Short Papers

A Coprime Blur Scheme for Data Security in Video Surveillance

Christopher Thorpe, Feng Li, *Member*, *IEEE*, Zijia Li, Zhan Yu, *Student Member*, *IEEE*, David Saunders, and Jingyi Yu, *Member*, *IEEE*

Abstract—This paper presents a novel coprime blurred pair (CBP) model to improve data security in camera surveillance. While most previous approaches have focused on completely encrypting the video stream, we introduce a spatial encryption scheme by strategically blurring the image/video contents. Specifically, we form a public stream and a private stream by blurring the original video data using two different kernels. Each blurred stream will provide the user who has lower clearance less access to personally identifiable details while still allowing behavior to be monitored. If the behavior is recognized as suspicious, a supervisor can use both streams to deblur the contents. Our approach is based on a new CBP theory where the two kernels are coprime when mapped to bivariate polynomials in the z domain. We show that coprimality can be derived in terms of the rank of Bézout matrix [3] formed by sampled polynomials, and we present an efficient algorithm to factor the Bézout matrix for recovering the latent image. To make our solution practical, we implement our decryption scheme on a graphics processing unit (GPU) to achieve real-time performance. Extensive experiments demonstrate that our new scheme can effectively protect sensitive identity information in surveillance videos and faithfully reconstruct the unblurred video stream when both CBP sequences are available.

Index Terms—Video surveillance, greatest common divisor, image deblurring, visual cryptography, CUDA

1 Introduction

VIDEO surveillance in public spaces has increased dramatically in recent history as a means to deter both terrorism and crime in urban environments. However, concern about the potential for abuse and the general loss of privacy has also grown along with the number of surveillance cameras. In recent years the problem of providing visual data in sensitive environments without impinging on the privacy of the public has been a major research topic in machine vision society. The challenge in protecting surveillance data lies in how to reduce both the number of personally identifiable features and the people with access to these features [8], [7], [12].

State-of-the-art information hiding techniques either rely on cryptography (codes and ciphers) to change the structure of data or use steganography to render the message invisible. For visual data, Naor and Shamir [19] first introduced the notion of visual cryptography. They also developed a visual secret sharing (VSS) scheme by dividing the secret image among n participants, where the image can only be recovered when a sufficient number of participants stack their respective pieces together. VSS can effectively protect the visual data. However, since the data received by each individual participant has been fully encrypted, it provides very little usable information. In the case of video surveillance, this indicates that the data received by low-clearance users would be completely useless.

In this paper, we present a novel visual data-hiding technology for data security in video surveillance. Different from previous "complete" encryption schemes, we introduce a "partial" data encryption scheme by blurring the image/video contents. Our goals are two-fold, to provide anonymity to the public and to limit user access to image streams with less degrees of blurring or no blurring at all. Our approach makes use of two blurred image streams, each of which suppresses the appropriate private attributes while retaining the ability to reconstruct an unblurred image stream if one has access to both image streams simultaneously.

Our solution builds upon the recent coprime blurred pair (CBP) model [17] in which the two kernels used for blurring the image are coprime when mapped to bivariate polynomials under the z-transform. The blurred contents in a single stream are difficult to restore using conventional blind deconvolution methods. However, when both streams are available, we can derive their coprimality in terms of the rank of Bézout matrix [3] formed by sampled polynomials. We further present an efficient algorithm to factor the Bézout matrix for recovering the latent image. To make our solution practical, we implement our decryption scheme on the graphics processing unit (GPU) to achieve real-time performance. Experiments show that our new scheme can effectively protect sensitive identity information in surveillance videos and reconstruct the original unblurred video stream for people with high-security clearance.

2 Related Work

Most existing visual data protection schemes fall into the framework of visual cryptography (VC) first proposed by Naor and Shamir [19]. The main idea is to partition the data into n pieces called the shares. Only when a sufficient number of shares are stacked together will human eyes recognize the image content. To achieve this goal, the authors superimpose random patterns of dots onto each share to produce noisy-looking images. However, the appearance of these noisy images in and of themselves might lead one to suspect data encryption. To alleviate this problem, newer schemes such as halftoning [33], [18], [29] attempt to separately deal with grayscale and color channels to minimize the loss in contrast. They also produce visually pleasing results for the shares. For example, the work by Zhou et al. [33] observes that the grayscale component carries the most significant visual cues and they chose to generate halftone shares on a second image.

The existing VC solutions, however, are not directly applicable to privacy protection in visual surveillance. Previous image obscuring methods completely hide the image contents, making the data completely useless to low-clearance users. For example, Upmanyu et al. [27] generate a set of random images for each video frame, each by itself conveying no meaningful information about the original frame. Ideally, low-clearance users should still be able to access some visual information (e.g., the behavior of the target) although their ability for viewing privacy-sensitive details such as facial features would be constrained. These observations have led to the notion of "blind vision" for addressing such privacy issues. Recent efforts include privacy-protected face detection [2], face

- *C. Thorpe, Z. Yu, D. Saunders, and J. Yu are with the Department of Computer and Information Sciences, University of Delaware, 101E Smith Hall, 18 Amstel Ave, Newark, DE 19716.*
- *F. Li is with Qualcomm Technologies, Inc., AV-310M, 10160 Pacific Mesa Bld., San Diego, CA 92121. E-mail: fengl@qualcomm.com.*
- *Z. Li is with the Science Park II, Doctoral Program Computational Mathematics, Johannes Kepler University, Altenberger Straße 66, Linz, A-4040, Austria, and the Key Laboratory of Mathematics Mechanization AMSS, Beijing 100190, China.*

Manuscript received 11 Dec. 2012; revised 15 June 2013; accepted 2 July 2013; published online 21 Aug. 2013.
Recommended for acceptance by M. Brown.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2012-12-0972.
Digital Object Identifier no. 10.1109/TPAMI.2013.161.

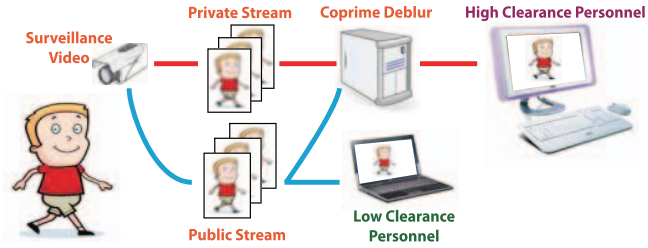

Fig. 1. The processing pipeline of our data-hiding technology for data security in video surveillance.

recognition [10], image filtering [13], and image retrieval [24]. Along the same line of the surveillance work by Yang et al. [8], [7], [12] that aims to separately treat privacy versus nonprivacy features, we present a "partial" visual encryption scheme to allow low-clearance users to partially analyze the visual data without revealing crucial identity information. A comprehensive study of VC can be found in [30].

A natural "partial" encryption solution is to strategically blur the imagery data. A blurred image B can be viewed as the convolution of a latent image L with a blur kernel K. Tremendous efforts have been made on solving the *blind* image deconvolution problem in which neither L nor K is known. Since blind deconvolution is an underconstrained problem, the state-of-the-art solutions rely on regularization to avoid trivial solutions [23], [31], [26]. The latest approaches attempt to use special priors such as image statistics [11], edge and gradient distributions [15], kernel sparsity and continuity [9], or color information [14] for both kernel estimation and image deconvolution. Despite these advances, robust deconvolution remains an open problem in image processing and computer vision [16]. However, because accurate deblurring is inherently difficult, it is ideal for partial visual encryption. The blurry video stream protects identity details while withstanding brute-force deconvolution methods intended to "decrypt" the data. Such "partial" encryption solutions can be viewed as a special case of deidentification, i.e., to obscure the identity but not action. Compared with the state of the art [1], [20], it does not require detection/segmentation/tracking algorithms to locate and black out subjects and can achieve near real-time performance.

Our solution is inspired by recently proposed dual-image deblurring techniques [21], [32], [9]. These methods use a pair of images captured toward the same scene under different aperture/shutter settings. For example, a blurry/noisy image pair can be captured with different shutter speeds. The image pair helps to estimate the kernel and to reduce the ringing artifacts [32] in reconstruction. A dual-blur pair [9] captures the scene under different motion blurs. It then estimates both blur kernels by constructing an equally blurred image pair. These methods suggest that the correlation between the images imposes important constraints that are useful for kernel and latent image estimations. In particular, it indicates that a single image/video stream will be impossible to decrypt (e.g., to a low-clearance user) but if both streams are available, they will be easy to decrypt (e.g., to a high-clearance user).

3 Algorithm Overview

Fig. 1 shows the processing pipeline of our data-hiding technology for data security in video surveillance. We apply coprime blur kernels to the input surveillance video sequence to generate two blurry video streams where all the sensitive information is hidden. One is the public stream and the other is the private stream. The private video sequence is streamed to high-clearance personnel through secure transmission lines. Only high-clearance personnel have access to both the private and public streams. Here, kernel coprimality means when viewing the blur kernels as 2D polynomials, these 2D polynomials are coprime and the degree of their GCD is 1. By using the coprime blur kernels for sensitive data hiding, we design an efficient and robust algorithm to faithfully reconstruct the ground truth input sequence for the high-clearance personnel. We also show that by implementing it on GPUs, we can further improve the computational efficiency of the algorithm to be real-time on a typical business workstation. This would make our data-hiding technology practical for everyday surveillance environments.

3.1 Blur Kernel Selection

The manner in which the blur kernel is implemented in our system is important because it impacts the security of the system and the privacy due to the blurring. The key is to use very large blur kernels when compared to the size of the input image sequence. The larger the kernel size, the more unknown variables there are to solve in an inverse problem, and thus the harder it is to decrypt the surveillance video. We therefore randomly construct the blur kernels of large size and use different kernels for each new frame in the stream. The use of large and temporally varying kernels eliminates temporal coherence and makes blind deconvolution even more difficult.

The end users do not need to know when the blur kernel has changed because they have no means of deblurring the image with only one image. Users with access to both image streams also do not need to know when a blur kernel changes or even the blur kernel itself because the deblurring processes simply relies on possessing two images simultaneously. However, changing the blur kernel frequently may affect the temporal coherence of the blurred regions and could result in distracting flickering of the blurred regions.

3.2 Multilevel Security

The coprime blur scheme described constructs two blurred video streams. One video stream is publicly accessible, but access to the second stream is restricted to users with high clearance and is effectively private. Our scheme is therefore characterized by a public/private encryption/decryption pair. Either of the blur kernels can act as a public encryption key, whereas both blur encrypted blur streams act as a private key and, as a result, restricting access to both streams is paramount. Our approach can be extended to implement a scheme that has more than two levels of security. We do this by dividing the users with access to the private video stream into levels. We restrict a level's access to the latent image by reducing the precision of the bits representing incoming blurred frames. For instance, users with the highest access will have full bit resolution, while others with lower clearance will not be able to access some subsets of the least significant bits. Restricting access to the least significant bits of a pixel will therefore degrade the quality of the latent image reconstruction.

4 Coprime Deblurring

We view the intensity at each pixel as the coefficients of the polynomial and directly treat the image as a matrix. For example, the blurred image B can be transformed to a bivariate polynomial $b(z_1, z_2)$ in z_1 and z_2 as

$$b(z_1, z_2) = \mathbf{z}_1^T \cdot B \cdot \mathbf{z}_2, \tag{1}$$

where $\mathbf{z_1} = [1, z_1, z_1^2, \ldots, z_1^{M-1}]^T$ and $\mathbf{z_2} = [1, z_2, z_2^2, \ldots, z_2^{N-1}]^T$, and M and N correspond to the horizontal and the vertical resolution of the image. Similarly, we can transform the latent image L and the blur kernel K into their corresponding polynomials $l(z_1, z_2)$ and $k(z_1, z_2)$, respectively.

Under the *z-transform*, the convolution of a latent image with the blur kernel becomes the multiplication of their corresponding polynomials, e.g., the z-transformed, blurred image pair B_1 and B_2 can be rewritten as

$$\begin{cases} b_1(z_1, z_2) = l(z_1, z_2) \cdot k_1(z_1, z_2) + n_{e_1}(z_1, z_2) \\ b_2(z_1, z_2) = l(z_1, z_2) \cdot k_2(z_1, z_2) + n_{e_2}(z_1, z_2), \end{cases} \tag{2}$$

where n_{e-1} and n_{e-2} are image noise.

If we assume that the two polynomials $k_1(z_1, z_2)$ and $k_2(z_1, z_2)$ are coprime, we can, in theory, directly obtain l as the approximate GCD of b_1 and b_2 (approximate because of the noise terms):

$$l(z_1, z_2) = \gcd\{b_1(z_1, z_2), b_2(z_1, z_2)\}. \tag{3}$$

The latent image can then be recovered using the inverse z-transform. Despite its simplicity, most solutions [5], [25], [28] are either only suitable for 1D GCD computation or are computationally expensive for the 2D GCD problem. Our solution follows the approximate GCD approach for deblurring a CBP [17].

4.1 Kernel Degree Estimation

To estimate the degree of the blur kernels, we analyze the singularity of the leading principal submatrices of the Bézout matrix [3]. In mathematics, a Bézout matrix (or Bézoutian) is a special square matrix associated with two polynomials. Such matrices can be used to test the stability of a given polynomial and they play an important role in control theory [6]. The Bézout matrix $\tilde{B}(b_1, b_2) = (\tilde{b}_{ij})$ of a given coprime pair $b_1(z_1, z_2)$ and $b_2(z_1, z_2)$ can be generated by first fixing either z_1 or z_2 and performing the N-point discrete Fourier transform (DFT) along the free dimension for both images. We first consider a pair of univariate polynomials $b_1, b_2 \in \mathbb{C}[z] \setminus \{0\}$ of degree m:

$$\begin{cases} b_1(z) = \sum_{i=0}^{m} u_i z^i, & u_m \neq 0, \\ b_2(z) = \sum_{i=0}^{m} v_i z^i, & v_m \neq 0. \end{cases} \tag{4}$$

The Bézout matrix $\tilde{B}(b_1, b_2) = (\tilde{b}_{ij})$ is an $m \times m$ matrix, which satisfies

$$\frac{b_1(x)b_2(y) - b_1(y)b_2(x)}{x - y} = [1, x, x^2, \ldots, x^{m-1}]\tilde{B}(b_1, b_2)[1, y, y^2, \ldots, y^{m-1}]^T.$$

In symbolic algebra, it has been shown [3] that we can derive the degree of the GCD between $b_1(z)$ and $b_2(z)$ as

$$\dim \text{NullSpace}(\tilde{B}(b_1, b_2)) = \deg(\gcd(b_1, b_2)). \tag{5}$$

Assume $\deg(l) = r$, $l(z)$ is the GCD of $b_1(z)$ and $b_2(z)$. From (5), it is easy to verify that $\text{rank}(\tilde{B}(b_1, b_2)) = m - r$. We have developed a fast scheme by checking the rank of the $s \times s$ leading principal submatrix $\tilde{B}_s(b_1, b_2)$ of $\tilde{B}(b_1, b_2)$ as

$$\begin{cases} \det(\tilde{B}_s(b_1, b_2)) \neq 0, & s \leq m - r, \\ \det(\tilde{B}_s(b_1, b_2)) = 0, & s > m - r, \end{cases} \tag{6}$$

specifically by checking if the first

$$1 \times 1, 2 \times 2, 4 \times 4, \ldots, 2^{\lceil \log_2(m-r+1) \rceil} \times 2^{\lceil \log_2(m-r+1) \rceil}$$

leading principal submatrices of $\tilde{B}(b_1, b_2)$ are singular. If so, we will use the size of the singular matrix as the blur kernel size t.

For a 2D polynomial pair, we fix one variable (z_1) with random values and directly use the 1D algorithm and bin the estimated t values. Next, we repeat our algorithm by swapping the dimensions and bin the results. Finally, we choose the kernel size estimate with the most votes. More details of the algorithm can be found in [17].

4.2 Kernel Estimation

We again start from 1D by defining the univariate kernel polynomial $k_1(z)$ and $k_2(z)$ as

$$\begin{cases} k_1(z) = \sum_{i=0}^{m-r} c_i z^i, & c_{m-r} \neq 0, \\ k_2(z) = \sum_{i=0}^{m-r} d_i z^i, & d_{m-r} \neq 0. \end{cases} \tag{7}$$

Recent work [25] in computer algebra has shown that the 1D kernel $\mathbf{c} = [c_0, c_1, \ldots, c_{m-r}]$ satisfies the following property:

$$\mathbf{c} = (J\tilde{B}(b_1, 1))_{m-r+1} \cdot [y_0, y_1, \ldots, y_{m-r-1}, 1]^T, \tag{8}$$

where J is an antidiagonal matrix with 1 as its nonzero entries, and vector $\mathbf{y} = [y_0, y_1, \ldots, y_{m-r-1}]^T$ satisfies

$$C\mathbf{y} = \mathbf{f}, \tag{9}$$

where $\mathbf{y} = [y_0, y_1, \ldots, y_{m-r-1}]^T$, $C = \tilde{B}_{m-r}(b_1, b_2)$, and $-\mathbf{f}$ is a vector formed by the first $m - r$ entries of the $m - r + 1$th column of $\tilde{B}(b_1, b_2)$. Therefore, we can compute the 1D kernel $k_1(z)$ using (8) by solving $\mathbf{y}$ from (9). Similarly, we can solve the kernel $k_2(z)$ by the following equation:

$$\mathbf{d} = (J\tilde{B}(1, b_2))_{m-r+1} \cdot [y_0, y_1, \ldots, y_{m-r-1}, 1]^T, \tag{10}$$

where $\mathbf{d} = [d_0, d_1, \ldots, d_{m-r}]$.

For the 2D case we extend the 1D algorithm by first uniformly sampling the polynomials in the first dimension (z_1) on the unit circle. At each sample point $z_1 = e^{-\frac{2\pi g \mathbf{i}}{t+1}}$, we obtain a pair of 1D polynomials in z_2 and then estimate their coprime kernels. For each CBP kernel, we compose the 1D results at all sample points into a vector and resample it in z_2 at points $z_2 = e^{-\frac{2\pi h \mathbf{i}}{t+1}}$ to form a kernel matrix Φ. We can change the order of our process by sampling in z_2 and estimating the 1D kernels in z_1. This will produce kernel matrix Ψ. Both Φ and Ψ appear to estimate the blur kernel on the 2D Fourier domain and we could apply inverse FFT to recover its corresponding 2D kernel. However, the 1D kernel estimated at each sampled point is the actual one only up to an unknown scale. We thus need to further solve for the scaling factors. Assume g and h are the indices to an element in Φ and Ψ. We need to estimate the scaling factors $\phi(g)$ for every row in Φ and $\psi(h)$ for every column in Ψ. For a kernel matrix of size txt, we have t^2 sampled points and $2t$ unknowns ($\phi(g)$ and $\psi(h)$). Therefore, we form an overdetermined linear system and apply the SVD to solve for the scaling factors. Finally, we apply an inverse FFT to the scale-corrected kernel matrices to obtain the actual 2D kernels.

5 CUDA Implementation

Our implementation of the CBP is composed of five major stages: polynomial evaluation (DFT), kernel degree estimation, 1D cofactor estimation, 2D kernel estimation, and inverse FFT. The processing pipeline is executed primarily on a GPU; however, some algebraic operations on small matrices such as SVD are implemented on the CPU. It was found that for small kernel sizes faster implementations for SVD exist on the CPU. Kernel degrees above 25 have a very detrimental impact on the running time and prevent the algorithm from running in real time. Therefore, SVD is only performed on square matrices of degree no more than 25 in our implementation. We optimize the evaluation of the image polynomials around the unit circle and the final cofactor evaluations for efficient memory access.

Recall that the heart of the deblurring process begins with kernel degree estimation. Evaluating $b_1(z_1, z_2)$ and $b_2(z_1, z_2)$ at some fixed value is simply retrieving the same row or column from the FFTs of the input video frames depending on which direction we have chosen to fix. We then leverage some subset of the threads available to construct the Bézout matrix in parallel. Next, we determine the degree of the blur kernel by performing SVD on the leading principle submatrices and choosing the smallest that is

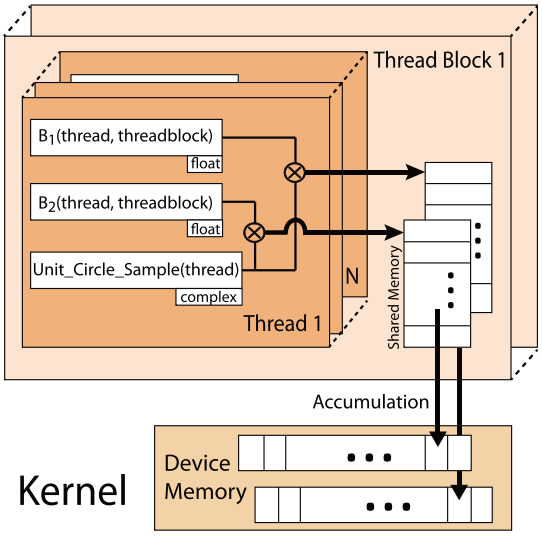

Fig. 2. CUDA device map for polynomial evaluation. Note that each thread only performs two multiplications and that results are written directly into shared memory. After accumulation of shared memory, the result is written back to device memory.

nonsingular. The general approach would be to use a binary search on the full Bézout matrix to find the smallest nonsingular submatrix. This approach, however, initially involves SVD on matrices too large to result in real-time operation of our algorithm. Therefore, we restrict the search to odd leading submatrices of size at least 9×9 and at most 25×25. As previously stated, small SVD calculations are done on the CPU as they are better suited to take advantage of the cache. To further speed the kernel degree estimation, we use four CPU threads to perform SVD on different sized submatrices simultaneously. If all the Bézout submatrices are singular, then the kernel degree must be maximal and 25 is chosen as the kernel degree t.

After determining the degree of the kernel, we once again use the Fourier coefficients although this time for computing 1D cofactor estimates. We now discuss our DFT implementation in greater detail before moving onto the 1D cofactor estimates. Evaluating the video frame in both the z_1 and z_2 directions required two calls to our CUDA DFT kernel. Given a $t \times t$ kernel, there will be $4t$ DFT calculations for the input blur pair. Although CUDA provides DFT operation via its cufft library, its performance for input sizes not power of 2 was unsatisfactory. The DFT kernel also needs to be efficiently generated across a wide range of frame sizes at the level of privacy desired. In our implementation, an N-point DFT in the z_1 direction on a 640×480 frame has 640 thread blocks, each with 480 threads. Similarly, in the z_2 direction, our kernel has 480 thread blocks and 640 threads per block. We have designed our DFT sampling kernel such that each thread only stores three values in the registers at any given time, as shown in Fig. 2. Two of those values are the normalized Fourier coefficients from the same location in the CBP and the third value is the current complex sample from the unit circle. For a $t \times t$ kernel there will be N samples of the unit circle in the z_1 direction DFT and M different samples of the unit circle in the z_2 direction. Each thread performs two complex multiplications, one for each image, and stores the result in that thread block's shared memory array. The shared memory array is then accumulated with $logN$ additions to produce the DFT coefficient for that thread block, after which the next unit circle sample is loaded from global memory into that register. The philosophy behind this approach is to ensure that any value used more than once resides either in a thread's register or shared memory within a thread block and is never loaded from global memory twice.

Once we perform DFT on the image polynomials in the z_1 and z_2 directions, we then begin the 1D cofactor computation by again constructing in parallel t Bézout matrices from z_1 and t more Bézout matrices from the coefficients in the z_2 direction. The vectors and matrices in this stage are only of size t. While this allows us to estimate kernel coefficients in parallel, we need to evaluate (8) and (10) as composite matrices along the diagonal to avoid the overhead of reconfiguring the thread-block structure. Similarly to the kernel degree estimation process these Bézout matrices are copied back to host memory to be factored by CPU SVD routines. The resulting 1D kernel cofactors are copied back to the GPU to be used in the 2D kernel-estimation process. The GCD deblurring algorithm also has a number of intermediate processing steps on small matrices or vectors such as solving for $\mathbf{y}$ in (9). Unlike the previously discussed SVD decomposition, these operations are all $O(n)$, and therefore present significant opportunity for performance gains.

2D kernel estimation continues by evaluating portions of the factored Bézout matrix on the unit circle to produce scaled coefficients of the kernel k_1's Fourier transform. To combine the two estimates of k_1 we must recover the scaling factors on rows of the first k_1 estimate and the columns of the second k_1 estimate. We therefore have $2t$ unknowns and $t \times t$ elements in the kernel estimates producing an overdetermined system. We, therefore, employ a least-squares approach and apply SVD again on the CPU to determine the scale factors. After normalization of the kernel estimates, we compute the average of the two kernel estimates and use it to deblur one frame of input blur pair via the inverse FFT in CUDA's cufft library. We use CUDA's FFT implementation because it relies on texture memory and can store the entire image, unlike our DFT evaluation kernel, which is optimized for small number of DFTs.

6 Results and Analysis

In this section, we demonstrate our deblurring scheme by showing its performance in a number of common surveillance scenarios. We also present running times for our implementation for various blur kernel sizes. We test our algorithm on a PC with a 3 GHz Intel Quad core CPU, 3 GB memory, and a NVIDIA Geforce GTX460 GPU with computing capability 2.1. Video sequences were captured with a Point Gray Flea2 firewire camera. Our blurring and deblurring algorithms were implemented using NVIDIA CUDA SDK 3.2. All input frames had a pixel resolution of 640×480. Blur kernel sizes varied between 19×19 and 25×25 based on the level of anonymity required for the scene. The effect of large kernels on the running time was also evaluated.

In Fig. 3 we show a typical outdoor public surveillance scenario of a sidewalk in an urban area. Note that the frames blurred with 23×23 kernels in the top row still allow the viewer to discern important qualities about the individual in the frame. Despite not being able to identify facial features one is still able to ascertain the color and type of clothing being worn. In addition, one can see that the individual has his hand raised to his ear, and thereby infer that he might be making a telephone call without necessarily being able to see the mobile phone. In this way authorities tasked with surveillance are still able to identify common behavior without having being presented with a sharp image.

In the event that greater scrutiny of the passerby is warranted a high-clearance user would now be able to discern much more about the target, as shown in the bottom row of Fig. 3. In addition to confirming that the passerby was indeed using a mobile phone, they could determine that he was wearing glasses and that his shoes had white tips. In urban surveillance scenes like this it is also common for cameras to inadvertently record the licence plates or passing cars. Notice the right lane of the street is within the camera's field of view. Employing our GCD blurring scheme in

Fig. 3. Deblurring results of a typical surveillance scene in an urban environment.

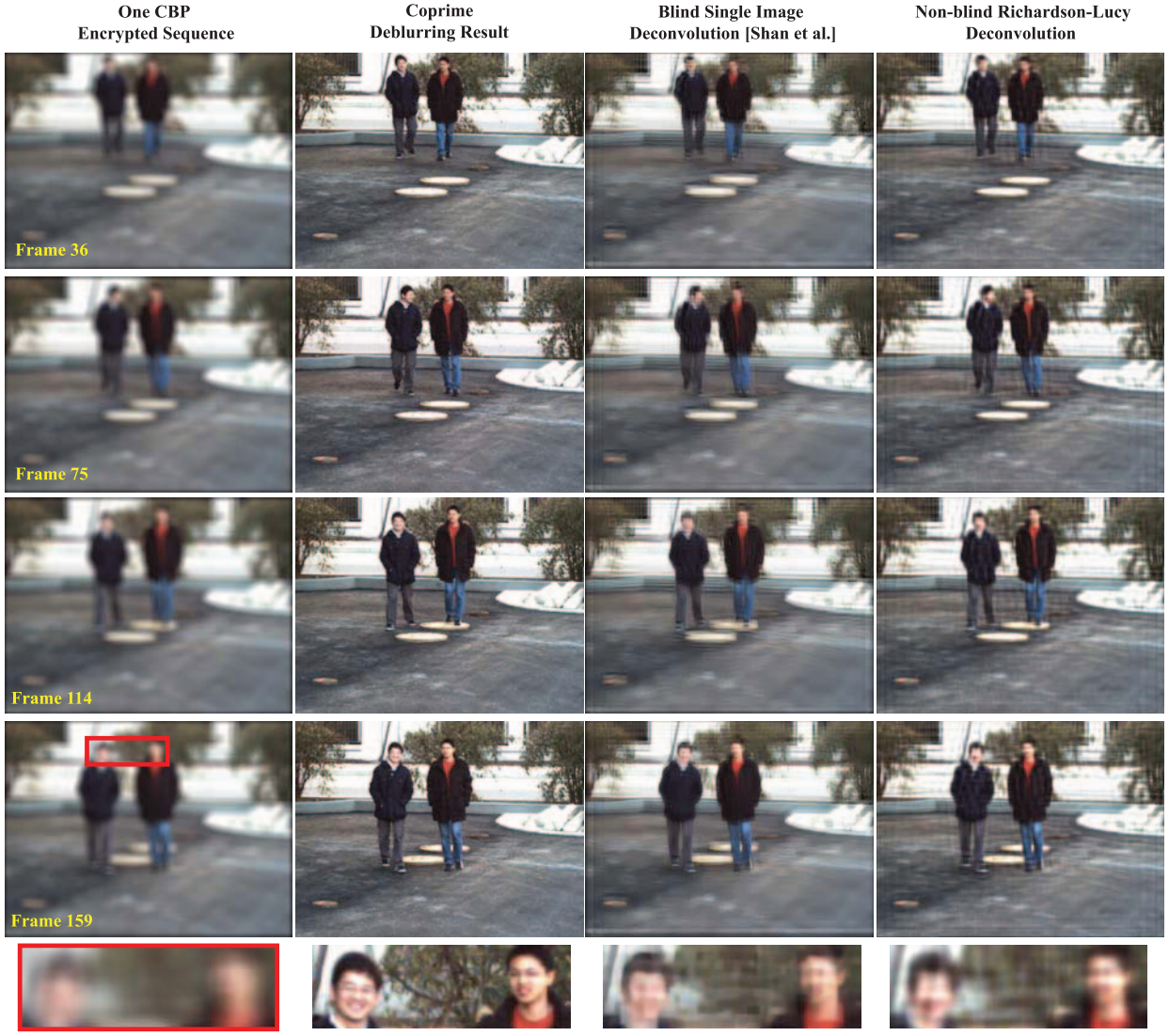

Fig. 4. Decrypted results with different deblurring methods on video frames encrypted by our coprime blurred scheme.

these sidewalk/street scenes would therefore restrict licence plate information to users with high clearance.

In Fig. 4 we present a parking lot surveillance video in which two distant individuals are walking toward the camera. The blur kernel in this series of frames is 19×19. We compare our deblurring results with those from the state-of-the-art blind single image deblurring [23], and the traditional nonblind Richardson-Lucy deconvolution [22]. Here, nonblind deconvolution means we provide the ground truth blur kernels for the traditional Richardson-Lucy algorithm. As can be seen in the figure, both the blind single-image deblurring algorithm [23] and nonblind Richardson-Lucy deconvolution [22] cannot recover the sensitive

Fig. 5. Deblurring results of a door entrance surveillance video sequence.

facial information from the subjects. However, in the second column our GCD deblurred frames have a much greater amount of detail and facial detail in particular. Only in the column presenting our GCD results are the details of the target's distant faces discernible.

In Fig. 5, we show a camera monitoring an entrance, a very common indoor surveillance scenario. Because objects of interest in the scene will be closer to the camera, the size of the blur kernel was increased to 25×25 so as to not preserve too much detail. In the blurred frames it is clear that two people enter the building and what color their clothing is. However, apart from that it is difficult to discern facial details or even the gender of the individuals entering the building. In the deblurred frames on the bottom row it is easy to discern important details about the individuals such as their faces and even that there is a car in the background.

Performance. Table 1 provides running times for our overall algorithm and component processes with various kernel sizes. From the total running times it is evident that our processing pipeline achieves frame rates that approach or exceed current standard video frame rates. Running times include the memory copies of the input frames to the GPU and the memory copy of the resulting unblurred frame back to host memory. We measure the running times using calls to NVIDIA's timer functions. Table 1 shows that of all the processing pipeline stages, the 1D kernel estimation is the most time-consuming. This is primarily due to the time associated with each SVD decomposition and the fact that the number of SVD factorizations grow exponentially with kernel size. However, other factors related to the construction of a greater number of Bézout matrices, copying them to the host memory, and copying the result of the SVD back to GPU scale in proportion to the number of SVD factorizations.

Less computationally expensive stages such as polynomial evaluation and kernel degree estimation contribute little to the overall running time of the algorithm. However, the polynomial evaluation running time increases proportionally with kernel degree t. The running time of the kernel degree estimation stage remains essentially constant as the size of the input remains the same regardless of the blur kernel size. The running time of the last stage, however, increases steadily with kernel size. This increase is due to the small polynomial evaluation carried out on the 1D kernel cofactors to produce the 2D kernel estimate and not to the constant running time to divide by the kernel and the application of the inverse FFT to deblur the image. We also execute our algorithm on the CPU producing running times of ~ 0.509 and ~ 3.404 seconds for 9×9 and 25×25 blur kernels. Running the vast majority of our algorithm on the GPU therefore produces speedups of 21 and 77, respectively.

TABLE 1
Running Times of the Pipeline Stages with Different Kernel Size for Images of Size 640×480 (in Milliseconds)

Pipeline Stage	Kernel Size		
	9×9	17×17	23×23
Polynomial Evaluation	0.65	0.94	1.39
Kernel Degree Estimation	0.81	0.81	0.83
1D Kernel Estimation	18.45	27.83	35.87
2D Kernel Est. and FFT	4.06	4.82	5.9
Total Time	23.97	34.4	43.99

7 Conclusions and Future Work

We have demonstrated a novel coprime blur scheme for visual data hiding in surveillance that partially obscures the image contents. Instead, the ability to recover all of the image information relied on access to a second stream of images and not a predetermined encryption key. Only with access to the second stream is one able to recover the blur kernel and, finally, the latent image. This keyless decryption allows a multitiered security clearance system to be implemented by structuring or modifying the video transmission infrastructure. We have implemented a GPU-based processing pipeline that produces deblurred video at frame rates above 20 fps for relatively large kernel sizes up to 25×25.

Our GPU implementation of the GCD image deblurring algorithm has several limitations. First, our approach may double the bandwidth for video transmission since two blurred sequences are needed for decryption for high-clearance personnel. The bandwidth requirement may reduce the number of instances where deploying our approach is practical. The running time for our approach is proportional to the blur kernel size, and therefore increasing the level of privacy can decrease the frame rate, imposing challenges to time-sensitive tasks such as facial or object recognition. This may be potentially compensated for by coupling the recent GPU recognition framework [4]. Such a privacy-protected recognition scheme has the potential to benefit our current security-focused society.

Acknowledgments

This project was supported in part by the US National Science Foundation (NSF) under grants IIS-CAREER-0845268 and IIS-RI-1016395, and by the US Air Force Office of Scientific Research under the YIP Award. Z. Li was supported by a NKBRPC 2011CB302400, the Chinese NSF grants 60821002/F02, 60911130369 and 10871194.

REFERENCES

[1] P. Agrawal and P. Narayanan, "Person De-Identification in Videos," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 21, no. 3, pp. 299-310, Mar. 2011.

[2] S. Avidan and M. Butman, "Blind Vision," *Proc. European Conf. Computer Vision*, 2006.

[3] S. Barnett, "A Note on the Bezoutian Matrix," *SIAM J. Applied Math.*, vol. 22, no. 1, pp. 84-86, 1972.

[4] M. Bayazit, A. Couture-beil, and G. Mori, "Real-Time Motion-Based Gesture Recognition Using the GPU," *Proc. IAPR Conf. Machine Vision Applications*, pp. 9-12, 2009.

[5] D.A. Bini and P. Boito, "Structured Matrix-Based Methods for Polynomial E-Gcd: Analysis and Comparisons," *Proc. Int'l Symp. Symbolic and Algebraic Computation*, pp. 9-16, 2007.

[6] W.L. Brogan, *Modern Control Theory*, third ed. Prentice-Hall Inc., 1991.

[7] Y. Chang, R. Yan, D. Chen, and J. Yang, "People Identification with Limited Labels in Privacy-Protected Video," *Proc. IEEE Int'l Conf. Multimedia and Expo*, pp. 1005-1008, 2006.

[8] D. Chen, Y. Chang, R. Yan, and J. Yang, "Tools for Protecting the Privacy of Specific Individuals in Video," *EURASIP J. Applied Signal Processing*, vol. 2007, pp. 107-107, 2007.

[9] J. Chen, L. Yuan, C.-K. Tang, and L. Quan, "Robust Dual Motion Deblurring," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[10] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft, "Privacy-Preserving Face Recognition," *Proc. Ninth Int'l Symp. Privacy Enhancing Technologies*, pp. 235-253, 2009.

[11] R. Fergus, B. Singh, A. Hertzmann, S. Roweis, and W. Freeman, "Removing Camera Shake from a Single Photograph," *Proc. ACM Siggraph*, pp. 787-794, 2006.

[12] S. Greiner and J. Yang, "Privacy Protection in an Electronic Chronicle System," *Proc. 34th Ann. IEEE Northeast Bioeng. Conf.*, 2008.

[13] N. Hu, S. Cheung, and T. Nguyen, "Secure Image Filtering," *Proc. Int'l Conf. Image Processing*, pp. 1553-1556, 2007.

[14] N. Joshi, C. Zitnick, R. Szeliski, and D. Kriegman, "Image Deblurring and Denoising Using Color Priors," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 1550-1557, 2009.

[15] A. Levin, "Blind Motion Deblurring Using Image Statistics," *Proc. Neural Information Processing Systems*, 2007.

[16] A. Levin, Y. Weiss, F. Durand, and W. Freeman, "Understanding and Evaluating Blind Deconvolution Algorithms," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 1964-1971, 2009.

[17] F. Li, Z. Li, D. Saunders, and J. Yu, "A Theory of Coprime Blurred Pairs," *Proc. IEEE Int'l Conf. Computer Vision*, pp. 217-224, 2011.

[18] E. Myodo, K. Takagi, S. Miyaji, and Y. Takishima, "Halftone Visual Cryptography Embedding a Natural Grayscale Image Based on Error Diffusion Technique," *Proc. IEEE Int'l Conf. Multimedia and Expo*, pp. 2114-2117, 2007.

[19] M. Naor and A. Shamir, "Visual Cryptography," *Proc. Advances in Cryptology*, 1995.

[20] E.M. Newton, L. Sweeney, and B. Malin, "Preserving Privacy by De-Identifying Face Images," *IEEE Trans. Knowledge and Data Eng.*, vol. 17, no. 2, pp. 232-243, Feb. 2005.

[21] A. Rav-Acha and S. Peleg, "Two Motion-Blurred Images Are Better Than One," *Pattern Recognition Letters*, vol. 26, no. 3, pp. 311-317, 2005.

[22] W. Richardson, "Bayesian-Based Iterative Method of Image Restoration," *J. Optical Soc. Am.*, vol. 62, no. 1, pp. 55-59, 1972.

[23] Q. Shan et al., "High-Quality Motion Deblurring from a Single Image," *ACM Trans. Graphics*, vol. 27, no. 3, p. 73, 2008.

[24] J. Shashank, P. Kowshik, K. Srinathan, and C. Jawahar, "Private Content Based Image Retrieval," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2008.

[25] D. Sun and L. Zhi, "Structured Low Rank Approximation of a Bezout Matrix," *Math. in Computer Science*, vol. 1, no. 2, pp. 427-437, 2007.

[26] Y.-W. Tai, P. Tan, and M.S. Brown, "Richardson-Lucy Deblurring for Scenes Under a Projective Motion Path," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1603-1618, Aug. 2011.

[27] M. Upmanyu, A.M. Namboodiri, K. Srinathan, and C. Jawahar, "Efficient Privacy Preserving Video Surveillance," *Proc. IEEE Int'l Conf. Computer Vision*, pp. 1639-1646, 2009.

[28] J.V.Z. Gathen, M. Mignotte, and I. Shparlinski, "Approximate Polynomial: Small Degree and Small Height Perturbations," *J. Symbolic Computation*, vol. 45, pp. 879-886, 2010.

[29] Z. Wang and G. Arce, "Halftone Visual Cryptography through Error Diffusion," *Proc. IEEE Int'l Conf. Image Processing*, pp. 109-112, 2006.

[30] J. Weir and W. Yan, "A Comprehensive Study of Visual Cryptography," *Trans. Data Hiding and Multimedia Security*, vol. 5, pp. 70-105, 2010.

[31] L. Xu and J. Jia, "Two-Phase Kernel Estimation for Robust Motion Deblurring," *Proc. 11th European Conf. Computer Vision*, pp. 157-170, 2010.

[32] L. Yuan, J. Sun, L. Quan, and H. Shum, "Image Deblurring with Blurred/Noisy Image Pairs," *ACM Trans. Graphics*, vol. 26, no. 3, p. 1, 2007.

[33] Z. Zhou, G. Arce, and G.D. Crescenzo, "Halftone Visual Cryptography," *IEEE Trans. Image Processing*, vol. 15, no. 8, pp. 2441-2453, Aug. 2006.