Reflectance and Illumination Video Editing using Fast User-Guided Intrinsic Decomposition

Nicolas Bonneel[*,+] Deqing Sun[*] Kalyan Sunkavalli[†] Sylvain Paris[†] Hanspeter Pfister[*]
[*]Harvard University [+]LIRIS/CNRS [†] Adobe

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

Figure 1: *An input video sequence **(a)** is decomposed into a component representing textures **(b) (top)** and a component for the incoming illumination **(b) (bottom)** in an interactive fashion with real-time feedback, and in a temporally consistent way. This allows, for example, for the editing of textures with consistent shading **(c)**. Note the changes to the brick pattern, the roof, and the pathway leading up to the building. Please refer to the accompanying supplementary video to get a sense for the temporal consistency of our result.*

Abstract

Object illumination and color are critical characteristics of a scene and being able to edit them allows artists to achieve powerful effects. Intrinsic image decomposition is the ideal component for this kind of tasks. By separating the illumination from the scene reflectance, it enables key operations such as recoloring and relighting. Significant progress has been done recently for decomposing static images. However, these algorithms rely on sophisticated optimization schemes that are computationally expensive and orders of magnitude too slow to be applied to video sequences. So much that even an optimized implementation would remain unpractical. In this paper, we introduce a user-guided algorithm that runs fast enough to be used in an interactive setting. Our strategy is to rely on an efficient sparse formulation – we also exploit the same kind of information as successful static methods but use it in ways that only have a minor impact on running time. The core of our approach is a gradient-domain ℓ_2-ℓ_p energy that models a sparse prior on reflectance gradients and a smooth prior on illumination. We show that the produced set of nonlinear equations can be solved very efficiently using look-up tables. Then, we provide scribbles to users to refine the decomposition. Our scribbles introduce local constraints in our optimization that add only a minimal overhead. Further, we extend these constraints to other similar image regions, thereby effectively enabling users to affect large regions with minimal effort. We also leverage multi-threading to precompute solutions a few frames ahead of the current one at a minimal cost. Coupled with the ability of our solver to use an initial guess to speed up convergence, this effectively shortens the computation time and offer a fast feedback to users. We demonstrate our approach on real sequences and show that we can obtain satisfying results with a reasonable amount of user interaction. We illustrate the benefits of our decomposition on video recoloring and shadow compositing.

1 Introduction

Manipulating illumination in a scene is the key element of a wide range of applications, such as relighting, white balancing, or texture editing, e.g., [Bousseau et al. 2009; Laffont et al. 2012; Barron and Malik 2013]. A versatile way to do is to separate the image data into a product of illumination and reflectance layers. This problem is often referred to as the *intrinsic decomposition* of an image, and allows to manipulate independently these layers without the need of a global understanding of the three-dimensional geometry of the scene or the light transport within it. This decomposition is particularly challenging because the effects of illumination and reflectance are conflated into a single observation, thereby making the problem severely ill-posed. Nonetheless, significant progress has been made recently and usable decompositions of real-world scenes can be obtained. The main elements to this recent advance are additional data such as depth maps [Lee et al. 2012; Barron and Malik 2013; Chen and Koltun 2013], several views of the same static scene [Laffont et al. 2012], or user guidance [Bousseau et al. 2009; Shen et al. 2011], as well as sophisticated priors such as reflectance sparsity [Omer and Werman 2004; Hsu et al. 2008] and non-local cues [Shen et al. 2008; Garces et al. 2012; Laffont et al. 2012; Zhao et al. 2012; Chen and Koltun 2013].

Bringing lighting editing tools into the realm of video processing is far from straightforward. The recent approaches for estimating intrinsic decompositions are successful on static images but their computation cost is prohibitively high if one is interested in videos. The complex solvers involved in these decompositions require minutes of computation per image, which is several orders of magnitude slower than video frame rates that are on the order of 30Hz, i.e., one frame every 33ms. This issue is even more acute since depth data and other viewpoints are not available for videos in general, which leaves only user guidance among the key elements recently proposed to achieve accurate decompositions of real-world scenes. However, user interaction requires interactive feedback and existing techniques are too slow. We argue that the needed acceleration is so large that even an optimized implementation on graphics hardware would not be sufficient. This is why we introduce a new algorithm dedicated to fast processing of videos. The centerpiece of our approach is a hybrid ℓ_2-ℓ_p solver that models priors on reflectance sparsity and illumination smoothness using only small image stencils. A small image footprint generates more efficient memory accesses and lends

itself better to optimization thanks to limited dependencies between pixels [Ragan-Kelley et al. 2012]. Further, we leverage parallel processing to precompute a few frames ahead of time, thereby enabling smooth forward navigation. Moreover, our solver can exploit this information as an initial guess to speed up convergence and react faster to user edits compared to a naive initialization of the solver. We also provide a set of scribbles to the users so that they can control the produced decompositions with indications such as achromatic regions and smooth illumination areas. To maximize the effectiveness of this user guidance, we infer additional constraints by searching for pixels with a similar appearance to those selected by users and treating them similarly. Put together, our approach provides interactive feedback and enables users to produce usable intrinsic decompositions of video sequences with a reasonable amount of interaction.

We first demonstrate our approach on a real-world footage with applications such as recoloring / retexturing, lighting editing, and shadow composting.

Contributions This paper makes the following contributions:

– We enable complex video editing applications at interactive rate, including recoloring, lighting editing, and shadow compositing.

– We introduce a fast algorithm to estimate intrinsic decompositions of videos. Our approach is based on a hybrid ℓ_2-ℓ_p optimization that can be solved very efficiently using precomputed look-up tables.

– We describe a set of scribbles that let users control the results and a technique to propagate this input in space and time to limit the amount of interaction that users need to do.

1.1 Related Work

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. It is also related to the intrinsic decomposition developed by Matsushita et al. [2004] for video-surveillance footage but we do not assume a fixed viewpoint. 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. While these applications are successful, they aim for a specific effect. In comparison, we seek a more versatile intrinsic decomposition that enables several applications such as recoloring and lighting editing.

Generic Video and Image Editing Our approach also shares technical similarities with methods that target other applications. For instance, we follow the same temporal metaphor as the filtering technique of Paris [2008] and the selection tool of Bai et al. [2009] and propagate information only forward in time. This allows users to edit the video in chronological order and be sure that the results at previous frames are not altered by subsequent edits. When we propagate information in time, we take the optical flow into account akin to Bai et al. [2009] and Lang et al. [2012] We also extend the range of our scribbles similarly to Boyadzhiev et al. [2012] in the context of white balancing static images. While we had to adapt most of these technical points to our specific scenario, we do not claim this as a major contribution of our 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 decomposition typically attempt to separate image gradients into reflectance and shading 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 are local in the sense that they make decisions about the reflectance and the shading at the pixel, from the observed intensity at (or in the neighborhood of) that pixel. 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 non-local priors on the reflectance [Shen et al. 2008; Gehler et al. 2011; Shen et al. 2011; Zhao et al. 2012], 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; Laffont et al. 2012] or by making use of depth information captured with RGBD cameras [Lee et al. 2012; Barron and Malik 2013; Chen and Koltun 2013]. While all these techniques produce higher-quality results, they do so at a very 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. However, because their underlying solver uses medium-size image stencils, it takes more than 10s to decompose a half-megapixel image which is too slow for interactive video editing and does not address the temporal consistency issue. In contrast, our technique can process images of the same resolution in less than half a second. Nonetheless, the strategy of relying on user guidance is effective and we follow the same approach in our work. Our work combines a local gradient-based intrinsic decomposition with refinements based on user-defined scribbles. We show that our approach can be seen of an extension of the Retinex algorithm and that it allows us to solve for the reflectance and shading at interactive rates, while allowing the user to control the quality of the decomposition.

2 Efficient Intrinsic Decomposition

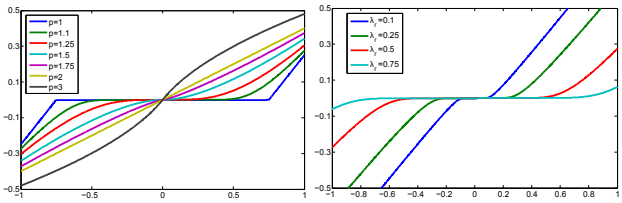
In this section, we describe our algorithm to decompose efficiently an input video I into a illumination layer S and a reflectance layer R. We assume a simple image formation model:

$$I = S \times R \tag{1}$$

where the multiplication is done per channel in the RGB color space. The core of our algorithm is a hybrid ℓ_2-ℓ_p energy formulation in the gradient domain that represents a sparsity prior on reflectance values and a smoothness prior on illumination. We achieve temporal consistency using a causal smoothness prior along the time dimension. We enable user control via a set of scribbles that add constraints into our energy. We automatically extend the scope of these constraints to reduce the amount of user interaction required. The rest of this section presents the details of each step.

2.1 Hybrid ℓ_2-ℓ_p Gradient Separation

Our decomposition processes each RGB channel independently so for now on, we will consider only one channel. First, we work in the log domain to transform the image formation into a sum: $\log I = \log S + \log R$. 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



(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. 10); $\tilde{r}_k$ 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.*

and the image formation model, we can write: $i = s + r$ and express this constraint as a least-squares energy term: $\|i - s - r\|^2$.

This term alone is highly ambiguous since only i is known, and both s and r are unknown. To address this issue, we add priors on s and r and provide scribbles to users to add constraints to the system. We first describe the priors. We assume that reflectance values are sparse [Omer and Werman 2004; Hsu et al. 2008], i.e., that scenes are mostly made by objects of constant colors separated by hard boundaries. This is typically modeled using a ℓ_p term on the gradients with $p < 2$, that is, with our notation: $\|r\|^p$. We also assume that illumination exhibits smoother variations due to shading 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.: $\|s\|^2$. We put all the terms together to form the energy E. We use a continuous notation so that we can apply variational tools later; we will discretize at the end to act upon pixels. With x and y being the image coordinates, this gives us:

$$E(s,r) = \iint \|i - s - r\|^2 + \lambda_s \|s\|^2 + \lambda_r \|r\|^p \, \mathrm{d}x \, \mathrm{d}y \qquad (2)$$

where λ_s and λ_r control the influence of the priors on s and r. First, we assume constant values for these two parameters. Then we show that it is beneficial to use spatially varying weights and explain how to extend our solver to still be efficient in this case.

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. We will show that we minimize Equation 2 using a simple look-up table.

First, we remark that the x and y components of the energy are independent and separable. In what follows, we concentrate of the x component, its y counterpart being treated similarly. Then, we express s as a function of r, that is, we assume r known and equal to $\bar{r}$, and derive a closed-form solution for s. Since s is a function, we apply a variational approach that is standard in such case. We add a perturbation η of magnitude ϵ applied to s and consider the energy $E(s + \epsilon\eta, \bar{r})$. For the optimal s, the derivative with respect to ϵ should be 0 at $\epsilon = 0$. We start with the derivative:

$$\frac{\partial}{\partial \epsilon} \int \|i - s - \epsilon\eta - \bar{r}\|^2 + \lambda_s \|s + \epsilon\eta\|^2 + \lambda_r \|\bar{r}\|^p \, \mathrm{d}x \qquad (3)$$

then switch the integral and the derivative, and cancel the sparse term that does not depend on ϵ:

$$\int \frac{\partial}{\partial \epsilon} \left(\|i - s - \epsilon\eta - \bar{r}\|^2 + \lambda_s \|s + \epsilon\eta\|^2 \right) \mathrm{d}x \qquad (4a)$$

$$= \int 2\eta \left(s + \epsilon\eta + \bar{r} - i\right) + 2\lambda_s \eta \left(s + \epsilon\eta\right) \mathrm{d}x \qquad (4b)$$

When $\epsilon = 0$, this quantity is equal to 0. By removing the $\epsilon\eta$ terms, the factor 2, and regrouping the s variables, we get:

$$\int \eta \left((1 + \lambda_s)s + \bar{r} - i\right) \mathrm{d}x = 0 \qquad (5)$$

Since this holds for all η functions, we have $(1 + \lambda_s)s + \bar{r} - i = 0$, which gives us the expression that we sought:

$$s = \frac{i - \bar{r}}{1 + \lambda_s} \qquad (6)$$

The next step is to solve for r. We first rewrite Equation 2 with the new expression of s (Eq. 6):

$$\int \|i - \frac{i - r}{1 + \lambda_s} - r\|^2 + \lambda_s \|\frac{i - r}{1 + \lambda_s}\|^2 + \lambda_r \|r\|^p \, \mathrm{d}x \qquad (7)$$

Factoring $(i - r)$ in the first term, taking the weights out of the first and second terms, and regrouping the $\|i - r\|^2$ terms, we get:

$$\int \frac{\lambda_s}{1 + \lambda_s} \|i - r\|^2 + \lambda_r \|r\|^p \, \mathrm{d}x \qquad (8)$$

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

$$\int \frac{\lambda_s}{1 + \lambda_s} \|i - \tilde{r}_{k+1}\|^2 + \lambda_r w_{k+1} \|\tilde{r}_{k+1}\|^2 \, \mathrm{d}x \qquad (9a)$$

$$\text{with } w_{k+1} = \frac{p}{2} |\tilde{r}_k|^{p-2} \qquad (9b)$$

Using the same variational reasoning as before, we get a closed-form expression of $\tilde{r}_{k+1}$ as a function of $\tilde{r}_k$:

$$\tilde{r}_{k+1} = \frac{2\lambda_s i}{2\lambda_s + \lambda_r (1 + \lambda_s) p |\tilde{r}_k|^{p-2}} \qquad (10)$$

Iterating this formula converges to a minimizer of mixed-norm energy (Eq. 8) 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 K times Equation 10. 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 function $\mathrm{lut}_{\lambda_s,\lambda_r}(\tilde{r}_k)$ 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 as we do next.

Taking Chrominance Variations into Account The energy function in Equation 2 encourages sparse but large reflectance gradients and dense but smaller shading 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 due to changes in reflectance [Grosse et al. 2009]. We leverage this by spatially modulating λ_s in Equation 2 as:

$$\lambda_s = \lambda_s ||i - \sum_k i_k / 3||, \qquad (11)$$

(a) Input video frame (b) $\lambda_s = 10, \lambda_r = 0.5, p = 1.05$ (c) $\lambda_s = 2, \lambda_r = 0.5, p = 1.05$ (d) $\lambda_s = 10, \lambda_r = 1.5, p = 1.75$

Figure 3: *This figure demonstrates the effect of the different parameters in Equation 10 on the resulting reflectance (top) and shading (bottom). A high weight on the shading gradient term leads to a blurry shading, but the reflectance is not sparse (b). Tilting the weights in favor of the reflectance term makes the reflectance gradients sparse leading to a result that captures the piece-wise constant nature of the reflectance more accurately (c). Finally, changing the norm of the reflectance term from a sparse $p = 1.05$ to a dense $p = 2$ makes both the reflectance and the shading look simply like contrast-enhanced (or diminished) versions of the image (d).*

where k denotes the three color channels. $\lambda_s(x, y)$ is thresholded to avoid degeneracies. At pixels where the variation in the chrominance is small, the weight of the ℓ_2 error term in Equation 2 is reduced, encouraging the the image gradient to be attributed to the shading. A large variation in the chrominance has the opposite effect, and leads to the image gradient being assigned to the reflectance.

Discussion An important step of the above derivation is expressing s as a function of r (Eq. 6) using a variational study of the energy. Another option could have been to enforce strictly the image formation model and set $s = i - r$. In this case, the energy becomes $\iint \lambda_s \|i - r\|^2 + \lambda_r \|r\|^p \, \mathrm{d}x \, \mathrm{d}y$. However, while this formulation is simpler, this system is stiffer because it must exactly adhere to the image formation model and is known to produce lower-quality decompositions [Chen and Koltun 2013] . Enforcing this as a soft constraint allows our optimization to handle deviations resulting from more complex image formation. We carefully compared the data produced by two approach, the exact model and our softer approach, and found that the main improvement comes from Equation 6 that favors a smooth illumination where the observed chromaticity is smooth, even though it may violate the image formation model.

Figure 3 illustrates the form the function $\mathrm{lut}_{\lambda_s,\lambda_r}(\tilde{r}_k)$ takes for different values of λ_s, λ_r, and p. Note that for certain values of these parameters, this functions approximates the thresholding function that the Retinex algorithm uses to separate image gradients into reflectance and shading gradients. This suggests that our look-up table is in fact a generalization of the Retinex algorithm, albeit one that is derived from the hybrid ℓ_2-ℓ_p energy formulation described above. In addition, unlike the Retinex algorithm, our look-up table is a softer function and allows for non-zero reflectance and shading gradients at the same pixel.

2.2 Reconstructing the Layers

For a given values of λ_r, we precompute $\mathrm{lut}_{\lambda_r}(\lambda_s, i)$. Then, for each pixel, we estimate the x and y components of r by applying lut_{λ_r} twice, once for each axis. We use Equation 6 to recover s. Then, we solve the Poisson equation to recover $\log S$:

$$E_p = \iint \|\nabla \log S_t(x, y) - s_t(x, y)\|^2 \, \mathrm{d}x \, \mathrm{d}y, \qquad (12)$$

and exponentiate to get the reflectance layer S. We estimate the illumination layer $R = I/S$ using the image formation model (Eq. 1). This approach has two advantages. First, compared to reconstructing S and R separately, we solve the Poisson equation once instead of twice. Second, using the Poisson equation to retrieve S produces better results than using it for R. Solving the Poisson equation introduces low-frequency residuals when the gradient field is not integrable. Such residual is more likely to be innocuous in the illumination layer that contains already smooth shading variations in general, but would be more conspicuous in the reflectance layer that is expected to be piecewise constant, e.g., for man-made scenes comprising only of a small set of materials.

2.3 Temporal Consistency

Applying the previous technique frame by frame would generate unpleasant flickering. On the other end of the spectrum, we could perform the decomposition on the entire space-time volume at once and a temporal smoothness term in our formulation. However, such an approach would not be practical either. First, the running times and memory requirement would be prohibitively large because of the sheer amount of data to process. Second, user interaction would be challenging since adding a stroke in one frame could possibly affect the entire video, include frames already treated by the user. For our work, we opt for an intermediate approach, we propagate information only forward in time. This is a user-friendly option since treating frames in chronological order ensures that already previously edited frames are not affected by subsequent strokes [Bai et al. 2009]. Moreover, this can be implemented by considering only two frames at a time and the achieved temporal consistency is on par with a full space-time approach [Paris 2008]. Also, to further speed up the process, we enforce temporal consistency only

during the Poisson reconstruction stage and still perform the gradient decomposition (§ 2.1) frame by frame.

We name $\big(u_t(x,y), v_t(x,y)\big)$ the optical flow, i.e., the pixel at (x,y) at frame t moves to $\big(x+u_t(x,y), y+v_t(x,y)\big)$ at frame $t+1$. In the rest of the discussion, we omit the dependency on (x,y) for clarity's sake. To ensure the temporal coherence of the illumination logarithm $\log S$, we define a temporal smoothness term that moves the solution of the current frame to the previous frame advected by the optical flow:

$$E_{t+1} = \iint \|\log S_{t+1}(x,y) - \log S_t(x+u_t, y+v_t)\|^2 \,\mathrm{d}x\,\mathrm{d}y. \tag{13}$$

This term is then added to the standard reconstruction energy so that, when reconstructing the frame t, we exploit the information of the previous frame $t-1$:

$$E = E_p + \lambda_t E_t, \tag{14}$$

where λ_t is a parameter balancing the two objectives. The second term can also be seen as a form of data attachment that prevents the new frame from deviating too much from the previous frame. This type of optimization problems has been well studied in the literature; it leads to the Screened Poisson equation [Bhat et al. 2008] and we solve it efficiently using a multigrid solver with Successive Over-Relaxation (SOR) iterations [Nocedal and Wright 2006].

3 User-guided refinement

While our hybrid gradient separation algorithm produces reasonable results, we provide scribbles to users so that they can refine the results. Further, to minimize the need for user interaction, we extend the scribbles to pixels with a similar appearance.

User Strokes We provide users with scribbles that let users specific 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, and eschew non-local pairwise constraints [Shen et al. 2008; Zhao et al. 2012; Chen and Koltun 2013] that lead to dependencies across many pixels. This would degrade the performance on two fronts by augmenting the amount of required computation and generating complex memory access patterns challenging to optimize [Ragan-Kelley et al. 2012].

The first two strokes that we use are the *constant reflectance* and the *constant shading* strokes that specify that the gradient of the reflectance (and shading respectively) at those points is 0, i.e.: $r(x,y) = 0$ and $s(x,y) = 0$. These constraints are straightforward to apply to s and r, and we do not need to solve again the main optimization problem. We only need to solve the Poisson equation, which is very efficient.

The second set of strokes are the *gray-scale reflectance/shading* strokes that specify that the reflectance/shading at the stroke is the same for all three color channels. The Poisson system that we solve to reconstruct the reflectance and shading (Eq. 13) is accurate up to a scale factor in each color channel and the gray-scale strokes help us resolve this scale ambiguity. We implement this constraint as follows. We solve Equation 13 to recover the shading in the green color channel first. The gray-scale shading stroke then encourages the solution in the red and blue channels to match the green solution S^{g} by adding an additional term as follows:

$$E = E_p + \lambda_t E_t + \lambda_c \|\log S_t^{\{\mathrm{r|b}\}} - \log S_t^{\mathrm{g}}\|^2, \tag{15}$$

where the λ_c controls how strongly we enforce the grayscale constraint, and $S_t^{\{\mathrm{r|b}\}}$ denotes the red or blue illumination channel. Similarly, the gray-scale reflectance stroke adds the term $\|(\log I_t^{\{\mathrm{r|b}\}} - \log S_t^{\{\mathrm{r|b}\}}) - (\log I_t^{\mathrm{g}} - \log S_t^{\mathrm{g}})\|^2$ to the energy. This additional constraint essentially applies a Dirichlet boundary condition to the Poisson system and the new energy can be solved using a modified screened Poisson equation. The rationale for this approach is that it avoids a two-way coupling between the channels in which all the channels would depend on all the others akin to Boyadzhiev et al. [2012] for instance. The two-way coupling would make the solver significantly slower by requiring all three channels to be solved together in single step. In comparison, our approach adds only a minimal overhead. We picked the green channel because its spectrum lies in the center of the visible light spectrum and covers best the latter.

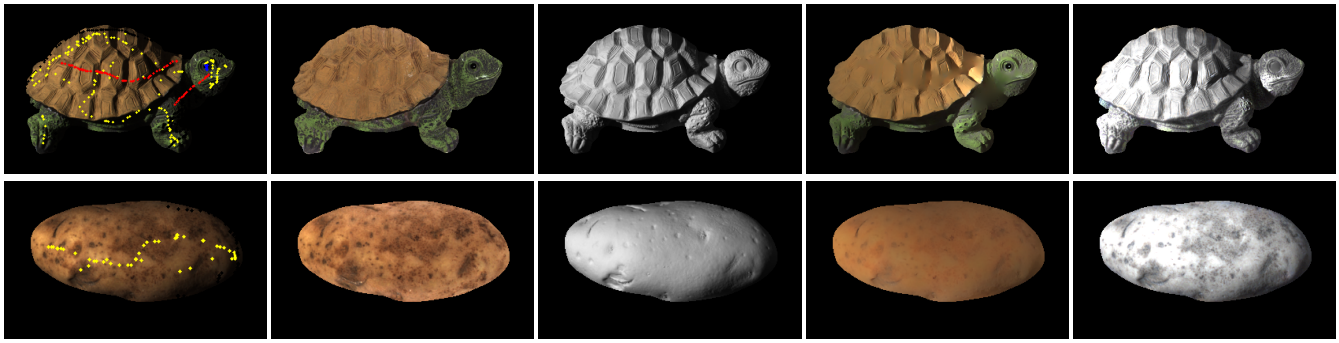
Propagating Strokes To speed up user interaction, we transfer user strokes *both spatially and temporally* to similar pixels that are found using coherency-sensitive hashing [Korman and Avidan 2011]. Unlike k-d tree-based nearest neighbor search data structures that are expensive to precompute (especially for video data), the coherence-sensitive hashing data structure for an n-frame video sequence is constructed in $\mathcal{O}(n)$ time and retrieves matches 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 is comprised 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 approach 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. In practice, to keep computation tractable, constraints are propagated to 8 frames and a maximum of $k = 3000$ neighbors per frame are retrieved.

4 Results and Discussion

In this section, we present our results on both static frames and video sequences and demonstrate a number of applications. Methods tailored for images are not designed for videos and, when applied to each frame independently, often produce temporal inconsistencies. Our technique, on the other hand produces temporally consistent results that capture the reflectance-shading separation well. The quality of our results is better evaluated on the accompanying video.

In our prototype implementation, we use a precomputed look-up table with parameters $p = 1.05$, $\lambda_r = 0.6$, and $\lambda_s = 1.0$ to estimate the reflectance and shading gradients (Eq. 10). The constant reflectance/shading strokes are used to directly edit these gradients. The computed shading gradients are then integrated using a fast and parallel CPU-based multigrid Poisson solver with SOR iterations. The gray-scale reflectance/shading strokes as well as the temporal smoothness constraints lead to boundary conditions that are incorporated into the Poisson solver as soft constraints with weights $\lambda_t = 0.2$ and $\lambda_c = 0.5$. To propagate the user strokes, we find similar pixels in space and time using Locality Sensitivity Hashing [Dong et al. 2008] that we implemented 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. Our prototype implementation solves



(a) Input image with user strokes | (b) Ground truth reflectance | (c) Ground truth shading | (d) Our reflectance | (e) Our shading

Figure 4: *The result of our method on images from the MIT intrinsic decomposition dataset [Grosse et al. 2009]. A few strokes are able to create a good intrinsic decomposition in real-time.*

(a) Input image with strokes | (b) Our reflectance | (c) Our shading

Figure 5: *Our technique demonstrated on an image from [Bousseau et al. 2009]. We are able to produce results that are qualitatively similar to their technique at speeds that are an order of magnitude faster.*

for 8 frames in parallel, and does the gradient reconstruction serially on a single thread. It takes 0.25-0.50s to process a 0.5 megapixel video frame. This allows us to give the user interactive feedback while they annotate the video with the constraints. Finally, we also propagate the user constraints forward by 8 frames using background threads.

Static images We compare the results of our technique to state-of-the-art techniques for static images. Figure 4 shows the results of our method on examples from the MIT database. A few strokes are sufficient to create results comparable to other techniques, but at interactive speeds. In Fig. 5, we decomposed an image presented in [Bousseau et al. 2009].

Videos We also evaluate our technique on challenging real-world video sequences as well as on a high-resolution realistic rendered animation of a 3-d scene. For the latter, we used the San Miguel scene (see Figure 6) to render 150 frames at 1280×960 resolution, 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 for of a real-world example, with the advantage of providing a ground-truth decomposition. This comparison is shown in Figure 6).

Discussion In our experiments, we found that our approach is fast enough for user interaction and accurate enough to be useful in practice. That said, we believe that performance could be further improved by implementing our algorithm on graphics hardware. And because of the nearest-neighbor search, our algorithm slows down if too many strokes are specified – that said, when we produced our results, we never reach the point where this would become an issue. Also, since we forwent nonlocal constraints to speed computation up, our decomposition is locally accurate but may exhibit inconsistencies on distant objects, e.g., if two similar objects appear on each side of the frame, their decompositions may not match. However, many editing operations as the ones shown in the rest of this section are also local and thus only requires local consistency. Our approach can handle colored lighting but sometimes underestimate the colorfulness of the lighting in extreme cases. This can be seen in the companion video when we seek to remove the colored light on the walking woman. While the light is significant less colored than the input, some colors remain visible. Finally, in our early experiments, we also tested temporal propagation both forward and backward in time. While we did not observe any significant accuracy gain, more user interaction was required because one needed to consistently move forward and backward in time to check the results. In comparison, propagating only forward in time yields a simpler editing workflow in which one only needs to process the frames in chronological order.

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 a painstaking 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 is Figures 1 and 7 (top); in both these cases the reflectance has a 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. The luminance-chrominance separation also fails when the color of

(a) Input video frame (with/without strokes) (b) Ground truth reflectance / shading (c) Our reflectance / shading without strokes (d) Our reflectance / shading with strokes

Figure 6: *Our technique demonstrated on one frame of a rendered video sequence (a). Our automatic hybrid ℓ_p-ℓ_2 decomposition technique is able to produce a good result on this challenging example (b). Adding a few user strokes and propagating them to similar pixels helps us improve this result further (c).*

the illumination varies spatially. This is demonstrated in Figure 7 where our decomposition allows us to easily edit the reflectance, while retaining the color casts in the original illumination. 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. 8).

Removing local color cast due to colored illumination is already a challenging task on static images that requires dedicated approaches, e.g., [Boyadzhiev et al. 2012]. Applying the same technique to a video is impractical due to processing time and the fact that temporal coherence is not taken into account. In comparison, our approach makes straightforward since it only requires converting the illumination layer into grayscale (Fig. 9).

Lighting-consistent video compositing Naively compositing videos that contain shadows produces unsightly results in with 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. 10).

5 Conclusion

We described a number of illumination and reflectance editing tools that rely on a new intrinsic decomposition method that is fast, user-assisted, temporally consistent, local in time, and handles color lighting. In contrast with other intrinsic decomposition techniques, our algorithm relies on a hybrid ℓ_p-ℓ_2 gradient separation formulation that is extremely fast to optimize. This makes it possible for the user to interact with the system to effectively and efficiently decompose video sequences into their reflectance and shading components. We show that this results in high-quality decompositions on a wide variety of images and video sequences. Most importantly, our decomposition enables a number of video editing tools that are otherwise very difficult to implement; these include recoloring, re-texturing, local illumination editing, and lighting-consistent video compositing.

References

BAI, X., WANG, J., SIMONS, D., AND SAPIRO, G. 2009. Video snapcut: Robust video object cutout using localized classifiers. *ACM Transactions on Graphics*.

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*, 3C26.

BHAT, P., CURLESS, B., COHEN, M., AND ZITNICK, L. 2008. Fourier analysis of the 2D screened poisson equation for gradient domain problems. In *ECCV*.

BJÖRCK, A. 1996. *Numerical Methods for Least Squares Problems*. Society for Industrial and Applied Mathematics.

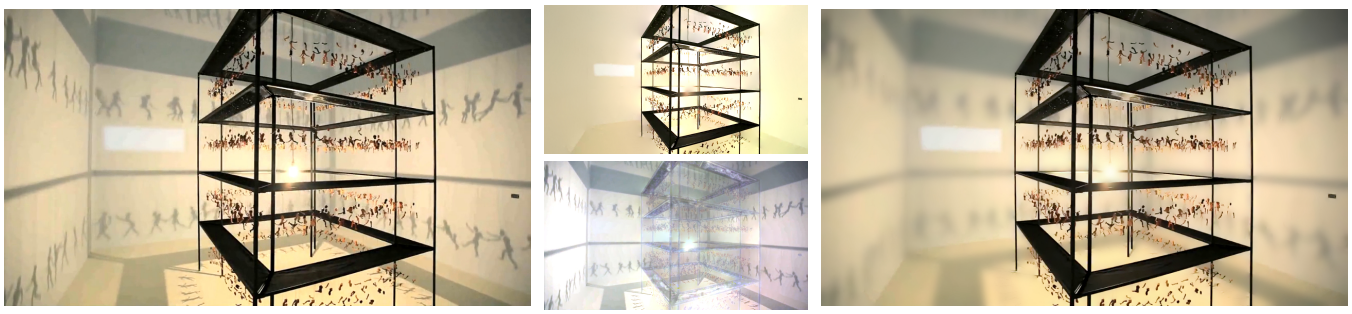
BONNEEL, N., SUNKAVALLI, K., PARIS, S., AND PFISTER, H. 2013. Example-based video color grading. *ACM Transactions on Graphics 32*, 4.

BOUSSEAU, A., PARIS, S., AND DURAND, F. 2009. User-assisted intrinsic images. In *ACM Transactions on Graphics*, vol. 28, 130.

BOYADZHIEV, I., BALA, K., PARIS, S., AND DURAND, F. 2012. User-guided white balance for mixed lighting conditions. *ACM Transactions on Graphics*.

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

Figure 7: *Our decomposition enables easy reflectance editing – re-coloring the girl's t-shirt (top) and painting the SIGGRAPH logo on to the snow (bottom). By separating the two components, we are able to make these edits this while retaining the original illumination in the video.*



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

Figure 8: *We use our decomposition to soften the shadows in this video sequence by filtering the illumination layer.*

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

DONG, W., WANG, Z., JOSEPHSON, W., CHARIKAR, M., AND LI, K. 2008. Modeling lsh for performance tuning. In *ACM Conference on Information and Knowledge Management*, 669–678.

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

FARBMAN, Z., AND LISCHINSKI, D. 2011. Tonal stabilization of video. *ACM Transactions on Graphics 30*, 4.

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

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.

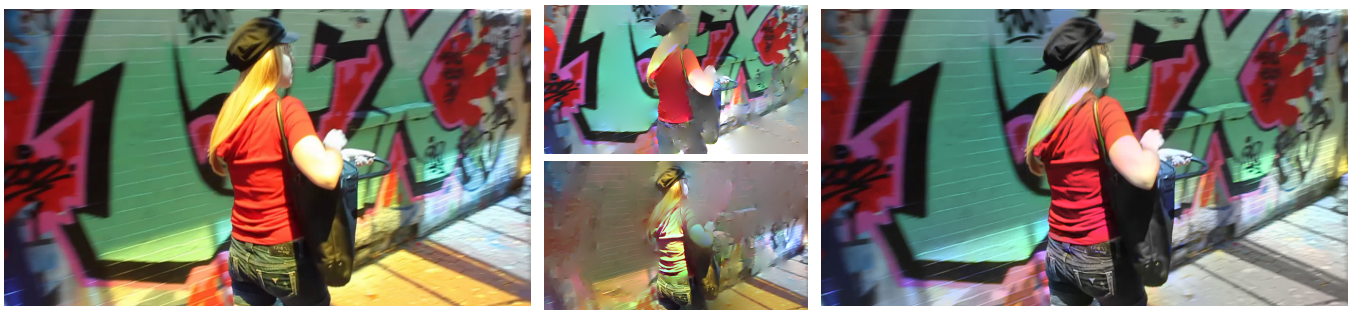
HSU, E., MERTENS, T., PARIS, S., AVIDAN, S., AND DURAND, F. 2008. Light mixture estimation for spatially varying white balance. *ACM Transactions on Graphics*.

KORMAN, S., AND AVIDAN, S. 2011. Coherency sensitive hashing. In *Proc. of the International Conference on Computer Vision*, ICCV.

LAFFONT, P.-Y., BOUSSEAU, A., PARIS, S., DURAND, F., AND DRETTAKIS, G. 2012. Coherent intrinsic images from photo collections. *ACM Transactions on Graphics 31*.

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

LANG, M., WANG, O., AYDIN, T., SMOLIC, A., AND GROSS, M. 2012. Practical temporal consistency for image-based graphics



(a) Input video sequence (b) Intrinsic decomposition (c) Locally white-balanced video

Figure 9: *We apply our method for local white balancing by neutralizing colors of the illumination layer. Despite some residual colors, the strong colored lighting has been largely reduced.*

(a) Input video sequences (b) Intrinsic decomposition (c) Video composite

Figure 10: *Our technique can be used for realistic video compositing. Here, we take two video sequences shot with different viewpoints (a) and decompose them into their respective reflectance and shading components (b). Compositing the two components separately and combining them allows us to create a video composite with realistic shadows and lighting.*

applications. *ACM Transactions on Graphics 31*, 4.

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*. 327–340.

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.

NOCEDAL, J., AND WRIGHT, S. J. 2006. Numerical optimization, second edition. *Numerical optimization*, 497–528.

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 Transactions on Graphics 31*, 4.

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*, vol. 2, IEEE, 68–75.

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.