30\%$ (c) $p = 50\%$ (d) $p = 70\%$ (e) $p = 90\%$

Figure 5: Recovered images according to different values of p, where p is the percentage of distinct frames available for image reconstruction. Such percentage is relative to the minimum number F of frame at different offsets which would give a full-rank P matrix. In this case, the same video as in Figure 4 was used: 10×10 pixelization squares have been used, so F is 100. Therefore $p = 30\%$ means that 30 distinct frames were available for image reconstruction. The recovery of the original image was performed 5 different times, each time using a given fraction of the total frames. The actual frames used have been selected randomly.

4.2 License plate recovery

The second example is a video recorded with a common consumer 460K pixel CCD Mini-DV camcorder. A 120-frame (4-second) sequence has been recorded by a person standing still and holding the video camera in her/his hands, while pointing it towards the back of a car. The part of the frames depicting the license plate of the car have then been pixelized with 4×4 pixel squares, according to the algorithm described in Section 2 (Figure 7(a)). Even though the person shooting the video was asked not to move, small oscillations are typical when the video recording device is held without a tripod. The pixelization of the sequence is done by a video processing filter, which is creating the pixelization squares in a fixed position with respect to the video frame borders rather than the subject being recorded.

We then exploited these continuously changing differences in position between the license place and the pixelization grid in order to recover the original license plate number. Image registration has been performed on the rest of the car image as described in the previous section.

The resulting image (Figure 7(b)) shows how the letters and numbers are clearly readable, even if for a human observer the original video sequence would have given no clue. This example proves how pixelization has failed its intent to preserve privacy. Even though from the video itself the details of the license plate are not revealed, someone interested in recovering the sensitive information can do so with an inexpensive video processing procedure.

4.3 Face recovery

The third example (Figure 8) is still real video, recorded in the same way as the previous example. This video is a recording of a face for 10 seconds (300 frames). The size of the pixelization squares is, in this case 8×8. In this longer video, there is a relevant number of frames that give an excessive contribution to the error of the reconstruction model. This can be seen by the plot of the squared error contribution $(b - A\hat{x})^2$ summed over each frame (Figure 9). Note that in the graph, the frames are not in chronological order, but have been ordered according to increasing squared error contribution. In fact, the contribution for the last 50 frames grows much faster than the rest of the frames. We therefore applied the technique described in Section 3.3, removing them from the video and recomputing the solution of the reconstruction problem again. It is interesting to note that, in this case, the outliers removal process helped in discarding the frames when the subject was blinking, therefore allowing us to retain good detail also in the area around the eyes.

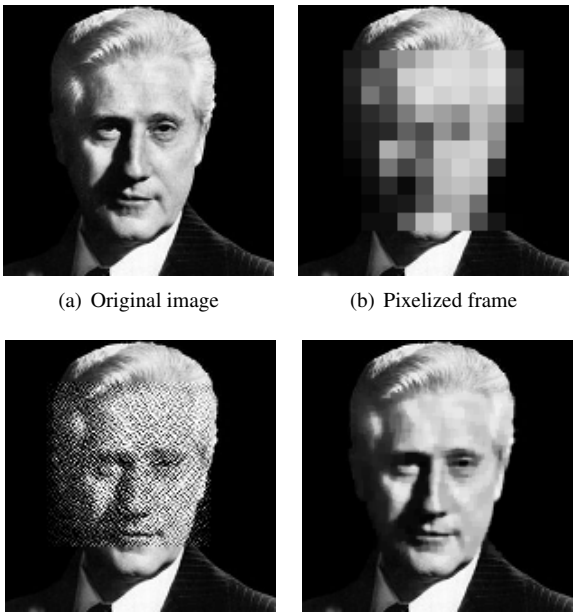
Again, the pixelization processing failed to hide the identity of the subject. The reconstructed face is clearly identifiable.

5 Potential Techniques for Improvement

The environment of the input video for which our approach would successfully work out-of-the-box is very limited. For example, we assume that the subject is still with respect to the background and the only allowed movements for the camera are horizontal/vertical translation or slight rotation (panning/tilting).

No zooming or horizon rotation are considered, although they might be typical for a hand held camcorder. Such camera movement can be taken into account by further developing the model in order to be able to describe all rigid movements of the image.

If the subject is not still with respect to the background, two different approaches may be followed. If the subject is not completely covered by the pixelized image, we can perform image registration on the uncovered parts



(a) Original image (b) Pixelized frame

(c) Pseudo-inverse reconstruction (d) MAP recovered image

Figure 4: Example of reconstructed images from an artificial video. A single image (a) has been shifted into 64 different positions and than pixelized (b). The original image has then been recovered with the pseudo-inverse method (c) or with the proposed algorithm (d) from the 64-frame video.

(e.g. the top of the head and the shoulders on the example of Figure 8, or the body of a running car). Alternatively, if the subject is completely covered by the pixelization, tracking of its movement can be attempted computing its center of mass, although the estimation would be clearly much less accurate.

Further developing this method of recovery of pixelized images towards allowing more degrees of freedom on the input video, 3D camera movement tracking can be considered, using software like *Vicon Boujou* [2] and augment the P matrix to take into account affine deformation of the subject. For example, this improvement would allow targeting the reconstruction of a license place of a passing car observed by a panning camera.

An additional important direction of improvement is taking into account the artifacts introduced by MPEG compression, which may badly interfere with the pixelization squares. As of the current implementation, we are addressing this problem by taking the median of the values of the pixelization squares. However, considering how the compression process distorts the image would help detecting better values of the pixelized image.

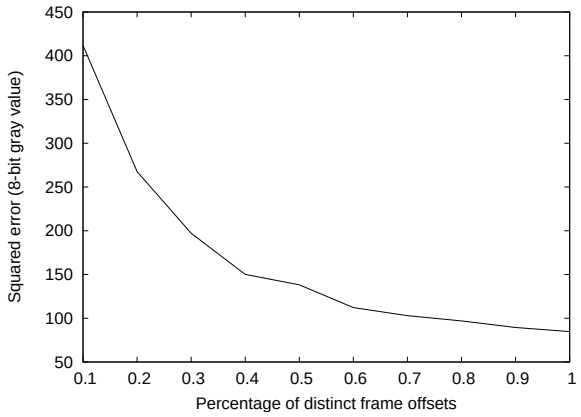
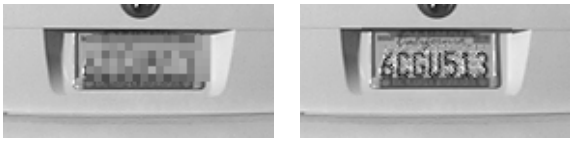


Figure 6: Mean squared error per pixel of the pixelized area vs. the percentage of distinct image position offset between frames available for reconstruction, with reference to the example in Figure 5.



(a) Pixelized frame (b) MAP Recovered image

Figure 7: Pixelized frame (a) and recovered image (b) from a 120-frame real video sequence of a license plate. Pixelization squares are 4×4 pixel big.

6 Related Work

The fact that pixelization may fail to preserve privacy is not new. Neustaedter et al. raise some concerns when blurring and pixelization are used for video-conferencing in home settings [10]. The authors reached the conclusion that it is not possible to provide a good idea of the the scene depicted in the video and at the same time preserve the privacy of the subject being recorded by using blurring/pixelization.

Also the effectiveness of pixelization applied to faces has already been questioned. Gross et al. show that pixelization has very low effectiveness in preserving privacy when pixelized video sequence is fed to a face recognition software matching the face against a face database [8]; quite unexpectedly, they also find that attempts to reconstruct the original image usually worsen the effectiveness of the recognition algorithm. Our concern is different: we do not want to avoid a face to be matchable against a database, but we want to recover the original image (e.g., a face or a license plate number) in a way that it is recognizable for the human eye.

More recently, Dufaux [5] analyzed the effectiveness

(a) Original image (b) Pixelized frame

(c) Average of pixelized frames (d) MAP Recovered image

Figure 8: Frame from original video (a) and pixelized frame (b). Image (c) has been obtained by averaging all the pixelized frames after image registration. Image (d) is the output of our recovery process. The pixelized frames are taken from a 300-frame real video sequence of a face. Pixelization squares are 8×8 pixel big.

of various Privacy Enabling Technologies applied to human faces in a video sequence. They compared different anonymization methods by means of similarity measures and effectiveness of application of face recognition techniques to the anonymized video. Again, pixelization was shown to be weaker than other methods.

The process of recovering an image after it has gone through a process of pixelization is a particular case of image super-resolution, i.e., the extraction of a high-resolution image from a sequence of low resolution ones. Super-resolution has been an active field of research in the last twenty years; a broad and detailed analysis of such work is presented in [12]. Our approach is similar to the work of Farsiu et al. [7] in the method used to find the recovered images: like them, we are using the steepest descend method to minimize the error of our linear model augmented with a total variation-based regularization of the image.

However, our model is augmented in order to take into account more information about our input (i.e., the fact that the pixelization is created by an averaging video filter and that there are uncovered pixels in some frames

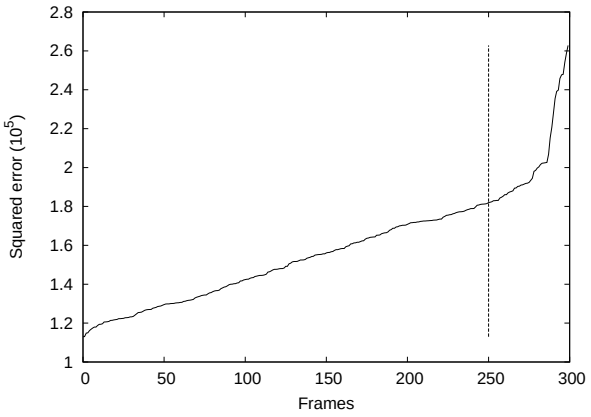


Figure 9: Sum of squared error contributions $(b_i - A_i\hat{x})^2$ per frame on the example of Figure 8. Frames have been ordered according to increasing squared error contribution.

which are helping us to recover the original image).

Additional research in the super-resolution field has targeted color images [6], which pose additional issues when the video sequence is coming without rescaling from a CCD sensor. Moreover, more sophisticate algorithms looking for a *maximum a posteriori* solution, and, at the same time, computing optimal image registration has been explored by Pickup [11]. Another interesting approach comes from Gunturk et al., whose model takes into account errors introduced by video compression processes [9]. These techniques have not been applied to our proof-of-concept experiments, but are however interesting paths to explore for future research.

7 Conclusions

Pixelization is a technique to blur an image (or the frame of a video) to either protect the viewer from offensive content or prevent the viewer from recognizing the characterizing features of an object (such as a face, a car's plate number, etc.)

We have developed a new approach and developed a tool that performs an attack against the privacy of pixelized videos. The results of our experiments show that, in many cases, images can be recovered from pixelized videos, and, therefore, this technique should not be used to achieve privacy.

Future work will focus on how additional information about the geometry and position of the objects to be recovered can be used to improve the reconstruction process, especially in cases where the object being blurred roto-translates with respect to the camera.

References

[1] Open Computer Vision Library, 2009. `http://www.opencv.org`.

[2] Vicon Boujou, 2009. `http://www.vicon.com/boujou/`.

[3] BOUGUET, J. Pyramidal implementation of the lucas kanade feature tracker description of the algorithm. *Intel Corporation, Microprocessor Research Labs, OpenCV Documents* (1999).

[4] BOYLE, M., EDWARDS, C., AND GREENBERG, S. The effects of filtered video on awareness and privacy. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work* (2000), ACM New York, NY, USA, pp. 1–10.

[5] DUFAUX, F. Video scrambling for privacy protection in video surveillance: recent results and validation framework. vol. 8063, SPIE, p. 806302.

[6] FARSIU, S., ELAD, M., AND MILANFAR, P. A practical approach to superresolution. In *Proceedings of SPIE* (2006), vol. 6077, pp. 24–38.

[7] FARSIU, S., ROBINSON, M., ELAD, M., AND MILANFAR, P. Fast and robust multiframe super resolution. *IEEE Transactions on Image Processing 13*, 10 (2004), 1327–1344.

[8] GROSS, R., SWEENEY, L., DE LA TORRE, F., AND BAKER, S. Model-Based Face De-Identification. In *2006 Conference on Computer Vision and Pattern Recognition Workshop* (2006), pp. 161–168.

[9] GUNTURK, B., ALTUNBASAK, Y., AND MERSEREAU, R. Super-resolution reconstruction of compressed video using transform-domain statistics. *Image Processing, IEEE Transactions on 13*, 1 (Jan. 2004), 33–43.

[10] NEUSTAEDTER, C., GREENBERG, S., AND BOYLE, M. Blur filtration fails to preserve privacy for home-based video conferencing. *ACM Transactions on Computer-Human Interaction 13*, 1 (2006), 1–36.

[11] PICKUP, L. Overcoming registration uncertainty in image super-resolution: maximize or marginalize? *EURASIP Journal on Advances in Signal Processing 2007* (2007), 1–14.

[12] PICKUP, L. C. *Machine Learning in Multi-frame Image Super-resolution*. Dissertation, University of Oxford, 2007.

[13] RUDIN, L., AND OSHER, S. Total variation based image restoration with free local constraints. In *IEEE International Conference Image Processing, 1994. Proceedings. ICIP-94.* (1994), vol. 1.

[14] TOMASI, C., AND MANDUCHI, R. Bilateral filtering for gray and color images. In *Computer Vision, 1998. Sixth International Conference on* (1998), pp. 839–846.

[15] VENKATRAMAN, D. Why blurring sensitive information is a bad idea. `http://dheera.net/projects/blur.php`, January 2007.