A Theory of Coprime Blurred Pairs

Feng Li[1] Zijia Li[2] David Saunders[1] Jingyi Yu[1]
[1]University of Delaware, Newark, DE 19716, USA, {feli,saunders,yu}@cis.udel.edu
[2]Key Laboratory of Mathematics Mechanization, AMSS, Beijing 100190, China,lizijia@amss.ac.cn

Abstract

We present a new Coprime Blurred Pair (CBP) theory that may benefit a number of computer vision applications. A CBP is constructed by blurring the same latent image with two unknown kernels, where the two kernels are coprime when mapped to bivariate polynomials under the z-transform. We first show that the blurred contents in a CBP are difficult to restore using conventional blind deconvolution methods based on sparsity priors. We therefore introduce a new coprime prior for recovering the latent image in a CBP. Our solution maps the CBP to bivariate polynomials and sample them on the unit circle in both dimension. We show that coprimality can be derived in terms of the rank of the Bézout Matrix [2] formed by the sampled polynomials and we present an efficient algorithm to factor the Bézout Matrix for recovering the latent image. Finally, we discuss applications of the CBP theory in privacy-preserving surveillance and motion deblurring, as well as physical implementations of CBPs using flutter shutter cameras.

1. Introduction

Image blurs confound many computer vision problems. A blurred image B can be viewed as the convolution of a latent image L with a blur kernel K. Tremendous efforts have been focused on solving the *blind* image deconvolution problem in which neither L nor K is known. Since blind deconvolution is an under-constrained problem, state-of-the-art solutions rely on regularization to avoid trivial solutions [22]. Latest approaches attempt to use special priors such as image statistics [9], edge and gradient distributions [14], kernel sparsity and continuity [7], or color information [12] for both kernel estimation and image deconvolution. We refer the readers to the recent paper by Levin et al. [16] for a comprehensive review.

Recently, a new class of dual-image deblurring techniques have been proposed. These techniques use a pair of images captured towards 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 [28] in reconstruction. A dual-blur pair [7] 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 a similar vein, we present a novel Coprime Blurred Pair (CBP) theory that may benefit a number of computer vision applications.

A CBP is a special subset of dual-blur pairs [7]. A dual-blur pair is obtained by blurring the same latent image with different kernels whereas in a coprime pair, the two kernels are not only different but also ´´coprime``. Consider a 1D example with n as the pixel index, $L(n)$ as the latent image, $K_1(n)$ and $K_2(n)$ as the kernels, and $B_i(n) = L(n) \otimes K_i(n), i = 1, 2$ as the blurred images, where $\otimes$ is the convolution operator. If we take the *z-transform* [4], we can map $L(n)$, $K_i(n)$, and $B_i(n)$ to polynomials $l(z)$, $k_i(z)$, and $b_i(z)$ and we have $b_i(z) = l(z) \cdot k_i(z), i = 1, 2$. We call $[B_1(n), B_2(n)]$ a CBP if their kernel polynomials $k_1(z)$ and $k_2(z)$ are coprime.

We first show that the blurred image contents in a CBP are difficult to restore using conventional blind deconvolution methods based on sparsity priors. We therefore introduce a new coprime prior for recovering the latent image in a CBP. Conceptually, since the blur kernels in a CBP are coprime, we can directly estimate the latent image as the Greatest Common Divisor (GCD) between the ´´blurred``polynomials $b_1(z)$ and $b_2(z)$. In practice, previous approximate GCD algorithms [19] or co-primeness condition [10] are computationally expensive and sensitive to noise. Therefore, they are less suitable for vision applications.

Our solution is to cast the coprime constraint on the blur kernels. We first map the CBP to bivariate polynomials and sample them on the unit circle in both dimension. We show that coprimality can be derived in terms of the rank of the Bézout Matrix [2] formed by the sampled polynomials and we present an efficient algorithm to factor the Bézout Matrix for recovering the latent image. Our new algorithm is

significantly more efficient than the GCD-based approaches and is also more accurate and robust compared to sparsity-based deblurring methods.

While image blurs have been commonly considered adverse in computer vision, we show that coprime blurs may benefit several vision tasks. For example, by measuring the coprimality between the blur images, we can easily estimate the size of the kernel and hence simplify previous guess-and-check approaches. We further explore applying the CBP theory for conducting privacy-preserving surveillance, a field that has attracted an increasing amount of attention in recent years [1, 26]. Our solution is to provide multi-level video surveillance streams by strategically blurring the video contents. Specifically, we form two coprime blurred video streams from the regular surveillance video, where the first is accessible to everyone within the surveillance network and the second will be only accessible to those with a high security clearance. Finally, we discuss physical implementations of CBPs using flutter shutter cameras.

2. Dual-Blur Pairs and Coprime Blur Pairs

In a dual-blur pair, each blurred image B_i is obtained by convolving the same latent image L with a different kernel K_i, plus noise N_{e_i}:

$$B_i = L \otimes K_i + N_{e_i}, \quad i = 1, 2\,. \tag{1}$$

Previous approaches have shown that it may be possible to estimate K_1, K_2, and L directly from B_1 and B_2. For example, Rav-Acha and Peleg [21] attempt to estimate the optimal kernels by constructing an equally blurred image pair: $E = \|B_1 \otimes K_2 - B_2 \otimes K_1\|^2$. We can rewrite $E = \Gamma Q \Gamma^T$, where $Q = [A_2, -A_1]^T[A_2, -A_1]$, A_1, A_2 are the convolution matrices with respect to the blur images B_1 and B_2; $\Gamma = [\Gamma_1; \Gamma_2]$, where Γ_1 and Γ_2 are column vectors unrolled from the kernel matrices K_1 and K_2 respectively.

Additional regularization terms are added to constrain the solution. For example, we can modify the optimization problem as:

$$\begin{aligned} &\underset{\Gamma}{\operatorname{argmin}}\ \Gamma Q \Gamma^T + \lambda \|\Gamma\|^\alpha, \\ &\text{subject to } \gamma_i \geq 0, \sum_{i=1}^{t^2} \gamma_i = 1, \text{ and } \sum_{i=t^2+1}^{2t^2} \gamma_i = 1, \end{aligned} \tag{2}$$

where t^2 indicates the number of elements in each kernel of size $t \times t$ and λ is the weight for the regularization term.

2.1. Gaussian Prior and Sparsity Prior

The regularization term in Eqn. (2) can be interpreted as priors. Two classical examples are the Gaussian prior and the sparsity prior. The Gaussian prior corresponds to using the ℓ_2 norm on the kernel. Its main advantage is that the corresponding optimization problem is convex and can be solved efficiently. The ℓ_2 norm, however, tends to force the elements in the kernel to have identical values, that is, the estimated kernel would be similar to a box filter if we use a large weight λ.

The sparsity prior accounts for the fact that a blur kernel often contains many zeros. An effective way to impose the sparsity prior is to use the ℓ_0 norm on the kernel (i.e., by counting the nonzero entries in Γ). Under the ℓ_0 norm, the corresponding objective function is unfortunately neither differentiable nor convex and finding the optimal solution is NP-hard. To resolve this issue, it is common practice to replace the sparsity prior with a hyper-Laplacian model [15] by using an ℓ_α norm with $\alpha < 1$. In Section 5, we use $\alpha = 0.5$ to emulate the sparsity prior.

Instead of using Gaussian or sparsity priors, we introduce a new coprime prior. Notice that if the first kernel is Gaussian and the second is sparse in the dual-blur pair, it will be challenging to find the proper α for robustly estimating the kernels, as shown in Fig. 1. However, such kernel pairs are generally coprime [13] and we can effectively use coprimality to deblur the imagery data (Section 4).

2.2. Coprime Prior

To precisely define the CBP model, we start with transforming an image to its corresponding bivariate polynomial using the *z-transform*. Specifically, we can 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,

$$b(z_1, z_2) = \mathbf{z}_1^T \cdot B \cdot \mathbf{z}_2, \tag{3}$$

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 $M \times N$ corresponds to the 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:

$$\begin{cases} b_1(z_1, z_2) = l(z_1, z_2) \cdot k_1(z_1, z_2) + n_{e_1}(z_1, z_2) \\ b_2(z_1, z_2) = l(z_1, z_2) \cdot k_2(z_1, z_2) + n_{e_2}(z_1, z_2). \end{cases} \tag{4}$$

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):

$$l(z_1, z_2) = \gcd\{b_1(z_1, z_2), b_2(z_1, z_2)\}. \tag{5}$$

(a) One Image in the CBP (b) Gaussian Prior Results (c) Sparsity Prior Results (d) Coprime Prior Results

Figure 1. Coprime prior vs. sparisity/Gaussian prior. We blur the garden scene image with a pair of kernels, one Gaussian and one sparse (random). (a) shows one of images in the blur pair. (b), (c) and (d) compare the deblurring results using the Gaussian prior, the sparsity prior ($\alpha = 0.5$), and our coprime prior, respectively. The ground truth and the recovered sparsity kernel is shown at the upper right corner in each image. Bottom row shows the close-up views.

The latent image can then be recovered using the inverse *z-transform*.

Although computing the approximate GCD between two polynomials is a well studied problem in symbolic algebra, most existing algorithms are not directly applicable to our problem. Most previous solutions [3, 25, 27] have focused on 1D GCD computation. In essence, similar to Eqn. (2), we could pack the 2D blur images into 1D vectors, and use the 1D algorithm to solve for the latent image and blur kernels in the vector format. This approach however, is not stable for images: an image of size $M \times N$ maps to a 1D polynomial with degree $MN-1$; thus places great demands on computation power and system memory. Moreover, this approach would easily produce trivial results because of accumulated errors. In contrast, our algorithm tackles the 2D GCD problem by solving a series of 1D GCD with degree $M-1$ or $N-1$ with high accuracy.

3. CBP Deblurring

Next we present an approximate algorithm for deblurring a CBP. To better illustrate our algorithm, we start from the 1D (univariate) case and then extend to the 2D (bivariate) case. For clarity, we keep the consistent notations for both cases: we present the polynomials using two variables z_1 and z_2 in the 2D case (e.g., $b(z_1, z_2), l(z_1, z_2)$, and $k(z_1, z_2)$) and use a single variable z to represent the 1D case (e.g, $b(z), l(z)$, and $k(z)$).

3.1. Kernel Degree Estimation

An important yet under-explored step in most existing blind image deconvolution algorithms is to estimate the kernel size t. Previous approaches have commonly adopted a guess-and-check scheme: one can repeat the algorithm multiple times with different kernel sizes and then compare the deconvolution results to find the optimal kernel size. We, in contrast, develop a novel technique that directly recovers the kernel size by analyzing the Bézout matrix [2].

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 [25] and they play an important role in control theory. In this paper, we propose to use the Bézout matrix for testing the coprimality between the kernels.

We first consider a pair of univariate polynomials $b_1, b_2 \in \mathbb{C}[z] \setminus \{0\}$ of degree m:

$$\begin{cases} b_1(z) = \sum_{i=0}^{m} u_i z^i, & u_m \neq 0, \\ b_2(z) = \sum_{i=0}^{m} v_i z^i, & v_m \neq 0. \end{cases} \tag{6}$$

The Bézout matrix $\tilde{B}(b_1, b_2) = (\tilde{b}_{ij})$ is an $m \times m$ matrix where $\tilde{b}_{ij} = \sum_{l=0}^{s}(u_{m-i+m-j-l-1}v_l - u_l v_{m-i+m-j-l-1})$, $i, j = 1, 2, \ldots, m$ and $s = \min(m - i - 1, m - j - 1)$. The Bézout matrix satisfies

$$\frac{b_1(y)b_2(x) - b_1(x)b_2(y)}{x - y} = [x^{m-1}, x^{m-2}, x^{m-3}, \ldots, x, 1] \tilde{B}(f_1, f_2)[y^{m-1}, y^{m-2}, y^{m-3}, \ldots, y, 1]^T.$$

In symbolic algebra, it has been shown [2] that we can derive the degree of the GCD between $b_1(z)$ and $b_2(z)$ as:

$$\deg(\gcd(b_1, b_2)) = \dim \text{NullSpace}(\tilde{B}(b_1, b_2)). \tag{7}$$

Assume $\deg(l) = r$, where $l(z)$ is the GCD of $b_1(z)$ and $b_2(z)$. From Eq.(7), it is easy to verify that the rank of $\tilde{B}(b_1, b_2)$ is $m - r = t$. This suggests that we can directly estimate the kernel degree t in terms of the rank of the Bézout matrix $\tilde{B}(b_1, b_2)$.

To further accelerate the estimation of rank$(\tilde{B}(b_1, b_2))$, we have developed a scheme similar to those in [25] by checking the rank of the first $s \times s$ leading principal submatrix $\tilde{B}_s(b_1, b_2)$ of $\tilde{B}(b_1, b_2)$ as

$$\begin{cases} \det(\tilde{B}_s(b_1, b_2)) \neq 0, & s \leq t, \\ \det(\tilde{B}_s(b_1, b_2)) = 0, & s > t. \end{cases} \tag{8}$$

Specifically, among the first $1 \times 1, 2 \times 2, 4 \times 4, \ldots, 2^{\lceil \log_2(t+1) \rceil} \times 2^{\lceil \log_2(t+1) \rceil}$ leading principal submatrices of $\tilde{B}(b_1, b_2)$, we find the first one that is singular, and will use its rank as the blur kernel size t.

For a 2D polynomial pair, we fix one variable (z_1) and directly use the 1D algorithm as follows:

Algorithm 1. *(Kernel Degree Estimation in z_2)*
Input: $b_1(z_1, z_2)$, $b_2(z_1, z_2)$
Output: *1D kernel degree t*

1. *Evaluate $b_1(z_1, z_2)$ and $b_2(z_1, z_2)$ at some point $z_1 = a$, thus b_1 and b_2 become to be 1D polynomials as $b_1(a, z_2)$ and $b_2(a, z_2)$.*
2. *$i = 0$.*
3. *Build the first $2^i \times 2^i$ leading principal submatrix $\tilde{B}_s(b_1(a, z_2), b_2(a, z_2))$ of the Bézout matrix $\tilde{B}$, $s = 2^i$.*
4. *IF $\tilde{B}_s(b_1(a, z_2), b_2(a, z_2))$ is singular, output t = rank$(\tilde{B}_s(b_1(a, z_2), b_2(a, z_2)))$,*
5. *IF not, $i = i + 1$, go to step 3.*

We apply Alg.1 a number of times (∼50 times in all our examples) by randomly choosing the sample points in z_1 and save the estimated t values. Next, we repeat our algorithm by swapping the dimensions and save the results. Finally, we choose the kernel size estimate with the most votes.

Notice that the core of our algorithm is to determine the rank of matrices $\tilde{B}_s(b_1, b_2)$. We use the singular value decomposition (SVD): let $U\Sigma V$ be the SVD of $\tilde{B}_s(b_1, b_2)$, where U and V are unitary matrix and $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_s)$ with $\sigma_1 \geq \ldots \geq \sigma_k \geq 0$. A brute-force approach is to use a threshold ϵ and count the number of eigenvalues whose values are greater than $\epsilon \cdot \sigma_1$. To make our algorithm more robust, we add an additional constraint: we look for the largest ratio (gap) η between the adjacent eigenvalues, i.e., the rank t of the matrix corresponds to the largest t that maximizes η in $\sigma_t \geq \eta \cdot \sigma_{t+1}$ and satisfies $\sigma_t \geq \epsilon \cdot \sigma_1$. Because we work up from small principal minor sizes by doubling, the run time for kernel degree estimation is O$(t^3 \log(t))$, whereas direct application of SVD to the Bézout matrix would cost O(n^3). This is very advantageous when the kernel size t is small.

3.2. Blur Kernel Estimation

Once we determine the kernel size, we set out to find the actual blur kernels.

1D Kernel. We again start with the 1D case. Recall that $l(z) = \gcd(b_1(z), b_2(z))$, $\deg(l) = r$, and $k_1(z)$ and $k_2(z)$ are the cofactors. We define the univariate polynomial $k_1(z)$ and $k_2(z)$ as

$$\begin{cases} k_1(z) = \sum_{i=0}^{t} c_i z^i, & c_t \neq 0, \\ k_2(z) = \sum_{i=0}^{t} d_i z^i, & d_t \neq 0. \end{cases} \tag{9}$$

Recent work [25, 17] in computer algebra has shown that the 1D kernel $\mathbf{c} = [c_0, c_1, \ldots, c_t]$ satisfies the following property

$$\mathbf{c} = (J\tilde{B}(b_1, 1))_{t+1} \cdot [y_0, y_1, \ldots, y_{t-1}, 1]^T, \tag{10}$$

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

$$C\mathbf{y} = \mathbf{f}\,, \tag{11}$$

where $\mathbf{y} = [y_0, y_1, \ldots, y_{t-1}]^T$, $C = \tilde{B}_t(b_1, b_2)$, and $-\mathbf{f}$ is a vector formed by the first t entries of the $t+1$-th column of $\tilde{B}(b_1, b_2)$. Therefore, we can compute the 1D kernel $k_1(z)$ using Eqn.(10) by solving $\mathbf{y}$ from Eqn.(11). Similarly, we can solve for the kernel $k_2(z)$ by the following equation,

$$\mathbf{d} = (J\tilde{B}(1, b_2))_{t+1} \cdot [y_0, y_1, \ldots, y_{t-1}, 1]^T, \tag{12}$$

where $\mathbf{d} = [d_0, d_1, \ldots, d_t]$. The complete algorithm for finding the GCD between a 1D polynomial pair is shown as follows:

Algorithm 2. *(GCD-based 1D Kernel Estimation)*
Input: $b_1(z)$, $b_2(z)$
Output: $k_1(z)$, $k_2(z)$

1. *Apply Alg.1 to estimate the blur kernel degree t.*
2. *Build C and $\mathbf{f}$ according to the blur kernel degree t.*
3. *Compute $\mathbf{y}$ by solving $C\mathbf{y} = \mathbf{f}$.*
4. *Compute kernel $k_1(z)$ according to Eqn.(10).*
5. *Compute kernel $k_2(z)$ according to Eqn.(12).*

2D Kernel. To find the coprime kernels in a 2D CBP, we first uniformly sample the polynomials in the first dimension (z_1) on the unit circle within the complex domain. 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 using Alg.2. For each CBP kernel, we compose the 1D results at all sample points into two vectors and re-sample them in z_2 at points $z_2 = e^{-\frac{2\pi h \mathbf{i}}{t+1}}$ to form a kernel matrix Φ_1 and Φ_2. 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_1 and Ψ_2.

Note that Φ_1 and Ψ_1 can be viewed as evaluating the blur kernel K_1 in the 2D Fourier domain and we could apply inverse FFT to recover K_1. However, the 1D kernel estimated at each sampled point is the actual one up to an (unknown) scale. We thus need to further solve for the scaling factors. Assume g and h are the indexes to an element in Φ_1 and Ψ_1. We need to estimate the scaling factors $\phi_1(g)$ for every

row in Φ_1 and $\psi_1(h)$ for every column in Ψ_1. Since for a kernel matrix of size $t \times t$, we have t^2 sampled points and $2t$ unknowns ($\phi_1(g)$ and $\psi_1(h)$). Therefore, we form an over-determined linear system and apply the SVD to solve for the scaling factors. Finally we apply an inverse FFT to the scale-corrected matrices to obtain the actual 2D kernels. Alg. 3 describes how to compute kernel K_1.

Algorithm 3. *(GCD-based 2D Kernel Estimation)*
Input: $b_1(z_1, z_2)$, $b_2(z_1, z_2)$, t
Output: *blur kernel* K_1

1. **for** $g = 0 : t$
   (a) *Evaluate* $b_1(z_1, z_2)$ *and* $b_2(z_1, z_2)$ *at* $z_1 = e^{-\frac{2\pi g \mathbf{i}}{t+1}}$ *to generate a pair of 1D polynomials.*
   (b) *Apply Alg.2 to compute a scaled 1D blur kernel* $c_0(e^{-\frac{2\pi g \mathbf{i}}{t+1}})k_1(e^{-\frac{2\pi g \mathbf{i}}{t+1}}, z_2)$.
   (c) *Evaluate the scaled kernel at* $z_2 = e^{-\frac{2\pi h \mathbf{i}}{t+1}}$, $h = 0, 1, \ldots, t$, *to generate the FFT evaluation of kernel* K_1. *We have*

$$k_1(e^{-\frac{2\pi g \mathbf{i}}{t+1}}, e^{-\frac{2\pi h \mathbf{i}}{t+1}}) = \phi_1(g)\Phi_1(g, h). \quad (13)$$

2. *Repeat step 1 but evaluate* b_1 *and* b_2 *first at fixed* z_2 *positions, then we have*

$$k_1(e^{-\frac{2\pi g \mathbf{i}}{t+1}}, e^{-\frac{2\pi h \mathbf{i}}{t+1}}) = \psi_1(h)\Psi_1(g, h). \quad (14)$$

3. *Combine Eq.(13) and (14) to compute the scaling factors* $\phi_1(g)$ *and* $\psi_1(h)$ *by solving the following linear system*

$$\phi_1(g)\Phi_1(g, h) - \psi_1(h)\Psi_1(g, h) = 0, \quad (15)$$
$$\text{for } g = 0, 1, \ldots, t; h = 0, 1, \ldots, t.$$

4. *Compute kernel* K_1 *by applying inverse FFT to*

$$k_1(e^{-\frac{2g\pi \mathbf{i}}{t+1}}, e^{-\frac{2h\pi \mathbf{i}}{t+1}}) = \frac{1}{2}(\phi_1(g)\Phi_1(g, h) + \psi_1(h)\Psi_1(g, h)). \quad (16)$$

Similarly, we can apply Alg.3 to compute the blur kernel K_2. With the recovered kernels, we can reconstruct the latent image, e.g., via regularization-based methods. If we use blur kernels that preserve high frequency such as the ones used in the flutter shutter [20], we can in fact directly apply FFT division to recover the latent image.

Recall that the blur kernel size (degree) in our case is usually much smaller than the latent image resolution. Therefore, our methods are much more efficient than previous methods based on recovering the latent image through GCDs [19]. For instance, we only need to evaluate the polynomials b_1 and b_2 at most t times along each dimension. This smaller number of evaluations can dramatically reduce the computation cost. In Alg.2 for 1D kernel estimation, the number of operations for estimating kernels of degree t are bounded by $O(t^3)$. Therefore, it takes $O(t^4)$ operations to estimate the blur kernels in the 2D Alg.3. For kernels of size up to $O(n^{1/2})$, 2D kernel estimation therefore has complexity $O(n^2)$. To recover the latent image, it takes $O(n^2 \log(n))$ operations for FFT division. Hence, the overall complexity of our algorithm is $O(n^2 \log(n))$ for recovering both the latent image and blur kernels in a CBP of image resolution $n \times n$. Table. 1 shows the processing speed of our CBP-based deblurring algorithm at different image and kernel resolutions.

Image Size	Kernel Size				
	11×11	21×21	31×31	41×41	51×51
320×240	0.23	0.53	0.97	2.13	4.33
640×480	1.05	1.40	2.41	3.57	5.88
1024×768	3.86	4.58	5.50	6.64	9.17

Table 1. The processing speed (in seconds) of our CBP kernel estimation and image deblurring algorithm. All results are obtained on a PC with Intel Pentium D CPU of 3.00GHz and 6GB memory using unoptimized MatLab code.

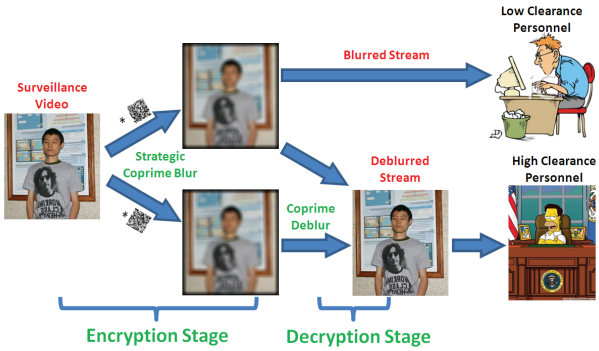

Figure 2. The processing pipeline of our CBP-based privacy-protected video surveillance.

4. CBP for Surveillance

Although image blurs have been commonly considered adverse in computer vision, they also provide a natural way for protecting the image contents and therefore can be used for information hiding. For example, concern about the potential for abuse and the general loss of privacy in video surveillance has also grown along with the number of surveillance cameras. Maintaining security in sensitive environments without impinging on the privacy of the public is a difficult balance to strike. Recently, the notion of "blind vision" has been introduced for addressing such privacy issues. Recent efforts include privacy-protected face detection [1], face recognition [8], image filtering [11], image retrieval [23], and video surveillance [26, 6, 5]. In light of these recent efforts, we present a scheme based on CBP to achieve multi-level identity protection.

We develop a privacy-preserving surveillance scheme by applying coprime blurs to the regular surveillance video data to form a public stream and a private stream. By having their contents blurred, the streams retain the ability to reconstruct an unblurred image if and only if one has access to *both* video streams simultaneously. Further, by carefully choosing the blur kernels, the two streams will have differ-

Figure 3. Decrypted results on video frames encrypted by our CBP scheme using different methods. Video data: courtesy of the EC Funded CAVIAR project/IST 2001 37540.

ent degrees of blurring in order to provide users with lower clearance less access to personally identifiable details while still allowing behavior to be monitored.

Fig. 2 illustrates our CBP surveillance pipeline. We strategically perform two blurring operations by convolving each frame with two *coprime* kernels. At this state both image streams are suitable for distribution to users according to their clearance level. In the final stage an unblurred version of the public video stream will be recovered by performing our CBP-based deblurring algorithm.

Our public/private image terminology is reminiscent of RSA public key cryptography scheme [18]. It utilizes an asymmetrical key scheme that depends on a public key for encryption and a private key for decryption. Even though the system provides all users with the recipient's public key once the sender has encrypted the message with the public key it can only be decrypted with the recipient's private key. Our approach applies the same principle by using a large prime polynomial to blur private details within an image. Since it is very difficult to conduct blind image deconvolution on a single stream [16] and all but users with the highest clearance have access to only one polynomial kernel each image stream by itself acts as a public key.

Just as the security of the RSA algorithm depends on keeping the original two large prime numbers a secret our system relies on the two image streams being separate from one another. The advantage of our approach is that by default an individual image stream respects the privacy of the public and the second stream effectively acts as an additional layer of privacy and as a private key. Therefore as long as an eavesdropper does not have access to both streams the unblurred image cannot be recovered. Moreover, since our CBP model provides a ´´partial``data encryption by strategically blurring the image, low-clearance participants can still analyze the image contents (behavior, motion, etc.), while the data received by low-clearance participants would be completely encrypted and visually useless when using the RSA algorithm.

In Fig. 3, we use two randomly generated blur kernels of size 21×21 to encrypt a surveillance video sequence

of resolution 384×288. We compare the deblurring results obtained by our coprime dual-deblur algorithm (row 4) and single image blind deconvolution [22] (row 2). The video sequence captures a person walking towards the camera. Results produced by single-image deblurring contains strong ringing artifacts that degrade the video quality. By using two blurred streams, our method is able to produce very high quality results, e.g., it accurately reconstructs the face of the subject.

5. Discussions

Limitations of Sparsity Prior. Often, a pair of coprime kernels are both coprime and sparse. This indicates that we can also use the sparsity prior for recovering the kernels. We therefore further compare our CBP technique with the sparsity prior based solution. To effectively model sparsity of the kernel, we adopt a technique similar to the recently proposed hyper-Laplacian scheme [15]. We use the $\ell_{0.5}$ norm for the objective function (Eqn. 2) to emulate the sparsity prior. We then apply the Iterative Re-weighted Least Squares (IRLS) method [24, 15] to find the optimal solution.

Fig. 3 (row 3) shows the recovered video sequence from the sparsity prior using the same dual-blur video pair. Compared with single-image blind deconvolution, the sparsity prior solution is able to greatly suppress the ringing artifacts and produce reasonable results. However, it loses many high frequency details compared to the CBP deblurring results. Similar artifacts can be observed in Fig.4, where we blur the campus scene image using a pair of 35×35 hand-drawn kernels. In our experiments, we found that the kernels produced by the sparsity prior tends to slightly ´´shrink``the kernels since it forces the kernels to be sparse. This explains the loss of high frequency features in the recovered latent image. Our method is able to preserve details but exhibits some ringing artifacts. This is due to the fact that the two hand-drawn kernels are not completely coprime and our kernel estimation contains errors.

Flutter Shutter Implementation. To implement the CBP, the simplest way is to use a process post capture to implement the blur. But with such methods there is some stage after capture where the image stream is not blurred and therefore susceptible to eavesdropping. To address this issue, we suggest an on-capture solution, e.g., by using the fluttered shutter (FS) camera.

The FS camera developed by Raskar et al. [20] opens and closes the shutter during the exposure process according to a pre-determined sequence. The pseudo-random sequence creates a broad-band filter that preserves high frequency details and is suitable for deconvolution. In the original FS work, the shutter sequence is assumed to be known [20]. We instead suggest using a pair of unknown but coprime shutter sequences. For example, these coprime shutter sequences can be implemented by randomly generating

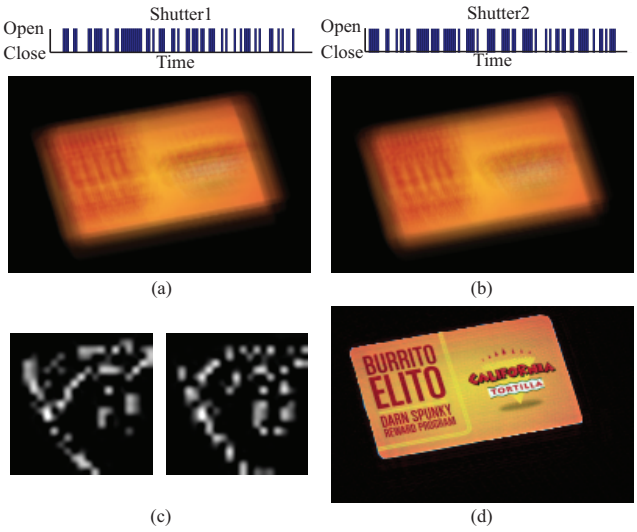

Figure 5. Constructing a CBP using a Flutter Shutter. (a) and (b) emulate two captured images of a moving object under two randomly chosen flutter shutter sequences. (c) shows our recovered blur kernels. (d) shows our recovered latent image.

two shutter sequences since two random polynomials are generally coprime [13].

Unlike the original FS camera that assumes static camera and fast moving targets, we assume that the targets are moving slowly and we emulate motion blurs on the camera side: we can mount the camera on an oscillating motor to introduce motion blurs during capture. Notice that the path of oscillation does not need to be known but rather that the flutter sequence be prime to each other. There are many computer controlled motor systems that can reliably reproduce small controlled oscillations at common frame rates.

In Fig. 5, we emulate our proposed FS implementation of CBP. We pick two random sequences (the top row of Fig. 5) and synthesize a pair of blurred images under 2D motions. We then directly apply our CBP deblurring technique to obtain the blur-free images. Fig.5(d) shows our deblurred results. Since the FS sequences well preserves high frequencies and they are highly coprime (due to random sequence selection), we are able to accurately estimate the two blur kernels and robustly deblur the images. We envision that the FS camera can be directly used to produce CBPs in the future. For example, we can mount a pair of FS cameras on an oscillating motor to introduce artificial motion blurs during capture. This will produce CBP video streams that are suitable for privacy protection.

Limitations and Future Work. We have proposed a novel CBP theory that has the potential to benefit a number of vision applications. We have by far focused on the theoretical foundation of the CBP model and have only applied our theory to privacy-protected video surveillance. As important part of future work, we plan to capture real dual motion-blurred image pairs [7] and test the validity of copri-

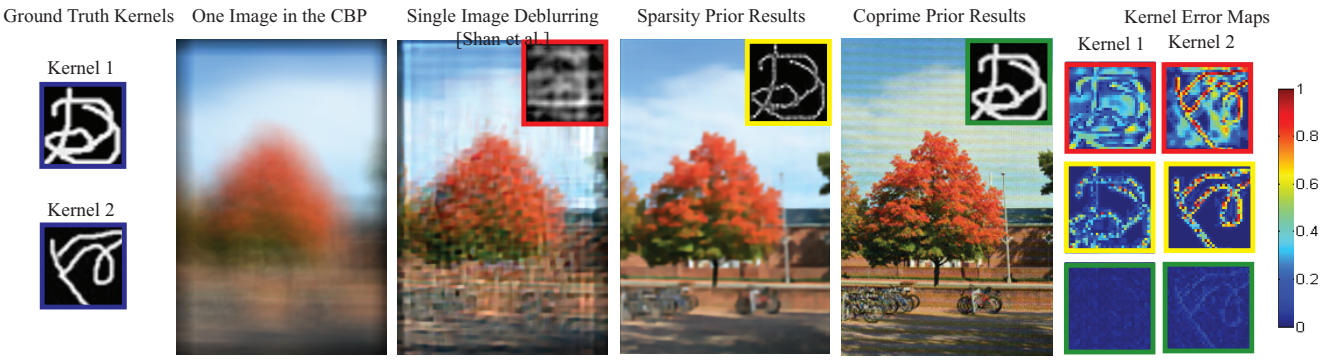

Figure 4. Deblurring results using different priors. The latent image is of resolution 182×273. We use two hand-drawn blur kernels of size 35×35. The recovered kernel is shown at the upper right corner in the result images. Error maps are shown in the last column.

mality on blur kernels caused by real camera motions. We also plan to test the accuracy of our method under different noise levels, intensity precisions (8 vs. 14 bit camera), and registration errors between the image pair. At last, we plan to explore the issue of kernel designs in both CBP-based surveillance and Flutter Shutter based CBP emulation. The choice of how the blur kernel is implemented is important because it impacts the degree of blurs and the security of our privacy-protected system. For example, it is possible to construct a bank of blur kernels and randomly choose a different one for each new frame in the stream.

Acknowledgement

Feng Li and Jingyi Yu were supported in part by the National Science Foundation under grants IIS-CAREER-0845268 and IIS-RI-1016395, and by the Air Force Office of Scientific Research under the YIP Award. Zijia Li was supported by a NKBRPC 2011CB302400, the Chinese National Natural Science Foundation under Grants 60821002/F02, 60911130369 and 10871194.

References

[1] S. Avidan and M. Butman. Blind vision. *In ECCV*, 2006. 2, 5

[2] S. Barnett. A note on the Bezoutian matrix. *SIAM Journal on Applied Mathematics*, 22(1):84–86, 1972. 1, 3

[3] D. A. Bini and P. Boito. Structured matrix-based methods for polynomial e-gcd: analysis and comparisons. In *ISSAC '07*. 3

[4] R. Bracewell. *The Fourier Transform and Its Applications, 3rd ed.* New York: McGraw-Hill, 1999. 1

[5] Y. Chang, R. Yan, D. Chen, and J. Yang. People identification with limited labels in privacy-protected video. In *ICME*, 2006. 5

[6] D. Chen, Y. Chang, R. Yan, and J. Yang. Tools for protecting the privacy of specific individuals in video. *EURASIP J. Appl. Signal Process.*, 2007:107–107, 2007. 5

[7] J. Chen, L. Yuan, C.-K. Tang, and L. Quan. Robust dual motion deblurring. In *CVPR*, 2008. 1, 7

[8] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft. Privacy-preserving face recognition. In *Privacy Enhancing Technologies*, pages 235–253. Springer, 2009. 5

[9] R. Fergus, B. Singh, A. Hertzmann, S. Roweis, and W. Freeman. Removing camera shake from a single photograph. In *ACM SIGGRAPH*, pages 787–794. ACM, 2006. 1

[10] G. Giannakis and R. Heath Jr. Blind identification of multichannel fir blurs and perfect image restoration. *IEEE TIP*, 9(11):1877–1896, 2000. 1

[11] N. Hu, S. Cheung, and T. Nguyen. Secure image filtering. In *ICIP*, pages 1553–1556, 2007. 5

[12] N. Joshi, C. Zitnick, R. Szeliski, and D. Kriegman. Image deblurring and denoising using color priors. In *CVPR*, 2009. 1

[13] E. Kaltofen, Z. Yang, and L. Zhi. On probabilistic analysis of randomization in hybrid symbolic-numeric algorithms. In *SNC'07*, pages 11–17. 2, 7

[14] A. Levin. Blind motion deblurring using image statistics. *In NIPS*, 19, 2007. 1

[15] A. Levin, R. Fergus, F. Durand, and W. Freeman. Image and depth from a conventional camera with a coded aperture. *ACM Transactions on Graphics (TOG)*, 26(3), 2007. 2, 7

[16] A. Levin, Y. Weiss, F. Durand, and W. Freeman. Understanding and evaluating blind deconvolution algorithms. In *CVPR*, pages 1964–1971, 2009. 1, 6

[17] Z. Li, Z. Yang, and L. Zhi. Blind image deconvolution via fast approximate GCD. In *ISSAC*, pages 155–162, 2010. 4

[18] A. Menezes, P. Van Oorschot, and S. Vanstone. *Handbook of applied cryptography*. CRC, 1997. 6

[19] S. U. Pillai and B. Liang. Blind image deconvolution using a robust gcd approach. *IEEE TIP*, 8(2):295–301, 1999. 1, 5

[20] R. Raskar, A. Agrawal, and J. Tumblin. Coded exposure photography: motion deblurring using fluttered shutter. *ACM Trans. Graph.*, 25:795–804, July 2006. 5, 7

[21] A. Rav-Acha and S. Peleg. Two motion-blurred images are better than one. *Patt. Recogn. Lett.*, 26(3):311–317, 2005. 2

[22] Q. Shan, J. Jia, A. Agarwala, et al. High-quality motion deblurring from a single image. *ACM SIGGRAPH*, 2008. 1, 7

[23] J. Shashank, P. Kowshik, K. Srinathan, and C. Jawahar. Private content based image retrieval. In *CVPR*, 2008. 5

[24] C. Stewart. Robust parameter estimation in computer vision. *Siam Review*, 41(3):513–537, 1999. 7

[25] D. Sun and L. Zhi. Structured low rank approximation of a Bezout matrix. *Mathematics in Computer Science*, 1(2):427–437, 2007. 3, 4

[26] M. Upmanyu, A. Namboodiri, K. Srinathan, and C. Jawahar. Efficient privacy preserving video surveillance. In *ICCV*, 2009. 2, 5

[27] J. Von Zur Gathen, M. Mignotte, and I. Shparlinski. Approximate polynomial: Small degree and small height perturbations. *Journal of Symbolic Computation*, 2010. 3

[28] L. Yuan, J. Sun, L. Quan, and H. Shum. Image deblurring with blurred/noisy image pairs. *ACM Transactions on Graphics*, 26(3):1, 2007. 1