Flutter Shutter Video Camera for Compressive Sensing of Videos

Jason Holloway Aswin C. Sankaranarayanan Ashok Veeraraghavan Salil Tambe
Department of Electrical and Computer Engineering
Rice University

Abstract

Video cameras are invariably bandwidth limited and this results in a trade-off between spatial and temporal resolution. Advances in sensor manufacturing technology have tremendously increased the available spatial resolution of modern cameras while simultaneously lowering the costs of these sensors. In stark contrast, hardware improvements in temporal resolution have been modest. One solution to enhance temporal resolution is to use high bandwidth imaging devices such as high speed sensors and camera arrays. Unfortunately, these solutions are expensive. An alternate solution is motivated by recent advances in computational imaging and compressive sensing. Camera designs based on these principles, typically, modulate the incoming video using spatio-temporal light modulators and capture the modulated video at a lower bandwidth. Reconstruction algorithms, motivated by compressive sensing, are subsequently used to recover the high bandwidth video at high fidelity. Though promising, these methods have been limited since they require complex and expensive light modulators that make the techniques difficult to realize in practice.

In this paper, we show that a simple coded exposure modulation is sufficient to reconstruct high speed videos. We propose the Flutter Shutter Video Camera (FSVC) in which each exposure of the sensor is temporally coded using an independent pseudo-random sequence. Such exposure coding is easily achieved in modern sensors and is already a feature of several machine vision cameras. We also develop two algorithms for reconstructing the high speed video; the first based on minimizing the total variation of the spatio-temporal slices of the video and the second based on a data driven dictionary based approximation. We perform evaluation on simulated videos and real data to illustrate the robustness of our system.

1. Introduction

Video cameras are inarguably the highest bandwidth consumer device that most of us own. Recent trends are driving that bandwidth higher as manufacturers develop sensors with even more pixels and faster sampling rates.

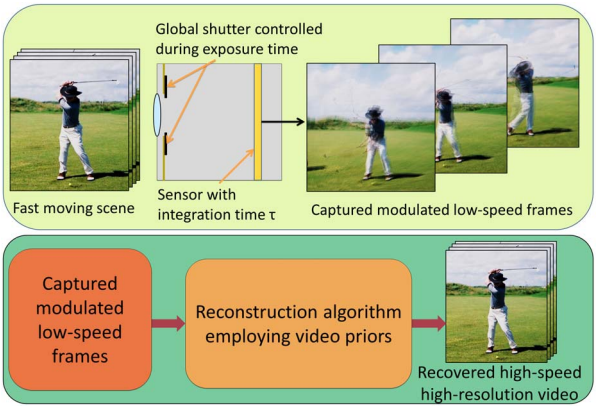

Figure 1. **Flutter Shutter Video Camera (FSVC):** The exposure duration of each frame is modulated using an independent pseudo-random binary sequence. The captured video is a multiplexed version of the original video voxels. Priors about the video are used to then reconstruct the high speed video from FSVC observations.

The escalating demand is forcing manufactures to increase the complexity of the readout circuit to achieve a greater bandwidth. Unfortunately, since the readout circuit shares area with the light sensing element of sensors, this usually results in smaller pixel fill-factors and consequently reduced signal-to-noise ratio. Further, additional circuit elements result in increased cost. This is why even high resolution digital cameras capture videos at reduced spatial resolution so that the effective bandwidth is constrained. While this spatio-temporal resolution trade-off seems fundamental, the fact that videos have redundancies implies that this bandwidth limit is artificial and can be surpassed. In fact, it is this redundancy of videos that enables compression algorithms to routinely achieve $25 - 50$x compression without any perceivable degradation.

Advances in computational cameras and compressive sensing have led to a series of compressive video acquisition devices that exploit this redundancy to reduce the bandwidth required at the sensor. The common principle behind all of these designs is the use of spatio-temporal light modulators and/or exposure control in the imaging system so that the captured video is a multiplexed version of the original video voxels. If the multiplexing is suitably controlled, then

appropriate reconstruction algorithms that exploit the redundancy in the videos can be used to recover the high resolution/high frame-rate video. One such technique is the single pixel camera [6] which reduced the bandwidth required for image acquisition using a random spatial modulation. More recently, there have been a series of imaging architectures [4, 9, 10, 12, 19, 20, 22] that have proposed various alternative ways to compressively sense high speed/resolution videos. While many of these techniques show promising results, they mostly suffer from the same handicap: the hardware modifications required to enable these systems is either expensive/cumbersome or is currently unavailable. In this paper, we propose the Flutter Shutter Video Camera (FSVC), in which the only light modulation is the coded control of the exposure duration in each frame of the captured video. FSVC is, in spirit, similar to many of these above-mentioned techniques, but unlike those techniques it is a simple modification to current digital sensors. In fact, not only are there many machine vision cameras that already have this ability (e.g., Point Grey Dragonfly2), almost all CMOS and CCD sensors can be adapted easily to control the exposure duration.

Contributions: The contributions of this paper are

- We show that simple exposure coding in a video camera can be used to recover high speed video sequences while reducing the high bandwidth requirements of traditional high speed cameras.
- We show that data independent and data-dependent video priors can be used for recovering the high speed video from the captured FSVC frames.
- We discuss the invertibility and compression achievable by various multiplexing schemes for acquiring high speed videos.

2. Related Work

The proposed FSVC relies on numerous algorithmic and architectural modifications to existing techniques.

High speed cameras: Traditional high speed cameras require sensors with high light sensitivity and massive data bandwidth—both of which add significantly to the cost of the camera. The massive bandwidth, caused by the large amount of data sensed over a short duration, typically requires a dedicated bus to the sensor [1]. High-performance commercial systems such as the FastCam SA5 can reach a frame-rate of about 100K fps at spatial resolution of 320×192, but cost about \$300K [1]. In contrast, the FSVC significantly mitigates the dual challenges of light sensitive sensors and data bandwidth by integrating over a much longer exposure time; this naturally increases the signal-to-noise ratio and reduces the bandwidth of the sensed data.

Motion deblurring: The ideas in this paper are closely related to computational cameras first developed for the motion deblurring problem. In motion deblurring [8, 13, 18], the goal is to recover a sharp image and the blurring kernel given a blurred image. Of particular interest, is the Flutter Shutter Camera [18] where the point spread function of the motion blur is shaped by coding the shutter during the exposure; this removes nulls in the point spread function and regularizes the otherwise ill-conditioned forward imaging process. An alternative architecture [13] uses parabolic motion of the sensor to achieve a well conditioned point spread function. While these approaches are only applicable to a small class of scenes that follow a motion model, there is a fundamental difference between video sensing and deblurring. Deblurring seeks to recover a *single* image and an associated blur kernel that encodes this motion. In contrast, video sensing attempts to recover multiple frames and hence, seeks a richer description of the scene and provides the ability to handle complex motion in natural scenes.

Temporal super-resolution: Video compressive sensing (CS) methods rely heavily on temporal super-resolution methods. Mahajan *et al*. [15] describe a method for plausible image interpolation using short exposure frames. But such interpolation based techniques suffer in dimly lit scenes and cannot achieve large compression factors.

Camera arrays: There have been many methods to extend ideas in temporal super-resolution to multiple cameras—wherein the spatial-temporal tradeoff is replaced by a camera-temporal tradeoff. Shechtman *et al*. [21] used multiple cameras with staggered exposures to perform spatio-temporal super-resolution. Similarly, Wilburn *et al*. [24] used a dense array of several 30 fps cameras to recover a 1000 fps video. Agrawal *et al*. [2] improved the performance of such staggered multi-camera systems by employing per-camera flutter shutter. While capturing high speed video using camera arrays produces high quality results (especially for scenes with little or no depth variations), such camera arrays do come with significant hardware challenges. Another related technique is that of Ben-Ezra and Nayar [3] who built a hybrid camera that uses a noisy high frame rate sensor to estimate the point spread function for deblurring a high resolution blurred image.

Compressive sensing of videos: There have been many novel imaging architectures proposed for the video CS problem. These include architectures that use coded aperture [16], a single pixel camera [6], global/flutter shutter [11, 22] and per-pixel coded exposure [12, 19].

For videos that can be modeled as a linear dynamical system, [20] uses a single pixel camera to compressively acquire videos. While this design achieves a high compression at sensing, it is limited to a rather small class of videos that can be modeled as linear dynamical. In [22], the flutter

shutter (FS) architecture is extended to video sensing and is used to build a camera system to capture high-speed periodic scenes. Similar to [20], the key drawback of [22] is the use of a parametric motion model which severely limits the variety of scenes that can be captured. The video sensing architecture proposed by Harmany *et al*. [11], employs a coded aperture and an FS to achieve CS "snapshots" for scenes with incoherent light and high signal-to-noise ratio. In contrast, the proposed FSVC, which also employs an FS, can be used to sense and reconstruct arbitrary videos.

Recently, algorithms that employ per-pixel shutter control have been proposed for the video CS problem. Bub *et al*. [4] proposed a fixed spatio-temporal trade-off for capturing videos via per-pixel modulation. Gupta *et al*. [10] extended the notion to flexible voxels allowing for post-capture spatio-temporal resolution trade-off. Gu *et al*. [9] modify CMOS sensors to achieve a coded rolling shutter that allows for adaptive spatio-temporal trade-off. Reddy *et al*. [19] achieve per-pixel modulation through the use of an LCOS mirror to sense high-speed scenes; a key property of the associated algorithm is the use of optical flow-based reconstruction algorithm. In a similar vein, Hitomi *et al*. [12] use per-pixel coded exposure, but, with an over-complete dictionary to recover patches of the high speed scene. The use of per-pixel coded exposure leads to powerful algorithms capable of achieving high compressions even for complex scenes. Yet, hardware implementation of the per-pixel coded exposure is challenging and is a significant deviation from current commercial camera designs. In contrast, the FSVC only requires a global shutter control; this greatly reducing the hardware complexity needed as compared to systems requiring pixel-level shutter control. Such exposure coding is easily achieved in modern sensors and is already a feature of several machine vision cameras.

3. The Flutter Shutter Video Camera

Flutter shutter (FS) [18] was originally designed as a way to perform image deblurring when an object moves with constant velocity within the exposure duration of a frame. Since FS was essentially a single frame architecture there was very little motion information that could be extracted from the captured frame. Therefore, linear motion [18] or some other restrictive parametric motion model [5] needs to be assumed in order to deblur the image. In contrast, we extend the FS camera into a video camera by acquiring a series of flutter shuttered images with changing exposure code in successive frames. The key insight is that, this captured coded exposure video satisfies two important properties,

1. Since each frame is a coded exposure image, image deblurring can be performed without loss of spatial resolution if motion information is available.
2. Multiple coded exposure frames enable motion information to be extracted locally. This allows us to handle complex and non-uniform motion.

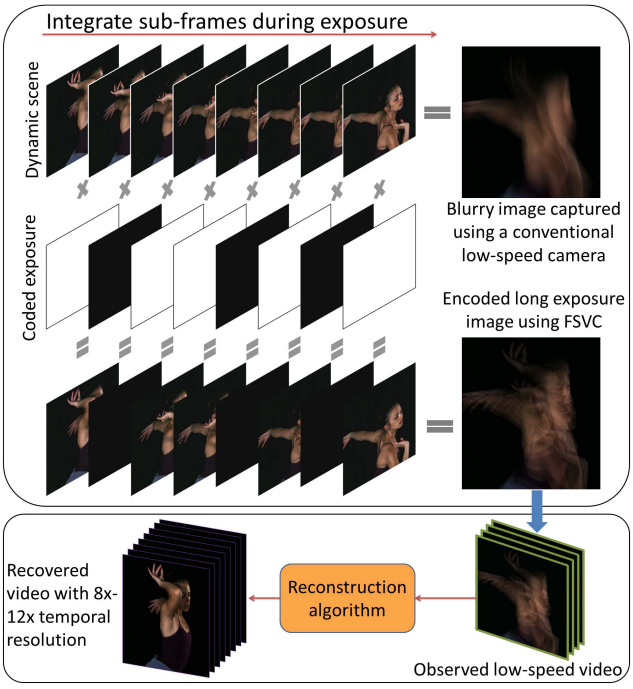

Figure 2. **FSVC Architecture:** Every captured frame is a sum of a pseudo-random sampling of sub-frames.

Thus, for FSVC to work reliably, it is pertinent that both properties are satisfied and that several successive captured frames are available during the decoding process. This stands in contrast with other methods such as [10] and [12] where motion information can be encoded within a single frame by independently changing the exposure time for different pixels.

3.1. Notation and problem statement

Let x be a high speed video of size $M \times N \times T$ and let x_t be the frame captured at time t. A conventional high speed camera can capture x directly, whereas a low speed video camera cannot capture all of the desired frames in x. Therefore, low speed cameras either resort to a short exposure video (in which the resultant frames are sharp, but noisy) or to a full-frame exposure (which results in blurred images). In either case, the resulting video is of size $N \times N \times (T/c)$ where c is the temporal sub-sampling factor. In the FSVC, we open and close the shutter using a binary pseudorandom sequence within the exposure duration of each frame. In all three cases, the observed video frames y_{t_l} are related to the high speed sub-frames x_t as

$$y_{t_l} = \sum_{t=(t_l-1)c+1}^{t_l c} S(t)x_t + n_{t_l}, \qquad (1)$$

where $S(t) \in \{0, 1\}$ is the binary global shutter function, x_t is the sub-frame of x at time t, and n_{t_l} is observation noise

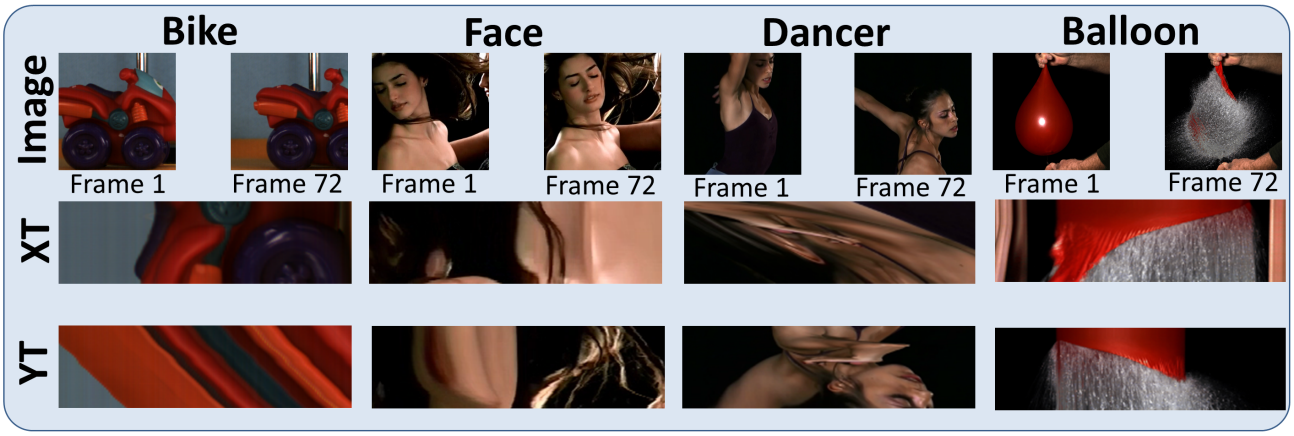

Figure 3. **Total Variation Prior:**The first row shows example frames from four different videos of increasing complexity in motion. The second and third rows show the XT and the YT slices for these videos. It is clear from the XT and the YT slices that there are very few high gradients and therefore minimizing total variation on the XT and YT slices is an appropriate prior for videos. Further, our mixing matrix essentially punches holes in the temporal dimension, i.e., some rows of the XT-YT slices are completely missing in the observations (corresponding to shutter being closed). Therefore, it is important to use a long sequence of XT and YT slice in order to perform the reconstruction. Also notice that edges in the XT and YT slices encode velocity information. For small compression factors and slow moving scenes, local regions of the video can be approximated using linear motion.

modeled as additive white Gaussian noise. For a full exposure camera $S(t) = 1, \forall\, t$, while for short exposure video $S(t)$ is 1 only for one time instant within each captured frame. Our goal is to modify the global shutter function and recover all sub-frames of x_t that are integrated during exposure. Since the observed pixel intensities y are a linear combination of the desired voxels x with weights given by S corrupted by noise, equation (1) can be written in matrix form as

$$\mathbf{y} = \mathbf{S}\mathbf{x} + \mathbf{n}, \tag{2}$$

where $\mathbf{S}$ is a matrix representing the modulation by the global shutter and the observation noise $\mathbf{n}$ is the same size as $\mathbf{y}$. While the modulation of the shutter affects all pixels, the pattern of modulation need not be the same for each integration time.

Equations 1 and 2 hold for all $m \times m \times \tau$ patches of a video, so the same notation will be used for patches and the full video. Unless otherwise mentioned, all equations refer to a patch of the video. Let $\mathbf{x}$ and $\mathbf{y}$ represent the vectorized form of the desired high-speed voxels x (e.g. $8 \times 8 \times 24$) and the observed voxels y (e.g. $8 \times 8 \times 3$) respectively. The observed video $\mathbf{y}$ has significantly fewer entries than the desired true video $\mathbf{x}$ resulting in an under-determined linear system.

4. Reconstruction algorithms

Frames captured using the FSVC are a linear combination of sub-frames with the desired temporal resolution. Given that the number of equations (observed intensities) recorded using the FSVC architecture is significantly smaller than the desired video resolution, direct inversion of the linear system is severely underconstrained. Inspired by advances in compressive sensing, we advocate the use of video priors to enable stable reconstructions.

4.1. Video Priors

Solving the under-determined system in equation (2) requires additional assumptions. These assumptions have typically taken the form of video priors. There are essentially two distinct forms of scene priors that have been used in the literature so far.

Data-independent video priors: One of the most common video priors used for solving ill-posed inverse problems in imaging is that the underlying signal is sparse in some transform basis such as the wavelet basis. This has been shown to produce effective results for several problems such as denoising and super-resolution [7]. In the case of video, apart from wavelet-sparsity one can also exploit the fact that consecutive frames of the video are related by scene or camera motion. In [19], it is assumed that (a) the video is sparse in the wavelet domain, and (b) optical flow computed via brightness constancy is satisfied between consecutive frames of the video. These two sets of constraints provide additional constraints required to regularize the problem. Another signal prior that is data-independent and is widely used in image processing is the total variation regularization. A key advantage with total variation-based methods is that they result in fast and efficient algorithms for video reconstruction. Therefore, we use total variation as one of the algorithms for reconstruction in this paper.

Data-dependent video priors: In many instances, the results obtained using data-independent scene priors can be further improved by learning data dependent over-complete dictionaries [7]. In [12], the authors assume

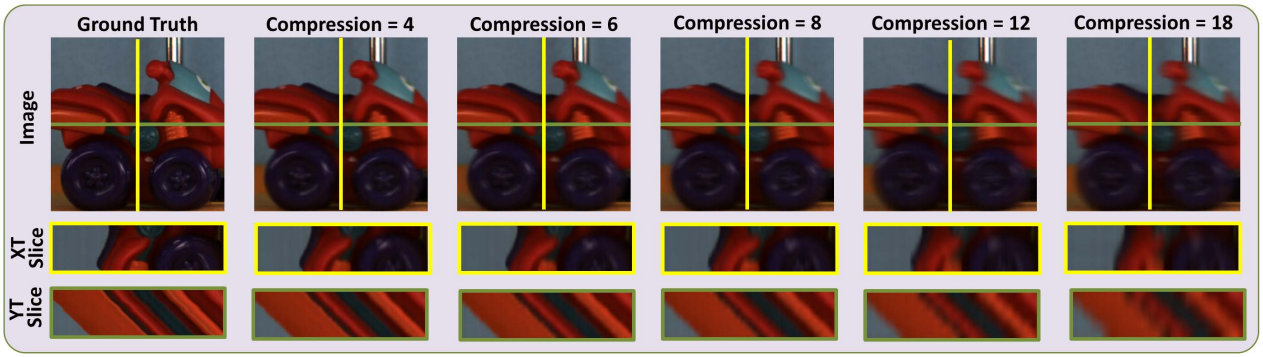

Figure 4. **Results of Flutter Shutter Video Camera** on a video of a toy bike translating with uniform velocity using TV reconstruction. The top row shows one frame of the reconstructed video for various compression factors. As the compression factor increases, the output degrades gracefully. The bottom two rows show the rotated XT and the YT slices corresponding to the column and row marked yellow and green in the first row. The XT and YT slices clearly show the quality of the temporal upsampling.

that patches of the reconstructed video are a sparse linear combination of elements of an overcomplete dictionary; this serves as a regularizing prior. We use a data-dependent over-complete basis as a regularizer and show performance superior to total variation-based methods especially when the compression factor is small ($\leq$ 8). The problem with using data-dependent regularization for very large compression factors is that the learned patch dictionary has to be much larger than that used in [12] since, as discussed earlier, the mixing matrix for FSVC is more ill-conditioned than the mixing matrix in [12] and [19]. Handling such large dictionary sizes is computationally infeasible and therefore, we use the total variation-based prior for handling larger compression factors.

4.2. Total Variation (TV) of XT and YT Slices

Shown in Figure 3 are four different videos with increasing complexity of motion. The second and third row of the figure shows the XT and the YT slices corresponding to the four videos in the first row. In spite of the complexity of the scene and the motion involved in these videos, it is clear that the XT and the YT slices are indeed nothing but deformed versions of images—the deformation being a function of 3D structure and non-uniform velocity of the scene. It is also apparent, that just like images, the XT and YT slices of videos are predominantly flat with very few gradients. Motivated by the sparse gradient distribution in natural images, minimal total variation has been used very successfully as a prior for images [17, 23] for various problems like denoising and deblurring. Similarly, we use minimal total variation in the XT and YT slices as a prior for reconstructing the XT and YT slices from the observations. Needless to say, 3D total variation will probably work even better, but we stick to 2D total variation on XT and YT slices, since this results in a much faster reconstruction algorithm. We use Tval3 [14] to solve the ensuing optimization problem on both the XT and YT slices; the high-speed video is recovered by averaging the solutions of the two optimization problems.

Total variation generally favors sparse gradients. When the video contains smooth motion, the spatio-temporal gradients in the video are sparse, enabling TV reconstruction to successfully recover the desired high-speed video. Recovering the high speed video using spatio-temporal slices of the video cube can thus be executed quickly and efficiently. A $256 \times 256 \times 72$ video channel with a compression factor of 4x can be reconstructed in less than a minute using MATLAB and running on a 3.4GHz quad-core desktop computer. Further, the algorithm is fast and efficient and degrades smoothly as the compression rate increases as shown in Figure 4.

4.3. Data driven dictionary-based reconstruction

While total variation-based video recovery results in a fast and efficient algorithm, promising results from Hitomi *et al*. [12] indicate that significant improvement in reconstruction quality may be obtained by using data driven dictionaries as priors in the reconstruction process. Since the mixing matrix produced by FSVC is far more ill-conditioned than that in [12], we need to learn patches that are larger in both spatial and temporal extent. Motion information is recorded by consecutive observations; we use four recorded frames to reconstruct the high speed video. When the compression rate is c, we learn video patches that are $18 \times 18 \times 4c$ pixels. As the compression rate increases, we need to learn patches that are larger both in spatial and temporal extents, so that the spatio-temporal redundancy can be exploited. Unfortunately, learning dictionaries is computationally infeasible as the dimension increases and so we limit the use of data-driven priors for compression factors less than 8. Using data-driven (DD) priors for such low compression factors resulted in a significant performance improvement over total variation minimization. In the future, as algorithms for dictionary learning become

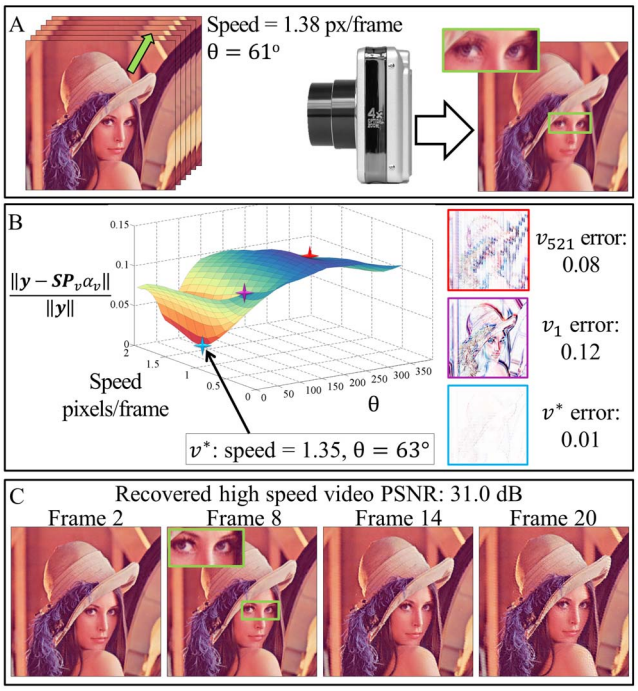

Figure 5. **DD Algorithm. A:** A simulated video of a moving Lena image captured by the FSVC with 6x compression; a captured frame is shown on the right. **B:** Local velocities are determined using equation 3. Overall, the error in the measurement space is smooth and achieves its minimum value at the closest velocity in the database to the actual motion. The error in one frame for three highlighted velocities is shown on the right. Error quickly rises for velocities not near v^* yielding errors that are an order of magnitude larger. **C:** Four frames of the recovered high speed video are shown.

more robust and computationally efficient, we expect that such data-driven priors will indeed perform better than total variation.

In implementing data-driven dictionaries, we make two small adaptations to the traditional dictionary learning and sparse approximation algorithms. First, we extract the principal components for each velocity v, independently. We achieve this by taking images and translating them by the appropriate velocity v to create artificial videos which contain scene elements that moving at the desired velocity. Then we extract patches of size $18 \times 18 \times 28$ from these videos. For each velocity v, we learn the top 324 principal components, and create a principal component matrix, P_v. In practice we generated a total of 521 velocities, sampling heading direction uniformly at 9 degrees and varying the speed from 0.15 pixels/frame to 1.95 pixels/frame (resulting in blur of up to $1.95 * c$ pixels/captured frame). The static motion case is also included. Thus, there are a total of 521 principal component matrices P_v.

For example, for a given compression rate of $c = 7$, we take $18 \times 18 \times 4$ patches from the captured FSVC video. For each such patch we recover a high temporal resolution patch

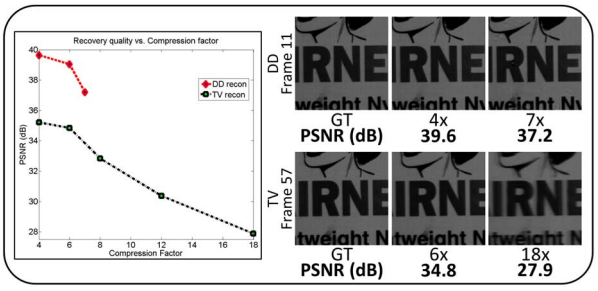

Figure 6. **Reconstruction quality vs compression. Left:** As the compression factor increases, the quality of reconstruction decreases. The DD reconstruction curve is limited to 7x. Both algorithms are tested using the same input video of a hairnets advertisement translating to the right; the TV reconstruction uses all 72 frames while the DD reconstruction only uses the first 24 frames (28 for 7x). **Right:** Frames from the reconstructed videos using both DD (top) and TV based (bottom) algorithms.

which is $18 \times 18 \times 28$ resulting in a temporal upsampling of $c = 7$. For each $18 \times 18 \times 4$ patch from the FSVC video, we estimate the best local velocity v^* as

$$v^* = \text{argmin}_v \, ||\mathbf{y} - \mathbf{S}\mathbf{P}_v\alpha_v||_2^2, \quad v = 1, \ldots, 521. \quad (3)$$

In equation (3), $\mathbf{y}$ is the vectorized observation patch, $\mathbf{S}$ is the observation matrix defined by the flutter shutter code, $\mathbf{P}_v$ is the principal component basis for the vth velocity, and α_v is the least squares estimates of the coefficients denoted by $\alpha_v = (\mathbf{S}\mathbf{P}_v)^\dagger \mathbf{y}$, where $\dagger$ represents the pseudo-inverse. Figure 5B shows that the error is much smaller for velocities near the true patch velocity and quickly rises for other velocities. Finally, each patch is reconstructed as

$$\widehat{\mathbf{x}} = \mathbf{P}_{v^*}\alpha_{v^*}. \quad (4)$$

The recovered high speed video in Figure 5C demonstrates the high quality reconstruction that can be achieved using the DD algorithm. After reconstructing each patch in the observation, overlapping pixels are averaged to generate the reconstructed high speed video. Recovering a high speed video with this algorithm is considerably slower than the TV-based reconstruction, a $252 \times 252 \times 28$ video channel with a compression factor of 7x takes approximately 5 minutes using the same computation setup described in Section 4.2. The more pressing issue, is that such a data-driven method suffers from two disadvantages especially when handling large compression factors: (1) learning is prohibitively slow, and (2) the locally linear motion assumption is violated when the temporal extent of patches becomes longer. In practice, we notice that this algorithm results in significant improvement over total variation for small compression factors.

5. Experiments

We evaluate the performance of FSVC through simulations on videos with frame rates of $120 - 1000$ fps. We also

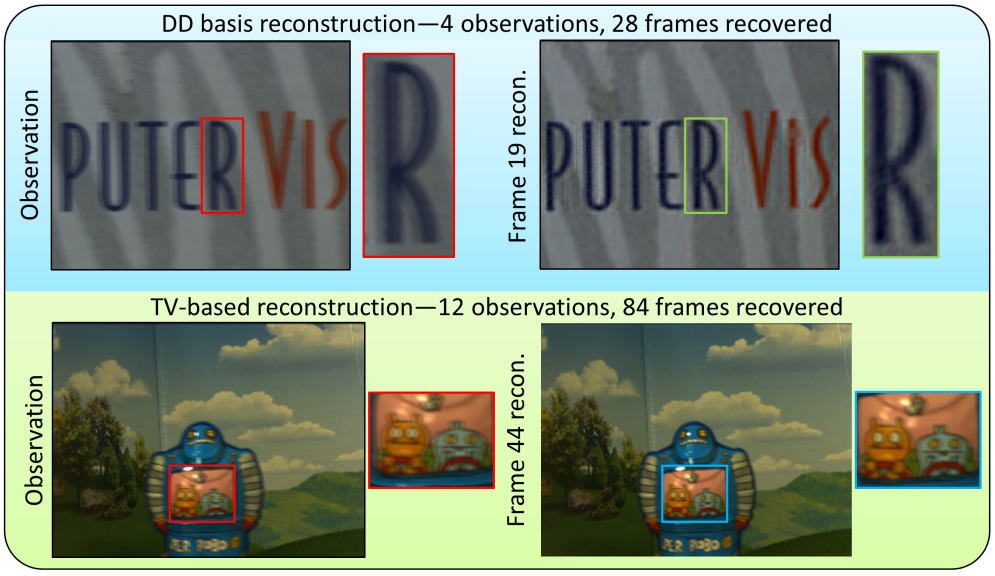

Figure 8. **Experimental Results.** Observation images with a spatial resolution of 234×306 were captured at 7 frames per second with a compression factor of 7x. **Top:** Four frames of a book being pulled to the left are captured by FSVC, ghosting artifacts can be observed in the outset. The high speed video is reconstructed using the DD algorithm and one frame is shown. The outset shows that the ghosting has been removed. **Bottom:** A toy robot is moved to the right along a non-linear path, 12 observations are captured using FSVC. Ghosting of the fine details on the robot can be seen in the outset. Reconstruction was done with the TV algorithm and one frame of the output is shown. The outset shows that fine details have been preserved.

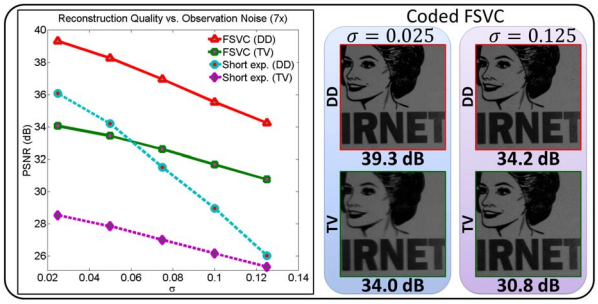

Figure 7. **Reconstruction quality vs. noise. Left:** As the standard deviation of the observation noise increases, the reconstruction quality of FSVC decays gracefully. Further the coding in FSVC provides a 4-5 dB performance improvement over short exposure. **Right:** The same frame from the high speed video recovered using both algorithms and when using FSVC; the quality degrades slowly with the addition of few visible artifacts.

capture data using a Point Grey Dragonfly2 video camera and reconstruct using both algorithms.

Effect of compression rate: We first test the robustness of our algorithms through simulation. Simulated observations are obtained by sampling a high speed ground truth video using the forward process of equation (2). To test the robustness of our algorithm at varying compression factors, we use a video of hairnets advertisement moving from right to left with a speed of ~ 0.5 pixels per frame (video credit: Amit Agrawal). The input video has values in the range $[0, 1]$, the observation noise has a fixed standard deviation, $\sigma = 0.025$, and the exposure code is held constant. The observed videos have a PSNR of ~ 34 dB. The effect of increasing the compression on the quality of the recovered video is highlighted in Figure 4. As the temporal super-resolution increases, FSVC retains fewer dynamic weak edges but faithfully recovers large gradients leading to a graceful degradation of video quality. Notice, in Figure 4, that the hub caps of the wheels of the bike are present in the reconstructed video even as the compression is increased to a factor of 18. FSVC has a high spatial resolution that allows slowly moving weak edges to be preserved in the video reconstruction. Shown in the plot in Figure 6, is the reconstruction PSNR of both algorithms as a function of the compression rate. Notice that for low compression factors, the data-driven algorithm performs better than total variation, since the locally linear motion assumption is not violated. As the compression factor increases, the peak signal-to-noise ratio (PSNR) of the reconstructed video decays gracefully for the TV-based algorithm. The ability to retain very high quality reconstructions for static/near-static elements of the scene and the graceful degradation of the TV algorithm, both of which are apparent from the reconstructions of the hairnet advertisement video in Figure 6, are very important attributes of any compressive video camera.

Robustness to observation noise: Figure 7 shows the effects of varying levels of noise on the fidelity of reconstruction for both the DD and TV algorithms. The same hairnets video is used as the ground truth and the standard

deviation of the noise varies from 0.025 to 0.125 in increments of .025. This corresponds to a range of PSNRs from $20.8 - 34.8$ dB. The compression factor has been fixed at 7x. In this experiment, we compare the quality of the high speed video recovered from images captured using FSVC and a standard video camera using a short exposure. The short exposure video is created by modifying the temporal code used in $\mathbf{S}$ to have a single 1 for the same time instant t during each exposure. The coded frames and short exposure frames are reconstructed using the DD and TV algorithms. As expected for a low-speed and approximately linear scene, the DD algorithm reconstruction is of higher quality than the TV reconstruction. This experiment shows two observations: (1) degradation of the FSVC reconstruction is graceful in the presence of noise and (2) FSVC coding improves the fidelity of reconstruction by approximately 5 dB compared to simple short exposure (at a compression rate of $c = 7$).

Experiments on real data: The FSVC can be implemented using the built-in functionality of the Point Grey Dragonfly2 video camera.[1] The coded exposure pattern is provided using an external trigger. Due to inherent limitations imposed by the hardware, images were captured at 7 frames per second with a compression factor of 7x. Hence, our goal is to obtain a working 49 frames per second camera by capturing 7 frames per second coded videos. Figure 8 shows the input coded images collected by the camera, and frames of the reconstructed video using the TV and DD algorithms. The top row of Figure 8 shows one observation frame recorded with FSVC of a book moving to the left. Ghosting artifacts in the observation frame demonstrate that using a traditional video camera with the same framerate would result in a blurry video. We recovered a high-speed video using the DD methods; as expected, the recovered video correctly interpolated the motion and removes the ghosting artifacts.

A second dataset was captured of a toy robot moving with non-uniform speed. One captured frame from this dataset is shown in Figure 8 (bottom-left). The outset shows an enlarged view of a blurry portion of the observed frame. Figure 8 (bottom-right) shows one frame from the reconstructed video and the outset shows the corresponding enlarged section of the recovered frame. Notice that the blurring is significantly reduced but since the motion of the toy is quite large, there is still some residual blurring in the reconstructed video.

6. Discussions and Conclusions

Benefits: Our imaging architecture provides three advantages over conventional imaging. It significantly reduces the bandwidth requirement at the sensor. It exploits exposure coding which is already a feature of several machine vision cameras making it easy to implement. It improves light throughput of the system compared to acquiring a short exposure low frame-rate video or [12] and allows acquisition at low light levels. These are significant advantages since the prohibitive cost of high-speed imagers is due to the requirement for high bandwidth and high light sensitivity. Figure 9 highlights these advantages of our imaging architecture as well as places it in a larger context of computational cameras developed over the last decade.

[1]Exposure mode 5 is used to control the timing and duration of multiple exposures within each integration time of the camera.

	Flutter shutter [16]	Parabolic motion [11]	Invertible motion [2]	SPC [6]	CS-LDS [18]	Coded strobing [20]	P2C2 [17]	CPEV [10]	Coded rolling shutter [8]	FSVC (this paper)
Bandwidth reduction	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Motion model	Linear	Linear	Linear	None	LDS	Periodic	None	None	Linear	None
Side information	Motion magnitude	Motion direction	None	None	None	None	None	None	None	None
Hardware complexity	Simple	Simple	Simple	Complex	Complex	Simple	Complex	Medium	Medium	Simple
Commercial availability	Yes	No	Yes	No	No	Yes	No	No	No	Yes
Light throughput	50%	100%	~100%	50%	50%	50%	50%	1/(Comp factor)	-	60%
Typical compression shown*	1	1	1	~5x	~200x	~80x	~32x	~32x	~4x	~(8-12)x

Figure 9. Comparison of the qualitative properties of compressive computational video cameras.

Compression Vs PSNR (dB)	P2C2 [19]	CPEV [12]	FSVC (this paper)
4	41.5	41.8	35.6
8	37.4	38.8	33.1
12	34.2	36.0	29.8
18	31.2	34.5	29.1

Table 1. **Comparison of compression vs reconstruction PSNR in dB** for compressive video acquisition architectures in the absence of observation noise. FSVC is a much simpler hardware architecture than either P2C2 or CPEV, but it results in reasonable performance.

Limitations: FSVC exploits the spatio-temporal redundancy in videos during the reconstruction phase of the algorithm. Scenes that do not have spatio-temporal redundancy such as a bursting balloons cannot be handled by the camera. Since the spatio-temporal redundancy exploited by traditional compression algorithms and our imaging architecture are very similar, as a rule of thumb one can assume that scenes that are compressed efficiently can be captured well using our method. FSVC uses a coded exposure and this causes a 33% reduction in light throughput since we use codes that are open 67% of the time.

Global vs per-pixel shutter: The proposed FSVC comes as an advancement over two recently proposed computational cameras [12, 19] that use a per-pixel shutter. The per-pixel shutter control enables different temporal codes at different pixels. In contrast, in our design, the control of the shutter is global and, hence, all pixels share the same temporal code. At first glance, this might seem like a small difference–yet, the implications of this are profound.

A global shutter leads to ill-conditioned measurement matrix. An easy way to observe this is to study properties of the adjoint operator of the measurement matrix defined in equation (2). The adjoint of an observed video is a high-resolution video with zeros for frames at time-instants when the shutter is closed. The abrupt transition from the video signal to the nulls introduces high temporal frequencies. Hence, the adjoint operator is highly coherent to high frequency patterns and is predisposed to selecting high-frequency atoms when used with a sparse approximation algorithm (as in [12]). In contrast, the adjoint operators associated with both the P2C2 and CPEV [12] are less coherent with such bases. Hence, it is much easier to obtain higher quality reconstructions with these. Shown in Table 1 is a comparison of reconstruction performance using 3 different modulation schemes, per-pixel flutter shutter as in P2C2 [19], per pixel single short exposure as in [12] and global flutter shutter video camera as proposed in this paper. It is clear that the simple and easy to implement architecture of FSVC results in reconstruction performance that is about 6 dB lower per-pixel coding architectures. Further, these experiments did not include observation noise. Since CPEV has a very low light throughput ($1/$compression), the performance of CPEV will degrade significantly (compared to P2C2 and FSVC) in the presence of noise especially in dimly lit scenes.

Acknowledgments. This work was supported by NSF awards IIS-1116718 and CCF-1117939.

References

[1] www.photron.com.

[2] A. Agrawal, M. Gupta, A. Veeraraghavan, and S. Narasimhan. Optimal Coded Sampling for Temporal Super-Resolution. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 599–606, Jun 2010.

[3] M. Ben-Ezra and S. K. Nayar. Motion-Based Motion Deblurring. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):689–698, 2004.

[4] G. Bub, M. Tecza, M. Helmes, P. Lee, and P. Kohl. Temporal Pixel Multiplexing for Simultaneous High-Speed, High-Resolution Imaging. *Nature Methods*, 7(3):209–211, 2010.

[5] Y. Ding, S. McCloskey, and J. Yu. Analysis of Motion Blur with a Flutter Shutter Camera for Non-Linear Motion. In *European Conference on Computer Vision*, pages 15–30, Sep 2010.

[6] 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*, 25(2):83–91, 2008.

[7] M. Elad. *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*. Springer Verlag, 2010.

[8] R. Fergus, B. Singh, A. Hertzmann, S. Roweis, and W. Freeman. Removing camera shake from a single photograph. In *ACM SIGGRAPH*, pages 787–794, Aug 2006.

[9] J. Gu, Y. Hitomi, T. Mitsunaga, and S. Nayar. Coded Rolling Shutter Photography: Flexible Space-Time Sampling. In *IEEE International Conference on Computational Photography*, pages 1 –8, Mar 2010.

[10] M. Gupta, A. Agrawal, A. Veeraraghavan, and S. Narasimhan. Flexible Voxels for Motion-Aware Videography. *European Conference on Computer Vision*, pages 100–114, Sep 2010.

[11] Z. T. Harmany, R. F. Marcia, and R. M. Willett. Spatio-temporal Compressed Sensing with Coded Apertures and Keyed Exposures. *ArXiv e-prints*, Nov. 2011.

[12] Y. Hitomi, J. Gu, M. Gupta, T. Mitsunaga, and S. K. Nayar. Video from a Single Coded Exposure Photograph using a Learned Over-Complete Dictionary. In *IEEE International Conference on Computer Vision*, pages 287 –294, Nov 2011.

[13] A. Levin, P. Sand, T. S. Cho, F. Durand, and W. T. Freeman. Motion-Invariant Photography. In *ACM SIGGRAPH*, Aug 2008.

[14] C. Li. An Efficient Algorithm for Total Variation Regularization with Applications to the Single Pixel Camera and Compressive Sensing. Master's thesis, Rice University, 2009.

[15] D. Mahajan, F. Huang, W. Matusik, R. Ramamoorthi, and P. Belhumeur. Moving Gradients: A Path-Based Method for Plausible Image Interpolation. In *ACM SIGGRAPH*, pages 42:1–42:11, Aug 2009.

[16] R. F. Marcia and R. M. Willett. Compressive Coded Aperture Video Reconstruction. In *EUSIPCO*, Aug 2008.

[17] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W. Yin. An Iterative Regularization Method for Total Variation-Based Image Restoration. *Multiscale Modeling and Simulation*, 4(2):460, 2005.

[18] R. Raskar, A. Agrawal, and J. Tumblin. Coded Exposure Photography: Motion Deblurring Using Fluttered Shutter. In *ACM SIGGRAPH*, pages 795–804, Aug 2006.

[19] 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*, pages 329 –336, Jun 2011.

[20] A. Sankaranarayanan, P. Turaga, R. Baraniuk, and R. Chellappa. Compressive Acquisition of Dynamic Scenes. *European Conference on Computer Vision*, pages 129–142, Sep 2010.

[21] E. Shechtman, Y. Caspi, and M. Irani. Space-Time Super-Resolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4):531–545, 2005.

[22] A. Veeraraghavan, D. Reddy, and R. Raskar. Coded Strobing Photography: Compressive Sensing of High Speed Periodic Videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(4):671 –686, Apr 2011.

[23] Y. Wang, J. Yang, W. Yin, and Y. Zhang. A new alternating minimization algorithm for total variation image reconstruction. *SIAM Journal on Imaging Sciences*, 1(3):248–272, 2008.

[24] B. Wilburn, N. Joshi, V. Vaish, E. Talvala, E. Antunez, A. Barth, A. Adams, M. Horowitz, and M. Levoy. High Performance Imaging Using Large Camera Arrays. *ACM SIGGRAPH*, pages 765–776, Aug 2005.