Eurographics Symposium on Rendering (2005)
Kavita Bala, Philip Dutré (Editors)

Colorization by Example

R. Irony[1], D. Cohen-Or[1], and D. Lischinski[2]

[1] Tel-Aviv University
[2] The Hebrew University of Jerusalem

Abstract

We present a new method for colorizing grayscale images by transferring color from a segmented example image. Rather than relying on a series of independent pixel-level decisions, we develop a new strategy that attempts to account for the higher-level context of each pixel. The colorizations generated by our approach exhibit a much higher degree of spatial consistency, compared to previous automatic color transfer methods [WAM02]. We also demonstrate that our method requires considerably less manual effort than previous user-assisted colorization methods [LLW04].

Given a grayscale image to colorize, we first determine for each pixel which example segment it should learn its color from. This is done automatically using a robust supervised classification scheme that analyzes the low-level feature space defined by small neighborhoods of pixels in the example image. Next, each pixel is assigned a color from the appropriate region using a neighborhood matching metric, combined with spatial filtering for improved spatial coherence. Each color assignment is associated with a confidence value, and pixels with a sufficiently high confidence level are provided as "micro-scribbles" to the optimization-based colorization algorithm of Levin et al. [LLW04], which produces the final complete colorization of the image.

Categories and Subject Descriptors (according to ACM CCS): I.4.9 [Image Processing and Computer Vision]: Applications;

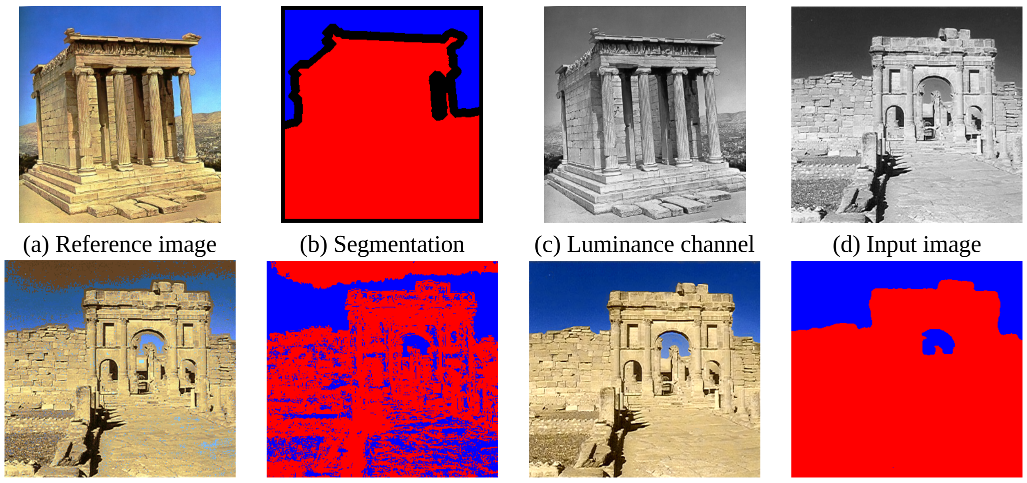
1. Introduction

Colorization, the process of adding color to monochrome images and video, has long been recognized as highly laborious and tedious. Despite several recent important advances in the automation of the process, a considerable amount of manual effort is still required in many cases in order to achieve satisfactory results.

For example, Levin *et al.* [LLW04] recently proposed a simple yet effective user-guided colorization method. In this method the user is required to scribble the desired colors in the interiors of the various regions. These constraints are formulated as a least-squares optimization problem that automatically propagates the scribbled colors to produce a completely colorized image. Other algorithms based on color scribbles have subsequently been proposed [Sap04, YS04]. While this approach has produced some impressive colorizations from a small amount of user input, sufficiently complex images may still require dozens, or more, carefully placed scribbles, as demonstrated in figure 2(a).

In addition to the manual effort involved in placing the scribbles, the pallet of colors must also be chosen carefully in order to achieve a convincing result, requiring both experience and a good sense of aesthetics. This difficulty may be alleviated by choosing the colors from a similar reference color image. In fact, Welsh *et al.* [WAM02] proposed an automatic colorization technique that colorizes an image by matching small pixel neighborhoods in the image to those in the reference image, and *transferring* colors accordingly. This approach is a special case of the more general *image analogies* framework [HJO*01], where a general filter is learned from the relationship between two images A and A' and then applied to an input image B to produce a filtered result B'. However, image analogies and its derivatives typically make local (pixel level) decisions and thus do not explicitly enforce a contiguous assignment of colors. The Levin *et al.* method, on the other hand, promotes contiguity by formulating and solving a global optimization problem.

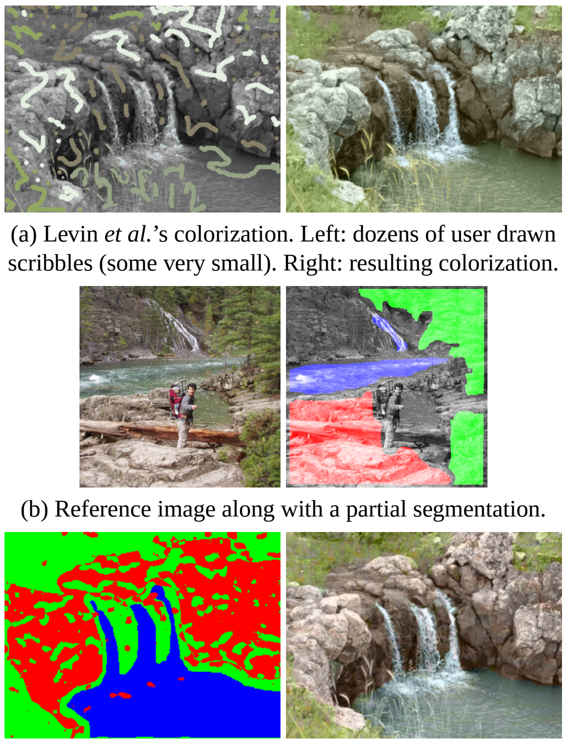
In this paper, we introduce a new color transfer method, which leverages the advantages of these two previous colorization approaches, while largely avoiding their shortcomings. Similarly to the method of Welsh *et al.*, our method colorizes one or more grayscale images, based on a user-provided reference — a partially segmented example color image. This requires considerably less input from the user than scribbling-based interfaces, and the user is relieved from the task of selecting appropriate colors (beyond sup-



(a) Reference image (b) Segmentation (c) Luminance channel (d) Input image

(e) Naive color transfer (f) Naive matching (g) Our colorization (h) Our matching

Figure 1: *Our approach vs. color transfer. A reference color image (a) was automatically segmented into two major regions (b). Transferring color to a grayscale image (d) by matching means and variances of pixel neighborhoods (as described in [WAM02]) produces a poor result in this case (e), since pixels in (d) are matched to pixels in (c) in an incoherent manner, as visualized in (f). Our approach produces a much better result (g), since it matches pixels in a more contiguous manner (h).*



(a) Levin *et al.*'s colorization. Left: dozens of user drawn scribbles (some very small). Right: resulting colorization.

(b) Reference image along with a partial segmentation.

(c) Our classification and resulting colorization.

Figure 2: *(a) The method of Levin et al. might require the user to carefully place a multitude of appropriately colored scribbles. (b) Our approach requires an example image with a few user-marked or automatically segmented regions, and produces a comparable colorization (c).*

plying the reference image). On the other hand, our method explicitly enforces spatial consistency, producing more robust colorizations than Welsh *et al.*, by using a spatial voting scheme followed by a final global optimization step. These advantages of our approach are demonstrated in figures 1 and 2.

Our approach is motivated by the observation that finding a good match between a pixel and its neighborhood in a grayscale image and a pixel in the reference image is not sufficient for a successful colorization. Often, pixels with the same luminance value and similar neighborhood statistics may appear in different regions of the reference image, which may have different semantics and different colors. For example, figure 1(e) shows the result of applying a simple nearest-neighbor matching based on the average luminance and the standard deviation in small pixels neighborhoods, and transferring the corresponding chromatic channels. In order to improve the results in such cases, Welsh *et al.* propose letting the user select pairs of corresponding swatches between the example and each input image, thus limiting the search for matching neighborhoods to particular regions. However, this user-assisted variant still relies on pixelwise decisions and does not enforce contiguity.

We argue that in order to assign colors correctly, a more elaborate analysis of the different regions in the reference image and of the relationship between pixels in the input image and these regions is necessary. Specifically, we begin by identifying several different source regions in the reference image, either by letting the user manually mark them, or by using automatic segmentation. Next, we construct a mapping between small pixel neighborhoods and points in a feature space, specifically designed to discriminate between pixels from different regions, based on local frequency anal-

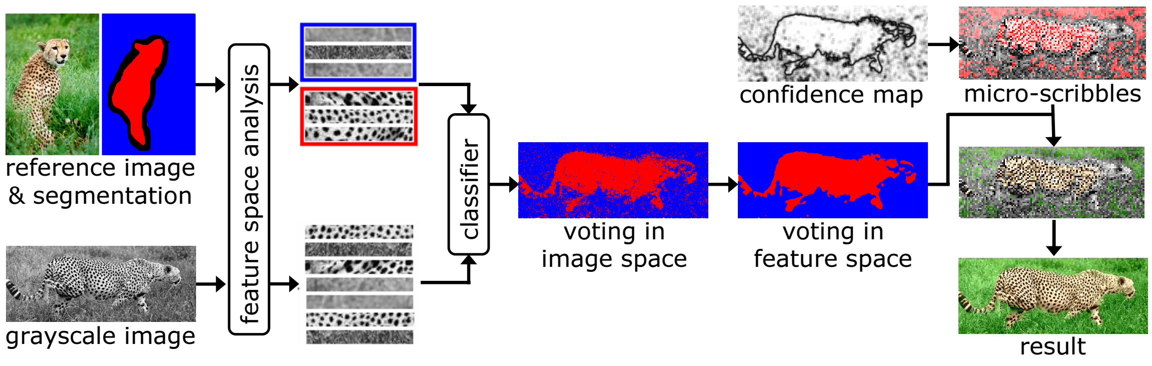


Figure 3: *An overview of our colorization technique: we begin by analyzing the segmented reference image and constructing an elaborate feature space, specifically designed to discriminate between different regions. Each pixel in the input image is then classified in this feature space using voting for robustness. Then, to make the decisions more spatially consistent, we explicitly enforce spatial consistency of colors by voting in image space, followed by a global optimization step.*

ysis in the luminance channel. This analysis is done once per reference image, and there is no need to mark corresponding regions in each input image (in contrast to the swatch-based variant of [WAM02]).

To colorize one or more grayscale images, we first classify each of their pixels to find out which region they match the best, using the feature space mentioned above. A robust classification scheme is crucial here, and we use voting both in feature space and in image space, for improved spatial consistency. Next, we transfer color only to pixels whose match is associated with a high level of confidence, and feed the colored pixels as "micro-scribbles" to the optimization algorithm of Levin *et al.* [LLW04], which produces the final complete colorization.

In summary, this paper makes the following contributions:

- We present a new automatic example-based colorization technique, meaning that once a reference image with some marked regions has been provided, any number of sufficiently similar grayscale images may be colorized without requiring any further input from the user.
- We describe a custom-tailored texture-based classifier derived from a low-level feature-space analysis of the reference image. Our analysis is close in spirit to linear discriminant analysis (LDA): it removes non-discriminating features and defines an effective classifier.
- Our method enforces spatially consistent color transfer by employing an image space voting scheme followed by a global optimization step.

2. Background

2.1. Colorization

Colorization is a term introduced by Wilson Markle in 1970 to describe the computer-assisted process he invented for adding color to black and white movies or TV programs [Bur]. In Markle's original colorization process [MH87] a color mask is manually painted for at least one reference frame in a shot. Motion detection and tracking are then applied, allowing colors to be automatically assigned to other frames in regions where no motion occurs. Colors in the vicinity of moving edges are assigned using optical flow, which often requires manual fixing by the operator.

Although the techniques used in more contemporary colorization systems are proprietary, and thus not much is known about them, it appears that these systems still rely on defining regions and tracking them between the frames of a shot [Sil98]. Since there are no completely automatic and robust segmentation and tracking algorithms, considerable user intervention in such systems is unavoidable. Consider, for example, BlackMagic, a commercial software for colorizing still images [Neu03]. This colorization tool provides the user with a variety of useful brushes and color palettes, but leaves the user with the burden of manually segmenting the image.

Reinhard *et al.* [RAGS01] describe an automatic system for transferring the color pallet from one color image to another. The user can guide the process by specifying pairs of corresponding swatches in the source and target images. This system works by modifying the mean and the variance of colors in the image, and thus is not directly applicable to colorization of grayscale images. The method of Welsh *et al.* [WAM02], which was discussed in the previous section, could be viewed as an extension of Reinhard's approach to the task of colorization. Recently, Chen *et al.* [CWSM04] used Welsh's approach to color objects extracted from grayscale images by an alpha matting computation.

Jia *et al.* [JSTS04] use a color transfer approach which considers color statistics and spatial constraints to recover a high quality image from two motion blurred images. Sykora *et al.* [SBZ04] presented a color-by-example technique for colorization of black and white cartoons, which combines image segmentation, patch-based sampling and probabilistic reasoning.

In addition to restoring colors in monochrome content, colorization is also used for pseudo-coloring medical images (X-ray, MRI, etc.) [GW87, Pra91]. In this case, the luminance values are mapped to color values, typically via a user-specified color lookup table.

2.2. Supervised Classification

There has been much work on methods for supervised classification and supervised segmentation; [HH97, HS89, HB03, PD02, Wei99] are just a few examples. Supervised classification methods typically consist of two phases: feature analysis and classification. In this paper we adopt a classification approach based on the K-nearest-neighbor (Knn) rule [DHS00]. This is an extremely simple yet effective method for supervised classification, which we will describe in more detail in section 3.1. Linear dimensionality reduction techniques are often used to make such classifiers both more efficient and more effective. For example, PCA-based techniques apply a linear projection that reduces the dimension of the data while maximizing the scatter of all projected samples. Linear Discriminant Analysis (LDA, also known as Fisher's Linear Discriminant) [BHK97, DHS00, Fis36], which finds a linear subspace in which the ratio of between-class scatter to that of within-class scatter is maximized. Improved variants of these techniques have recently been proposed by Shental *et al.* [SHWP02] and Goldberger *et al.* [GRHS04]. Our approach is also based on LDA, but rather than looking for a single optimal transformation (under certain assumptions that do not generally hold), we carry out dimensionality reduction using two consecutive projections: the goal of the first projection is to reduce inter-class variability, while the second maximizes intra-class variability.

3. Colorization by Example

Our algorithm colorizes one or more input grayscale images, based on a partially segmented reference color image. By partial segmentation we mean that one or more mutually disjoint regions in the image have been established, and each region has been assigned a unique label. These regions need not cover the entire image, but each region should be roughly uniform in color and texture. Example of partially segmented reference images are shown in figures 1(a–b) and 2(b). Such segmentations may be either computed automatically, or marked manually by the user.

An overview diagram of our approach is shown in figure 3. The approach consists of the following main conceptual stages: (i) training, (ii) classification, (iii) color transfer, and (iv) optimization.

In the training stage, the luminance channel of the reference image along with the accompanying partial segmentation are provided as a *training set* to a supervised learning algorithm. Informally, this algorithm constructs a low-dimensional feature space in which it is easy to discriminate between pixels belonging to differently labeled regions, based on a small (grayscale) neighborhood around each pixel. This construction is described in more detail in section 3.1.

In the classification stage we attempt to robustly determine, for each grayscale image pixel, which region should be used as a color reference for this pixel. This is done by voting among the pixel's nearest neighbors in the feature space constructed in the previous step, as described in section 3.1 as well. For improving the spatial coherence of the resulting classification, we additionally employ voting among each pixels neighbors in image space (section 3.2).

The matches found for each pixel and its image space neighbors also determine the color that should be assigned to each pixel, along with a measure of confidence in that choice. Finally, colored pixels with a sufficiently high level of confidence are given as "micro-scribbles" to the optimization-based colorization algorithm of Levin *et al.*, which interpolates these colors to all the remaining pixels. These last two stages are described in section 3.3.

3.1. Feature Spaces and Classifiers

Given the reference color image and its partial segmentation, our first task is to construct a feature space and a corresponding classifier. Recall that the partial segmentation consists of several regions, each associated with a unique label. Every pixel in one of these regions defines a labeled *feature vector*, a point in the feature space. Given a previously unseen feature vector, the goal of the classifier is to decide which label should be assigned to it.

Note that in our case the pixels to be classified come from the input grayscale images. Therefore, the classifier cannot rely on the colors of the pixels in the training set, and must be able to distinguish between different classes mainly based on texture. This implies that we should associate each pixel with a feature vector representing its monochromatic texture. In our current implementation we use the Discrete Cosine Transform (DCT) coefficients of a k by k neighborhood around the pixel as its feature vector. One of the advantages of using DCT coefficients is that they are a rather simple texture descriptor, which is not too sensitive to translations and rotations, since changes in phase and in direction do not affect the DCT representation. The DCT transform is applied only to the luminance channel of the reference image. This yields a k^2-dimensional feature space, populated with the labeled feature vectors corresponding to the training set pixels.

Once the feature space has been populated by labeled vectors, a novel feature vector may be naively classified by assigning it the label of its nearest feature space neighbor. However, in general, the training set pixels will not form nicely separated clusters in the feature space, so a more sophisticated classifier is required. One such classifier is defined by the K-nearest-neighbor (Knn) rule [DHS00]. This classifier examines the K nearest neighbors of the feature

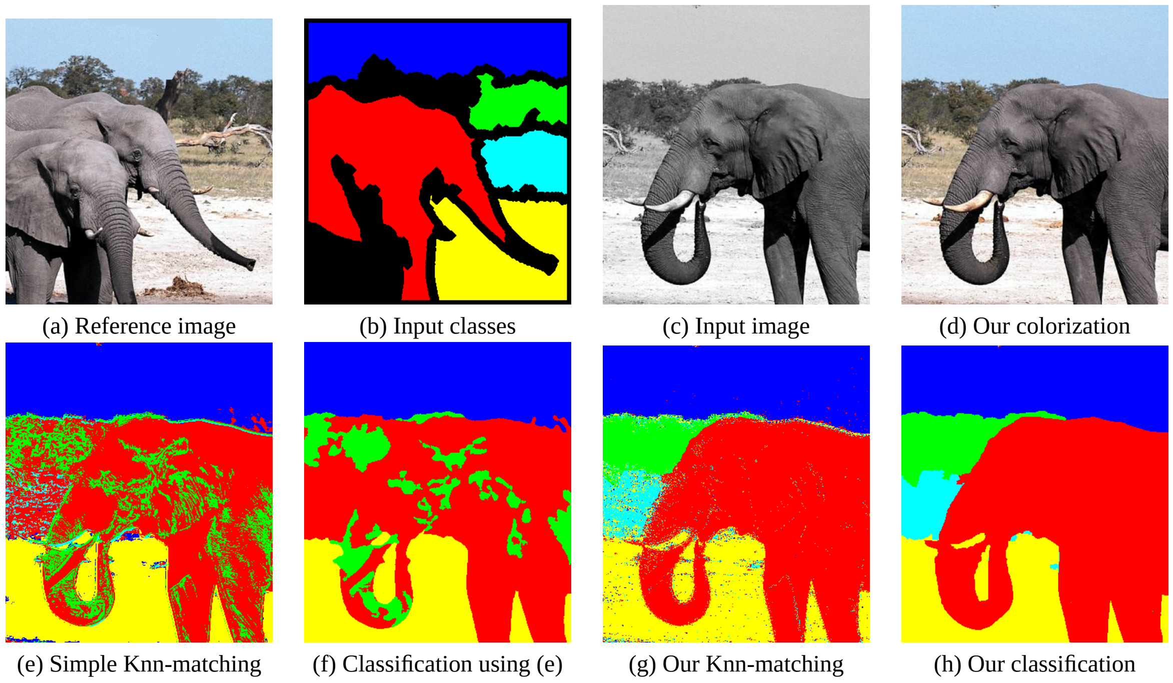


Figure 4: *Simple Knn-matching based on similar luminance value and neighborhood statistics (e) vs. our matching (g). The improved Knn-matching in color space results in better classification after considering spatial consistency: (f) classification based on simple Knn matching, (h) classification based on our matching.*

vector and chooses the label by a majority vote. Yet, applying the Knn classifier directly in the high-dimensional feature space may still lead to many erroneous classifications (figure 4(e)). Better results may be obtained by switching to a low-dimensional subspace, custom-tailored according to the training set, using an approach similar to linear discriminant analysis (LDA).

Let *intra-differences* be the difference vectors between points within the same class, and *inter-differences* be the difference vectors between points in different classes. We would like our classifier to ignore intra-differences, and make its decisions mainly based on inter-differences. By transforming (rotating) the space so that the new axis is aligned with the principle direction of the intra-difference vectors, and projecting the points onto the minor directions, we ignore irrelevant dimensions. That subspace is then transformed again so as to enhance the inter-differences among the classes.

The principle is illustrated in figure 5. Figure 4 shows the difference between applying Knn directly in the original high-dimensional feature-space (e–f) and to the subspace (g–h). Clearly the naive Knn classifier fails to discriminate between the bushes and the elephant.

To realize this idea we use PCA and projections. We first randomly sample a number of intra-difference vectors, apply PCA, and remove the eigenvectors that correspond to high eigenvalues. Then, similarly, we randomly sample the inter-differences in the resulting subspace and apply PCA again, this time keeping the eigenvectors corresponding to the largest eigenvalues. The result of this process is a transformation T which transforms the vector of k^2 DCT coefficients to a point in the low-dimensional subspace. We can

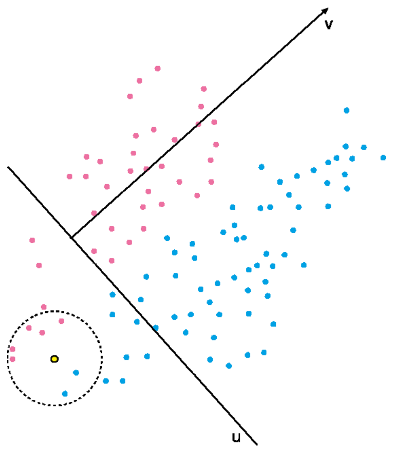


Figure 5: *Applying Knn in a discriminating subspace: the feature space is populated by points belonging to two classes: magenta and cyan. The yellow highlighted point has a majority of magenta-colored nearest neighbors. After rotating the space to the UV coordinate system, where V is the principle direction of the intra-difference vectors, and then projecting the points onto the U axis, all of the nearest neighbors are cyan.*

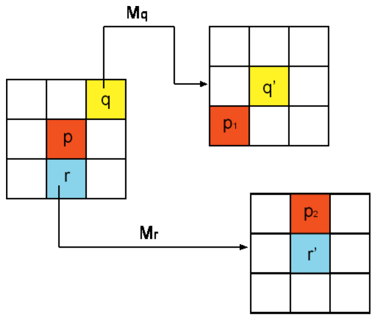


Figure 6: *Assigning color to pixel p: each neighbor of p (e.g., q, r) has a matching neighborhood in the reference image (M_q and M_r respectively), which "predicts" a different color for p (M_q predicts the color at position p_1 in the reference image, while M_r predicts the color at position p_2). The color of p is a weighted average of these predictions.*

now define the distance between pixels p and q as

$$D(p,q) = \|Tf(p) - Tf(q)\|_2\,, \tag{1}$$

where $f(x)$ is the vector of DCT coefficients corresponding to the $k \times k$ neighborhood centered at x.

To summarize, in order to classify a pixel p in an input grayscale image I, we compute $f(p)$, transform the resulting vector using T, and apply the Knn classifier in the low-dimensional subspace. In order to accelerate the classification process we construct the feature space using only a randomly sampled subset of the labeled pixels in the training set. This reduces computation times considerably without introducing noticeable visual artifacts in the resulting colorization, as noted in [WAM02].

3.2. Image Space Voting

Although the Knn classifier described above is more robust than a naive nearest-neighbor classifier, there can still be many misclassified pixels, as demonstrated in figure 7(b): quite a few pixels inside the body of the cheetah are classified as belonging to the background, and vice versa. A better classification may be produced by explicitly encouraging a more spatially coherent labeling.

Consider $N(p)$, the $k \times k$ neighborhood around a pixel p in the input image. This neighborhood might contain differently labeled pixels; in fact p might be surrounded by pixels with a different label. To rectify such situations, we would like to apply something like the median filter, which is commonly used for noise removal. However, there is no order relation among the different labels, so we use the following approach instead: we replace the label of p with the *dominant label* in $N(p)$. The dominant label is the label with the highest confidence $\mathrm{conf}(p,\ell)$, where the confidence is defined as

$$\mathrm{conf}(p,\ell) = \frac{\sum_{q \in N(p,\ell)} W_q}{\sum_{r \in N(p)} W_r}. \tag{2}$$

Here $N(p,\ell)$ is the set of pixels in $N(p)$ with the label ℓ, and the weights W_q depend on the distance $D(q, M_q)$, between the pixel q and its best match M_q. M_q is the nearest neighbor of q in the feature space, which has the same label as q. Specifically,

$$W_q = \frac{\exp(-D(q,M_q))}{\sum_{r \in N(q)} \exp(-D(r,M_r))}. \tag{3}$$

The confidence $\mathrm{conf}(p,\ell)$ is typically high in neighborhoods where all (or most) pixels are labeled ℓ, and low on boundaries between regions, or in other difficult to classify spots.

In essence, the filtering operation described above is a weighted vote over the pixels in $N(p)$. Figure 7(c) shows how this image space voting improves the spatial coherence of the resulting classification.

3.3. Color Transfer and Optimization

At this point we are ready to define how color is transferred from the example color image L to an input grayscale image I. We work in the YUV color space, where Y is the monochromatic luminance channel, which we use to perform the classification, while U and V are the chrominance channels. The choice in YUV color space is for consistency with Levin *et al.* though there might be better color spaces for colorization. Let $C(p)$ denote the chrominance coordinates of a pixel p. After classifying each pixel $p \in I$ as described above, the color of p (with label ℓ) is given by the weighted average

$$C(p) = \sum_{q \in N(p,\ell)} W_q C(M_q(p)). \tag{4}$$

As defined above, M_q denotes the best match of $q \in I$ in the example image L, and $M_q(p)$ denotes the pixel in L whose position with respect to M_q is the same as the position of p with respect to q (see figure 6). In other words, we examine all of the pixels in $N(p,\ell)$, each of which has a matching neighborhood in L that "predicts" a different color for p, and compute a weighted average of these predictions.

Transferring color in this manner produces a colorized result, but since some areas might still be misclassified, the colorization will be wrong in such areas. Figure 7(c) shows such a colorization, and it can be seen that several regions inside the body of the cheetah have been assigned a green color. To improve the colorization, we transfer color only to pixels whose confidence in their label is sufficiently large, $\mathrm{conf}(p,\ell) > 0.5$, and provide the colored pixels as constraints to the optimization-based color interpolation scheme of Levin *et al.* [LLW04]. Thus, our classification and color transfer stages may be viewed as a method for automatic generation of color "micro-scribbles".

As explained in [LLW04] the optimization-based interpolation is based on the principle that neighboring pixels with similar luminance should have similar colors. Thus, the interpolant attempts to minimize the difference between the color assigned to a pixel p and the weighted average of the

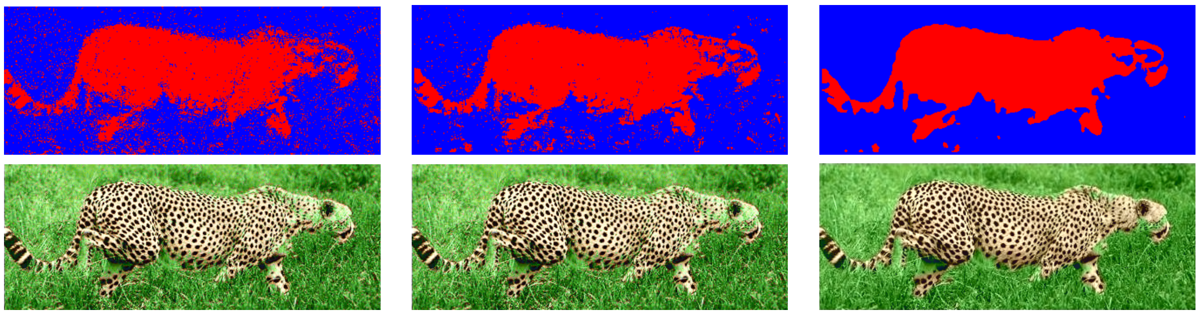
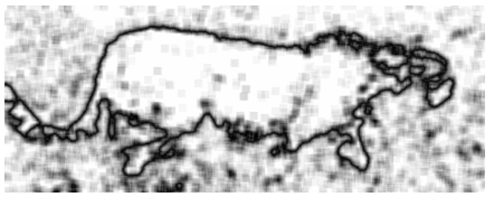
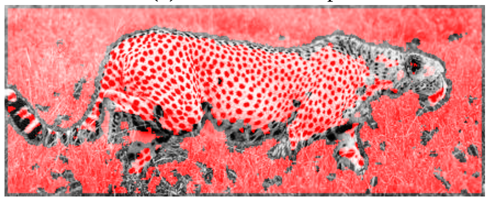
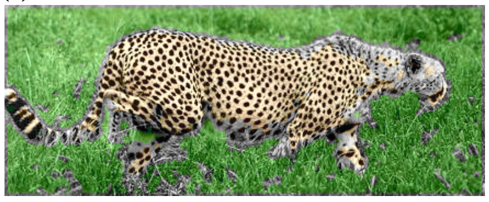

(a) Naive nearest neighbor (b) Voting in feature space (c) Voting in image space

Figure 7: *A visualization of the classifications and the resulting colorizations corresponding to different classifiers applied to a grayscale image of a walking cheetah. The training set for these classifications is shown in figure 3.*


(a) Confidence map


(b) Pixels with confidence above 0.5 marked in red


(c) Micro-scribbles

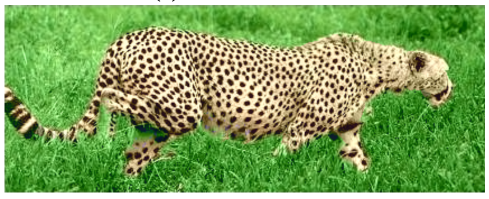

(d) Final result after optimization

Figure 8: *Generating automatic scribbles: pixels with confidence above a predefined threshold are provided as input to the optimization stage.*

colors of its neighbors, where the weights are determined by the similarity of their luminance. Formally, one seeks the minimum of $J(C)$, where

$$J(C) = \sum_{p \in I} \left(C(p) - \sum_{q \in N(p)} w_{pq} C(q) \right)^2 , \tag{5}$$

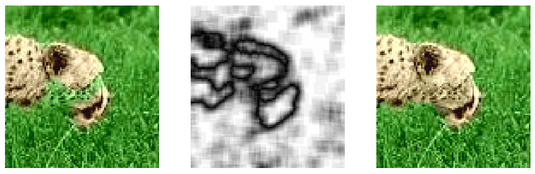

(a) (b) (c)

Figure 9: *Closeup of the head: (a) before optimization, (b) confidence map, (c) final result after optimization.*

and

$$w_{pq} \propto e^{-(Y(p)-Y(q))^2/2\sigma_p{}^2} \tag{6}$$

subject to the input constraints. The reader is referred to [LLW04] for further details.

Figure 8 shows a visualization of the pixel confidences in the cheetah image, and the automatic micro-scribbles that we give as an input to the optimizer. The result of the optimization is shown in figure 8(d) and a closeup of the cheetah's head is shown in figure 9. Note the improvement in the colorization of the head.

4. Results

All of the results shown throughout this paper were obtained using 7×7 neighborhoods, so our initial feature space has 49 dimensions, corresponding to the DCT coefficients. The classifier described in section 3.1 is built by sampling 500 intra-difference vectors and 500 inter-difference vectors (except for figure 10, where we used 200 samples), projected to form a feature subspace of 10 dimensions. Pixels with a confidence value above 0.5 were provided as micro-scribbles to the optimization stage.

In figure 1 we compare our results with the results achieved by automatic pixelwise color transfer [WAM02]. In Figures 4 and 7 we further show that naive classifiers alone do not yield satisfactory results and that our feature space analysis and image space voting greatly improve classification (and hence colorization) quality.

We also compare our method to the user-assisted method

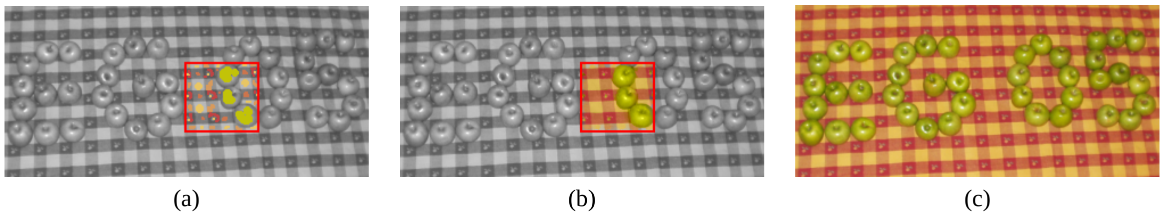


Figure 10: *Colorization without a reference image: (a) A small region in the input image is annotated with scribbles. (b) The region is colorized using the method of Levin et al. (c) The automatic colorization of the entire image using our method.*

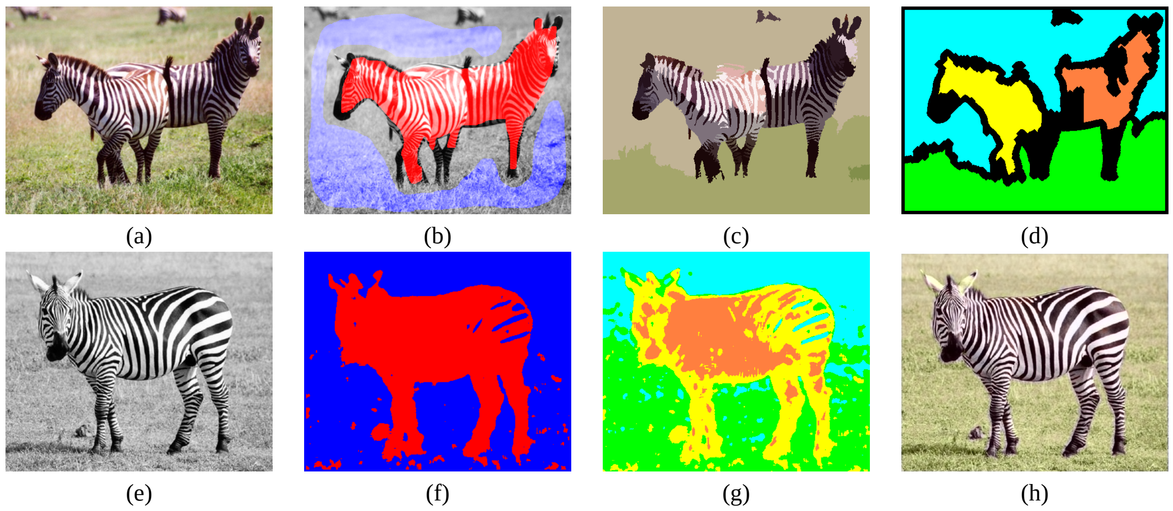


Figure 11: *Automatic region marking reduces human effort, while producing very similar results. (a) Reference image. (b) Manually marked regions: zebra (marked in red) and grass (marked in blue). (c) Automatic (mean shift) segmentation. (d) Regions obtained by merging and shrinking segments in (c). Zebra regions are now marked in orange and yellow, while grass regions are in green and cyan. (e) A new grayscale zebra image. (f) Classification using the manually marked regions in (b). (g) Classification using the automatic regions in (d). The union of the orange and yellow pixels is nearly identical to the red pixels in (f), meaning that both classification agree on which pixels best match the zebra region in the reference image, and produce the same colorization (h).*

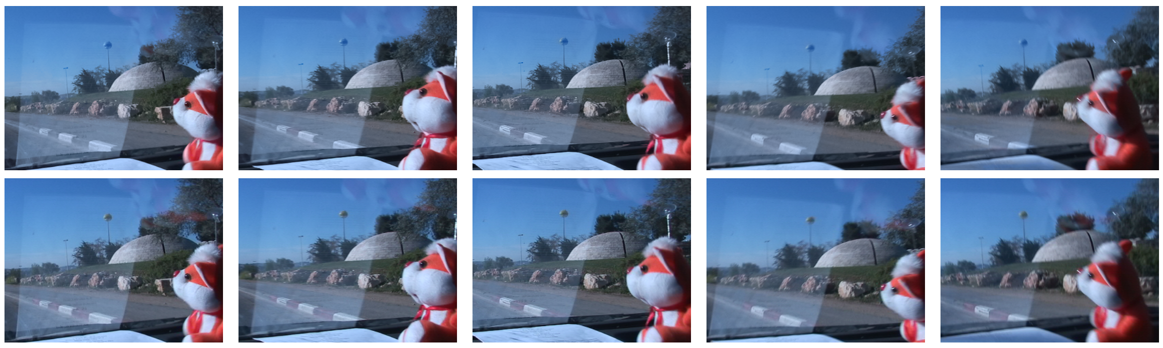


Figure 12: *Top: sample frames from a video sequence colorized by our method given a single colored frame as input. Levin et al. had to manually scribble in 9 frames out of the 33. Bottom: the corresponding frames from the original color video sequence, shown here as ground truth reference.*

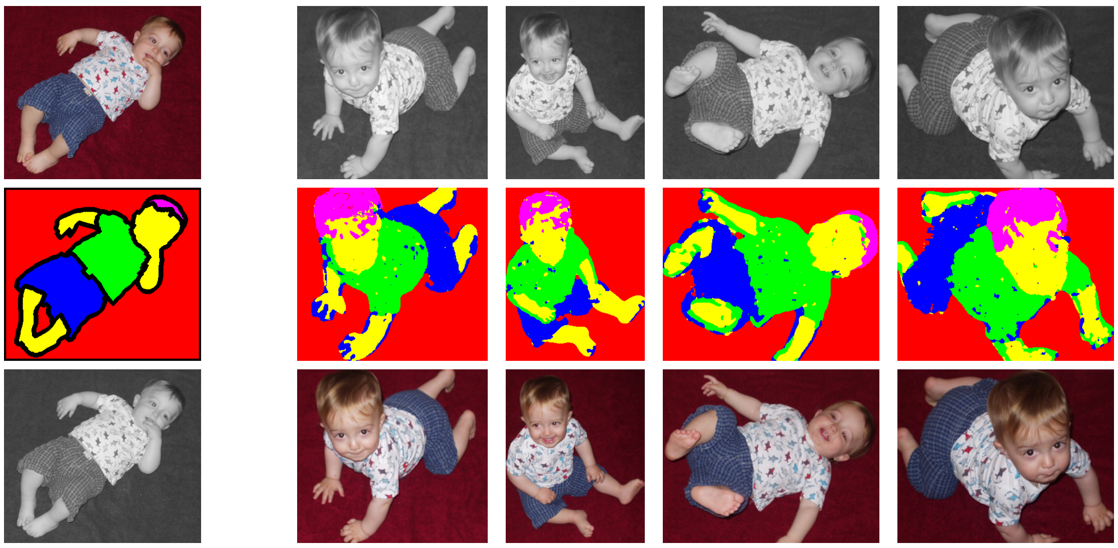


Figure 13: *Colorization of a series of grayscale images from a single example. The left column shows the reference image, its collection of (automatically generated) regions, and its luminance channel. On the right, the top row shows the input grayscale images; the middle row shows corresponding classifications; and the bottom row shows the resulting colorizations.*

of Levin *et al.* [LLW04] in figure 2. The advantage of their technique is that it does not require a reference image. Indeed, a suitable reference image might not be available. In such cases, it is sometimes sufficient to manually colorize a small part of the image using Levin's method, and then complete the colorization automatically, using that small part as the reference. This is demonstrated in figure 10, where the user colorized a few apples and a small portion of the tablecloth using scribbles (as in [LLW04]) and indicated the corresponding regions, while our method finished the job. Thus, the user-assisted workflow scenario of Levin *et al.* can still benefit from the use of our method. The converse is also true. In some cases our method produces an erroneous colorization, typically due to a mistake in classification (for example, some of the thin reeds in figure 2 take their color from the water or the rocks in their background). In cases such as these, the user may intervene and fix such mistakes by adding a few scribbles and repeating the optimization stage.

In most of our examples we used automatic segmentation to identify the different regions in the reference image, except in figure 2, where the regions were marked manually. We use the EDISON code [RIU] to perform meanshift segmentation, with its parameters adjusted to yield large segments. We also applied morphological operators to separate the segments (see figures 1, 4 and 11: the gaps between segments are shown in black). In figure 11 we show a comparison between manual and automatic segmentation of the reference image. As demonstrated in the figure, the differences in the classification are very small, implying that the method is not sensitive to the segmentation.

A significant advantage of our method lies in its ability to automatically colorize an entire series of grayscale images. This is demonstrated in figure 12, which shows several frames from a video sequence that was colorized by out method using a single colored frame as the input example. To guarantee temporal coherence in the colorized video, the automatically generated micro-scribbles were fed into the spatio-temporal volume of the video and optimization was then applied in three-dimensions. For comparison, Levin *et al.* reported scribbling in 9 out of 33 frames to colorize the same sequence.

An even more challenging example of sequence colorization is shown in 13. Note that the grayscale input images are similar but lack any frame coherence, so propagating colors from one image to another is not an option here. The variation among the images in figure 13 are larger than of a typical video stream. Our method avoids any use of tracking and greatly simplifies the work of the user. Of course, the results are only valid as long as the colored example is valid: Once objects that do not appear in the reference image are introduced, the user has to either color them, or provide another adequate reference.

5. Conclusion and Future Work

In this paper we presented a new technique for colorizing grayscale images by example. The method uses a classifier defined by a DCT analysis and custom-tailored to the given segmented reference image. The classification of pixels in the input grayscale image aims at creating a spatially coherent labeling by employing voting both in feature space and in image space. To further improve the transfer of the chromatic channels from the example to the grayscale image only pixels with high confidence are provided as automatic micro-scribbles to a global optimization stage, which yields the final colorization. Our method shows significant improvement over techniques that use only pixelwise decisions and naive classifications.

In the future we plan to investigate the possibility of combining our classification-based approach with automatically established swatches. This may be done by searching for representative regions in the input images that match reference regions particularly well. Once such representatives have been established, the remaining input pixels might be able to find better matches in a transitive fashion, through one of these representatives. The premise here, as in [WAM02], is that it is generally easier to find good matches between neighborhoods in the same image than across different images.

We would also like to explore other, more sophisticated, monochrome texture descriptors, such as the Gabor transform, steerable pyramids, and other wavelet-related transforms. By relying on descriptors that constitute a better model of the human visual system, we hope to be able to further improve the classification and matching capabilities of our approach.

Acknowledgements

The authors would like to thank Anat Levin for sharing her code with us. This research was supported in part by the Israel Science Foundation and by the Israeli Ministry of Science and Technology.

References

[BHK97] BELHUMEOUR P. N., HESPANHA J. P., KRIEGMAN D. J.: Eigenfaces vs. Fisherfaces: recognition using class specific linear projection. *IEEE Trans PAMI 19*, 7 (July 1997), 711–720. 4

[Bur] BURNS G.: Colorization. Museum of Broadcast Communications: Encyclopedia of Television, http://www.museum.tv/archives/etv/index.html. 3

[CWSM04] CHEN T., WANG Y., SCHILLINGS V., MEINEL C.: Grayscale image matting and colorization. *Proc. ACCV* (2004), 1164–1169. 3

[DHS00] DUDA R. O., HART P. E., STORK D. G.: *Pattern Classification*, 2nd ed. Wiley Interscience, New York, 2000. 4

[Fis36] FISHER R.: The use of multiple measures in taxonomic problems. *Ann. Eugenics 7* (1936), 179–188. 4

[GRHS04] GOLDBERGER J., ROWEIS S., HINTON G., SALAKHUTDINOV R.: Neighbourhood component analysis. *Neural Information Processing Systems 17* (2004). 4

[GW87] GONZALEZ R., WINTZ P.: *Digital Image Processing*. Addison-Wesley, 1987. 4

[HB03] HERMES L., BUHMANN J.: Semi-supervised image segmentation by parametric distributional clustering. *LNCS Energy Minimization Methods in Computer Vision and Pattern Recognition* (2003), 229–245. 4

[HH97] HANSEN M., HIGGINS W.: Relaxation methods for supervised image segmentation. *IEEE Trans PAMI 19* (1997), 949–962. 4

[HJO*01] HERTZMANN A., JACOBS C. E., OLIVER N., CURLESS B., SALESIN D. H.: Image analogies. In *Proc. SIGGRAPH 2001* (Aug. 2001), Computer Graphics Proceedings, Annual Conference Series, pp. 327–340. 1

[HS89] HSIAO J., SAWCHUK A.: Supervised textured image segmentation using feature smoothing and probabilistic relaxation techniques. *IEEE Trans PAMI 11* (1989), 1279–1292. 4

[JSTS04] JIA J., SUN J., TANG C.-K., SHUM H.-Y.: Bayesian correction of image intensity with spatial consideration. In *Proc. ECCV* (2004), pp. 342–354. 3

[LLW04] LEVIN A., LISCHINSKI D., WEISS Y.: Colorization using optimization. *ACM Transactions on Graphics 23*, 3 (2004), 689–694. 1, 3, 6, 7, 9

[MH87] MARKLE W., HUNT B.: Coloring a black and white signal using motion detection. Canadian patent no. 1291260, Dec. 1987. 3

[Neu03] NEURALTEK: BlackMagic photo colorization software, version 2.8. *http://www.timebrush.com/blackmagic* (2003). 3

[PD02] PARAGIOS N., DERICHE R.: Geodesic active regions and level set methods for supervised texture segmentation. *International Journal of Computer Vision* (2002), 223–247. 4

[Pra91] PRATT W. K.: *Digital Image Processing*. John Wiley & Sons, 1991. 4

[RAGS01] REINHARD E., ASHIKHMIN M., GOOCH B., SHIRLEY P.: Color transfer between images. *IEEE Computer Graphics and Applications* (September/October 2001), 34–40. 3

[RIU] RIUL: Edison. RIUL, Center for Advanced Information Processing, Rutgers University, http://www.caip.rutgers.edu/riul/research/code/EDISON. 9

[Sap04] SAPIRO G.: Inpainting the colors. *IMA Preprint Series, Institute for Mathematics and its Applications University of Minnesota*, 1979 (2004). 1

[SBZ04] SYKORA D., BURIANEK J., ZARA J.: Unsupervised colorization of black-and-white cartoons. In *Proc. NPAR* (2004), pp. 121–127. 3

[SHWP02] SHENTAL N., HERTZ T., WEINSHALL D., PAVEL M.: Adjustment learning and relevant component analysis. In *Proc. ECCV* (2002), pp. 776–792. 4

[Sil98] SILBERG J.: The Pleasantville post production team that focussed on the absence of color. Cinesite Press Article, http://www.cinesite.com/core/press/articles/1998/10_00_98-team.html, 1998. 3

[WAM02] WELSH T., ASHIKHMIN M., MUELLER K.: Transferring color to greyscale images. *ACM Transactions on Graphics 21*, 3 (July 2002), 277–280. 1, 2, 3, 6, 7, 10

[Wei99] WEISS Y.: Segmentation using eigenvectors: A unifying view. In *Proc. ICCV* (1999), pp. 975–982. 4

[YS04] YATZIV L., SAPIRO G.: Fast image and video colorization using chrominance blending. *IMA Preprint Series, Institute for Mathematics and its Applications University of Minnesota*, 2010 (2004). 1