LOW-COMPLEXITY VIDEO COMPRESSION AND COMPRESSIVE SENSING

M. Salman Asif [g], Felix Fernandes [s], Justin Romberg [g]

[g] School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA
[s] Dallas Technology Laboratory, Samsung Telecom America, Richardson, TX

ABSTRACT

Compressive sensing (CS) provides a general signal acquisition framework that enables the reconstruction of sparse signals from a small number of linear measurements. To reduce video-encoder complexity, we present a CS-based video compression scheme. Modern video-encoder complexity arises mainly from the transform-coding and motion-estimation blocks. In our proposed scheme, we eliminate these blocks from the encoder, which achieves compression by merely taking a few linear measurements of each image in a video sequence. To guarantee stable reconstruction of the video sequence from only a few measurements, the decoder must effectively exploit the inherent spatial and temporal redundancies in a video sequence. To leverage these redundancies, we consider a motion-adaptive linear dynamical model for videos. Recovery process involves solving an ℓ_1-regularized optimization problem, which iteratively updates estimates for the video frames and motion within adjacent frames.

1. INTRODUCTION

Current video coding technology has developed assuming that a high-complexity encoder in a broadcast tower would support millions of low-complexity decoders in receiving devices. However, with the proliferation of inexpensive video recording devices, such as camcorders and cellphones, user-generated content has become commonplace. Therefore, there is a need for low-complexity encoding technology that can be deployed in these low-cost, low-power devices [1]. Because power consumption is proportional to the encoder complexity, current high-complexity encoders consume much power to provide high compression ratios. Therefore, to increase battery life in mobile devices, a low-complexity encoder with good coding efficiency is highly desirable.

In this paper, we discuss a general video compression framework, where the encoder is very simple, and most of the computational complexity is shifted to the decoder. Note that decoders are usually located in mains-connected devices such as set-top boxes, TVs and computers; hence complexity and consequently higher power consumption are tolerable. The main complexity in standardized video encoders arises from the motion estimation and transform-coding blocks. To reduce complexity, we eliminate these blocks in our proposed video encoder. We assume that the encoder compresses a video sequence by taking a small number of linear measurements for each image in the sequence, either by processing each reading on the conventional CCD sensor or directly observing the scene with modified imaging hardware [2]. At the decoder, we reconstruct the video sequence from the compressed measurements by exploiting the inherent spatial and temporal structure in the video signals. We model the measurements and the underlying video sequence in the form of a linear dynamical system, where adjacent frames are related to each other via (unknown) inter-frame motion. To aide the recovery process, we assume that the images in the video sequence and the motion-compensated differences have sparse representation. As we discuss later in detail, our proposed encoding and decoding schemes leverage ideas from standard video coding and compressive sensing.

Current video coding schemes achieve compression using a high-complexity encoder that exploits the spatial and temporal structure in the video sequence [3]. At the encoder, typically we first estimate motion between adjacent frames (e.g., using block-matching), and then we use transform coding (e.g., DCT or wavelets) on the inter-frame motion-compensated differences and a reference frame. Motion estimation and transform coding blocks often dominate the computational complexity of the encoder. The decoder, in contrast, is much simpler. Its only task is to use the inter-frame motion information (transmitted by the encoder) to combine the reference frame with the motion-compensated residuals to reconstruct the video sequence.

Compressive sensing (CS) provides a signal acquisition framework in which only a small number of (non-adaptive) linear measurements are required to recover a (structured) sparse signal [4]. Conventional compression schemes often require fully sampled signals at the encoder, and they compress signals by keeping a small amount of data after some processing (e.g., motion estimation and transform coding in MPEG and H.264 video coding), and discarding the rest. In contrast, CS combines compression with acquisition by acquiring only as many measurements as would be necessary for the signal recovery. The number of measurements required for the signal recovery depends on the sparse structure of the signal (also related to the degrees of freedom in the signal). This combined acquisition and compression reduces the burden on the sensing devices in two ways: 1) full signal acquisition is not required and 2) any additional compression is not required. Furthermore, the computational burden shifts to the decoder. It is the decoder's task to reconstruct the signal from the compressed measurements, using any available information about the signal structure. CS decoders typically recover signals by solving an optimization problem that promotes sparsity in the signals while maintaining fidelity to the measurements.

To recover video from compressed measurements, our proposed decoder exploits both spatial and temporal structures in the video signal. The inter-frame motion provides a very good model for the temporal structure. However, in our compression framework, the inter-frame motion is not readily available at the decoder. Therefore, we use a two-step approach to iteratively update the estimates for the images in the video sequence and the inter-frame motion. At every iteration, we estimate images in the video sequence using any available motion (or temporal) information, and then refine the inter-frame motion estimates using the recovered images. Similar two-step approach has been used in [5, 6].

2. BACKGROUND

2.1. Video coding principles

A typical video encoder divides a video sequence into disjoint groups of T frames. Out of the T frames in each group, one frame is designated to be the I (intra-coded) frame and the rest are designated to be P (predictive) frames (or some times B (bi-predictive) frames too). The I-frame is encoded as a static image. P-frames are encoded in terms of motion-compensated residuals between the original P-frames and their respective motion-compensated predictions from the neighboring frames. Since adjacent frames in a video sequence are very similar to each other, the prediction error is usually very sparse and allows efficient encoding. Inter-frame motion between pairs of two images is typically estimated using block-matching. To predict an image A from an image B, block matching first divides A into non-overlapping blocks of equal size (e.g., 8x8 or 16x16), and then finds the closest matching block of same size in B for every block in A. The motion-compensated predicted image A is constructed by replacing each block in A with its closest matching block from B. The relative locations of the blocks are stored in the form of motion vectors. I-frame, motion-compensated residuals, and the associated motion vectors constitute the compressed data for a group of T frames.

2.2. Compressive sensing (CS)

The CS theory suggests that a (structured) sparse signal can be recovered from a small number of linear, incoherent measurements. Consider a signal of interest $\mathbf{x} \in \mathbb{R}^N$ that is measured as

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{e}, \tag{1}$$

where $\mathbf{A}$ is an $M \times N$ measurement matrix with $M \ll N$, and $\mathbf{e}$ denotes noise in the measurements. In the CS framework, signal recovery from the underdetermined system in (1) is possible if the signal $\mathbf{x}$ has sparse representation in some known basis Ψ (i.e., $\alpha = \Psi^T\mathbf{x}$ is a sparse vector) and the measurement matrix $\mathbf{A}$ satisfies certain incoherence conditions. A typical ℓ_1-regularized optimization problem used for the recovery can be written as

$$\text{minimize } \tau\|\Psi^T\mathbf{x}\|_1 + \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2, \tag{2}$$

where $\tau > 0$ is a regularization parameter. The ℓ_1 regularization term in (2) promotes Ψ-domain sparsity in the solution and the ℓ_2 term keeps the solution close to the measurements. Note that CS can be viewed as a universal encoding scheme that is not tied to any specific representation basis. We can use any sparse basis Ψ that gives the best signal reconstruction for the given set of measurements. For instance, if $\mathbf{x}$ is an image, we can apply ℓ_1 regularization in the spatial domain by selecting Ψ as a wavelet transform or a finite difference (total variation) transform [4]. Similarly, if $\mathbf{x}$ is a video sequence, we have the freedom to choose whichever Ψ gives the best results. One possibility is to recover each image in the video sequence independently, using only the spatial regularization; however, this does not take any advantage of the inherent temporal structure in the video signals. Another approach is to assume that the inter-frame differences are sparse, and add an ℓ_1 regularization term for inter-frame differences in the recovery problem [7]. However, the temporal variations in video sequences are not fully captured by the inter-frame differences. Inter-frame motion provides a much better representation for the temporal variations in a video sequence.

In the next section, we discuss our proposed optimization problem for the video recovery, which uses an inter-frame motion based model for the temporal variations, and uses ℓ_1 regularization in both spatial and temporal directions.

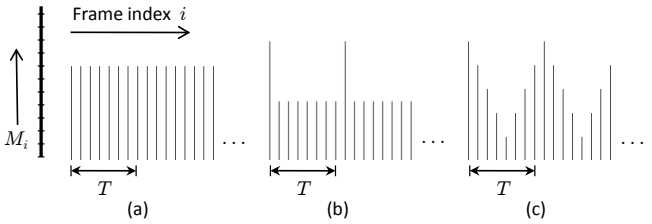

Fig. 1: Distribution of measurements in a group of T frames. (a) Uniform distribution. (b) One boundary frame gets more measurements (analogous to I and P frames). (c) Custom distribution.

3. VIDEO COMPRESSIVE SENSING

3.1. Encoder design

Our goal is to have a simple encoder with low computational complexity. Therefore, we assume just one task from the encoder; that is, it compresses each image in a video sequence by taking a small number of linear measurements. This task can be performed either using some modified imaging hardware (e.g., single-pixel camera [8]) without explicitly capturing and storing the original images or by post-processing the full-resolution acquired images.

Consider a video sequence $\mathbf{x}_1, \mathbf{x}_2, \ldots$, where each $\mathbf{x}_i \in \mathbb{R}^N$ is an N-pixel image in the video sequence. The encoder generates a set of measurements $\mathbf{y}_i = \mathbf{A}_i\mathbf{x}_i + \mathbf{e}_i$ for every image $\mathbf{x}_i$, where $\mathbf{A}_i$ denotes the measurement matrix of size $M_i \times N$ for the ith image, and $\mathbf{e}_i$ denotes noise in the measurements. The ratio of N and M_i is the compression rate for the ith image.

One important feature we borrow from traditional video coding literature in our encoder design is the flexibility to compress images in the video sequence with variable compression rate. Suppose our desired compression rate allows M measurements per T frames. We allow the encoder to distribute the available measurement count to T frames in any desired fashion. Figure 1 depicts three possible choices for measurement distribution in a group of T frames: (a) Every image gets an equal number of measurements. (b) One boundary image in every group of T images gets significantly more measurements than the rest. This scheme is analogous to the I-frame and P-frames concept in the standard video coding. (c) An arbitrary non-uniform distribution of measurements. The motivation behind using non-uniform distribution of measurements is that a few good quality images help a lot to raise quality of the heavily compressed neighboring images during the reconstruction. In the case of non-uniform measurements, we define the compression ratio as TN/M, where M is the total number of measurements utilized by T frames.

The quality of reconstruction depends on two things: the number of measurements (more the better) and the type of measurements. We do not make any assumption on the specific type of measurements. A general rule for CS applications is to use the measurements that are spread out in the transform domain where the signal of interest is sparse [9]. Some commonly used measurements for images in the CS framework include subsampled DCT, subsampled noiselets, and random convolution [2].

3.2. Decoder design

3.2.1. Motion-adaptive dynamical model

To model the temporal variations, we represent video frames in the form of a linear dynamical system, where each frame is related to its immediate neighbors via inter-frame motion, as depicted in Fig. 2. The following linear system gives a combined model for the linear measurements and inter-frame relationship:

$$\mathbf{y}_i = \mathbf{A}_i\mathbf{x}_i + \mathbf{e}_i \tag{3a}$$

$$\mathbf{x}_i = \mathbf{F}_{i-1}\mathbf{x}_{i-1} + \mathbf{f}_i \tag{3b}$$

$$\mathbf{x}_i = \mathbf{B}_{i+1}\mathbf{x}_{i+1} + \mathbf{b}_i, \tag{3c}$$

where $\mathbf{F}_{i-1}$ and $\mathbf{B}_{i+1}$ denote the forward and the backward motion operators, and $\mathbf{f}_i$ and $\mathbf{b}_i$ denote the forward and the backward motion-compensated residuals, respectively. Motion operators can be viewed as interpolation operators that move the pixel values according to the inter-frame motion.

3.2.2. Recovery algorithm

In our proposed recovery algorithm, we jointly recover a group of images, following the linear dynamical model in (3). It helps to select the groups at the decoder following the measurement distribution at the encoder. For instance, if the encoder uses measurement distribution in Fig. 1(b), we divide the sequence at the decoder into overlapping groups of $T+1$ images, where the two boundary frames in each group have more measurements, and they are shared by the adjacent groups. Such a group division provides one "free" additional high-quality frame in each group, which can be very beneficial with motion regularization in both forward and backward directions.

Individual images in a natural video sequence have sparse representation in spatial domain; for example, a wavelet transform or total variation. The motion-compensated residual images $\mathbf{f}_i = \mathbf{F}_{i-1}\mathbf{x}_{i-1} - \mathbf{x}_i$ and $\mathbf{b}_i = \mathbf{B}_{i+1}\mathbf{x}_{i+1} - \mathbf{x}_i$ also exhibit sparsity, either in the pixel domain or with some spatial transform. We exploit the sparsity of original images and motion-compensated residuals in the recovery process. Note that $\mathbf{F}_{i-1}$ and $\mathbf{B}_{i+1}$ require information about motion between $\mathbf{x}_i$ and its immediate neighbors $\mathbf{x}_{i-1}$ and $\mathbf{x}_{i+1}$, respectively. To estimate inter-frame motion, we need explicit images; whereas we only have compressed measurements at the decoder. Therefore, we adopt an iterative approach for reconstruction, where we alternate between estimating video frames using the available motion information, and using the estimated video frames to refine the motion estimate. Video recovery algorithms with similar alternating motion update principles have also appeared in [5, 6].

Our recovery algorithm consists of the following two-step iterative procedure: 1) Initialization. 2) Motion adaptation.

Initialization: Solve the following ℓ_1 regularization problem to get the initial frame estimates:

$$\text{minimize} \sum_i \|\mathbf{A}_i\mathbf{x}_i - \mathbf{y}_i\|_2^2 + \tau\|\mathbf{x}_i\|_\Psi + \lambda\|\mathbf{x}_{i-1} - \mathbf{x}_i\|_\Psi, \tag{4}$$

where $\|\cdot\|_\Psi$ denotes ℓ_1 regularization term to promote spatial sparsity in the Ψ domain (i.e., $\|\mathbf{z}\|_\Psi = \|\Psi^T\mathbf{z}\|_1$); for example, wavelets or total-variation. The first term promotes sparsity in the spatial transform of each image, and the second term promotes sparsity in the inter-frame difference.

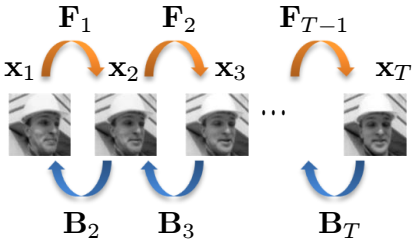

Fig. 2: Bi-directional inter-frame motion interpretation.

Motion adaptation: This step can be further divided into two intermediate steps, and can be repeated multiple times to improve the reconstruction quality:

i. *Motion estimation:* Use reconstructed frames to estimate/refine the inter-frame motion, and define/update the forward and backward motion operators $\mathbf{F}_i$ and $\mathbf{B}_i$, for all i.

ii. *Motion compensation:* Solve the following optimization problem for the dynamical system in (3):

$$\text{minimize} \sum_i \|\mathbf{A}_i\mathbf{x}_i - \mathbf{y}_i\|_2^2 + \tau\|\mathbf{x}_i\|_\Psi + \ldots$$
$$+ \alpha\|\mathbf{F}_{i-1}\mathbf{x}_{i-1} - \mathbf{x}_i\|_\Psi + \beta\|\mathbf{B}_{i+1}\mathbf{x}_{i+1} - \mathbf{x}_i\|_\Psi. \tag{5}$$

The regularization parameters: τ, λ, α, and β can be adapted according to the problem at hand. We found it useful to start α and β with a small value at the first iteration, and increase them as the motion estimate improves. We have purposely written all the regularization terms as a general norm $\|\cdot\|_\Psi$ to emphasize that any suitable transform Ψ can be used with the ℓ_1 regularization. Furthermore, we do not restrict ourselves to any particular motion estimation scheme either. However, since we estimate motion from the reconstructed images, and not the original images, motion estimation scheme should be robust to noise and other artifacts. In our experiments, we found that block matching algorithms do not perform very well. We found phase-based motion estimation schemes to be more useful [10]. Optical flow based schemes work well too [11].

4. EXPERIMENTS AND RESULTS

To evaluate performance of our proposed compression scheme we performed various experiments on four standard test sequences: coastguard, container, foreman, and hall. Figure 3 presents one image from each of the four sequences. The coastguard sequence contains the most abrupt temporal variations among the four, foreman ranks second, followed by hall, and the container sequence has the least and the slowest scene variations. In all our experiments, we used 128x128 center portion of the first 129 frames of the four sequences.

In our experiments, we evaluated the performance of our proposed recovery scheme with non-adaptive, linear measurements. We compared results with two conventional low-complexity encoders: standard JPEG[1] and linear DCT approximation. Our experiment setup is as follows: We divided 129 frames into 16 overlapping groups of 9 frames, where the frames at indices $9, 17, \ldots, 121$ are

[1]For a fair comparison with CS, we do not use entropy coding with JPEG.

Fig. 3: Snapshots of original test sequences.

shared by two adjacent groups.We encoded each group according to the scheme in Fig. 1(b), where the number of measurement for the boundary frames were double the number of measurements for the rest. Linear measurements for each frame comprised of two parts: a 16x16 block of scaling coefficients from discrete wavelet transform (using Daubechies 4 filters), while the remaining measurements consist of subsampled noiselet coefficients. Every group of 9 images was then reconstructed from compressed measurements using the recovery method outlined in Sec. 3.2.2. We used 2-D dual-tree complex wavelet transform [12] as the sparsity basis Ψ for spatial ℓ_1 regularization in (4) and (5). The dual-tree wavelet transform is a nearly shift-invariant, overcomplete transform, with good directionality properties, and works significantly better than standard orthogonal wavelets. We estimated motion using phases of the subband coefficients from the complex dual-tree wavelets, using the hierarchical method described in [10]. We solved the optimization problems in (4) and (5) using the ℓ_1-analysis formulation in the NESTA toolbox [13]. All the experiments were performed on the PACE cluster operated by OIT at Georgia Tech.

The recovery results for the four sequences at different compression rates are presented in Fig. 4. The performance curves plot average peak signal to noise ratio (PSNR) of all the reconstructed images at the given compression ratio. CS-MC (solid blue line with $*$ marker) represents the results for our proposed motion-compensation based recovery method after 3 motion-adaptation iterations. CS-DIFF (broken blue line with $+$ marker) represents results for the initial reconstruction with frame-difference. CS (broken blue line $\times$ marker) represents results for frame-by-frame reconstruction (without frame-difference in (4)). We compared recovery results with JPEG compression, which compressed every image by thresholding its block-DCT coefficients using the so-called quantization mask. The compression ratio for JPEG at any value of quality factor was calculated as the ratio of N and the number of nonzero block-DCT coefficients. The results are presented as qJPEG (broken magenta line with o marker). We also recorded results for linear DCT approximation, which compresses an image by keeping only low frequency DCT coefficients, selected in a predefined zigzag order; the results are presented as ldct (dashed red line with lower-triangle marker).

As we can see in Fig. 4, PSNR quality of the reconstructed videos increases significantly by adding temporal regularization in the recovery process (see the improvement of CS-MC (or CS-DIFF) over CS). Furthermore, motion-compensated reconstruction (CS-MC) outperforms qJPEG, even though qJPEG is a data-adaptive compression scheme. qJPEG comes close to CS-MC in performance only for the coastguard sequence, which has quite complex motion, especially around frame 70.

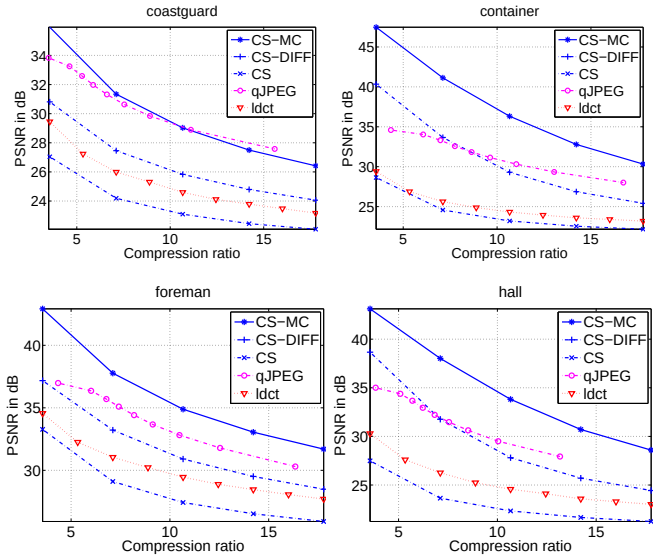

Fig. 4: Linear, non-adaptive measurements: wavelet scaling + noiselet coefficients. Motion-compensated CS (CS-MC), frame-difference (CS-DIFF), frame-by-frame (CS). Comparison with JPEG (qJPEG) and linear DCT (ldct) approximation.

5. REFERENCES

[1] R. Puri, A. Majumdar, and K. Ramchandran, "PRISM: a video coding paradigm with motion estimation at the decoder," *IEEE Transactions on Image Processing*, vol. 16, pp. 2436–2448, Oct. 2007.

[2] J. Romberg, "Imaging via Compressive Sampling [Introduction to compressive sampling and recovery via convex programming]," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 14–20, March 2008.

[3] G. J. Sullivan and T. Wiegand, "Video compression - from concepts to the H.264/AVC standard," *Proceedings of the IEEE*, vol. 93, pp. 18–31, Jan. 2005.

[4] E. Candès, "Compressive sampling," *Proceedings of the International Congress of Mathematicians, Madrid, Spain*, vol. 3, pp. 1433–1452, 2006.

[5] J. Park and M. Wakin, "A multiscale framework for compressive sensing of video," in *Picture Coding Symposium*, p. 1–4, 2009.

[6] D. Reddy, A. Veeraraghavan, and R. Chellappa, "P2C2: programmable pixel compressive camera for high speed imaging," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 329–336, June 2011.

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

[8] M. Wakin, J. Laska, M. Duarte, D. Baron, S. Sarvotham, D. Takhar, K. Kelly, and R. Baraniuk, "An architecture for compressive imaging," *IEEE International Conference on Image Processing*, pp. 1273–1276, 8-11 Oct. 2006.

[9] E. Candès and J. Romberg, "Sparsity and incoherence in compressive sampling," *Inverse Problems*, vol. 23, no. 3, pp. 969–985, 2007.

[10] J. Magarey and N. Kingsbury, "Motion estimation using a complex-valued wavelet transform," *IEEE Transactions on Signal Processing*, vol. 46, no. 4, pp. 1069–1084, 1998.

[11] C. Liu, *Beyond Pixels: Exploring New Representations and Applications for Motion Analysis*. Doctor of philosophy, Massachusetts Institute of Technology, 2009.

[12] I. Selesnick, R. Baraniuk, and N. Kingsbury, "The dual-tree complex wavelet transform," *IEEE Signal Processing Magazine*, vol. 22, no. 6, pp. 123–151, 2005.

[13] S. Becker, J. Bobin, and E. Candès., "NESTA: A fast and accurate first-order method for sparse recovery," *SIAM J Imaging Sciences*, vol. 4, no. 1, 2011.