A MULTISCALE FRAMEWORK FOR COMPRESSIVE SENSING OF VIDEO

Jae Young Park

Department of EECS, University of Michigan
Ann Arbor, MI 48109

Michael B. Wakin[*]

Division of Engineering, Colorado School of Mines
Golden, CO 80401

ABSTRACT

Compressive Sensing (CS) allows the highly efficient acquisition of many signals that could be difficult to capture or encode using conventional methods. From a relatively small number of random measurements, a high-dimensional signal can be recovered if it has a sparse or near-sparse representation in a basis known to the decoder. In this paper, we consider the application of CS to video signals in order to lessen the sensing and compression burdens in single- and multi-camera imaging systems. In standard video compression, motion compensation and estimation techniques have led to improved sparse representations that are more easily compressible; we adapt these techniques for the problem of CS recovery. Using a coarse-to-fine reconstruction algorithm, we alternate between the tasks of motion estimation and motion-compensated wavelet-domain signal recovery. We demonstrate that our algorithm allows the recovery of video sequences from fewer measurements than either frame-by-frame or inter-frame difference recovery methods.

1. INTRODUCTION

Many naturally occurring signals are sparse or compressible in the sense that when expressed in the proper basis, relatively few of the expansion coefficients are large. Such a concise representation naturally leads to efficient compression algorithms. The emerging theory of Compressive Sensing (CS) indicates that such models can also be used to simplify the acquisition of high-dimensional images or signals that might otherwise be difficult to collect or encode [1, 2]. Rather than collecting an entire ensemble of signal samples, CS requires only a small number of random linear measurements, with the number of measurements proportional to the sparsity level of the signal. Thus, in both standard compression and CS applications, finding a basis that most sparsely represents the signals is a problem of common interest. In this paper, we consider the application of CS to video signals in order to lessen the sensing and compression burdens these high-dimensional signals impose on imaging systems.

There are numerous applications where CS video systems could be helpful. Standard video capture systems require a complete set of samples to be obtained for each frame, at which point a compression algorithm may be applied to exploit spatial and temporal redundancy. In some applications, such as imaging at non-visible (e.g., infrared) wavelengths, it may be difficult or expensive to obtain these raw samples. In other applications, such as multi-image capture in camera networks, implementing a compression algorithm may itself be a challenge. We argue that these burdens may be reduced by using compressive imaging hardware (such as the "single-pixel camera" [3]) where random measurements are collected independently from each frame and no additional compression protocol is needed. In exchange, the challenge of implementing such a system comes in developing efficient sparsifying representations and the corresponding algorithms for video recovery from random measurements.

As discussed above, an efficient representation of video signals must effectively remove spatial and temporal redundancies. In the long literature of standard video compression [4], a variety of methods have been proposed to exploit these redundancies. One common approach combines motion compensation and estimation [5] algorithms with image compression techniques. While some of these central ideas can be absorbed into the CS framework, there is an important challenge that we must address. Unlike the standard video compression problem where the frames of the video are explicitly available to perform motion estimation, in CS only random measurements of the underlying video are available. We are faced with a chicken-or-egg problem: Given the video frames, we could estimate the motion; or given the motion we could better estimate the frames themselves.

In this paper, we propose a multiscale framework for reconstruction that involves iterating between motion estimation and sparsity-based reconstruction of the frames themselves. Our representation framework is built around the LIMAT [6] method for standard video compression, in which motion compensation is used to improve sparsity in the three-dimensional (3D) wavelet domain. Section 2 discusses the necessary background topics. Section 3 describes our algorithm, and Sec. 4 presents results and compares our algorithm to existing approaches. We conclude in Sec. 5.

2. BACKGROUND

2.1. Compressive Sensing (CS)

Consider a finite dimensional signal $x \in \mathbb{R}^N$. A typical CS acquisition scenario correlates x against M different test

[*]This research was partially supported by DARPA Grant HR0011-08-1-0078. Email: jaeypark@umich.edu, mwakin@mines.edu.

functions, $\phi_m \in \mathbb{R}^N$ where $m = 1, 2, ..., M$, to obtain the measurements, $y \in \mathbb{R}^M$ with $M \ll N$. This is written as $y = \Phi x$, where $\Phi \in \mathbb{R}^{M \times N}$ is the measurement matrix with ϕ_m^T as its rows. With $M \ll N$, the reconstruction problem of x from y is ill-posed. However, most naturally occurring signals are *sparse* or *compressible*. This means when x is represented in an appropriate $N \times N$ basis Ψ such that $x = \Psi\alpha$, α has only a few significant coefficients. Let $x_K = \Psi\alpha_K$ denote the K-largest term approximation, where α_K is obtained by keeping only the K-largest entries in absolute value of α and setting the rest to zero. Then, the norm of $x - x_K$ is small or even zero for the classes of signals described above. One standard reconstruction procedure is to solve

$$\min_{\alpha} \|\alpha\|_1 \quad \text{s.t.} \quad y = \Phi\Psi\alpha. \tag{1}$$

Under certain conditions on Φ [2], solving (1) gives a reconstruction that obeys $\|x^* - x\|_1 \leq C\|x - x_K\|_1$, where C is a well-behaved constant, $x^* = \Psi\alpha^*$, and α^* is the result of solving (1). From this we can infer that given a fixed number of random measurements, the accuracy of reconstruction depends highly on the compressibility of the underlying signal. Hence, for the most accurate reconstruction, one needs to choose Ψ where x is most compressible.

2.2. Previous CS video reconstruction methods

Several methods for CS video reconstruction have been proposed, each relying on a different sparsifying transform Ψ. One natural approach is to recover each frame independently using the 2D discrete wavelet transform (2D-DWT) for Ψ on each frame [3]. An alternative approach is to use the 3D-DWT for Ψ and reconstruct the entire video all at once [3]. Yet another approach, termed "C_n" and proposed for compressive coded aperture video reconstruction [7], relies on small inter-frame differences together with a spatial 2D-DWT to produce a sparse representation of the underlying video.

A common difficulty faced by the methods above is how to effectively remove temporal redundancies in the presence of high frequency or global motion in the video. Frame-by-frame reconstruction does not consider any temporal correlation and relies solely on the sparsity of each frame. Using a fixed temporal filter or taking inter-frame differences is not sufficient to fully remove the vast amount of temporal redundancies, e.g., when large objects move several pixels between adjacent frames or when the entire scene undergoes translational motion. Motivated by this, we seek to exploit a framework proposed in the standard video compression literature that uses motion compensation and estimation along the temporal dimension to reduce the temporal redundancies.

2.3. LIMAT

The LIMAT [6] algorithm for video compression uses *lifting* [8] as its basic framework to build a fully invertible transform. The lifting scheme is a simple construction of second generation wavelets, which can adapt to the various characteristics present in the signal. By applying motion-compensated lifting steps to implement the temporal wavelet transform, LIMAT can adaptively account for motion within the video to effectively exploit the temporal correlation. Spatial correlations are subsequently exploited using the 2D-DWT.

Let us denote the k^{th} frame of the n frame video sequence by x_k, where $k \in \{1, 2, \ldots, n\}$. The lifting transform partitions the video into even frames $\{x_{2k}\}$ and odd frames $\{x_{2k+1}\}$ and attempts to predict the odd frames from the even ones using a forward motion compensation operator. This operator, denoted $\mathcal{F}$, takes as input one even frame and a collection of motion vectors denoted v_f that describe the anticipated motion of objects between that frame and its neighbor. For example, suppose that x_{2k} and x_{2k+1} differ by a 3-pixel shift that is captured precisely in v_f; then as a result $x_{2k+1} = \mathcal{F}(x_{2k}, v_f)$ exactly. Applying this prediction to each pair of frames and keeping only the prediction errors, we obtain a sequence of highpass residual detail frames (see (2) below). The prediction step is followed by an update step that uses an analogous backward motion compensation operator denoted $\mathcal{B}$ and motion vectors v_b. The combined lifting steps

$$h_k = x_{2k+1} - \mathcal{F}(x_{2k}, v_f) \tag{2}$$

$$l_k = x_{2k} + \frac{1}{2}\mathcal{B}(h_k, v_b) \tag{3}$$

produce an invertible transform between the original video and the lowpass $\{l_k\}$ and highpass $\{h_k\}$ coefficients. For maximum compression, the lifting steps can be iterated on pairs of the lowpass frames until there remains only one. Ideally, with perfect motion compensation, the $n - 1$ highpass frames will consist only of zeros, leaving only one frame of nonzero lowpass coefficients, and making the sequence significantly more compressible. As a final step, it is customary to apply the 2D-DWT to each lowpass and highpass frame to exploit any remaining spatial correlations.

In our proposed algorithm, we use block matching (BM) to estimate motion between a pair of frames. The BM algorithm divides the reference frame into non-overlapping blocks. For each block in the reference frame the most similar block of equal size in the destination frame is found and the relative location is stored as a motion vector. There are several possible similarity measures such as the l_2 norm.

We note that in order to use LIMAT in solving (1), motion information must be supplied prior to solving the l_1 minimization problem. Once we have the estimates of the motion vectors, the lifting steps (2) and (3) can be defined, which in turn gives us a transform Ψ. Unfortunately, extracting motion information prior to reconstruction is highly non-trivial.

3. MULTISCALE RECONSTRUCTION

We consider a video sequence containing n frames $\{x_k\}_{k=1}^n$, each of height h and width w; that is, $x_k \in \mathbb{R}^{hw}$ for each

$k = 1, 2, \ldots, n$. We suppose that compressive measurements of each frame are collected according to a protocol that we describe in Sec. 3.1; such measurements could be acquired using CS imaging hardware such as the "single-pixel camera" [3]. In Sec. 3.2, we propose an algorithm for recovering the original video sequence from the complete ensemble of compressive measurements. Our algorithm employs the LIMAT framework to exploit motion information and remove temporal redundancies. To overcome the chicken-or-egg problem, we propose an iterative *multiscale* framework, reconstructing successively finer resolution approximations to each frame and alternately using these approximations to estimate the motion. This approach also exploits the limited complexity of motion information at coarse spatial resolutions.

At each scale of the algorithm, we seek to reconstruct a spatially lowpass-filtered and decimated version of each video frame. Letting $j \in \{1, 2, \ldots, \log_2(n)\}$ denote the scale, each decimated frame has size $d(j) = h/(n/2^j) \times w/(n/2^j)$. For each frame $k = 1, 2, \ldots, n$, we define

$$L_{j,k} := R_j W_{2D}^{s(j)} x_k \ \in \mathbb{R}^{d(j)}, \tag{4}$$

where $W_{2D}^{s(j)}$ represents the $s(j) = \log_2(n) - j$ level 2D discrete Haar wavelet transform (2D-DHWT), and R_j represents the $d(j) \times hw$ linear restriction operator, which outputs the $d(j)$ scaling coefficients and omits the remaining $hw - d(j)$ wavelet coefficients. We denote $L_j \in \mathbb{R}^{nd(j)}$ as the stacked vector of all $L_{j,k}$. We note that, at the finest scale $j = \log_2(n)$, we simply have $L_{j,k} = x_k$, and we also note the multiscale relationship:

$$L_{j-i,k} = R_{j-i} W_{2D}^{i} L_{j,k}, \ i = 1, 2, \ldots, j-1. \tag{5}$$

3.1. Measurement protocol

The encoder we propose requires only frame-by-frame random measurements of each $L_{j,k}$. Letting $M(j)$ denote the desired number of measurements from each frame at scale j, we let $\Phi_{j,k}$ denote an $M(j) \times d(j)$ random measurement matrix for each $k = 1, 2, \ldots, n$ and define the random measurements $y_{j,k} \in \mathbb{R}^{M(j)}$ as follows:

$$y_{j,k} = \Phi_{j,k} L_{j,k} = \Phi_{j,k} R_j W_{2D}^{s(j)} x_k. \tag{6}$$

We note that each measurement collected is a linear function of a single video frame.

Finally, let us denote $y_j \in \mathbb{R}^{nM(j)}$ as the stacked vector of all $y_{j,k}$, and Φ_j as the $nM(j) \times nd(j)$ random matrix with all $\Phi_{j,k}$ as its block diagonals; we then have $y_j = \Phi_j L_j$.

3.2. Reconstruction algorithm

Our proposed decoder begins at the coarsest scale ($j = 1$) and proceeds to finer scales. At each scale j, the decoder can be divided into two stages. In the first stage, we reconstruct L_j using motion vectors estimated at coarser scales, and in the second stage we use the reconstructed L_j to estimate motion vectors (denoted v_j). To reconstruct L_j at each scale, we solve an ℓ_1 problem (1) by seeking sparsity in the motion-compensated wavelet basis provided by LIMAT (denoted $\Psi_j(v_{j-1})$). Due to (5), the measurements available for reconstructing L_j consist not only of y_j but also of $y_{j'}$ for all $j' < j$. Letting

$$Y_j := \begin{pmatrix} y_j \\ \vdots \\ y_2 \\ y_1 \end{pmatrix} \quad \text{and} \quad \Omega_j := \begin{pmatrix} \Phi_j \\ \vdots \\ \Phi_2 R_2 W_{2D}^{j-2} \\ \Phi_1 R_1 W_{2D}^{j-1} \end{pmatrix},$$

we have $Y_j = \Omega_j L_j = \Omega_j \Psi_j(v_{j-1}) \alpha_j$, where α_j represents the vector of expansion coefficients for L_j in the motion-compensated LIMAT basis obtained from the most recently estimated motion vectors v_{j-1}. The algorithm can be summarized as follows:

Algorithm 1: Reconstruction procedure

Input: $\{Y_j\}_{j=1}^{\log_2(n)}$ and $\{\Omega_j\}_{j=1}^{\log_2(n)}$
Output: $L_{\log_2(n)}$
Initialization : v_0 = *zero-motion vectors* ;
for $j \leftarrow 1$ **to** $\log_2(n)$ **do**
  Solve $\min_{\alpha_j} \|\alpha_j\|_1$ s.t. $Y_j = \Omega_j \Psi_j(v_{j-1}) \alpha_j$;
  $L_j = \Psi_j(v_{j-1}) \alpha_j$;
  Estimate v_j using L_j via BM algorithm
end

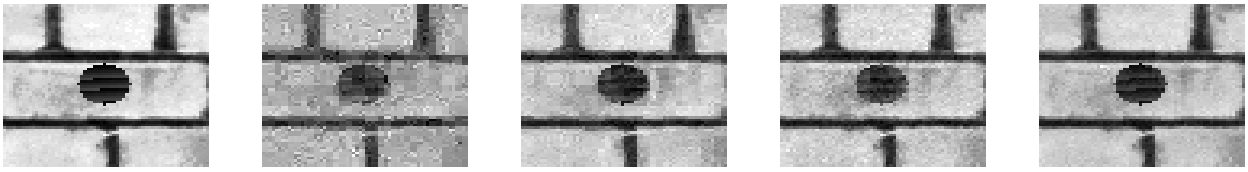
Because we cannot estimate motion before recovering an estimate of the frames, our assumption in Step 1 reduces the motion compensation operators $\mathcal{F}$ and $\mathcal{B}$ to the identity.

4. RESULTS

In this section we compare the proposed algorithm to existing algorithms such as frame-by-frame reconstruction, 3D-DHWT, and the C_n reconstruction method proposed in [7].

We present results for two different test videos. The first video is synthetic and consists of a stationary background with a textured circle moving horizontally at a constant speed of 4 pixels/frame. There are 8 frames, each of size 64×64; Fig. 1(a) shows the fifth frame of this video. The second video is an edited version of the standard "Coastguard" test video: we select every other frame starting with the first frame (16 in total) and crop each frame to size 128×128. Figure 2(a) shows the tenth frame of this video zoomed in on a 100×100 region. Motion in this video is more complex, with the water having changes in reflection and surface patterns, the background land moving on average 2 pixels/frame to the right, the smaller boat moving slightly between frames, and the larger boat moving on average 5 pixels/frame to the right.

For the synthetic video the total number of pixels $N = 32678$. Due to the increasing complexity of the inverse problem at each scale, we use measurement rates $M(1) = 240$,



(a) Original fifth frame (b) frame-by-frame (c) 3D-DHWT (d) C_n with $n = 8$ (e) Proposed algorithm

Fig. 1. *Reconstructed synthetic video: (a) original, (b)* $MSE = 724.08$*, (c)* $MSE = 206.37$*, (d)* $MSE = 137.83$*, (e)* $MSE = 56.93$*.*

(a) Original tenth frame (b) frame-by-frame (c) 3D-DHWT (d) C_n with $n = 16$ (e) Proposed algorithm

Fig. 2. *Reconstructed "Coastguard" video: (a) original, (b)* $MSE = 239.7$*, (c)* $MSE = 194.5$*, (d)* $MSE = 233.8$*, (e)* $MSE = 147.9$*.*

$M(2) = 400$, and $M(3) = 620$; in total $M = 10080 \approx 0.3N$. For the "Coastguard" video $N = 262144$ and we use measurement rates $M(1) = 256$, $M(2) = 528$, $M(3) = 892$, and $M(4) = 1598$; in total $M = 52384 \approx 0.2N$. Figures 1(b)-(e) show the fifth frame of the reconstructed synthetic video, and Figs. 2(b)-(e) show the tenth frame of the reconstructed "Coastguard" video, zoomed in on a 100×100 region. In each case, the MSE shown is for the entire reconstructed video; for both videos, our algorithm gives the best result in terms of MSE. To be clear, the methods we test differ not in the measurements they use (and therefore not in the bitrate if those measurements were quantized) but instead differ only in the model and algorithm used for reconstruction.

The test on the synthetic video shows the importance of motion compensation. The quality of frame-by-frame reconstruction depends solely on the sparsity of the individual frames. However, because each frame is not sufficiently sparse in the 2D-DHWT domain, this method is inferior to those that account for temporal correlation. Although the MSE values do not differ as dramatically in the second video as in the first, the more complex video is best reconstructed by our motion-compensated method. In both the C_n and 3D-DHWT methods, a lack of sufficient sparsity causes more significant artifacts in the reconstructed video; in the case of the C_n algorithm the artifacts tend to propagate and worsen toward the end of the sequence.

5. CONCLUSIONS

We have proposed a new algorithm for CS video reconstruction. Unlike previous approaches, our algorithm directly compensates for motion between frames to achieve a sparse representation. Our algorithm requires only frame-by-frame random measurements, which can be taken with proposed CS imaging hardware. Test results show our proposed algorithm outperforms others in terms of MSE, but at present it is also the most computationally demanding of the methods tested in Sec. 4. (At the other extreme, frame-by-frame reconstruction requires the least computation but yields the highest MSE.) In ongoing work, we are examining computationally efficient alternatives to the large ℓ_1 problem at fine scales.

For single-view imaging, our CS-based algorithm will not necessarily provide bit-rate savings when compared with traditional MPEG compression. However, the ability to capture a video using few measurements may be useful in applications where is it difficult or expensive to obtain raw samples. Another interesting application of the proposed algorithm is in multi-view imaging. In such a system, multi-camera arrays are used to capture a scene from several different angles. Several limitations—in transmission bandwidth, power, storage, and so on—could be overcome with the application of CS [9]. The proposed algorithm could be applied by treating each signal captured by each camera as a frame of a video sequence. This remains a subject of ongoing work and experimentation.

6. REFERENCES

[1] D.L. Donoho, "Compressed sensing," *IEEE Trans. Inform. Theory*, vol. 52, no. 4, pp. 1289–1306, April 2006.

[2] E. Candès, "Compressive sampling," in *Proc. Int. Congress Math.*, Madrid, Spain, August 2006, vol. 3, pp. 1433–1452.

[3] M.B. Wakin, J.N. Laska, M.F. Duarte, D. Baron, S. Sarvotham, D. Takhar, K.F. Kelly, and R.G. Baraniuk, "Compressive imaging for video representation and coding," in *Picture Coding Symposium - PCS 2006*, Beijing, China, April 2006.

[4] T. Wiegand, G.J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circ. Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, July 2003.

[5] J. Jain and A. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Trans. Commun.*, vol. 29, no. 12, pp. 1799–1808, Dec 1981.

[6] A. Secker and D. Taubman, "Lifting-based invertible motion adaptive transform framework for highly scalable video compression," *IEEE Trans. Image Process.*, vol. 12, no. 12, pp. 1530–1542, Dec. 2003.

[7] R. Marcia and R. Willet, "Compressive coded aperture video reconstruction," in *Proc. European Signal Processing Conf. (EUSIPCO)*, 2008.

[8] I. Daubechies and W. Sweldens, "Factoring wavelet transforms into lifting steps," *J. Fourier Anal. Appl.*, vol. 4, no. 3, pp. 245–267, 1998.

[9] D. Daron, M. B. Wakin, M. Duarte, S. Sarvotham, and R. G. Baraniuk, "Distributed compressed sensing," Rice University Technical Report TREE-0612, Nov 2006.