Factored Time-lapse Video

Theo Turner

April 2018

1 Introduction

Time-lapse photography captures a large amount of information about the lighting of a scene over a period of time. If this data is factored into its individual components–illumination, reflectance and shadows–the scene may be computationally re-lit for the purposes of advanced image editing operations. This document details the author's attempt at an implementation of this factoring following the methodology presented in the initial proposal of factored time-lapse video by Sunkavalli et al. in 2007 [1].

2 Algorithm outline

Light factoring is programmed using MATLAB. The algorithm involves splitting scene lighting into sunlight and skylight components. It can be summarised as follows:

1. Read time-lapse video and mask sky pixels.
2. For each colour channel, independently:
   (a) Detect shadows.
   (b) Extract skylight and the skylight basis curve.
   (c) Extract sunlight and the sunlight basis curve.
3. Reconstruct images and visualise results.

Each of the above steps are detailed in the following sections. Results, including full videos of the shadow maps for each of the scenes used are attached alongside this document.

3 Reading time-lapse data

Both real and synthetic time-lapse data has been captured. Synthetic data will be discussed in a later section.

The program reads in a video file and stores each frame as a two-dimensional matrix (number of pixels $\times$ 3 colour channels). The entire time-lapse video is therefore stored as a three-dimensional matrix (number of frames $\times$ number of pixels $\times$ 3). Ground truth (masking sky pixels) is taken using a still frame from the video, manually selecting the pixels in an image editing program and creating a binary black-and-white image. Ground truth is saved as a vector in the same way as the frames.

(a) A frame from the real time-lapse.

(b) The ground truth image for this video.

The spatio-temporal volume of light data is taken as the elements of the time-lapse matrix for which the corresponding index in the ground truth image has a true value. We call this volume $F(t)$. Factoring the individual components of $F(t)$ is possible because under the assumption of a clear sky, it is equal to the sum of sky illumination $I_{sky}(t)$ and directional light from the sun $I_{sun}(t)$ under a map of shadows $S(t)$, where $S(t)$ determines if there is a contribution to illumination from the sun at a given pixel, i.e.

$$F(t) = I_{sky}(t) + S(t) * I_{sun}(t)$$

4 Detecting shadows

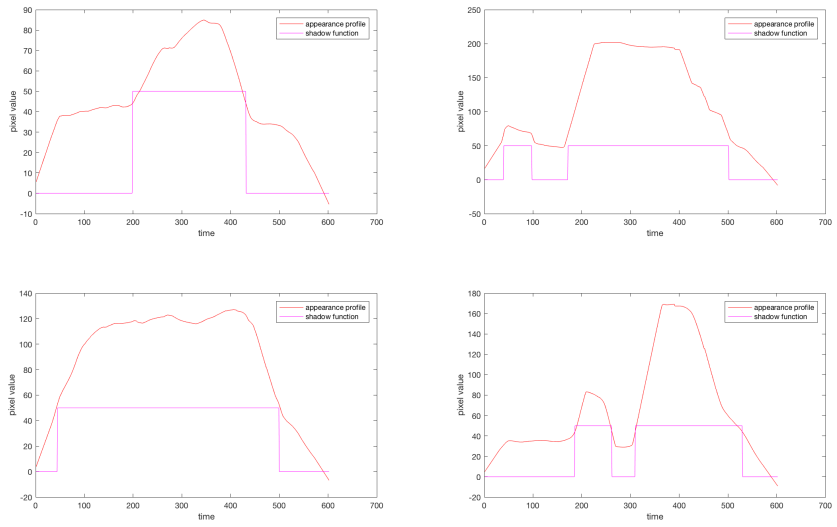
This step is done independently for each colour channel c, i.e. the three layers of the final dimension of the spatio-temporal volume are passed in one at a time. In addition, as sky pixels have been masked, the algorithm observes non-sky pixels only.

In order to prevent erratic movement in the final shadow map, the light volume is first smoothed. Also, before any shadow detection takes place, the user enters an estimation of the percentage of time any given non-sky pixel is in the shade. We call this value p.

The frames are sorted such that those with lowest overall intensity appear first. The maximum intensity for a given pixel to be deemed in the shade is set as *one and a half times the median of the first p% of frames.* This value was acquired experimentally by Sunkavalli et al. Pixels at or below this value

are marked as 0, pixels above it are marked as 1. The result is a binary matrix $S(t_c)$ of the positions of shadows over the course of the frames.

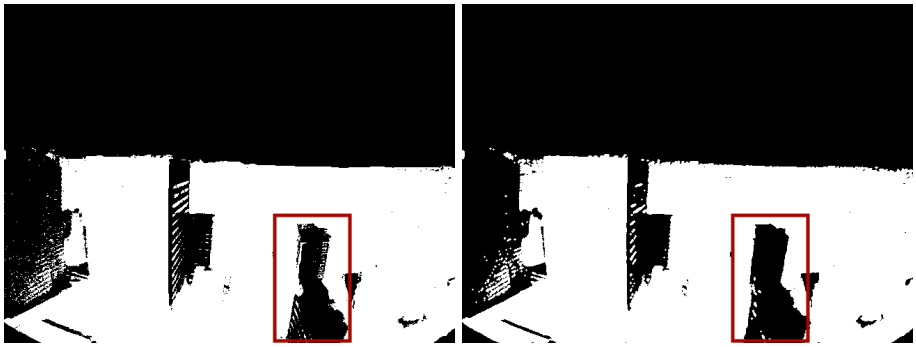
(c) Frame number 280 from the time-lapse with four points marked.



(d) The appearance profile (intensity) and estimated shadow function over the course of the frames at points A, B, C and D (left to right, top to bottom).

The shadow map may be improved by the addition of an edge-preserving filter to reduce noise. A bilateral filter adapted from papers by Tomasi and Manduchi, 1998 and Banterle et al., 2012 has been applied to achieve this,

producing the final shadow map [2, 3].



(e) Shadow map at frame number 280 without bilateral filtering.

(f) Shadow map at the same frame with bilateral filtering applied.

5 Extracting skylight and the skylight basis curve

Once again, this step is done independently for each colour channel. The component $I_{sky}(t_c)$ can be expressed as a product of skylight W_{sky} (per-pixel weights) and the skylight basis curve $H_{sky}(t_c)$ (corresponding time curve).

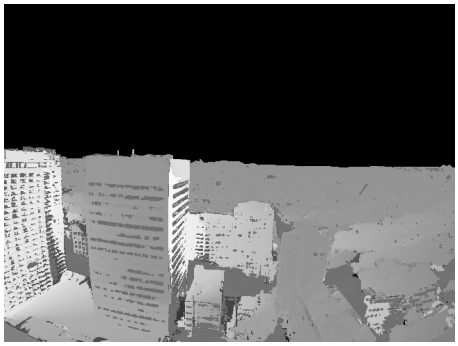
In order to factor each component, an alternating constrained least squares solution is computed on each group of the spatio-temporal volume $F(t_c)$. This factoring is adapted from a 2006 publication by Lawrence et al. [4]. Groups are determined using k-means clustering of the spatio-temporal volume and saved as a vector comprised of the centroids of the clusters, one for each of the skylight and skylight basis curve components. These vectors, called *slices*, are used to compute an alternating least squares solution for each component in turn.

Given the dimensionality of the spatio-temporal volume (observed one colour channel at a time), the alternating least squares problem can be expressed as a linear least squares problem for each row of the volume's matrix $f(t_c)$. It can therefore be computed in a standard for loop. First, the shadows are taken into account by element-wise multiplication by the complement of the shadow map (since only sky illumination is being considered). Second, the least squares solution is found: a vector v such that the norm of $\frac{1}{2}(x * v - f(t_c))$ is minimal. Here, x comes from the basis curve slice in the case of solving for skylight (a loop over the pixels) and the skylight slice in the case of solving for the basis curve (a loop over the frames). The final skylight and skylight basis curve matrices are populated using these vectors.

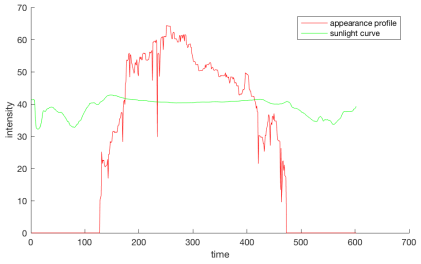
The skylight volume $I_{sky}(t_c)$ can be reconstructed using the product of W_{sky} and $H_{sky}(t_c)$.

6 Extracting sunlight and the sunlight basis curve

Once again, this step is done independently for each colour channel. Functionally, extracting sunlight and the sunlight basis curve is analogous to extracting skylight and the skylight basis curve, though with a few key differences. First, slices are constructed not from the spatio-temporal volume, but a 'sunlight volume,' the spatio-temporal volume with the skylight component subtracted. Second, shadows are taken into account by element-wise multiplication by the shadow map itself rather than its complement. Third, the per-pixel time shifts are taken into account: shifts Φ are computed using the offset in cross-correlation between the sunlight volume and sunlight basis curve slices, and updated to minimise the per-pixel euclidean error between $H(t_c)$ and $F(t_c)$ in a loop over the pixels. For each update of the sunlight component, the basis curve is translated forward by the shifts; for each update of the sunlight basis curve, the sunlight volume is translated backward by them.



(g) Shift map (normalised for improved visualisation).



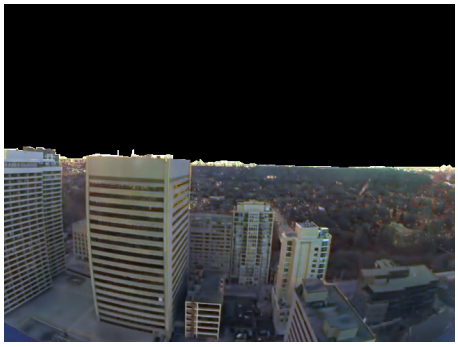
(h) Sunlight curve and appearance profile at point C.

Other than the aforementioned differences, alternating constrained least squares factorisation and reconstruction of the full sunlight component is done in the same manner as with skylight and the skylight basis curve.

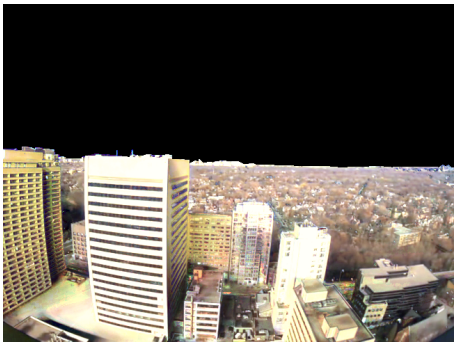
7 Reconstructing images and visualising results

Through factorisation, the following components are recovered for each colour channel c: $S(t_c)$, $I_{sky}(t_c)$, $I_{sun}(t_c)$, W_{sky}, W_{sun}, $H_{sky}(t_c)$, $H_{sun}(t_c)$ and Φ.

The time-lapse scene may be re-lit using these components. The factors for each colour channel are summed and used as pixel data for points in the image covered by the ground truth. It is possible, for example, to construct images showing how the scene would appear in the absence of sunlight or without shadows.



(i) Skylight image.



(j) Sunlight image.

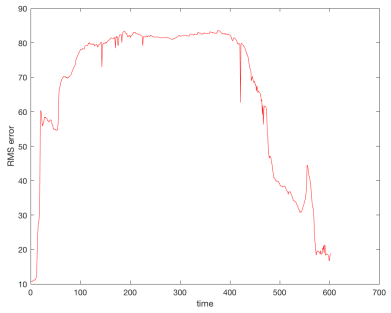
(k) Reconstructed sky image.

(l) Re-lit scene without shadows.

The reconstruction error of the algorithm can be visualised by reconstructing the full spatio-temporal volume using the original decomposition equation on each colour channel in turn:

$$F(t_c) = I_{sky}(t_c) + S(t_c) * I_{sun}(t_c)$$

The spatio-temporal volume for each colour channel is used to fill its corresponding layer in the final dimension of the full volume. Reconstruction error over time can then be plotted by taking the absolute difference between the reconstructed volume and the original volume at each frame and computing the root mean square of the differences across all pixels.



(m) Reconstruction error over time. Unsurprisingly, the algorithm is most accurate when the entire frame is in the shade and there are no moving shadows.

8 Synthetic images and high dynamic range rendering

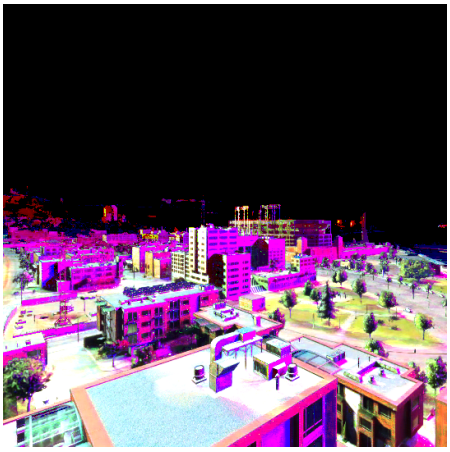
High dynamic range (HDR) rendering techniques, widespread in computer-generated imagery for film and television as well as more recent video games, often adjusts light intensity and colour at different points in a single frame to preserve detail. This presents an interesting problem for factored time-lapse video: the appearance profile of one point may be similar to that of another even if their lighting conditions are drastically different.

(n) A frame from a synthetic scene with HDR rendering [5]. Point A is in the shade, whereas point B is in sunlight. Both points are on the same building, and therefore should only appear similar in the same lighting conditions. In this frame, the lighting conditions for each point differ greatly, but the illumination is similar.

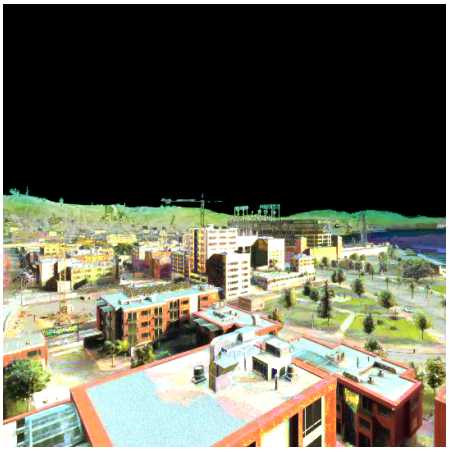
(o) Another frame from the same synthetic scene showing two points which appear to be on surfaces of the same colour. Compare this with the image on the left, which shows that the points are on surfaces of different colours.

This difficulty is most prominent in the colour of the reconstructed sunlight image. Colour adjustments performed by HDR rendering post-processes in particular can result in a final spatio-temporal volume overly 'heavy' in a single colour.

The problem has a surprisingly simple solution, however. A *colour weighting profile* $C(t)$ can be obtained by averaging the pixel intensities in the individual colour channels for each frame. Using this profile, the reconstructed spatio-temporal volume can be re-weighted such that the three colour channels illuminate the scene equally. This is achieved by using scalar multiples on the matrices specified by the first dimension (frame number). The results of this re-weighting, though not perfect, appear significantly more realistic than those of an unweighted reconstruction.



(p) Sunlight image with regular component reconstruction.



(q) Sunlight image with reconstruction weighted by a colour weighting profile.

9 Conclusion

Factoring time-lapse video is not only useful for advanced image editing operations, but may serve as a background model for temporal scene variations. Notably, through extraction of information about shadows, reflectance and illumination, the algorithm can construct an estimate of surface normals. It is, however, memory intensive and unable to handle object motion. Nevertheless, given the ease of implementation and robustness of factored time-lapse video, it proves to be a more than satisfactory approach to dynamic scene re-lighting.

References

[1] K. Sunkavalli, W. Matusik, H. Pfister, and S. Rusinkiewicz, "Factored time-lapse video," in *ACM Transactions on Graphics (TOG)*, vol. 26, no. 3. ACM, 2007, p. 101.

[2] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Computer Vision, 1998. Sixth International Conference on.* IEEE, 1998, pp. 839–846.

[3] F. Banterle, M. Corsini, P. Cignoni, and R. Scopigno, "A low-memory, straightforward and fast bilateral filter through subsampling in spatial domain," in *Computer Graphics Forum*, vol. 31, no. 1. Wiley Online Library, 2012, pp. 19–32.

[4] J. Lawrence, A. Ben-Artzi, C. DeCoro, W. Matusik, H. Pfister, R. Ramamoorthi, and S. Rusinkiewicz, "Inverse shade trees for non-parametric

material representation and editing," *ACM Transactions on Graphics (TOG)*, vol. 25, no. 3, pp. 735–745, 2006.

[5] *Watch Dogs 2 with added HDR rendering post-process effects.* Ubisoft Montreal, 2016.