Unifying Color and Texture Transfer for Predictive Appearance Manipulation

Fumio Okura [1,2,3] Kenneth Vanhoey [2] Adrien Bousseau [2] Alexei A. Efros [4] George Drettakis [2]

[1] Nara Institute of Science and Technology [2] Inria [3] Osaka University [4] University of California, Berkeley

Figure 1: Given *source*, *exemplar* and *target* images (left box), local color transfer [SPDF13, LRT*14] does not change the texture on the tree in the target; while texture transfer [HJO*01, EF01] destroys target structure such as the poles in the field and the bushes at the horizon. We unify the two approaches by predicting where color transfer is sufficient and where texture transfer is needed, which allows our algorithm to synthesize leaves on the tree while preserving other details of the target.

Abstract

Recent color transfer methods use local information to learn the transformation from a source *to an* exemplar *image, and then transfer this appearance change to a* target *image. These solutions achieve very successful results for general mood changes, e.g., changing the appearance of an image from "sunny" to "overcast". However, such methods have a hard time creating new image content, such as leaves on a bare tree. Texture transfer, on the other hand, can synthesize such content but tends to destroy image structure. We propose the first algorithm that unifies color and texture transfer, outperforming both by leveraging their respective strengths. A key novelty in our approach resides in teasing apart appearance changes that can be modeled simply as changes in color versus those that require new image content to be generated. Our method starts with an analysis phase which evaluates the success of color transfer by comparing the* exemplar *with the* source*. This analysis then drives a selective, iterative texture transfer algorithm that simultaneously predicts the success of color transfer on the* target *and synthesizes new content where needed. We demonstrate our unified algorithm by transferring large temporal changes between photographs, such as change of season – e.g., leaves on bare trees or piles of snow on a street – and flooding.*

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—I.3.7 [Computer Graphics]: Color, shading, shadowing, and texture—

1. Introduction

Color and texture transfer techniques allow for drastic modifications of image content by learning the transformation from a *source* image to an *exemplar* (Figure 1(a,b)) and applying the same transformation onto a *target* image (Figure 1(c,d,e)). Color transfer is particularly effective in capturing the overall mood of an *exemplar* image, such as photographic style [BPD06] and time and weather in time-lapse sequences [SPDF13, LRT*14]. However, color transfer alone cannot capture changes of shape and texture. As an example, while color transfer successfully captures the appearance of the sky and ground in Figure 1(c,d), it fails to create leaves on the tree. In contrast, texture transfer is capable of hallucinating details in the *target* by copying

image patches from the *exemplar*. However, because texture synthesis replaces the image content, it often destroys the image structure and existing details, as is the case on the ground in Figure 1(e). As a result, while texture transfer has been used with success in non-photorealistic rendering [HJO*01, BCK*13], it should be used with parsimony to preserve realism of the *target*.

We propose the first algorithm that unifies color and texture transfer and leverages their respective strengths. Our work is inspired by the observation that recent patch-based color transfer [SPDF13] and texture transfer algorithms [HJO*01, BCK*13] share many common ingredients. In particular, both methods rely on dense patch matching between *source* and *target* images to locally transfer information from the *source*/*exemplar* pair to the *target*. Local color transfer methods are restricted to a transformation of the color distribution of the patch, while texture transfer approaches copy pixel values to overwrite the existing content. Based on this observation, our algorithm *predicts* if local color transfer is sufficient to capture the appearance of the *exemplar*. If yes, we apply the color transformation since it is less invasive than pixel copy. Otherwise, we perform texture synthesis by replacing the *target* pixels with ones from the *exemplar*.

We demonstrate our unified algorithm by transferring large temporal changes between photographs, such as change of season and flooding. Figure 1(e) illustrates our transfer from winter to autumn. This application scenario raises many technical challenges, some of which are beyond the scope of this work. In particular, we focus on synthesizing plausible images given suitable input and we manually select *source*/*exemplar*/*target* triplets of images to illustrate our contribution. In addition, while results such as Figures 1(f) and 8 were generated automatically, we took advantage of user-guided co-segmentation to improve the matching on some challenging *source*/*target* pairs (Figure 9). We believe that the rapid progress in the fields of image retrieval [SMGE11] and image matching [HXM*13] will soon provide automatic solutions to these related problems.

In summary, we make the following contributions:

- We introduce the idea of selecting between complementary appearance manipulation techniques, i.e., color or texture transfer, by *predicting* their chance of success.
- We describe a synthesis algorithm that *selectively* applies color or texture transfer to capture the appearance of the *exemplar* without compromising on the *target* structure.

The results show that our method allows drastic manipulation of scene appearance, to include phenomena such as flooding, snow storms or summer leaves on bare trees.

2. Related Work

Our work builds on the complementary fields of color and texture transfer. We refer readers to [FPC*14] and [WLKT09] for surveys of these two fields.

Color transfer. Pioneering color transfer methods transform the global color distribution of a *target* image to agree with the color distribution of an *exemplar* [RAGS01, PKD07]. While these methods perform well on pairs of images with regions of similar scale and color, they can produce unpredictable results when the *exemplar* and *target* images are too different. For this reason, subsequent methods use segmentation [TJT05, ICOL05] and user indications [WAM02, AP10] to localize the color transfer to similar regions of the two images. Recent methods achieve even finer locality by relying on over-segmentation [LRT*14] or patch matching [SPDF13]. Shih et al. [SPDF13] and Laffont et al. [LRT*14] demonstrate impressive time and season transfer from time-lapse videos. These videos are typically captured from a distance and most often do not contain the changes of content that we target, such as the appearance of leaves on the branches of a tree. We found that state-of-the art color transfer algorithms are not sufficient to handle such changes (Figure 1(d)).

Texture transfer. Texture transfer [EF01], also known as Image Analogies [HJO*01], takes as input a pair of aligned *source* and *exemplar* images, along with a *target* image that has the same appearance as the *source*. The algorithm generates a dense matching between the *source* and *target* to guide the synthesis of the *exemplar* texture over the *target*. Texture transfer has been used for a variety of applications, including super-resolution [FJP02] and non-photorealistic rendering [BCK*13]. Closer to our goal is the work of Bonneel et al. [BVDPLD10] and CG2Real [JDA*11]. The former uses texture transfer to enhance a coarse 3D model, while the latter applies color and texture transfer in sequence to hallucinate realistic details over poor synthetic renderings. In contrast, our method locally decides if color or texture transfer should be applied by predicting which will best preserve the existing structure of real images. Concurrently to our work, Diamanti et al. [DBP*15] allow users to annotate multiple exemplar images to form appearance clusters that are interpolated during texture synthesis. Similar ideas could be applied in our context to improve synthesis quality.

Prediction of synthesis quality. Our approach has similar motivation to the work of Swamy et al. [SCBH11] who use human judgments to train a predictive model of texture synthesis quality based on various image statistics. The model by Kopf et al. [KKDK12] also relies on human training to predict the success of image completion from statistics of the *exemplar* regions used to fill holes. Our method is complementary to the latter since we predict the quality of *color transfer* for scenarios where the goal is to modify existing content rather than filling in missing regions.

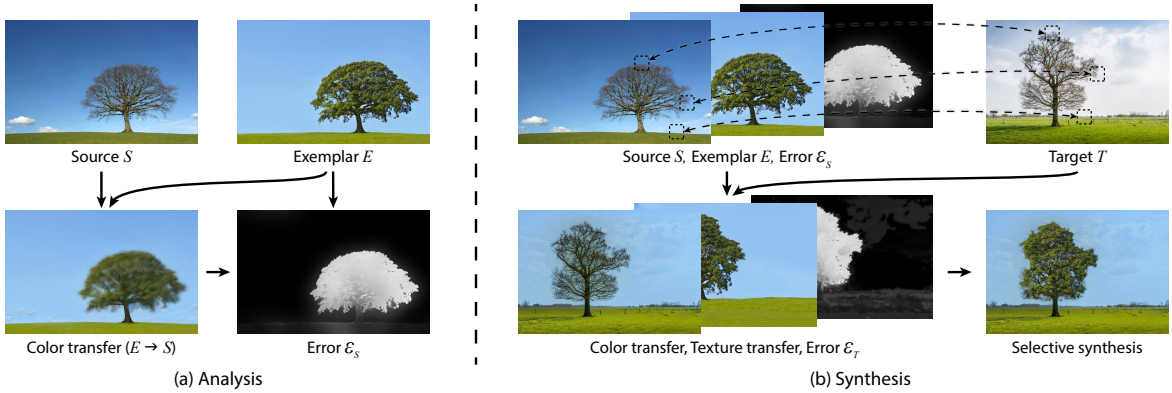


Figure 2: Our method is composed of two main phases. The analysis phase (a) estimates the areas of the *source*/*exemplar* pair that are not well reproduced by color transfer alone. The synthesis phase (b) transfers the estimated error onto the *target* and uses this prediction to selectively choose between color transfer and texture transfer.

3. Background on Color and Texture Transfer

We now introduce necessary background on the color and texture transfer algorithms we build upon. In our context, similarly to Image Analogies [HJO*01], the input is a triplet of images, which we name *source* S, *exemplar* E and *target* T. The *source*/*exemplar* pair corresponds to images of the same scene captured at different times. These images should be taken from the same viewpoint, even though alignment is often approximate due to slight changes of camera pose and scene geometry over time. The *target* image should be visually similar to the *source*, the two images being related by per-pixel matches [BSFG09]. The desired *output* O is an image that depicts the same scene as the *target*, but with the visual appearance of the *exemplar*.

For color transfer, our approach builds on the method of Shih et al. [SPDF13], which computes dense matching between square patches of the *source* and *target* and transfers color using a per-patch affine color transformation. Formally, denoting $T(p)$ a patch from image T centered at pixel p, $f_p(T(p))$ its affine color transformation, and $S(q)$ its matching patch in image S, the algorithm minimizes

$$\underset{f_p,O}{\operatorname{argmin}} \sum_p \|E(q) - f_p(S(q))\|^2 + \frac{1}{\lambda} \sum_p \|O(p) - f_p(T(p))\|^2 \quad (1)$$

where the first term captures the transformation that best relates the *source* to the *exemplar*, while the second term captures the *output* that best corresponds to the *target* when the same transformation is applied (see [SPDF13] for additional details and regularization term). The weight λ controls the influence of the two terms, a high value favors greater transfer of the *exemplar* colors while a low value better preserves the *target*.

For texture transfer, our algorithm builds on the texture synthesis method of Wexler et al. [WSI07] and its variants [SCSI08, DSB*12]. This method performs texture synthesis in an iterative coarse-to-fine fashion. At each iteration, the algorithm first computes dense matches between square patches of the *source*/*exemplar* and *target*/*output* pairs. Since the patches of neighboring pixels overlap, each *output* pixel is assigned multiple *exemplar* patches. These multiple contributions are averaged to update the color of each *output* pixel, a process called *voting*. By iterating the matching and voting steps, the algorithm solves for the matches q and *output* image O that minimize the patch energy

$$\underset{q,O}{\operatorname{argmin}} \sum_p \|S(q) - T(p)\|^2 + \sum_p \|O(p) - E(q)\|^2 \quad (2)$$

where the first term captures matching between *source* and *target*, while the second term captures the output that best corresponds to the matched *exemplar*.

Equation 2 bears a striking resemblance to Equation 1, the main difference being that [WSI07] directly optimizes for the *output* that minimizes the difference to the *exemplar*, while [SPDF13] optimizes for an intermediate color transformation. In addition, the algorithm by Wexler et al. optimizes for matches that reduce differences between both *source* and *target*, and *exemplar* and *output*. Shih et al. conversely assume that matches are fixed, which allows them to minimize Equation 1 with a standard linear solver.

Our unified algorithm selectively applies one algorithm or the other by first predicting the success of color transfer from an analysis of the *source*/*exemplar* pair.

4. Unified Color and Texture Transfer

A key novelty in our approach resides in teasing apart appearance changes that can be modeled simply as changes in color (e.g., blue sky to overcast sky, green grass to brown grass) versus changes that require new visual signal to be synthesized on the *target* (e.g., bare tree to leafy tree). Unfortunately, we do not know the correct solution beforehand,

since we do not have access to the ground truth *output*. However, we can use the *source/exemplar* (*S*/*E*) pair as a training set since we know that E is the correct solution for S!

The analysis part of our algorithm thus consists in transferring color from E to S and measuring the error ε_S between the resulting color-transferred source image ($E \to S$) and the ground truth E (Figure 2(a)). The pixels of S and E that are not well related by a color transformation are the ones that are likely to require texture synthesis to be successfully transferred. During synthesis, *target* patches that match to areas of low error in the *source* are handled by color transfer, while others are handled by pixel copy (Figure 2(b)). Sections 4.1 and 4.2 describe our analysis and synthesis methods, respectively. We provide implementation details on color transfer and image matching in Section 5.

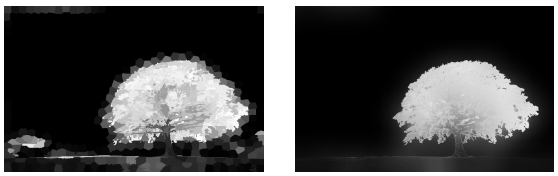
4.1. Predicting Success of Color Transfer

The first part of our algorithm compares small regions of the *exemplar* (E in Figure 2(a)) to the corresponding regions in the color-transferred *source* ($E \to S$) to predict the error ε_S of color transfer. We first discuss our choice of a suitable metric for this comparison and then describe how we compute and filter the metric on small image regions.

Choice of metric. We experimented with a number of image metrics to perform our prediction. In particular, we found that a simple Mean Square Error is often too sensitive to the small misalignments typical of our *source/exemplar* pairs. Comparing RGB covariance matrices, as recently used for texture smoothing [KEE13], partly addresses misalignment but is not sufficient to distinguish textures of similar color distribution but different pattern. We thus considered popular texture descriptors [LM01, VZ05, CMK*14] and found that the texton histograms described by Varma and Zisserman [VZ05] perform best on our data. We provide visual comparisons of these metrics as supplemental material.

In a nutshell, the method builds a dictionary of small image patches, the so-called textons, using K-means clustering on a large training set of filtered patches. In our implementation we use all patches of the *exemplar* E and *color-transferred source* $E \to S$ as the training set. Given an image region to analyze, the patch centered on each pixel of the region is assigned the unique label of its closest texton. The descriptor of the region then consists of the histogram of its texton labels. Finally, we compute the error prediction ε_S between E and $E \to S$ as the χ^2 distance between the histograms of their respective regions.

We experimented with dictionaries of size $K = 16$ to 1024 and found that a size of 48 textons offers a good trade-off between expressiveness and over-fitting on all our datasets. We also experimented with different filter banks and obtained the best results with the rotation-invariant filters described by Varma and Zisserman [VZ05], except that we did not use the Gaussian filter since it only captures smooth regions for which color transfer typically works well.



(a) ε_S per super-pixel (b) ε_S after filtering

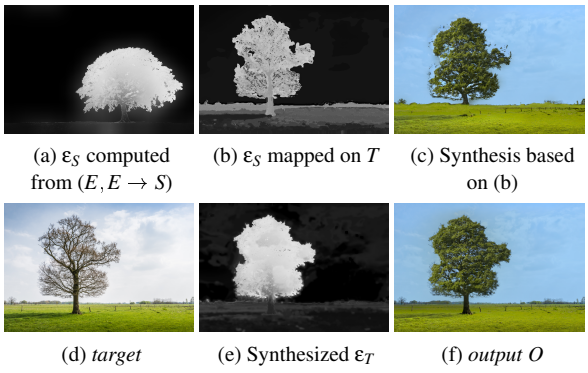
Figure 3: We first compute the error prediction ε_S on super-pixels (a). We then apply an outlier rejection and a bilateral filter to improve spatial coherence (b).

Computation and filtering. While texton histograms can be computed over square image patches, we found that super-pixels better preserve boundaries between regions of different textures. We use the algorithm by Achanta et al. [ASS*12] to compute the super-pixels on the 6-channel image composed of the stacked *source* and *exemplar*, which results in the same segmentation in the two images.

Computing the texton metric for each super-pixel independently often yields a noisy estimate due to slight variations of texture over the image and misalignment between *source* and *exemplar* (Figure 3(a)). We improve the spatial coherence of our estimation by first grouping super-pixels that are visually similar and assigning the median error of each group to its members. We apply mean-shift clustering to group super-pixels according to the distance between their texton histograms, using Euclidean distance for speed-up. We then apply a cross-bilateral filter [ED04] guided by the *exemplar* image to smooth out discontinuities along super-pixel boundaries. Laffont et al. [LRT*14] describe a similar outlier rejection and filtering method in the context of color transfer. Figure 3 illustrates our error prediction before and after filtering.

4.2. Selective Texture Transfer

We now have the color transfer error ε_S predicted from the *source/exemplar* pair (Figures 2(a), 3(b)). We could directly map this error onto the *target* using dense matches (Figure 4(a,b)), and fill the areas of high error with texture synthesis. However, this approach often produces visual artifacts in cases where we want the new content to change the shape of the region where it is synthesized. Figure 4(c) illustrates such artifacts where leaves are synthesized only *inside* the branches of a tree rather than *around* them. We address this issue by updating the error prediction on the *target* during the iterative texture synthesis. This effectively allows the synthesized region to grow around the tree in this example (Figure 4(e,f)), resulting in leaves being synthesized outside the original tree silhouette. This approach is similar



(a) ε_S computed from $(E, E \to S)$ (b) ε_S mapped on T (c) Synthesis based on (b)

(d) *target* (e) Synthesized ε_T (f) *output* O

Figure 4: We could directly map the error ε_S measured on the *source* and *exemplar* (a) onto the *target* (d) to predict where synthesis should be used (b). However, this approach produces low quality results as the synthesis is constrained to occur within the silhouette of the bare tree (c). In contrast, our algorithm updates the error prediction ε_T during synthesis (e) allowing the leaves to grow around the tree (f).

in spirit to guided texture synthesis [HJO*01, BVDPLD10] where the guidance field is synthesized jointly with the texture.

Algorithm 1 provides the pseudo-code of our synthesis procedure, which we implement as an adaptation of the iterative optimization of Wexler et al. [WSI07]. Our algorithm takes as input the *source*/*exemplar*/*target* triplet along with the predicted error ε_S, which measures the success of color transfer for each pixel of the *source*/*exemplar* pair. As a preprocess, we use Otsu's method [Ots79] to automatically select an appropriate threshold on ε_S. This method estimates the optimum threshold to cluster low and high error pixels in two classes with minimal intra-class variance. We apply this threshold at each iteration of the algorithm to convert the error ε_T predicted on the *target* to a binary mask $\tilde{\varepsilon}_T$, which indicates where to perform synthesis.

We initialize the *output* O^0 by applying color transfer from *exemplar* to *target*. At each iteration t, we first build a set of per-pixel matches $\mathcal{N}^t$ between the stacked *source*/*exemplar* (S/E) and the *target*/*output* (T/O^{t-1}) image pairs using PatchMatch [BSFG09]. These matches minimize Equation 2 given the *output* of the previous iteration. We then perform the *voting* step of the optimization, during which we accumulate the colors of the matched *exemplar* patches over the new *output* O^t. Voting thus updates each pixel of the *output* with the average color of the *exemplar* pixels to which it matches, which effectively minimizes Equation 2 given the current matches. However, in contrast to [WSI07], the *exemplar* patch also votes for the *predicted error* on *target* ε_T. Thresholding ε_T provides us with the binary mask $\tilde{\varepsilon}_T$, which we soften by applying a cross-bilateral filter guided by the texture-synthesized image. We use the resulting soft mask as an α weight to blend the synthesized image O^t with the color-transferred image O^0. We insist on the fact that blending occurs at each iteration of the synthesis, which is critical to obtain a seamless composite between color and texture transfer.

Algorithm 1 Selective Texture Transfer

Input: source S, exemplar E, target T, predicted error ε_S
// Select threshold of predicted error
$\tau \leftarrow$ Otsu (ε_S)
// Initialize output with color transfer
$O^0 \leftarrow$ ColorTransfer $(S, E \to T)$
for all levels from coarse-to-fine **do**
  for all iterations t **do**
    // Compute dense matching between S/E and T/O
    $\mathcal{N}^t \leftarrow$ PatchMatch $(S/E, T/O^{t-1})$
    // Initialize voting buffers for color and error synthesis
    $O^t \leftarrow 0, \varepsilon_T \leftarrow 0$
    for all patches p of size w^2 in T/O **do**
      // Recover matching patch in S/E
      $q \leftarrow \mathcal{N}^t(p)$
      // Accumulate votes for color synthesis
      $O^t(p) \leftarrow O^t(p) + \frac{1}{w^2} E(q)$
      // Accumulate votes for error prediction
      $\varepsilon_T(p) \leftarrow \varepsilon_T(p) + \frac{1}{w^2} \varepsilon_S(q)$
    end for
    // Threshold error prediction
    $\tilde{\varepsilon}_T \leftarrow$ Threshold (ε_T, τ)
    // Soften binary mask
    $\alpha \leftarrow$ CrossBilateral $(\tilde{\varepsilon}_T, O^t)$
    // Combine color transfer and texture transfer
    $O^t = \alpha\, O^t + (1 - \alpha)\, O^0$
  end for
end for

5. Implementation Details

Color Transfer. We perform color transfer with a variant of the method proposed by Shih et al. [SPDF13]. In the original formulation, the algorithm applies the same color transformation from *source* to *exemplar* and from *target* to *output*, as illustrated in Figure 6a. This approach performs well in the context of relighting, where one wants to transfer changes of tone between objects of possibly different intrinsic colors, such as transferring a sunset lighting from a green house to a blue building. However, it can produce unnatural colors in our context, as shown in Figure 5(b,c) where the *output* grass receives a saturated green because the *source* grass is darker than the *target* grass.

We adapted the algorithm by Shih et al. [SPDF13] to directly compute the affine color transformations between *target* and *exemplar*, using the *source* solely as a means to build correspondences (Figure 6(b)). Using the same notation as in

(a) S/E (b) target (c) Shih et al. [SPDF13] (d) Laffont et al. [LRT*14] (e) Ours

Figure 5: Comparison of color transfer implementations. Prior work computes the affine color transformations between the *source* and *exemplar* and applies them on *target*, which can produce unnatural colors in our context (c,d). Instead, we directly compute the color transformations between the *target* and *exemplar* (e).

Equation (1), our modified algorithm minimizes

$$\underset{f_p,O}{\operatorname{argmin}} \sum_p \|E(q) - f_p(T(p))\|^2 + \frac{1}{\lambda}\sum_p \|O(p) - f_p(T(p))\|^2 \tag{3}$$

where the first term now captures the transformation that best relates the *target* to the *exemplar*. This approach produces an *output* with similar colors to the *exemplar* (Figure 5(d)). Note that we used matches provided by the PatchMatch algorithm [BSFG09] for this comparison, while Shih et al. [SPDF13] use a different matching algorithm to leverage additional information in time-lapse video data.

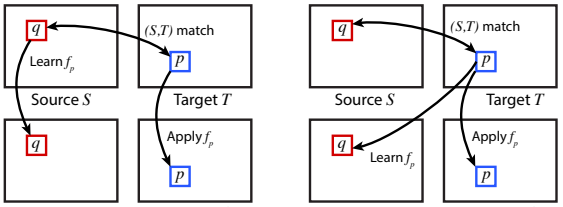
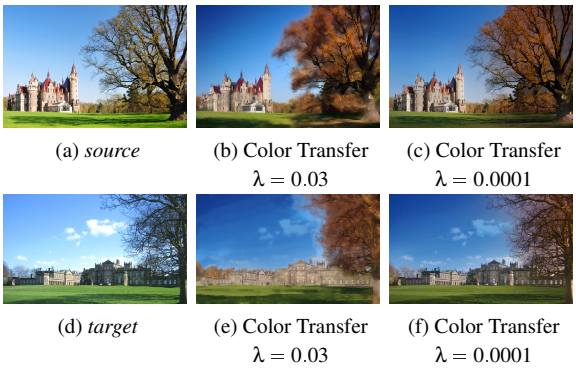


Figure 6: [SPDF13] learn local affine color transformations between *source* and *exemplar* and apply these transformations on *target* (a). Instead, we directly compute the local affine transformations between *exemplar* and *target* (b).

Image matching and co-segmentation. Both the color and texture transfer steps of our algorithm rely on the computation of dense correspondences between *source* and *target*. However, while we tried to select *source*/*exemplar* pairs that are visually similar to the desired *target*, such pairs are rare. As a result, automatic matching algorithms like PatchMatch [BSFG09] were challenged by some of the images we selected. Inspired by user-guided color transfer algorithms [AP10], we improved matching on some challenging cases by providing a user-assisted co-segmentation of the *source* and *target* images. We use the Photoshop quick selection tool for this purpose and subsequently refine the segments with automatic matting. We then constrain the matching algorithm to only consider patches that belong to the same segments, and update the *output* segments during synthesis. In Section 6 we indicate for each result whether it was computed using co-segmentation. Improvements of PatchMatch that account for image transformations could alleviate the need for co-segmentation [HSGL11, HXM*13].



(a) *source* (b) Color Transfer $\lambda = 0.03$ (c) Color Transfer $\lambda = 0.0001$

(d) *target* (e) Color Transfer $\lambda = 0.03$ (f) Color Transfer $\lambda = 0.0001$

Figure 7: We use a high value of the color transfer parameter λ when transferring color from *exemplar* to *source* in order to also transfer local shading and shadows (b). A smaller value preserves the lighting of the *source* (c), which pollutes our metric. In contrast, we use small value of λ to transfer color to *target* in order to preserve its lighting (f).

Parameters. We use the same parameters for all results in the paper, except when mentioned specifically. In particular, we set the parameters of the super-pixel segmentation to produce 500 segments on average and we use mean-shift with a relatively small bandwidth of 0.09 to group super-pixels during filtering of ε_S. For synthesis we use 10 coarse-to-fine levels, and decrease the number of iterations from 20 to 4 going from coarse to fine; for higher resolution images more levels can be necessary. We set the range parameter of the cross-bilateral filters to $\sigma_r = 1/40$ and the spatial parameter to $\sigma_s = 1/32$ of image height, except for the snowy street example in Figure 9 where we used a larger filter of $\sigma_s = 1/6$ of image height because our metric has a higher variance over the snow part that is highly saturated in this image.

The color transfer algorithm of Shih et al. [SPDF13] has a

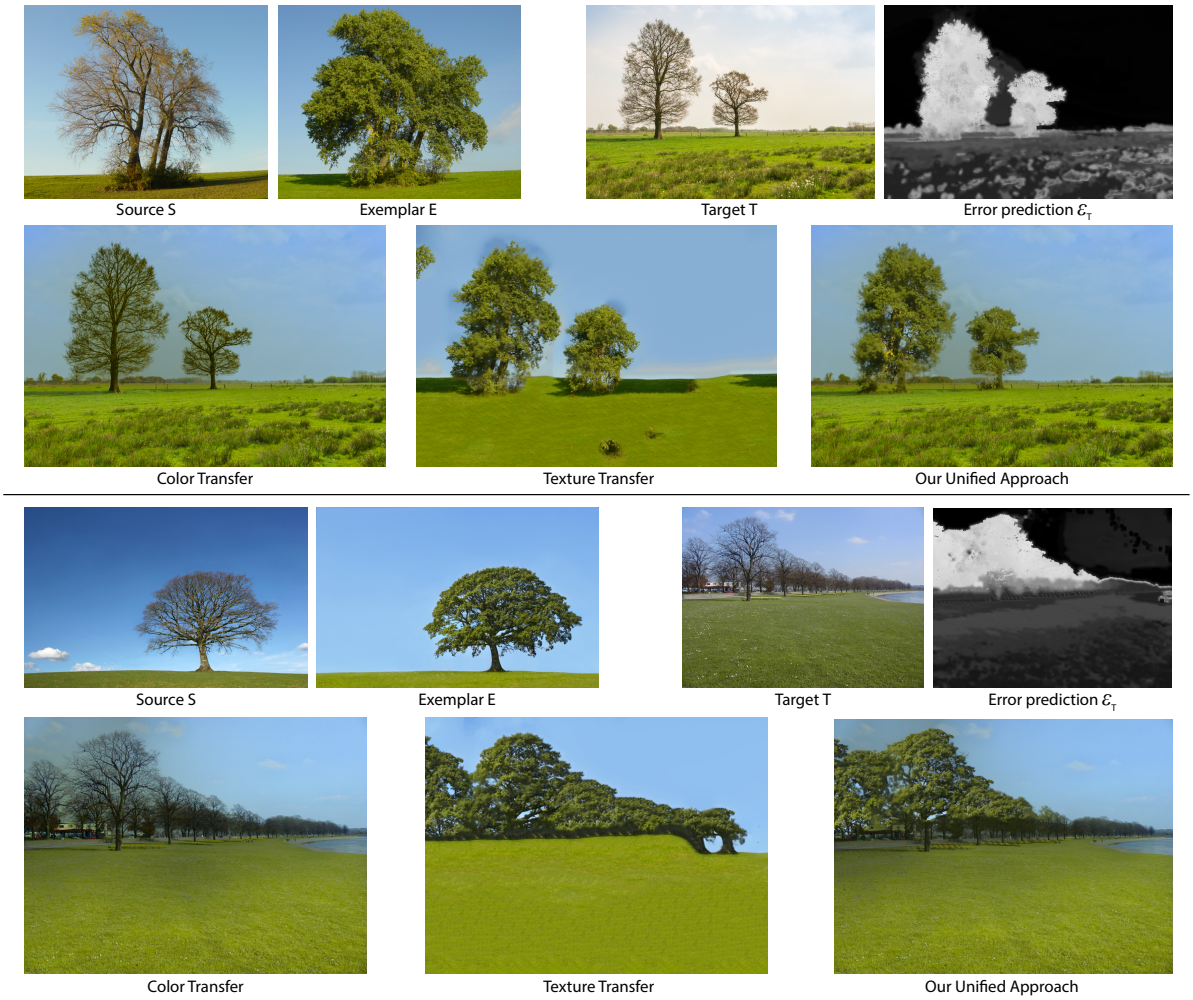


Figure 8: Results generated **automatically** with our method. Our metric successfully predicts that synthesis is needed to create leaves on the trees. Other details of the scenes, such as grass on the ground, are preserved. In contrast, color transfer does not create any new content, while texture transfer removes important structure such as the lake in the second row.

parameter λ to balance between transfer of the *exemplar* appearance and preservation of the *target* structure (see Equation 3). During the analysis phase of our algorithm we set a high value of $\lambda = 0.03$ in order to transfer as much color as possible between the *source* and its aligned *exemplar*. This ensures that even local shading and shadows are transferred and that remaining differences are only due to change of texture. However, applying the same parameter during synthesis doesn't preserve enough of the *target* structure (see Figure 7). We thus use a lower value of $\lambda = 0.0001$ for synthesis.

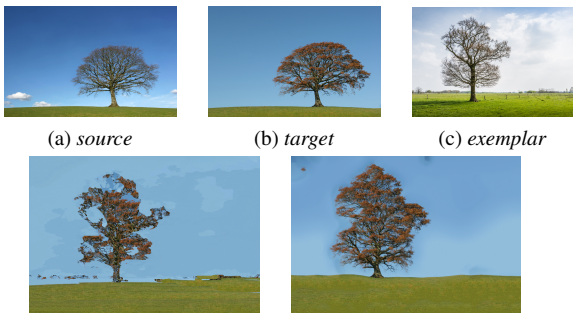
6. Results

Figure 8 provides results computed *automatically* with our method. Figure 9 contains more complex scenes that we processed using the user-provided co-segmentations shown in Figure 10. Our algorithm successfully captures the visual appearance of drastic temporal changes, converting a scene from winter to summer, creating piles of snow in the street and populating trees with foliage. Many of the remaining artifacts in our results are due to the limitations of the underlying texture synthesis algorithms. Nevertheless, in all cases our unified transfer is more convincing than color transfer or texture transfer alone. While we used our own implementations of color and texture transfer to perform this comparison, Figure 5 and 11 show that these implementations are on par or better than reference implementations. Please refer to supplemental materials for additional results, all images, error metrics and thresholded masks.

We performed a ground truth comparison by splitting a pair of aligned images in two as illustrated in Figure 12.

Figure 9: Results generated with our method and user-provided co-segmentation. Our algorithm transfers leaves in the trees, piles of snow in the street and water over flooded fields. In contrast, applying texture transfer on the entire image results in broken structure, such as the castle in the top row and the church in the bottom row. Texture synthesis also fails to synthesize a plausible wall in the street scene due to a lack of exemplar content.

Figure 10: User-provided co-segmentations used for the challenging *source*/*target* combinations of Figure 9.



(a) *source* (b) *target* (c) *exemplar*

(d) Image analogies (e) Our implementation

Figure 11: Our implementation of texture transfer (e) includes modern improvements over the reference implementation [HJO*01] (d).

The left hand sides of the two images are used as the *source*/*exemplar* pair, whereas the right hand side of the first image is used as the *target*. The second image's right hand side then serves as a ground truth reference. Our output is visually close to the ground truth and better conveys the spread of water over the fields than color transfer alone.

7. Limitations and Future Work

Our algorithm inherits the limitations of its components, namely color and texture transfer and image matching. Currently, texture transfer only performs moderately well for transformations requiring image content to be "removed", e.g., from a leafy tree to a bare tree. Figure 13(top) illustrates limited success on such a case, where the synthesized branches do not align well with the trunk of the tree. Figure 13(bottom) illustrates failure of matching two trees because they are from different species and over different backgrounds. Nevertheless, our approach synthesized some leaves in the areas where it found matches.

Our method is most useful to capture transformations that involve both color and structural changes, tree foliage being

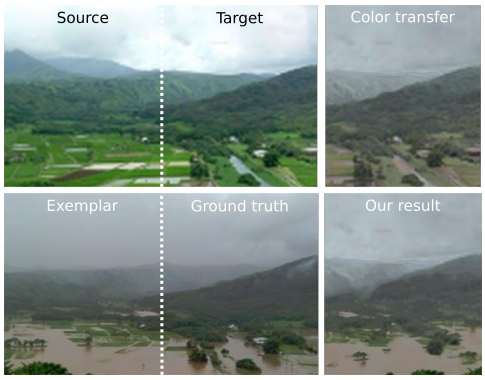


Figure 12: Ground truth comparison. A ground truth set is built from an aligned image pair. The first image is split into *source* and *target* (top left). The second image (bottom left) is split into an *exemplar* (lhs of the image) and a ground truth reference (rhs). Our result is close to the the ground truth (lower right), unlike the color transferred target (top right).

(a) S/E (b) *target* (c) *output*

Figure 13: Our approach inherits the limitations of texture transfer and has difficulty creating plausible branches from a dense foliage (top). Image matching is also difficult between trees of different species or against different backgrounds (bottom). As a result, our algorithm only generates leaves on a the top branches in this example.

a common case of such transformations. Our method also assumes that the input images capture an analogy relationship between *source* and *exemplar*. This assumption fails if the *source* contains two regions with a similar appearance in the *source* but different appearances in the *exemplar*, such as two trees that have similar branches in winter but different leaves in summer.

As any by-example approach, the success of our method greatly depends on the input *source*/*exemplar*/*target* triplet. We have demonstrated our algorithm by selecting triplets

(a) Source (b) Exemplar

(c) Target (d) Output

Figure 14: We created an artificial *source/exemplar* pair by combining the aligned images of a tree with an image of a grass field and an image of a flower field (a,b). Our algorithm synthesizes flowers over the darkest parts of the grass field while it uses color transfer for the remaining parts.

manually from online photo collections. In the future we plan to explore the use of modern image retrieval and matching algorithms [SMGE11] to automate the search for *source* images that best match a *target* and for *exemplar* images that best align with a *source*.

The selection of images can also be greatly simplified by creating artificial *source/exemplar* pairs as a montage of multiple pictures. Figure 14 shows such a *source/exemplar* composition where we combined the images of a tree in winter and summer with the image of a grass field and the image of a flower field. Note that we roughly positioned the flowers in the *exemplar* so that they correspond to the darker areas of the grass field in the *source*. Our algorithm transfers this appearance by synthesizing flowers over the dark grass of the *target*. Image retrieval techniques could also help finding suitable *source* and *exemplar* images that best match the different parts of a *target*.

8. Conclusion

We have unified two complementary methods for exemplar-based appearance manipulation, namely color and texture transfer. We do this by exploiting their respective strengths and identifying their common methodological components.

The first key element of this unification is a method to *predict* the image regions where color transfer is insufficient by analyzing the *source/exemplar* pair. The second element is our selective texture transfer algorithm that updates the error prediction during synthesis for improved results. Thanks to these innovations our method achieves appearance manipulation largely superior to previous approaches, which were restricted to either color or texture transfer.

The problem we addressed is very hard in the general case. In this work, we assumed that the input *source/exemplar/target* triplets were given. Developing automatic methods to find *source/exemplar* pairs suitable for a given *target* and a desired appearance transformation is a very exciting avenue for future work, and would naturally extend approaches such as [LRT*14]. Another central component is the quality of matching; recent methods [HSGL11, HXM*13] show how matching can be improved by accounting for color transformations. We believe that our application can benefit from similar strategies in the future. The actual texture synthesis algorithm we used for transfer can also be improved, possibly by accounting for foreground/background separation [KSD*14]. Finally, we are confident that the overall strategy we adopted can apply to other image manipulation tasks where two complementary algorithms can be combined based on a prediction of their success.

Acknowledgments. We thank Sylvain Paris as well as the anonymous reviewers for their comments and suggestions. This work was supported by the Inria CRISP associate team and research and software donations from Adobe. Fumio Okura was supported by JSPS Strategic Young Researcher Overseas Visits Program for Accelerating Brain Circulation (G2503) and JSPS KAKENHI 25-7448.

Image credits. Figure 1(a,b): Marilyn Barbone (Shutterstock); Figure 1(c): Ruud Morijn (Shutterstock); Figure 7(a): Adam Brzuszek (Shutterstock); Figure 7(d): Alan J. White; Figure 8 top *source/exemplar*: public domain (author: Cherubino), *target*: Ruud Morijn (Shutterstock); Figure 8 top *source/exemplar*: public domain (author: Cherubino), *target*: Ruud Morijn (Shutterstock); Figure 8 bottom *source/exemplar*: Marilyn Barbone (Shutterstock), *target*: freeimages.co.uk; Figure 9 top *source/exemplar*: Adam Brzuszek (Shutterstock), *target*: Alan J. White; Figure 9 middle top *source/exemplar*: David Stewart (Boston.com, Getty-Images), *target*: JPRichard (Shutterstock); Figure 9 middle bottom *source*: Bradley West (flickr), *exemplar*: Aaron Feinberg (flickr), *target*: Surfing The Nations (flickr); Figure 9 bottom *source*: Shaun Stothard (flickr), *exemplar*: Tim Green (flickr), *target*: Gerald England; Figure 12: Nathan Teixeira; Figure 13 top (a) Peter Hansen(Shutterstock), (b) FotoYakov (Shutterstock), bottom (a) Shaun Stothard (flickr), (b) Adam Swaine (flickr). Many thanks to all for allowing us to use these images.

References

[AP10] AN X., PELLACINI F.: User-controllable color transfer. *Computer Graphics Forum 29*, 2 (2010), 263–271. 2, 6

[ASS*12] ACHANTA R., SHAJI A., SMITH K., LUCCHI A., FUA P., SUSSTRUNK S.: Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Trans. Pattern Analysis Machine Intelligence (PAMI) 34*, 11 (Nov. 2012), 2274–2282. 4

[BCK*13] BÉNARD P., COLE F., KASS M., MORDATCH I., HEGARTY J., SENN M. S., FLEISCHER K., PESARE D., BREEDEN K.: Stylizing animation by example. *ACM Transactions on Graphics 32*, 4 (2013), 119. 2

[BPD06] BAE S., PARIS S., DURAND F.: Two-scale tone management for photographic look. *ACM Transactions on Graphics (Proc. SIGGRAPH) 25*, 3 (2006), 637–645. 1

[BSFG09] BARNES C., SHECHTMAN E., FINKELSTEIN A., GOLDMAN D.: Patchmatch: a randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics (Proc. SIGGRAPH) 28*, 3 (2009), 24. 3, 5, 6

[BVDPLD10] BONNEEL N., VAN DE PANNE M., LEFEBVRE S., DRETTAKIS G.: Proxy-guided texture synthesis for rendering natural scenes. In *Proc. VMV 2010* (2010), pp. 87–95. 2, 5

[CMK*14] CIMPOI M., MAJI S., KOKKINOS I., MOHAMED S., VEDALDI A.: Describing textures in the wild. In *Computer Vision and Pattern Recognition (CVPR)* (2014). 4

[DBP*15] DIAMANTI O., BARNES C., PARIS S., SHECHTMAN E., SORKINE-HORNUNG O.: Synthesis of complex image appearance from limited exemplars. *ACM Transactions on Graphics 34*, 2 (2015). 2

[DSB*12] DARABI S., SHECHTMAN E., BARNES C., GOLDMAN D. B., SEN P.: Image melding: combining inconsistent images using patch-based synthesis. *ACM Transactions on Graphics (Proc. SIGGRAPH) 31*, 4 (2012), 82. 3

[ED04] EISEMANN E., DURAND F.: Flash photography enhancement via intrinsic relighting. *ACM Transactions on Graphics (Proc. SIGGRAPH) 23* (2004). 4

[EF01] EFROS A. A., FREEMAN W. T.: Image quilting for texture synthesis and transfer. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (2001), pp. 341–346. 1, 2

[FJP02] FREEMAN W. T., JONES T. R., PASZTOR E. C.: Example-based super-resolution. *Computer Graphics and Applications, IEEE 22*, 2 (2002), 56–65. 2

[FPC*14] FARIDUL H. S., POULI T., CHAMARET C., STAUDER J., TREMEAU A., REINHARD E.: A Survey of Color Mapping and its Applications. In *Eurographics 2014 - State of the Art Reports* (2014), The Eurographics Association. 2

[HJO*01] HERTZMANN A., JACOBS C. E., OLIVER N., CURLESS B., SALESIN D. H.: Image analogies. In *Proc. SIGGRAPH'01* (2001), pp. 327–340. 1, 2, 3, 5, 9

[HSGL11] HACOHEN Y., SHECHTMAN E., GOLDMAN D. B., LISCHINSKI D.: Non-rigid dense correspondence with applications for image enhancement. *ACM Transactions on Graphics (Proc. SIGGRAPH) 30*, 4 (2011), 70. 6, 10

[HXM*13] HU S.-M., XU K., MA L.-Q., LIU B., JIANG B.-Y., WANG J.: Inverse image editing: Recovering a semantic editing history from a before-and-after image pair. *ACM Transactions on Graphics 32*, 6 (2013), 194:1–194:11. 2, 6, 10

[ICOL05] IRONY R., COHEN-OR D., LISCHINSKI D.: Colorization by example. In *Proceedings of the Sixteenth Eurographics conference on Rendering Techniques* (2005), pp. 201–210. 2

[JDA*11] JOHNSON M. K., DALE K., AVIDAN S., PFISTER H., FREEMAN W. T., MATUSIK W.: Cg2real: Improving the realism of computer generated images using a large collection of photographs. *Visualization and Computer Graphics, IEEE Transactions on 17*, 9 (2011), 1273–1285. 2

[KEE13] KARACAN L., ERDEM E., ERDEM A.: Structure-preserving image smoothing via region covariances. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia) 32*, 6 (2013). 4

[KKDK12] KOPF J., KIENZLE W., DRUCKER S., KANG S. B.: Quality prediction for image completion. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia) 31*, 6 (2012). 2

[KSD*14] KALANTARI N. K., SHECHTMAN E., DARABI S., GOLDMAN D. B., SEN P.: Improving Patch-Based Synthesis by Learning Patch Masks. In *International Conference on Computational Photography (ICCP)* (2014). 10

[LM01] LEUNG T., MALIK J.: Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal Computer Vision 43*, 1 (2001). 4

[LRT*14] LAFFONT P.-Y., REN Z., TAO X., QIAN C., HAYS J.: Transient attributes for high-level understanding and editing of outdoor scenes. *ACM Transactions on Graphics (Proc. SIGGRAPH) 33*, 4 (2014). 1, 2, 4, 6, 10

[Ots79] OTSU N.: A Threshold Selection Method from Gray-level Histograms. *IEEE Transactions on Systems, Man and Cybernetics 9*, 1 (1979), 62–66. 5

[PKD07] PITIÉ F., KOKARAM A. C., DAHYOT R.: Automated colour grading using colour distribution transfer. *Computer Vision and Image Understanding 107*, 1 (2007), 123–137. 2

[RAGS01] REINHARD E., ASHIKHMIN M., GOOCH B., SHIRLEY P.: Color transfer between images. *IEEE Computer graphics and applications 21*, 5 (2001), 34–41. 2

[SCBH11] SWAMY D. S., CHANDLER D. M., BUTLER K. J., HEMAMI S. S.: Parametric quality assessment of synthesized textures. In *Proc. Human Vision and Electronic Imaging* (2011). 2

[SCSI08] SIMAKOV D., CASPI Y., SHECHTMAN E., IRANI M.: Summarizing visual data using bidirectional similarity. *Computer Vision and Pattern Recognition (CVPR)* (2008). 3

[SMGE11] SHRIVASTAVA A., MALISIEWICZ T., GUPTA A., EFROS A. A.: Data-driven visual similarity for cross-domain image matching. *ACM Transaction of Graphics(Proc. of ACM SIGGRAPH Asia) 30*, 6 (2011). 2, 10

[SPDF13] SHIH Y., PARIS S., DURAND F., FREEMAN W. T.: Data-driven hallucination of different times of day from a single outdoor photo. *ACM Transactions on Graphics (TOG) 32*, 6 (2013). 1, 2, 3, 5, 6

[TJT05] TAI Y.-W., JIA J., TANG C.-K.: Local color transfer via probabilistic segmentation by expectation-maximization. In *Computer Vision and Pattern Recognition (CVPR)* (2005), vol. 1. 2

[VZ05] VARMA M., ZISSERMAN A.: A statistical approach to texture classification from single images. *International Journal of Computer Vision 62*, 1–2 (2005), 61–81. 4

[WAM02] WELSH T., ASHIKHMIN M., MUELLER K.: Transferring color to greyscale images. *ACM Transactions on Graphics (Proc. SIGGRAPH) 21*, 3 (2002), 277–280. 2

[WLKT09] WEI L.-Y., LEFEBVRE S., KWATRA V., TURK G.: State of the art in example-based texture synthesis. In *Eurographics 2009, State of the Art Report, EG-STAR* (2009), Eurographics Association. 2

[WSI07] WEXLER Y., SHECHTMAN E., IRANI M.: Space-time completion of video. *IEEE Trans. Pattern Analysis Machine Intelligence (PAMI) 29*, 3 (Mar. 2007), 463–476. 3, 5