Optimizing Color Consistency in Photo Collections

Yoav HaCohen
Hebrew University

Eli Shechtman
Adobe Research

Dan B Goldman
Adobe Research

Dani Lischinski
Hebrew University

Figure 1: *Editing a photo collection with our method. First row: input images exhibiting inconsistent appearance. Red arrows indicate pairs of images that were detected to share content. Second row: automatically induced consistent appearance. Third row: after propagating user adjustment of the leftmost photo (photos with similar content are affected more strongly). Fourth row: propagation of an adjustment done to the sixth photo. Previous adjustment remains as constraint. (Note: adjustments are deliberately exaggerated in this example.)*

Abstract

With dozens or even hundreds of photos in today's digital photo albums, editing an entire album can be a daunting task. Existing automatic tools operate on individual photos without ensuring consistency of appearance between photographs that share content. In this paper, we present a new method for consistent editing of photo collections. Our method automatically enforces consistent appearance of images that share content without any user input. When the user does make changes to selected images, these changes automatically propagate to other images in the collection, while still maintaining as much consistency as possible. This makes it possible to interactively adjust an entire photo album in a consistent manner by manipulating only a few images.

Our method operates by efficiently constructing a graph with edges linking photo pairs that share content. Consistent appearance of connected photos is achieved by globally optimizing a quadratic cost function over the entire graph, treating user-specified edits as constraints in the optimization. The optimization is fast enough to provide interactive visual feedback to the user. We demonstrate the usefulness of our approach using a number of personal and professional photo collections, as well as internet collections.

Links: DL PDF WEB

1 Introduction

The ease with which we are able to take digital photographs presents both an opportunity and a challenge. We capture dozens to hundreds of images – often from multiple cameras – during a single event such as a day hike or a dinner party. Many of these images could clearly benefit from adjustments to color and contrast, but manually adjusting each photo is hardly an option. Automatic enhancement tools exist, but they operate on each image independently, without attempting to ensure *consistency of appearance* between photographs depicting the same subject or scene.

Inconsistent appearance of photos in a personal album may result from changes in lighting conditions, from different camera settings, or from different cameras altogether, where such inconsistencies become even more apparent. Professional photographers may avoid these issues by controlling the lighting, shooting with carefully calibrated, manually set fixed camera settings, and using manual darkroom post processing[1]. However, these solutions require professional equipment, skill, and a significant amount of time for large photo albums.

In this paper, we present a new method for automatically ensuring color consistency in typical real-world personal photo albums, where photographs depict some common content but may differ in color, lighting conditions, viewpoint and non-rigid geometric transformation of objects. Our method may be used to induce consistent appearance in such albums without any user input. Should the user choose to adjust color or tone in selected photographs, our method automatically propagates these appearance changes to other photographs that share the same content. In the process, we attempt to balance between achieving color consistency and preserving the dynamic range and natural appearance of individual photos. A few snapshots from an interactive session with our method are shown

[1] For example, see discussions in the following photography forums: `http://tinyurl.com/{cnnwfo6, cqzuln5, c76787o}`

in Fig. 1. Enforcing consistency transforms the original images in the top row into those shown in the second row. Next, adjusting the leftmost photo and setting it as reference instantly propagates the red cast to the other photos in the set (third row). Photos that share more content with the reference photo are affected more strongly. Finally, adjusting the sixth photo propagates the green-cyan cast to the other photos (third row). The leftmost reference image remains constrained, and thus the second and third photos on the left remain almost unchanged, as they share much more content with that reference photo.

Our approach leverages recent developments in finding dense correspondences and transferring color between real-world photo pairs [HaCohen et al. 2011]. To extend this approach to photo albums, we construct a *match graph* in which the photos are represented by nodes and shared content is represented by edges. Each edge is assigned a weight based on the size of the corresponding regions between the two photos and the quality of the correspondence. Consistency of appearance between connected photos is achieved by minimizing a quadratic cost function over the entire graph. A number of regularization terms and constraints are introduced in order to balance between color consistency across photos, preserving the dynamic range of each photo, and limiting deviations from the original appearance. These make the system solvable even in the absence of user-provided constraints. The cost function may be efficiently re-optimized after every adjustment to any single photo in the album, thereby propagating the edit to the entire album and providing visual feedback at interactive rates.

2 Related Work

Automatic Enhancement

Many tools exist for automatic enhancement of photographs. For example, basic contrast enhancement may be achieved through histogram equalization, or by stretching the tonal range (e.g., the *Auto Levels* tool in Adobe Photoshop). Automatic white balance is commonly achieved by using some assumptions about the scene (e.g., Grey-World, White-Patch and Grey-Edge [van de Weijer et al. 2007]), or using a skin color model (e.g., in Apple's Aperture). Despite the simplicity of some of these tools, they are often quite effective and appear to be in use in several popular commercial software packages, alongside an array of other enhancement presets. However, these tools operate on each image independently and will not, in general, result in consistent appearance.

Appearance Propagation

Many researchers have studied the problem of color transfer between images (e.g., [Reinhard et al. 2001; Pitié et al. 2007; Kagarlitsky et al. 2009; An and Pellacini 2010; Oskam et al. 2012; HaCohen et al. 2011]). However, these methods consider a pair of images at a time, and do not have a natural extension for achieving consistent appearance across a set of images. In this work we are interested in propagating color from multiple source photos to multiple target photos (where any image may serve as either a source or as a target) and achieve a consistent appearance across the entire set.

Snavely *et al.* [2008] perform appearance stabilization in short photo sequences of geometrically aligned static scenes (e.g., monuments or building facades) by accumulating an affine color transformation matrix between successive photos. Farbman and Lischinski [2011] report that error accumulation prevents such an approach from scaling to long sequences of video frames and suggest a non-parametric approach for that task, where color differences are first computed using approximate correspondences and then interpolated in color space. Levin *et al.* [2004] use optical flow to propagate chromaticity information through a sequence of grayscale video frames from user-specified constraints (color scribbles). These methods are not applicable to general photo collections, which do not lend themselves to optical flow and may contain non-rigidly transforming content.

Dale *et al.* [2009] and Joshi *et al.* [2010] correct the colors of a target photo based on its similarities to multiple examples in unordered photo collections (online or personal). However, these methods do not scale well to more complex transformations such as general tone curves, and require co-segmentation with examples of similar scenes (Dale *et al.*) or facial images of the same person (Joshi *et al.*) captured from a similar view-point.

Kang *et al.* [2010] suggest learning personal user preferences for automatic photo enhancement from a pre-defined training set manually adjusted by the user. Subsequent works eliminated the need for each user to adjust a training set [Bychkovsky et al. 2011; Caicedo et al. 2011]. The method of Bychkovsky *et al.* [2011] learns on the fly how a specific user's editing style differs from those learned from a professional photo retoucher. While their approach supports adjusting an entire photo album at once and revises the adjustment as the user fine-tunes individual photos, they do not consider shared content between images, and make no attempt to ensure consistency of appearance as our method does. A user may want to edit subsets of an album with different content in different ways (as demonstrated in Fig. 1), which might not correspond to a unified learnable style. Also, their approach focuses on luminance adjustments and has not been applied to more general color manipulations.

Commercial packages such as Adobe Photoshop Lightroom and Apple's Aperture offer the option of copying edits made to one image and pasting (applying) them to any number of selected images. However, direct copying and pasting of edits does not take image content into account; thus, it cannot ensure consistency and often results in undesirable changes in appearance, as demonstrated in Figure 2.

Laffont *et al.* [2012] describe a method for computing intrinsic images from a photo collection. Decomposing images in this manner enables the application of a variety of powerful edits to the entire collection in a consistent manner. However, this method performs a 3D reconstruction of points and normals in the scene, and thus requires many images of the same static scene. Our approach can be viewed as orthogonal to theirs, as we aim to perform simpler appearance manipulations, but on a wider variety of photo collections.

Yücer *et al.* [2012] and Hasinoff *et al.* [2010] also address automatic transfer of edits from one image to other images of the same object or scene. However, their edits consist of painting or removing details in images, rather than global adjustments of appearance, and they do not consider indirect propagation of edits.

Non-Rigid Dense Correspondence (NRDC) [2011] is a state-of-the-art method for finding corresponding regions and color transfer between two photos that share common content. This method can handle shared content under non-rigid deformations and color variations, and computes a parametric color transfer model between pairs of images. In this paper we extend this method to transfer appearance across multiple photos while taking mutual interactions into account, thereby enforcing consistent appearance in an entire unordered collection with non-rigid shared content. As described earlier, our method may be used to propagate adjustments of one or many photos to the rest of the collection, including images that share no content with those being edited. The propagation respects connected components, i.e., changing an image affects those images that are more similar to it than others. To our knowledge, this is the first method to propagate appearance characteristics between multiple unordered photos of non-rigid content in this manner.

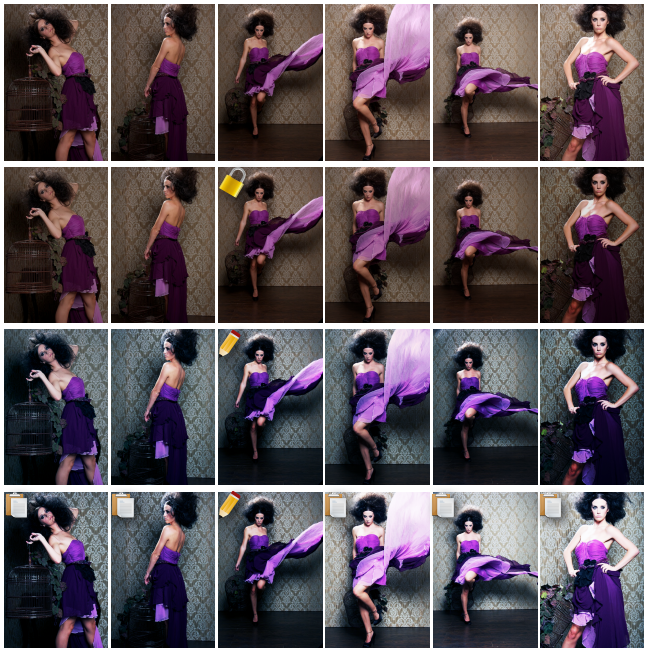

Figure 2: *Our method vs. naïve copy and paste of edits. 1st row: Professional input photos with subtle appearance differences. 2nd row: reducing appearance inconsistencies using the original appearance of the third photo as a reference. 3rd row: Propagating edits from the third photo to the rest of the photos using our method. 4th row: Naïvely copying the edits from the third photo to the rest (as done, e.g., by Adobe Lightroom's Sync Settings feature). Photography by Tawny Horton. Model: Jayme Jaynes.*

Matching Photos in a Collection

Match graphs of images have been used in many applications, such as 3D rendering, photo navigation, content-based retrieval, image clustering and label propagation. Commonly, these methods rely on a 3D reconstruction of a static scene based on sparse local feature matching and RANSAC followed by bundle adjustment for placing all photos in a common coordinate system and further outlier rejection [Snavely et al. 2006; Agarwal et al. 2009]. When the number of photos to be matched is large, computing an accurate match between all possible pairs is too expensive. Therefore, a lightweight *link prediction* mechanism is used to suggest candidate pairs that are likely to yield meaningful matches, significantly reducing the graph construction time at the expense of a partial graph edge coverage [Frahm et al. 2010; Kim et al. 2012].

However, these methods are designed for static content only (e.g., landmarks, buildings) and ignore non-rigid objects. In contrast, our method targets personal photo collections, where salient non-rigid objects – such as people – may dominate the subject matter and thus cannot be ignored. Adopting the techniques outlined earlier (sparse local feature matching followed by bundle adjustment) would only be able to match static backgrounds, and thus is inadequate for our needs. NRDC, on the other hand, is a unified mechanism for matching and outlier rejection. It yields a dense pixel-wise correspondence accompanied by a confidence map (for outlier rejection), and is capable of matching content under non-rigid geometric transformations and various appearance differences. Face-tagging can be easily employed to aid NRDC in matching faces of the same people, although we found that this step is not needed.

Our system also relies on a link prediction mechanism inspired by Agarwal *et al.* [2009]. However, it was necessary to adapt it to our setting for several reasons. First, no nodes are redundant, as we wish to adjust all images. Second, we target for a personal computer implementation, rather than hundreds of cores. Third, global visual descriptors are not sufficient for link prediction in our case as they do not capture non-rigid variations well. On the other hand, descriptors based on faces and timestamps are very effective for personal albums but are less relevant for general internet collections.

A few recent methods [Barnes 2011; Faktor and Irani 2012; Gould and Zhang 2012] extended the PatchMatch algorithm [Barnes et al. 2009] to find a nearest neighbor field of patch correspondences in a large collection of photos. While their theoretical complexity is close to linear in the number of photos, they are still very slow in practice, as no link prediction mechanism is employed. Moreover, as with PatchMatch, the resulting nearest neighbor field often does not provide a sufficiently accurate and reliable correspondence [Ha-Cohen et al. 2011]. That is why these methods are usually applied at a relatively low resolution for computer vision applications like label propagation [Gould and Zhang 2012] and unsupervised discovery of categories [Faktor and Irani 2012]. In this work we propose an effective link prediction mechanism for NRDC-based matching of images in personal photo albums.

3 Overview

Given a photo collection, our goal is to ensure consistent appearance (similar color and exposure) of shared content across multiple photos of the collection, and to maintain this consistency as the user performs interactive adjustments on selected images. To achieve this goal, as a pre-processing step, we construct a *match graph* $G = \{V, E\}$ whose vertices $V = \{I_i\}_{i=1}^{n}$ represent individual photos in the collection and whose edges E contain information regarding the correspondences between photo pairs. Using this match graph we minimize a quadratic cost function that penalizes color differences between matching areas.

The cost function and the optimization are described in more detail in Section 4. Optimizing over the entire match graph makes it possible to ensure consistent appearance even between pairs of photos that do not share any content directly. Thus, our framework supports *indirect* propagation, and edits made to a single image can, in principle, propagate to the entire collection.

Computing the full match graph is expensive, since each edge involves computing a dense correspondence between a pair of images. However, because we enable indirect propagation, it typically suffices to construct only a sparse subset of the full set of edges; namely, only edges connecting images with a substantial amount of shared content. To this end, we trained an SVM-based classifier for quickly predicting which pairs of images may have a significant correspondence, leading to a drastic reduction in the match graph construction time by reducing the number of accurate correspondence computations to be linear in the number of photos. Our classifier and the resulting link prediction strategy are described in more detail in Section 5.

4 Appearance Consistency Optimization

In order to achieve consistent appearance between multiple photos in a collection we attempt to strike a balance between three potentially contradictory goals: (i) ensuring that pixels depicting the same content have the same color across different images; (ii) avoiding unsightly visual artifacts, such as gradient reversals or severe loss of contrast; and (iii) attempting to preserve as much as possible the original dynamic range of each photo. This is a non-

trivial task: for example, aligning the appearance of two photos captured with different exposure settings might produce either clipped pixels in one photo or reduced dynamic range in the other.

Our approach is to seek a set of dedicated color transformations f_i (one for each image I_i), such that the resulting transformed images comply with our appearance consistency requirements. This is done by solving the following optimization problem:

$$\{\hat{f}_i\}_{i=1}^n = \underset{\{f_i\}_{i=1}^n}{\arg\min} \sum_{i\neq j} A(f_i, f_j) + \sum_{i=1}^n C_{soft}(f_i) \quad \text{subject to: } C_{hard}(f_i), \forall i \in \{1,...,n\} \tag{1}$$

Here $A(f_i, f_j)$ is a pairwise affinity term that penalizes color differences between shared content, while $C_{soft}(f_i)$ is a unary term that softly enforces certain constraints on the color transformations, in addition to hard constraints enforced on them by $C_{hard}(f_i)$. Below we describe our color transformation model and the different types of constraints, and then discuss the affinity term.

Color transformation model. To model the color transformations f_i, we use an expressive global parametric model, similar to the one used by HaCohen *et al.* [2011]. Specifically, each f_i consists of three curves (one per RGB channel). Each curve is a smooth piecewise-quadratic spline with 7 knots at (0, 0.2, 0.4, 0.6, 0.8, 1), which translates to 8 degrees of freedom per curve. This model is flexible enough to compensate for a variety of common appearance differences, such as gamma curves, S-curves, color temperature changes, and other common global operators. The shape of each curve is regularized via unary soft constraint terms:

$$\begin{aligned} C_{soft}(f_i) &= \lambda_1 \textstyle\sum_{x\in\{0,1\}} |f_i(x) - x|^2 \\ &+ \lambda_2 \textstyle\sum_{x\in\{0.2j-0.1\}_{j=1}^5} |f_i(x) - x|^2 \\ &+ \lambda_3 \textstyle\sum_{x\in\{0.2j-0.1\}_{j=1}^5} |f_i''(x)|^2. \end{aligned} \tag{2}$$

λ_1 and λ_2 control how much to pull the curve towards identity (no change) at the end points of the range (0 and 1) and at five midpoints between the knots (middle of each spline segment). λ_3 controls the smoothness of the curve by penalizing for large second derivatives. In our implementation we set $\lambda_1 = 50000$, $\lambda_2 = 170$ and $\lambda_3 = 0.08$ by default, but our GUI provides the user with two sliders: one to control preservation of dynamic range (via λ_1) and another to control preservation of original appearance (via λ_2). In addition, the curve is forced to be strictly monotonic (at the spline segment midpoints) and to cross the x axis right of the origin (as adding brightness is often not desired). This is done via the following hard constraints:

$$C_{hard}(f_i): \begin{array}{ll} \text{i.} & 0.2 \le f_i'(x) \le 5, \quad \forall x \in \{0.2j-0.1\}_{j=1}^5 \\ \text{ii.} & f_i(0) \le 0 \end{array} \tag{3}$$

Although we compute dense (pixel-wise) correspondences between images, we chose to use the global parametric color model described above over a local one for several reasons: First, a global model is more robust to errors in the correspondence. Second, a global model is easier to regularize, as we do using our constraints. Finally, a global parametric model is more efficient to optimize due to the small number of unknowns it involves. This is particularly important for providing interactive feedback when manipulating large photo collections.

Affinity term. One might consider defining the pairwise affinity term $A(f_i, f_j)$ in eq. (1) using the weighted SSD (sum of squared differences) between color-mapped pairs of matching pixels:

$$A(f_i, f_j) = \sum_{\mathbf{p}} w_{i,j}(\mathbf{p}) \left| f_i(I_i(\mathbf{p})) - f_j(I_j(M_{i,j}(\mathbf{p}))) \right|^2 \tag{4}$$

where $M_{i,j} : \mathbb{N}^2 \to \mathbb{R}^2$ is the partial pixel-wise mapping that maps pixels in I_i to I_j, and $w_{i,j} : \mathbb{N}^2 \to [0,1]$ is the confidence map associated with this mapping. Both of these functions are computed using the NRDC method [HaCohen et al. 2011].

Although this is a natural choice to make, this definition of pairwise affinities often yields poor results when multiple photos are involved. The source of the problem is that a photo in the collection might be "pulled" in different directions by different neighbors in the graph. If the correspondences associated with each of these graph edges do not cover the dynamic range in a uniform manner, a direct attempt to minimize differences between the colors of matching pixels might yield ill-behaved curves and visual artifacts.

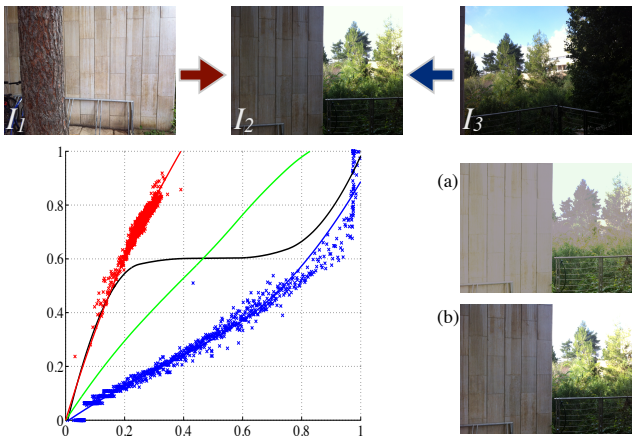

Figure 3: *Comparison between the affinity terms in Eq. (4) and Eq. (6). Here, I_1 and I_3 are fixed and the goal is to propagate their appearance to I_2 by fitting a monotonic curve (only luminance for simplicity) that takes into account the two opposite relationships. The red and the blue points are sampled from the correspondence between the pairs (I_2, I_1) and (I_2, I_3) respectively. The red and the blue curves were fitted to the red and the blue points respectively (Eq. 5). The black curve was fitted by all points (Eq. 4). The green curve was fitted by points sampled uniformly from the red and the blue curves (Eq. 6). (a) Result of applying the black curve on I_2. (b) Result of applying the green curve on I_2.*

This problem is demonstrated by the example in Figure 3: Let I_1, I_2, I_3 be photos such that I_2 has a dark region that corresponds to a brighter region in I_1, and has a bright region that corresponds to a darker region at I_3. Attempting to find a single simple curve that fits the transformations for all the individual matching pixel pairs in this case results in a severe loss of contrast in the midtones of I_2 (black curve and result (a) in Figure 3).

However, if we let the two images I_1 and I_3 "pull" with a more uniform weight across the entire dynamic range, this problem would be avoided in a more consistent way; namely, since I_2 is being pulled into two opposite directions, it should move in a consistent manner in the direction of the stronger pull (or remain nearly unchanged, when the pulls are of roughly equal strength). This outcome is represented by the green curve and result (b) in Figure 3, that shows how contrast is much better preserved.

The above example demonstrates that, in general, each edge of the graph should uniformly affect the dynamic range of the image nodes to which it connects. One way to achieve this is by up-weighting color samples from the correspondence between each pair, located at sparsely populated color regions. However, such re-weighting is not effective if the corresponding regions include no samples at all in some parts of the dynamic range. Instead, we fit a curve for each matching pair and sample it *uniformly*. Thus, we can robustly take into account all observed corresponding pixels including their relative weights, while hallucinating missing parts of the curves according to our regularization terms.

Formally, this leads to a two-step approach. For each matched pair of images I_i, I_j, we first fit the transformation $g_{i,j}$ that optimally matches the colors in I_i to the corresponding pixels in I_j:

$$g_{i,j} = \arg\min_{f_i} \sum_{\mathbf{p}} w_{i,j}(\mathbf{p}) \left(f_i(I_i(\mathbf{p})) - I_j(M_{i,j}(\mathbf{p})) \right)^2 \qquad (5)$$

We then formulate the pairwise affinity term using $L+1$ uniformly spaced samples across the dynamic range (for each color channel):

$$A(f_i, f_j) = w'_{i,j} \sum_{l=0}^{L} \left(f_i(l/L) - f_j\left(g_{i,j}(l/L)\right) \right)^2 \qquad (6)$$

where $w'_{i,j}$ is a scalar weighting the contribution of the connection between I_i and I_j. We define $w'_{i,j}$ as the average of the confidence values (between 0 and 1) over the two images: $w'_{i,j} = \frac{1}{|\{\mathbf{p}\}|} \sum_{\mathbf{p}} w_{i,j}(\mathbf{p})$. We found that the optimization in (1) results in much better-behaved solutions using this definition of the pairwise affinities (see Fig. 3).

4.1 Propagating User Edits

The method described so far does not require any user input. Given a collection of photographs with shared content it simply applies a set of color transformations – one for each image – which result in consistent color appearance across the entire collection. However, the method readily supports the introduction of user specified constraints. For example, the user may indicate that she prefers one or more of the images in the collection unaltered (also called here "reference" images). In this case, we simply constrain the corresponding transformations f_i to identity. As a result only the other images will be transformed so as to achieve a consistent appearance with the reference images. In case of conflicting constraints, unconstrained images that are more similar to one of the references will be adjusted more towards the appearance of that reference, whereas images with roughly equal affinity will remain unchanged.

In some cases, after the images have been made consistent, the user might want to further adjust one or more of the images, e.g., modify the exposure or the color temperature or manipulate the tone curves directly. The resulting curves of the edited images are then constrained, and the subsequent optimization propagates the user's edits to the other images in the collection.

Fig. 4 demonstrates the effect of choosing different photos as reference images.

5 Accelerating Match Graph Construction

Computing the complete match graph for a large photo collection can be time-consuming. There is a quadratic number of pairs to consider, and for each pair we need to compute a dense correspondence using NRDC [HaCohen et al. 2011], which takes a few seconds per pair. However, we find in practice that most pairs of images share no content, even in collections from a single camera over a short time span. These pairs either do not exist as edges in the full graph, or exist only as edges with low weight. Furthermore, the approach described in Section 4 functions well even with only a sparse subset of the edges in the graph. Ideally, this subset should contain the strongest connections in the collection, and retain the connected components present in the full graph.

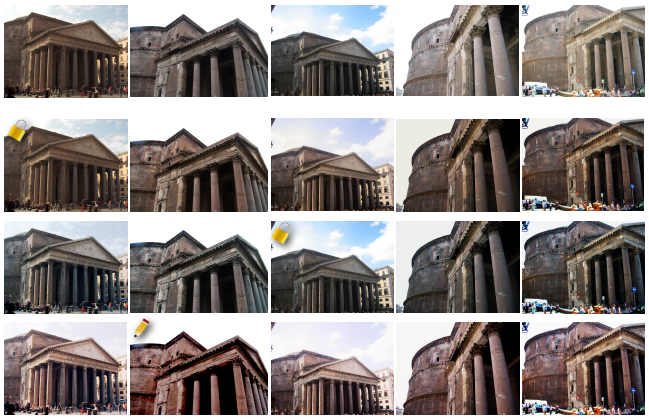

Figure 4: *The effect of using different photos as references. First row: input photos retrieved from an internet collection with significant appearance variability due to different cameras and different lighting conditions. Second-fourth rows: in each row a different photo is set as a reference.*

In this section we describe a new *link prediction* mechanism, inspired by Agarwal *et al.* [2009], but adapted to work effectively for typical personal photo collections. Our link prediction scheme eschews the full dense correspondence calculation on the majority of image pairs, thereby greatly accelerating match graph construction. Given a pair of photos I_i and I_j the link predictor can quickly estimate the likelihood that applying NRDC between these two images would result in a significant connection in the graph.

Our approach consists of training a support vector machine (SVM) to classify pairs of images into one of two categories: pairs that are likely to have a significant correspondence between them, and pairs that are unlikely to have such a correspondence. To create a training set for this SVM, we performed a full NRDC computation between all pairs in a photo collection with 69 images, labeling every pair with more than 2% of corresponding pixels as "matchable" and the rest as "non-matchable". We found that 22% of the pairs in this collection are matchable; this is roughly the average percentage of matchable pairs in a set of six personal albums we collected to evaluate our method (see Sec. 6). We start by extracting several feature vectors for every image in the collection. We experimented with both linear and kernel SVM and found the latter to perform better for our task. Therefore, for every pair of images I_i and I_j we apply a set of kernels on different types of feature vectors. The result is a continuous classification measure of the "matchability" between the two images (rather than a discrete classification), obtained by the dot product of the SVM weights and the vector of the kernels $K(I_i, I_j)$.

Based on this matchability classifier, we propose a link prediction strategy that iteratively chooses which image pairs to match to construct our image graph. Roughly speaking, our strategy attempts to balance between overall graph connectivity and densifying subgraphs within a bounded budget of NRDC matching attempts. Below we describe our matchability classifier and link prediction strategy in more detail.

5.1 Pairwise Matchability Classifier

Formally, given a collection of images, we first extract a set of descriptors $D(I_i) = \{d_1(I_i), ..., d_m(I_i)\}$ from each image I_i. For each descriptor d_i we define a similarity function $k_i(I_i, I_j)$ that measures the similarity between the pair (I_i, I_j) with respect to that descriptor. The feature vector used to train the SVM is the concatenation of these similarities:

$$K(I_i, I_j) = \big(k_1(I_i, I_j), \ldots, k_m(I_i, I_j)\big)$$

The final matchability classification score is then given by the inner product of the SVM's learned weights $\mathbf{w}$ and this feature vector:

$$\mathscr{M}(I_i, I_j) = \mathbf{w}^T K(I_i, I_j).$$

Many image properties and descriptors have been proposed and used to assess similarity between images. After extensive experimentation on typical personal albums, we chose a set of descriptors described in the next paragraphs.

Bag of visual words [Sivic and Zisserman 2003]: A histogram of densely sampled SIFT descriptors [Lowe 2004], quantized by a codebook of size 600. The codebook is obtained using K-means on a random subset of 10^5 SIFT descriptors sampled from the collection itself[2]. We also tried to concatenate histograms from different regions of the image, but found that due to the high geometrical variance often present in personal photo albums, a single histogram for the entire image performed better. We define the similarity between two bag-of-words descriptors $d_{BoW}(I_i)$ and $d_{BoW}(I_j)$ as $k_{BoW}(I_i, I_j) = 2\sum_h \frac{d_{BoW}(I_i)_h \cdot d_{BoW}(I_j)_h}{d_{BoW}(I_i)_h + d_{BoW}(I_j)_h}$ where h denotes a histogram bin index. Note that the codebooks used during learning and testing are different, but this does not directly affect the input to the SVM, which uses the distance between histograms. We found a collection specific codebook to work better than a generic one. Note that we do not compute direct SIFT matches at any step, since HaCohen *et al.* [2011] has demonstrated that SIFT matches are not effective for the task of color transfer between pairs of images with non-rigid transformations.

Tiny Image: An RGB, 8×8 pixel thumbnail of the image. We define the similarity between two "tiny image" descriptors $d_{tiny}(I_i)$ and $d_{tiny}(I_j)$ as $k_{tiny}(I_i, I_j) = 1 - \frac{\|d_{tiny}(I_i) - d_{tiny}(I_j)\|_1}{3 \times 8 \times 8}$.

Time stamp: We extract the capture timestamps $d_{time}(I_i)$ and $d_{time}(I_j)$ from the photo meta-data, and define their similarity as $k_{time}(I_i, I_j) = \exp\big(-\big\|d_{time}(I_i) - d_{time}(I_j)\big\|\big)$. If this information is not available for both photos, we define $k_{time} = 0$.

Faces: If faces in the collection were detected and recognized, (either manually or by a face recognition method in commercial photo management software) they can be used to infer matchability. Face similarity between two photos I_i and I_j is defined as the number of matching faces that appear in both photos: $k_{faces}(I_i, I_j) = \sum_h^{|\mathbf{faces}|} d_{faces}(I_i)_h \cdot d_{faces}(I_j)_h$ where $d_{faces}(I)_h$ is one if the person h appears in I and zero otherwise.

We compute $\mathscr{M}(I_i, I_j)$ once for each pair of images in the collection. Although the number of pairs is quadratic in the size of the collection, these computations are lightweight and in practice take only a few seconds for a collection with one thousand photos (negligible relative to the dense correspondence step). In extremely large albums, for which a quadratic number of matchability score computations would not be efficient enough, these computations may be limited to each image's first few dozens or hundreds of nearest neighbors in time, geographic location, or any available per-image information that is positively correlated with our classifier.

[2] Using the VLFeat (`www.vlfeat.org`) toolbox.

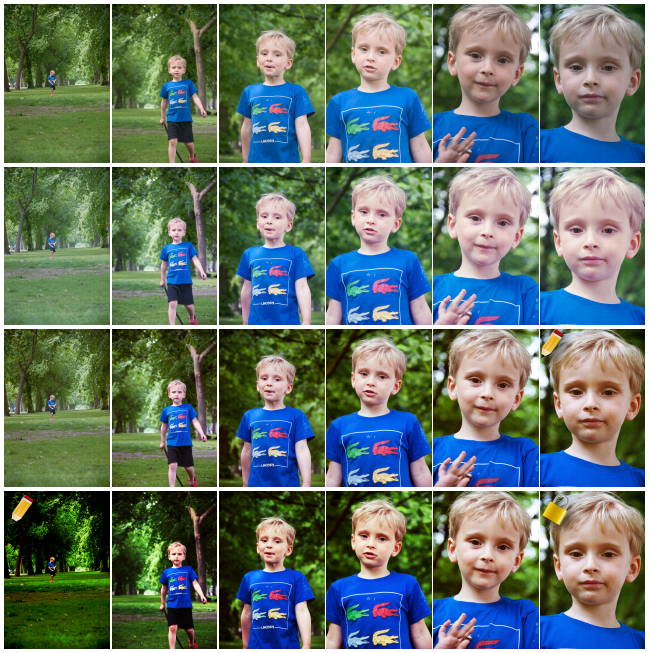

Figure 8: *Propagating appearance from multiple photos using our method. First row: Input photos with color and exposure differences. Second row: Automatic consistency using our method. Third row: Propagating appearance from the rightmost photo. Fourth row: Propagating appearance from two photos at once.*

5.2 Link Prediction Strategy

The matchability classifier yields an estimated graph G^*, where the weight of an edge between I_i and I_j is set to $\mathscr{M}(I_i, I_j)$. We then construct the actual match graph G by computing the full dense correspondence between each node I_i and a bounded number of neighbors in decreasing order of weight, in the following three phases. First, consider the first N_1 nearest neighbors $\{I_j\}_{j=1}^{N_1}$ in G^* such that $\mathscr{M}(I_i, I_j) > 0$. Second, consider the next N_2 neighbors of each node without restriction. Third, consider the next N_3 that are not already in the same connected component. As in Agarwal *et al.* [2009], we are interested in the fewest possible number of connected components, so that we can propagate colors from a single image to as many other images in the collection as possible.

We use $N_1 = 15$, $N_2 = 3$ and $N_3 = 3$ in all of our experiments. Note that although the estimated graph G^* may contain a quadratic number of edges, the strategy described above only makes a linear number of full NRDC matching attempts. As in the SVM training step, we discard edges in which fewer than 2% of pixels are found to correspond.

We also tried to densify the graph using "query expansion" [Agarwal et al. 2009], as well as the method of [Kim et al. 2012], but we found that due to the ability of our consistency optimization method to take indirect links between photos into account, this step was not necessary, and therefore we did not include it in our link prediction strategy. However, it may be useful for other applications that benefit from denser graphs.

Figure 5: *Comparison with automatic white-balance and tone of commercial tools. First row: Input photos with appearance differences. Second row: The result of Apple Aperture's automatic tone and white-balance which relies on either skin-tone if a face is automatically detected, (columns 1, 2, 6, 7) or else a natural gray assumption. Third row: The result of Adobe Lightroom's auto white-balance function. Fourth row: Our automatic color consistency optimization result.*

Figure 6: *Harmonized wedding photos with our method. First row: Input photos with different color casts due to different lighting conditions. Second row: Automatic color consistency optimization.*

Figure 7: *Another example in which input photos from an internet collection (first row) have been made consistent using our method (second row) by constraining the second image as a reference.*

6 Results and Evaluation

Figures 1, 2, 4, 5, 6, 7 and 8 show results of our method on several different types of albums and image editing scenarios. Figure 2 shows the usefulness of our method even for professional photography. Despite the near-optimal conditions in the studio, our system can still improve color consistency and accelerate photographers' photo editing workflow. Figure 8 shows the effects of multiple constraints distributed across several connected photos. In this case, each edit affects the remaining photos in proportion to the amount of content they share. Figures 4 and 7 demonstrate the ability of our method to ensure color consistency even in internet collections of landmarks captured by different photographers, with different cameras, in different seasons, and under various lighting conditions. Although we do not expect people to have interest in enforcing consistency over such collections, these experiments demonstrate the operating range of our method.

We compared our appearance consistency optimization method to the auto white-balance and auto tone functions of Apple Aperture and Adobe Photoshop Lightroom. According to the product documentation, Aperture's auto white-balance algorithm utilizes automatically detected faces to perform white-balance based on skin color and uses "natural gray" when no faces have been found. The

second row in Fig. 5 shows that despite the appearance of the same people in a photo collection the method based on skin color did not perform a consistent white balance on this set. (Aperture reports that the skin-based algorithm was used for the photos in columns 1,2, 6 and 7). Adobe Lightroom also produced an inconsistent appearance for this set (third row in the same figure). In contrast, our method automatically adjusts these photos in a consistent way (last row). Note that although our method does not specifically target white-balance, enforcing consistency eliminated the warm cast in the second column because most of the photographs in the set were already white-balanced. Also note that our system made the colors of the image at the second column washed-out to make them consistent with the input image in the first column. In such cases, when the result is consistent but not pleasing, the user may edit any of the images and constrain it to propagate its colors to the rest of the album, as demonstrated in other figures. For the sake of this comparison we show only the fully automatic result of our system.

In addition, we compared our propagation approach to the "Sync Settings" feature of Adobe Lightroom, which copies edit operations from one photo and applies them to other photos. The last row in Fig. 2 shows the result of using this feature to copy edits from the third photo to the other photos. The differences between the input photos lead to different results when applying the same edits blindly. In contrast, our method propagates the edits consistently, as seen in the third row. Once a photo is constrained by the user, its appearance is propagated to similar photos even before any edits are made (third photo in second row), since our method uses only the current pixel colors and does not require editing.

We have developed a demo application that allows interactive photo collection enhancement using our method. We refer the reader to the supplementary video, which illustrates several albums edited using our system.

6.1 Running Times and Implementation Details

Interactive editing

Applying a piece-wise quadratic spline curve on each channel of image I can be expressed as matrix-vector product $F\mathbf{x}$ where $\mathbf{x}$ are the polynomial coefficients, and F is a matrix that depends on the image I and the spline basis functions. Therefore, our objective in Eq. 1 is quadratic with linear inequality constraints and can be efficiently solved using Quadratic Programming. We used a solver [3] based on the Goldfarb-Idnani active-set dual method. Our implementation caches the building blocks of the matrices required for this representation (a small constant memory per edge in the match graph). After applying an edit on an image, only the relevant building blocks should be updated, so the system can be re-optimized efficiently. We use the GPU to apply the resulting color transformations on photos as the images are rendered. Thus, when editing an album, only the color transformation parameters need to be resent to the GPU rather than the entire image library.

In our experiments we found that the time it takes to update the matrices and recompute the color transformations typically varies between 0.23 and 0.77 seconds (all times were measured on a MacBook Pro with 2.3GHz Intel Core i7 (2820qm) CPU and 8GB of RAM). The building blocks for the initial solution are computed as part of our pre-processing phase that we describe below.

Pre-processing

We compute the match graph on smaller versions of the photos with maximum dimension of 640 pixels. Our modified implementation of NRDC takes on average 5.37 seconds for a matchable image pair, and 1.32 seconds for a non-matchable pair by terminating NRDC if no significant correspondence is found after three coarse-scale iterations.

[3] QuadProg++ by Luca Di Gaspero: `quadprog.sourceforge.net`

Album Name	Wedding	Ireland	Regents
Number of photos	865	127	61
Matchable pairs	7.7% (28,623)	5.1% (410)	26% (473)
Pairs verified	7871	851	553
Pairs found	3482	240	305
Times for Individual Components			
Codebook creation	63s	55s	75s
BoW descriptors	762s	157s	56s
Tiny image descriptors	6.11s	0.97s	0.48s
Matchability scores	7.77s	0.03s	0.01s
Verifying candidates	6:45:35	0:35:20	0:32:31
Comparison			
Total (Our method)	6:47:52	0:38:46	0:34:43
All Pairs, Modified NRDC	~7 days	3:27:03	1:12:46
All Pairs, Original NRDC[4]	~24 days	12:12:09	2:45:25

Table 1: *Running times of individual components of our match graph construction and comparison to the alternatives of attempting to match all pairs with our modified version of NRDC that rejects non-matchable pairs faster, and with the original NRDC algorithm. See section 6.1 for more details.*

We measured match graph construction times for three albums using three methods. First, our method as described in Section 5. Second, a naïve computation of all pairs, but using our modified NRDC version that rejects non-matchable pairs faster. Third, a naïve computation of all pairs with the original NRDC algorithm[4]. We define "matchable pairs" as pairs for which our modified NRDC version found more than 2% of corresponding pixels. Table 1 lists the running times of each component of our match graph construction algorithm, and compares its total running time to these two other alternatives. The table shows that our method is significantly faster, due both to the use of link prediction mechanism and the faster rejection of non-matchable pairs.

6.2 Matchability Classification Evaluation

We evaluate the contribution of each of the descriptor types we use as part of our matchability classifier. We isolate the contribution of each descriptor by measuring its *marginal* contribution to the SVM performance. We measure the performance of our matchability classifier using its recall ($\frac{|\{\text{classified as matchable}\}\cap\{\text{matchables}\}|}{|\{\text{matchables}\}|}$) and precision ($\frac{|\{\text{classified as matchable}\}\cap\{\text{matchables}\}|}{|\{\text{classified as matchable}\}|}$), on six test albums with threshold 0 (this is the SVM's threshold that we use in the first phase of the link prediction algorithm). These albums contain different types of content (nature, city event, park, indoor events, and an internet collection of a landmark), and varying percentages of matchable pairs. We repeat this test with four other classifiers, each trained in the same way as our SVM based matchability classifier, but without one of the descriptors we used. Table 2 lists the average recall and precision on these test albums, and shows that all of the descriptors we use are meaningful and complement each other.

[4] Times for this method were estimated based on smaller subsets.

Descriptors	Average Recall	Average Precision
All	66%	87%
All except time	66%	67%
All except tiny	46%	83%
All except faces	59%	85%
All except BoW	32%	87%

Table 2: *Measuring the marginal contribution of each of the descriptors we use. See section 6.2 for more details.*

7 Summary, Limitations and Future Work

We have presented a novel and practical method for automatic optimization of color consistency, which enables interactive editing of photo collections. Our method currently accounts for global color appearance variations only (although some local variations may be approximated by global curves as in the case of "highlights and shadows"). In the future, we plan to extend the method to local edits as well. One possible solution might be to build a graph where the nodes are local regions, as opposed to entire images, and edits propagate within images in an edge-aware manner. Another limitation of our approach is that our color transfer model is based on RGB curves and does not model saturation changes well. HaCohen *et al.* [2011] showed that saturation can be modeled by an additional cross-channel linear operator after estimating the curves. Applying a similar solution to our problem will probably require using a more elaborate solver as the objective function will no longer be quadratic. Finally, our method works best when the collection contains a substantial amount of shared content, as typically occurs in personal and professional collections (especially before triage). Photos that share little (or no) content with others remain close to their original appearance because the relative weight of the identity term gets higher. This is a desired outcome if the original appearance of these photos is reasonable, but in other cases it might be better to start by applying an automatic enhancement method (like [Bychkovsky et al. 2011]), that could provide a default behavior for such weakly connected photos.

Acknowledgements: We thank Tawny Horton for granting us permission to use her photographs of the model Jayme Jaynes in Figure 2. This work was supported in part by the Israel Science Foundation founded by the Israel Academy of Sciences and Humanities.

References

AGARWAL, S., SNAVELY, N., SIMON, I., SEITZ, S. M., AND SZELISKI, R. 2009. Building Rome in a day. In *Proc. IEEE ICCV*.

AN, X., AND PELLACINI, F. 2010. User-controllable color transfer. *Computer Graphics Forum 29*, 2, 263–271.

BARNES, C., SHECHTMAN, E., FINKELSTEIN, A., AND GOLDMAN, D. B. 2009. PatchMatch: a randomized correspondence algorithm for structural image editing. *ACM Trans. Graph. 28*, 3.

BARNES, C. 2011. *PatchMatch: A Fast Randomized Matching Algorithm with Application to Image and Video*. PhD thesis, Princeton University.

BYCHKOVSKY, V., PARIS, S., CHAN, E., AND DURAND, F. 2011. Learning photographic global tonal adjustment with a database of input / output image pairs. In *Proc. IEEE CVPR*.

CAICEDO, J. C., KAPOOR, A., AND KANG, S. B. 2011. Collaborative personalization of image enhancement. In *Proc. IEEE CVPR*.

DALE, K., JOHNSON, M. K., SUNKAVALLI, K., MATUSIK, W., AND PFISTER, H. 2009. Image restoration using online photo collections. In *Proc. IEEE ICCV*.

FAKTOR, A., AND IRANI, M. 2012. "Clustering by Composition" - unsupervised discovery of image categories. In *Proc. ECCV (7)*, 474–487.

FARBMAN, Z., AND LISCHINSKI, D. 2011. Tonal stabilization of video. *ACM Trans. Graph. 30*, 4, 89:1–89:9.

FRAHM, J.-M., GEORGEL, P. F., GALLUP, D., JOHNSON, T., RAGURAM, R., WU, C., JEN, Y.-H., DUNN, E., CLIPP, B., AND LAZEBNIK, S. 2010. Building Rome on a cloudless day. In *Proc. ECCV (4)*, vol. 6314, 368–381.

GOULD, S., AND ZHANG, Y. 2012. PATCHMATCHGRAPH: building a graph of dense patch correspondences for label transfer. In *Proc. ECCV*, vol. Part V, 439–452.

HACOHEN, Y., SHECHTMAN, E., GOLDMAN, D. B., AND LISCHINSKI, D. 2011. Non-rigid dense correspondence with applications for image enhancement. *ACM Trans. Graph. 30*, 4, 70:1–70:9.

HASINOFF, S. W., JÓŹWIAK, M., DURAND, F., , AND FREEMAN, W. T. 2010. Search-and-replace editing for personal photo collections. In *Proc. ICCP*.

JOSHI, N., MATUSIK, W., ADELSON, E. H., AND KRIEGMAN, D. J. 2010. Personal photo enhancement using example images. *ACM Trans. Graph. 29*, 2 (April), 12:1–12:15.

KAGARLITSKY, S., MOSES, Y., AND HEL OR, Y. 2009. Piecewise-consistent color mappings of images acquired under various conditions. In *Proc. ICCV*, 2311–2318.

KANG, S. B., KAPOOR, A., AND LISCHINSKI, D. 2010. Personalization of image enhancement. In *Proc. IEEE CVPR*.

KIM, K. I., TOMPKIN, J., THEOBALD, M., KAUTZ, J., AND THEOBALT, C. 2012. Match graph construction for large image databases. *In Proc. ECCV*.

LAFFONT, P.-Y., BOUSSEAU, A., PARIS, S., DURAND, F., AND DRETTAKIS, G. 2012. Coherent intrinsic images from photo collections. *ACM Trans. Graph. 31*, 6, 202:1–11.

LEVIN, A., LISCHINSKI, D., AND WEISS, Y. 2004. Colorization using optimization. *ACM Trans. Graph. 23*, 3, 689–694.

LOWE, D. G. 2004. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision 60*, 2, 91–110.

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

PITIÉ, F., KOKARAM, A. C., AND DAHYOT, R. 2007. Automated colour grading using colour distribution transfer. *Comput. Vis. Image Underst. 107* (July), 123–137.

REINHARD, E., ASHIKHMIN, M., GOOCH, B., AND SHIRLEY, P. 2001. Color transfer between images. *IEEE Comput. Graph. Appl.* (September).

SIVIC, J., AND ZISSERMAN, A. 2003. Video Google: A text retrieval approach to object matching in videos. In *Proc. IEEE ICCV*, 1470.

SNAVELY, N., SEITZ, S. M., AND SZELISKI, R. 2006. Photo tourism: exploring photo collections in 3D. *ACM Trans. Graph. 25* (July), 835–846.

SNAVELY, N., GARG, R., SEITZ, S. M., AND SZELISKI, R. 2008. Finding paths through the world's photos. *ACM Trans. Graph. 27*, 3, 11–21.

VAN DE WEIJER, J., GEVERS, T., AND GIJSENIJ, A. 2007. Edge-based color constancy. *IEEE Trans. Im. Proc. 16*, 9, 2207–2214.

YÜCER, K., JACOBSON, A., HORNUNG, A., AND SORKINE, O. 2012. Transfusive image manipulation. *ACM Trans. Graph. 31*, 6, 176:1–176:9.