applied optics

Compressive imaging via a rotating coded aperture

Michael L. Don,[1,*] Chen Fu,[2] and Gonzalo R. Arce[2]

[1]United States Army Research Laboratory, Aberdeen Proving Ground, Maryland 21005, USA
[2]Department of Electrical and Computer Engineering, University of Delaware, Newark, Delaware 19716, USA
*Corresponding author: michael.l.don2.civ@mail.mil

Received 30 August 2016; revised 27 October 2016; accepted 3 November 2016; posted 3 November 2016 (Doc. ID 274823); published 1 December 2016

Compressive sensing has been used to increase the resolution of sensor arrays, allowing high-resolution images to be obtained from low-resolution or even single pixel sensors. This paper introduces a rotating coded aperture for compressive imaging that has advantages over other sensing strategies. The design of the code geometry is motivated by constraints imposed by the imager's rotation. The block–unblock code pattern is optimized by minimizing the mutual coherence of the sensing matrix. Simulation results are presented, using the final code design to successfully recover high-resolution images from a very small sensor array.

OCIS codes: (110.0110) Imaging systems; (110.1220) Apertures; (110.1758) Computational imaging.

https://doi.org/10.1364/AO.56.00B142

1. INTRODUCTION

Image sensors in the visual spectrum have been following a trend of growing resolution accompanied by decreasing costs. Sensors sensitive in other regions of the electromagnetic spectrum, such as some infrared imagers, have not shared this trend, and their cost remains prohibitively high for many low-cost applications. Compressive sensing (CS) has been used to increase the resolution of low-resolution sensors, or even to produce high-resolution images from a single sensor, thus reducing sensor cost [1,2]. Additionally, each compressive measurement captures more light than in typical imaging, which can reduce system noise. Images are compressed during the measurement process, leading to fewer measurements and shorter acquisition times than other super-resolution techniques, such as raster scanning.

Compressive sensing produces measurements by projecting the image onto a pseudo-random code. This can be accomplished in a variety of ways, including the use of traditional coded apertures. Coded aperture compressive imaging typically uses one or a small number of aperture codes, due to the practical difficulty of swapping the codes during the imaging process. This results in only a few compressed measurements for each sensor, which supports only a modest increase in resolution [3,4]. Researchers at Rice University circumvented this problem by developing a single pixel camera based on a digital micromirror device (DMD) [1]. This programmable array of micromirrors can quickly change its pattern to produce many compressed measurements, allowing the creation of high-resolution images from a single sensor. The DMD, however, is itself an intricate device that complicates the camera design and has limited transmission below near-infrared wavelengths. If a simple method could be found to create many measurements using a coded aperture, it would be an ideal solution.

Work has been performed using a translating motion, moving a coded aperture side-to-side to create multiple measurements, but this motion has undesirable nonlinearities [5]. The design also suffers from the added complexity of the necessary translation equipment. This paper presents a coded aperture design that produces multiple measurements through rotation. Figure 1 shows an example camera architecture consisting of an objective lens, the coded aperture, an imaging lens, and a low-resolution sensor array. The coded aperture and the sensor array are assumed to be the same size, leading to a spacing of twice the focal length to the imaging lens. When mounted on a rotating platform, such as a spinning projectile, it uses the natural rotation to produce the compressed samples. The sensor array moves together with the coded aperture, leading to an extremely simple design with no moving parts. When mounted on a stationary platform, a rotary actuator must be employed to effect rotation, thus complicating the design. Cost-effective rotary actuators have been used in imaging before [6], still retaining the advantage of continuous motion in contrast to the linear motion design. The idea to rotate a coded aperture to produce compressive measurements has been proposed before [7], but this is the first time a coded aperture has been designed specifically for compressive sensing through rotation.

First, a brief introduction to compressive sensing theory is presented. For those unfamiliar with compressive sensing, several

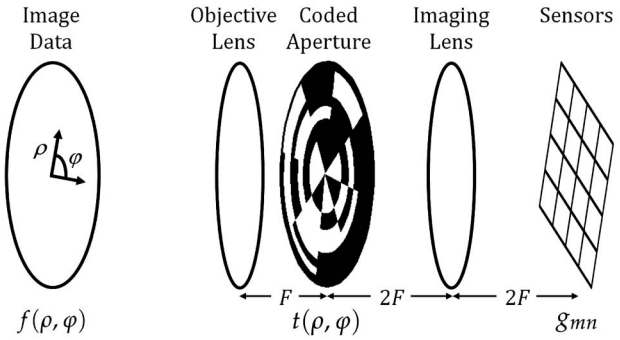

Fig. 1. Architecture of a rotating compressive imager.

longer tutorials can be found in Refs. [8–13]. Next, an initial coded aperture geometry is introduced with the aim of producing a high-resolution image from a single sensor. This design is evaluated, developed, and ultimately rejected. A second geometry is then proposed and evaluated. The initial goal of using a single sensor is abandoned in favor of using a small sensor array. This leads to a final system design and optimized coded aperture geometry. Once the geometry is determined, a further optimization of the code pattern is presented. A list of nomenclature is provided in Appendix B. Note that only rotational motion is modeled in this paper. Use on a spinning projectile will require further research to model all projectile motion [14,15].

2. COMPRESSIVE SENSING

Compressive sensing is a relatively new field that seeks to reduce the number of samples needed to reconstruct sparse data [16,17]. Given a data vector $\mathbf{f} \in \mathbb{R}^N$ and sensing matrix $\mathbf{H} \in \mathbb{R}^{K\times N}$, a measurement vector $\mathbf{g} \in \mathbb{R}^K$ is calculated as

$$\mathbf{g} = \mathbf{H}\mathbf{f}. \tag{1}$$

With a sparse representation $\boldsymbol{\theta}$ of $\mathbf{f}$ in basis $\boldsymbol{\Psi}$, this becomes

$$\mathbf{g} = \mathbf{H}\boldsymbol{\Psi}\boldsymbol{\theta} = \mathbf{A}\boldsymbol{\theta}, \tag{2}$$

with the system matrix $\mathbf{A} = \mathbf{H}\boldsymbol{\Psi}$. Since $K < N$, the system is under-determined, leading to an infinite number of solutions for $\boldsymbol{\theta}$ given measurements $\mathbf{g}$. Compressive sensing theory states [18], however, that if the number of measurements satisfies

$$K \geq C \cdot \mu^2 \cdot \|\boldsymbol{\theta}\|_0 \cdot \log N, \tag{3}$$

the sparsest solution satisfying Eq. (2) will recover $\boldsymbol{\theta}$. C is a positive constant. $\|\boldsymbol{\theta}\|_0$ is the L0-norm, which counts the number of nonzero values in $\boldsymbol{\theta}$. μ is the mutual coherence of $\mathbf{A}$, which is the maximum coherence value μ_{ij} defined as

$$\mu_{ij} = \frac{|\mathbf{a}_i^T \mathbf{a}_j|}{\|\mathbf{a}_i\| \cdot \|\mathbf{a}_j\|}, \qquad 1 \leq i,j \leq N \quad \text{and} \quad i \neq j. \tag{4}$$

In other words, μ is the largest absolute and normalized inner product between the columns in $\mathbf{A}$. The sparsest solution to Eq. (2) can be found through L1-norm minimization:

$$\min_{\boldsymbol{\theta}} \|\boldsymbol{\theta}\|_1 \quad \text{subject to} \quad \mathbf{g} = \mathbf{H}\boldsymbol{\Psi}\boldsymbol{\theta}. \tag{5}$$

Since the number of measurements required for data recovery is related to μ, sensing matrices exhibiting low values of μ increase CS performance. μ, then, can be considered a useful metric for evaluating the performance of a sensing matrix given a chosen sparse basis. We will see that coherence has limited value in designing the coded aperture geometry, but will prove useful in optimizing the block–unblock code pattern once the geometry has been determined.

3. APERTURE CODE DESIGN

A. Initial Geometry

The initial aperture code was designed based on three criteria: the geometry should simplify calculations, preserve sparsity, and promote image quality. Based on these criteria, the initial design with two rings displayed is shown in Fig. 2(a). The polar geometry serves to simplify calculations during rotation. The code sections can be mapped to a matrix as shown in Fig. 2(b), with indices (s, r) denoting spoke s and ring r. This also serves to simplify calculations by allowing algorithms to employ conventional matrix theory. Since the image geometry is typically the same as the code geometry, the code design will directly determine image quality. The matrix mapping preserves contiguous regions of the image, thereby retaining the image's sparsity. All code sections have an equal area, and every ring has an equal height. This leads to fairly uniform pixel shapes that promote image quality. Given these constraints, the number of spokes in ring r is

$$S_r = S_1(2r - 1), \tag{6}$$

where $S_1 = 4$ is the number of spokes in the innermost ring. Instances where $S_1 \neq 4$ serve as code examples that do not directly map to a matrix. Figure 2(c) shows an example image in the polar code geometry, with a visualization of the associated matrix in Fig. 2(d). Polar formatted images have been proposed before for various purposes [19–21], but the design here differs in that it will be optimized specifically for compressive sensing.

The code sections are assigned a random block–unblock pattern as in Fig. 1. Using a single sensor, one measurement is taken at each incremental rotation of the code. Each rotation

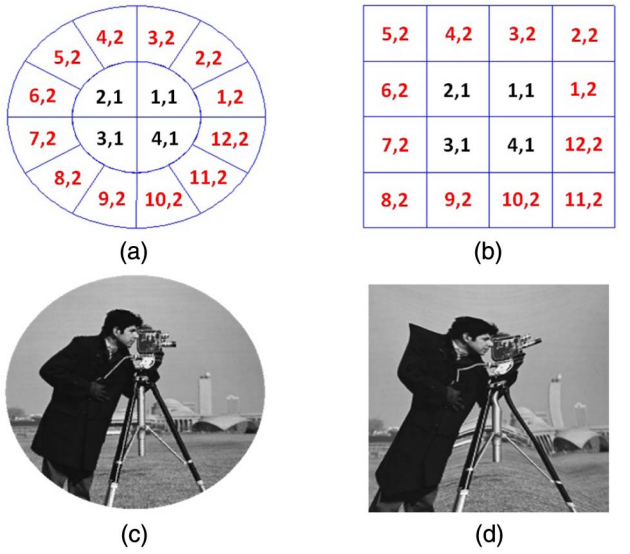

Fig. 2. (a) Initial coded aperture design and (b) matrix mapping. An example 256 × 256 image is shown in (c), with a visualization of the associated matrix in (d).

provides a "new" code in relation to the scene, which is assumed to be stationary. Using this sensing scheme, a problem arises. In typical compressive sensing scenarios, the code can be completely changed between measurements. For example, when using a DMD, a completely new pattern can be programmed for each compressive sample. When acquiring samples through code rotation, the situation is more complicated. Even if the code is rotated the arc length of a section of the outer most ring, thereby completely changing the code sections for that ring, the inner sections will be only partially rotated since they contain fewer sections than the outer ring. Figure 3 illustrates this phenomenon. Moreover, it is likely that even the outer code sections will not be fully rotated between measurements. There are only a limited number of rotations possible before the code is rotated 360 deg and the measurements begin to repeat themselves. In this situation, the rotation between measurements must be decreased to increase the total number of samples. This will typically result in rotations smaller than the arc length of even the outer code sections.

For example, a polar code with 128 rings has a total of about 65K sections, with an outer ring of 1020 sections. Even if the number of compressive samples is only 10% of the total number of pixels, requiring about 6.5K samples, the code can be rotated only about one-sixth of an outer section arc length between samples. It seems likely that this small rotation angle will create a large correlation between samples, degrading the compressing sensing performance.

To understand the effect of small rotations on compressive sensing, the experiment can be simplified to a one-dimensional (1D) example of a single ring, as illustrated in Fig. 4. First, the case of rotating the arc length of a whole code section between each measurement is considered. A ring of N image pixels is represented as a continuous function $f(\rho, \varphi)$, with $0 \le \rho \le \Delta$. A vector of discrete image pixels can then be defined as $\mathbf{f} \in \mathbb{R}^N$, with element f_s on spoke s given by

$$f_s = \iint_{\Omega(f_s)} f(\rho, \varphi)\rho \mathrm{d}\varphi \mathrm{d}\rho, \tag{7}$$

where

$$\Omega(f_s) = \{(\rho, \varphi) | 0 \le \rho \le \Delta, \\ 2\pi(s-1)/N \le \varphi \le 2\pi s/N\}. \tag{8}$$

Refer to the dimension annotations in Fig. 4(a) for an illustration of these pixel boundaries. $t^\ell(\rho, \varphi)$ denotes a code ring at measurement ℓ with the same geometry as $f(\rho, \varphi)$. $\mathbf{t}^\ell$ is then defined with elements

$$t_s^\ell = \iint_{\Omega(f_s)} t^\ell(\rho, \varphi)\rho \mathrm{d}\varphi \mathrm{d}\rho. \tag{9}$$

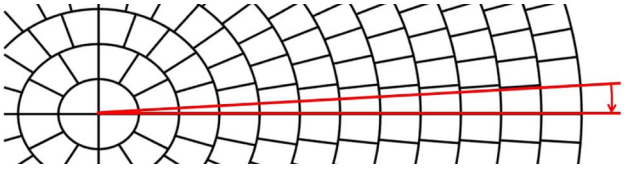

Fig. 3. Rotating the initial code geometry by the arc length of one code section in the outer ring results in only the partial rotation of inner code sections.

The clockwise rotation of an arc length of a whole code section between measurements becomes an upward circular shift of $\mathbf{t}^\ell$, where

$$t_s^\ell = \begin{cases} t_1^{\ell-1}, & \text{if } s = N \\ t_{s+1}^{\ell-1}, & \text{otherwise.} \end{cases} \tag{10}$$

Figure 4 shows this rotation, with the initial code $t^1(\rho, \varphi)$ in Fig. 4(a) rotated by a full code section to become $t^2(\rho, \varphi)$ in Fig. 4(b). The spoke numbering in Fig. 4(a) index $\mathbf{t}^1$, while the those in Fig. 4(b) index $\mathbf{t}^2$, illustrating the rotation in Eq. (10). Each measurement can now be represented as

$$g^\ell = (\mathbf{t}^\ell)^T \mathbf{f}, \tag{11}$$

with K measurements forming

$$\mathbf{g} = [g^1, \ldots, g^K]^T. \tag{12}$$

$\mathbf{t}^\ell$ becomes the corresponding K rows of sensing matrix $\mathbf{H} \in \mathbb{R}^{K \times N}$, with a partial anti-circulant structure:

$$\mathbf{H} = \begin{bmatrix} (\mathbf{t}^1)^T \\ (\mathbf{t}^2)^T \\ \vdots \\ (\mathbf{t}^K)^T \end{bmatrix} = \begin{bmatrix} t_1^1 & t_2^1 & \cdots & \cdots & t_{N-1}^1 & t_N^1 \\ t_2^1 & t_3^1 & \cdots & \cdots & t_N^1 & t_1^1 \\ \vdots & & & & & \vdots \\ t_K^1 & \cdots & t_N^1 & t_1^1 & \cdots & t_{K-1}^1 \end{bmatrix}. \tag{13}$$

Simulations were performed with $\mathbf{H}$ constructed from rotating a randomly generated code with 64 unit area sections. A random 1D image vector was generated with 5% of its coefficients $\boldsymbol{\theta}$ non-zero. In general, the example data in this paper were limited to a 5% sparsity. The data was recovered using a primal-dual algorithm [22] and compared to the original data, with a mean squared error below 1×10^{-3} considered a successful recovery. Multiple simulations were repeated with newly generated random code patterns and image vectors for a given CS ratio, $\kappa = K/N$, with the percent of successful recoveries recorded as the recovery rate. Figure 5 shows the results of four cases. Results using a non-circulant, completely random Bernoulli sensing matrix and a standard basis, i.e., $\mathbf{\Psi} = \mathbf{I}$, are shown as a baseline. Three circularly shifting simulation results are shown for standard, discrete cosine transform (DCT), and wavelet base representations. The standard basis results closely match the baseline case. The wavelet basis shows a small degradation in performance, while the DCT

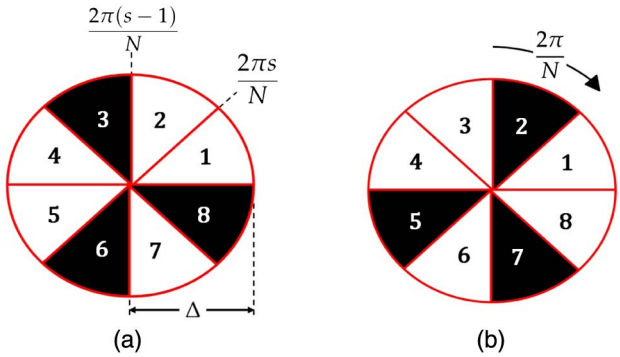

Fig. 4. Example of a single ring with $N = 8$ sections. The block–unblock code is shown is black and white, with the pixel boundaries outlined. Panel (a) shows the code at the first measurement, while (b) shows the code rotated clockwise by one full section for the second measurement.

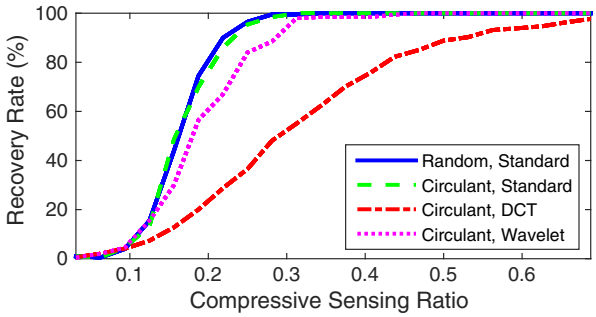

Fig. 5. Recovery rate versus CS ratio K/N for a completely random Bernoulli sensing matrix using the standard basis, and anti-circulant sensing matrices in the standard, DCT, and wavelet bases.

is significantly affected. Previous work has shown similar results for circulant sensing matrices [23].

Figure 6 shows the largest 100 coherence values, defined in Eq. (4), sorted in descending order for the 16 measurement case. It is clear that the DCT has the highest coherence values, which correspond to its inferior recovery results. The other coherence values, however, do not correlate well to CS performance. The coherence values of the completely random sensing matrix are higher than those of the rotating wavelet case, and yet the random case performed better in the recovery simulations. Surprisingly, it appears that CS performance is not always directly related to coherence. Therefore, we will discontinue further analysis of coherence during the design of the code geometry, returning to it in Section 4, dealing with code pattern optimization.

The results for the wavelet basis are encouraging, but as mentioned previously, in order to produce enough measurements the code rotations will have to be less than a single section. Figures 7(a) and 7(b) show an example of rotation by 1/2 of a code section. After the code's first rotation, shown in Fig. 7(b), the code no longer is aligned to the outlined pixel boundaries. This prevents the rotations in $t(\rho, \varphi)$ from being modeled as a rotation in the discrete vector $\mathbf{t}$ as in Eq. (10). Instead, $t(\rho, \varphi)$ is used to model rotation directly, with a rotation of $1/d$ of a section given by

$$t^{\ell}(\rho, \varphi) = t^{\ell-1}(\rho, \varphi + 2\pi/dN). \tag{14}$$

All else remains the same, including the calculation of discrete code elements t_s^{ℓ} by integrating $t^{\ell}(\rho, \varphi)$ over the pixel area as in Eq. (9). This results in intermediate values in the sensing matrix,

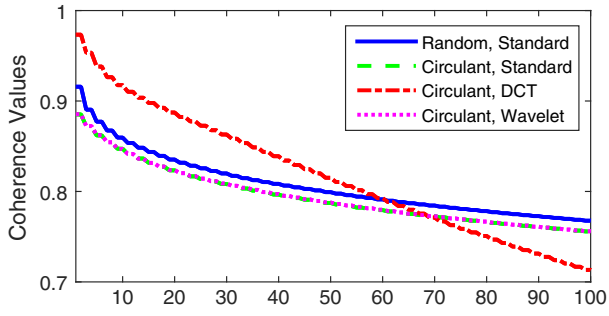

Fig. 6. Largest 100 coherence values, sorted in descending order, for the simulations in Fig. 5 using 16 measurements.

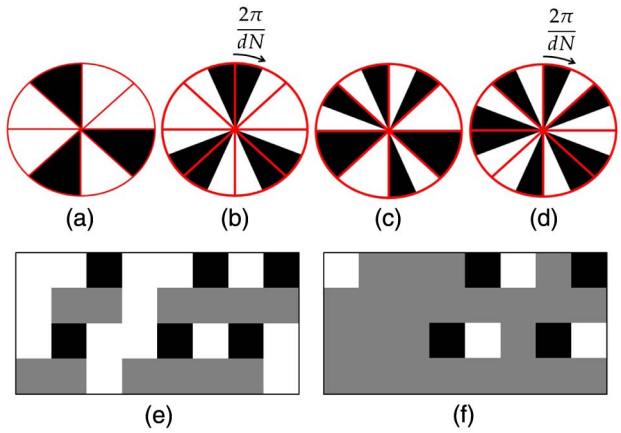

Fig. 7. Example partial rotation of an eight-section code with $d = 2$. (a) The original code, with (b) rotated by 1/2 of a section. The outlined pixel boundaries remain stationary as the code rotates. (e) The resulting sensing matrix with four measurements. (c) An example of a divided code with $d = 2$, giving two code sections per image pixel. The code is rotated by one code section to produce (d). The resulting sensing matrix is shown in (f) with four measurements.

which is no longer anti-circulant. Figure 7(e) shows a visualization of a sensing matrix generated from the code in Figs. 7(a) and 7(b). The first row corresponds to the initial code [Fig. 7(a)], with the next three rows resulting from 1/2 section rotations. The intermediate values occur when a pixel region contains both block and unblock code sections.

The simulations displayed in Fig. 5 were repeated for partial rotations, with the results shown in Fig. 8. These and all further simulations are exclusively in the wavelet basis, where natural images exhibit significant sparsity [24]. A clear decrease of performance is observed as the rotation step size becomes smaller. Obviously, producing more measurements by reducing the rotation step size alone is not a practical solution.

When the code is rotated by partial sections, there is a strong correlation between measurements, which reduces CS performance. A new scheme is required to decrease this correlation. One method is to divide the coded aperture into sections that are smaller than the image pixels. The number of spokes in the code ring becomes dN, with a rotation step size $1/d$ of a code section. This transforms the partial section rotation in Eq. (14)

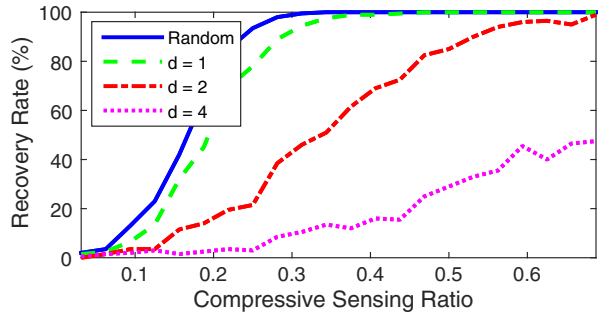

Fig. 8. Recovery rate versus the CS ratio for the baseline completely random Bernoulli sensing matrix, a circulant sensing matrix that rotates by one code section for each measurement, and sensing matrices formed using partial rotations. All simulations use the wavelet basis.

into a full code section rotation. Figures 7(b) and 7(c) illustrate this scheme, with code sections half the size of the image pixels. Figure 7(f) shows the resulting sensing matrix, with intermediate values occurring when a pixel area contains both block and unblock code sections. Simulation results using the divided codes are shown in Fig. 9, indicating that the rotation step size now has a negligible effect on performance.

Now that a workable 1D code has been found, the two-dimensional (2D) case can be readdressed. Just as the single ring required the code to be divided to obtain more measurements, so too the 2D code will have to be divided. This leads to a modified code with sections smaller than the image pixels, as shown in the example in Fig. 10(b). The code has the same number of spokes in every ring, with enough spokes so that each rotation step encompasses a whole code section. The pixel geometry remains the same, as shown in Fig. 10(a). Note that since arc lengths of the inner pixels are greater than those of the outer pixels, the inner pixels contain more code sections than the outer pixels.

Once again, an image with R rings is denoted as $f(\rho, \varphi)$, with a discrete image pixel in ring r and spoke s given by

$$f_{sr} = \iint_{\Omega(f_{sr})} f(\rho, \varphi)\rho \mathrm{d}\varphi \mathrm{d}\rho, \tag{15}$$

where

$$\Omega(f_{sr}) = \{(\rho, \varphi)|\Delta(r-1) \le \rho \le \Delta r, \\ 2\pi(s-1)/S_r \le \varphi \le 2\pi s/S_r\}. \tag{16}$$

Δ is the ring height, and the number of spokes in ring r is given by S_r, as defined in Eq. (6). The total number of pixels is now $N = 4R^2$. (s, r) are mapped to indices (i, j) to create element f_{ij} of matrix $\mathbf{F} \in \mathbb{R}^{\sqrt{N}\times\sqrt{N}}$, as shown in Fig. 2(b) and formally described in Appendix A. $t^{\ell}(\rho, \varphi)$ is the coded aperture at measurement ℓ with discrete sections

$$t^{\ell}_{sr} = \iint_{\Omega(f_{sr})} t^{\ell}(\rho, \varphi)\rho \mathrm{d}\varphi \mathrm{d}\rho, \tag{17}$$

which become elements t^{ℓ}_{ij} of $\mathbf{T}^{\ell}$ using the same mapping as $\mathbf{F}$. Code rotation is modeled as

$$t^{\ell}(\rho, \varphi) = t^{\ell-1}(\rho, \varphi + 2\pi/S), \tag{18}$$

with S denoting the number of spokes of the code. A single measurement is then represented by

$$g^{\ell} = \mathrm{vec}(\mathbf{T}^{\ell})^T \mathbf{f}, \tag{19}$$

where $\mathbf{f} = \mathrm{vec}(\mathbf{F})$. This leads to the complete sensing formulation in Eq. (1) with

$$\mathbf{g} = [g^1, \ldots, g^K]^T, \tag{20}$$

and $\mathbf{H} \in \mathbb{R}^{K\times N}$ given by

$$\mathbf{H} = [\mathrm{vec}(\mathbf{T}^1), \ldots, \mathrm{vec}(\mathbf{T}^K)]^T. \tag{21}$$

Figure 11 shows the results of a simulation using a code with eight rings. The original image [Fig. 11(a)] is shown together with the successfully recovered image [Fig. 11(b)]. The CS ratio was 0.4, with the code rings divided into 102 spokes corresponding to the 102 measurements. This results in code sections smaller than even the pixels in the outer ring, which contains only 60 pixels. The performance here is not as good as the 1D case, where data was recovered reliably with a CS ratio of 0.3. The situation gets worse for larger simulations. Figure 11(c) shows a 16-ring image with a recovery attempt in Fig. 11(d). Here, even using 512 measurements, which is a CS ratio of 0.5, reconstruction remained elusive. Even though the 1D code had positive results, the 2D code performed

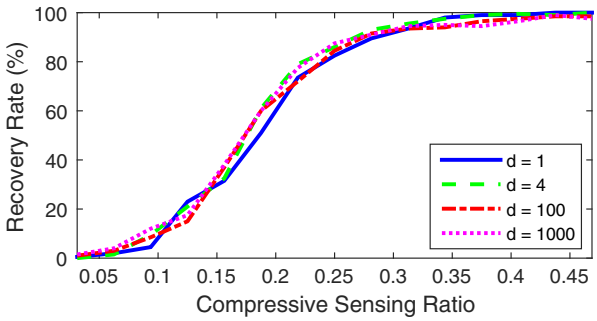

Fig. 9. Recovery rate versus CS ratio for sensing matrices formed using divided code sections.

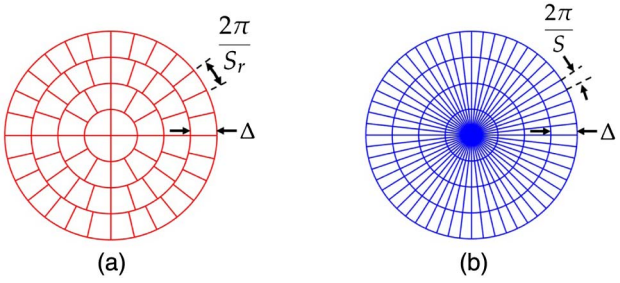

Fig. 10. (a) Initial pixel geometry with S_r spokes in ring r, and (b) updated code geometry, with an equal amount of S spokes in every ring.

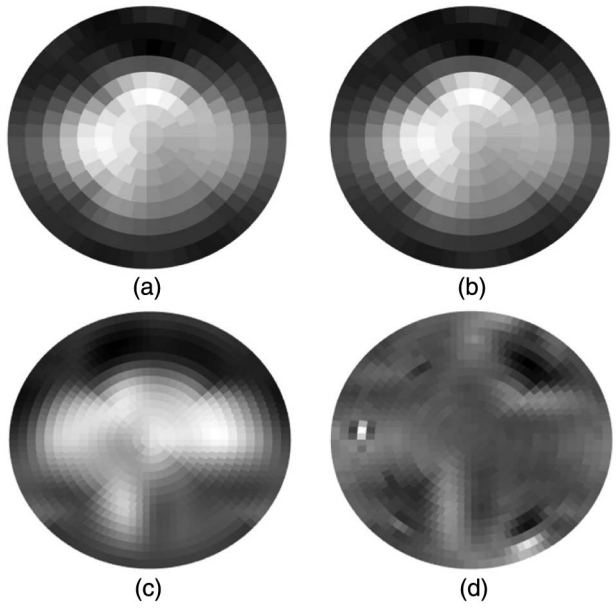

Fig. 11. (a) Eight-ring polar image using the initial pixel geometry with (b) recovery results; and (c) a 16-ring polar image with (d) recovery results.

poorly. One notable difference between these cases is that the number of code divisions for each pixel grows from the outer to inner rings, whereas the number of code divisions remains constant in the 1D case. In order to make the 2D case closer to the 1D case, a new code and pixel geometry is proposed.

B. New Geometry

Figure 12(a) shows the new spoke–ring pixel geometry. Now that all of the R rings have the same number of spokes, the number of spokes per ring S_r becomes a constant S, which more closely mimics the 1D case. The original geometry was chosen to create equal size pixels, which promotes image quality. In this new geometry, the image quality is sacrificed to some extent in order to improve recovery performance. The image quality can still be high, however, when a sufficient number of pixels are used, as demonstrated in the 256 × 256 image in Fig 12(c). This new geometry leads to a direct matrix mapping where the spokes are the rows and the rings are the columns, as shown in Fig. 12(b). Any additional index mapping to (i, j) is now extraneous and can be dispensed with. The height of each ring is now varied in order to keep the image pixel area constant. Δ now denotes only the height of the first ring, with the outer radius of ring r given by $\Delta\sqrt{r}$.

Given this geometry, discrete pixels elements of image matrix $\mathbf{F} \in \mathbb{R}^{S_r \times R}$ are

$$f_{sr} = \iint_{\Omega(f_{sr})} f(\rho, \varphi)\rho \mathrm{d}\varphi \mathrm{d}\rho, \tag{22}$$

where

$$\Omega(f_{sr}) = \{(\rho, \varphi) | \Delta\sqrt{r-1} \le \rho \le \Delta\sqrt{r}, \\ 2\pi s/S_r \le \varphi \le 2\pi(s-1)/S_r\}. \tag{23}$$

The code sections are divided, making the total number of spokes in the code $S > S_r$. Elements of the code matrix $\mathbf{T} \in \mathbb{R}^{S_r \times R}$ can then be formed by integrating $t(\rho, \varphi)$ over an image pixel area:

$$t_{sr}^{\ell} = \iint_{\Omega(f_{sr})} t^{\ell}(\rho, \varphi)\rho \mathrm{d}\varphi \mathrm{d}\rho. \tag{24}$$

The rest of the sensing process is the same as before, with the code rotation and formulation of the sensing process given by Eqs. (18)–(21).

Unfortunately, this geometry also produces poor reconstructions for larger images, with results for a 32 × 32 image similar to those shown in Fig. 11(d). It seems that the performance results seen in the 1D case of divided code sections cannot be translated to a 2D code. A method is required for obtaining more measurements during rotation without dividing the code sections. To accomplish this, the single sensor can be enlarged to a small $M \times M$ sensor array. Although this will somewhat increase sensor cost, it is necessary to improve recovery performance. In addition, multiple measurements can now be obtained simultaneously, decreasing acquisition time or alternatively allowing for a greater sensor integration period. A sufficient number of sensors are required to ensure that enough measurements can be obtained through rotating the code at least one whole code section between measurements. For a desired CS ratio κ, the minimum sensor array size is given by

$$M^2 \ge \frac{\kappa N}{S}, \tag{25}$$

where $N = SR$ is the total number of code sections. Since the code sections are no longer divided, N also represents the total number of image pixels. S and R do not have to be equal. In fact, increasing the ratio of spokes to rings not only decreases the number of sensors required, but also improves image quality by creating more evenly shaped pixels. Figure 13 shows a comparison between using four times the number of spokes as rings on the left, and an equal number of spokes and rings on the right. Both polar images have the same number of total pixels, but the image with more spokes is clearer. Using a spoke-to-ring ratio of 4:1 for a few example image resolutions, Fig. 14 shows the obtainable CS ratio for a given sensor size. Other variables, such as spin rate and sensor integration time, may affect the final choice of sensor size in a practical system.

Figure 15(b) shows an example 8 × 8 polar code overlaid by a 4 × 4 square sensor array in dashed lines. Although a square array is the practical configuration that will be used, we will first consider the simpler case of the polar sensor array in Fig. 15(a),

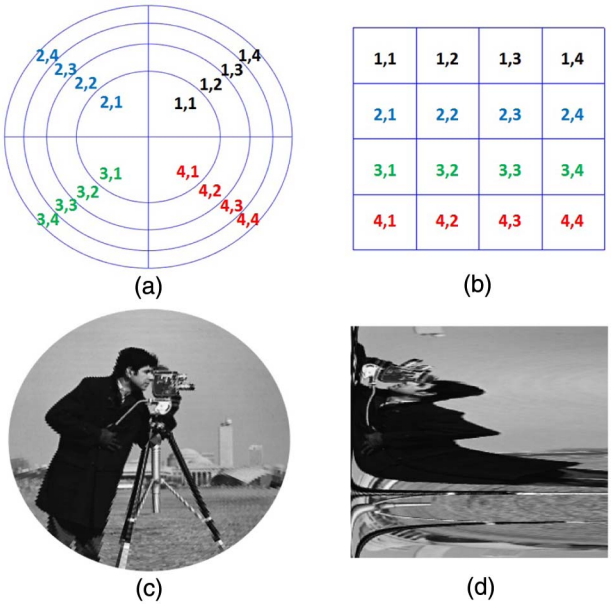

Fig. 12. (a) New polar pixel pattern with an equal number of sections for each ring and (b) the matrix mapping. An example high-resolution image is shown in (c) with a visualization of the matrix mapping in (d).

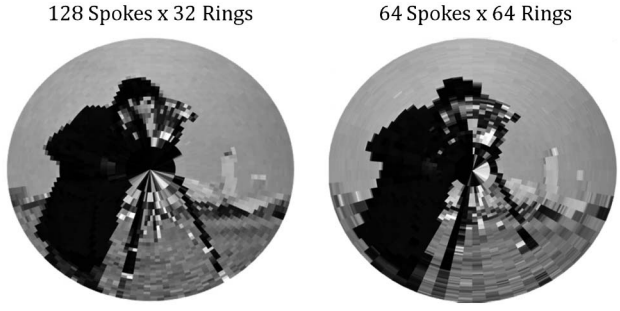

Fig. 13. Polar images using more spokes than rings exhibit higher image quality.

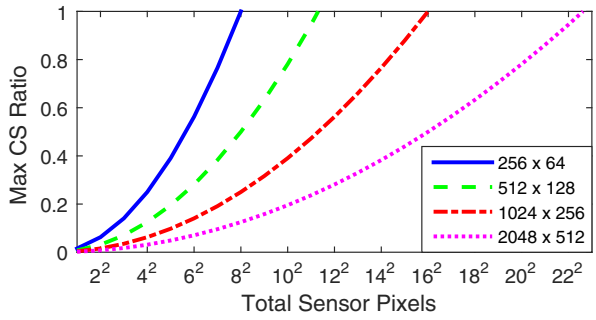

Fig. 14. Achievable CS ratio for a given sensor size. Four curves are shown for four example image resolutions.

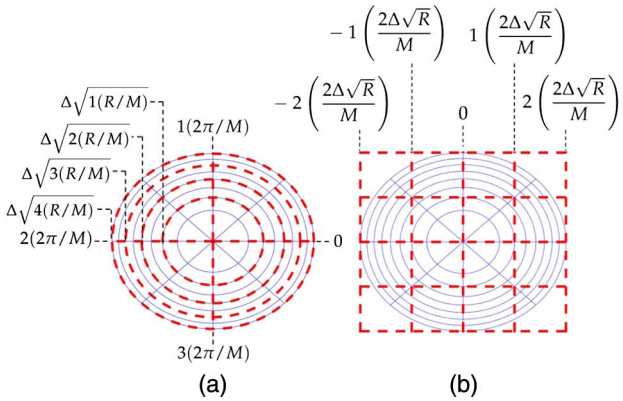

Fig. 15. $M \times M = 4 \times 4$ rectangular and polar sensor grids overlaid on a coded aperture centered at the origin with $R = 8$ rings. In (a), $R/M = 2$ indicating that each sensor ring contains two code rings. In (b), $2\Delta\sqrt{R}/M$ is the width of one sensor.

where the sensor boundaries align with code boundaries. Sensors are modeled by a mask function $w_{mn}(\rho, \varphi)$, indicating the field of view (FOV) of the sensor on spoke m and ring n of the polar sensor array. $0 \le w_{mn}(\rho, \varphi) \le 1$, with 0 indicating the point is completely outside the sensor's FOV while 1 indicates it is completely inside the FOV. For simplicity, assume only the area directly in front of the sensor is within the FOV giving

$$w_{mn}(\rho, \varphi) = \begin{cases} 1, & \text{if } (\rho, \varphi) \in \Omega(w_{mn}) \\ 0, & \text{otherwise} \end{cases}, \tag{26}$$

where

$$\Omega(w_{mn}) = \Big\{(\rho, \varphi) | \Delta\sqrt{(n-1)(R/M)} \le \rho \le \Delta\sqrt{n(R/M)}, \times (m-1)(2\pi/M) \le \varphi \le m(2\pi/M)\Big\}, \tag{27}$$

for a code centered at the origin. Refer to the dimension annotations in Fig. 15(a) for an illustration of these polar sensor boundaries. The sensor mask matrix $\mathbf{W}_{mn} \in \mathbb{R}^{S \times R}$ can then be defined with elements w_{srmn}, indexed by (s, r), indicating the proportion of code section t_{sr} in the FOV of sensor (m, n) as

$$w_{srmn} = \frac{\iint_{\Omega(t_{sr})} w_{mn}(\rho, \varphi)\rho \mathrm{d}\varphi \mathrm{d}\rho}{\pi\Delta^2/S}. \tag{28}$$

Note that the code and image pixel patterns are now identical, making $\Omega(t_{sr}) = \Omega(f_{sr})$.

$\mathbf{T}_{mn}$ can now be defined as the effective code matrix for sensor (m, n), in which everything is masked except for the portion of $\mathbf{T}$ within the sensor's FOV. This is represented by the element-wise product of $\mathbf{T}$ with the sensor mask matrix $\mathbf{W}_{mn}$ given by

$$\mathbf{T}_{mn} = \mathbf{T} \circ \mathbf{W}_{mn}. \tag{29}$$

Since the code is rotated a whole code section between measurements, rotation ℓ can now be modeled as an upward circular rotation of $\mathbf{T}$, denoted as $\mathbf{T}^{\ell}$ and defined by elements

$$t_{sr}^{\ell} = \begin{cases} t_{1r}^{\ell-1}, & \text{if } s = S_r \\ t_{(s+1)r}^{\ell-1}, & \text{otherwise.} \end{cases} \tag{30}$$

The rotation of $\mathbf{W}_{mn}$ can similarly be defined as $\mathbf{W}_{mn}^{\ell}$, making the effective code for sensor (m, n) at rotation ℓ

$$\mathbf{T}_{mn}^{\ell} = \mathbf{T}^{\ell} \circ \mathbf{W}_{mn}^{\ell}. \tag{31}$$

Alternatively, since the code and sensors rotate together, $\mathbf{T}_{mn}$ can be calculated first, and then rotated to form $\mathbf{T}_{mn}^{\ell}$. Equation (19) representing one measurement in the single sensor case now becomes

$$g_{mn}^{\ell} = \mathrm{vec}(\mathbf{T}_{mn}^{\ell})^T \mathbf{f}, \tag{32}$$

where g_{mn}^{ℓ} is the measurement in snapshot ℓ of sensor (m, n). Note that ℓ now represents snapshot ℓ of the whole sensor array, with each snapshot producing M^2 measurements. The total number of snapshots is denoted as K', making the total number of measurements $K = K'M^2$. The measurement vector for sensor (m, n) is

$$\mathbf{g}_{mn} = [g_{mn}^1, \ldots, g_{mn}^{K'}]^T. \tag{33}$$

$\mathbf{H}_{mn} \in \mathbb{R}^{K' \times N}$ denotes the corresponding sensing matrix for sensor (m, n) given by

$$\mathbf{H}_{mn} = [\mathrm{vec}(\mathbf{T}_{mn}^1), \ldots, \mathrm{vec}(\mathbf{T}_{mn}^{K'})]^T. \tag{34}$$

Concatenating $\mathbf{g}_{mn}$ for all the sensors into a single vector $\mathbf{g} \in \mathbb{R}^K$ gives

$$\mathbf{g} = [\mathbf{g}_{11}^T, \ldots, \mathbf{g}_{1M}^T, \mathbf{g}_{21}^T, \ldots, \mathbf{g}_{2M}^T, \ldots, \mathbf{g}_{M1}^T, \ldots, \mathbf{g}_{MM}^T]^T, \tag{35}$$

with the corresponding full sensing matrix $\mathbf{H} \in \mathbb{R}^{K \times N}$ given by

$$\mathbf{H} = [\mathbf{H}_{11}^T, \ldots, \mathbf{H}_{1M}^T, \mathbf{H}_{21}^T, \ldots, \mathbf{H}_{2M}^T, \ldots, \mathbf{H}_{M1}^T, \ldots, \mathbf{H}_{MM}^T]^T. \tag{36}$$

The full sensing process $\mathbf{g} = \mathbf{H}\mathbf{f}$ of an example 2×2 sensor array with two snapshots is written in an expanded form as

$$\begin{bmatrix} g_{11}^1 \\ g_{11}^2 \\ g_{12}^1 \\ g_{12}^2 \\ g_{21}^1 \\ g_{21}^2 \\ g_{22}^1 \\ g_{22}^2 \end{bmatrix} = \begin{bmatrix} \mathrm{vec}(\mathbf{T}^1 \circ \mathbf{W}_{11}^1)^T \\ \mathrm{vec}(\mathbf{T}^2 \circ \mathbf{W}_{11}^2)^T \\ \mathrm{vec}(\mathbf{T}^1 \circ \mathbf{W}_{12}^1)^T \\ \mathrm{vec}(\mathbf{T}^2 \circ \mathbf{W}_{12}^2)^T \\ \mathrm{vec}(\mathbf{T}^1 \circ \mathbf{W}_{21}^1)^T \\ \mathrm{vec}(\mathbf{T}^2 \circ \mathbf{W}_{21}^2)^T \\ \mathrm{vec}(\mathbf{T}^1 \circ \mathbf{W}_{22}^1)^T \\ \mathrm{vec}(\mathbf{T}^2 \circ \mathbf{W}_{22}^2)^T \end{bmatrix} \mathbf{f}. \tag{37}$$

Figure 16 illustrates the construction of the $\mathbf{H}$ matrix in Eq. (37) using a 4×4 coded aperture. A visualization of the continuous code function $t(\rho, \varphi)$ with the corresponding $\mathbf{T}$ matrix is shown in the upper left. A real coded aperture would

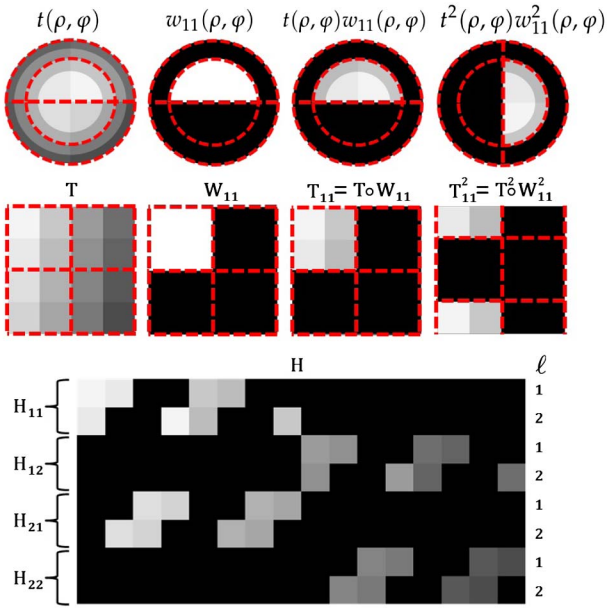

Fig. 16. Example code and FOV functions for a polar sensor array with visualizations of their corresponding matrices.

have a random pattern of block–unblock sections, but here a gradated pattern is used for illustration purposes. A 2 × 2 polar sensor array is superimposed on the code with a dashed line. Next, $w_{11}(\rho,\varphi)$ and $\mathbf{W}_{11}$ are shown, which mask everything outside the FOV of sensor (1, 1). $\mathbf{T}_{11} = \mathbf{T} \circ \mathbf{W}_{11}$ is displayed next, along with its the continuous equivalent $t(\rho,\varphi)w_{11}(\rho,\varphi)$. Finally, the code and $\mathbf{T}_{11}$ matrix rotated for the second snapshot are shown. The complete sensing matrix $\mathbf{H}$ is shown beneath for $K' = 2$ snapshots, with the submatrices $\mathbf{H}_{mn}$ indicated on the left, and the rows corresponding to snapshot ℓ shown on the right. Note that the first row of $\mathbf{H}$ is $\mathrm{vec}(\mathbf{T}_{11}^1)$, and the next row is $\mathrm{vec}(\mathbf{T}_{11}^2)$, with the upward rotation of $\mathbf{T}_{mn}$ creating a leftward rotation of the corresponding elements in the rows of $\mathbf{H}_{mn}$.

For a square sensor array, m and n now index the rows and columns of the sensors, respectively. All of the previous calculations remain the same, except that $w_{mn}(\rho,\varphi)$ is now expressed in Cartesian coordinates as

$$w_{mn}(x,y) = \begin{cases} 1, & \text{if } (x,y) \in \Omega(w_{mn}) \\ 0, & \text{otherwise} \end{cases}, \tag{38}$$

where

$$\Omega(w_{mn}) = \Bigg\{ (x,y) \Bigg| \left((n-1) - M/2\right)\left(\frac{2\Delta\sqrt{R}}{M}\right) \le x \le (n - M/2)\left(\frac{2\Delta\sqrt{R}}{M}\right), \left(M/2 - (m-1)\right)\left(\frac{2\Delta\sqrt{R}}{M}\right) \le y \le (M/2 - m)\left(\frac{2\Delta\sqrt{R}}{M}\right) \Bigg\}. \tag{39}$$

Refer to the dimension annotations in Fig. 15(b) for an example of these boundaries. All of the other equations remain the same, with Eq. (28) used again to determine the sensor mask matrix $\mathbf{W}_{mn}$, and the effective code matrix for sensor (m, n) at snapshot ℓ once again given by $\mathbf{T}_{mn}^{\ell} = \mathbf{T}^{\ell} \circ \mathbf{W}_{mn}^{\ell}$.

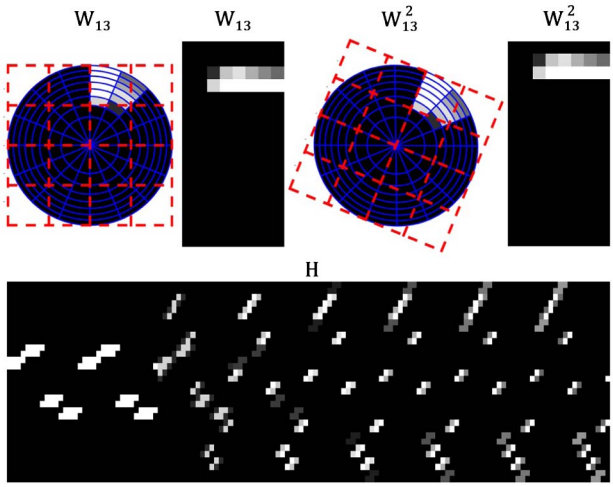

Fig. 17. Visualization of FOV matrices for a 16 × 8 polar code with a 4 × 4 square sensor array. The resulting sensing matrix for two snapshots, and an all unblocked code is shown on the bottom.

Figure 17 illustrates an example code with 16 spokes and eight rings with a 4 × 4 square sensor array. In the upper left, $\mathbf{W}_{13}$ is visualized in the polar geometry with the sensor boundaries overlaid with a dashed line. Recall that matrix element w_{sr13} is the proportion of code section t_{sr} in the FOV of sensor (1, 3), producing intermediate values for those sections only partially within the sensor. Matrix $\mathbf{W}_{13}$ is also shown visualized in a rectangular matrix format, with (s, r) indexing rows and columns. On the right, $\mathbf{W}_{13}^2$ is shown for the second snapshot, with the corresponding polar and rectangular visualizations. On the bottom, the sensing matrix is displayed for $K' = 2$ snapshots. For illustration purposes, an all unblock code was used, making $\mathbf{H}$ dependent on the $\mathbf{W}_{mn}$ matrices alone. The rows of $\mathbf{H}$ are vectorizations of the $\mathbf{W}_{mn}$ matrices, with the same structure as illustrated for $\mathbf{H}$ in Fig. 16.

Figure 18 compares the recovery performance of an 8 × 8 polar sensor array to the square sensor configuration for an example 256 × 64 image. Unlike previous simulation results, the sparsity of the image was not limited to 5%. In addition, the gradient projection for the sparse reconstruction (GPSR)

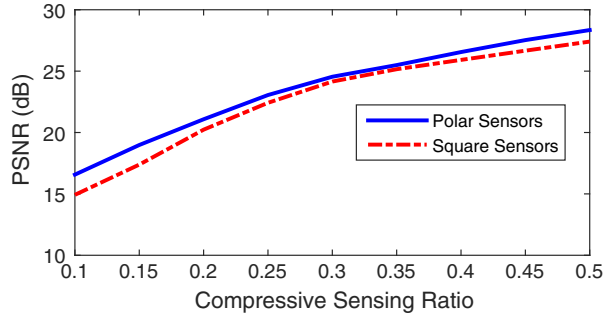

Fig. 18. Comparison between the recovery performance of polar and square sensor arrays.

algorithm was employed for recovery which performed better for larger images [25]. Now that multiple sensors are being used and full code sections are completely rotated, image recovery is possible, with an average peak signal-to-noise ratio (PSNR) of 28 dB obtained for the polar sensors, and 27 dB for the square sensors using a CS ratio of 0.5. Figure 19 shows simulation results using a 8 × 8 square sensor array with a 512 × 128 image and a CS ratio of 0.5. The recovered image has an improved PSNR of 31.5 dB due to the increased sparsity of the larger image. Use of this coded aperture design in other scenarios that exhibit greater sparsity, such as spectral imaging, will similarly demonstrate enhanced performance [26].

Compressive sensing theory provides guarantees of not only exact data recovery in ideal conditions, but also ensures that recovery is stable in noisy environments for well-formed sensing matrices, i.e., measurements corrupted with bounded noise produce recovery results with bounded error [16]. Figure 20 shows the noise performance of 8 × 8 polar and square sensor arrangements for a 256 × 64 image. The image pixel levels were normalized to one unit, with Gaussian noise applied to the sensor measurements. Since the image is not completely sparse, the non-zero coefficient values dominate the noise at low noise levels, resulting in a relatively flat response. At higher noise levels the measurement noise dominates, leading to a roughly 1 dB drop in the recovered PSNR for every 1 dB increase in measurement noise. The stable response to measurement noise demonstrates the robustness of the sensing matrix and consequently the coded aperture geometry. Further noise modeling of vibration effects, rotation center misalignment, and sensor specific noise sources will be important for the practical implementation of the imager, but is outside the scope of this paper. Some noise sources can be mitigated by incorporating them into the model, such as blurring effects due to finite sensor integration time [26]. Additional research is required to determine if similar techniques can be applied to other noise sources, or if external image stabilization hardware will be required.

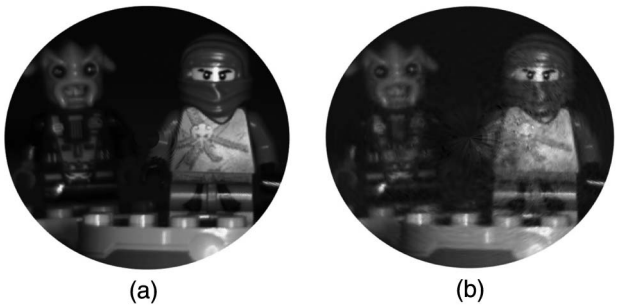

Fig. 19. (a) Original and (b) recovered images using the new coded aperture geometry with a rectangular sensor array.

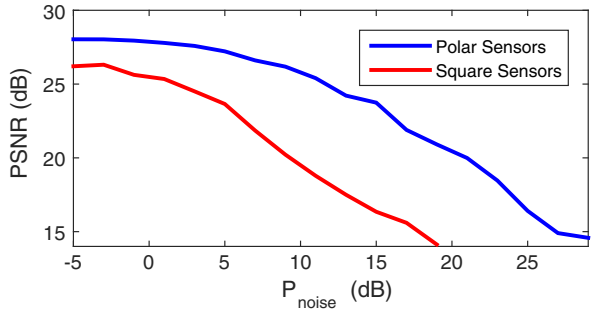

Fig. 20. Recovered image PSNR versus sensor measurement noise power for 8 × 8 polar and square sensor arrangements for an example 256 × 64 pixel image.

4. CODE PATTERN OPTIMIZATION

A. Code Pattern Optimization via Coherence Minimization

In the preceding cases, block and unblock code sections appeared with equal probability. The code pattern can be optimized, however, to create a better-conditioned sensing matrix. Several code optimization approaches have been proposed, including developing the restricted isometry property (RIP) in coded aperture snapshot spectral imaging (CASSI) [27], applying blue noise patterns [28–31] as the distribution of pixelated polarizer [32], and implementing the uniform sensing criteria for computing tomosynthesis aperture code design [33].

Another line of sensing matrix optimization lies in the mutual coherence of the system matrix. As previously noted, a sensing matrix with lower coherence is generally better conditioned, leading to better reconstruction quality. Efforts have been made to optimize an unconstrained sensing matrix via coherence minimization [34–36]. However, these optimization methods cannot be applied here where the sensing matrix is constrained by the binary block–unblock code, the code rotation, and the sensor arrangement. Direct binary search (DBS) algorithms [24,37], however, were designed to solve binary matrix related optimization problems and can be used here as well.

A DBS algorithm involves flipping and swapping operations to each binary section with its eight neighbor sections. These operations are performed to each of the coded aperture sections sequentially while the changes of the cost function are recorded. Certain operations are kept when the cost function value has the largest reduction. The algorithm runs iteratively until the reduction of the cost function value is no longer observed. A local minimum of the cost function is thus obtained. The applications of the DBS in previous coded aperture optimization approaches can be found in Refs. [32,33].

The optimization algorithm minimizes the coherence μ defined as the maximum of Eq. (4). However, the calculation of μ requires heavy computing. Furthermore, the DBS algorithm evaluates the changes of the cost function value with every valid swapping or toggling operation, requiring multiple μ calculations for each aperture code section. Thus, it is important to simplify the calculation of coherence based on the limited changes in the aperture code section operations. Note that the coherence calculation is identical to

$$\mu = \max_{i<j} \left| \frac{(\mathbf{A}^T\mathbf{A})_{ij}}{\sqrt{(\tilde{\mathbf{a}}\,\tilde{\mathbf{a}}^T)_{ij}}} \right|, \tag{40}$$

$$\mathbf{G} = \tilde{\mathbf{A}}^T\tilde{\mathbf{A}}, \tag{41}$$

where vector $\tilde{\mathbf{a}}$ contains the diagonal elements in $\mathbf{A}^T\mathbf{A}$. To avoid calculating $\mathbf{A}^T\mathbf{A}$ directly, it is possible to construct

the desired matrix with its element values before changes, and the changes of sensing matrix $\mathbf{A}$ due to the section swapping and flipping operations. Denote the newly changed $\mathbf{A}$ as $\mathbf{A}'$. Then the change in the sensing matrix is $\mathbf{A}_\Delta = \mathbf{A}' - \mathbf{A}$. Thus, the update of $\mathbf{A}^T\mathbf{A}$ is

$$(\mathbf{A}^T\mathbf{A})' = (\mathbf{A}^T + \mathbf{A}_\Delta^T)(\mathbf{A} + \mathbf{A}_\Delta) = \mathbf{A}^T\mathbf{A} + (\mathbf{A}^T\mathbf{A})_\Delta + (\mathbf{A}^T\mathbf{A})_\Delta^T, \quad (42)$$

where $(\mathbf{A}^T\mathbf{A})_\Delta = \mathbf{A}_\Delta^T(0.5\mathbf{A}_\Delta + \mathbf{A})$. As only a portion of rows in $\mathbf{A}_\Delta$ contain non-zero elements, the calculation of $(\mathbf{A}^T\mathbf{A})'$ is simplified by constructing $(\mathbf{A}^T\mathbf{A})_\Delta$ instead.

Using the same approach, the calculation of $\mathbf{A}$ can be converted into the calculation of $\mathbf{A}_\Delta$ given by

$$\mathbf{A}_\Delta = \mathbf{H}_\Delta\mathbf{\Psi}, \quad (43)$$

where $\mathbf{H}_\Delta$ represents the changes in $\mathbf{H}$. This matrix multiplication is simplified by only multiplying the non-zero columns in $\mathbf{H}_\Delta$ by the corresponding rows in $\mathbf{\Psi}$. $\mathbf{H}_\Delta$ is easily obtained from the changes in aperture code $\mathbf{T}$.

To further accelerate the algorithm, a more strict checking process is performed for validating the swapping and toggling operation to each aperture code section. In this process, the location of μ is recorded as (i, j). For any swapping or toggling operation that changes the coded aperture pattern, the corresponding coherence value μ_{ij} in position (i, j) is first calculated. For any $\mu_{ij} \geq \mu$, the updated coherence $\mu' \geq \mu_{ij} \geq \mu$, making the corresponding section operation invalid. Thus, the algorithm will continue to calculate the updated cost function value only if $\mu_{ij} < \mu$.

B. Simulation

To evaluate the optimization algorithm, we first test the algorithm with a 64^2 resolution random aperture code. 64 snapshots are simulated with a 4×4 sensor array, giving a CS ratio of 0.25. The initial coherence of the system is 0.92. The algorithm ends in 14 iterations, reaching a local minimal coherence of 0.36. Figure 21 shows the convergence of the coherence during each iteration.

The algorithm was then tested with higher resolution. Figure 22(a) is a random block–unblock coded aperture with 128^2 polar sections; 128 snapshot measurements were simulated with an 8×8 sensor array to construct a sensing matrix with $\mu = 0.83$. Using this random pattern as the initial code, the resulting optimized aperture code is displayed in Fig. 22(b) with the coherence reduced to 0.44.

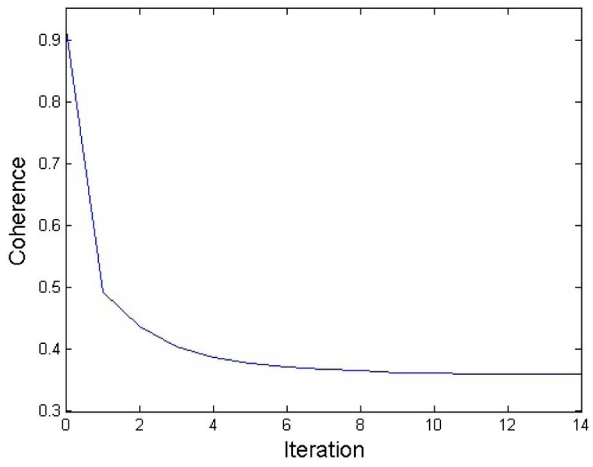

Fig. 21. Coherence convergence is recorded in 14 iterations.

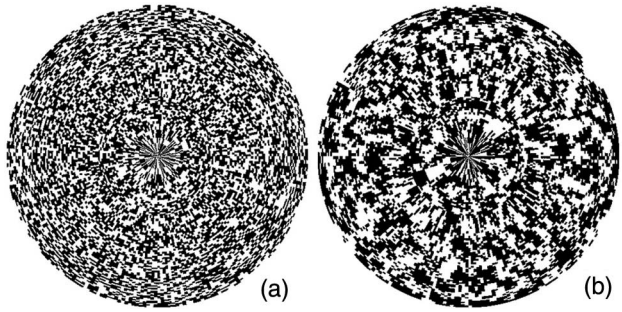

Fig. 22. (a) Comparison of initial random aperture code and (b) the optimized code pattern.

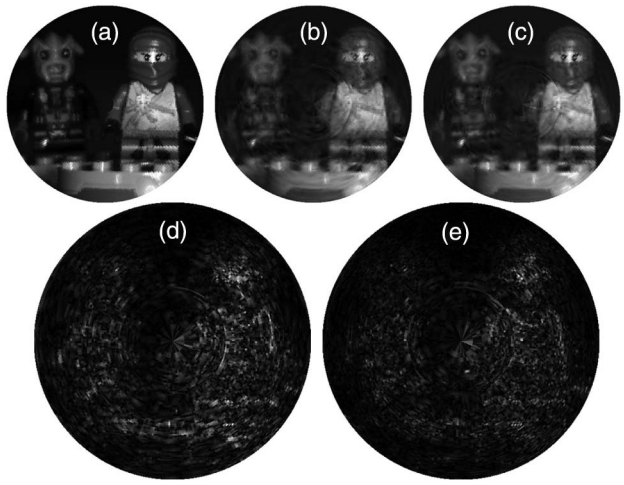

Fig. 23. (a) Original image. (b) Reconstruction using random aperture code with 27 dB PSNR. (c) Reconstructed image using the optimized code with 29 dB PSNR. (d) The absolute errors of random reconstruction. (e) The absolute errors of the optimal recovery.

To evaluate the performance of the optimized aperture code, a reconstruction comparison is shown in Fig. 23. Figure 23(b) shows the reconstruction using the initial random aperture code with a PSNR of 27 dB. The reconstruction using the optimized code achieves a 29 dB PSNR, as shown in Fig. 23(c). For a visual comparison, the absolute errors are calculated for both cases as shown in Figs. 23(d) and 23(e). Smaller errors are observed in the recovered image using the optimized aperture code.

5. CONCLUSION

The design of a coded aperture for compressive imaging through rotation has been presented. Initially, a code with relatively uniform pixel shapes was proposed in order to maximize image quality. The design then evolved through two major revisions aimed at increasing CS recovery performance. First, the code was restricted to the same number of spokes for all rings.

This deformed the pixel shapes, somewhat decreasing image quality. Next, the single sensor was replaced by a small sensor array, marginally increasing the cost and complexity of the design. In addition, the initial unrealistic polar sensor array was replaced with a more practical, although inferior performing, square array. Image quality, complexity, recovery performance, and practicality are all balanced in the final design to form a workable solution. Although mutual coherence was not helpful as a performance metric during the code geometry design, it was successfully used to optimize the block–unblock code pattern. A laboratory experiment is being performed to verify these simulation results, and work has been performed applying this rotating coded aperture to spectral imaging [26]. Further research is required to model the full system motion of a spinning projectile, but the results presented here indicate a promising future for the implementation of a rotating compressive imager.

APPENDIX A

The mapping of pixel f_{sr} to element f_{ij} of matrix $\mathbf{F}$ in the initial geometry in Fig. 2 can be divided into three steps. First, the center point of f_{sr} is calculated as

$$\varphi_{sr} = \frac{2\pi(s-1)+\pi}{S_r}, \tag{A1}$$

$$\rho_{sr} = \Delta\left(r - \frac{1}{2}\right). \tag{A2}$$

Next, $(\rho_{sr}, \varphi_{sr})$ is transformed into the corresponding point (x_{sr}, y_{sr}) in Cartesian coordinates in the square code illustrated in Fig. 2(b) by

$$y = \begin{cases} \rho\varphi(4/\pi), & 0 \le \varphi < \pi/4 \\ \rho, & \text{if } \pi/4 \le \varphi < 3\pi/4 \\ -\rho(\varphi-\pi)(4/\pi), & 3\pi/4 \le \varphi < 5\pi/4 \\ \rho, & 5\pi/4 \le \varphi < 7\pi/4 \\ \rho(\varphi-2\pi(4/\pi), & 7\pi/4 \le \varphi < 2\pi \end{cases}, \tag{A3}$$

$$x = \begin{cases} \rho, & 0 \le \varphi < \pi/4 \\ -\rho(\varphi-\pi/2)(4/\pi), & \pi/4 \le \varphi < 3\pi/4 \\ \rho, & 3\pi/4 \le \varphi < 5\pi/4 \\ \rho(\varphi-3\pi/2)(4/\pi), & 5\pi/4 \le \varphi < 7\pi/4 \\ \rho, & 7\pi/4 \le \varphi < 2\pi \end{cases}. \tag{A4}$$

Finally, (x_{sr}, y_{sr}) is changed to indices (i, j) using

$$i = x_{sr}/\Delta + R + 1/2, \tag{A5}$$

$$j = R - y_{sr}/\Delta + 1/2. \tag{A6}$$

APPENDIX B

Here we list the nomenclature used in this paper. When a symbol has more than one meaning, its use is obvious from the context. This includes differentiating a superscript ℓ denoting the snapshot number, and a superscript denoting an exponent. Some definitions suggest limitations that may not apply. For example, Δ is defined as the height of the first ring, but for codes with rings of equal heights, Δ is the height of every ring.

NOMENCLATURE

$\mathbf{A}$	System matrix $\mathbf{A} = \mathbf{H\Psi}$
$\mathbf{a}$	Column of $\mathbf{A}$
$\tilde{\mathbf{a}}$	Diagonal elements in $\mathbf{A}^T\mathbf{A}$
d	Code division factor
$f(\rho, \varphi)$	Continuous representation of image
f_s	Discrete 1D image pixel in spoke s
f_{sr}	Discrete 2D image pixel in ring r and spoke s
f_{ij}	Image pixel f_{sr} mapped to matrix element
$\mathbf{F}$	Image matrix with elements f_{ij}
$\mathbf{f}$	1D image with elements f_s or vec($\mathbf{F}$)
g	A measurement in a single sensor system
g_{mn}	A measurement of sensor (m, n)
$\mathbf{g}$	Complete measurement vector
$\mathbf{g}_{mn}$	Measurement vector associated with sensor (m, n)
$\mathbf{H}$	Complete sensing matrix
$\mathbf{H}_{mn}$	Sensing matrix associated with sensor (m, n)
K	Total number of measurements
K'	Number of sensor array snapshots
ℓ	Superscript denotes snapshot number
M	One dimension of an $M \times M$ sensor array
N	Total number of pixels in image
R	Total number of rings
S	Number of spokes in code
S_r	Number of pixel spokes in ring r
$t(\rho, \varphi)$	Continuous representation of coded aperture
t_s	Integral of 1D code $t(\rho, \varphi)$ over pixel f_s
t_{sr}	Integral of 2D code $t(\rho, \varphi)$ over pixel f_{sr}
t_{srmn}	Element (s, r) of matrix $\mathbf{T}_{mn}$
t_{ij}	t_{sr} mapped to matrix element
$\mathbf{T}$	Code matrix with elements t_{ij}
$\mathbf{T}_{mn}$	Code matrix associated with sensor (m, n)
vec($\mathbf{Z}$)	Vectorization of matrix $\mathbf{Z}$
$w_{mn}(\rho, \varphi)$	Continuous function of polar sensor (m, n) FOV
$w_{mn}(x, y)$	Continuous function of square sensor (m, n) FOV
w_{srmn}	Proportion of t_{sr} in FOV of sensor (m, n)
$\mathbf{W}_{mn}$	Matrix indexed by (s, r) with elements w_{srmn}
z, $\mathbf{Z}$	General purpose variable, matrix
Δ	Height of first ring
κ	Compressive sensing ratio, K/N
$\mathbf{\Psi}$	Sparse basis
$\Omega(z)$	Region of z

Funding. Army Research Laboratory (ARL) (W911NF-14-2-0108).

REFERENCES

1. M. F. Duarte, M. A. Davenport, D. Takhar, J. N. Laska, T. Sun, K. E. Kelly, and R. G. Baraniuk, "Single-pixel imaging via compressive sampling," IEEE Signal Process. Mag. **25**(2), 83–91 (2008).
2. R. F. Marcia and R. M. Willett, "Compressive coded aperture superresolution image reconstruction," in *IEEE International Conference on Acoustics, Speech and Signal Processing* (IEEE, 2008), pp. 833–836.
3. A. Mohan, X. Huang, J. Tumblin, and R. Raskar, "Sensing increased image resolution using aperture masks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, 2008).
4. W. L. Chan, K. Charan, D. Takhar, K. F. Kelly, R. G. Baraniuk, and D. M. Mittleman, "A single-pixel terahertz imaging system based on compressed sensing," Appl. Phys. Lett. **93**, 121105 (2008).

5. P. Llull, X. Liao, X. Yuan, J. Yang, D. Kittle, L. Carin, G. Sapiro, and D. J. Brady, "Coded aperture compressive temporal imaging," Opt. Express **21**, 10526–10545 (2013).
6. D. Cabib, M. Lavi, A. Gil, E. Ohel, J. Dolev, and U. Milman, "A long wave infrared (LWIR) spectral imager (7.7 to 12.3 μm) based on cooled detector array and high resolution circular variable filter (CVF)," Proc. SPIE **8896**, 88960R (2013).
7. D. Brady and M. Gehm, "Coding and modulation for hyperspectral imaging," U.S. patent 7,336,353 (26 February 2008).
8. E. J. Candès and M. B. Wakin, "An introduction to compressive sampling," IEEE Signal Process. Mag. **25**(2), 21–30 (2008).
9. H. Rauhut, "Compressive sensing and structured random matrices," in *Theoretical Foundations and Numerical Methods for Sparse Recovery* (de Gruyter, 2010), Vol. **9**, pp. 1–92.
10. R. M. Willett, R. F. Marcia, and J. M. Nichols, "Compressed sensing for practical optical imaging systems: a tutorial," Opt. Eng. **50**, 072601 (2011).
11. G. R. Arce, D. J. Brady, L. Carin, H. Arguello, and D. S. Kittle, "Compressive coded aperture spectral imaging: an introduction," IEEE Signal Process. Mag. **31**(1), 105–115 (2014).
12. J. Tan, Y. Ma, H. Rueda, D. Baron, and G. R. Arce, "Compressive hyperspectral imaging via approximate message passing," IEEE J. Sel. Topics Signal Process. **10**, 389–401 (2016).
13. Y. Wu, I. O. Mirza, G. R. Arce, and D. W. Prather, "Development of a digital-micromirror-device-based multishot snapshot spectral imaging system," Opt. Lett. **36**, 2692–2694 (2011).
14. G. Cooper and M. Costello, "Flight dynamic response of spinning projectiles to lateral impulsive loads," J. Dyn. Syst. Meas. Control **126**, 605–613 (2004).
15. F. Fresconi, G. Cooper, I. Celmins, J. DeSpirito, and M. Costello, "Flight mechanics of a novel guided spin-stabilized projectile concept," *AIAA Atmospheric Flight Mechanics Conference*, Reston, Virigina (2011).
16. E. J. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," IEEE Trans. Inf. Theory **52**, 489–509 (2006).
17. D. L. Donoho, "Compressed sensing," IEEE Trans. Inf. Theory **52**, 1289–1306 (2006).
18. E. Candes and J. Romberg, "Sparsity and incoherence in compressive sampling," Inverse Prob. **23**, 969–985 (2007).
19. M. H. Buonocore, W. R. Brody, A. Macovski, and S. L. Wood, "Polar pixel Kalman filter for limited data computed tomography (CT) image reconstruction," Proc. SPIE **0206**, 109–114 (1979).
20. C. Mora and M. Rafecas, "Polar pixels for high resolution small animal PET," in *IEEE Nuclear Science Symposium Conference Record* (IEEE, 2006), Vol. **11**, pp. 2812–2817.
21. V. J. Traver and A. Bernardino, "A review of log-polar imaging for visual perception in robotics," Robot. Auton. Syst. **58**, 378–398 (2010).
22. E. Candes and J. Romberg, "ℓ_1-magic: recovery of sparse signals via convex programming" (2005), http://statweb.stanford.edu/~candes/l1magic/downloads/l1magic.pdf.
23. W. Yin, S. Morgan, J. Yang, and Y. Zhang, "Practical compressive sensing with Toeplitz and circulant matrices," Proc. SPIE **7744**, 77440K (2010).
24. M. Analoui and J. P. Allebach, "Model-based halftoning using direct binary search," Proc. SPIE **1666**, 96–108 (1992).
25. R. D. Nowak and S. J. Wright, "Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems," IEEE J. Sel. Top. Signal Process. **1**, 586–597 (2007).
26. C. Fu, M. Don, and G. Arce, "Compressive spectral imaging via polar coded aperture," IEEE Trans. Comput. Imag. (to be published).
27. H. Arguello and G. R. Arce, "Colored coded aperture design by concentration of measure in compressive spectral imaging," IEEE Trans. Image Process. **23**, 1896–1908 (2014).
28. D. L. Lau, R. Ulichney, and G. R. Arce, "Blue and green noise halftoning models," IEEE Signal Process. Mag. **20**(4), 28–38 (2003).
29. D. L. Lau, G. R. Arce, and N. C. Gallagher, "Digital color halftoning with generalized error diffusion and multichannel green-noise masks," IEEE Trans. Image Process. **9**, 923–935 (2000).
30. J. B. Rodriguez, G. R. Arce, and D. L. Lau, "Blue-noise multitone dithering," IEEE Trans. Image Process. **17**, 1368–1382 (2008).
31. D. L. Lau and G. R. Arce, *Modern Digital Halftoning* (CRC Press, 2001).
32. C. Fu, H. Arguello, B. M. Sadler, and G. R. Arce, "Compressive spectral polarization imaging by a pixelized polarizer and colored patterned detector," J. Opt. Soc. Am. A **32**, 2178–2188 (2015).
33. A. P. Cuadros, C. Peitsch, H. Arguello, and G. R. Arce, "Coded aperture optimization for compressive x-ray tomosynthesis," Opt. Express **23**, 32788–32802 (2015).
34. M. Elad, "Optimized projections for compressed sensing," IEEE Trans. Signal Process. **55**, 5695–5702 (2007).
35. J. M. Duarte-Carvajalino and G. Sapiro, "Learning to sense sparse signals: Simultaneous sensing matrix and sparsifying dictionary optimization," DTIC document (2008).
36. J. Xu, Y. Pi, and Z. Cao, "Optimized projection matrix for compressive sensing," EURASIP J. Adv. Signal Process. **2010**, 560349 (2010).
37. M. A. Seldowitz, J. P. Allebach, and D. W. Sweeney, "Synthesis of digital holograms by direct binary search," Appl. Opt. **26**, 2788–2798 (1987).