Image Compression Based on Compressive Sensing: End-to-End Comparison with JPEG

Xin Yuan, *Senior Member, IEEE* and Raziel Haimi-Cohen, *Senior Member, IEEE*

***Abstract*—We present an end-to-end image compression system based on compressive sensing. The presented system integrates the conventional scheme of compressive sampling and reconstruction with quantization and entropy coding. The compression performance, in terms of decoded image quality versus data rate, is shown to be comparable with JPEG and significantly better at the low rate range. We study the parameters that influence the system performance, including (i) the choice of sensing matrix, (ii) the trade-off between quantization and compression ratio, and (iii) the reconstruction algorithms. We propose an effective method to jointly control the quantization step and compression ratio in order to achieve near optimal quality at any given bit rate. Furthermore, our proposed image compression system can be directly used in the compressive sensing camera, *e.g.* the single pixel camera, to construct a *hardware compressive sampling* system.**

***Index Terms*—Compressive sensing, image compression, quantization, entropy coding, sparse coding, reconstruction, JPEG, JPEG2000.**

I. Introduction

Compressive Sensing (CS) [2], [3], [4] has been proposed more than a decade ago as a method for dimensionality reduction of signals which are known to be sparse or compressible in a specific basis representation. By "sparse" we mean that only a relatively small number of the coefficients of the representation are non-zero, whereas "compressible" indicates that the magnitude of the coefficients decays quickly, according to a power law, hence the signal can be well approximated by a sparse signal. In the CS paradigm, the signal is projected onto a low-dimension space, resulting in a *measurements vector*. If the signal is sparse, it is possible to exactly reconstruct it from the measurements vector. If the measurements are noisy or the signal is not sparse but compressible the reconstruction yields an approximation to the original signal. Natural images are inherently compressible in the frequency or wavelet domain and therefore suitable for CS. The past five years saw an impressive progress in this field with new reconstruction algorithms [5], [6], [7] achieving better reconstructed image quality at a lower compression ratio (the ratio of the dimension of the measurements vector to the number of pixels in the original image). These algorithms go beyond sparsity and leverage other properties of natural images, such as having low rank [6], or being being capable of denoising [7].

Encouraged by these achievements, we set out to create an end-to-end image compression system based on CS. In itself CS is *not* a complete signal compression system because its "compressed signal", the measurements vector, is an array of real numbers rather than a sequence of bits or bytes. Thus, in order to build a complete system we added a source coding stage, in which the measurements are quantized, and a channel coding stage, which the quantized measurements into a byte sequence, using entropy coding.

The authors are with Nokia Bell Labs, 600 Mountain Avenue, Murray Hill, NJ, 07974, USA, xyuan@bell-labs.com. The MATLAB code used to generate the results in this paper can be downloaded at CS vs. JPEG Demo and more results are available at [1].

A. Related Work

Goyal *et al.* [8] applied an information-theoretic approach to assess the effectiveness of a CS based compression system for sparse signals $\boldsymbol{x} \in \mathbb{R}^N$, with only K non-zero entries, a.k.a. K-sparse. Their benchmark was the "baseline" method, where the coded bit sequence consisted of the sparsity pattern (*i.e.* a specification of the indices of the non-zero entries in $\boldsymbol{x}$) and the quantized and coded non-zero entries. They showed that the rate-distortion functions of a CS-based compression system are considerably worse than those of the baseline method for two reasons. First, the number of measurement M required by CS to recover $\boldsymbol{x}$ is several times larger than K, $M/K \geq \log(N/K)$, and the number of bits needed to represent the additional $M-K$ quantized variables exceeds the number of bits needed to specify the sparsity pattern, especially when the number of bits per measurement is high. Second, the quantization noise in the baseline method is proportional to K, whereas in CS with a random sensing matrix, it is proportional to M. Goyal *et al.* suggest that the use of distributed lossless coding and entropy-coded dithered quantization might potentially alleviate those problems, but the complexity added by those methods would probably make them impractical.

Despite this pessimistic outlook, the research of the effect of quantization on CS measurements got significant attention in the last few years. Laska and Baraniuk studied the trade-off between the number of measurements and quantization accuracy [9] and showed that the sensitivity of the reconstruction accuracy to the quantization noise varies dramatically with the compression ratio. Laska *et al.* showed that even a small number of saturated measurements, *i.e.* measurements whose quantization error is not bounded, may cause a considerable degradation on reconstruction accuracy [10]. Methods for accounting for the quantization effect in the reconstruction algorithm were studied in [11]. The extreme case of 1-bit quantizer was investigated in [9], [12], [11]. The asymptotic normality and approximate independence of measurements generated by various sensing matrices were shown in [13]. Dai *et al.* compared various quantizer designs and studied their effect on reconstruction accuracy [14], [15]. The distribution

of measurements generated by video and image CS systems, which included quantization, were also described in [16], [17]. However this significant body of research was of limited value for our purposes. First, these works assume a random, or a structurally random sensing matrix, while the sensing matrices suitable for our purposes could be different (see Sec. II-A). Second, most of these works did not assume any channel coder and therefore did not study the resulting bit rates. Those that considered a channel coder did not study the interaction between the quantizer and the channel coder and the trade-offs in their design.

B. Image Compression via JPEG and JPEG2000

Since its introduction in 1992, JPEG [18] has been one of the most popular image compression methods in use. JPEG2000 [19], which was released ten years later, has superior compression properties at a wider bit rate/quality range. A brief overview of these standards is given in Sec. II-F, where their architectures are compared with that of the proposed compressive sensing based image compression system (CSbIC).

At the time of its introduction, the higher complexity of JPEG2000 could be an impediment for its adoption, but as computing capabilities improved, this was not longer an issue for most cases. Nevertheless, JPEG remains the image compression tool of choice in a wide range of applications. It appears that the advantages of JPEG2000 were not necessary for many targeted applications, which raises the question if there is any need for yet another image compression scheme. However, a compression scheme based on CS should have some unique properties which are radically different from those of any conventional signal compression scheme, which may justify its adoption.

C. Uniqueness of CS based Image Compression

A compression scheme based on CS enjoys the following unique properties which are primarily different from those of any conventional signal compression scheme.

- The encoder is of much lower complexity than the decoder. In fact, a significant part of the encoder processing, namely the measurements generation, can be done in the analog domain, *e.g.* by a single-pixel camera or a lensless camera [20], [21], [22], [23], [24].
- The decoder is not fully defined by the encoder. Different reconstruction algorithms can be applied to the same measurements vector at different situations, and as more prior information becomes available the reconstruction algorithm can be improved to take it into account [25], [6], [26], [27], [5], [7], [24], [28].
- The loss of a small fraction of the measurements generally results in only a minor degradation in reconstruction quality. This may be used to achieve robustness to channel impairments without the usual overhead of error protection.

These unique properties suggest that CSbIC would be very useful in applications such as media sensor networks, where the sensors need to be inexpensive and have low energy consumption, the network operates at low power and is usually unreliable, and much prior information may be gathered while the system is running. Therefore, CSbIC may be useful in a class of applications which are quite different from those that JPEG and JPEG2000 were designed for.

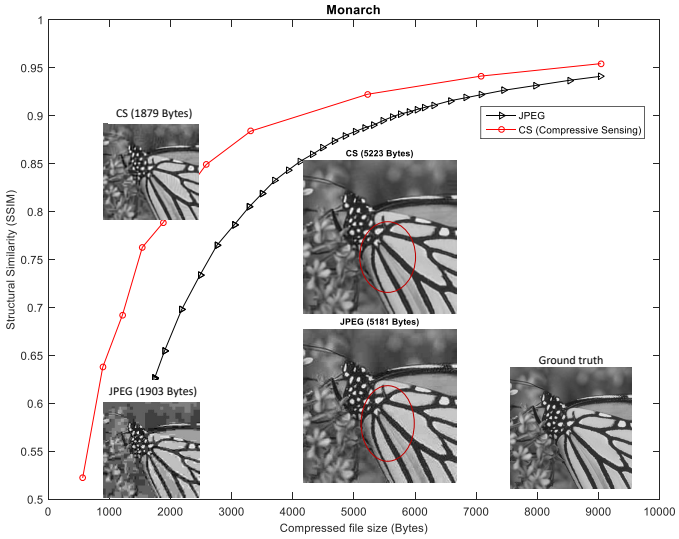

Fig. 1. Comparison of JPEG and CS-based compression of the "Monarch" image. Curves of structural similarity (SSIM) vs. compressed file size, as well as decoded images of similar file size are shown. Please zoom-in on red circles for details.

D. Contributions of This Work

This paper makes the following contributions:

i) We provide an end-to-end architecture for a CSbIC system, including compressive sensing, source coding, channel coding and a mechanism to adjust the system parameters to control data rate and reconstruction quality.

ii) We address the theoretical issues raised by Goyal *et al.* [8] in a practical way, by using *domain-specific* knowledge: (i) we employ reconstruction algorithms that do not rely on sparsity but on other properties of natural images, and (ii) we use deterministic sensing matrices which are known to be effective for this such signals.

iii) Having an end-to-end system enables us to measure its performance in terms of quality versus data rate, and thus benchmarks it against the leading image compression standards, namely JPEG and JPEG2000. We show that our CSbIC system is on-par with JPEG, with a clear advantage in the low data rate range. Please refer to Fig. 1 as an example.

iv) We describe various design choices in each of the system components and study the effect that these choices have on the overall system performance.

The system that we describe is far from being fully optimized, and throughout the paper we point out where further improvement may be made. Nevertheless, even at this preliminary stage the performance of our system makes it a viable al-

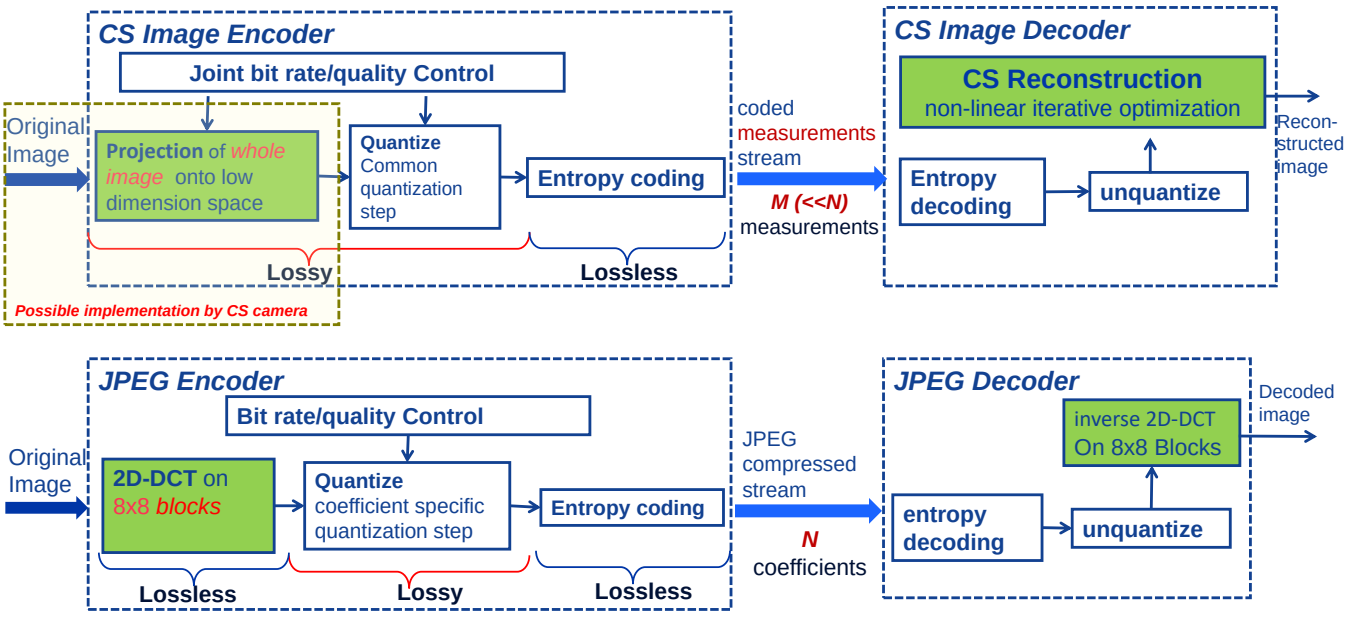

Fig. 2. Image compression architecture comparison between proposed CSbIC (top) and JPEG (bottom).

ternative to the conventional methods (refer to Figs 4-6 for comparison with JPEG).

The rest of this paper is organized as follows. Sec. II describes the system architecture and discusses the design choices in each component. Sec. III provides the general framework of reconstruction algorithms. Sec. IV presents the results of our performance testing and Sec. V discusses the implications of our results.

II. System Architecture

A diagram of the system architecture is given in Fig. 2. Each of the encoding steps is matched by a corresponding decoding step (in reverse order), with the exception of the bit rate/quality control block, which appears only in the encoder. In the following we present a detailed description of each of those processing steps.

A. Sensing Matrix and Measurements Generation

We consider monochromatic images of $N_v \times N_h$ pixels. The pixels of the input image are organized column by column as a pixel vector $\boldsymbol{x} \in \mathbb{R}^N$ ($N = N_v N_h$). The pixel vector $\boldsymbol{x}$ is multiplied by a sensing matrix $\boldsymbol{\Phi} \in \mathbb{R}^{M \times N}$, $M \ll N$, yielding the measurements vector

$$\boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{x}. \tag{1}$$

$\boldsymbol{\Phi}$ is quite a large matrix, even for small images. Therefore, for practical reasons, $\boldsymbol{\Phi}$ is never stored, and the operation (1) is implemented as a fast transform. It is well known that a 2-dimensional discrete cosine transform (2D-DCT) is very effective in decorrelating an image, and most of the energy of the image is concentrated in the low frequency transform coefficients. Recently, new sensing matrices were introduced which leverage this property [22], [29]. These matrices are *not* incoherent with the common sparsity bases of natural images, as classical CS theory would require for guaranteeing robust reconstruction [2], [4], but generally they perform better than the classical sensing matrices in our application, *i.e.* image compression. For example, (1) can be implemented by performing 2D-DCT on the image pixels and then reordering the resulting 2D-DCT coefficients in a "zig-zag" order (similar to the one used in JPEG encoding) and selecting the first M low-frequency coefficients [22].

In some applications, such as the single-pixel camera and lensless camera [20], [21], [22], a binary-valued matrix, *e.g.* a matrix whose entries are only ± 1 (or $\{0, 1\}$) is more suitable for hardware implementation. In this case we approximate the 2D frequency decomposition by using a 2D Walsh-Hadamard transform (2D-WHT) [30]. Let

$$\mathbf{W} = \mathbf{W}_h \otimes \mathbf{W}_v, \tag{2}$$

where $\otimes$ denotes the Kronecker product [31] and $\mathbf{W}_h, \mathbf{W}_v$ are Walsh-Hadamard matrices in sequency order [32], that is, the k^{th} row of $\mathbf{W}_h, \mathbf{W}_v$, has $k-1$ zero crossings (if N_v or N_h is not a power of two, the image is padded by zero pixels). Similar to the 2D-DCT case, the selected measurements are the first M coefficients of the zig-zag ordered entries of $\mathbf{W}\boldsymbol{x}$ [29]. Note that $\mathbf{W}\boldsymbol{x}$ can also be computed numerically in an efficient way, using the Fast Walsh-Hadamard Transform.

We can get the CS theoretical guarantee for successful reconstruction, w.h.p. (with high probability), by replacing the deterministic matrices described above with random matrices, whose entries are independent, identically distributed (IID) random variables (RVs), with Gaussian or Rademacher distributions [33], [34]. These matrices are universal, *i.e.* w.h.p. they are incoherent with any given sparsity basis. Further-

more, the measurements generated by those random matrices are mutually independent and asymptotically normally distributed, which is helpful in the quantization and coding design. Such fully random matrices do not allow fast transform implementation of (1), but similar desired properties and performance guarantees were shown for structurally random matrices (SRM) [13], [35], [36], where $\mathbf{\Phi}\boldsymbol{x}$ is obtained by applying a randomly selected permutation to $\boldsymbol{x}$, computing a fast transform, such as DCT or WHT, on the permuted vector, and randomly selecting M of the transform coefficients as measurements (the DC coefficient is always selected). We denote these matrices SRM-DCT and SRM-WHT, respectively.

B. Quantization

The quantizer maps the measurements vector $\boldsymbol{y}$ into a finite sequence of codewords taken from a finite codebook $\mathcal{C}$ and the dequantizer maps the codewords sequence into a measurements vector which is an approximation of the original measurements vector. If the sensing matrix is deterministic the measurements are highly uncorrelated; if it is a SRM the measurements are nearly independent. Hence the advantage of vector quantization [37] over scalar quantization is small and does not justify its added complexity [14]. Therefore, we consider a scalar quantizer which maps each measurement $\{y_i\}_{i=1}^{M}$ to a codeword $q_i = Q_i(y_i) \in \mathcal{C}$, where $Q_i : \mathcal{R} \to \mathcal{C}$ is the quantizer of the i^{th} measurement. In this work we use the same quantizer for all measurements, hence in the following we omit the subscript i from Q_i.

The simplest scalar quantizer is the uniform quantizer. We select the "mid-tread" type, defined by

$$Q(y) \stackrel{\text{def}}{=} \max(-L, \min(L, \tilde{Q}(y))), \tag{3}$$

$$\tilde{Q}(y) \stackrel{\text{def}}{=} \lfloor (y-\mu)/s + 0.5 \rfloor, \tag{4}$$

where $\lfloor y \rfloor$ denotes the largest integer not exceeding y, $\mu = \frac{1}{M}\sum_{i=1}^{M} y_i$ is the mean of the measurements, s is the quantizer's step, $\tilde{Q}(y)$ is the unclipped quantized value, and L is a positive integer which determines the range $s(L-0.5)$ of the actual quantizer $Q(y)$. Consequently there are $2L+1$ codewords, $\mathcal{C} = \{-L, \cdots, L\}$. Since the distribution of the measurements is highly non-uniform, the codewords distribution is also not uniform, hence in order to represent codewords effectively by a bit sequence we need to use variable length coding (VLC) in the channel coder. On the other hand, an optimal quantizer (in the mean square sense) or an entropy constrained quantizer [37] usually results in nearly equally populated quantization regions, which makes it possible to use fixed length coding (FLC) with little data rate penalty. Thus the design choice is between a simple quantizer with a sophisticated VLC, versus a sophisticated quantizer with a simple FLC. We opted for the first option because designing an optimal quantizer requires knowledge of the measurements distribution, which is difficult to estimate for a deterministic sensing matrices. Another reason to use a uniform quantizer is that the reconstruction may be sensitive to the presence of even few measurements with large errors [10], which is often the case with non-uniform quantizers.

If $Q(y) = c$ and $|c| < L$, we define the dequantizer by

$$Q^{-1}(c) = cs + \mu, \tag{5}$$

hence the quantization error is bounded by

$$|y - Q^{-1}(Q(y))| \le 0.5s. \tag{6}$$

On the other hand, if $|c| = L$, the quantized measurement is *saturated* and the quantization error cannot be bounded. Even a small number of saturated measurements can cause severe quality degradation unless they are specially handled in the reconstruction [10]. The simplest way to do it is by not using the saturated measurements at all; attempts to modify the reconstruction algorithm to use those measurements showed little gain over simply discarding them. Another option (not considered in [10]) is to code the value of $\tilde{Q}(y)$ for each saturated measurement y in some *ad hoc* method and transmit is as additional information. In both cases saturated measurements incur a penalty, either in the form of transmitted codewords which are not used, or as *ad hoc* transmission of $\tilde{Q}(y)$. Therefore, we select L large enough to make saturation a rare event. In fact, L can be set sufficiently large to eliminate saturation completely, but a very large codebook may have an adverse effect on channel coding (see Sec. II-D3). We found that a good trade-off is to select L so that quantizer's range $s(L-0.5)$ is about 4 standard deviations of the measurements. However, the system is not very sensitive to this parameter — performance does not change much if the range is 3 or 6 standard deviations. We also compared ignoring saturated measurements to sending $\tilde{Q}(y)$ for each of them. We chose the latter because it performed slightly better and had the important advantage of not requiring any change in the reconstruction algorithm.

With all the sensing matrices considered, the first measurement y_0 is the DC coefficient, which is the sum of all pixels in the image. Since the pixels are unsigned, y_0 is much larger than the other measurements and is always saturated, therefore it requires special handling: y_0 is excluded when calculating the mean (μ), and the standard deviation of the measurements, and $Q(y_0) = L$ is not included in the quantized measurement. Instead, $\tilde{Q}(y_0)$ is coded in an *ad hoc* fashion and transmitted separately.

Unless the quantization is very coarse, its effect on the measurements can be modeled as adding white noise, uniformly distributed in $[-s/2, s/2]$, which is uncorrelated with the measurements [37]. Hence the variance of the quantization noise in each measurement is

$$\sigma_Q^2 = s^2/12. \tag{7}$$

The integral pixel values are generally obtained by sampling an analog signal and rounding the samples values to the nearest integer. Hence the pixel values contain digitization noise with variance of $1/12$. This noise appears in the measurement y_j with variance

$$\sigma_D^2 = \|\boldsymbol{\phi}_j\|_2^2/12, \tag{8}$$

where $\boldsymbol{\phi}_j$ is the j^{th} row of $\mathbf{\Phi}$. In the sensing matrices that we consider $\|\boldsymbol{\phi}_j\|_2$ is constant, $\|\boldsymbol{\phi}_j\|_2 = \|\mathbf{\Phi}\|_2$. Clearly, there is no point in spending bits to accurately represent the

digitization noise, hence we need to have $\sigma_Q \geq \sigma_D$ and consequently $s \geq \|\mathbf{\Phi}\|_2$. Typical quantizer step sizes are between $\|\mathbf{\Phi}\|_2$ to $50\|\mathbf{\Phi}\|_2$.

C. Image Quality Control

The compression ratio R and the quantizer step size s control the coded image size and the quality of the reconstructed image. One can get to the same reconstruction quality with various combinations of these parameters but the coded image size varies greatly. Our experiments (Section IV-B) showed that at any given quality the lowest bit rate is achieved when

$$Rs = C\|\mathbf{\Phi}\|_2, \tag{9}$$

where C is a constant. The optimal value of C depends on the type of sensing matrix and varies from image to image. However, we found that using $C = 2.0$ is a good general value for all pictures. Thus, in our tests s is determined by R using (9), with $C = 2.0$. This quantization step is sufficiently fine to allow modeling the quantization noise as uncorrelated, uniformly distributed white noise.

D. Lossless Coding

The lossless encoder encodes the codeword sequence generated by the quantizer, as well as some miscellaneous information (*e.g.* μ, s, and the *ad hoc* representation of saturated measurements), as a bit sequence. The lossless decoder exactly decodes the codeword sequence and the miscellaneous information from the bit sequence.

1) Coded Numbers Format : Various types of numbers are coded by the lossless encoder (and decoded by the lossless decoder). Each type is encoded in a different way:

Unbounded signed or unsigned integers are integers whose maximal possible magnitude is not known in advance. They are represented by byte sequences bytes, where the most significant bit (MSB) In each byte is a continuation bit — it is clear in the last byte of the sequence and set in all other bytes. The rest of the bits are payload bits which represent the integer. The number of bytes is the minimal number that has enough payload bits to fully represent the unsigned or unsigned integer.

Real numbers, which are natively stored in single or double precision floating point format [38] are coded as pairs of unbounded signed integers representing the mantissas and the exponents in the floating point format.

Bit arrays are zero padded to a length which is a multiple of 8 and coded as a sequence of bytes, 8 bits per bytes.

Bounded unsigned integer arrays are arrays of unsigned integers, each of which may be represented by a fixed number of bits. An array of n integers, each of which can be represented by b bits, is encoded as a bit array of bn bits.

Each of these number formats can be easily decoded. Note that these formats are byte aligned for simpler implementation.

2) Entropy Coding : The codewords $\pm L$ represent saturated measurements. Whether those measurements are ignored or transmitted separately, there is no distinction between saturation from above or below, hence we merge these two labels into a single label L, thus we have $2L$ codewords: $-L+1, \ldots, L$.

Let $p_c, c \in \mathcal{C}$ be the probability of a measurement to be quantized to c. If the codewords $\{Q(y_i)\}_{i=1}^{M}$ are IID random variables, then a tight lower bound on the expected number of bits required to transmit these codewords is the entropy rate:

$$H \stackrel{\text{def}}{=} -M \sum_{c\in\mathcal{C}} p_c \log_2 p_c. \tag{10}$$

Arithmetic coding (AC) [38] represents the codeword sequence by a bit sequence, the length of which can get arbitrarily close to the entropy rate for a large M. We use AC to encode the codewords sequence. The AC bit sequence is coded as a bit array.

Since the probabilities $p_c, c \in \mathcal{C}$ are not known *a priori*, they need to be estimated and sent to the receiver, in addition to the AC bit sequence. These probability estimates can be obtained in two ways: For SRMs, the measurements are approximately normally distributed [13], hence for $|c| < L$, p_c is the normal probabilities of the quantization intervals $[cs + \mu - 0.5s, cs + \mu + 0.5s)$, and $p_L = 1 - \sum_{|c|<L} p_c$. Thus, all that needs to be sent to the receiver is the estimated standard deviation of the measurements, which is coded as a real number. For deterministic sensing matrices, it is necessary to compute a histogram of the quantized measurements sequence, use it to determine the probabilities and then code the histogram and send it to the receiver along with the AC bit sequence. Sending the histogram is an overhead, but it is small in comparison to the gain achieved with arithmetic coding. In fact, even with measurements generated by SRM, in many cases the total bit rate achieved when using AC with a histogram is better than the total bit rate when using the normal distribution assumption, because the gain obtained by accurately describing the actual codeword frequencies is more than the overhead required of transmitting the histogram.

In natural images the magnitudes of the coefficients of 2D-DCT or 2D-WHT decay quickly, hence measurements generated using a deterministic sensing matrix are not identically distributed, which violates the assumptions under which AC is asymptotically optimal. In order to handle this problem we partition the codeword sequence into sections, and for each section we compute a histogram and an AC sequence separately. This, of course, makes the overhead of coding the histograms significant. In the following we describe how the histograms are coded efficiently and how to select a locally optimal partition of the codewords sequence.

3) Coding of Histograms : In order to be efficient, the code of histograms of short codeword sequences should be short as well. Fortunately, such histograms often have many zero counts, which can be used for efficient coding. A histogram is coded in one of three ways:

Full histogram: A sequence of $2L$ unbounded unsigned integers, containing the counts for each codeword. This method is effective when most counts are non-zero.

Flagged histogram: A bit array of $2L$ bits indicates for each codeword whether the corresponding count is non-zero, and a sequence of unbounded unsigned integers contains the non-zero counts. This method is effective when a significant

share of the counts is zero.

Indexed histogram: A bounded integer indicates the number of non-zero counts, an array of bounded integers contains the indices of the non-zero counts, and a sequence of unbounded unsigned integers contains the non-zero counts. This method is effective when most of the counts are zero. In the extreme case of a single non-zero count the AC bit sequence is of zero length, hence this histogram coding is effectively a run length encoding (RLE).

The histogram is coded in these three ways and the shortest code is transmitted. A 2-bit histogram format selector (HFS) indicates which representation was chosen. The HFSs of all sections of the codeword sequence are coded as a bit array. Thus, each section of the codeword sequence is represented by the HFS, the selected histogram representation and the AC bit sequence.

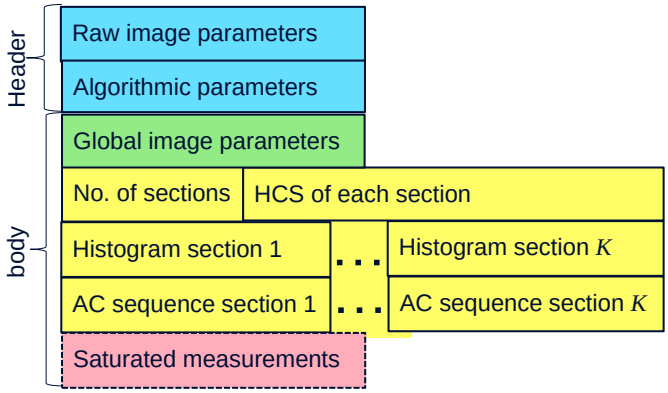

Fig. 3. Coded image structure.

4) Partitioning into AC sections: Partitioning the codeword sequence into AC sections requires estimating the number of bits in each coded section. Let $\{h_s(c), c \in \mathcal{C}\}$ be the histogram of section S, where $h_s(c)$ is the count for codeword c. In order to avoid repeated AC encoding computation, the number of bits in the AC sequence of S is estimated, based on (10), by

$$\hat{\mathcal{L}}_{\mathrm{AC}}(s) \stackrel{\mathrm{def}}{=} 8\left\lceil \sum_{c\in\mathcal{C}} h_s(c)\log_2(M_s/h_s(c))/8 \right\rceil, \qquad (12)$$

where $\lceil a \rceil$ is the least integer not smaller than a, and

$$M_s = \sum_{c\in\mathcal{C}} h_s(c) \qquad (13)$$

is the number of the codewords in the section. The total section code length is estimated by

$$\hat{\mathcal{L}}(s) = \hat{\mathcal{L}}_{\mathrm{AC}}(s) + \mathcal{L}_{\mathrm{H}}(s) + 2, \qquad (14)$$

where $\mathcal{L}_H(s)$ is the number of bits used for the histogram coding and the last term on the right hand side is the bits used for the HFS.

The partition of the codeword sequence into sections is preformed using a greedy algorithm (Algorithm 1). We begin by partitioning the sequence into RLE sections, and in each iteration we merge up to m consecutive sections, selected so that the merging yields the greatest possible reduction in total coded sections length. m is a constant which was set to 4 in our experiments. When the algorithm starts all the bits are spent on HFSs and histogram representation, and none on AC bit sequences. As the algorithm progresses and sections are merged, more bits are spent on AC, and the histograms become fewer in number, but aving more non-zero counts. It is plausible that we could get even better compression if we used more efficient ways for histogram representation, *e.g.* by using more parametric models of approximated histograms (in addition to the normal distribution).

Algorithm 1 partitioning the codeword sequence into entropy coded sections.

1. Initialization:
   a. Partition the codewords sequence $\{Q(y_j)\}_{j=1}^{M}$ into sections $\{S_k\}_{k=1}^{K}$ such that each section contains only one codeword (RLE).
   b. Compute the histograms' counts $h_k(c) \stackrel{\mathrm{def}}{=} h_{S_k}(c)$, $c \in \mathcal{C}$, $k = 1 \ldots K$.
   c. Compute the coded sections' lengths using (14): $l_k \stackrel{\mathrm{def}}{=} \hat{\mathcal{L}}(S_k)$, $\forall k = 1, \ldots, K$.
2. For a fixed $m > 0$ Let

$$\mathcal{P} = \{(k,j) | 1 \le k < j \le \min(K, k+m-1)\} \qquad (11)$$

and for each pair $(k,j) \in \mathcal{P}$, let $S_{k,j}$ be the section obtained by merging sections $S_k, \ldots, S_j$.
   a. Compute the histograms counts of $S_{k,j}$: $h_{k,j}(c) = \sum_{r=k}^{j} h_r(c)$, $c \in \mathcal{C}$.
   b. Using the histogram and (14), compute the gain of merging $S_k, \ldots, S_j$: $g_{k,j} = \sum_{r=k}^{j} l_r - \hat{\mathcal{L}}(S_{k,j})$.
3. If $\forall (k,j) \in \mathcal{P} : g_{k,j} < 0$, exit.
4. Update the partition:
   a. Let $(k^*, j^*) = \arg\max_{(k,j)\in\mathcal{P}} g_{k,j}$.
   b. Merge sections $S_{k^*}, \ldots, S_{j^*}$ into one section, with histogram and code length of $\{h_{k^*,j^*}(c),\ c \in \mathcal{C}\}$ and $\sum_{r=k^*}^{j^*} l_r - g_{k^*,j^*}$, respectively.
   c. Go to step 2.

5) Coded Image Structure: The data structure of the coded image is shown in Fig. 3. The coded image begins with a header specifying the image parameters (size, bits per pixel, color scheme, *etc.*) and the encoder's algorithmic choices (*e.g.* sensing matrix type, compression ratio, quantizer parameters). The body of the code consists of some global image parameters (μ, s, L, $\tilde{Q}(y_0)$, *etc.*) followed by the entropy coded quantized measurements, which are the bulk of the transmitted data: the HFSs, the histograms and the AC sequences for each section. If the values of $\tilde{Q}(y)$ are transmitted for saturated measurements (corresponding to terms equaling L in the codeword sequence), these are coded as an array of unbounded signed integers.

E. Decoding

Decoding is done in reverse order of encoding, as follows:

- The numerical parameters and bit arrays in the coded image are parsed.

- The quantization codewords are recovered by arithmetic decoding.
- The unsaturated quantized measurements are computed using (5). For the saturated measurements (those having a codeword of L), if values of $\tilde{Q}(y)$ are transmitted, they are used. Otherwise, the values of the quantized saturated measurements are set to zero.
- The sensing matrix is determined according to the algorithmic choices in the coded image. If there was no *ad hoc* transmission of $\tilde{Q}(y)$ for saturated measurements, the rows corresponding to these measurements are set to zero. In practice this is done by replacing the original sensing matrix $\mathbf{\Phi}$ by $D\mathbf{\Phi}$, where D is a $M \times M$ diagonal matrix whose diagonal elements are zero for saturated measurements and one for unsaturated measurements.
- The image is reconstructed using the sensing matrix and the quantized measurements, as described in detail in Sec. III.

F. Module Comparison of CSbIC with JPEG and JPEG2000

We now compare the architectures of CSbIC, JPEG and JPEG2000 and consider the aspects which may lead to performance differences. Fig. 2 compares the architecture of CSbIC with that of JPEG side by side. The main common points and differences are:

- JPEG, JPEG2000 and CSbIC begin with a linear projection of the image onto a different space. However:
  - JPEG2000 may partition the image into tiles of varying sizes, which are processed separately. This is equivalent to using a block-diagonal projection matrix, where each block corresponds to a tile. In JPEG the tiles (referred to as blocks) are of fixed size of 8×8 pixels. In CSbIC the projection is done on the whole image, which was adequate for the image sizes which we experimented with. For larger images, adding tiling should be straight forward.
  - In both JPEG and JPEG2000, each tile/block is projected on a space of the same dimension, N, hence there is no data loss in this operation, whereas in CSbIC the projection is lossy since it is on a M-dimensional space, $M \ll N$.
  - In JPEG the projection is a 2D-DCT with the output organized in zig-zag order, while in JPEG2000 it is a 2D-wavelet transform. In CSbIC, a 2D-DCT based projection is one of several options.
  - JPEG uses block to block prediction of the DC coefficient in order to deal with the issue of the DC coefficient being much larger than the other coefficients. In JPEG2000 this is done by subtracting a fixed value from all pixels before the projection. In contrast, CSbIC takes care of this issue in the quantization stage. The effect of these different methods is similar and has little impact on performance.
- CSbIC uses a simple uniform scalar quantizer with an identical step size for all measurements. JPEG uses the same type of quantizer, but the step size is selected from a quantization table and is different for each coefficient, resulting in quantization noise shaping, which may give JPEG an advantage at higher quality/data rate (Fig. 4). JPEG2000 also performs quantization noise shaping through varying step size, and in addition, its quantizer is not exactly uniform—the quantization interval around zero (the "dead-zone") is larger than the other quantization intervals, effectively forcing small wavelet coefficients to zero and reducing the amount of bits spent on coding them. It is plausible that using noise-shaping and slight non-uniformities in the quantization would improve the performance of CSbIC as well.
- The bit rate and quality trade-off in JPEG and JPEG2000 is controlled by tuning the operation of a single module — the quantizer. In contrast, the CSbIC this trade-off is controlled by jointly tuning two different modules: The projection, or measurement capturing module is tuned by changing the compression ratio, and the quantizer is tuned by changing the quantization step.
- In JPEG, entropy coding is based on Huffman coding and RLE. JPEG2000 uses arithmetic coding, with a sophisticated adaptive algorithm to determine the associated probabilities. CSbIC uses arithmetic coding, which is known to be better than Huffman coding, but instead of using adaptive estimation of the probabilities, the codewords are partitioned into sections and for each section a histograms of codewords is computed and sent as side information. The overhead of the transmitted histograms may be a disadvantage of CSbIC relative to JPEG2000.
- In JPEG and JPEG2000, the decoder generates the image from the dequantized transform coefficients by an inverse 2D-DCT or wavelet transform, respectively — a simple linear operation which does not rely on any prior information not included in the received data. In contrast, the CS reconstruction in CSbIC is an iterative, non-linear optimization, which relies on prior assumptions about the structure of the image (*e.g.* sparsity).

III. Image Reconstruction

Since the seminal works of Candès *et al.* [2] and Donoho [3], various reconstruction algorithms have been developed [26], [39], [40], [5], [41], [42]. The early reconstruction algorithms leveraged the property of natural images of being compressible when projected by a suitable sparsity operator $\mathbf{D}$:

$$\boldsymbol{f} = \mathbf{D}\boldsymbol{x}, \tag{15}$$

where $\boldsymbol{f}$ denotes the projected vector and it is usually forced to be sparse. $\mathbf{D}$ can be a pre-defined basis (DCT or wavelet), or learned on the fly [43]. Another popular sparsity operator is the Total Variation (TV), where $\mathbf{D}$ is a projection on a higher dimension space [5], [25].

Recently, better results were obtained by algorithms such as D-AMP [7] and NLR-CS [6], which exploit established image denoising methods or the natural images property of having a low rank on small patches. While using different

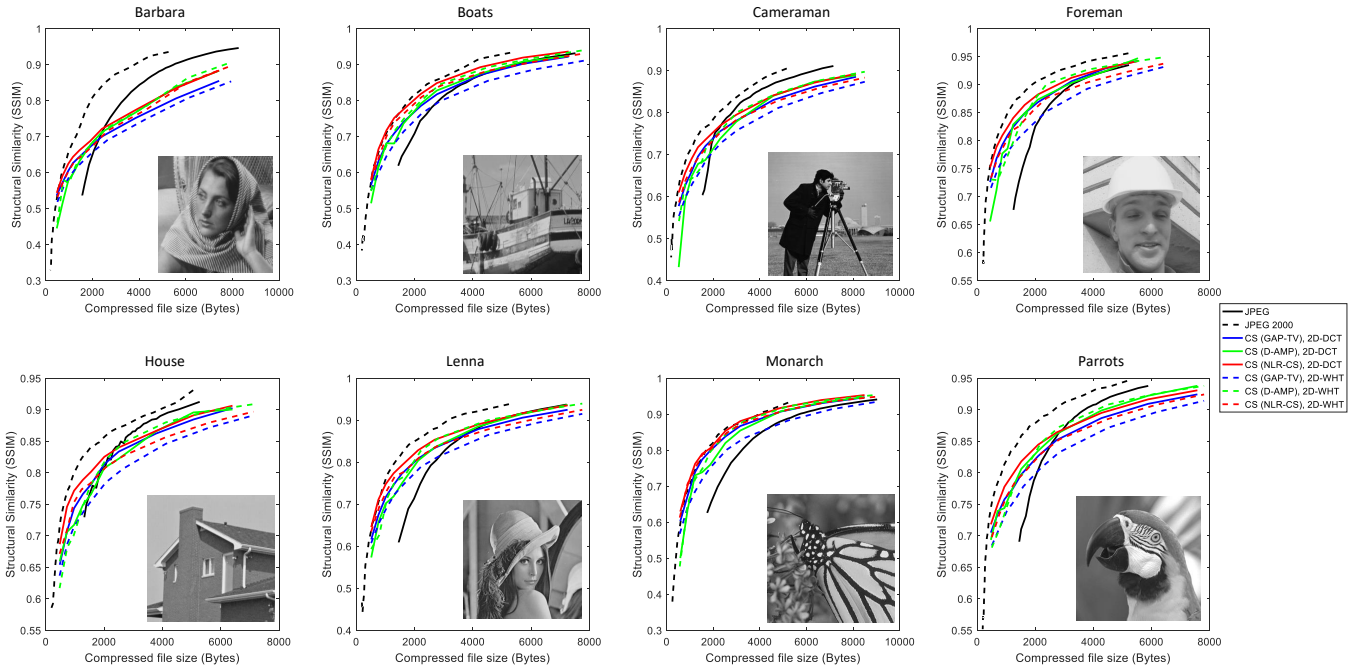

Fig. 4. Performance diagrams, SSIM vs. compressed file size (in bytes), comparing JPEG (black solid curves), JPEG2000 (black dash curves) with CSbIC compression using different sensing matrices – 2D-DCT (solid) and 2D-WHT (dash), and different reconstruction algorithms — GAP-TV (blue), NLR-CS (red) and D-AMP (green).

projection operators $\mathbf{D}$, most of these algorithms compute $\hat{\boldsymbol{x}}$, the estimated signal, by solving the minimization problem

$$\hat{\boldsymbol{x}} = \text{argmin}_{\boldsymbol{x}} \|\mathbf{D}\boldsymbol{x}\|_p, \quad \text{s.t.} \quad \boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{x}, \tag{16}$$

where p can be 0 ($\|\boldsymbol{f}\|_0$ denotes the number of non-zero components in $\boldsymbol{f}$) or 1, using $\|\boldsymbol{f}\|_1$ as a computationally-tractable approximation to $\|\boldsymbol{f}\|_0$ [44]. Alternatively, $\| \ \|_p$ can stand for the nuclear norm $\| \ \|_*$ to impose a low rank assumption [6].

Problem (16) is usually solved iteratively, where each iteration consists of two steps [24]. Beginning with an initial guess, the first step in each iteration projects the current guess onto the subspace of images which are consistent with the measurements, and the second step denoises the results obtained in the first step. Various projection and denoising algorithms [43], [45] can be employed in this general framework in order to achieve excellent results.

In our experiments we have used an improved variant of TV known as Generalized Alternating Projection Total Variation (GAP-TV) [28], as well as D-AMP and NLR-CS.

IV. Performance Testing

Our test material consisted of 8 monochrome images of 256x256 8-bit pixels (Barbara, Boats, Cameraman, Foreman, House, Lenna, Monarch, Parrots — see Fig. 4). These images were processed in a variety of test conditions, specified by encoder and reconstruction parameters. The outcome of processing an image in a particular test condition is the data rate, as expressed by the size of the coded image file (in bytes), and the reconstructed image quality, measured by structural similarity (SSIM) [46]. We preferred SSIM over Peak-Signal-to-Noise-Ratio (PSNR) as a measure of quality because SSIM better matches the quality perception of human visual system (refer to Fig. 5 and [1] for PSNR results, which provide similar observations to SSIM). This observation is verified by the exemplar reconstructed images compares with JPEG images in Fig. 6. Results are presented as points in SSIM vs. coded image size diagram. By connecting the points corresponding to test cases with different rate conntrol parameters (but identical in all other parametrs) we get performance curves which can be compared with each other in order to determine the better operating parameters. Performance curves for JPEG or JPEG2000 were obtained using the MATLAB implementation of those standards (the "imwrite" function). All the results presented in this section and more results can be found at [1], and these results can be reproduced by the code downloadable from [1].

A. *Effect of The Choice of Sensing Matrix*

The performance with the deterministic sensing matrices, 2D-DCT and 2D-WHT, was always significantly better than with the SRMs with the same fast transforms, SRM-DCT and SRM-WHT, respectively (Fig. 7), regardless of the reconstruction algorithm which was used. Within each of those groups (deterministic matrices and SRMs), the sensing matrices based on DCT generally yielded better performance than the those based on WHT (Fig. 7 right). However, (i) the difference, in terms of SSIM for the same file size, is generally smaller than the performance difference between deterministic

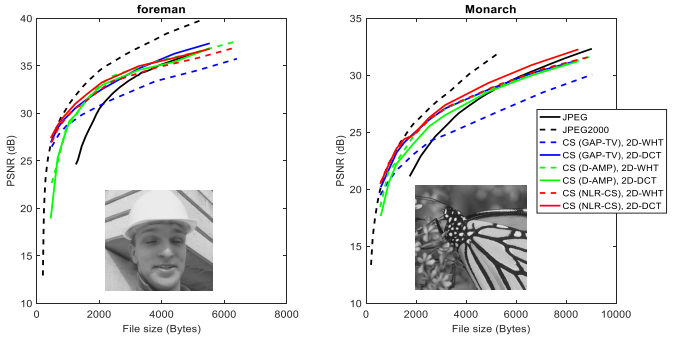

Fig. 5. PSNR vs. compressed file size (in bytes), comparing JPEG (black solid curves), JPEG2000 (black dash curves) with CSbIC compression using different sensing matrices – 2D-DCT (solid) and 2D-WHT (dash), and different reconstruction algorithms — GAP-TV (blue), NLR-CS (red) and D-AMP (green). Results with 8 images are available at [1].

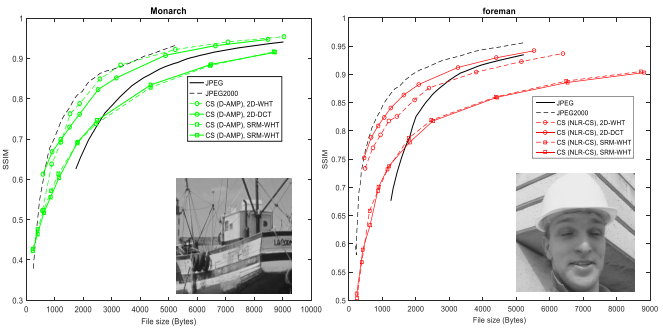

Fig. 7. Performance plots (SSIM vs. compressed file size in bytes) for the two images of CS with different sensing matrices. D-AMP (left) and NLR-CS (right) are used for reconstruction. Results with 8 images and 3 algorithms are available at [1].

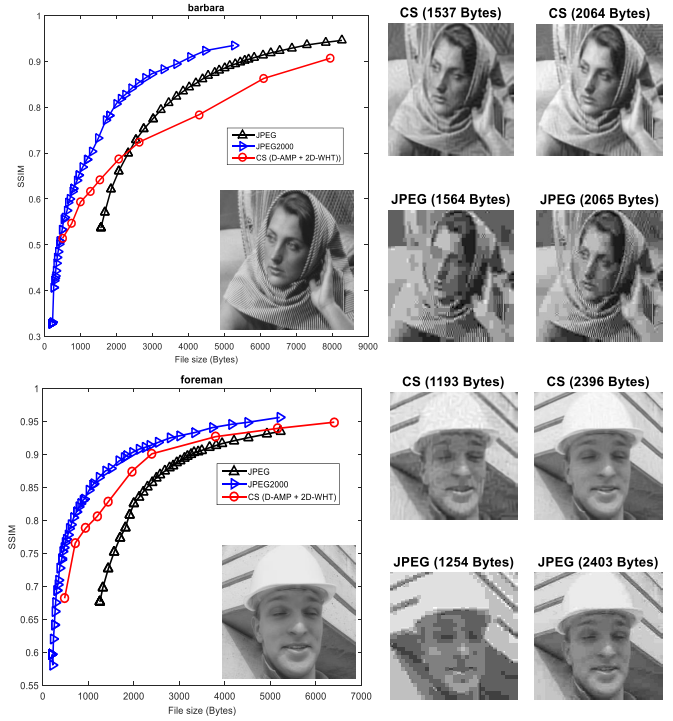

Fig. 6. Left: SSIM vs. compressed files size (in bytes) D-AMP+2D-WHT is used in CSbIC. Right: exemplar reconstructed images from CSbIC and JPEG with similar file size. More results are available at [1].

and structurally random matrices, (ii) the difference between SRM-DCT and SRM-WHT is generally much smaller than the difference between 2D-DCT and 2D-WHT, and (iii) the magnitude of the difference varies with the image and the reconstruction algorithm which is used. In particular, with GAP-TV the performance difference between DCT and WHT based matrices was most pronounced, whereas with D-AMP WHT performance was very close to that of DCT, and in a few cases the WHT based matrices slightly outperformed the DCT based ones.

An obvious reason for these differences is the fraction of the signal energy which is captured by the measurements with each sensing matrix. The DC measurement is the same, up to a scaling factor, in all four matrices. Therefore, when speaking about signal energy and measurements we refer only to the AC component and exclude the first measurement which is the DC component. Since SRMs randomize and whiten the signal prior to applying the transform, their measurements capture about M/N of the signal energy. The 2D-DCT measurements capture a much larger fraction of the signal energy because most of the signal energy is concentrated in the low frequencies. Since 2D-WHT is a crude approximation of spectral decomposition, its low "frequency" components capture less energy than those of 2D-DCT, but much more than the M/N of the SRM matrices. This argument is certainly true, but it is only a part of the explanation. Consider the sensing matrix given by

$$\tilde{\mathbf{\Phi}} = \mathbf{\Theta}\mathbf{\Phi}, \tag{17}$$

where $\mathbf{\Phi}$ is a $M \times N$ 2D-DCT or 2D-WHT deterministic sensing matrix and $\mathbf{\Theta}$ is given by

$$\mathbf{\Theta} = \begin{bmatrix} 1 & \mathbf{0}_{1\times(M-1)} \\ \mathbf{0}_{(M-1)\times 1} & \tilde{\mathbf{\Theta}} \end{bmatrix}, \tag{18}$$

where $\tilde{\mathbf{\Theta}}$ is a random orthonormal $(M-1)\times(M-1)$ matrix, defined by

$$\tilde{\mathbf{\Theta}} = \mathbf{SHR}, \tag{19}$$

where $\mathbf{R}$ is a diagonal matrix whose diagonal entries are IID random variables which get the values of ± 1 with equal probability; $\mathbf{H}$ is the DCT transform matrix of order $M-1$, with rows scaled to have unit norm; and $\mathbf{S}$ is a random selection matrix, *i.e.* a matrix in which each row and each column contains one entry of 1 and the rest are zero.

The measurements computed by $\tilde{\mathbf{\Phi}}$ are obtained by a random orthonormal transformation (ROT) of the measurements computed by $\mathbf{\Phi}$, hence we denote the sensing matrices $\tilde{\mathbf{\Phi}}$ ROT-2D-DCT and ROT-2D-WHT, respectively. The ROT matrices capture the same components of the signal as the corresponding deterministic matrices and the addition of the orthonormal transformation $\mathbf{\Theta}$ should not matter to any of the reconstruction algorithms. Furthermore, the quantization step, given by eq. (9), is the same and so are the quantization

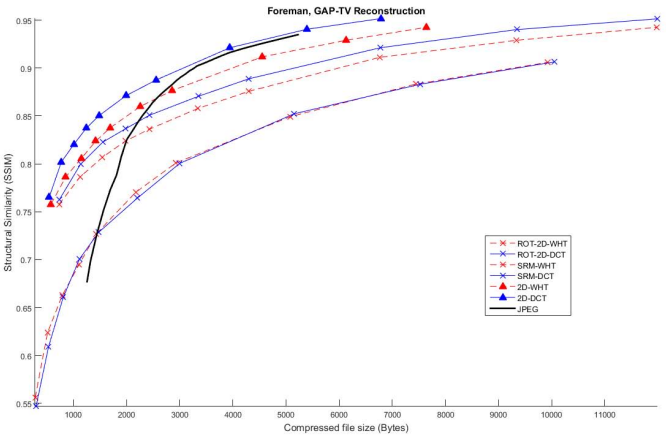

Fig. 8. Reconstructed image quality vs. file size of "Foreman" image with deterministic, SRM and ROT sensing matrices and with GAP-TV reconstruction.

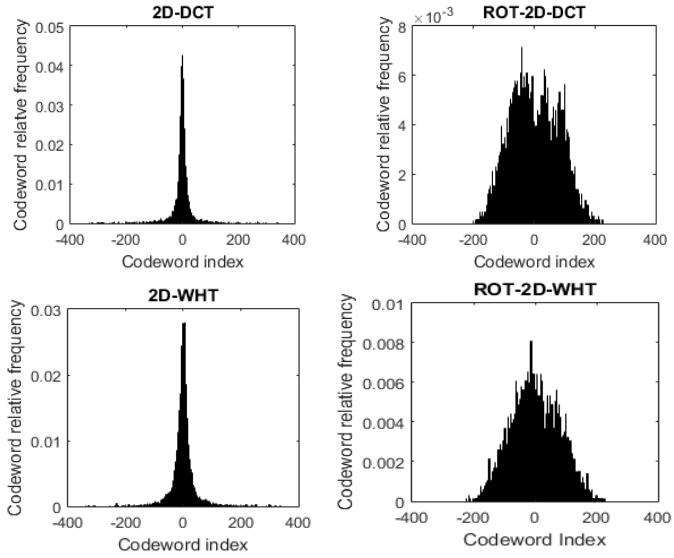

Fig. 9. Codeword histograms of of "Foreman" for deterministic and ROT sensing matrices and with compression ratio of 0.1. The distribution of the codewords in the deterministic case is much tighter than for the ROT matrices, and it is tighter for 2D-DCT.

noise and digitization noise, given by eq. (8). Nevertheless, the performance of the ROT matrices, while significantly better than the SRM matrices, was not as good as the deterministic ones. The difference in SSIM between the performance curves of a deterministic matrix and the corresponding ROT matrix was about a third or a quarter of the difference between the performance curves of the same deterministic matrix and the corresponding SRM matrix (see example in Fig. 8). Thus, multiplying the measurements vector $\mathbf{\Phi x}$ by $\mathbf{\Theta}$, causes a significant performance degradation. By its definition, $\tilde{\mathbf{\Theta}}$ is a DCT based SRM with a compression ratio of 1, which uses *local randomization*, that is, the input signal is randomized by randomly toggling the signs of its entries. This randomization method is different from the *global randomization* used in the SRM-DCT sensing matrix, where the input signal is randomized by randomly permuting its entries, but both variants have a similar effect on the distribution of the measurements: when a vector is multiplied by either one of these matrices, the distribution of the entries of the result is approximately zero mean gaussian or a mixture of zero mean gaussians with similar covariances [13]. Figure 9 shows the effect of multiplication by $\tilde{\mathbf{\Theta}}$ on the AC measurements. The measurements generated by the deterministic matrices have a narrowly peaked distribution with a long tail, while the measurements generated by the ROT matrices have a much wider, gaussian like distribution, with a shorter tail. As a result the entropy coding needs significantly more bits per measurement for coding the ROT measurements. The distribution of the measurements generated by SRM-DCT and SRM-WHT has also a Gaussian like shape, similar to that of the ROT generated measurements. Therefore, we may conclude that the difference between the SRM to the corresponding deterministic matrices has two components: one component is the different fraction of signal energy captured by each method, and is represented by the improvement from the SRM matrices performance to that of the ROT matrices. A second component relates to the different distributions of the measurements in the deterministic case and it is represented by the improvement from the ROT matrices performance to that of the corresponding deterministic matrices.

One can notice in Figure 9 that the histogram of 2D-WHT is a little wider than that of 2D-DCT. Thus, the advantage of 2D-DCT over 2D-WHT seems also to be not only because of the amount of signal energy captured in each case but also because of differences in the measurements distribution.

The arguments above fail to explain why, with some reconstruction algorithms, SRM-DCT performs better than SRM-WHT, since the captured fraction of signal energy, and the measurements distribution are similar in both cases. Furthermore, they do not explain why this difference is large with some reconstruction algorithms and hardly noticeable with others. Thus, there appear to be some subtle interactions between the prior assumptions used by the reconstruction algorithm and the type of the sensing matrix.

B. Analysis of Joint Quality Control

The effect of quantization on performance can be seen in Fig. 10. In general, as the quantization step is decreased, the compressed file size and the reconstructed image quality are increased. However, the rate of increase is not constant. For each compression ratio there seems to be three distinct regions for the quantization step. In the lower quality/data rate region, decreasing the quantization step results in a significant gain in quality with little increase in the compressed file size; in the higher quality/data rate region the opposite is true: decreasing the quantization step increases the file size significantly with hardly any quality gain; and in the narrow intermediate region between those two we get moderate increase in quality and file size as the quantization step is decreased. The intermediate region occurs at higher quantization step for lower compression ratios. As is clear from Fig. 10, for any required quality or file size, the best combination of compression ratio and quantization step is the one at which the constraint is met when

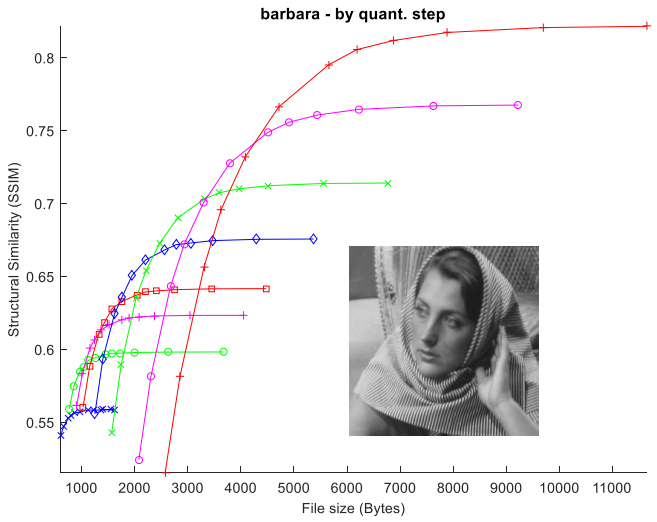

Fig. 10. Reconstructed image quality vs. file size for various combinations of compression ratio and quantization step (Barbara, 2D-WHT, GAP-TV). Each curve corresponds to a particular compression ratio {0.02, 0.03, 0.04, 0.05, 0.07, 0.10, 0.15, 0.20} and the points (markers) on each curve represent different quantization steps.

the quantization step is at the transition region. We determined empirically that this is satisfied when the compression ratio and quantization step satisfy (9) While the optimal value for the constant in the right hand side varies from picture to picture, we found that a constant of $C = 2.0$ gave good results for all images and we used it in our experiments.

C. Effect of the Reconstruction Algorithm

We found that the more modern reconstruction algorithms, NLR-CS and D-AMP, give a better performance than GAP-TV (Fig. 4). In most cases NLR-CS is a little better than D-AMP, especially at low-quality/low bit rate settings. We note however, that both NLR-CS and D-AMP require careful manual parameter tuning and data normalization in order to perform well, so it is possible that the results would be somewhat different with better tuning.

The performance gains of D-AMP, and NLR-CS come at the cost of a much higher complexity. The reconstruction time of GAP-TV, D-AMP and NLR-CS is 0.6, 35 and 147 seconds receptively, with the same compression rate on the same Intel i7 CPU with 24G memory. The algorithms were implemented in Matlab, but it appears that even with optimized implementation the last two algorithms would run considerably slower than GAP-TV, because they are inherently more complex.

D. Effect of Quantization

In order to get a better insight into the performance of our system, we isolate the component of the error which is due to quantization by running tests where the measurements vectors are reconstructed without quantization. Of course, in this case we cannot draw the usual performance curves of SSIM vs. compressed image file size, because without quantization (and entropy coding), there is no meaningful "compressed image file". Instead, we draw curves of quality, with and without quantization, vs. the compression ratio (CSr). In Fig. 11 we plot the SSIM of reconstructed images processed by CSbIC (including quantization), with that of images reconstructed directly from the unquantized measurements vectors. As expected, the results without quantization are consistently better than the results with quantization. In SSIM terms, the total decoding error is the difference between 1, which represents perfect reconstruction, and the SSIM value of the image decoded by the CSbIC. The error component due to quantization is the gap between curves with and without quantization. This gap varies with the reconstruction algorithms and compression ratio, both in absolute terms and relative to the total decoding error. GAP-TV seems to be not that sensitive to quantization error, with a SSIM decrease of 0.02–0.07 due to quantization. NLR-CS seems to be the most sensitive, with SSIM decrease of 0.04–0.10. The behavior of D-AMP is not uniform: at compression ratios below 0.10 the SSIM decrease is similar to NLR-CS, but at higher compression ratios the SSIM difference decreases, down to 0.01–0.04 at a compression ratio of 0.25.

E. Comparison with JPEG and JPEG 2000

CSbIC was tested with various sensing matrices and various reconstruction algorithms. As might be expected, the best combination depends on the particular image as well as on the quality or data rate operating point. However, in general the best performance was achieved with 2D-DCT+NLR-CS or 2D-WHT+D-AMP. In almost all test images CSbIC is decidedly better than JPEG at low bit rates (Fig. 4). A visual inspection shows (Fig. 6) that at low bit rates JPEG images have blocking artifacts, while in CSbIC the artifacts are more subtle.

In half of the images CS compression is better than JPEG for all quality levels. In the rest of the images JPEG is better for the higher quality cases, with the crossover happening at SSIM of 0.7–0.9 (0.8–0.9 for two out of the four images).

At this stage, the performance CSbIC is inferior to that of JPEG2000, but at the low quality regions, below about 0.7 SSIM, their performance is quite similar (Fig. 4).

V. Discussion

Both JPEG and CS leverage the inherent compressibility of natural images, but in different ways: In JPEG this is done at the encoder — the 2D DCT decomposition on 8x8 pixel blocks essentially represents the signal according to a sparsity basis; in classical CS the compressibility assumption comes into play only at the reconstruction. JPEG also exploits another property of natural images, namely that the high magnitude coefficients are the low frequency ones, therefore JPEG allocates more bits to their accurate representation. This domain-specific insight was built into CSbIC by using the deterministic sensing matrices described in Sec. II-A, which improved performance significantly in comparison to the SRMs of classical CS.

JPEG achieves lower bit rates solely through quantization, while CSbIC does it by quantization, and in addition, by

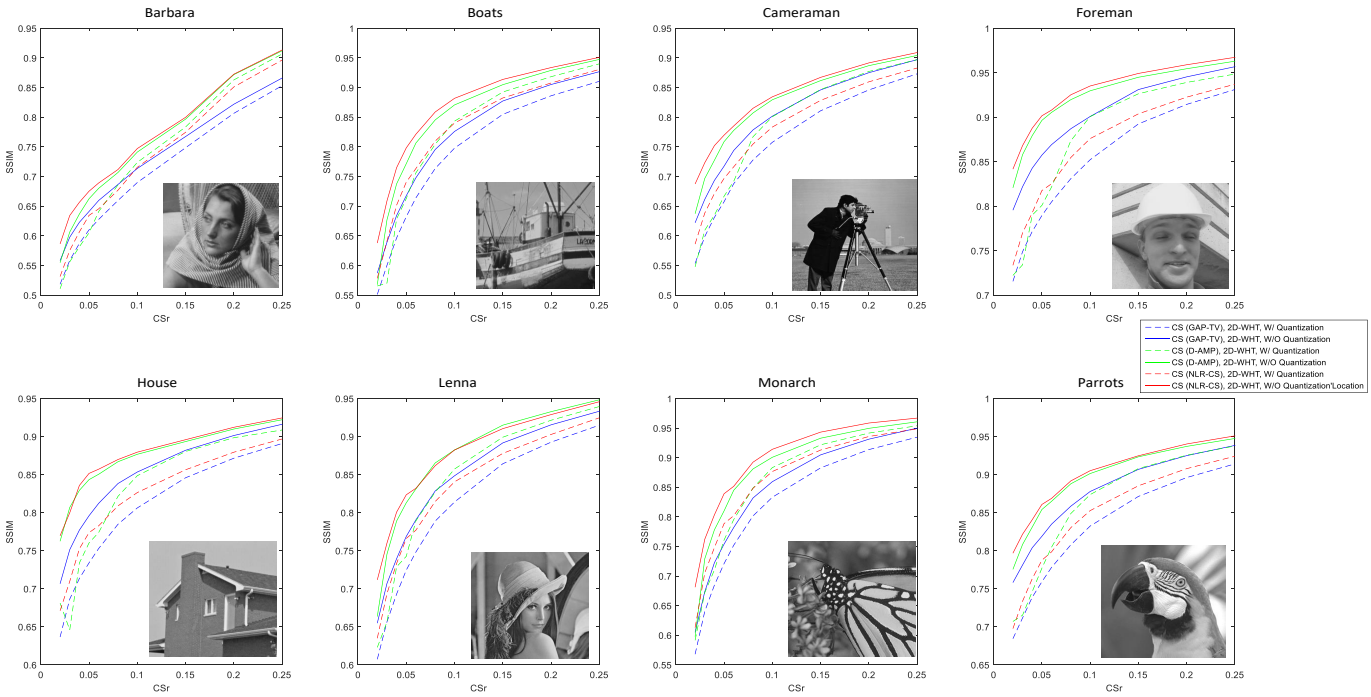

Fig. 11. SSIM vs. compression ratio (CSr), comparing the reconstructed image with quantization (dash lines) and without quantization (solid lines) for different reconstruction algorithms.

discarding most of the transform coefficients during the measurements selection. This is possible because CS reconstruction recovers the missing coefficients by invoking various structural assumptions about the original image. This gives CSbIC a significant advantage at low bit rates, where excessive quantization results in poor performance of JPEG. Another possible explanation for that is that JPEG processes 8x8 pixel blocks nearly independently. At low bit rates, only a small number of DCT coefficients are adequately represented in each JPEG block, which may be too little for blocks with much detail and causing the blocking artifacts which were observed. In contrast, all CS measurements contribute equally to the description of the whole image.

CSbIC yields better performance with certain types of sensing matrices and reconstruction algorithms. However, the less successful candidates also performed decently, and they may be the preferred choice in some situations. Binary-valued sensing matrices may be preferred for capturing measurements in the analog domain, and GAP-TV reconstruction may be useful when speed is important, for example, for quick image browsing.

JPEG is the culmination of many years of research into the properties of natural images. Wherever we could we leveraged this body of knowledge in CSbIC. However, much improvement can still be made, both in the quantization and entropy coding aspects of the system and in its CS aspects. The latter includes the sensing matrix design, the reconstruction algorithm, and the derivation of theoretical guarantees for successful reconstruction.

REFERENCES

[1] "CS vs. JPEG," https://sites.google.com/site/eiexyuan/csvsjpeg, accessed: 2017-02-24.

[2] E. J. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 489–509, February 2006.

[3] D. L. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, April 2006.

[4] M. Elad, *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*, 1st ed. Springer Publishing Company, Incorporated, 2010.

[5] C. Li, W. Yin, H. Jiang, and Y. Zhang, "An efficient augmented lagrangian method with applications to total variation minimization," *Computational Optimization and Applications*, vol. 56, no. 3, pp. 507–530, 2013.

[6] W. Dong, G. Shi, X. Li, Y. Ma, and F. Huang, "Compressive sensing via nonlocal low-rank regularization," *IEEE Transactions on Image Processing*, vol. 23, no. 8, pp. 3618–3632, 2014.

[7] C. A. Metzler, A. Maleki, and R. G. Baraniuk, "From denoising to compressed sensing," *IEEE Transactions on Information Theory*, vol. 62, no. 9, pp. 5117–5144, Sept 2016.

[8] V. K. Goyal, A. K. Fletcher, and S. Rangan, "Compressive sampling and lossy compression," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 48–96, 2008.

[9] J. N. Laska and R. G. Baraniuk, "Regime change: Bit-depth versus measurement-rate in compressive sensing," *IEEE Transactions on Signal Processing*, vol. 60, no. 7, pp. 3496–3505, July 2012.

[10] J. N. Laska, P. T. Boufounos, M. A. Davenport, and R. G. Baraniuk, "Democracy in action: Quantization, saturation, and compressive sensing," *Applied and Computational Harmonic Analysis*, vol. 31, no. 3, pp. 429 – 443, 2011.

[11] A. Zymnis, S. Boyd, and E. Candes, "Compressed sensing with quantized measurements," *IEEE Signal Processing Letters*, vol. 17, no. 2, pp. 149–152, Feb 2010.

[12] J. N. Laska, Z. Wen, W. Yin, and R. G. Baraniuk, "Trust, but verify: Fast and accurate signal recovery from 1-bit compressive measurements," *Signal Processing, IEEE Transactions on*, vol. 59, no. 11, pp. 5289–5301, Nov. 2011.

[13] R. Haimi-Cohen and Y. M. Lai, "Compressive measurements generated by structurally random matrices: Asymptotic normality and quantization." *Signal Processing*, vol. 120, pp. 71–87, 2016.

[14] W. Dai, H. V. Pham, and O. Milenkovic, "A comparative study of quantized compressive sensing schemes," in *2009 IEEE International Symposium on Information Theory*, June 2009, pp. 11–15.

[15] ——, "Distortion-rate functions for quantized compressive sensing," in *Networking and Information Theory, 2009. ITW 2009. IEEE Information Theory Workshop on*, June 2009, pp. 171–175.

[16] Y. Baig, E. M. K. Lai, and J. P. Lewis, "Quantization effects on compressed sensing video," in *Telecommunications (ICT), 2010 IEEE 17th International Conference on*, April 2010, pp. 935–940.

[17] D. Venkatraman and A. Makur, "A compressive sensing approach to object-based surveillance video coding," in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, April 2009, pp. 3513–3516.

[18] Information Technology – Digital Compression And Coding Of Continuous Tone Still Images – Requirements And Guidelines, "CCITT Recommendation T.81," 1992.

[19] Information technology – JPEG 2000 image coding system: Core coding system, "ITU-T Recommendation T.800," 2002.

[20] M. F. Duarte, M. A. Davenport, D. Takhar, J. N. Laska, T. Sun, K. F. Kelly, and R. G. Baraniuk, "Single-pixel imaging via compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 83–91, 2008.

[21] G. Huang, H. Jiang, K. Matthews, and P. Wilford, "Lensless imaging by compressive sensing," pp. 2101–2105, Sept 2013.

[22] J. Romberg, "Imaging via compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 14–20, 2008.

[23] X. Yuan, H. Jiang, G. Huang, and P. Wilford, "Lensless compressive imaging," *arXiv:1508.03498*, 2015.

[24] ——, "SLOPE: Shrinkage of local overlapping patches estimator for lensless compressive imaging," *IEEE Sensors Journal*, vol. 16, no. 22, pp. 8091–8102, November 2016.

[25] J. Bioucas-Dias and M. Figueiredo, "A new TwIST: Two-step iterative shrinkage/thresholding algorithms for image restoration," *IEEE Transactions on Image Processing*, vol. 16, no. 12, pp. 2992–3004, December 2007.

[26] M. A. T. Figueiredo, R. D. Nowak, and S. J. Wright, "Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems," *IEEE Journal of Selected Topics in Signal Processing*, pp. 586–597, Dec. 2007.

[27] S. Ji, Y. Xue, and L. Carin, "Bayesian compressive sensing," *IEEE Transactions on Signal Processing*, vol. 56, no. 6, pp. 2346–2356, June 2008.

[28] X. Yuan, "Generalized alternating projection based total variation minimization for compressive sensing," in *2016 IEEE International Conference on Image Processing (ICIP)*, Sept 2016, pp. 2539–2543.

[29] J.-H. Ahn, "Compressive sensing and recovery for binary images," *IEEE Transactions on Image Processing*, vol. 25, no. 10, pp. 4796–4802, 2016.

[30] W. K. Pratt, J. Kane, and H. C. Andrews, "Hadamard transform image coding," *Proceedings of the IEEE*, vol. 57, no. 1, pp. 58–68, Jan 1969.

[31] M. F. Duarte and R. G. Baraniuk, "Kronecker compressive sensing," *IEEE Transactions on Image Processing*, vol. 21, no. 2, pp. 494–504, Feb 2012.

[32] B. J. Fino and V. R. Algazi, "Unified matrix treatment of the fast walsh-hadamard transform," *IEEE Transactions on Computers*, vol. C-25, no. 11, pp. 1142–1146, Nov 1976.

[33] E. J. Candes and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?" *IEEE Trans. Inf. Theor.*, vol. 52, no. 12, pp. 5406–5425, Dec. 2006.

[34] ——, "Decoding by linear programming," *IEEE Transactions on Information Theory*, vol. 51, no. 12, pp. 4203–4215, Dec 2005.

[35] T. T. Do, L. Gan, N. H. Nguyen, and T. D. Tran, "Fast and efficient compressive sensing using structurally random matrices," *IEEE Transactions on Signal Processing*, vol. 60, no. 1, pp. 139–154, Jan 2012.

[36] T. T. Do, T. D. Tran, and L. Gan, "Fast compressive sampling with structurally random matrices," in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, March 2008, pp. 3369–3372.

[37] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Norwell, MA, USA: Kluwer Academic Publishers, 1991.

[38] G. G. Langdon, "Arithmetic coding," *IBM J. Res. Develop*, vol. 23, pp. 149–162, 1979.

[39] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, January 2011.

[40] M. A. Figueiredo, J. M. Bioucas-Dias, and R. D. Nowak, "Majorizationminimization algorithms for wavelet-based image restoration," *IEEE Transactions on Image Processing*, vol. 16, no. 12, pp. 2980–2991, 2007.

[41] Y. Wang, J. Yang, W. Yin, and Y. Zhang, "A new alternating minimization algorithm for total variation image reconstruction," *SIAM Journal on Imaging Sciences*, vol. 1, no. 3, pp. 248–272, 2008.

[42] X. Liao, H. Li, and L. Carin, "Generalized alternating projection for weighted-$\ell_{2,1}$ minimization with applications to model-based compressive sensing," *SIAM Journal on Imaging Sciences*, vol. 7, no. 2, pp. 797–823, 2014.

[43] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

[44] E. Candes, M. Wakin, and S. Boyd, "Enhancing sparsity by reweighted ℓ_1 minimization," *Journal of Fourier Analysis and Applications*, vol. 14, no. 5, pp. 877–905, 2008.

[45] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3d transform-domain collaborative filtering," *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2080–2095, August 2007.

[46] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.