Intrinsic Video and Applications

Genzhi Ye[1] Elena Garces[2] Yebin Liu[1] Qionghai Dai[1] Diego Gutierrez[2]
[1]Tsinghua University [2]Universidad de Zaragoza

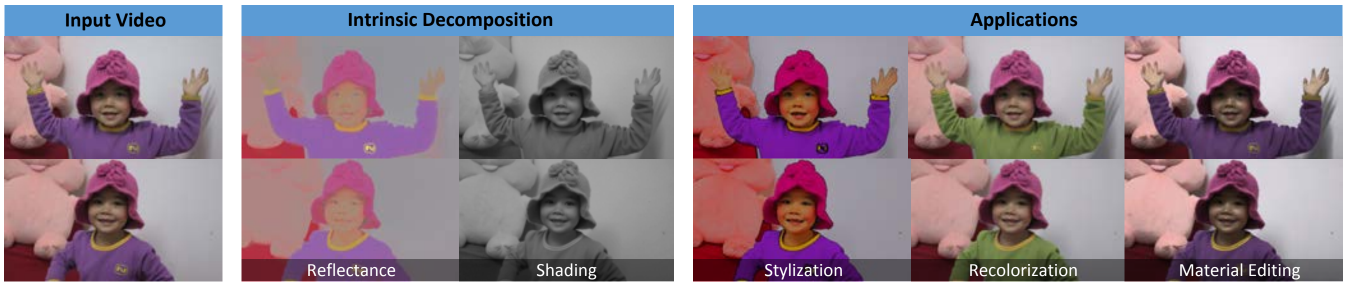

Figure 1: *We propose a novel method for intrinsic video decomposition, which exhibits excellent temporal coherence. Additionally, we show a varied number of related applications such as video segmentation, color transfer, stylization, recolorization, and material editing (the last three included in this image). Please refer to the accompanying video for these and all the other results shown in the paper.*

Abstract

We present a method to decompose a *video* into its intrinsic components of reflectance and shading, plus a number of related example applications in video editing such as segmentation, stylization, material editing, recolorization and color transfer. Intrinsic decomposition is an ill-posed problem, which becomes even more challenging in the case of video due to the need for temporal coherence and the potentially large memory requirements of a global approach. Additionally, user interaction should be kept to a minimum in order to ensure efficiency. We propose a probabilistic approach, formulating a Bayesian Maximum a Posteriori problem to drive the propagation of clustered reflectance values from the first frame, and defining additional constraints as priors on the reflectance and shading. We explicitly leverage temporal information in the video by building a causal-anticausal, coarse-to-fine iterative scheme, and by relying on optical flow information. We impose no restrictions on the input video, and show examples representing a varied range of difficult cases. Our method is the first one designed explicitly for video; moreover, it naturally ensures temporal consistency, and compares favorably against the state of the art in this regard.

CR Categories: I.4.1 [Computer Graphics]: Digitization and Image Capture—Reflectance I.4.8 [Computer Graphics]: Scene Analysis—Shading

Keywords: intrinsic decomposition, temporal coherence, video editing

Links: DL PDF WEB VIDEO DATA

1 Introduction

Decomposing an image into its intrinsic shading and reflectance layers is an ill-conditioned problem with direct applications in computer graphics and image processing, such as retexturing and relighting. In the past few years there have been numerous works tackling this problem from different angles. Some propose fully automatic methods from single images [Tappen et al. 2005; Garces et al. 2012], or rely on user annotations [Bousseau et al. 2009; Shen et al. 2011]. Others leverage information from multiple images [Laffont et al. 2012] or time-lapse sequences [Weiss 2001; Sunkavalli et al. 2007]. Nevertheless, the problem of decomposing a *video* shot into its intrinsic components remains unexplored.

Intrinsic video decomposition is particularly challenging, since temporal coherence must be preserved, even in the presence of dynamic lighting or occluded surfaces coming into view due to object or camera motion. Naively applying any existing intrinsic images algorithm to every individual frame yields poor results, due to the extremely ill-posed nature of the problem and the lack of built-in temporal coherence. Solving the problem on a few keyframes and then interpolating also leads to bad results, since there is no guarantee that resulting reflectance values will be coherent across keyframes. Last, trying to solve the problem globally for the whole sequence would be impractical due to large memory requirements. Another possible approach would be to rely on video segmentation: however, given the rich complexity of video shots, no existing algorithm can guarantee a reliable temporal coherent segmentation. Instead, we focus on an accurate and efficient *propagation* of the reflectance from an initial intrinsic decomposition on the first frame.

Propagating reflectance is different from other propagation approaches (such as video colorization) given the impossibility of building a reliable feature space: The information we wish to propagate (reflectance) is not explicitly coded in the RGB values of the frames and, as we argued, obtaining it on a per-frame basis leads to disturbing flickering artifacts. Instead, we propose a relaxed propagation approach based on a Bayesian framework and solve a Maximum A Posteriori (MAP) problem. To avoid accumulation errors, we define a local confidence threshold and stop the propagation when the number of unreliable pixels surpasses it. We then leverage *shading* information to complete the reflectance layer at the stopping frame, and propagate backwards. We iterate this process using a coarse-to-fine approach.

Our approach has the following desirable characteristics: 1) it is efficient, since at each step it only uses information from the current and previous frame; 2) it takes advantage of information in the temporal dimension given its causal-anticausal scheme; 3) it is temporally stable; 4) it leverages the characteristics of intrinsic images, both from reflectance and shading; and 5) it keeps cumbersome user interaction to a minimum (most of the results shown in this paper are fully automatic, while a few others required a few scribbles on the first frame only). Additionally, we show some video editing applications of our work, including segmentation, recolorization, color transfer, stylization and material editing.

2 Related work

Intrinsic image decomposition from a single image Automatic decomposition of a single image into its intrinsic components usually relies on Retinex theory [Land and McCann 1971], by analyzing local pixel derivatives, to distinguish a change in reflectance from a change of shading [Tappen et al. 2005]. Some methods add priors to a Retinex-based framework, either on the illumination, the reflectance, or both [Jiang et al. 2010; Gehler et al. 2011; Shen and Yeo 2011]. Garces et al. [2012] build clusters of similar chromaticity, from which connections and constraints are defined. Inspired by this, we work on a simplified cluster space of similar *reflectance*, obtained from an initial single-frame decomposition. Lombardi and Nishino [2012] introduce a probabilistic formulation with data-driven and entropy constraints, but the method is limited to obtaining the reflectance of single objects with homogeneous materials. A recent work by Zhao et al. [2012], building on the work by Shen et al. [2008], formulates Retinex as a linear optimization, forcing distant pixels with the same texture to have the same reflectance. A different set of techniques rely on user intervention to constrain the problem by specifying sparse sets of pixels with similar properties [Bousseau et al. 2009; Shen et al. 2011]. Applying any of these techniques on each frame of a video causes disturbing flickering artifacts, due to the lack of built-in temporal coherence mechanisms.

Intrinsic image decomposition from multiple images A few methods try to leverage information from multiple images of the same static scene, and analyze pixel variations under varying illumination [Weiss 2001; Hauagge et al. 2013]. A more flexible solution is given by Laffont et al. [2012], which reconstructs a point-based 3D representation of the scene. Similarly, Liu et al. [2008] use several images of the same scene with the purpose of image colorization. Different from these techniques, our method is designed to work on video sequences, where the key simplifying assumption of a static scene no longer holds.

Intrinsic video decomposition The most similar approach to ours is the work of Yan et al. [2010]. The authors rely on per frame intrinsic decomposition in *local* regions, in order to re-texture specific objects in a video. However, editing a complete video this way would again produce temporal inconsistencies. Specific intrinsic video has only recently been done by Lee et al. [2012], but it requires additional depth information from Kinect. Matsushita et al. [2004] use a simpler approach where the goal is only to eliminate shadows in open scenes, as a pre-processing step for video surveillance, and not to produce a complete intrinsic decomposition. Finally, the heuristic approach of Sunkavalli et al. [2007] decompose a time-lapse sequence into its intrinsic components assuming certain properties of the sky light of the scene. The method works well on static images, but is not able to cope with camera movements.

Video colorization Colorization algorithms are also somewhat related to our problem [Levin et al. 2004; Yatziv and Sapiro 2006; Bhat et al. 2010; Oskam et al. 2012]. However, better results can be obtained for such a task if the intrinsic components are available, where shading information does not interfere in the process. Moreover, these methods usually require user input every few frames; in contrast, our method is fully automatic in most sequences, requiring at most minimal user input only on the first frame.

Temporal consistency Guaranteeing temporal consistency across frames is a challenge for video editing algorithms. Paris [2008] combines isotropic diffusion and Gaussian convolution to adapt classical algorithms like mean shift segmentation or bilateral filtering to video streams. Farbman and Lischinski [2011] propagate values using a combined technique of optical flow and interpolation, for the purpose of tonal stabilization. Recently, Bonneel et al. [2013] applied curvature-flow smoothing in the space of color transformations to transfer color palettes between videos. These techniques are adapted to the particular problems they address, and cannot be easily modified for our purposes. Last, Lang et al. [2012] propose an efficient framework to enforce temporal smoothness across frames. They approximate a global optimization, and show very good results for applications such as disparity estimation, depth upsampling or colorization. However, even after a decent amount of scribbles and parameter tuning, there is an inherent trade-off between the spatial filtering necessary to perform temporal filtering. For the case of intrinsic video, given the large variability across frames of the individual intrinsic decompositions, this causes clear ghosting artifacts, as Figure 11 shows.

3 Overview

Let $f_t(t = 0, 1...)$ be the frames of an input video sequence, with color values $I_{p,t}$ for each pixel p in each time step t. At each frame, the intrinsic decomposition problem can be formulated as finding the reflectance $R_{p,t}$ and shading $S_{p,t}$ layers that satisfy:

$$I_{p,t} = R_{p,t} \times S_{p,t} \qquad (1)$$

where $\times$ denotes per-channel multiplication. The great challenge lies in ensuring temporal coherence of the results, while avoiding the huge memory requirements that analyzing the video as a whole would impose.

We first obtain the reflectance-shading decomposition of the first frame, and obtain clusters of similar reflectance. We then formulate a probabilistic framework that allows us to propagate reflectance values along the frames of the video. In particular, we solve a Maximum A Posteriori (MAP) problem, using a smoothness prior and imposing additional constraints to ensure robustness in the results. At each step, we only propagate values above a certain confidence threshold, and stop the propagation when the number of unknown pixels becomes significant. We follow a coarse-to-fine approach, and perform a *local* decomposition at the stopping frame, using known reflectance values as constraints, and proceed to propagate the new values backward in time. This combined propagation-completion, forward-backward approach allows us to deal with challenging cases like occluded objects.

This process is iterated three times. Last, we apply a smoothness constraint on the shading on the few remaining unassigned pixels, from which we derive the missing reflectance values by simple division. The algorithm then begins again the propagation in a similar manner from the last processed frame, until a new stopping frame is found or the whole sequence is processed. Figure 2 and Algorithm 1 show an overview.

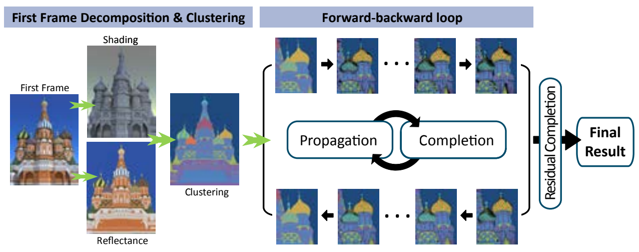

Figure 2: *Overview of our algorithm. We first decompose the initial frame into its intrinsic components. To reduce inconsistencies, we first cluster the initial reflectance values (Section 4). A forward-backward loop propagates this clustered reflectance in the temporal domain, leaving out unreliable pixels (Section 5). After this propagation step, some pixels may still remain unassigned; we rely on Retinex theory to apply shading constraints, thus obtaining the final result (Section 6).*

4 Initial decomposition and clustering

We first aim to find a clustered reflectance decomposition on the first frame, which will be the input to our probabilistic propagation framework. Given the ill-posed nature of the problem, we need a flexible approach that produces good results automatically, while allowing a certain amount of user interaction to improve the results if needed. We base our approach on a common Retinex formulation [Land and McCann 1971]; in particular we extend the single-image formulation by Zhao et al. [2012], who define a global optimization framework and allow user interaction with constant-albedo and constant-shading strokes [Bousseau et al. 2009] (refer to Appendix A for a quick overview). Naively running the algorithm on every frame yields obvious flickering artifacts (see Section 8 and the accompanying video), while extending the optimization to the whole video is prohibitively expensive and may lead to poor results.

We thus extend the original formulation by Zhao et al. in three ways: First, by allowing the user to define *local* optimizations in selected areas of the image. Second, by moving from a pixel-based representation to a more consistent (and efficient) cluster-based approach. And third, by imposing additional constraints during propagation, derived from temporal information in the video. We present details of the first two extensions in the following paragraphs, while the third is explained in Section 5.1.

Local optimizations The global optimization by Zhao and colleagues is based on a global threshold T defined over the whole image, which determines how sensitive the decomposition is to changes in chromaticity. In particular, the authors define a balance factor $\omega_{p,q}$ (where p, q denotes all neighboring pairs of pixels) as follows:

$$\omega_{p,q} = \begin{cases} 0, & \text{if} \parallel J_p - J_q \parallel > T \\ 100, & \text{otherwise} \end{cases} \tag{2}$$

where J denotes chromaticity. We set $T = 10^{-3}$ for all the results shown in the paper. A lower value of T tends to produce an over smooth shading, while higher values assign most of the intensity variations to the shading layer, approximating the results to a chrominance-luminance decomposition. Since in Zhao's formulation the threshold T is defined globally, this may lead to unequal results in different parts of the image.

We thus allow for local variations of T within different regions of the image, which the user identifies by simply drawing approximate

Algorithm 1 Intrinsic Video decomposition

```
1: $[R_0, S_0]$ = IntrinsicImageDecomposition($I_0$)
2: $[\mathcal{C}, \mathcal{I}, \mathcal{R}]$ = ReflectanceClustering($I_0, R_0$)
3: for $each\ iteration$ do
4:     $f_t \leftarrow 1$
5:     repeat
6:         $f_s \leftarrow lastFrame$
7:         /* Forward propagation */
8:         for $f := f_t : lastFrame$ do
9:             $R_f \leftarrow$ ReflectancePropagation($R_{f-1}, \mathcal{C}, \mathcal{I}$)
10:            if StoppingCondition($R_f$) then
11:                $f_s = f$
12:                break
13:            end if
14:        end for
15:        $[R_{f_s}, S_{f_s}] \leftarrow$ ReflectanceCompletion($R_{f_s}, S_{f_s}$)
16:        $[\mathcal{C}, \mathcal{I}, \mathcal{R}] \leftarrow$ ClusteringUpdate($R_{f_s}, \mathcal{C}, \mathcal{R}$)
17:        /* Backward propagation */
18:        for $f := f_s : f_t$ do
19:            $R_f \leftarrow$ ReflectancePropagation($R_{f+1}, \mathcal{C}, \mathcal{I}$)
20:        end for
21:        $f_t \leftarrow f_s + 1$
22:    until $f_s = lastFrame$
23: end for
24: $[R, S] \leftarrow$ ResidualCompletion($R, S$)
25: return $R, S$
```

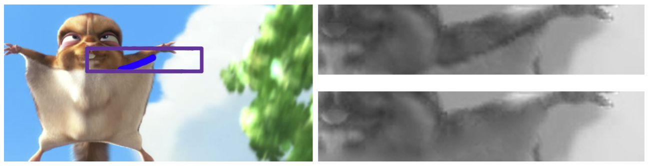

Figure 3: *Local optimization. Using a global threshold T, reflectance variations with low chromaticity change may be misclassified as shading (top-right). By allowing T to vary locally in a user-defined region, this is corrected (bottom-right). The user-defined scribble appears in blue. Video credits: "Big Buck Bunny" (2008) ©Blender Foundation.*

masks (using for instance lazy snapping [Li et al. 2004]). Following Retinex theory, we assume that within a mask shading varies smoothly, which is therefore captured by the lower intensity gradients. We then redefine T locally as $T' = 0.05 \times \mu(\nabla J)$, where μ is the mean of the chromaticity gradient of the pixels inside the mask. Figure 3 shows an example.

Reflectance clustering Given the large volume of data contained in a video, a pure pixel-based approach is inefficient, while being more error-prone and causing temporal instabilities in the form of jittering artifacts. Hence, we *cluster* the initial reflectance decomposition by grouping together sets of pixels sharing similar reflectance values. To reduce memory requirements, we first obtain an over-segmented image, where all pixels in a segment are spatially connected. We use a graph-based segmentation approach [Felzenszwalb and Huttenlocher 2004]. Although this method is very sensitive to changes in parameters, we take advantage of the fact that we only need a rough initial segmentation, and use the same fixed parameters for all our results: size of the Gaussian blur $\sigma = 0.8$, segmentation cut threshold $\mathcal{K} = 50$, and minimum size of the segment $min = 256$. We then group these segments into larger clusters where pixels no longer need to be connected in image space. We iteratively merge two segments if

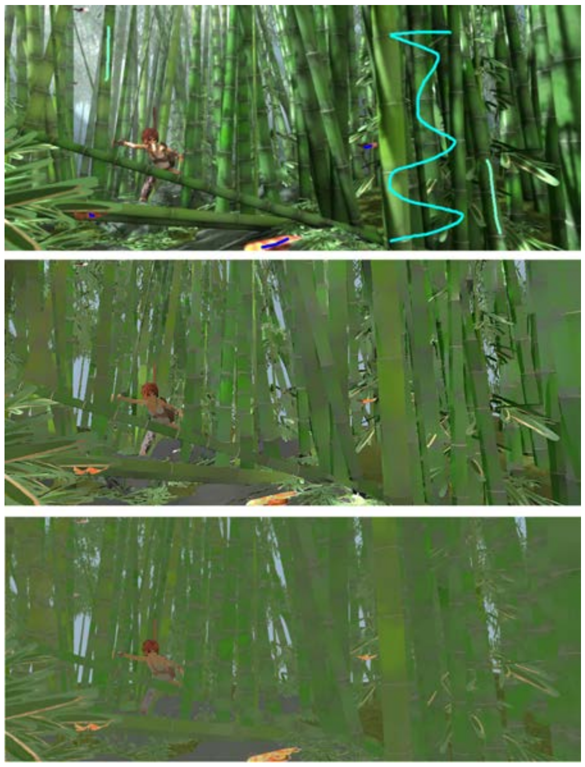

Figure 4: *Reflectance clustering. Top: Input frame with user scribbles for constant-albedo regions. Middle: Reflectance from Zhao et al. [2012]. Bottom: Our reflectance after the clustering step. Note that our reflectance is more consistent and lends itself better to temporal propagation. Video from MPI-Sintel online database [Butler et al. 2012].*

the difference between their average RGB values is smaller than the specified threshold $\mathcal{K}$. This process yields our final reflectance clustering $\mathcal{C}$.

Additionally, we create a suitable data structure for our subsequent reflectance propagation (Section 5). This structure maps image values with clustered reflectance values. For each cluster $C_k \in \mathcal{C}$, we compute two different tables indexed by 3D (RGB) vectors. The first table $\mathcal{R}_k$ is simply a histogram of reflectance values; the function $\rho(k, \mathbf{r})$ returns the number of pixels with reflectance $\mathbf{r}$ in cluster C_k. The second table $\mathcal{I}_k$ stores, on the one hand, a histogram of color values, where the function $\gamma(k, \mathbf{l})$ returns the number of pixels with color $\mathbf{l}$ in cluster C_k; additionally, it stores the associated reflectance $\mathbf{r}_{k,l}$ for each color $\mathbf{l}$. If more than one pixel have the same color but different reflectances in the same cluster, the average reflectance is stored. This helps ameliorate possible inconsistencies in the initial decomposition, yielding a more coherent reflectance. Tables $\mathcal{R}_k$ and $\mathcal{I}_k$ will be later used and updated during our probabilistic reflectance propagation and completion steps (Sections 5 and 6). Figure 4 shows a comparison of our result with the initial decomposition of Zhao et al. [2012].

5 Reflectance Propagation

We now proceed to propagate reflectance values along the temporal dimension, following a coarse-to-fine iterative approach. Reflectance propagation consists of three steps: forward propagation, reflectance completion, and backward propagation. Last, a final residual reflectance completion takes place. Figure 5 illustrates this process.

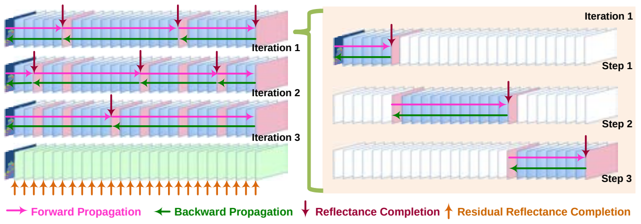

Figure 5: *Reflectance propagation. We propagate reflectance information over successive frames, until a stopping frame is found (depicted in pink). We then apply a reflectance completion step on the stopping frame, and proceed to propagate backwards. We run this basic cycle of forward propagation - reflectance completion - backward propagation until we reach the end of the video, and iterate three times in a coarse-to-fine approach. Last, we perform a residual reflectance completion step.*

5.1 Probabilistic framework

We introduce a Bayesian formulation for reflectance propagation, which enables the integration of reflectance-color statistical inference from tables $\mathcal{R}_k$ and $\mathcal{I}_k$, which are updated at each step, plus a location-reflectance prior from the former frame. Moreover, we jointly optimize pixel clustering with reflectance propagation. With this combined analysis on color, intrinsic reflectance and clustering, we obtain temporally consistent results.

Given a pixel p in frame f_t, with an input RGB color vector $\mathbf{l}_{p,t}$, we aim to find the matching cluster in the previous frame f_{t-1} with the most similar reflectance value (the reflectance of pixel p is unknown at this point), while avoiding over-fitting. We introduce a probabilistic formulation to find the cluster index $\bar{k}$ which maximizes the posterior probability $\mathcal{P}(C_k|p)$. We can define a Maximum A Posteriori (MAP) problem as:

$$\bar{k}(p) = \arg\max_{k} \mathcal{P}(C_k|p) \propto \mathcal{P}(p|C_k) \cdot \mathcal{P}(C_k) \tag{3}$$

where $\mathcal{P}(p|C_k)$ is the likelihood that pixel p belongs to cluster C_k, and $\mathcal{P}(C_k)$ acts as a prior. We discuss these two terms in the following paragraphs.

To limit our search to the most probable candidate regions (thus improving accuracy and efficiency), we leverage the information contained in the video and compute optical flow [Brox et al. 2004] between f_t and f_{t-1}, to obtain the cross-frame motion vector $\boldsymbol{u}_p$. This vector defines the corresponding pixel p' in frame f_{t-1} as $p' = p + \boldsymbol{u}_p$. We then only query the pixels inside an $N \times N$ ($N = 50$) window $W_{p'}$ defined in f_{t-1} and centered at p'. Our MAP problem (Equation 3) is therefore only solved for all the clusters intersecting window $W_{p'}$.

Likelihood $\mathcal{P}(p|C_k)$ The likelihood of a pixel p belonging to a cluster C_k is based on the probability density function of cluster C_k. We formulate a standard non-parametric density estimation problem to estimate the fitness of assigning a pixel to a cluster according to pre-clustered pixels from the previous frame, for which a Parzen window-based approximation is a convenient and effective method. We define our normalized likelihood term $\mathcal{P}(p|C_k)$ as:

$$\mathcal{P}(p|C_k) = \frac{1}{|\mathcal{I}_k|} \sum_{\mathbf{l}} \gamma(k, \mathbf{l}) G\left(\frac{\mathbf{l} - \mathbf{l}_{p,t}}{d}\right) \tag{4}$$

where $|\mathcal{I}_k|$ is the total number of pixels of cluster C_k, and G represents a Parzen window defined by a 3-D Gaussian kernel function, with width $d = 5$. In this form, $\mathcal{P}(p|C_k)$ represents the probability density function of cluster C_k. Figure 6 (c) shows the propagated reflectance at this stage.

However, we note that this definition is biased towards clusters with a large number of pixels. Since each cluster may contain several different reflectances, if we only divide Equation 4 by $|\mathcal{I}_k|$, reflectances with more pixels within the cluster (large $\gamma(k, \mathbf{l})$) will dominate. We thus introduce a unit function U:

$$U(\gamma) = \begin{cases} 1, & \gamma > 0 \\ 0, & \gamma = 0 \end{cases} \tag{5}$$

and redefine $\mathcal{P}(p|C_k)$ as:

$$\mathcal{P}(p|C_k) = \frac{1}{|U(\mathcal{I}_k)|} \sum_{\mathbf{l}} U(\gamma(k, \mathbf{l})) G\left(\frac{\mathbf{l} - \mathbf{l}_{p,t}}{d}\right) \tag{6}$$

Note that we also apply the unit step function on $\mathcal{I}_k$, where $|U(\mathcal{I}_k)|$ represents the number of *non-zero* entries in $\mathcal{I}_k$. Including the unit step function in our formulation makes the likelihood independent of the number of pixels in each cluster, effectively removing bias (see Figure 6 (d)).

Prior $\mathcal{P}(C_k)$ We adopt a smoothness prior common to many MAP solutions. Given the pixel p' computed from p by optical flow, we define $D(p', q')$ as the Euclidean spatial distance between pixels p' and q', where q' belongs to the same frame as p' and satisfies the following three conditions: $q' \in W_{p'}$, $q' \in C_k$ and $q' \neq p'$. Our prior thus becomes:

$$\mathcal{P}(C_k) = \min_{q'} \left(D(p', q')\right)^{-\frac{1}{2}} \tag{7}$$

Note that $\mathcal{P}(C_k)$ is independent of the intrinsic properties of a pixel, and can be calculated after the former frame has been processed: it thus acts as a prior even if it is not explicitly formulated as a probability. Figure 6 (e) shows how the smoothness prior improves the reflectance propagation.

5.2 Additional constraints

Solving our MAP problem in Equation 3 for frame f_t, we obtain the cluster candidate $\bar{k}(p)$ indicating the best cluster match for pixel p. However, assigning the corresponding cluster index to every pixel before proceeding to the next frame would accumulate errors over time (as Figure 12 in Section 8 shows). This is in part due to factors such as antialiased edges, non-rigid motion, occluded surfaces or time-varying shading, which increase the number of pixels in a given frame without a suitable cluster in the previous frame.

To avoid this, we take a conservative approach and only assign a definite cluster index to those pixels above a *high confidence* threshold. We again leverage the extra information in the temporal domain and postpone the judgment of the unresolved pixels to subsequent processing (Section 6). To define our high confidence threshold, we analyze two conditions: We first check whether the posterior of $\bar{k}$ satisfies $\mathcal{P}(C_{\bar{k}}|p) > \varepsilon$. We empirically set the threshold $\varepsilon = 0.8$ (the maximum value of $\mathcal{P}$ is 1.0). Additionally, we ask how close this probability is to the probability associated to the second-best cluster $C_{\bar{k}'}$, by checking whether $\mathcal{P}(C_{\bar{k}}|p) > \beta\mathcal{P}(C_{\bar{k}'}|p)$. We fix $\beta = 2$, which we found to work well for all the results shown in the paper and the supplementary video.

If these two conditions are satisfied, pixel p is assigned to a cluster, and its reflectance value is set to $\mathbf{r}_{\bar{k},l}$. In this fashion, we successfully assign about 95% of the pixels for each frame. The reflectance

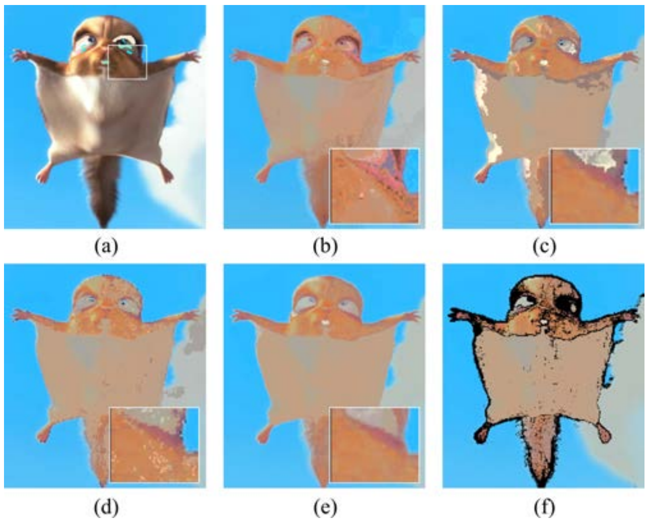

Figure 6: *Effect on reflectance propagation of every component of our algorithm: (a) Original color frame (including user strokes for constant-albedo regions; highlighted area enlarged in the insets). (b) Reflectance propagation without clustering. (c) Reflectance propagation using Eq. 4. (d) Reflectance propagation using Eq. 6, adding the step function U. (e) Reflectance propagation using both Eq. 6 and the distance term of Eq. 7. (f) Removing low-confidence reflectance using Section 5.2 on the result of (e).*

of the rest of the pixels is left undefined, and will be assigned later using information from other frames. Figure 6 (f) shows the result after removing the pixels with low-confidence reflectance.

5.3 Stopping condition

We continue propagating reflectance values on successive frames $f_{t+1}, f_{t+2}, \cdots$, following the same approach. Unassigned pixels in f_t remain undefined in subsequent frames, so the total number of such pixels increases as the propagation continues forward. As a consequence, applying our propagation algorithm over long stretches in the video sequence may lead to poor results, with too many unassigned pixels in the end.

To bound the number of unassigned pixels, we define a stopping condition for our forward reflectance propagation, and stop when a given threshold is reached. Instead of relying on the total number of unassigned pixels over the whole image, we take a local approach and stop when the ratio of unassigned pixels is greater than $\alpha = 80\%$ in any local $M \times M$ window W_M on the current frame. This allows for timely detection of growing *regions* of unassigned pixels, which would be too difficult to solve later if a global threshold had not been reached yet. We begin with a value of $M = 30$, which will be progressively reduced in subsequent iterations (see Subsection 6.1). Figure 7 (a) shows the result of the forward propagation once the stopping condition has been met. The remaining undefined pixels of all the processed frames are left to later processing, explained in the following sections.

6 Reflectance Completion

Once the propagation has stopped at frame f_s, we perform a *local* intrinsic decomposition on the sparse set of unknown pixels in that frame. We take advantage of the propagated reflectance values, and add them as constraints into our framework. We first define a mask Ω^0 containing all the unknown pixels of frame f_s, and perform

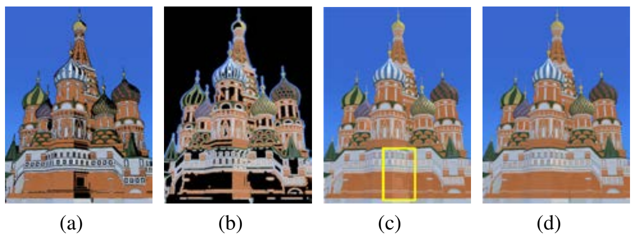
Figure 7: *Reflectance completion: (a) Forward reflectance stops at frame $f_s = f_{31}$. Unassigned pixels are depicted in black. (b) Partial, constrained reflectance completion over Ω^*. (c) Simply combining the forward propagation reflectance with the partial completed reflectance causes reflectance discontinuities (see for instance the yellow rectangle). (d) The result of our Poisson reflectance smoothing.*

a morphological expansion with a radius of four pixels, so that the new expanded mask Ω^* now also contains known reflectance values (see Figure 7 (b)). We denote $\Omega^1 = \{\Omega^* - \Omega^0\}$ as the set of pixels with known reflectance, which are included as constraints adding a new term $\lambda_c E_c$ to Equation 12 (see the Appendix), where:

$$E_c = \sum_{p \in \Omega^1} (r_p - r_p^1)^2 \tag{8}$$

where r_p and r_p^1 are the log-reflectance values of the unknown and known pixels respectively. We set $\lambda_c = 1000$.

To avoid discontinuities introduced by the new reflectance values (see Figure 7 (c)), we apply Poisson blending [Pérez et al. 2003] with Dirichlet boundary conditions on the arbitrary outlines of the regions:

$$\Delta R_c = \Delta R^1 \ over\ \Omega^0,\ with\ R_c|_{\partial\Omega^0} = R^1|_{\partial\Omega^0} \tag{9}$$

where Δ is the Laplacian operator, and R^1 and R_c are the known and the newly obtained (completed) reflectance values on Ω^0, respectively. Figure 7 (d) shows the final result.

Clustering update We now need to find matching clusters C_k for the pixels with new reflectance values. We rely on table $\mathcal{R}_k$ since the variation of pixel values in reflectance space is smaller than in the original color space. Similar to Equation 6, we calculate the likelihood of pixel p belonging to cluster C_k by maximizing:

$$\mathcal{P}(p|C_k) = \frac{1}{|U(\mathcal{R}_k)|} \sum_{\mathbf{r}} U(\rho(k,\mathbf{r})) G\left(\frac{\mathbf{r} - \mathbf{r}_{p,s}}{d}\right) \tag{10}$$

where $\mathbf{r}_{p,s}$ is the reflectance of pixel p at frame f_s, and $|U(\mathcal{R}_k)|$ is the number of non-zero entries of $\mathcal{R}_k$ for cluster C_k. Note that in this case we cannot rely on our smoothness prior in Equation 7, since nearby pixels in the previous frame will likely be unassigned also. We thus do not limit the search to local windows, but search all clusters instead. Additionally, since the completion works robustly, we do not need to enforce the conservative constraints in Section 5.2 before assigning a cluster. Once the index $\bar{k}$ with maximum probability has been found, we update the corresponding $\mathcal{I}_{\bar{k}}$, $\mathcal{R}_{\bar{k}}$ tables to ensure an effective subsequent propagation.

Note that thanks to this updating process, our reflectance propagation of frame k is not entirely determined by its adjacent frame only. In our probability formulation (Equation 3), the first term (Equation 6) is based on the global histogram $\mathcal{I}_k$, which is constantly updated based on information from all the frames that have already been traversed. The smoothness term in Equation 7 does depend only on the previous frame.

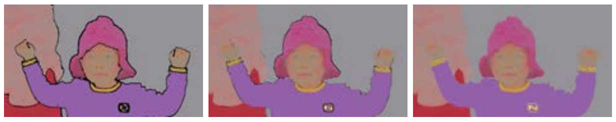
Figure 8: *Results of the iterative propagation and final completion. Left: Unassigned (black) pixels after the first iteration. Middle: Reduced number of unassigned pixels after the three iterations. Right: Final result after reflectance completion.*

6.1 Coarse-to-Fine Propagation

Our next steps complete the remaining unassigned pixels, following a coarse-to-fine approach. From the (partially) completed f_s we first launch a backward propagation towards f_t. We follow the same basic approach as the forward propagation, based on the color table $\mathcal{I}_k$ and using again our Bayesian framework defined in Equation 3. Note that at each step, we only propagate the reflectance to pixels that have not been assigned yet, leaving the rest unchanged. Intuitively, this backward propagation helps assign correct reflectance values to occluded objects in the forward propagation: working backwards in time, these are visible from the beginning.

This forward-backward propagation scheme is then iterated to complete the rest of the unassigned pixels. At each iteration, we decrease the threshold for the stopping condition: specifically, we maintain the threshold ratio $\alpha = 80\%$, but progressively reduce the size of the window W_M. We adopt a three-pass iteration scheme, using $M = 30, 20,$ and 10 respectively. This offers a good trade-off to complete unassigned pixels effectively without needing to run our reflectance completion algorithm at each frame. Figure 8 (left and middle) shows an example.

Residual Reflectance Completion While we could continue the iteration until all pixels are assigned a reflectance value, this may be impractical or error-prone in some difficult cases. Instead, we rely on Retinex theory and leverage *shading* information. We assume C^1 shading continuity on the unassigned regions, an assumption that has been used in previous Retinex-based works [Funt et al. 1992; Shen and Yeo 2011; Gehler et al. 2011]; in our case this works particularly well given the small size of the remaining unassigned areas (less than 1% in our experiments).

Similar to our reflectance completion step, we impose smooth interpolation on the domain Ω^0 of unassigned pixels by applying the following Poisson equation:

$$\Delta S_c = 0\ over\ \Omega^0,\ with\ S_c|_{\partial\Omega^0} = S^1|_{\partial\Omega^0} \tag{11}$$

where S^1 and S_c are the known and the new completed shading values respectively. The final reflectance values are simply obtained by a pixel-wise division of color over shading (Figure 8, right).

7 Evaluation

We first evaluate our algorithm by comparing it to a ground truth, synthetic example. We have rendered a 3D model of *St. Basil* with dynamic lighting and a rotating camera, using $Maya$. The ground truth reflectance and shading layers are obtained assigning a constant and a white Lambertian shader, respectively. Figure 9 (left), shows the comparison between the rendered ground truth sequence

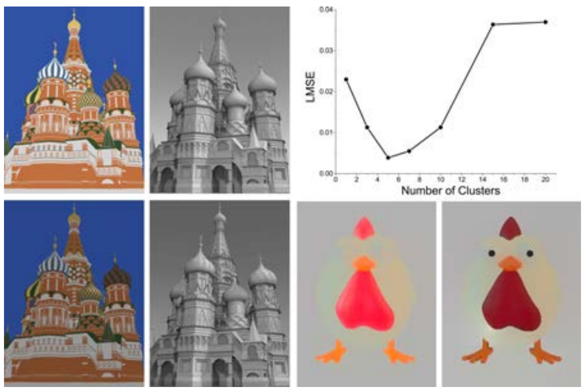

Figure 9: *Left: Comparison between synthetic ground truth for reflectance and shading (top) and our corresponding results (bottom). Top-right: The evolution of the Local Mean Squared Error (LMSE) with the number of initial reflectance clusters. Bottom-right: Comparison between chrominance (left) and our reflectance layer (right). Note how the reflectance of the materials is much better depicted with our method.*

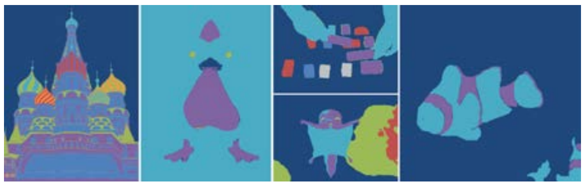

Figure 10: *Representative clusters of the videos included in this paper. The parameters of the segmentation are the same for all the sequences (details in Section 4).*

and our algorithm; the entire animation is included in the video. Figure 9 (top-right) shows a representative graph of the evolution of the Local Mean Squared Error (LMSE) with the number of initial reflectance clusters. We have found that the error is consistently minimized with 5-10 clusters in all our sequences. With too few clusters, the algorithm cannot differentiate reflectances properly, whereas a larger number increases error since the second condition in our high confidence threshold (Section 5.2) can hardly be met. Moreover, in Figure 9 (bottom-right) we compare our reflectance results for *chicken* against its chromaticity channels. Our reflectance represents the underlying materials more truthfully, while being able to separate black features (no chromaticity) like the pupils. Figure 10 shows some representative clusters of the scenes included in this paper.

Although ours is the first intrinsic decomposition method devised to work with video sequences, we objectively compare against four other alternatives, by plotting the LMSE with respect to the ground truth for the *St. Basil* sequence. In particular, we first compare against two state-of-the-art methods devised for single images [Zhao et al. 2012; Garces et al. 2012], decomposing each frame of the video individually. Figure 12, left (black and blue lines) shows the results. Because of motion-induced occlusions and time varying shading, the results of a frame-by-frame approach are inevitably unstable, with large, sudden changes between frames. These spikes in error translate into disturbing flickering artifacts even for the best of the two algorithms, as the accompanying video shows. Our method yields very stable results, with the lowest LMSE (red line).

The green line (also in Figure 12, left) shows the result of applying the recent method by Lang et al. [2012] over the frames re-

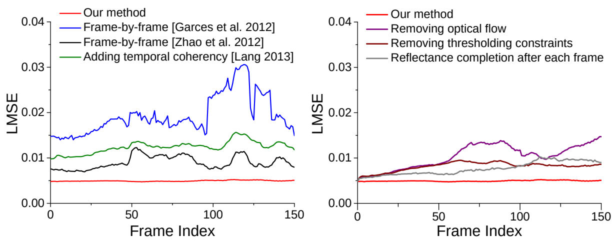

Figure 12: *Left: Comparison of LMSE of our algorithm (red line) and three other different approaches, including two per frame decompositions [Zhao et al. 2012; Garces et al. 2012] and the recent method by Lang et al [2012] which enforces temporal stability over Zhao's per frame result. Our algorithm yields the most stable results, avoiding flickering, and the lowest LMSE. Right: Different variations of our algorithm; removing or altering some of its key components leads to temporal instability and larger error.*

turned by Zhao's method. The resulting error curve is somewhat smoother than frames processed individually, indicating overall improved temporal stability (although some of the largest error peaks remain). However, the overall error increases compared to per-frame decomposition by Zhao's method. The reason is that Lang's method is not well suited for the particular case of intrinsic images since, as we have seen, this initial per-frame decomposition shows very large differences between frames, which in turn forces a very aggressive smoothing in the temporal domain. The spatial smoothing that the method imposes along with its temporal filtering, leads to clear ghosting artifacts and over-blurred edges. Figure 11 shows some direct comparisons with our method for two different sequences.

Figure 12, right, shows the LMSE of different variations of our algorithm to highlight the influence of its most relevant components. It can be seen how performing reflectance completion after each frame (gray line), removing our conservative high-confidence threshold defined in Section 5.2 (deep red) or removing optical flow (purple line) increase both the overall error and the temporal instability of the results.

8 Results and Applications

Figure 13 shows additional results (we refer the reader to the accompanying video for the complete set). All our results have been produced automatically, unless user scribbles are shown on the initial frame. Their frame length varies between 100 and 500 frames, which is in accordance to the average shot in modern TV and movies [Lang et al. 2012]. The initial decomposition runs at interactive rates. Since we only propagate frames until the local error threshold is surpassed, we have experienced no error accumulation or temporal drift in any of the videos tested. Note also how in the *chicken* sequence parts of the body disappear and appear again, which our algorithm handles gracefully thanks to its forward-backward structure. In *objects*, the camera loops around the scene; notice the temporal coherence of the reflectance images, including the first and last frames of the sequence (leftmost and rightmost frames in the image). In the video, *Squirrel* presents a particularly challenging case including a furry animal, moving shadows and motion blur, but our algorithm still yields good results.

For one frame in a video sequence at 800 × 600 spatial resolution, our algorithm takes slightly over one minute using a four-threaded unoptimized implementation on a standard PC. Automatic decomposition and reflectance clustering of the first frame requires about 30 seconds, plus 20 seconds for the forward propagation in the first

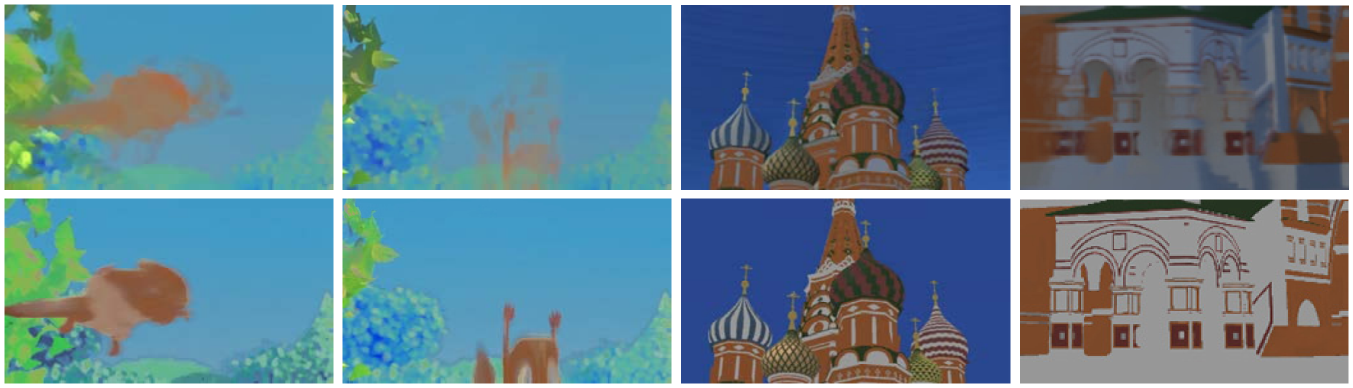

Figure 11: *Comparison of Lang's method [2012], imposing temporal coherency to the decomposition of each individual frame, (top) and our result (bottom) for the squirrel and St. Basil sequences. The built-in spatial filtering in Lang's method, coupled with the large variability across frames of per-frame decompositions, cause some clear ghosting artifacts, more exaggerated in the presence of quick motion (squirrel). Additionally, fine features become blurry (see the rightmost comparison for St. Basil).*

iteration and 5 seconds for each reflectance completion and residual reflectance completion steps. Since the propagation for each pixel is neighbor-independent, the algorithm is suitable for parallel processing and could be significantly accelerated on a GPU.

Moreover, our intrinsic video decomposition method can be used as a basic platform for video editing applications. Here we show some examples, and refer the reader again to the supplemental video.

Video segmentation Available video segmentation methods rely heavily on local color and shape information; therefore such methods usually fail to track an object in natural videos where illumination or shape may change. Our algorithm can be used for object segmentation based on propagated *clustered reflectance* information. Since the reflectance propagation effectively removes the shading component, it is more robust in case of changes in illumination. Moreover, our method does not rely on local statistics, which makes it also robust in case of fast topology changes of the segmented objects. Figure 15 shows the results for five frames of the *nemo* sequence, equally spaced 225 frames apart, compared with the result using the popular video SnapCut [Bai et al. 2009] and the online efficient hierarchical graph-based video segmentation (EHGV) tool [Grundmann et al. 2010]. For a fair comparison with these methods, in the SnapCut example the first frame has been manually labeled as foreground and background, in order to provide visually comparable results, while for EHGV we use the automatic cluster result directly. It can be seen how both SnapCut and EHGV progressively accumulate error and completely miss at least one of the fishes in the end. In contrast, our method improves performance in the presence of fast motion, dramatic changes in shape, and similar color between foreground and background objects, even over a large number of frames, without error drift.

Material editing We can re-render the surface materials by manipulating the shading layer, defining a simple mapping function between the original and the new shading. The user only needs to define a sparse set of control point in the shading grayscale space, and the mapping function is obtained by cubic interpolation. Figure 14, top-left, shows how diffuse surfaces can be made to appear shiny by applying the function shown.

Color transfer Given a source image of different scene, our intrinsic decomposition allows us to efficiently transfer its color and tone to a target video, by simply performing histogram matching on the reflectance layers of both the source image and the first frame of the video. Our algorithm efficiently propagates this new reflectance information; multiplying by the corresponding shading images yields the final color-transferred video. Given our temporally consistent cluster information, we can further composite the original foreground object onto the new background (Figure 14, bottom-left).

Recolorization By adding simple scribbles on the input image (Figure 14, top-right), users can define a reflectance transfer function between the input and the target reflectance values. In our implementation, the target reflectance values are defined by the color of the scribbles. For each pixel in any subsequent frame, this transfer function is applied to the original reflectance. Note that similar effects have been achieved using propagation techniques [Li et al. 2010; Xu et al. 2009], although here we directly manipulate the reflectance layer.

Stylization We can easily achieve interesting non-photorealistic results by manipulating each intrinsic layer separately. Figure 14, bottom-right, shows two different depictions of the same video, increasing and decreasing the saturation of the reflectance layer, respectively, while flattening the shading layer. An edge layer has been subsequently added to enhance the effect.

9 Discussion

In summary, we have presented a novel approach for the challenging problem of intrinsic video decomposition, as well as several example applications in video editing. Our method is temporally coherent, does not impose a large memory footprint, and does not require heavy user interaction. We believe this is the first work successfully addressing this problem, without imposing any restrictions on the input videos.

Our approach is not free of limitations. Working on a per-frame basis allows our method to be very light on memory requirements and processing time; however it is currently not optimized for speed, and it still does not work in real time. Similar to other video editing approaches, different shots need to be processed separately: if the content between shots varies drastically (i.e., two completely different scenes in a movie), our propagation scheme cannot guarantee good results. Last, if the initial frame is very dark or very saturated, it may be hard to find meaningful reflectance clusters. Despite this, we hope our work inspires both future research on intrinsic decomposition of video sequences, as well as novel video editing techniques that take advantage of the individual manipulation of reflectance and shading information.

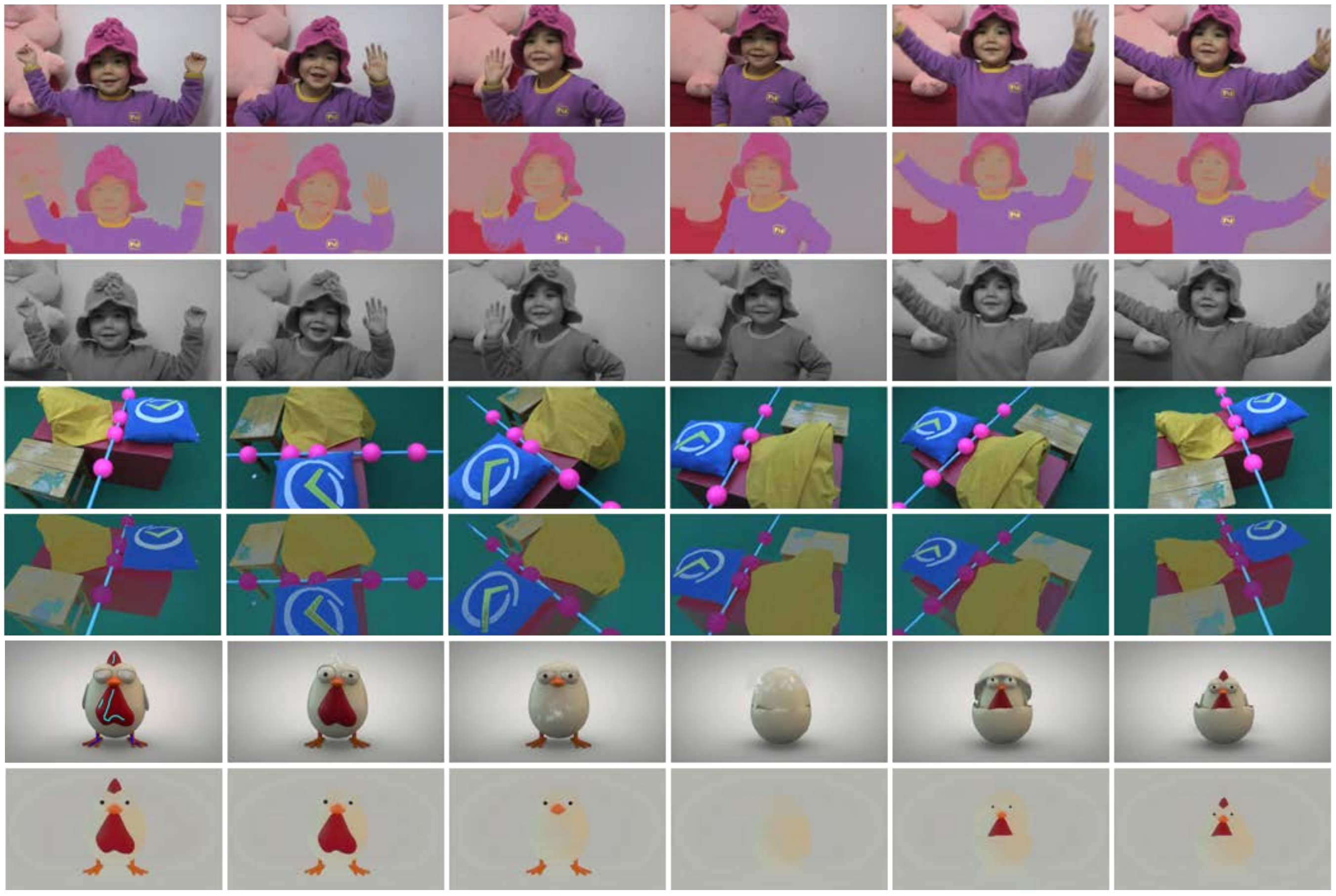

Figure 13: *Results of our intrinsic video decomposition for the dancing baby, objects and chicken sequences (including strokes for chicken for constant-shading regions in gray and for constant-albedo in color). Please refer to the video for these and more complete examples. Video Credits (bottom): "Chicken or Egg First?" (2013) ©Medreza Assegaf.*

Acknowledgements

We want to thank the reviewers for their insightful comments, Steven Poulakos for the comparisons with Lang et al.'s work, and Adrian Jarabo, Xiang Yu and Pierre-Yves Laffont for their help in different video sequences. We are grateful for the original video from Nick Hope: Symbiosis & Anemonefish - Reef Life of the Andaman - Part 18 (used with permission), Medreza Assegaf: Chicken or Egg First? (used with permission). This research has been funded by the NSFC (projects No.61327902, 61035002, and 61120106003), the 863 Program (No.2013AA01A604), and the Beijing Key Laboratory of Multi-dimensional & Multi-scale Computational Photography (MMCP), as well as the European Commission (7FP, projects GOLEM and VERVE), and the Spanish Ministry of Science and Technology (TIN2010-21543). The Gobierno de Aragón additionally provided support through the TAMA project and a grant to Elena Garces.

References

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

BHAT, P., ZITNICK, C. L., COHEN, M., AND CURLESS, B. 2010. Gradientshop: A gradient-domain optimization framework for image and video filtering. *ACM Trans. Graph. 29*, 2.

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 Asia) 28*, 5, 130.

BROX, T., BRUHN, A., PAPENBERG, N., AND WEICKERT, J. 2004. High accuracy optical flow estimation based on a theory for warping. In *Proc. ECCV*, Springer, 25–36.

BUTLER, D. J., WULFF, J., STANLEY, G. B., AND BLACK, M. J. 2012. A naturalistic open source movie for optical flow evaluation. In *Proc. ECCV*, Springer, 611–625.

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

FELZENSZWALB, P. F., AND HUTTENLOCHER, D. P. 2004. Efficient graph-based image segmentation. *International Journal of Computer Vision 59*, 2, 167–181.

FUNT, B. V., DREW, M. S., AND BROCKINGTON, M. 1992. Recovering shading from color images. In *Proc. ECCV*, Springer, 124–132.

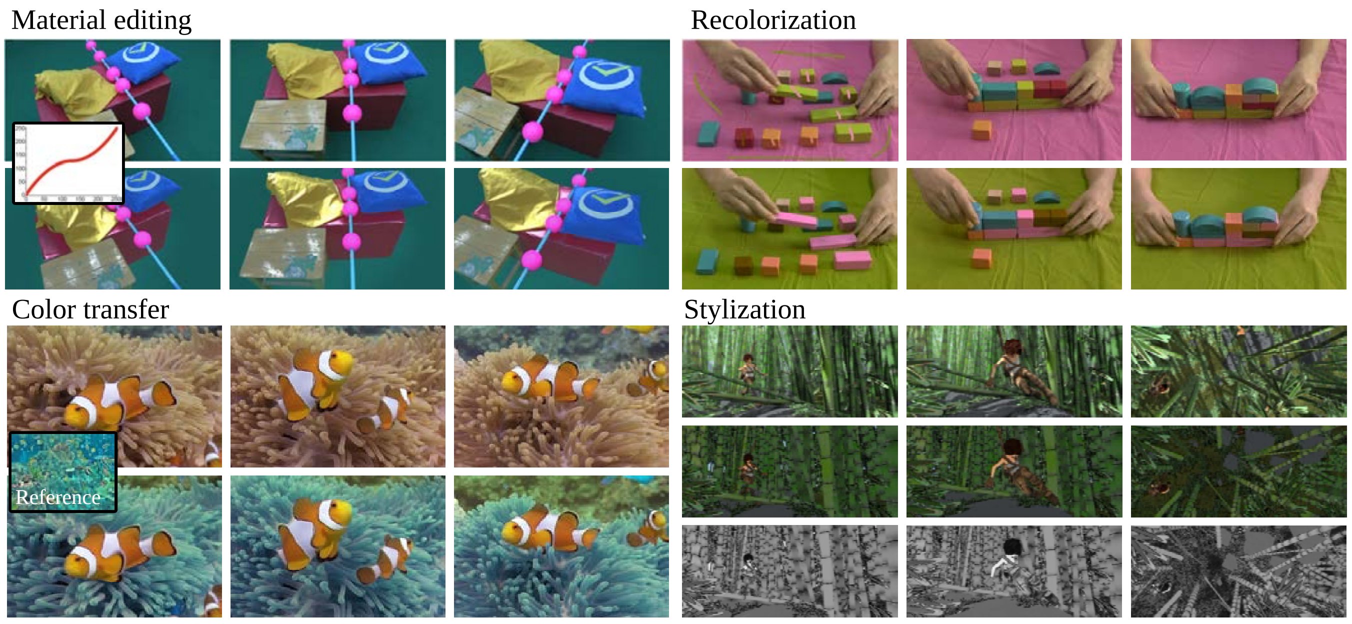

Figure 14: *More example applications of our intrinsic video decomposition. The two small insets show the function mapping input and output shading (material editing), and the source image (color transfer). In the recolorization example, the target reflectance value is defined by the color of the scribble. Video Credits (bottom-left): "Symbiosis & Anemonefish - Reef Life of the Andaman - Part 18 " (2012) ©Nick Hope.*

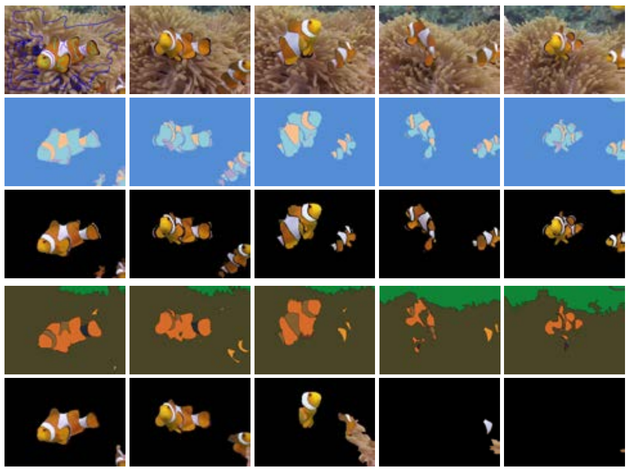

Figure 15: *Example and comparison of automatic video segmentation. First row: Input image frames (first column input strokes for constant-albedo regions). Second row: Our automatic clustered reflectance. Third row: Our result labeling foreground/background. Fourth row: Result of automatic EHGV segmentation [Grundmann et al. 2010]. Fifth row: Result of using SnapCut [Bai et al. 2009] labeling foreground/background.*

GARCES, E., MUNOZ, A., LOPEZ-MORENO, J., AND GUTIERREZ, D. 2012. Intrinsic images by clustering. In *Computer Graphics Forum (EGSR)*, vol. 31, 1415–1424.

GEHLER, P. V., ROTHER, C., KIEFEL, M., ZHANG, L., AND SCHÖLKOPF, B. 2011. Recovering intrinsic images with a global sparsity prior on reflectance. In *Proc. NIPS*, 765.

GRUNDMANN, M., KWATRA, V., HAN, M., AND ESSA, I. 2010. Efficient hierarchical graph-based video segmentation. In *Proc. CVPR*, IEEE.

HAUAGGE, D., WEHRWEIN, S., BAVAL, K., AND SNAVELY, N. 2013. Photometric ambient occlusion. In *Proc. CVPR*, IEEE.

JIANG, X., SCHOFIELD, A. J., AND WYATT, J. L. 2010. Correlation-based intrinsic image extraction from a single image. In *Proc. ECCV*, Springer, 58–71.

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*.

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

LANG, M., WANG, O., AYDIN, T., SMOLIC, A., AND GROSS, M. 2012. Practical temporal consistency for image-based graphics applications. *ACM Trans. Graph. (SIGGRAPH) 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 *Proc. ECCV*, Springer, 327–340.

LEVIN, A., LISCHINSKI, D., AND WEISS, Y. 2004. Colorization using optimization. *ACM Trans. Graph. (SIGGRAPH) 23*, 3.

LI, Y., SUN, J., TANG, C.-K., AND SHUM, H.-Y. 2004. Lazy snapping. *ACM Trans. Graph. (SIGGRAPH) 23*, 3, 303–308.

LI, Y., JU, T., AND HU, S.-M. 2010. Instant propagation of sparse edits on images and videos. In *Computer Graphics Forum*, vol. 29, Wiley Online Library, 2049–2054.

LIU, X., WAN, L., QU, Y., WONG, T.-T., LIN, S., LEUNG, C.-S., AND HENG, P.-A. 2008. Intrinsic colorization. *ACM Trans. Graph. (SIGGRAPH Asia) 27*, 5, 152:1–152:9.

LOMBARDI, S., AND NISHINO, K. 2012. Reflectance and natural illumination from a single image. In *Proc. ECCV*, Springer, 582–595.

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

OSKAM, T., HORNUNG, A., SUMNER, R., AND GROSS, M. 2012. Fast and stable color balancing for images and augmented reality. In *Proc. 3DIMPVT*, IEEE, 49–56.

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

PÉREZ, P., GANGNET, M., AND BLAKE, A. 2003. Poisson image editing. *ACM Trans. Graph. (SIGGRAPH) 22*, 3, 313–318.

SHEN, L., AND YEO, C. 2011. Intrinsic images decomposition using a local and global sparse representation of reflectance. In *Proc. CVPR*, IEEE, 697–704.

SHEN, L., TAN, P., AND LIN, S. 2008. Intrinsic image decomposition with non-local texture cues. In *Proc. CVPR*, IEEE, vol. 0, 1–7.

SHEN, J., YANG, X., JIA, Y., AND LI, X. 2011. Intrinsic images using optimization. In *Proc. CVPR*, IEEE, 3481–3487.

SUNKAVALLI, K., MATUSIK, W., PFISTER, H., AND RUSINKIEWICZ, S. 2007. Factored time-lapse video. *ACM Trans. Graph. (SIGGRAPH) 26*, 3, 101.

TAPPEN, M., FREEMAN, W., AND ADELSON, E. 2005. Recovering intrinsic images from a single image. *IEEE Trans. Pattern Anal. Mach. Intell. 27*, 9, 1459–1472.

WEISS, Y. 2001. Deriving intrinsic images from image sequences. In *Proc. ICCV*, IEEE, vol. 2, 68–75.

XU, K., LI, Y., JU, T., HU, S.-M., AND LIU, T.-Q. 2009. Efficient affinity-based edit propagation using kd tree. In *ACM Trans. Graph. (SIGGRAPH Asia)*, vol. 28.

YAN, X., SHEN, J., HE, Y., AND MAO, X. 2010. Re-texturing by intrinsic video. In *Proc. DICTA*, IEEE, 486–491.

YATZIV, L., AND SAPIRO, G. 2006. Fast image and video colorization using chrominance blending. *IEEE Trans. Image Processing, 15*, 5, 1120–1129.

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

A Retinex-based Optimization

We summarize here the main aspects of the paper by Zhao and colleagues [2012] relevant to our work. The problem of intrinsic decomposition of a single image is posed as an optimization. Working in log-space, the problem is defined as $i_p = s_p + r_p$, where $i_p = \log(I_p), r_p = \log(R_p)$ and $s_p = \log(S_p)$ (I_p, R_p and S_p are the image, reflectance and shading pixel values, according to our Equation 1). The following function is then minimized:

$$\arg\min_s E(s) = \lambda_l E_l(s) + \lambda_r E_r(s) + \lambda_a E_a(s) \tag{12}$$

where λ_l, λ_r and λ_a are positive weights (set to 1, 10000 and 1000 respectively), $E_l(s)$ represents the common Retinex constraint minimizing the differences in shading and reflectance between adjacent pixels, $E_r(s)$ is a non-local albedo constraint and $E_a(s)$ is a normalization factor. Explicitly:

$$E_l(s) = \sum_{(p,q)\in N} \left[(s_p - s_q)^2 + w_{(p,q)}(r_p - r_q)^2\right] \tag{13}$$

where N denotes the set of all neighboring pairs of pixels and $w_{(p,q)}$ is the balance factor introduced in Equation 2. The next term is:

$$E_r(s) = \sum_{G_r^i\in\Gamma_r} \sum_{(p,q)\in G_r^i} (r_p - r_q)^2 \tag{14}$$

where G_r^i is a set of pixels with similar albedo, and Γ_r is the set of pixel groups previously detected to share the same albedo. Last, the normalization term is:

$$E_a(s) = \sum_{p\in G_a} (s_p - 1)^2 \tag{15}$$

where G_a contains the brightest pixel(s).