Interactive intrinsic video editing

Nicolas Bonneel, Kalyan Sunkavalli, James Tompkin, Deqing Sun, Sylvain Paris, Hanspeter Pfister

Interactive Intrinsic Video Editing

Nicolas Bonneel[1,2*] Kalyan Sunkavalli[3] James Tompkin[2] Deqing Sun[2] Sylvain Paris[3] Hanspeter Pfister[2]
[1]CNRS-LIRIS [2]Harvard School of Engineering and Applied Sciences [3]Adobe Research

(a) Input video (b) Intrinsic decomposition (c) Editing the reflectance only

Figure 1: *A video sequence (a) is interactively decomposed into temporally consistent components for reflectance (b, top) and illumination (b, bottom). Now, editing the textures in the reflectance image does not affect the illumination (c): changes to the brick walls, the roof tiles, and the pathway leading up to the building all maintain the complex illumination of the light through the trees. We encourage readers to zoom into this figure to see the details in our results, and refer them to the accompanying video to see the temporally consistent nature of our decomposition.*

Abstract

Separating a photograph into its reflectance and illumination intrinsic images is a fundamentally ambiguous problem, and state-of-the-art algorithms combine sophisticated reflectance and illumination priors with user annotations to create plausible results. However, these algorithms cannot be easily extended to videos for two reasons: first, naïvely applying algorithms designed for single images to videos produce results that are temporally incoherent; second, effectively specifying user annotations for a video requires interactive feedback, and current approaches are orders of magnitudes too slow to support this. We introduce a fast and temporally consistent algorithm to decompose video sequences into their reflectance and illumination components. Our algorithm uses a hybrid ℓ_2-ℓ_p formulation that separates image gradients into smooth illumination and sparse reflectance gradients using look-up tables. We use a multi-scale parallelized solver to reconstruct the reflectance and illumination from these gradients while enforcing spatial and temporal reflectance constraints and user annotations. We demonstrate that our algorithm automatically produces reasonable results, that can be interactively refined by users, at rates that are two orders of magnitude faster than existing tools, to produce high-quality decompositions for challenging real-world video sequences. We also show how these decompositions can be used for a number of video editing applications including recoloring, retexturing, illumination editing, and lighting-aware compositing.

CR Categories: I.3.8 [Computer Graphics]: Applications I.4.8 [Image Processing and Computer Vision]: Scene Analysis—Color,Shading

Keywords: intrinsic decomposition, reflectance, illumination, video processing, recoloring, compositing

Links: DL PDF

*e-mail:nbonneel@seas.harvard.edu

1 Introduction

A number of image editing operations, including recoloring, relighting, white balancing, and texture editing, require the materials properties of the objects in the photograph, and the illumination in the scene to be handled independently. This requires the separation of an image into a product of reflectance and illumination layers: the *intrinsic decomposition* problem. This is a particularly challenging decomposition because the effects of reflectance and illumination are combined into a single observation, which makes the inverse problem of separating them severely ill-posed.

Nonetheless, significant progress has been made on this problem recently, and usable decompositions of real-world scenes have been obtained by incorporating additional data such as depth maps [Lee et al. 2012; Barron and Malik 2013; Chen and Koltun 2013], multiple images of the same scene [Weiss 2001; Laffont et al. 2012], or sophisticated priors such as reflectance sparsity [Gehler et al. 2011] and non-local texture cues [Shen et al. 2008; Garces et al. 2012; Laffont et al. 2012; Zhao et al. 2012]. Incorporating these additional terms typically result in sophisticated algorithms which take minutes to solve for a single image. Recent work has extended these approaches to video [Ye et al. 2014], but the computation time for short sequences of video is measured in hours.

In addition, these approaches may require scene-specific parameter tuning, hence requiring multiple resolutions of the problem until a satisfactory set of parameters is found. In the worst case, they may simply fail to produce the correct result, because the intrinsic decomposition problem is fundamentally ambiguous. Difficult cases can only be resolved with user guidance [Bousseau et al. 2009; Shen et al. 2011]. Herein lies a problem. The computation cost of state-of-the-art intrinsic image algorithms is prohibitively high for videos,

and so artists suffer long turnaround times while iterating parameters and editing. To achieve interactive editing, we need a significant increase in speed from minutes per frame to seconds per frame.

We introduce a new algorithm designed for the fast intrinsic decomposition of videos that enables interactive reflectance and illumination editing. The centerpiece of our approach is a hybrid $\ell_2 - \ell_p$ formulation that enforces a smooth prior on illumination gradients and a sparse prior on reflectance gradients and can be solved efficiently using look-up tables. We reconstruct the reflectance and illumination from these gradients by adding spatial and temporal constraints; the former to disambiguate low-frequency variations in the illumination, and the latter to enforce temporal coherence on the reflectance. Users can resolve further ambiguities by scribbling on the video to indicate regions of constant reflectance and illumination, and receive feedback immediately. The interactive speeds of our solver are the result of two design decisions; first, by largely working in the gradient domain, we ensure that most dependencies are between local pixels, leading to more efficient memory accesses and better optimizations [Ragan-Kelley et al. 2012], and second, we design a multi-scale parallelized solver that converges quickly by taking advantage of the inherent smoothness in the illumination.

We state our contributions explicitly:

1. We introduce a fast algorithm to separate image gradients into smooth illumination and sparse reflectance gradients using a hybrid ℓ_2–ℓ_p optimization that can be solved very efficiently using precomputed look-up tables.
2. We present a technique to reconstruct the reflectance and illumination from these gradients while enforcing spatial and temporal smoothness using a fast multi-scale parallelized solver that produces high-quality, temporally consistent, intrinsic video decompositions.
3. We describe a set of scribbles that let users control the results with interactive feedback, and a technique to propagate this input in space and time to limit the amount of interaction that users need to do.
4. We demonstrate complex video editing applications at interactive rates, including recoloring, retexturing, lighting editing, and shadow compositing.

Together, our contributions provide, for the first time, an interactive system for intrinsic video decompositions, which vastly improves turnaround times for complex video editing operations.

1.1 Related Work

Intrinsic images Barrow and Tenenbaum [1978] introduced the notion of intrinsic images to separate the contributions of illumination and reflectance. Since then, many techniques have been proposed to solve this problem. We refer to the survey of Grosse et al. [2009] for a general overview of the topic.

Intrinsic image decompositions typically attempt to separate image gradients into reflectance and illumination gradients. The classic Retinex algorithm [Land et al. 1971] does this by simply thresholding the image gradients. Subsequent work has expanded on this idea by accounting for color variations [Grosse et al. 2009] and by learning a classifier to do the separation [Tappen et al. 2005]. All these approaches only consider the observed intensities in a local neighborhood. While that makes these techniques very fast, the quality of the results is still limited. The quality of the decompositions can be improved by imposing priors on non-local reflectance [Shen et al. 2008; Gehler et al. 2011; Shen et al. 2011; Zhao et al. 2012; Bell et al. 2014], on gradient distributions [Li and Brown 2014; Shen and Yeo 2011], on the number of reflectance colors [Shen and Yeo 2011], or on the underlying scene geometry and illumination [Barron and Malik 2012]. Alternatively, the conditioning of the problem can be improved by leveraging multiple images captured under varying illumination [Weiss 2001; Matsushita et al. 2004; Laffont et al. 2012] or view directions [Laffont et al. 2013], or by making use of depth information captured with RGBD cameras [Lee et al. 2012; Barron and Malik 2013; Chen and Koltun 2013].

While these techniques produce high quality results, they do so at a high computational cost, making them impractical for video sequences. In addition, most typical video sequences do not exhibit illumination changes and one cannot assume a depth channel is available in general. Bousseau et al. [2009] allow users to guide the decomposition using scribbles (similar to work on interactive material modeling [Dong et al. 2011]), which Carroll et al. [2011] use to interactively separate diffuse colored interreflections. However, because their underlying solver uses medium-size image stencils, it takes more than 10 seconds to decompose a half-megapixel image which is too slow for interactive video editing. In addition, most of the techniques have been developed for single images (or static scenes) and do not cope with temporal consistency when applied to uncontrolled video sequences.

Illumination editing in videos Our approach relies on intrinsic decompositions of videos similarly to Lee et al. [2012] but does not require a depth channel. We instead rely on user guidance. Other applications manipulate illumination in videos without actually decomposing the data. For instance, Farbman and Lischinski [2011] stabilize the white balance of consumer-grade sequences, and Bonneel et al. [2013] transfer the color grading of movies onto other videos. These techniques are designed for specific applications. In comparison, we seek a more versatile intrinsic decomposition that enables several applications such as recoloring and lighting editing.

The work closest to ours is the concurrently proposed intrinsic video algorithm of Ye et al. [2014]. While they rely on user input like us, the annotations are specified only on the first frame and then propagated to subsequent frames. The technique takes one minute to process a half-megapixel frame, and so takes many hours for short sequences. If there is an error in the decomposition, and additional scribbles are needed, the turn-around time for an artist is very long. In contrast, our technique processes images of the same resolution in half a second. We decompose an optimization over gradients using L_p norms to one approximated by look-up tables, similar to Krishnan et al. [2009]. This enables truly interactive video decomposition for the first time.

2 Efficient Intrinsic Decomposition

In this section, we describe our algorithm to efficiently decompose an input video I into a illumination layer S and a reflectance layer R. We assume that the illumination is monochromatic (i.e the illumination in the scene has a constant color). The observed intensity at pixel x is then given by:

$$\mathbf{I}(x) = S(x)\ \mathbf{R}(x), \tag{1}$$

where $\mathbf{I}$ and $\mathbf{R}$ are RGB-vectors and S is a scalar.

The core of our algorithm is a hybrid ℓ_2–ℓ_p energy formulation in the gradient domain that separates image gradients into reflectance and illumination gradients by enforcing a sparsity prior on reflectance values and a smoothness prior on illumination. We reconstruct the reflectance and illumination layers from these separated gradients. We enforce that pixels with similar chrominance have similar reflectance to disambiguate the decomposition non-locally. We achieve temporal consistency by enforcing a causal smoothness on the reflectance

along the time dimension. We also enable user control via a set of scribbles that adds constraints to our reconstruction process. We automatically propagate these constraints to reduce the amount of user interaction required. Finally, we combine all these constraints into a linear system that we solve using a multi-scale parallelized solver. The rest of this section presents the details of each step.

2.1 Hybrid ℓ_2–ℓ_p Gradient Separation

For the sake of efficiency, we perform most of the computation on single-channel luminance images. We define the image luminance as the geometric mean of the individual RGB components, i.e., $I = (\mathbf{I}_r\,\mathbf{I}_g\,\mathbf{I}_b)^{1/3}$, and reflectance luminance R, similarly, as the geometric mean of the reflectance components. This gives us the relation $I(x) = R(x)\;S(x)$, which we solve for R and S. Finally, the color reflectance $\mathbf{R}(x)$ can be estimated as $\mathbf{I}(x)/S(x)$.

We work in the log domain to transform the image formation into a sum: $\log I(x) = \log S(x) + \log R(x)$. This has the added advantage that we do not have to constrain the solution to be strictly non-negative (as reflectance and illumination should be). Then, we formulate our approach in the gradient domain. For this, we introduce lowercase variables to represent logarithmic gradients, e.g., $i = \nabla \log I$. With this notation and the image formation model, we can write: $i(x) = s(x) + r(x)$ and express this constraint as a least-squares energy term: $\|i(x) - s(x) - r(x)\|^2$.

This term alone is ambiguous since only i is known, and both s and r are unknown. To address this issue, we add priors on s and r. We assume that reflectance values are sparse [Omer and Werman 2004; Hsu et al. 2008], i.e., that scenes are mostly made up of objects of constant colors separated by hard boundaries. This is typically modeled using a ℓ_p term on the gradients with $p < 2$, i.e., with our notation: $\|r(x)\|^p$. Low p values assume very sparse reflectance gradients, while $p = 2$ would assume normally distributed reflectance gradients over the image. We also assume that illumination exhibits smoother variations due to illumination on curved surfaces and soft shadows for instance [Land et al. 1971]. We model this prior with a ℓ_2 term on illumination gradients to favor a denser distribution of gradient values, i.e.: $\|s(x)\|^2$. We put all the terms together to form the energy E:

$$E(s,r) = \sum_x \|i - s - r\|^2 + \lambda_s\|s\|^2 + \lambda_r\|r\|^p \qquad (2)$$

where λ_s and λ_r control the influence of the priors on s and r (Figure 3) ; dependencies on pixel x have been omitted for conciseness.

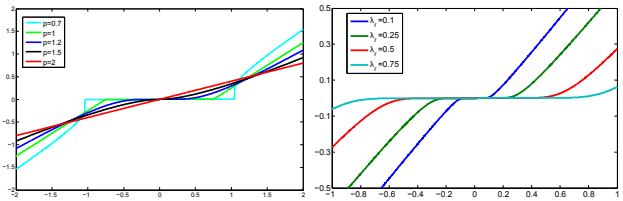
Minimizing the energy Solving mixed-norm optimization problems as Equation 2 requires time-consuming combinatorial approaches in general. However, we can do significantly better in our specific case. Inspired by Krishnan et al. [2009], we will show that we can minimize Equation 2 using a simple look-up table.

Note that this formulation treats the horizontal and vertical components of the gradients independently. The formulations we derive below, can therefore be applied to each axis separately. First, we derive an expression for s in terms of r by minimizing the energy $E(s,r)$. We do this by differentiating $E(s,r)$ with respect to the unknown s and setting the derivative to 0 giving us the relation:

$$s \;=\; \frac{i-r}{1+\lambda_s} \qquad (3)$$

Substituting the expression for s from Equation 3 into Equation 2 and re-arranging the terms gives us:

$$E(r) = \sum_x \frac{\lambda_s}{1+\lambda_s}\|i - r\|^2 + \lambda_r\|r\|^p \qquad (4)$$



(a) Varying p ($\lambda_s = \lambda_r = 0.5$) (b) Varying λ_r ($p = 1.1$ and $\lambda_s = 0.5$)

Figure 2: *Behavior of the look-up table $lut_{\lambda_s,\lambda_r,p}(\tilde{r}_k)$ (Eq. 6); $\tilde{r}_k$ varies along the x-axis. The p variable controls the smoothness of the separation – at values closer to 1 it starts approximating a clipping function that removes smaller gradients (a). The weights λ_s and λ_r pull the function in different directions, with a higher λ_r making it more like a clipping function, and a larger value of λ_s making the function smoother.*

To minimize this expression, we use Iterative Re-weighted Least Squares to replace the ℓ_p term [Björck 1996, § 4.5] with a weighted ℓ_2 term. This amounts to solving a quadratic cost function to construct a series of solutions, $\tilde{r}_k$, that progressively gets closer to the optimal solution $r_\star$. At each iteration, the estimate at $k+1$ is obtained by minimizing the following least-squares problem:

$$\sum_x \frac{\lambda_s}{1+\lambda_s}\|i - \tilde{r}_{k+1}\|^2 + \lambda_r w_{k+1}\|\tilde{r}_{k+1}\|^2 \qquad (5a)$$

$$\text{with } w_{k+1} = \frac{p}{2}|\tilde{r}_k|^{p-2} \qquad (5b)$$

Differentiating this cost function and setting it to 0 gives us a closed form expression for $\tilde{r}_{k+1}$ as a function of $\tilde{r}_k$:

$$\tilde{r}_{k+1} = \frac{2\lambda_s i}{2\lambda_s + \lambda_r(1+\lambda_s)p|\tilde{r}_k|^{p-2}} \qquad (6)$$

Iterating this formula converges to a minimizer of the mixed-norm energy (Eq. 4) and we use $\tilde{r}_K$ with a large enough K as an approximation of the solution $r_\star$. That is, in practice, given i, λ_s, and λ_r, we evaluate $\tilde{r}_K$ by iterating Equation 6 K times. We observed that $K = 100$ was sufficient in all our experiments. Further, given λ_s and λ_r, the output r value depends only on i. When λ_s and λ_r are fixed, we use this property to precompute the look-up table $\text{lut}_{\lambda_s,\lambda_r}(\tilde{r}_k)$ for varying values of $\tilde{r}_k$ and store it in a look-up table that can be accessed very efficiently at run time. We can also build a higher-dimensional table to enable varying parameters.

Taking chrominance variations into account The energy function in Equation 2 encourages sparse but large reflectance gradients and dense but smaller illumination gradients. This does not account for illumination effects like shadows that can lead to large gradients. However, shadows typically do not give rise to large variations in *chrominance* values; these are more likely to be caused by changes in reflectance [Grosse et al. 2009]. Similar to them, we threshold illumination gradients based on chrominance gradients. Specifically, we set the shading gradient to zero whenever the chrominance of the image gradient is above a user-specified threshold T^{chr}.

Discussion Figure 2 illustrates the form the function $\text{lut}_{\lambda_s,\lambda_r}(\tilde{r}_k)$ takes for different values of λ_s, λ_r, and p. For $p \approx 0.5$, this function approximates the thresholding function that the Retinex algorithm uses to separate image gradients into reflectance and illumination gradients. This suggests that our look-up table is a generalization of the Retinex algorithm. However, unlike the Retinex algorithm, our

(a) Baseline decomposition (b) Increased ℓ_2 weight (c) Increased ℓ_p weight (d) Increased p

Figure 3: *This figure demonstrates the effect of the different parameters in Equation 6 on the resulting reflectance (top) and illumination (bottom). Increasing the ℓ_2 weight (b) leads to a blurry illumination, but the reflectance is not sparse, e.g., the smooth variations on the girl's face; increasing the ℓ_p weight (c) pushes stronger gradients to the illumination component, e.g., the illustration on her shirt. Changing the norm of the reflectance term from a sparse $p = 0.4$ to a denser $p = 1.2$ makes the decomposition smoother albeit with less separated gradients (d). This comparison is performed without the non-local term and color handling.*

look-up table is a softer function that allows for non-zero reflectance and illumination gradients at the same pixel. As shown in Figs. 7 and 4, this leads to higher quality results at depth discontinuities.

Our model does not strictly enforce the image formation model and set $s = i - r$. If we did so, the energy becomes $\sum_x \lambda_s \|i - r\|^2 + \lambda_r \|r\|^p$. While this formulation is simpler, this system is stiffer because it must exactly adhere to the image formation model and can lead to artifacts. Enforcing this as a soft constraint allows our optimization to handle deviations resulting from more complex image formation. We carefully compared the results produced by the two approaches, and found that the main improvement comes from Equation 3 that favors a smooth illumination where the observed chromaticity is smooth, even though it may violate the image formation model (e.g., see Figure 4).

2.2 Reconstructing the Layers

For given values of λ_r and λ_s, we precompute $\mathrm{lut}_{\lambda_r,\lambda_s}(i)$. Then, for each pixel, we estimate the horizontal and vertical gradients of r by applying $\mathrm{lut}_{\lambda_r,\lambda_s}$ twice, once for each axis. We use Equation 3 to recover s. Then, we solve for S' that satisfies these gradients at every pixel x and time t by minimizing the term:

$$E_p(S'_t) = \sum_x \|\nabla S'_t(x) - s_t(x)\|^2, \tag{7}$$

and exponentiate it to get the illumination layer $S = \exp(S')$. Finally, we estimate the color reflectance layer $\mathbf{R}$ using the image formation model (Eq. 1). This approach has two advantages. First, reconstructing the single-channel S is more efficient than solving for each color channel of the reflectance $\mathbf{R}$. Second, minimizing the ℓ_2-norm on the gradients of S as we do in Equation 7 produces better results than doing so for reflectance because it introduces low-frequency residuals when the gradient field is not integrable. Such residuals are more likely to be innocuous in the illumination layer that contains smooth variations, but would be more conspicuous in the reflectance layer that is expected to be piecewise constant.

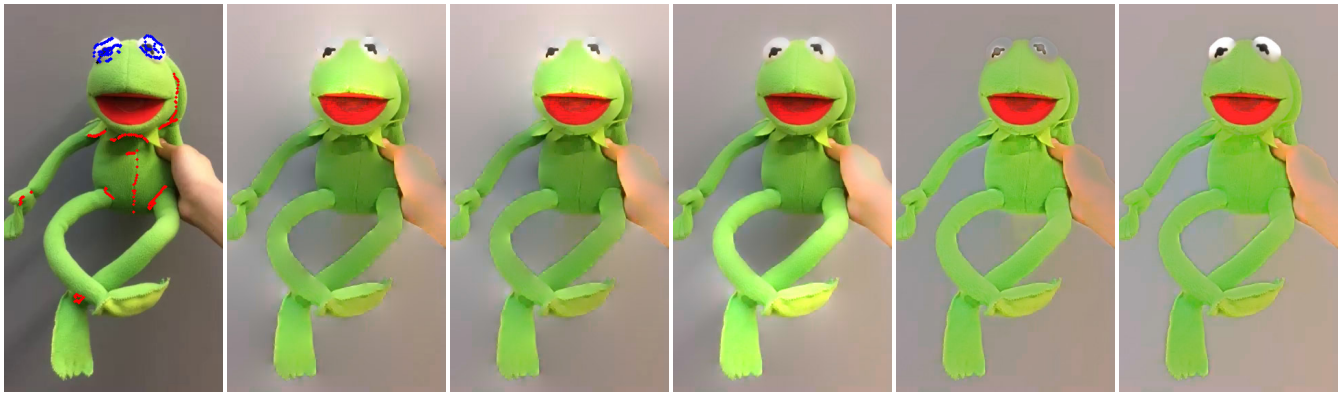
Spatial reflectance constraints While our hybrid $\ell_2 - \ell_p$ formulation leads to a simple and elegant method for separating image gradients into reflectance and illumination gradients, reconstructing the reflectance and illumination from them directly, i.e, by minimizing Equation 7, can lead to results with low-frequency errors. Previous work has addressed this by incorporating non-local reflectance constraints, typically by enforcing that pixels with similar chrominances have similar reflectances [Shen et al. 2008; Zhao et al. 2012]. Introducing these non-local constraints leads to denser linear systems and significantly slows down the decomposition [Zhao et al. 2012], resulting in a solution computed in 6 seconds to one minute on our examples. To avoid this, we incorporate a few non-local constraints that address the low-frequency errors while adding negligible overhead to the linear system we solve.

We globally cluster the chrominance of every pixel in the video into a set of C clusters using k-means. For every frame, we select a pixel whose chrominance is closest to each cluster centroid – this pixel is used as a representative for that cluster in that frame. For every pixel in a frame, we enforce its reflectance to remain close to the reflectances of the N nearest cluster representatives, with a weight w dependent on its distance to the representative in chrominance space. We then construct the following non-local regularization term:

$$\begin{aligned} E_{nl}(S'_t) &= \sum_x \sum_{n=1}^{N} w(x, x_n) \|\log R_t(x) - \log R_t(x_n)\|^2 \\ &= \sum_x \sum_{n=1}^{N} w(x, x_n) \|S'_t(x) - S'_t(x_n) \\ &\qquad - \log I_t(x) + \log I_t(x_n)\|^2, \end{aligned} \tag{8}$$

where x_n is the representative coordinate of the n^{th} nearest cluster in chrominance space. We adopt the definition of the chrominance as $\mathbf{C}_k = \log(\mathbf{R}_k) - \sum_k \log(\mathbf{R}_k)$ for the three color channels [Grosse et al. 2009]. In our implementation, $C = 8$ and $N = 2$, and $w(x, x_n) = \exp(-\|\mathbf{C}(x) - \mathbf{C}(x_n)\|^2/100)$. This adds only $N = 2$ non-zero terms to the linear system for every pixel. In addition, the fact that all pixels in a frame are connected to the same set of representative pixels makes it easy to cache their values, and speeds up our solver.

Temporal constraints Applying the previous technique frame by frame can lead to results with unpleasant flickering artifacts. However, the reflectance of a scene is typically temporally coherent and we would like to enforce this in our optimization. One way to do this would be to add a temporal smoothness term to our formulation and solve for the decomposition on the entire space-time volume.



(a) Input frame and scribbles (b) Retinex (c) Transfer function without flexibility (d) Full transfer function (e) Non-local term added (f) User input added

Figure 4: *We evaluate the use of the different terms in our optimization. The hard thresholding of Retinex-based methods (b) and enforcing the image formation model strictly (c) lead to aliasing artifacts at object boundaries which increases temporal inconsistencies (Fig. 6). Our $\ell_2 - \ell_p$ decomposition that allows deviations from the image formation model is smoother (d) especially at object boundaries. Adding the non-local constraints reduces low frequency artifacts (e) but might have issues at pixels with ambiguities (like the black and white eyes). The user can refine the result in these areas using a few strokes (e); constant reflectance scribbles are shown in red, constant illumination in blue.*

However, the size of the problem makes the computation cost and the memory requirements for the solver prohibitively large, precluding an interactive solution. In our work, we opt for an intermediate approach that enforces temporal smoothness only between adjacent frames. More specifically, while solving for the intrinsic decomposition at a particular frame, we add a temporal regularization that keeps the solution close to the result from the previous frame. This has been shown to lead to globally consistent results for other video processing applications [Paris 2008]. In addition, solving for the decomposition in chronological order is a user-friendly option because it ensures that frames that have already been annotated by the user are not affected by subsequent strokes.

We name $u_t(x_t)$ the forward optical flow, i.e., the pixel x_t at frame t moves to $x_{t+1} = x_t + u_t(x_t)$ at frame $t+1$. To enforce the temporal coherence of the reflectance component, we define a regularization term that moves the solution of the current frame to the flow-advected solution from the previous frame:

$$\begin{aligned} E_{t+1}(S'_{t+1}) &= \sum_x \|\log R_{t+1}(x_{t+1}) - \log R_t(x_t)\|^2 \qquad (9) \\ &= \sum_x \|S'_{t+1}(x_{t+1}) - S'_t(x_t) - \log I_{t+1}(x_{t+1}) \\ &\quad + \log I_t(x_t)\|^2. \end{aligned}$$

The resulting linear system has the the same size as for a single frame but leads to globally coherent results. We encourage readers to evaluate the temporal smoothness in the accompanying video.

Multi-scale solver We combine the standard gradient reconstruction energy with the non-local constraints and the temporal coherence term to get the combined error term:

$$E(S'_t) = E_p(S'_t) + \lambda_{nl} E_{nl}(S'_t) + \lambda_t E_t(S'_t), \qquad (10)$$

where λ_{nl} and λ_t control the weights given to the non-local and temporal constraints respectively. λ_t is modulated by per-pixel optical flow confidence, measured as the distance between the forward and backward optical flows u_t and $\tilde{u}_{t+1}$: $\lambda'_t = \lambda_t \|\tilde{u}_{t+1}(x_{t+1}) + u_t(x_t)\|$. We solve for the illumination that minimizes this energy at every time instant, starting with the first frame of the video and moving onto each subsequent frame.

Minimizing the energy term in Equation 10 leads to a sparse linear system in terms of the per-frame illumination S'_t. While this linear system has the size of the number of pixels in each frame, each row of this linear system has only $N+4$ off-diagonal non-zero entries (corresponding to the gradient and non-local terms respectively); temporal terms only affect the diagonal), and can be efficiently solved using parallelized Jacobi iterations. However, Jacobi iterations tend to resolve low frequencies very slowly, and we resolve this by using a multi-scale solver. We build a pyramid representation for each frame and construct the gradient, non-local, and temporal constraints at each level. The coordinates of the representative pixels of non-local constraints are adjusted to adapt to the pyramid resolution. We solve for the illumination at the coarsest level of the pyramid using a direct solver, and then upsample this result to the higher resolution and use it as the initialization to the Jacobi solver. We progressively refine this solution at the higher resolutions using the Jacobi solver, until we have computed the illumination at the full resolution of the video. This multi-scale approach takes advantage of the inherent smoothness of the illumination layer, and significantly speeds up the convergence of the sparse solver without the need for a full algebraic multigrid solver. We found that this approach converges significantly faster than Successive Over Relaxation and Conjugate Gradient methods, as well as direct LDLT solvers that take approximately 10 seconds per frame.

3 User-guided Refinement

While our hybrid gradient separation algorithm in combination with the non-local and temporal constraints produces reasonable results in many cases, real-world videos exhibit many challenging ambiguities that we address using user input. In order to support this, we provide scribbles to users so that they can refine the results.

User strokes We provide users with scribble tools to specify constraints that satisfy two requirements: they are easy for a user to specify in a video frame, and they have a limited effect on the computational complexity of the solver. To this end, we specify the user constraints in the *gradient* domain, ultimately only altering the right hand side of existing equations (Eq. 10).

We use two strokes, for *constant reflectance* and *constant illumina-*

(a) Reflectance (b) Illumination

Figure 5: *We compare our technique (third row) to the decompositions of Shen et al. [2011] (first row) and Zhao et al. [2012] (second row). Shen et al. has brushes and takes 6 min. while Zhao et al. is automatic and takes 6 s. Our method takes 0.5 s per frame, and provides a more satisfactory decomposition for our applications.*

tion, that specify that the gradient of the reflectance (and illumination, respectively) at those points is 0, i.e.: $r_t(x) = 0$ and $s_t(x) = 0$. These constraints are straightforward to apply to s and r, and do not require us to perform the gradient separation again.

Propagating strokes To speed up user interaction, we propagate user strokes both spatially and temporally to similar pixels using coherency-sensitive hashing [Korman and Avidan 2011] (CSH). As mentioned before, we only propagate user strokes *forward* in time. This ensures that frames that have already been processed are not affected by subsequent strokes (similar to other video processing techniques [Bai et al. 2009]).

We use CSH because its data structure for an n-frame video sequence can be constructed in $\mathcal{O}(n)$ time and matches retrieved in $\mathcal{O}(n)$ time. We construct the feature vector for the matching by sampling 11×11 patches of RGB values and projecting them into a 16-dimensional space that is derived from a Principal Component Analysis (PCA) of these patches. To make the PCA tractable, we compute it from a sub-sampling of all the patches in the video sequence.

Our user annotation interface consists of a brushing tool with a user-specified radius. When the user paints a constraint, we use the patch directly under the stroke as a query, and extend the constraint to k matching pixels that lie within the radius. This radius allows users to control how far strokes can be propagated, thereby adapting to the specificities of the scene. To propagate the constraints temporally, we advect the strokes to subsequent frames using the optical flow, and discard the constraint once it is advected outside of the frame. In practice, to keep computation tractable, a maximum of $k = 3000$ neighbors per frame are retrieved.

4 Results and Discussion

In this section, we evaluate our algorithm on both single images and video sequences and compare with state-of-the-art techniques. The quality of our results is better evaluated on the accompanying video.

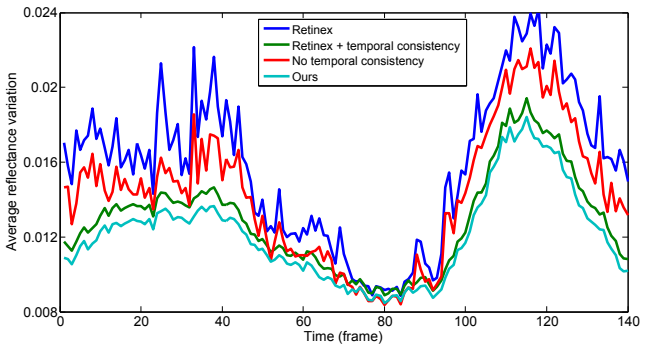


Figure 6: *The average derivative of reflectance for pixels that should have constant reflectance, showing the general stability of our method against Retinex applied per frame. Removing our temporal consistency also increases the average derivative.*

In our prototype implementation, we let the user interactively adjust the parameters p, λ_r, λ_s and T^{chr} to obtain the desired reflectance and illumination gradients (Eq. 6). These gradients are then used in conjunction with the non-local terms and temporal regularization, weighted by λ_t and λ_{nl}. Typical ranges for these parameters are $p \in [0.4, 0.5]$, $\lambda_r \in [0, 0.5]$, $\lambda_s \in [0, 10]$, $\lambda_{nl} \in [0, 1]$, $T^{\text{chr}} \in [0, 0.2]$. The constant reflectance and illumination strokes are used to directly edit the reflectance and illumination gradients. The final illumination layer is reconstructed by solving a linear system using a fast parallelized CPU-based multiscale solver with Jacobi iterations.

To propagate the user strokes, we implemented the search using an optimized version of the LSHKIT library [Dong 2014]. Both the temporal advection of the user strokes and the temporal smoothing term are driven by optical flow based on Liu et al. [2009]. We precompute this optical flow as a preprocessing step which takes 0.25-0.50s to process a 0.5MP video frame. This allows us to give the user interactive feedback while they annotate the video with the constraints. Finally, we propagate the user constraints forward by 12 frames using background threads.

In Figure 4, we evaluate the different terms in our decomposition algorithm. We observe that allowing deviations from the image formation model (Eq. 3) reduces artifacts at object boundaries, non-local terms remove low-frequency errors, temporal constraints lead to time-coherent results, and user input can resolve other ambiguities. Importantly, every video is different and might need slight variations of these terms. By making the solver interactive, we allow users to quickly adapt the result to their requirements.

Static images We compare the results of our technique to state-of-the-art techniques for static images. In Fig. 5, we compare to the work of Shen et al. [2011] and Zhao et al. [2012]. In the supplemental material, we show our decomposition on benchmarking examples from Grosse et al. [2009], as well as a comparison with the method of Bousseau et al. [2009]. In all these cases, a few strokes are sufficient to create results that are comparable to other techniques, but at interactive speeds.

Videos First, we evaluate our technique on a realistically-rendered animation of the San Miguel 3D scene (Figure 7), 100 frames at 1280×960, using Metropolis light transport with the PBRT rendering engine [Pharr and Humphreys 2010]. This sequence features detailed geometry, spatially-varying reflectances, complex outdoor illumination, and an intricate camera path; as such, it is a good approximation of a real-world example, with the advantage of pro-

(a) Input video frame (b) Ground truth (c) [Zhao et al. 2012] (no user strokes) (d) Our result (no user strokes)

Figure 7: *We demonstrate our technique on one frame of a rendered video sequence (a) with ground truth reflectance and illumination (b). Using the single image method of Zhao et al. [2012] on a per-frame basis produces a result that is spatially inaccurate and temporally inconsistent (c). Their non-local constraints have considered the albedo of the tree, ground and tables to be the same. Even though it does not account for the chromatic illumination, our algorithm is able to produce a good result on this challenging example even without any user interaction (d). Please refer to the supplementary video to appreciate the temporal behavior of the results.*

viding a ground-truth decomposition. As shown in Figure 7), our technique produces a result that compares well with the ground truth and is significantly better than the state-of-the-art single image technique of Zhao et al. [2012].

Additionally, we compare our method with the concurrent work of Ye et al. [2014] on a real-world video sequence in Figure 8 and an animated video in Figure 9. Like us, they incorporate user input to disambiguate challenging areas, but unlike us, this user input is specified on the first frame and the solution is propagated to the rest of the video. Also, their method runs in the order of hours, precluding any interactive refinement. On the other hand, the user can interactively refine the result using our technique. In addition, users can quickly specify annotations in different frames and this is useful for videos with significant camera and scene motion.

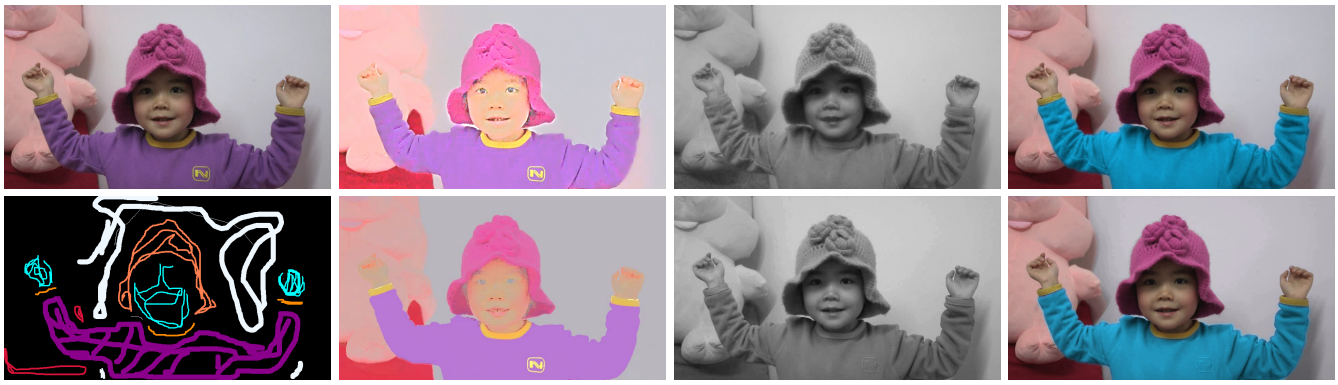
Stability Methods tailored for images are not designed for videos and, when applied to each frame independently, often produce temporal inconsistencies. Our technique produces temporally consistent results that capture the reflectance-illumination separation well. To demonstrate the stability of our method, we track 100k pixels which should have constant reflectance and no motion across a video. Figure 6 shows the average derivative of these reflectance values. If our temporal consistency is good, then this should be close to zero. This demonstrates the advantage of our approach over the frame-by-frame Retinex algorithm, and the advantage of our temporal consistency term. Our approach is also more accurate than Retinex (Fig. 4).

Discussion While our CPU-based solver is quite fast, we believe that the performance could be further improved by implementing our algorithm on graphics hardware. For instance, Jacobi solvers are well-suited to parallelization on the GPU, and our spatially coherent and sparse non-local constraints will not have a significant impact on porting the algorithm. Due to the nearest-neighbor search, our algorithm could slow down if too many strokes are specified. However, when we produced our results, we never reached the point where this would become an issue. In practice, the examples we show were edited with less than 20 minutes of user interaction and sometimes, were produced with no interaction at all. Table 1 contains detailed statistics about our input videos, and computation and interaction times that were required to produce our results.

Our tool assumes monochromatic lighting. This speeds up the algorithm, and we found that it leads to accurate decompositions for most videos where this assumption is close. However, in presence of illumination with spatially varying colors, or strong colored inter-reflections, this can lead to artifacts (e.g., see the shirt in Fig.10, and the pink inter-reflections on the girl's arms in Fig. 11).

Our $\ell_2 - \ell_p$ model assumes that illumination is smoothly varying; this assumption can be violated due to high-frequency lighting effects caused by hard cast shadows or high frequency geometry, and our technique produces an overly smooth shading layer in these cases. This can be observed in Fig. 8 (b) where high-frequency residual shadows can be seen in the reflectance, or in Fig. 5 where our illumination is smoother than other methods. While our non-local constraints help mitigate this issue to an extent (by forcing high-frequency achromatic variations out of the reflectance and into the illumination), more user interaction is often necessary in these cases. Generally, the decomposition obtained using Ye et al.'s approach [2014] with user interaction is more accurate when compared to our method without any user interaction. However, this comes at the expense of increased computation, which drastically slows down iteration time for correcting artifacts versus our interactive method.

In our early experiments, we tested temporal propagation both forward and backward in time. While we did not observe significant gain in accuracy, more user interaction was required because one needed to step forward and backward to check the results. In comparison, propagating only forward in time yields a simpler workflow in which one only needs to process the frames in chronological order.



(a) Input video / Ye et al. scribbles (b) Reflectance (c) Shading (d) Recoloring

Figure 8: *Comparison of our approach* (top) *to Ye et al. [2014]* (bottom). *Both approaches require user input, with Ye et al.'s all coming on the first frame (a). However, any mistakes cannot be quickly corrected, as their method requires four hours to compute the result. In comparison, our approach takes two minutes to compute and gives immediate feedback to user scribbles. The benefits of this can be seen in the yellow logo on the girl's shirt. Due to the slow feedback, it appears incorrectly in the shading layer of Ye et al., but it does not appear in our interactively refined shading decomposition (c). As a result, our recolored video does not have this artifact (d).*

(a) Input video frame (b) Ye et al. [2014] (with user strokes) (c) Our result (without user strokes)

Figure 9: *On this animated video sequence (a), our technique is able to, without any user strokes, produce a comparable result (c) to the user-assisted result of Ye et al. [2014] (b).*

5 Applications

We demonstrate our method on various applications benefiting from an editable and temporally consistent intrinsic decomposition. Our main use of the intrinsic decomposition is to independently alter the illumination layer and the reflectance layer.

Reflectance editing Editing the reflectance of an object in a video is easy when the color is uniform; in such cases, a simple chrominance change suffices. However, this becomes laborious when the reflectance has high-frequencies that also appear in the luminance. One cannot simply paint over it since it would alter the illumination. However, with our decomposition, painting in the reflectance layer performs the desired operation since illumination remains unchanged. We demonstrate this in Figures 1 and 11 (top); in both these cases the reflectance has high-frequency variations in both luminance and chrominance. By using our decomposition, we are able to paint over the original reflectance, while still preserving the spatially-varying illumination in the presence of complex motion. In all these examples, we paint the reflectance in a single frame, and use optical-flow tracking to advect the paint strokes in time.

Illumination editing Turning hard shadows into soft shadows is a challenging operation if one only has access to raw data since blurring the shadows ends up blurring the textures in the scene too. In comparison, with our decomposition, this task becomes straightforward since one needs to blur the illumination layer (Fig. 12).

Lighting-consistent video compositing Naively compositing videos that contain shadows produces unsightly results in which the shadows overlap instead of merging. Our decomposition enables the proper merging. Denoting S_1 and S_2 the shading layers of the two videos to be composited, we compute $\min(S_1, S_2)$ as the new shading layer within a segmentation of the foreground video obtained with Video Snapcut for instance [Bai et al. 2009] (Fig. 13).

(a) Input frame (b) Albedo (c) Shading

Figure 10: *Our method assumes monochromatic lighting. The presence of chromatic lighting or strong interreflections that produce colored shading violate our model, and can produce significant artifacts that cannot be corrected, even with heavy user interactions.*

6 Conclusions

In this work, we have presented the first interactive intrinsic decomposition algorithm for video sequences. Our technique is built on top of a hybrid $\ell_2 - \ell_p$ algorithm that efficiently computes reflectance and shading gradients from a image. We present a fast solver that combines these gradients with spatial and temporal constraints to reconstruct temporally consistent shading and reflectance videos at interactive rates. The interactive nature of our solver is a significant contribution over previous work, making it possible for users to navigate the parameter space of the algorithm and make annotations while receiving immediate feedback. Our technique can be used to efficiently derive high-quality intrinsic decompositions for a wide variety of real-world video sequences. Most importantly, our decomposition enables a number of video editing tools that are oth-

Sequence	Figure	# Frames	Resolution	Solver (per frame)	# Strokes	Interaction
House	1	91	1024×576	0.60 s	325	20 min
Kermit	4 (f)	291	320 ×568	0.18 s	101	6 min
San Miguel	7 (d)	100	1280×960	1.30 s	32	15 min
Kid	8 (top)	285	960 ×540	0.51 s	10	6 min
Chicken	9 (c)	148	640 ×360	0.24 s	0	0 min
Girl	11	98	640 ×360	0.24 s	276	15 min
SIGGRAPH Cart	Video (3'24)	141	1024×576	0.59 s	0	0 min
Shadow art	12	134	1024×576	0.62 s	28	10 min
Compositing (background)	13 (a)	215	1024×576	0.60 s	330	20 min
Compositing (foreground)	13 (b)	107	1024×576	0.62 s	207	20 min

Table 1: *Statistics for our input videos. We report approximate interaction times and the number of strokes for each complete video sequence. Note that we focused on high quality decompositions and did not try to minimize the amount of time spent or the number of user strokes.*

(a) Input video sequence (b) Intrinsic decomposition (c) Reflectance-edited video (d) Naive chrominance editing

Figure 11: *Our decomposition enables easy reflectance editing such as re-coloring the girl's t-shirt. By separating the two components, we are able to make these edits while retaining the original illumination in the video. A naive editing of the chrominance leaves luminance variations due to the illustration on the shirt.*

erwise difficult to implement; these include recoloring, retexturing, illumination editing, and lighting-consistent video compositing.

Acknowledgements

We thank the SIGGRAPH reviewers for their feedback. This work was partially supported by NSF grants CGV-1111415, IIS-1110955, OIA-1125087, and LIMA - Région Rhône-Alpes. We thank the authors of the video footage: McEnearney (Fig. 1), R. Cadieux (Figs. 3, 5, and 11), G. M. Lea Llaguno (Fig. 7), Ye et al. [2014] (Fig. 8), M. Assegaf (Fig. 9), B. Yoon (Fig. 12), The Scene Lab via Dissolve Inc. (Fig. 13 (a)).

References

BAI, X., WANG, J., SIMONS, D., AND SAPIRO, G. 2009. Video snapcut: Robust video object cutout using localized classifiers. *ACM Trans. on Graph. (SIGGRAPH) 28*, 3.

BARRON, J. T., AND MALIK, J. 2012. Color constancy, intrinsic images, and shape estimation. *ECCV*.

BARRON, J. T., AND MALIK, J. 2013. Intrinsic scene properties from a single RGB-D image. In *CVPR*.

BARROW, H., AND TENENBAUM, J. 1978. Recovering intrinsic scene characteristics from images. *Computer Vision Systems*.

BELL, S., BALA, K., AND SNAVELY, N. 2014. Intrinsic images in the wild. *ACM Trans. Graph. (SIGGRAPH) 33*, 4.

BJÖRCK, A. 1996. *Numerical Methods for Least Squares Problems*. SIAM.

BONNEEL, N., SUNKAVALLI, K., PARIS, S., AND PFISTER, H. 2013. Example-based video color grading. *ACM Trans. Graph. (SIGGRAPH) 32*, 4.

BOUSSEAU, A., PARIS, S., AND DURAND, F. 2009. User-assisted intrinsic images. *ACM Trans. Graph. (SIGGRAPH) 28*, 5.

CARROLL, R., RAMAMOORTHI, R., AND AGRAWALA, M. 2011. Illumination decomposition for material recoloring with consistent interreflections. *ACM Trans. Graph. (SIGGRAPH) 30*, 4.

CHEN, Q., AND KOLTUN, V. 2013. A simple model for intrinsic image decomposition with depth cues. In *ICCV*.

DONG, Y., TONG, X., PELLACINI, F., AND GUO, B. 2011. Appgen: Interactive material modeling from a single image. *ACM Trans. Graph. (SIGGRAPH Asia) 30*, 6.

DONG, W., 2014. LSHKIT: A C++ locality sensitive hashing library.

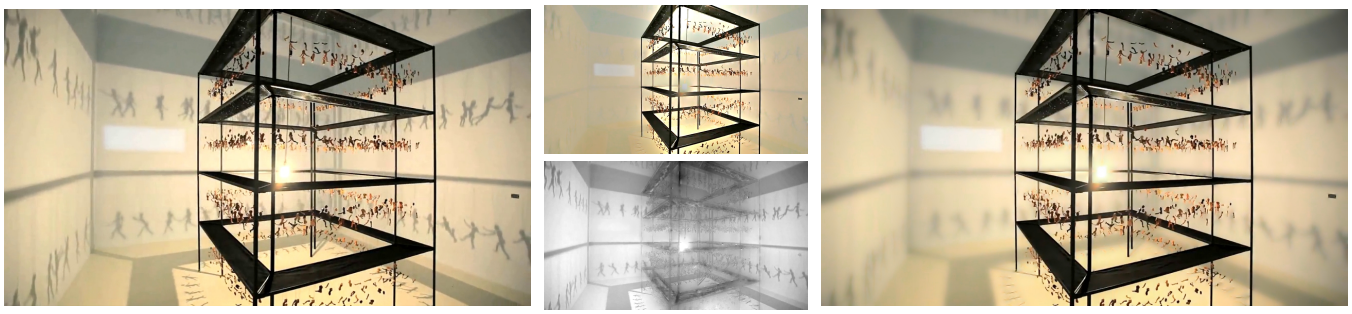
FARBMAN, Z., AND LISCHINSKI, D. 2011. Tonal stabilization of video. *ACM Trans. Graph. (SIGGRAPH) 30*, 4.

GARCES, E., MUNOZ, A., LOPEZ-MORENO, J., AND GUTIERREZ, D. 2012. Intrinsic images by clustering. *Computer Graphics Forum (EGSR) 31*, 4.

GEHLER, P., ROTHER, C., KIEFEL, M., ZHANG, L., AND SCHLKOPF, B. 2011. Recovering intrinsic images with a global sparsity prior on reflectance. In *NIPS*.

GROSSE, R., JOHNSON, M. K., ADELSON, E. H., AND FREEMAN, W. T. 2009. Ground truth dataset and baseline evaluations for intrinsic image algorithms. In *ICCV*, 2335–2342.

HSU, E., MERTENS, T., PARIS, S., AVIDAN, S., AND DURAND, F. 2008. Light mixture estimation for spatially varying white balance. *ACM Trans. Graph. (SIGGRAPH)*.



(a) Input video sequence (b) Intrinsic decomposition (c) Shadow-edited video

Figure 12: *Intrinsic decompositions can be used to edit lighting in videos. In this example, we decompose the video into reflectance and illumination components, blur the illumination component, and recombine the original reflectance and the blurred illumination to produce a video sequence with soft shadows. This form of lighting manipulation cannot be done without our decomposition; directly filtering the input video will blur even reflectance and object edges.*

(a) Background video frame (b) Foreground video frame (c) Naive compositing (d) Our compositing

Figure 13: *Our technique can be used for realistic video compositing. Here, we take two video sequences shot with different viewpoints (a, b) and decompose them into their respective reflectance and shading components. Compositing the two components separately and combining them allows us to create a video composite with realistic shadows and lighting (d). A naive compositing produces inconsistent shadows (c).*

KORMAN, S., AND AVIDAN, S. 2011. Coherency sensitive hashing. In *ECCV*.

KRISHNAN, D., AND FERGUS., R. 2009. Fast image deconvolution using hyper-laplacian priors. In *NIPS*.

LAFFONT, P.-Y., BOUSSEAU, A., PARIS, S., DURAND, F., AND DRETTAKIS, G. 2012. Coherent intrinsic images from photo collections. *ACM Trans. Graph. (SIGGRAPH Asia) 31*, 6.

LAFFONT, P.-Y., BOUSSEAU, A., AND DRETTAKIS, G. 2013. Rich intrinsic image decomposition of outdoor scenes from multiple views. *IEEE Trans. Vis. Comput. Graph. 19*, 2, 210 – 224.

LAND, E. H., JOHN, AND MCCANN, J. 1971. Lightness and retinex theory. *Journal of the Optical Society of America 61*, 1–11.

LEE, K. J., ZHAO, Q., TONG, X., GONG, M., IZADI, S., LEE, S. U., TAN, P., AND LIN, S. 2012. Estimation of intrinsic image sequences from image+depth video. In *ECCV*.

LI, Y., AND BROWN, M. S. 2014. Single image layer separation using relative smoothness. In *CVPR*.

LIU, C. 2009. *Beyond pixels: exploring new representations and applications for motion analysis.* PhD thesis, MIT.

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

OMER, I., AND WERMAN, M. 2004. Color lines: Image specific color representation. In *CVPR*.

PARIS, S. 2008. Edge-preserving smoothing and mean-shift segmentation of video streams. In *ECCV*.

PHARR, M., AND HUMPHREYS, G. 2010. *Physically Based Rendering, Second Edition: From Theory To Implementation.*

RAGAN-KELLEY, J., ADAMS, A., PARIS, S., LEVOY, M., AMARASINGHE, S. P., AND DURAND, F. 2012. Decoupling algorithms from schedules for easy optimization of image processing pipelines. *ACM Trans. Graph. (SIGGRAPH) 31*, 4.

SHEN, L., AND YEO, C. 2011. Intrinsic images decomposition using a local and global sparse representation of reflectance. In *CVPR*.

SHEN, L., TAN, P., AND LIN, S. 2008. Intrinsic image decomposition with non-local texture cues. In *CVPR*.

SHEN, J., YANG, X., JIA, Y., AND LI, X. 2011. Intrinsic images using optimization. In *CVPR*.

TAPPEN, M., FREEMAN, W., AND ADELSON, E. 2005. Recovering intrinsic images from a single image. *PAMI 27*, 9, 1459–1472.

WEISS, Y. 2001. Deriving intrinsic images from image sequences. In *ICCV*.

YE, G., GARCES, E., LIU, Y., DAI, Q., AND GUTIERREZ, D. 2014. Intrinsic Video and Applications. *ACM Trans. Graph. (SIGGRAPH) 33*, 4.

ZHAO, Q., TAN, P., DAI, Q., SHEN, L., WU, E., AND LIN, S. 2012. A closed-form solution to retinex with nonlocal texture constraints. *PAMI 34*, 7, 1437–1444.