To appear in the ACM SIGGRAPH conference proceedings

Factored Time-Lapse Video

Kalyan Sunkavalli* Wojciech Matusik† Hanspeter Pfister‡ Szymon Rusinkiewicz§
MERL Princeton University

(a) Original (b) Reconstructed, no shadows (c) Sun illumination only (d) Modified reflectance

Figure 1: *We decompose a time-lapse sequence of photographs (a) into sun, sky, shadow, and reflectance components. The representation permits re-rendering without shadows (b) and without skylight (c), or modifying the reflectance of surfaces in the scene (d).*

Abstract

We describe a method for converting time-lapse photography captured with outdoor cameras into *Factored Time-Lapse Video* (FTLV): a video in which time appears to move faster (i.e., lapsing) and where data at each pixel has been factored into shadow, illumination, and reflectance components. The factorization allows a user to easily relight the scene, recover a portion of the scene geometry (normals), and to perform advanced image editing operations. Our method is easy to implement, robust, and provides a compact representation with good reconstruction characteristics. We show results using several publicly available time-lapse sequences.

CR Categories: I.4.8 [Image Processing and Computer Vision]: Scene Analysis—Time-varying Imagery I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Color, shading, shadowing, and texture

Keywords: Image-based rendering and lighting, inverse problems, computational photography, reflectance

1 Introduction

Time-lapse photography, in which frames are captured at a lower rate than that at which they will ultimately be played back, dates back to the late 19-th century[1]. Classic time-lapse photography subjects are clouds, stars, plants, and flowers. Today, most time-lapse image sequences are collected by the thousands of cameras (webcams) whose images can be accessed using the Internet. They typically provide outdoor views of cities, construction sites, traffic, the weather, or natural phenomena such as volcanoes. Outdoor webcams are also used for surveillance to monitor outside activities around companies, ports, or warehouses. So-called webcam directories index thousands of live webcams, providing instant online access to time-lapse photos from around the world.

Time-lapse photography can create an overwhelming amount of data. For example, a single camera that takes an image every 5 seconds will produce 17,280 images per day, or close to a million images per year. Image or video compression reduces the storage requirements, but the resulting data has compression artifacts and is not very useful for further analysis. In addition, it is currently difficult to edit the images in a time-lapse sequence, and advanced image-based rendering operations such as relighting are impossible. *A key challenge in dealing with time-lapse data is to provide a representation that efficiently reduces storage requirements while allowing useful scene analysis and advanced image editing.*

In this paper we focus on time-lapse sequences of outdoor scenes under clear-sky conditions. The camera viewpoint is fixed and the scene is mostly stationary, hence the predominant changes in the sequence are changes in illumination. Under these assumptions we have developed a method that provides a complete decomposition of the original dataset into shadow, illumination, and reflectance components. We call this representation *Factored Time-Lapse Video* (FTLV).

Our method begins by locating the onset of shadows using the time-varying intensity profiles at each pixel. We identify points in shadow and points in direct sunlight to separate skylight and sunlight components, respectively. We then analyze these spatio-temporal volumes using matrix factorization. The results are basis curves describing the changes of intensity over time, together with per-pixel offsets and scales of these basis curves, which capture spatial variation of reflectance and geometry. The resulting representation is compact, reducing a time-lapse sequence to three images, two basis curves, and a compressed representation for shadows. Reconstructions from the data show better error characteristics than standard compression methods such as PCA.

FTLVs are an intrinsic image-like scene representation that allow a user to analyse, reconstruct and modify illumination, reflectance, or

*e-mail: sunkaval@merl.com
†e-mail:matusik@merl.com
‡e-mail:pfister@merl.com
§e-mail:smr@cs.princeton.edu

[1]Georges Méliès' motion picture *Carrefour De L'Opera* (1897) (wikipedia.org)

geometry. The shadows may be discarded or retained, depending on the ultimate application, while other "outliers" such as pedestrians or cars are implicitly ignored. FTLVs can also be used for various computer vision tasks such as background modeling, image segmentation, and scene reconstruction. In this paper we demonstrate several applications for FTLVs, including relighting, shadow removal, advanced image editing, and pictorial rendering.

2 Previous Work

Time-Lapse Sequence Analysis Barrow and Tenenbaum [1978] introduced the concept of *intrinsic images* to represent intrinsic characteristics of a scene, such as illumination, reflectance, and surface geometry. Weiss [2001] uses a maximum-likelihood framework to estimate a single reflectance image and multiple illumination images from time-lapse video. His work was extended by Matsushita et al. [2004] who derive time-varying reflectance and illumination images from surveillance video.

Matusik et al. [2004] use time-lapse data to compute the reflectance field (or light transport) of a scene for a fixed viewpoint. They represent images as a product of the reflectance field and the incoming illumination. However, the method requires estimating the incident illumination using a light probe camera, and the estimated reflectance field combines the effects of reflectance and shadows.

Koppal and Narasimhan [2006] acquire image sequences with a randomly moving light source to cluster the image into regions that have similar normals. These normal clusters are then used as priors to bootstrap a variety of vision algorithms, including the decomposition of the image into the terms of a linearly separable BRDF model. Because the illumination is required to follow an unstructured trajectory, Koppal and Narasimhan's work is applicable only to outdoor time-lapse data captured over a larger period of time (many days or weeks).

Unlike this previous work, we arrive at a decomposition of time-lapse sequences into shadows, partial scene geometry, and time-varying reflectance and illumination. This provides better estimates of intrinsic image qualities and more accurate scene analysis.

Reflectance Factorizations Lawrence et al. [2006] describe a factorization method to decompose complex surface reflectance functions (spatially varying BRDFs) into a sum of products of lower dimensional (1D or 2D) terms. Similarly, Gu et al. [2006] decompose a time-varying surface appearance into lower dimensional representations that are space-time dependent. Both of these methods accomplish similar goals — they factorize large datasets of complex surface reflectances into terms that are highly compact and at the same time physically meaningful and editable. Since they acquire and model the full eight-dimensional BRDF they can render under any viewing, lighting, and, in the case of Gu et al.[2006], temporal condition. However, the complexity of the BRDF acquisition setup makes it impractical for complex, outdoor scenes.

Our work bridges the gap between intrinsic images and the factorization of complex reflectance functions. In doing so we go beyond the mid-level representation of intrinsic images without requiring the acquisition setup and calibration of a complete BRDF estimation.

Inverse Rendering Inverse rendering measures rendering attributes — lighting, textures, and BRDF — from photographs. Although there is prolific literature on inverse rendering, most of the work focuses on small objects and indoor scenes. Yu and Malik [1998] and Debevec et al. [2004] recover photometric properties from photographs of outdoor architectures. They are able to relight them and create photo-realistic images from arbitrary viewpoints. However, their methods require measurements of the incident illumination and surface materials and a 3D model of the scene geometry. Similarly, Nimeroff et al. [1994] render scenes under natural illumination by combining basis images that are pre-rendered using a set of basis illuminations. But they also require measurements of scene geometry and reflectances.

Seitz et al. [2005] and Nayar et al. [2006] separate the illumination in a scene into its direct and global components using controlled lighting. They demonstrate this for real-world materials and comment on the contribution of the two terms to surface appearance. We separate illumination into a global sky and a direct sun component while ignoring secondary inter-reflections, scattering, or translucency.

Video Analysis and Editing There has been considerable research on analyzing video sequences to segment out objects and extract mattes [Chuang et al. 2002; McGuire et al. 2005; Li et al. 2005; Wang et al. 2005], to extract shadows [Chuang et al. 2003], to determine object motion and texture [Bregler et al. 1997; Schödl et al. 2000; Bhat et al. 2004], and to combine video frames into panoramas [Agarwala et al. 2005]. In addition, there has been recent work that modifies video by inserting objects [Li et al. 2005; Wang et al. 2005] or shadows [Chuang et al. 2002], enhancing small motion [Liu et al. 2005], or applying non-photorealistic stylization [Litwinowicz 1997; Wang et al. 2004; Winnemöller et al. 2006]. In contrast, the present paper performs a different type of analysis, focusing not on inferring objects' shape, motion, or texture, but rather on decomposing appearance and reflectance. Our analysis enables a different class of edits, such as changing reflectance and lighting. In addition, by extracting partial geometry of the scene, we are able to perform stylization using algorithms such as exaggerated shading [Rusinkiewicz et al. 2006], which requires more knowledge about the scene than simply pixel intensities.

3 Representation

Our goal is to decompose the space-time volume $\mathbf{F}(t)$ of the time-lapse image sequence into factors that will enable us to analyze and edit the scene. As the sun moves, the observations at every pixel in the time-lapse sequence result in a continuous *appearance profile* [Koppal and Narasimhan 2006] (see Figure 4(a)). The appearance profile (red curve) is a vector of intensities $F_i(t)$ measured at pixel P_i over time (i.e., frame number). It is a complicated function of the illumination, scene geometry, and surface reflectance.

Under the clear-sky assumption we can approximate the illumination as a sum of an ambient term corresponding to sky illumination and a single-directional light source corresponding to the radiance of the sun. Using the linearity of the rendering equation, the spatio-temporal volume $\mathbf{F}(t)$ can therefore be expressed as a sum of the sky light and sunlight components:

$$\mathbf{F}(t) = \mathbf{I}_{sky}(t) + \mathbf{S}_{sun}(t) * \mathbf{I}_{sun}(t). \quad (1)$$

Here $\mathbf{S}_{sun}(t)$ is the shadowing term that describes if a pixel is in shadow (and therefore has no sunlight contribution) or not.

One of the key insights in this paper is that for outdoor time-lapse sequences under clear-sky conditions *the appearance profiles of all points in the scene are similar up to an offset along the time axis and a scale factor.* This is similar to the notion of *orientation-consistency* introduced by Hertzmann and Seitz [2005], which states that, under the right conditions, two points with the same surface orientation must have the same or similar appearance

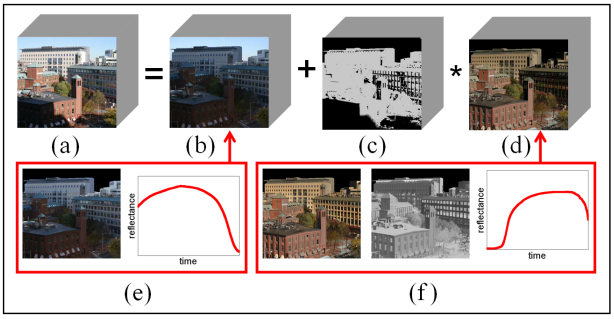


Figure 2: *An overview of the FTLV factorization. We separate the spatio-temporal volume (a) $\mathbf{F}(t)$ into the sky component (b) $\mathbf{I}_{sky}(t)$ and the sun component (d) $\mathbf{I}_{sun}(t)$ modulated by the shadow volume (c) $\mathbf{S}(t)$. The sky component is factorized into (e) per-pixel weights $\mathbf{W}_{sky}$ and a time-curve $\mathbf{H}_{sky}(t)$ while the sun component is factorized into (f) albedo $\mathbf{W}_{sun}(t)$, reflectance $\mathbf{H}_{sun}(t)$ and per-pixel shifts $\mathbf{S}(t)$.*

in an image. They compute the surface normals of a target object that has been imaged together with one or more reference objects with known shape (sphere) and similar BRDF. Koppal and Narasimhan [2006] also noted that scene points with the same surface normal often exhibit extrema in their profiles at the same time instant, irrespective of their material properties. They use a metric that matches appearance profile extremas, and unsupervised clustering, to compute orientation consistencies between scene points of unknown normals and BRDFs. The key difference in our work is that we seek to separate the contributions of shadows, skylight, and sun illumination. Similar to the work by Gu et al. [2006], we represent the corresponding appearance profiles as a linear combination of basis curves that are offset and scaled.

Based on our insight, we approximate $\mathbf{I}_{sun}(t)$ with a *single* basis curve $\mathbf{H}_{sun}(t)$ scaled by per-pixel weights $\mathbf{W}_{sun}$. In addition, we allow a per-pixel time offset Φ to the sunlight curve that stands in for the normal dependence of the appearance profile:

$$\mathbf{I}_{sun,i}(t) \approx \mathbf{W}_{sun,i}\,\mathbf{H}_{sun}(t + \Phi_i). \quad (2)$$

We call weight matrix $\mathbf{W}_{sun}$ the *sunlight image*, the basis curve $\mathbf{H}_{sun}(t)$ the *sunlight basis curve* and the offset Φ the *shift map*. Since the sun is a strong directional light source, $\mathbf{H}_{sun}(t)$ is an estimate of the 1-D slice of surface reflectance corresponding to the camera viewpoint and the arc described by the sun's motion.

The time-shifted basis is an accurate representation for appearance profiles because of the nature of sun illumination. Since the sun moves at a constant angular velocity, the appearance profiles at pixels with different surface normals correspond to different uniformly sampled 1-D slices (corresponding to the camera viewpoint and the plane of the sun) of the 4-D BRDF. For both diffuse and specular surfaces it has been shown that the shape of this slice is largely the same with the fundamental difference being the normal-dependent shift that Φ captures. If the arc described by the sun is known, under the assumption that the appearance profile is maximum when the sunlight is normally incident on the surface, the computed shift map can be converted into an estimate of the surface normal. Since the sun moves only in a plane, the shift map is an estimate of the surface normal projected onto this plane.

In order to estimate $\mathbf{I}_{sky}(t)$ in Equation (1), we look to find a *single* illumination-vs.-time basis curve $\mathbf{H}_{sky}(t)$ for the entire image sequence, such that the appearance of any shadowed pixel P_i may be reproduced as:

$$\mathbf{I}_{sky,i}(t) \approx \mathbf{W}_{sky,i}\,\mathbf{H}_{sky}(t). \quad (3)$$

We call the matrix $\mathbf{W}_{sky}$ of per-pixel weights the *skylight image*, and the basis curve $\mathbf{H}_{sky}(t)$ the *skylight basis curve.* Note that we do not apply an offset to the skylight curve in Equation (3) because the diffuse nature of sky illumination makes the offset hard to estimate.

The final representation (see Figure 2) therefore is:

$$\mathbf{F}(t) \approx \mathbf{W}_{sky}\,\mathbf{H}_{sky}(t) + \mathbf{S}_{sun}(t) * \mathbf{W}_{sun}\,\mathbf{H}_{sun}(t + \Phi). \quad (4)$$

Our goal is to separate $\mathbf{F}(t)$ into $\mathbf{I}_{sky}(t)$, $\mathbf{S}_{sun}(t)$, and $\mathbf{I}_{sun}(t)$ and estimate all the per-pixel weights, per-pixel shifts, and time-curves without knowledge of scene geometry, reflectance, illumination, or camera calibration. We make the observation that we can estimate $\mathbf{I}_{sky}(t)$ from points that are in shadow, whereas points in the sun contain $\mathbf{I}_{sky}(t)+\mathbf{I}_{sun}(t)$. Our approach is to first estimate $\mathbf{S}_{sun}(t)$ and to use this to separate $\mathbf{I}_{sky}(t)$ and $\mathbf{I}_{sun}(t)$ and estimate the rest of the terms. The next section describes how we estimate $\mathbf{S}_{sun}(t)$ from the time-lapse data, and Section 5 shows how we use matrix factorization to solve for $\mathbf{W}_{sky}$, $\mathbf{H}_{sky}(t)$, $\mathbf{W}_{sun}$, Φ and $\mathbf{H}_{sun}(t)$.

4 Shadow Estimation

Figure 3 shows one frame of a time-lapse sequence of the Santa Catalina Mountains in Arizona. We will use this frame and pixels A, B, and C throughout our discussion. All computations are performed in RGB color space and independently on the three color channels. For simplicity we will focus on the red channel only.

Figure 3: *Frame 275 of the Arizona time-lapse sequence.*

Our model assumes that the color of visible pixels is due to reflection from surfaces in the scene. Therefore we do not consider sky pixels in our representation. We compute the sky mask (using Photoshop's magic wand tool) for one frame and use it to later composite the sky from the original data to the reconstructed images. This has the added benefit that we preserve moving clouds, which are an important visual component in time-lapse videos.

As can be seen in Figure 4(a) pixel intensities differ dramatically if the scene point is illuminated by the sun or if it is in shadow. We use these discontinuities in the appearance profiles to estimate shadow images. We first compute the median value m_{min} of the n smallest intensities at each pixel. We typically assume that each point is in shadow at least 20% of the time, so n is 20% of the total number of frames in the sequence. We set the shadow function $S_i(t)$ (purple curve in Figure 4(a)) to one for each $F_i(t) > km_{min}$ and to zero otherwise. We heuristically found that $k = 1.5$ worked well for most of the sequences in our experiments.

Figure 5 (top) shows the value of the shadow function for all pixels in frame 275. We call this the binary shadow image. It is generally quite noisy due to moving objects (e.g., trees, smoke, or people) and changes in the illumination (e.g., clouds). Therefore, to compute the final shadow image $\mathbf{S}_{sun}$ we use an edge-preserving bi-lateral filter [Tomasi and Manduchi 1998] (Figure 5 (bottom)). We could improve the shadow images by explicitly removing dynamic objects, either by computing median images for short time

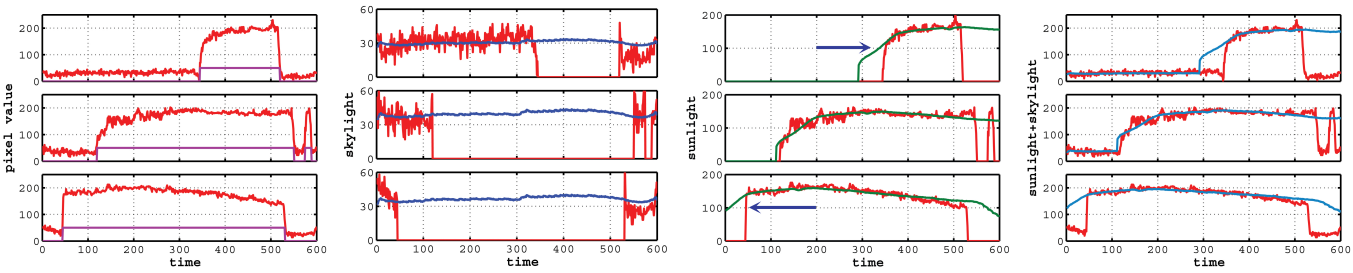


Figure 4: *(a) Appearance profiles (red) and shadow functions (purple) of points A, B, and C (top to bottom). (b) Appearance profiles when points A, B, and C (top to bottom) are in shadow (red) and the estimated skylight curves (blue). Note the different scale on the intensity axis. (c) Appearance profiles when points A, B, and C are in the sun (red) and the estimated sunlight curves (green). The arrows indicate the direction of the shift of the basis curves. (d) Original appearance profiles (red) and the sum of sunlight and skylight curves (blue).*

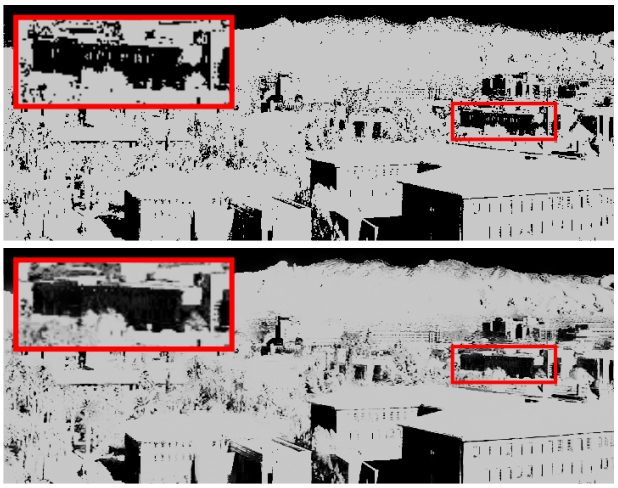


Figure 5: *Top: Binary shadow image. Bottom: Shadow image after edge-preserving bi-lateral filtering. The inset shows a closeup that illustrates the filtering.*

intervals in the input sequence [Matsushita et al. 2004] or by using more sophisticated background models.

5 Time-Lapse Factorization

The computational framework we use to decompose appearance profiles into $\mathbf{W}$ and $\mathbf{H}$ factors is Alternating Constrained Least Squares (ACLS) [Lawrence et al. 2006], which has previously been applied to the problem of decomposing measured reflectance data into intuitively editable components. ACLS is similarly well suited for our task of decomposing appearance profiles, since its ability to incorporate domain-dependent constraints — such as non-negativity, sparseness, and smoothness — leads to stable and intuitive decompositions. ACLS can also incorporate a confidence matrix $\mathbf{C}$ to deal with missing data; setting an entry of $\mathbf{C}$ to zero will cause the corresponding measurement to have no effect on the factorization. We use the matrix $\mathbf{C}$ to implicitly separate skylight and sunlight components.

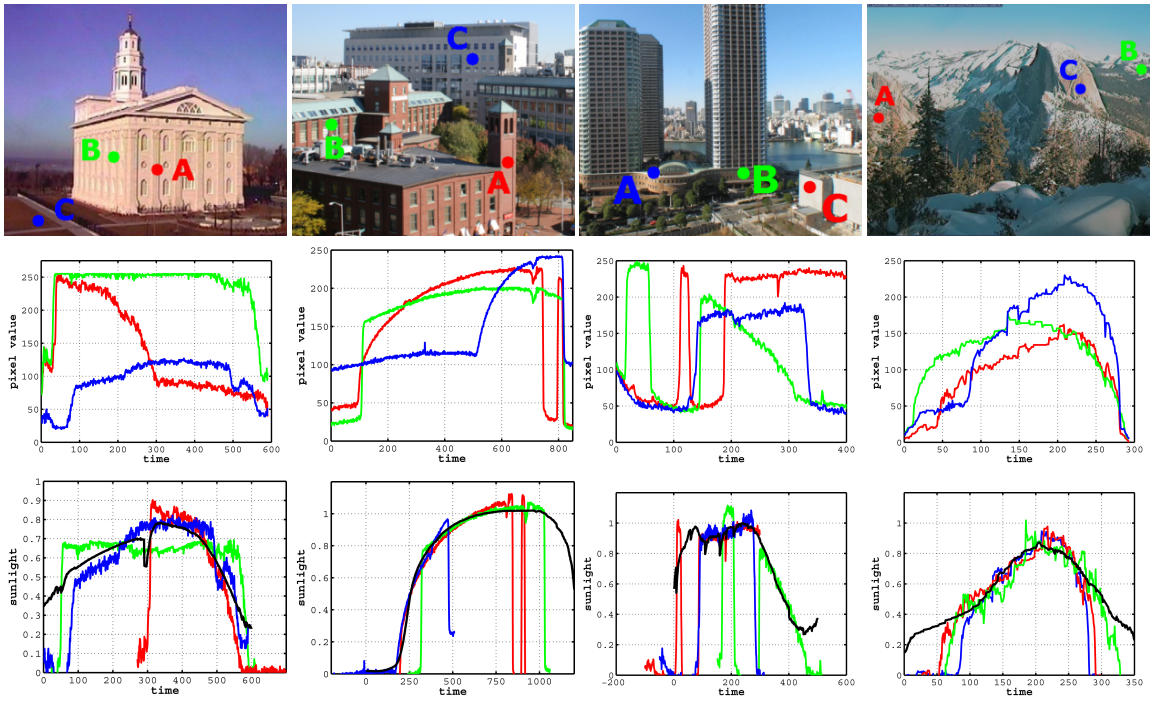
Formally, each application of ACLS decomposes an $m \times n$ *data matrix* $\mathbf{F}(t)$, where m is the number of pixels in the image and n is the number of frames in the time-lapse sequence, into the product of an $n \times k$ *weight matrix* $\mathbf{W}$ and a $k \times m$ *basis matrix* $\mathbf{H}(t)$. The algorithm takes k, the number of basis curves, as input. We set $k = 1$ for the decompositions shown in this paper. We apply ACLS in two separate steps to factor a matrix of measured spatio-temporal appearance profiles $\mathbf{F}(t)$, first factoring $\mathbf{I}_{sky}(t)$ and then solving for $\mathbf{I}_{sun}(t)$).

5.1 Skylight Factorization

Multiplying the appearance profiles $F_i(t)$ in Figure 4(a) by $(1 - S_i(t))$ yields the appearance profile of points in shadow as shown in Figure 4(b). We perform ACLS factorization using $\mathbf{F}(t)$ as the data matrix and $(1 - \mathbf{S}(t))$ as the confidence matrix to solve Equation (3). This ensures that we consider only shadowed pixels. Note that this is a different strategy from using interpolation to fill in data for unshadowed pixels, as we would need for factorization methods that do not consider confidence. However, because we have many frames in the sequence, the system of equations is highly over-constrained for a small number of basis curves. Intuitively, if we are missing data at one pixel we probably observe it at another pixel with a similar normal and appearance profile.

Figure 4(b) shows the skylight basis curve and its fit to the input data. We reconstruct the estimated $\mathbf{I}_{sky}(t)$ using Equation (3). Figure 6 shows the skylight image $\mathbf{W}_{sky}$ and a single frame of the reconstructed $\mathbf{I}_{sky}(t)$. The skylight image is related to both surface albedo and ambient occlusion, where the ambient term for a point on a surface is determined by how occluded that point is by other surfaces in the scene (i.e., the darker the pixel, the less skylight it receives or reflects). The reconstructed images $\mathbf{I}_{sky}(t)$ approximate how the scene would look throughout the time-lapse sequence in the absence of direct sunlight.

Figure 6: *Top: Skylight image. Bottom: Reconstructed sky image.*



(a) Temple (b) Square (c) Tokyo (d) Yosemite

Figure 7: *Example frames for each dataset (first row) and appearance profiles (second row) of pixels A, B, and C (red channel). In the third row are the aligned appearance profiles and estimated sunlight basis curves (black). The profiles were aligned and scaled using the estimated shifts Φ_i and $\mathbf{W}_{sun}$ for pixels A, B, and C.*

5.2 Sunlight Factorization

The reconstructed images $\mathbf{I}_{sky}(t)$ are subtracted from the original data $\mathbf{F}(t)$ and the result is clamped to 0 to form the matrix $\mathbf{I}_{sun}(t)$. Multiplying $\mathbf{I}_{sun,i}(t)$ by $S_i(t)$ yields the appearance profile of frames when the points are illuminated only by sunlight as shown in Figure 4(c). Using $\mathbf{I}_{sun}(t)$ as the data matrix and $\mathbf{S}(t)$ as the confidence matrix $\mathbf{C}$ thus ensures that only the sunlight component at every pixel is considered during the factorization. We run ACLS to solve Equation (2) and compute the basis curve $\mathbf{H}_{sun}$, the sunlight image $\mathbf{W}_{sun}$, and the shifts Φ_i.

In order to adapt ACLS to handle the time-offsets necessary to implement Equation (2), we modify the iterative update stage of the algorithm. Specifically, the original algorithm alternates between phases in which $\mathbf{H}(t)$ is held fixed while $\mathbf{W}$ is optimized using least squares, then vice versa (this is an instance of the principle of expectation maximization in inference). In order to incorporate the shifts Φ_i, we shift the entire matrix $\mathbf{H}(t)$ by $+\Phi_i$ when updating $\mathbf{W}_i$ and, similarly, shift each row i of $\mathbf{F}(t)$ by $-\Phi_i$ during updates of $\mathbf{H}(t)$. Finally, we introduce a third update phase during the iteration, in which we update Φ_i by finding, for each pixel, the shift that minimizes error. The metric used here is the same as that for updating $\mathbf{W}$ and $\mathbf{H}$, i.e., the confidence-weighted Euclidean error between the scaled and offset basis curve $\mathbf{H}(t)$ and $\mathbf{F}(t)$. As with ACLS, our modified algorithm reduces error at each iteration, and is guaranteed to converge to a (possibly local) minimum.

Figure 4(c) shows the sunlight basis curve $\mathbf{H}_{sun}(t)$ and its fit to the input data, and Figure 8 shows the sunlight image $\mathbf{W}_{sun}$ and the reconstructed image $\mathbf{I}_{sun}(t)$. The harsh black shadows in the reconstructed image $\mathbf{I}_{sun}(t)$ are similar to images taken in a vacuum without atmospheric light scattering, such as images from the moon. Figure 8 (bottom) shows the shifts Φ for the sun component. This image is a good estimate of the partial geometry (normals) of the scene. If we have time-lapse sequences from the same static viewpoint for different days, it is possible to recover more than just this one-dimensional approximation of surface normals.

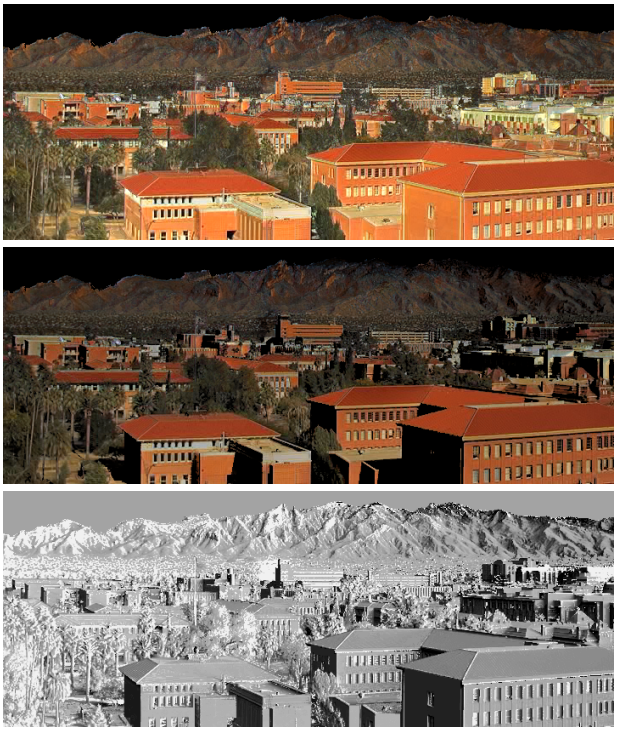


Figure 8: *Top: Sunlight image. Middle: Reconstructed sun image. Bottom: Shift map.*

Figure 9: *Original images (a) of the Square sequence compared to our reconstruction with (b) and without shadows (c).*

6 Results

The datasets for our experiments come from online webcams [2]. As shown in Table 1, the sequences are of varying length and resolutions, typically captured at intervals in the range of every 15 seconds to every 5 minutes. All of the datasets were originally compressed. We extracted the individual frames and automatically removed the regions covered by the sky using the sky-mask. All results in this paper and video were computed without sky. For some of the results in the paper and the video we composited the sky regions back into the sequences.

6.1 Reconstruction Quality

Figure 7 shows sample images from four sequences with three pixels marked on each. The figure shows the appearance profiles for the red channel of the pixels over time (similar results are obtained for green and blue). Similar to Figure 4, the profiles show shadowing in the form of discontinuities but otherwise appear similar to each other. In Figure 7 in the bottom row we show the alignment of these separate time-varying profiles using the scales $\mathbf{W}_{sun,i}$ and shifts Φ_i we estimated in our factored representation for each pixel. The black curves show the sunlight basis curves for each sequence. The scaled and time-shifted profiles match the basis curves very well.

Figure 9 qualitatively shows the accuracy of our reconstruction for the Square sequence. The reconstruction with shadows accurately matches the original images. Specific problem areas in this sequence are the streets and sidewalks due to parked cars, moving traffic, and people. Interestingly, our shadow estimation technique picks up the shadows of moving objects, which are especially visible in a few of the sequences shown in the accompanying video.

[2]Data courtesy of: Santa Catalina Mountains, Arizona - Dec 5 2006 (http://www.cs.arizona.edu/camera); Nauvoo Temple. Illinois - Dec 10 2002 (http://deseretbook.com/nauvoo/archive); Yosemite: Half-dome from Glacier Point - Dec 19 2006 (http://www.halfdome.net/); Tokyo Riverside Skyline - Jan 08 2007 (http://tokyosky.to/).

In some cases we median-filtered the shadows $\mathbf{S}_{sun}(t)$ temporally to remove flickering due to moving people, cars, etc. The bottom row in the figure shows our reconstruction without shadows. It effectively removes the "ghost" shadows of moving objects and the large shadow that is visible throughout most of the sequence.

6.2 Error Analysis

Table 1 shows the overall RMS image reconstruction errors for FTLV that were computed across all temporal frames and spatial locations. The error is quite large compared to traditional image and video compression. This is not surprising, since FTLV does not encode moving objects. To get a better qualitative handle on the reconstruction error we compare FTLV to principal component analysis (PCA), a standard matrix rank-reduction algorithm. Similar to ACLS, PCA decomposes the original spatio-temporal matrix $\mathbf{F}$ into the sum of the mean $\bar{F}$ and a product of weight matrices $\mathbf{W}$ and basis vectors $\mathbf{H}$. The number of basis vectors determines the fidelity of the reconstruction.

Figure 10 shows a plot of the PCA RMS reconstruction error versus the number of PCA terms for each of our sequences. We indicate

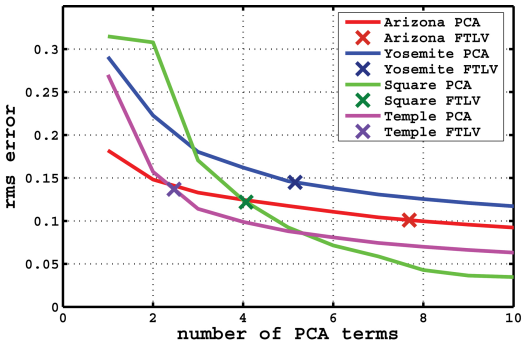


Figure 10: *PCA RMS reconstruction error vs. number of PCA terms. The symbol* $\mathbf{X}$ *indicates the RMS error of FTLV.*

Dataset	# Imgs	Resolution	RMS Error	PCA Terms	Raw	PCA	FTLV w/o Shadows	FTLV + Shadows
Arizona	610	720 × 278	14.5%	5 + 1	350 MB	1,118 kB	475 kB	1,665 kB
Temple	590	464 × 355	13.7%	2 + 1	278 MB	63 kB	40 kB	541 kB
Yosemite	293	480 × 360	10.1%	8 + 1	115 MB	1,040 kB	297 kB	2,289 kB
Tokyo	380	920 × 612	15.9%	4 + 1	353 MB	1,200 kB	868 kB	3,601 kB
Square	850	648 × 330	12.2%	4 + 1	441 MB	815 kB	765 kB	1,257 kB

Table 1: *Datasets and file sizes. Sky pixels were removed from the Raw and PCA estimates for fair comparison with FTLV.*

Figure 11: *Closeup of a frame in the Arizona sequence. Top: Original image. Middle: FTLV reconstruction with 14.5% RMS error. Bottom: PCA reconstruction with the same error using five terms plus the mean. Note the blurry shadows in the PCA reconstruction.*

the corresponding FTLV error by the symbol $\mathbf{X}$. The number of PCA terms that are necessary to achieve the same FTLV RMS error is also shown in Table 1. PCA typically needs more terms (plus the mean) since FTLV is equivalent to three terms (images). A visual comparison of the corresponding reconstructed sequences shows that FTLV is more effective in preserving sharp details, especially at shadow boundaries (see Figure 11). A reasonable reconstruction of shadows requires a large number of PCA terms (at least 15 for our datasets). In addition, PCA does not produce a meaningful description of the data. In particular, PCA allows negative values in $\mathbf{W}$ and $\mathbf{H}$, resulting in a representation whose terms cannot be edited independently.

6.3 Compression

As can be seen from Table 1, FTLV is a highly compact and efficient representation for time-lapse sequences. For these comparisons, we store the skylight, sunlight, and shift images using high-quality JPEG. The binary shadow functions are stored per pixel. For each pixel, we store the frame numbers at which shadows start and end and do LZW compression on these number-pairs. The two basis curves are stored in ASCII text files. Table 1 shows the file sizes for raw images, PCA, FTLV without shadows, and FTLV with shadows. The file size of FTLVs is dominated by the encoded shadows. Of course this is very scene dependent. Scenes with highly complex shadows (pixels go in and out of shadow many times) have multiple shadow start-end pairs and this is why scenes such as Yosemite require more storage while Temple or Square require less.

6.4 Editing and NPR

A major benefit of FTLV is that it factors the scene into physically meaningful components, each of which can be edited to create interesting effects. Edits to the shift image affect surface appearance by changing their pseudo-normals. An example of this can be seen in Figure 12, where we show closeups of three frames from the Yosemite sequence. We added the imprint of the SIGGRAPH logo to the snow in the foreground by editing the shift map. We also edited the shadows in that region to keep the logo in sunlight.

Edits to the sunlight image and curve have an effect on surface reflectance. Figure 1(d) shows an example of this for the Square sequence. We changed the surface appearance of various rooftops by changing the sunlight curve to be more specular. We changed the sunlight image to add the SIGGRAPH logo and added windows to the building in the front by editing the skylight, sunlight, and shifts. We also selectively removed shadows, for example, for the building in the background. Note that by simply editing the skylight, sunlight, or shifts we affected the entire time-lapse sequence. As can be seen in the accompanying video, the results look visually plausible and could certainly be improved with more artistic care.

In order to demonstrate the flexibility of our method, we have also experimented with non-photorealistic effects to generate stylized images and videos of the input scenes. We first transform the shift maps Φ into pseudo-normal maps by mapping shifts to angles along an arc through the sky. In order to slightly add to the plausibility of the normals, we apply a small shift to the normals based on the ratio between the computed sky and sun maps (reasoning that vertical surfaces generally receive less sky illumination than horizontal ones do). While these normals are certainly not accurate, we nevertheless expect that they are related to the true normals by some (continuous) function, and they are sufficient as input to rendering techniques such as exaggerated shading [Rusinkiewicz et al. 2006], which seek to emphasize local *differences* between normals. We generate our final NPR results by compositing the exaggerated shading with the sun and sky color maps, as well as (optionally) the shadow maps. Figure 13 shows results obtained using this technique for still images, while the supplementary video shows the result for moving scenes. Note that the computation is temporally coherent over time, leading to smooth video results (with only sharp temporal discontinuities due to the motion of shadows).

Figure 12: *Edits in the Yosemite sequence to add a logo to the snow. The edits were made in the shift ("normal map") image, and the results show plausible time-varying behavior.*

6.5 Discussion

As with any least squares factorization approach there are some practical considerations to keep in mind while applying FTLV to real-world data. These are closely tied to the behaviour of the factorization and its sensitivity to the various parameters.

6.5.1 Initialization

We have found that in practice FTLV is sensitive to the initialization of the shifts Φ but robust to the initialization of $\mathbf{W}$ and $\mathbf{H}$. We typically initialize Φ with random values in the range $[-(n/2), +(n/2)]$, where n is the number of frames. This range may vary from dataset to dataset and is dependent on the distribution of normals in the scene and the temporal sampling of the data. For example the Square sequence, with a larger span of effective normals and temporal sampling, had a larger range. For the Temple sequence, which has fewer sets of normals, the range was smaller.

Also, while FTLV is robust to small errors in shadow estimation, drastic errors will corrupt the results. The simple heuristics we use to compute shadows have worked for most of our datasets. Pixels that are always in the shadows are treated as having $\mathbf{W}_{sun} = 0$ while for pixels that are always in the sun $\mathbf{W}_{sky}$ and $\mathbf{W}_{sun}$ are estimated with additional constraints based on the sunlight and skylight intensities estimated from other pixels.

6.5.2 Computational Costs

Like the original ACLS algorithm and other least squares optimizations with similarly large amounts of data, FTLV is computationally intensive. The additional iterations on Φ_i may be computationally expensive (especially if the range of values of Φ_i that we search over is very large) but in relative terms do not add significant processing to the algorithm. Quantitatively, for the Square dataset with 180,000 pixels and 850 frames, our sunlight factorization converges in about 45 mins on a P4 3.0 GHz with 2 GB of RAM.

6.5.3 Sampling

FTLV performance is also dictated by the temporal sampling of the input time-lapse sequence. The temporal-sampling limitations on our algorithms are closely tied to the nature of scene (diffuse scenes would work with lower sampling, complex reflectances would require more data points) and camera and image quality (noise, saturation, and compression will corrupt data). In our experiments, for the Square dataset (which has fairly smooth and clean appearance profiles) we have obtained equivalent results from 60 frames instead of 850.

7 Conclusions and Future Work

FTLVs are a compact, intuitive, factored representation for time-lapse sequences that separate a scene into its reflectance, illumination, and geometry factors. They enable a number of novel image-based scene modeling and editing applications. The intuitive representation enables applications such as shadow removal, relighting, advanced image editing, and painterly rendering. As sophisticated background models FTLVs can be used to model temporal scene variations and improve tracking. Their compact nature enables compression of large time-lapse datasets.

While our current results factor scenes into single basis curves it is not difficult to imagine using multiple curves. Our intuition is that multiple basis curves would relate to different material properties (e.g., diffuse and specular curves). This will both reduce error and enable material-based separation of scene elements.

We are currently looking at methods to express images as arbitrary fractional sums of the sky and sun terms. In future work we would also like to extend FTLVs to arbitrary illumination, including indoor scenes, and scenes with participating media, e.g., a time-lapse sequence captured on a foggy day.

Another limitation is that FTLVs do not handle motion. Moving objects show up in the residue between original and reconstruction and could be added back into the sequence. However, to improve the FTLV factorization it would be beneficial to remove moving objects first, or to use an initially computed FTLV as a background model for dynamic object tracking and removal. One could then try to find a compact representation for dynamic scene elements, including clouds and the sky.

8 Acknowledgments

We would like to thank the anonymous reviewers for their insightful comments and Jennifer Roderick for proofreading the paper. Szymon Rusinkiewicz is supported by the Sloan Foundation and the National Science Foundation, grant CCF-0347427.

References

AGARWALA, A., ZHENG, K. C., PAL, C., AGRAWALA, M., COHEN, M., CURLESS, B., SALESIN, D., AND SZELISKI, R. 2005. Panoramic video textures. *ACM Trans. on Graph. 24*, 3, 821–827.

BARROW, H., AND TENENBAUM, J. 1978. *Recovering intrinsic scene characteristics from images*. Academic Press, 3–26.

BHAT, K., SEITZ, S., HODGINS, J. K., AND KHOSLA, P. 2004. Flow-based video synthesis and editing. *ACM Trans. on Graph. 23*, 3, 360–363.

BREGLER, C., COVELL, M., AND SLANEY, M. 1997. Video rewrite: driving visual speech with audio. In *Proc. of ACM SIGGRAPH*, ACM Press, New York, NY, USA, 353–360.

CHUANG, Y.-Y., AGARWALA, A., CURLESS, B., SALESIN, D. H., AND SZELISKI, R. 2002. Video matting of complex scenes. *ACM Trans. on Graph. 21*, 3 (July), 243–248.

CHUANG, Y.-Y., GOLDMAN, D. B., CURLESS, B., SALESIN, D. H., AND SZELISKI, R. 2003. Shadow matting and compositing. *ACM Trans. on Graph. 22*, 3, 494–500.

DEBEVEC, P., TCHOU, C., GARDNER, A., HAWKINS, T., POULLIS, C., STUMPFEL, J., JONES, A., YUN, N., EINARSSON, P., LUNDGREN, T., FAJARDO, M., AND MARTINEZ, P., 2004. Estimating Surface Reflectance Properties of a Complex Scene under Captured Natural Illumination. USC ICT Technical Report ICT-TR-06.2004.

GU, J., TU, C.-I., RAMAMOORTHI, R., BELHUMEUR, P., MATUSIK, W., AND NAYAR, S. 2006. Time-varying surface appearance: acquisition, modeling and rendering. *ACM Trans. on Graph. 25*, 3, 762–771.

HERTZMANN, A., AND SEITZ, S. M. 2005. Example-based photometric stereo: Shape reconstruction with general, varying brdfs. *IEEE Trans. on PAMI 27*, 8, 1254–1264.

KOPPAL, S. J., AND NARASIMHAN, S. G. 2006. Clustering appearance for scene analysis. In *Proc. of CVPR*, vol. 2, 1323 – 1330.

LAWRENCE, J., BEN-ARTZI, A., DECORO, C., MATUSIK, W., PFISTER, H., RAMAMOORTHI, R., AND RUSINKIEWICZ, S. 2006. Inverse shade trees for non-parametric material representation and editing. *ACM Trans. on Graph. 25*, 3, 735–745.

LI, Y., SUN, J., AND SHUM, H.-Y. 2005. Video object cut and paste. *ACM Trans. on Graph. 24*, 3, 595–600.

LITWINOWICZ, P. 1997. Processing images and video for an impressionist effect. In *Proc. of ACM SIGGRAPH*, ACM Press, New York, NY, USA, 407–414.

LIU, C., TORRALBA, A., FREEMAN, W. T., DURAND, F., AND ADELSON, E. H. 2005. Motion magnification. *ACM Trans. on Graph. 24*, 3, 519–526.

MATSUSHITA, Y., NISHINO, K., IKEUCHI, K., AND SAKAUCHI, M. 2004. Illumination normalization with time-dependent intrinsic images for video surveillance. *IEEE Trans. on PAMI 26*, 10, 1336–1347.

MATUSIK, W., LOPER, M., AND PFISTER, H. 2004. Progressively-refined reflectance functions from natural illumination. In *Rendering Techniques*, Eurographics Association, A. Keller and H. W. Jensen, Eds., 299–308.

MCGUIRE, M., MATUSIK, W., PFISTER, H., HUGHES, J. F., AND DURAND, F. 2005. Defocus video matting. *ACM Trans. on Graph. 24*, 3, 567–576.

NAYAR, S. K., KRISHNAN, G., GROSSBERG, M. D., AND RASKAR, R. 2006. Fast separation of direct and global components of a scene using high frequency illumination. *ACM Trans. on Graph. 25*, 3, 935–944.

NIMEROFF, J. S., SIMONCELLI, E., AND DORSEY, J. 1994. Efficient Re-rendering of Naturally Illuminated Environments. In *Fifth Eurographics Workshop on Rendering*, Springer-Verlag, Darmstadt, Germany, 359–373.

RUSINKIEWICZ, S., BURNS, M., AND DECARLO, D. 2006. Exaggerated shading for depicting shape and detail. *ACM Trans. on Graph. 25*, 3, 1199–1205.

SCHÖDL, A., SZELISKI, R., SALESIN, D., AND ESSA, I. 2000. Video textures. In *Proc. of ACM SIGGRAPH*, ACM Press, New York, NY, USA, 489–498.

SEITZ, S. M., MATSUSHITA, Y., AND KUTULAKOS, K. N. 2005. A theory of inverse light transport. In *Proc. of ICCV*, II: 1440–1447.

TOMASI, C., AND MANDUCHI, R. 1998. Bilateral filtering for gray and color images. In *Proc. of ICCV*, 839–846.

WANG, J., XU, Y., SHUM, H.-Y., AND COHEN, M. F. 2004. Video tooning. *ACM Trans. on Graph. 23*, 3, 574–583.

WANG, J., BHAT, P., COLBURN, R. A., AGRAWALA, M., AND COHEN, M. F. 2005. Interactive video cutout. *ACM Trans. on Graph. 24*, 3, 585–594.

WEISS, Y. 2001. Deriving intrinsic images from image sequences. In *Proc. of ICCV*, II: 68–75.

WINNEMÖLLER, H., OLSEN, S., AND GOOCH, B. 2006. Real-time video abstraction. *ACM Trans. on Graph. 25*, 3, 1221–1226.

YU, Y., AND MALIK, J. 1998. Recovering photometric properties of architectural scenes from photographs. In *Proc. of ACM SIGGRAPH*, ACM Press, New York, NY, USA, 207–217.

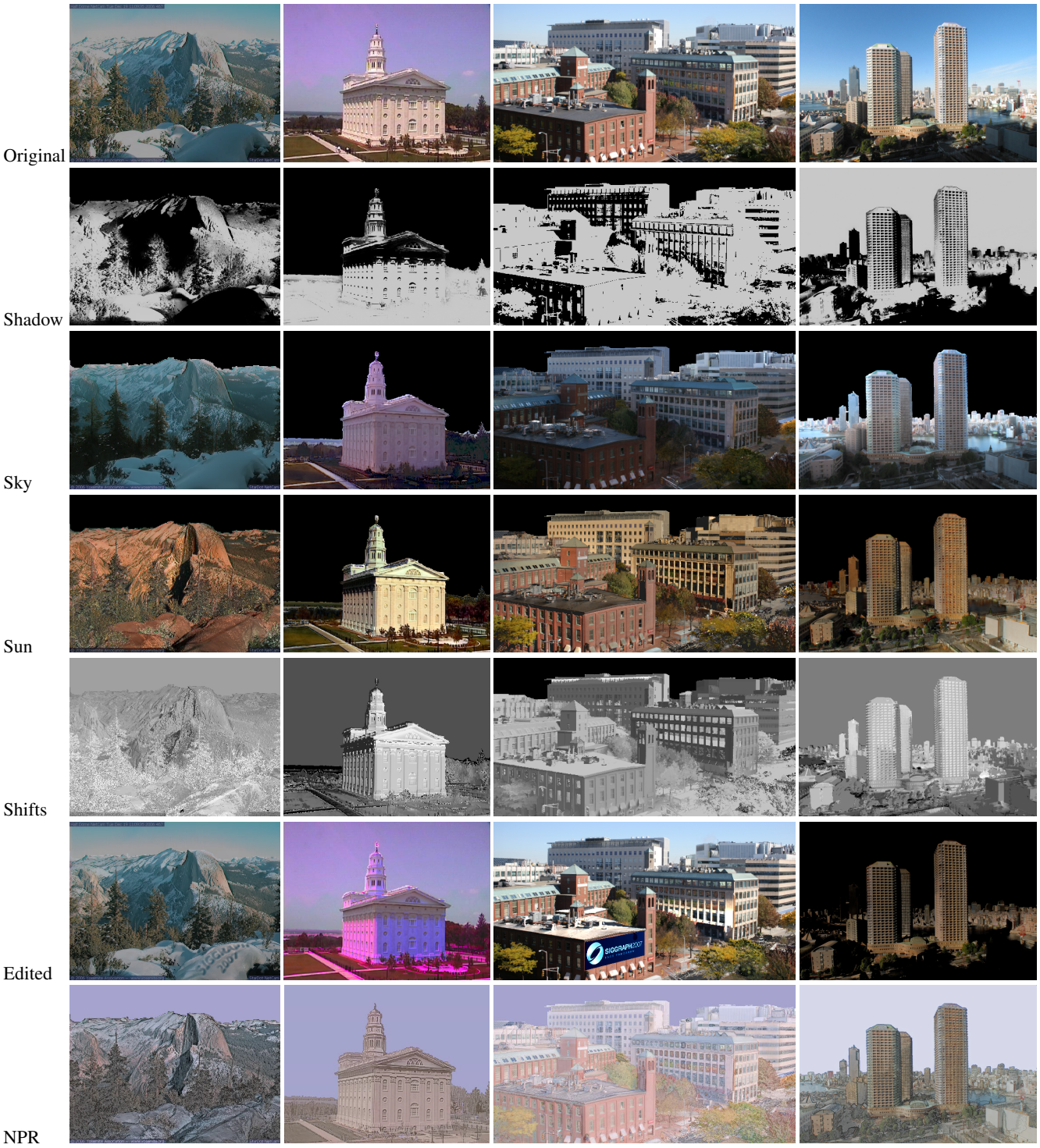


Figure 13: *Example images for the Yosemite, Temple, Square, and Tokyo sequences. FTLV essentially represents the spatio-temporal video volume with a set of shadow images (second row) and the skylight, sunlight, and shift images (next three rows). The edited examples (second to last row) show, from left to right, editing the shift image, relighting, changing reflectances, and removing the skylight. NPR examples are shown in the last row.*