A FAST DECODER FOR COMPRESSED SENSING BASED MULTIPLE DESCRIPTION IMAGE CODING

Md Mashud Hyder and Kaushik Mahata

School of Electrical Engineering and Computer Science, University of Newcastle,
Callaghan, NSW 2308, Australia.
md.hyder@studentmail.newcastle.edu.au, kaushik.mahata@newcastle.edu.au

ABSTRACT

Multiple description coding (MDC) offers an elegant approach to data transmission over lossy packet-based networks. This paper proposes an MDC decoder for Compressed Sensing (CS) based MDC. Our decoder minimizes ℓ^0 norm of the total variation of the image in a recursive manner, making it effective when different descriptions experience different time delays in the network. The proposed approach brings in a significant performance improvement in reconstruction accuracy and reconstruction time.

Index Terms— Multiple description coding, compressed sensing, ℓ^0 minimization, image coding

1. INTRODUCTION

In Multiple Description Coding (MDC) an information source is coded into several chunks of data (descriptions) so that the source can be recovered from a small subset of chunks with a reasonable accuracy. This makes MDC effective for transmitting image and video over lossy networks [1]. Many MDC methods have been proposed. The simplest methods partition the original image into multiple sub-images, which are coded separately giving different descriptions. JPEG was extended in this way in [2]. The main drawback of this approach is that the loss of a few bits in a block can make this block and all the following blocks in a row undecodable. Many error resilient approaches have been proposed [3].

Wavelet based MDC is another popular approach. A MDSQ-based wavelet algorithm is proposed in [4] by optimally selecting the number of diagonals and the quantization steps of the MD scaler quantizer. Different strategies are also used to introduce an amount of redundancy among the descriptions. However, most the strategies are designed for a general information source without the ability to adapt to the specific image under consideration. This cause insufficient redundancy and when certain packets are lost or a channel transmission fails, only partial redundancy can significantly contribute to recover the corrupted contents. Nevertheless, the recent theory of compressed sensing (CS) [5] allows to add redundancies to an image, while its specific features emphasized [6]. However, decoding algorithm in [6] is often slow when the successive descriptions arrive at the receiver at different time instants. In this paper, we propose a recursive algorithm called recursive ℓ^0 approximation (RecLZA). It achieves better reconstruction quality from few descriptions. When the descriptions arrive at the receiver at different time instants RecLZA updates the reconstructed image using a fast recursive approach.

The research is supported by the Australian Research Council.

2. THE MDC ENCODER

The MDC encoder is similar to the encoder proposed in [6]. Consider a $p \times p$ image z. Let, $x_0 = \mathrm{vec}(z)$, where vec(.) is the matrix vectorization operator. Let $n = p^2$. Construct $\Theta \in \mathbb{R}^{n\times n}$ such that all its entries are mutually independent, and identically distributed random variables with mean zero and variance unity. Form

$$Y = \Theta x_0 \tag{1}$$

Clearly, $Y \in \mathbb{R}^n$. The measurement vector Y and matrix Θ are divided into $q \geq 2$ equal parts and indexed them from $\{1, \cdots, q\}$:

$$Y = [\, y_1' \ \cdots \ y_q' \,]', \quad \Theta = [\, \Phi_1' \ \cdots \ \Phi_q' \,]',$$

so that $y_i = \Phi_i x_0 \in \mathbb{R}^s$, and $\Phi_i \in \mathbb{R}^{s\times n}$. It is assumed that $s = n/q$ is an integer. Each y_i represents an individual description, and is transmitted independently. In practice, a small subset of descriptions is sufficient for decoder to reconstruct x_0. Hence, the number of rows of Θ can be smaller than n.

3. THE PROPOSED MDC DECODER

It is assumed that the decoder can generate the same Θ by using the same seed as the encoder. When all the descriptions are available, the decoder can reconstruct x_0 by collecting $\{y_i\}_{i=1}^q$ in Y, and computing $x_0 = \Theta^{-1}Y$. However, in practice only a subset of descriptions is available in decoder at an instant. Suppose that $r^{(1)} < q$ descriptions are available and their indices are $i(1), \ldots, i(r^{(1)})$. Generate $y = [y_{i(1)}' \ y_{i(2)}' \cdots \ y_{i(r^{(1)})}']'$ and $\Phi = [\Phi_{i(1)}' \ \Phi_{i(2)}' \cdots \ \Phi_{i(r^{(1)})}']'$ then we have[1],

$$y = \Phi x \tag{2}$$

where $y \in \mathbb{R}^m$, $\Phi \in \mathbb{R}^{m\times n}$, and $m = s \times r^{(1)}$. Since $m < n$, the equation $\Phi x = y$ has infinitely many solutions, which span an affine set $\mathbb{X} \subset \mathbb{R}^n$ of dimension $n - m$. In order to recover x_0 from y, we need to know a special property of x_0 such that among all the points in $\mathbb{X}$ only x_0 satisfies that property. In particular, it has been shown in [7, 8] that there exists a map $g : \mathbb{X} \to R^n$ such that $g(x)$ is a

[1] Note that A' denotes the transpose of a matrix A.

sparse vector only if $x = x_0$. The map g is described in detail in the next section. The sparsity property of $g(x_0)$ allows the recovery of x_0 as

$$x_0 = \arg\min_x \|g(x)\|_0 \quad \text{subject to} \quad y = \Phi x, \tag{3}$$

where $\|g(x)\|_0$ is ℓ^0 norm of $g(x)$:

$$\|g(x)\|_0 := \lim_{\epsilon \to 0} \{|g_1(x)|^\epsilon + \cdots + |g_n(x)|^\epsilon\},$$

which is the number of nonzero components in $g(x)$. The similar idea is used in CS [5]. Unfortunately, solving (3) is NP-hard, while it gives the highest possibility of sparse recovery from smaller m. Basis Pursuit (BP) is another popular approach for solving (3) where the ℓ^0 norm is replaced by ℓ^1 norm [9] and it can be recast as a linear program (LP). With high probability, LP can recover x_0 from (3) when level of sparsity[2] (k) of $g(x_0)$ is less than $m/2$ [5]. It is used in the decoder of [6] to solve (3). The proposed RecLZA algorithm can efficiently solve (3) by using an ℓ^0 approximation approach. The quality of reconstructed image is better in RecLZA compared to LP. Moreover, in some settings, it is faster than LP.

3.1. The Total Variation Map

The fact that the gradient of an image is sparse [8] is used to define the map g. Let $z_{i,j}$ denotes the pixel in the i^{th} row and j^{th} column of image z. We define the horizontal and vertical gradient oparators $D_h : \mathbb{R}^{p\times p} \to \mathbb{R}^{p\times p}$ and $D_v : \mathbb{R}^{p\times p} \to \mathbb{R}^{p\times p}$ as

$$[D_h(z)]_{i,j} = \begin{Bmatrix} z_{i+1,j} - z_{i,j} \ ; \ i < p \\ 0 \ ; \ i = p \end{Bmatrix}$$
$$[D_v(z)]_{i,j} = \begin{Bmatrix} z_{i,j+1} - z_{i,j} \ ; \ j < p \\ 0 \ ; \ j = p \end{Bmatrix}$$

The total variation map performs better in (3). In the following it will be convenient work with the vectorized versions vec$\{D_h(z)\}$ and vec$\{D_v(z)\}$. Clearly there exists $\Gamma_h \in \mathbb{R}^{n\times n}$ and $\Gamma_v \in \mathbb{R}^{n\times n}$ such that

$$\begin{aligned} \text{vec}\{D_h(z)\} &= \Gamma_h x \\ \text{vec}\{D_v(z)\} &= \Gamma_v x \end{aligned} \tag{4}$$

The expressions for Γ_h and Γ_h can be derived in a straightforward manner. Let h_i and v_i denote i^{th} row of Γ_h and Γ_v, respectively. Then we define $g : \mathbb{R}^n \to \mathbb{R}^n$ such that the i-th component of $g(x)$ is given by

$$[g(x)]_i = \sqrt{(h_i x)^2 + (v_i x)^2} = \sqrt{x' S_i x} \tag{5}$$

where $S_i = h_i' h_i + v_i' v_i$, $i = 1, 2, \ldots, n$.

3.2. ℓ^0 Norm Approximation

RecLZA approximately formulate the objective function in (3) to which gradient based method can be applied. The Gaussian functions seems useful for this purpose [10]. Define

$$f_\sigma(\alpha) = \exp(-0.5\alpha^2/\sigma^2). \tag{6}$$

[2]Only $k \ll n$ elements in $g(x_0)$ are nonzero.

Clearly $f_\sigma(0) = 1$. In addition, for any given $\alpha > 0$ we have $\lim_{\sigma\to 0} f_\sigma(\alpha) = 0$. Consequently, the function

$$F_\sigma(x) = \sum_{i=1}^{n} f_\sigma(g_i(x)).$$

behaves like $n - ||g(x)||_0$ when $\sigma \to 0$. Defining (3):

$$x_* = \arg\max_x F_\sigma(x) \quad \text{subject to} \quad y = \Phi x, \tag{7}$$

we see that $x_* \to x_0$ as $\sigma \to 0$. However, $F_\sigma(x)$ has many local maxima for small value of σ. Consequently, it is very difficult to directly maximize $F_\sigma(x)$ for a very small σ. Nevertheless, as σ increases, $F_\sigma(x)$ becomes smoother, and for a sufficiently large σ, one has $x_* = \Phi'(\Phi\Phi')^{-1}y$ [10]. Hence, the standard procedure is to take a large σ initially and solve (7). Subsequently, σ is reduced by some small factor and (7) is solved again. The procedure is repeated until a convergence criterion is satisfied. Since the value of σ changes slowly, the numerical algorithm to solve (7) is always initialized close to the actual maximum and it has small likelihood to get trapped in a local maxima.

3.3. Algorithm Derivation

The Lagrangian $L(x, \nu)$ associated with the problem (7) is

$$L(x, \nu) = F_\sigma(x) + \nu'(\Phi x - y), \tag{8}$$

where $\nu \in \mathbb{R}^{m\times 1}$ is the vector of Lagrange multipliers. Now (7) implies that there exists ν_* such that (x_*, ν_*) is a stationary point of $L(x, \nu)$, *i.e*,

$$\begin{aligned} \frac{\partial L(x_*, \nu_*)}{\partial x} &= \frac{\partial F_\sigma(x_*)}{\partial x} + \Phi'\nu_* = 0. \\ \frac{\partial L(x_*, \nu_*)}{\partial \nu} &= \Phi x_* - y = 0. \end{aligned} \tag{9}$$

Also using the definition of g and F_σ it can be shown that

$$\frac{\partial F_\sigma(x)}{\partial x} = \Upsilon_\sigma(x)x, \quad \Upsilon_\sigma(x) = \frac{-1}{\sigma^2}\sum_{i=1}^{n} f_\sigma(g_i(x))S_i. \tag{10}$$

Now using (9) and (10) we have

$$\begin{pmatrix} \Upsilon_\sigma(x_*) & \Phi' \\ \Phi & 0 \end{pmatrix} \begin{pmatrix} x_* \\ \nu \end{pmatrix} = \begin{pmatrix} 0 \\ y \end{pmatrix} \tag{11}$$

The system of equation (11) is symmetric, and $\Upsilon_\sigma(x)$ is a sparse matrix. This allows us to solve the system above using an iterative method [11]. It also follows from (11) that

$$x_* = \Upsilon_\sigma^{-1}(x_*)\Phi' \left[\Phi\, \Upsilon_\sigma^{-1}(x_*)\Phi'\right]^{-1} y. \tag{12}$$

Equation (12) is nonlinear, and cannot be solved analytically. However one possible avenue is to use (12) in a fixed point iteration, for which the the following Lemma is proved in [12].

Lemma 1 *Let us define the map $\zeta : \mathbb{R}^n \to \mathbb{R}^n$ such that*

$$\zeta(x) = \Upsilon_\sigma^{-1}(x)\Phi' \left[\Phi\, \Upsilon_\sigma^{-1}(x)\Phi'\right]^{-1} y. \tag{13}$$

Then $\Phi\zeta(x) = y$. Let $x \in \mathbb{R}^n$ such that $\Phi x = y$, $\zeta(x) \neq x$, and

$$\left[\frac{\partial F_\sigma(x)}{\partial x}\right] \neq 0. \tag{14}$$

Table 1. RecLZA Algorithm

function (x^e, σ_e)=RecLZA$(x^{(0)}, y, \Phi, \sigma)$
1. Choose $\gamma, \rho \in \{0, 1\}$
repeat
2. Set $\lambda = 1$.
3. while $F_\sigma(\lambda\zeta(x^{(i)}) + (1-\lambda)x^{(i)}) < F_\sigma(x^{(i)})$
   $\lambda = \gamma\lambda$.
   end
4. $x^{(i+1)} = \lambda\zeta(x^{(i)}) + (1-\lambda)x^{(i)}$.
5. Calculate $\tau = \|x^{(i+1)} - x^{(i)}\|_2$ and $h_r = \frac{h^{(i+1)}}{h^{(i)}}$
6. If $\tau < \rho\sigma$ or $h_r > 1$ then $\sigma = \rho\sigma$
until stopping criterion satisfied
7. Set $x^e = x^{(i+1)}$ and $\sigma_e = \sigma$
end function

Then there exists λ *satisfying* $0 < \lambda \leq 1$ *such that*

$$F_\sigma\left\{\lambda\zeta(x) + (1-\lambda)x\right\} > F_\sigma(x).$$

□

Lemma 1 ensures convergence of a refined version of the fixed point iteration to a local maxima.

3.4. The Proposed Algorithm for Decoder

Based on the idea of previous section, the algorithm to solve (12) is shown in Table 1. Assume that initially $r^{(t)}$ descriptions are available at decoder. Their indices are $\{i(1), \cdots, i(r^{(t)})\}$. Generate $y^{(t)} = [y'_{i(1)}\ y'_{i(2)} \cdots y'_{i(r^{(t)})}]'$ and $\Phi_t = [\Phi'_{i(1)}\ \Phi'_{i(2)} \cdots \Phi'_{i(r^{(t)})}]'$. The initial value of x is calculated as $x^{(0)} = \Phi'_t(\Phi_t\Phi'_t)^{-1}y^{(t)}$. Set $y = y^{(t)}, \Phi = \Phi_t, \sigma = 1$ and call RecLZA.

As described, the probability of reconstruction of x_0 from (3) is high when sparsity k of $g(x_0)$ is less than $m/2$, where m is the size of y. For a larger m, we can assume that $k < m/2$. Let $\overline{x}$ is recovered by using RecLZA. Hence, it is desired that $g(\overline{x})$ should have at least $n - m/2$ smaller elements. In particular, we calculate a parameter $h^{(i)}$ after each iteration such that $n - m/2$ components of $g(x^{(i)})$ have absolute values less than $h^{(i)}$. Clearly, smaller $h^{(i)}$ indicates better signal to noise ratio (SNR) of recovered signal. The parameter h_r is used to measure the change of $h^{(i)}$ after each step. How fast σ will be lowered depends on h_r and ρ. The parameter ρ also determines how much σ will be lowered. We set $\rho = 0.5$ in our experiments. The decreasing factor γ determines how fast we backtrack in the line search step. It is standard practice to choose $0 < \gamma < 1$.

After starting with $x^{(0)}, y, \Phi$ and σ, the algorithm updates $x^{(i)}$ after every iteration. It also traces the values of τ and h_r. If step 3 is satisfied for some τ or h_r, then σ is lowered by ρ. The process is repeated until a stopping criterion is satisfied. The stopping criterion of RecLZA is based on h. If h does not decrease in an iteration while σ is kept fixed, then σ is lowered by a factor ρ *i.e.* $\sigma = \rho\sigma$ and RecLZA starts iteration again. However, after decreasing σ if h can not decrease then RecLZA stops iteration and set $\sigma_e = \sigma$.

When new descriptions arrive at the decoder we must update our estimate as the SNR of the reconstructed signal improves with increase in the number of descriptions. Assume that after starting with $r^{(t)}$ descriptions, the RecLZA reconstructs a signal $x^{(t)} = x^e$ and corresponding smallest σ is $\sigma_t = \sigma_e$. Now the decoder receives additional $r^{(t+1)}$ descriptions and their indices are $\{i(1), \cdots, i(r^{(t+1)})\}$. Unlike the algorithm in [6] RecLZA adopts a recursive approach to account for the additional data, which saves the computation time significantly. Let

$$y^{(t+1)} = [y'_{i(1)}\ \cdots\ y'_{i(r^{(t+1)})}]',\ \ \Phi_{t+1} = [\Phi'_{i(1)}\ \cdots\ \Phi'_{i(r^{(t+1)})}]'$$

Now we have an additional constraint $y^{(t+1)} = \Phi_{t+1}x$ to satisfy. This can also be solved by calling RecLZA. In fact we can accelerate the convergence speed significantly by good initialization. Since the new solution cannot be far from x^e, we initialize RecLZA as close to x^e as possible. We find $w^{(t)}$ in the orthogonal complement of span$\{\Phi_t\}$ in span$\{[\Phi'_t\ \Phi'_{t+1}]'\}$ such that $\|y^{(t+1)} - \Phi_{t+1}x^{(t)} - \Phi_{t+1}w^{(t)}\|_2$ is minimized. Then set $x^{(0)} = x^{(t)} + w^{(t)}, y = [y^{(t)'}\ y^{(t+1)'}]', \Phi = [\Phi'_t\ \Phi'_{t+1}]', \sigma = \sigma_t$ and call RecLZA again.

4. EXPERIMENTS

Each of the following set of experiments was performed on various images. However, for brevity, we only present the results obtained by applying the algorithm on the “Camera Man” image [7]. In case of RecLZA and Linear Programming (LP)[3] the matrix Θ was generated by $n \times n$ randomly select entries from a mean-zero Gaussian distribution. For RecLZA, the value of ρ and γ are fixed to 0.5. For DCT, the descriptions are generated by randomly sampling the coefficient in frequency domain. For faster computation, the entire image (256×256) is divided into blocks sized 32×32. In the encoder, Y is divided into $q = 20$ equal descriptions. In this simulation, we do not consider transmission error *i.e.* all packets are transmitted properly without any bit loss. Our simulations are performed in MATLAB7 environment using an Intel Core 2 Duo, 2.66 GHz processor with 2GB of memory.

Figure 1 gives PSNR plots for different algorithms. The reconstruction starts with 4 descriptions, and subsequently, one description is added at a time. We also compare performance of RecLZA when no recursion is used in it. In case of RecLZA without recursion and LP, each time when a new description arrives, we add it with previous descriptions and construct y and Φ. Initialize $x^{(0)}$ as minimum 2-norm solution of $\Phi x = y$. Note that for RecLZA the PSNR is higher, which also improves at a faster rate as more descriptions are added.

Figure 2 illustrate the efficiency of proposed recursive RecLZA in terms of required iterations and computation time. Note that recursive RecLZA has remarkable performance improvement compared to other two. For example, initially when 4 descriptions are used, both recursive RecLZA and RecLZA without recursion require 4 iterations and same time. However, when another 3 descriptions are added, i.e. descriptions= 7, recursive RecLZA requires only 3 iterations compared to 10 of RecLZA without recursion to achieve similar PSNR. Again, in terms of required computation time, LP shows the worst performance (average 30 sec) whereas recursive RecLZA requires only 3 seconds (average). Another similar result is shown in Figure 3 when two descriptions are added at a time.

[3]http://www.acm.caltech.edu/l1magic/

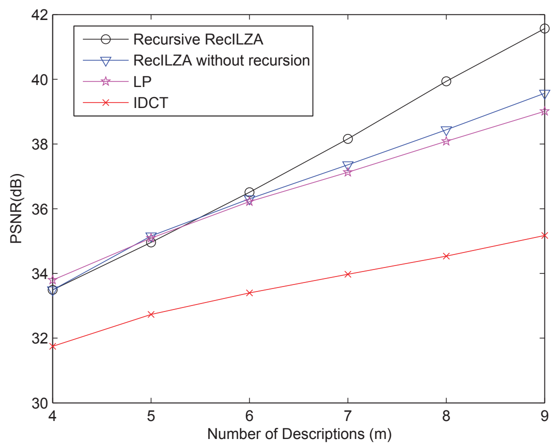

Fig. 1. PSNR versus number of descriptions for “Camera Man” image (256×256). One description is added at a time.

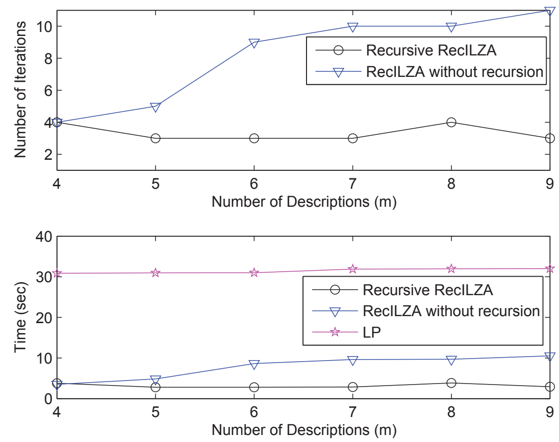

Fig. 2. Number of iterations and computation time required to reconstruct image when one description is added at a time.

5. CONCLUSION

The paper presents a CS based fast MDC decoder. The decoder minimizes ℓ^0 norm of the total variation of the image in a recursive manner. The computational cost around equation (12) can be minimized efficiently by computing the inversion in function level *e.g.,* by using DCT or FFT. In addition, our algorithm requires only a small number of iterations to converge, making it ideal to implement on accelerated hardware platform like FPGAs.

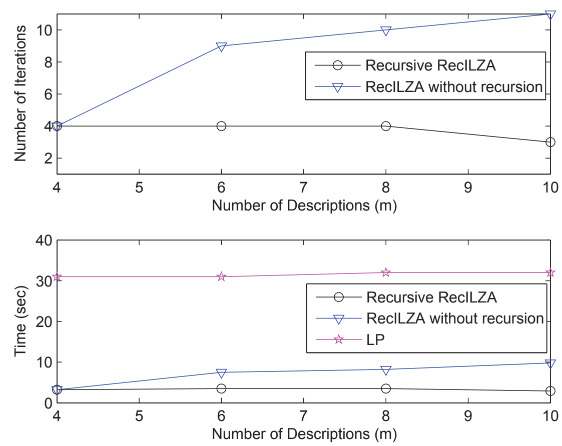

Fig. 3. Two descriptions are added at a time.

6. REFERENCES

[1] V.K. Goyal, “Multiple description coding: compression meets the network,” *IEEE Signal Processing Magazine*, vol. 18, pp. 74–93, September 2001.

[2] V.A. Vaishampayan, “Design of multiple description scalar quantizers,” *IEEE Transactions on Information Theory*, vol. 39, no. 3, pp. 821–834, May 1993.

[3] D.W. Redmill and N.G. Kingsbury, “The erec: an error-resilient technique for coding variable-length blocks of data,” *IEEE Trans.on Image Processing*, vol. 5, pp. 565– 574, April 1996.

[4] S.D. Servetto, K. Ramchandran, V.A. Vaishampayan, and K. Nahrstedt, “Multiple description wavelet based image coding,” *IEEE Transactions on Image Processing*, vol. 9, no. 5, pp. 813–826, May 2000.

[5] D. L. Donoho, “Compressed sensing,” *IEEE Trans. Information Theory*, vol. 52, pp. 1289–1306, April 2006.

[6] Yifu Zhang, Shunliang Mei, Quqing Chen, and Zhibo Chen, “A multiple description image/video coding method by compressed sensing theory,” in *IEEE International Symposium on Circuits and Systems, ISCAS 2008*, 2008, pp. 1830–1833.

[7] Justin Romberg, “Imaging via compressive sampling,” *IEEE Signal Processing Magazine*, vol. 25, pp. 14–20, March 2008.

[8] Leonid I. Rudin, Stanley Osher, and Emad Fatemi, “Nonlinear total variation based noise removal algorithms,” *Physica D*, vol. 60, pp. 259–68, November 1992.

[9] S. S. Chen, D. L. Donoho, and M. A. Saunders, “Atomic decomposition by basis pursuit,” *SIAM Journal on Scientific Computing*, vol. 20, pp. 33–61, 1998.

[10] G. H. Mohimani, M. Babaie-Zadeh, and C. Jutten, “Fast sparse representation based on smoothed l0 norm,” in *7th International Conference on Independent Component Analysis and Signal Separation*, 2007, pp. 389–396.

[11] C.C. Paige and M.A. Saunders, “Solution of sparse indefinite systems of linear equations,” *SIAM J. Numer. Anal.*, vol. 12, pp. 617–629, 1975.

[12] M. Hyder and K. Mahata, “An approximate l0 norm minimization algorithm for compressed sensing,” in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2009, pp. 3365–3368.