Vis Comput (2010) 26: 1349–1360
DOI 10.1007/s00371-009-0412-7

ORIGINAL ARTICLE

Color-to-gray conversion using ISOMAP

**Ming Cui · Jiuxiang Hu · Anshuman Razdan ·
Peter Wonka**

Published online: 17 December 2009

Abstract In this paper we present a new algorithm to transform an RGB color image to a grayscale image. We propose using nonlinear dimension reduction techniques to map higher dimensional color vectors to lower dimensional ones. This approach generalizes the gradient domain manipulation for high dimensional images. Our experiments show that the proposed algorithm generates competitive results and reaches a good compromise between quality and speed.

Keywords ISOMAP · Color to gray · Color space

1 Introduction

In this paper we investigate how ISOMAP [35], a non-manifold learning technique, can be used for color-image processing. We also present an ISOMAP-based framework to map a higher dimensional image to a lower dimensional image, e.g. map a color RGB image to a grayscale image. The problem can be formulated as follows. An $m \times n$ multi-channel image can be seen as a higher dimensional tensor $I_D \in R^{m\times n\times D}$, where each of the mn pixels corresponds to a color vector C_i with D spectral samples. As output of this algorithm we want to map this image to a lower dimensional display range, i.e. a tensor $I_d \in R^{m\times n\times d}$ with all entries constrained to lie between 0 and 1 and $d < D$. In this paper we will consider $D = 3$ and $d = 1$ and map the input image to gray-scale.

We set two goals: (1) The color distances from the input color space can be controlled by the user in the output color space. (2) The algorithm should make use of the dynamic range of the display device to show details in the image. We will propose an elegant solution that combines both goals in a unified framework.

We are interested in applying manifold learning techniques to the problem at hand. This gives two interesting results: (1) a new operator for image processing; (2) the mapping quality and speed.

One popular technique in the image processing community is to extract the gradient field and then manipulate it to a desirable target. In the end, the target images are reconstructed from the target gradient field [16]. When dealing with multidimensional input, the calculation of the gradient becomes controversial. Our manifold learning approach generalizes the idea to multidimensional data: we first get the matrix of the pairwise distances for the input pixels and then manipulate the distances in the matrix. In the end, we reconstruct an output that preserves the manipulated distances.

The quality and speed performances of our algorithm are compared to several recent approaches, published by Gooch et al. [17], Rasche et al. [27], Grundland et al. [18] and Smith et al. [33]. Approaches of Gooch et al. [17] and Rasche et al. [27] are computationally slow. The algorithm by Gooch et al. [17] does not allow higher dimensions because it is intrinsically linked to the $L^*a^*b^*$ color space, and the algorithm by Rasche et al. [27] does not scale well to a higher number of spectral samples in an image. By contrast, our solution computes a nonlinear mapping by using a linear operator in a sub-manifold of the higher dimensional color space. This approach gives similar visual quality as well as improves computation times and can extend to higher dimensions. Our algorithm is slower than a fixed global mapping, e.g. of Smith et al. [33]. While such a simple operator can get great results on a large number of images, it is easy

M. Cui (✉) · J. Hu · A. Razdan · P. Wonka
Arizona State University, Phoenix, AZ, USA
e-mail: ming.cui@asu.edu

to show that a fixed global mapping can eliminate arbitrarily large features.

Our major contributions are as follows:

- We are the first to apply nonlinear manifold learning to the color-to-gray conversion problem. Our algorithm gives competitive results compared to state-of-the-art algorithms.
- In our RGB to gray mapping algorithm we propose a new way of nonlinearly adjusting the contrast by a single parameter.

One major design decision is if the mapping should be global or local. While most recent tone mapping algorithms favor a local mapping, Rasche et al. [27] argue that a global mapping is important to avoid artifacts when it comes to mapping higher dimensional color vectors to lower dimensional ones. It is worth mentioning that the default implementation of Rasche et al. [27] compares every pixel to every other pixel when minimizing the objective function. The authors also suggested an alternative implementation by limiting the comparison to only a small spatial neighborhood for each pixel. This will accelerate their algorithm. However, this also turns the algorithm into a local contrast enhancement operator since widely separated points in original space may be assigned to the same output intensity if surrounded by sufficiently different other color values. Smith et al. [33] also have a local edge sharpening step. In this paper we will present a global mapping algorithm. However, our algorithm can use a local mapping to increase contrast as a post-process. As other existing algorithms have the same option, we will not make a potential post-process a focal point of this paper.

2 Related work

There is a large number of techniques to convert a high dynamic range luminance image to a low dynamic range luminance image. These techniques are broken down into local and global methods. Global mappings ensure that identical color values are mapped to identical color values, so that each pixel in an image can be mapped separately [1, 37]. Local mappings are typically more complex and slower; however, they can adapt the mapping function locally to produce better results [13, 16, 22, 29]. As these methods have several advantages and disadvantages, recent work also focused on combining tone mapping operators [23] and evaluation of tone mapping [21, 26].

In recent years, transforming a color image into a grayscale image attracted the interest of several researchers [4, 14, 17, 18, 24, 27, 28, 34, 36]. The problem is to find a lower dimension embedding of the original data that can best preserve the contrast between the data points in original data. These papers are strongly related to our work and we compare our results against two of them in this paper. The main difficulty of previous works is that they use complex and slow nonlinear optimization algorithms. We believe that this is too complex for the problem at hand. In contrast, we want to follow the strategy of manifold learning and first detect a sub-manifold in higher dimensional data before computing a mapping [5, 12, 25, 30, 31, 35]. It is worth mentioning that recently two other accelerated methods were proposed and they reported a very good quality. Grundland et al. [18] make use of predominant component analysis and accelerate with Gaussian pair sampling. Smith et al. [33] first use a fast global mapping and then use a local edge sharpening technique based on the Laplacian pyramid. We also compare our results to theirs in this paper.

The second similar problem is multispectral and hyperspectral image visualization. Traditionally, these images have been visualized as a cube with a suite of interactive tools [32]. One set of tools allows to extract one spectral band at the time or cycle through spectral bands as an animation. To create RGB images, interactive tools can be used to specify red, green, and blue values as linear combinations of spectral bands. That means an RGB value is computed by a matrix vector multiplication. Along these lines several authors suggest methods how to automatically create linear combinations of spectral bands to define the green, red, and blue color channels of visualization [8, 20, 38, 39]. In this paper we compare our results to two such methods, Jacobson et al.'s [20] and visualization based on PCA [38]. Recent investigation suggest that nonlinearity exists in hyperspectral data [19]. Actually ISOMAP has been adopted for hyperspectral image visualization in [2, 3]. We believe it is interesting to extend their work to color-to-gray image conversion. A faster visualization strategy for hyperspectral visualization was proposed by Cui et al. [9], but their method cannot be directly applied to the color-to-grayscale problem.

There is a larger number of general dimensionality reduction algorithms in the literature. Prominent examples are ISOMAP [35], Local Linear Embedding (LLE) [30], Laplacian Eigenmap Embedding [5]. ISOMAP is a special version of multidimensional scaling, which uses geodesic distance instead of Euclidean distances between the points. LLE tries to preserve the local linear structure of the original point set and casts it as an eigenvalue problem. Laplacian Eigenmap Embedding formulates the problem as a spectral graph cut and also solves it as an eigenvalue problem. In recent years, more advanced versions of manifold learning algorithms were proposed. These include Hessian Eigenmap Embedding [12], Conformal Maps [31], and Diffusion Maps [25]. These methods are usually computationally more expensive.

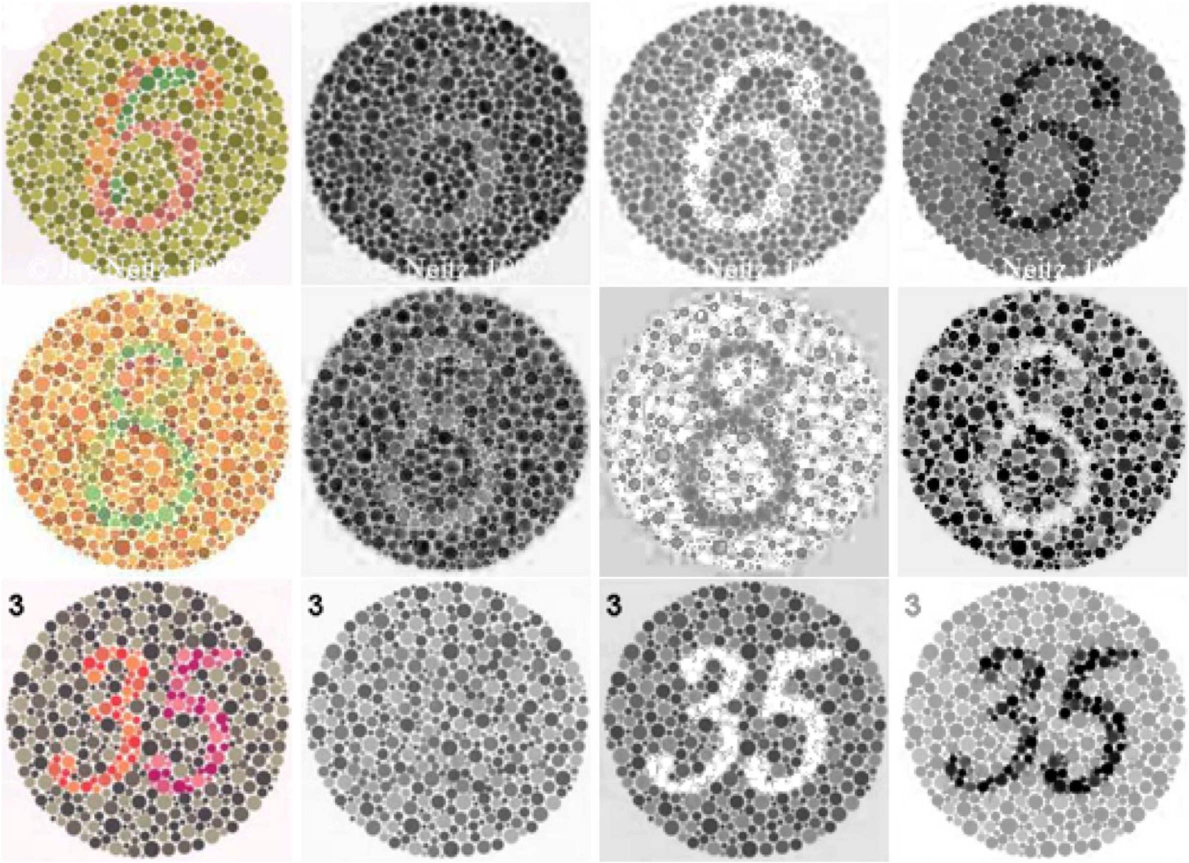


Fig. 1 From *left* to *right*: original image, PCA mapping, color2gray mapping, and our result

3 Overview

3.1 Algorithm goals

We formally state the problem as follows. The input to the algorithm is an $m \times n$ image as tensor $I_D \in R^{m \times n \times D}$, where each of the $N = mn$ pixels corresponds to a color vector C_i with D spectral samples. The output of this algorithm is a tensor $I_d \in R^{m \times n \times d}$, where each of the N pixels corresponds to a color vector c_i with d spectral samples and all entries are constrained to lie between 0 and 1. For color-to-gray conversion, $D = 3$ and $d = 1$. There is a one-to-one correspondence between a color vector (pixel) C_i and c_i.

Our first goal is to find a global mapping that preserves the pairwise distances between all input pixels. This goal can be formalized as finding a mapping that minimizes E:

$$E = \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \big(\|c_i - c_j\| - dist(C_i, C_j) \big)^2. \tag{1}$$

E can also be presented in a matrix form:

$$E = \frac{1}{2} \|M_c - M_C\|_F \tag{2}$$

where F denotes the Frobenius norm; this equation is in matrix form. M_c and M_C are both matrices. $M_c(i, j) = \|c_i - c_j\|$ and $M_C(i, j) = dist(C_i, C_j)$.

The second goal is making use of the dynamic range of the display to show image details. This goal is partially in conflict with the first goal and difficult to qualify in a formula, but we found a consistent way to integrate the second goal with the first one by modeling a distance function $dist(C_i, C_j)$ that provides some user control of the output. It is very important that we only make consistent modifications. For example, a local tone mapping operator can produce colorful images, but the original meaning of the input is not preserved. This can be very counterproductive for visualization, because pixels are no longer comparable. Similarly, a global operator, such as histogram equilization in all color bands, sometimes introduces artificial features that are not present in the data set.

3.2 Algorithm overview

An overview of our algorithm is shown in Fig. 2. It computes a nonlinear mapping. In general, a nonlinear mapping is much better at adapting to the structure of the data and it

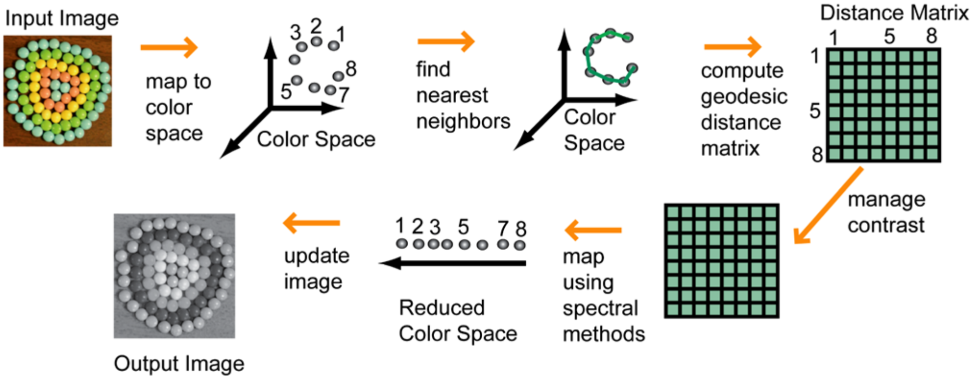


Fig. 2 Overview of our algorithm

was therefore also used in previous approaches. The algorithm includes the following stages.

Color space preprocessing We take an input image and consider each pixel as a higher dimensional color vector. This gives us a set of vectors in a higher dimensional color space. If the input image has RGB color vectors, we additionally map all pixels from RGB to $L^*a^*b^*$ color space.

Sub-manifold detection Find a sub-manifold in higher dimensional space, by computing geodesic rather then Euclidian distances. This stage includes finding nearest neighbors, computing a geodesic distance matrix, and managing the contrast by transforming the matrix. The output of this stage is a distance matrix defining pairwise distances between all pairs C_i and C_j.

Optimized mapping Find an optimized mapping from higher to lower dimensional color vectors. At this stage, each color vector C_i is mapped to a lower dimensional color vector c_i based on a matrix decomposition. This operation is very fast and finds a global optimum.

Color space postprocessing The color mapping can be used to construct a lower dimensional image I_d. Postprocessing can include local (a gradient domain Poisson solver [16]) or global (histogram equalization) tone mapping operators.

Acceleration strategy While the above algorithm steps define a working algorithm, we need to accelerate the algorithm and reduce memory consumption by using a subsampling strategy. The main idea is to sub-sample the rows of the matrix D_C.

4 An introduction to ISOMAP

In this section we give a brief introduction to ISOMAP, a very successful strategy for manifold learning that was proposed by Tenenbaum et al. [35]. ISOMAP in essence is a special version of the classical multidimensional scaling (classical MDS) algorithm [6].

4.1 Classical MDS algorithm

Classical MDS [6] provides a solution for (2). Since a global optimum cannot be found for (2), classical MDS does not minimize the F-norm of the difference matrix $M_c - M_C$ in (2) directly. Instead, it minimizes the difference of two transformed matrices. The transform first computes an element-wise square of a matrix and then centers it. The centering operator τ for a matrix M can be computed by $\tau(M) = -HMH/2$, and $H = I - 1/N * O$ with O being a matrix of all ones. If we denote the element-wise square of M_c and M_C as M_c^2 and M_C^2, respectively, we can express the objective of the transformed minimization problem as

$$E = \left\| \tau(M_c^2) - \tau(M_C^2) \right\|_F. \tag{3}$$

Geometrically, we are now minimizing the pairwise angular distances instead of the pairwise Euclidean distances. The benefit we gain from this transform is that the global optimum of (3) can be computed in closed form. Let us denote $\lambda_1, \lambda_2, \ldots, \lambda_d$ as the largest d eigenvalues of matrix $\tau(M_C^2)$ and $v_1, v_2, \ldots, v_d$ as their corresponding eigenvectors. Then the d-dimensional output c_i is computed as [11]

$$c_i = \begin{bmatrix} \sqrt{\lambda_1} \cdot v_{1i} \\ \sqrt{\lambda_2} \cdot v_{2i} \\ \vdots \\ \sqrt{\lambda_d} \cdot v_{di} \end{bmatrix}. \tag{4}$$

4.2 The ISOMAP algorithm

In classical MDS, how to calculate $dist(C_i, C_j)$ is left for the user to decide. In the ISOMAP algorithm, Tenenbaum et al. proposed using the geodistance between the input data points for $dist(C_i, C_j)$: the input data set is treated as a graph. Each input point in the original D-dimensional space is a node in the graph and is connected to its k nearest neighbors (k is a parameter provided by the user). The distance between two points C_i and C_j is calculated as the shortest path between the two corresponding nodes in the graph.

The ISOMAP algorithm has two computational bottlenecks [10]. First, for a graph that has N nodes, it takes $(O(N^2 \log N))$ to find all pairwise shortest paths in the matrix D_C using Dijkstra's algorithm. Second, we need to solve the eigenvalue problem for the $N \times N$ matrix $\tau(M_C)$, which takes $O(N^3)$. The overall complexity is $O(N^3)$. For a medium size image with size 300 by 300, the total number of pixels, which is N, equals 90000 and the algorithm is very slow.

4.3 The landmark ISOMAP algorithm

In [11], an accelerated version of ISOMAP is proposed, which is called landmark ISOMAP. The new algorithm starts by selecting only a small fraction of the whole input point set called landmark points $L_1, L_2, \dots, L_n$. If we denote n as the number of landmark points then usually we pick $n = \lceil N * 0.02 \rceil$. The original ISOMAP algorithm is run on the n landmark points to get a skeleton for the output in the d-dimensional output space. The rest of the output is embedded into the skeleton by projecting to the first d principle axes of the landmark points. We denote the squared pairwise distance matrix for the landmark points as L_C^2 and λ_i, and v_i as the eigenvalues and eigenvectors of $\tau(L_C^2)$, as before. We further define δ_i as a column vector of the squared distances from C_i to all the landmark points and δ_L to be the mean of all the column vectors of L_C^2. Now we can express the jth component of c_i as

$$c_{ij} = -\frac{1}{2} \frac{v_j}{\sqrt{\lambda_j}} (\delta_i - \delta_L). \tag{5}$$

Since only the pairwise distances between the landmarks and the remaining points are needed for the interpolation, the cost for Dijkstra's algorithm is reduced to $O(nN \log N)$. The ISOMAP algorithm on landmark points requires $O(n^3)$. Since $n \ll N$, the overall complexity is reduced to $O(N \log N)$.

5 Algorithm details

In the following we explain in detail how the landmark ISOMAP algorithm is used for our color-to-gray mapping problem.

5.1 Color space preprocessing

Given an RGB image with m rows and n columns, we can interpret each pixel as a color vector C_i in a D-dimensional color space yielding $N = mn$ color vectors. If we are working with RGB images as input, we transform each of the vectors C_i into $CIEL^*a^*b^*$ color space. This transformation is useful, because the Euclidean distance between the pixels in $L^*a^*b^*$ is similar to the visual difference perceived by human eyes. The L^* coordinate refers to the luminance and the a^* and b^* coordinates describe the position of the pixel in a two-dimensional chromatic space. The positive a^*-axis points towards red and the negative a^*-axis points towards green. Similarly, the positive b^*-axis points towards yellow and the negative b^*-axis points towards blue. As the color space preprocessing is optional, we use the same variables C_i to denote the input and the output of this stage. $L^*a^*b^*$ color space preprocessing only works when the input is an RGB image. Figure 3 shows an example of color space. For multispectral images it is important to drop the spectral bands that are destroyed by atmospheric water vapor. Additionally, we provide the option to scale individual spectral bands (we did not use this feature in our results).

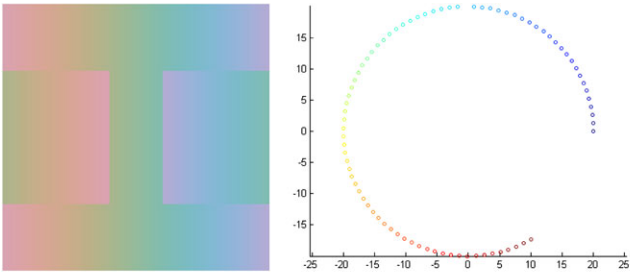


Fig. 3 *Left*: input image. Right: color distribution in $L * a * b$ color space. This figure shows how an image is mapped to $CIEL^*a^*b^*$ color space. The shown color distribution will require a nonlinear mapping to ensure a meaningful conversion to grayscale. Please note that no line can be found in color space so that a color projection onto the line results in a useful grayscale conversion

5.2 Sub-manifold detection

This part of the algorithm computes pairwise distances between two color vectors C_i and C_j. There are several choices for a distance function $dist(C_i, C_j)$ from (1) and we want to propose a different one from previous work. For example, Rasche et al. [27] use the simple Euclidian distance function (in $CIEL^*a^*b^*$ color space) and reduce the problem to a multidimensional scaling problem that requires nonlinear majorization techniques to find a local optimal solution. Color2gray [17] defines a more complex distance function as a combination of the luminance difference and the difference of projection on a certain direction in the chromatic space. This distance function leads to a nonlinear optimization problem that the authors solve with a conjugate-gradient method.

Following the ISOMAP framework described in Sect. 4, we propose to use a transformed geodesic distance $dist(C_i, C_j) = f(geodesic(C_i, C_j))$ between two color vectors. The geodesic distance is computed in a manifold spanned by the color vectors and brings a significant advantage. The strength of this algorithm is that we can compute a *global* optimum directly with multidimensional scaling. However, please note that the geodesic distance computation provides a mapping of one color space to another that is still nonlinear and is able to adapt to the structure of the data.

The distance computation has three steps. First, we need to construct a graph by computing the k-nearest-neighbors (KNN) of each input point C_i and connect each point with its k nearest neighbors with an edge weighted by their mutual distances. In our experiments, k is usually set to 10 50. Second, we need to compute a geodesic distance matrix by computing shortest distances from all the landmark points L_i to the rest of all points C_i in the graph. Third, we can manipulate the geodesic distances to provide some user control to improve the contrast in the final image.

An implementation detail worth mentioning is that the graph constructed by the KNN algorithm might not be connected. For example, k input points can be tightly clustered and form a clique of the graph, so that they will be isolated from the rest of the point set. Since we need to know the distance matrix from the landmark points to all the points and the entries of the matrix cannot be infinity, we use a simple and fast solution to alleviate this problem. We find the dominant component in the graph that has the largest number of points. For any other component, we add an edge connecting the two closest points between this component and the dominant component.

5.3 Contrast management

Our contrast management is motivated by a strategy common to high dynamic range image tone mapping. The input image is usually first transformed to a new domain in which the manipulation becomes easier. Then the output image is recovered from the manipulated image in the new domain. In [16] the input is transformed into the gradient domain, then the gradient is manipulated and the output is recovered by solving a PDE. In [15] the input is transformed into the wavelet domain, then the wavelets coefficients are manipulated and the output is recovered using the inverse wavelet transform. Similarly, we transform the input into the pairwise distances domain, then we manipulate the distances, and the output is recovered by the ISOMAP algorithm.

The manipulation can be done by nonlinearly scaling the $geodesic(C_i, C_j)$ distance function with another function, f. This allows to enhance contrast, while still using a distance-preserving mapping. We chose a simple function $f(x) = x^{\lambda}$ that allows the user to control one parameter in the mapping. The output of our contrast management step is a distance matrix D_C with entries $x_{ij} = f(geodesic(C_i, C_j))$. This matrix can be used to directly project color vectors in a lower dimensional color space as described in the next subsection. We illustrate the power of the parameter λ using two examples shown in Figs. 4 and 5. Intuitively, the parameter can either enhance local contrast (small distances) and reduce global contrast (large distances), or enhance global contrast and reduce local contrast. Please note that the parameter controls geodesic distances.

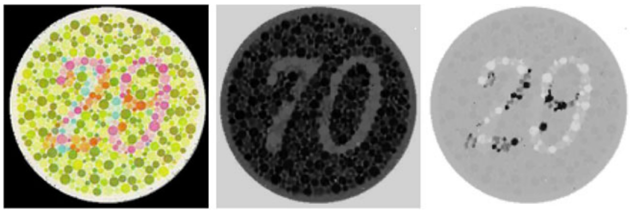


Fig. 4 *Left*: input image. *Middle*: a small $\lambda = 1$ reveals a 70. *Right*: a bigger $\lambda = 4$ reveals a 29

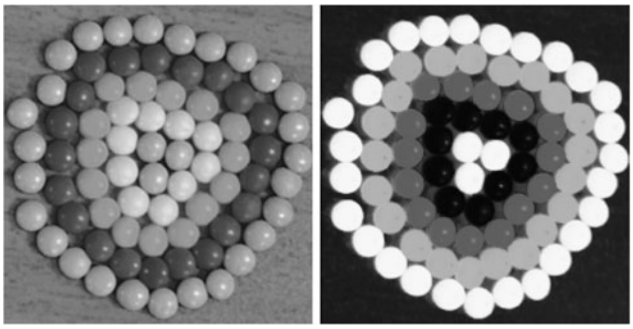


Fig. 5 Possible settings for the candy example from Fig. 11. *Left*: a small $\lambda = 0.8$ preserves the details of the image well and the spherical shape and the highlights of the candy are well visible. *Right*: a bigger $\lambda = 3$ reduces local contrast and the spheres look like flat circles

In Fig. 6 we demonstrate the usefulness of nonlinear mapping and geodesic distances. When we compare the results of PCA and our algorithm, we can see that PCA is unable to detect the nonlinear structure of the problem and does not show the two rectangles. While it is not reasonable to assume that the colors of a real-world example with thousands of colors are aligned in this way, this nonlinear structure problem would appear many times in subsections of the color space.

5.4 Landmark selection

In the original paper [11] the author proposes to select the matrix rows randomly. The author suggests that a more sophisticated clustering technique might have disadvantages because the clustering problem is domain-specific and the computational cost of clustering is high. While these arguments are reasonable for a general algorithm, our experiments show that clustering can improve the results of our algorithm. First, we found that this is fairly risky to make a random selection, as parts of the color space with few color vectors can receive no or insufficient samples. In graphics applications it might be interesting to ignore outliers, but for visualization applications outliers often convey important and meaningful information. Second, we need to build the kd-tree to accelerate the KNN algorithm so that we can reuse it for the computation of more uniformly distributed landmarks. To select landmarks we can set a parameter s that specifies the sub-sampling factor. We traverse the kd-tree depth first and stop the traversal if a node contains less than $s \times N$ points. We select the centroid of all the points contained in the kd-tree node as a landmark.

5.5 Color space postprocessing

After we get a set of color vectors c_i, we can assemble the low dimensional image. If the output is grayscale, we linearly map the range of all output intensity values to the range of [0, 1]. Additionally, we implemented histogram equalization and a gradient domain tone mapping algorithm [16] as potential post-process to enhance contrast. However, while we found that these algorithms can make certain images to look nicer, they make it hard to compare with other algorithms. First, these postprocessing operations are in conflict with the goals of the original algorithm and second, the comparison with other methods would be influenced. We also omit a comparison to Smith et al. [33]. They basically use an edge sharpening operator to achieve their results in difficult cases, but we consider this a potential post-process.

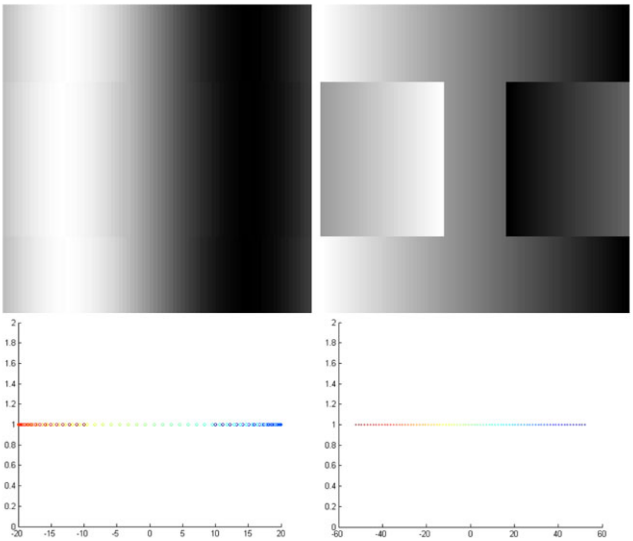


Fig. 6 *Top left*: the PCA result on the input image in Fig. 3. *Top right*: our result. *Bottom left*: we show how the original colors are projected to 1D for the PCA result. *Bottom right*: we show how the original colors are projected to 1D for our result

6 Results

6.1 Visual comparison

The authors in [7] gave an exhaustive comparison of the state-of-the-art color-to-gray algorithms. We run our algorithm through the same set of test images with a fixed parameter $\lambda = 0.4$. The number of nearest neighbors k is fixed to 50. There are more complicated algorithms to dynamically decide k according to the neighborhood of a particular point, but we opted not to use these computationally more expensive approaches. In the results, we found that with the help of λ we are able to enhance the details in the original image in a novel way that no other algorithms can achieve. In Figs. 7–10 we show some examples. In each case we picked the best two algorithms ranked in [7] and compare to our results. In Fig. 7, it can be observed that our result can distinguish the color difference between the two antennas. Also, the body colors are better stratified than in the other two. The outlines in the upper parts of the wings are also the clearest among the three. In Fig. 8, we can see that in the middle right part of the tree, there are three blocks of red leaves. Our method can highlight these three red blocks. In Fig. 9, both other two methods mapped the red petals to similar colors as the green leaves. In contrast, our method mapped the red petals to the brightest color in the whole image, which is satisfactory. In Fig. 10, at the bottom left of the original images, there are some faint circles that only a careful observe can find. All other methods neglected these circles. In contrast, our algorithm highlights these circles.

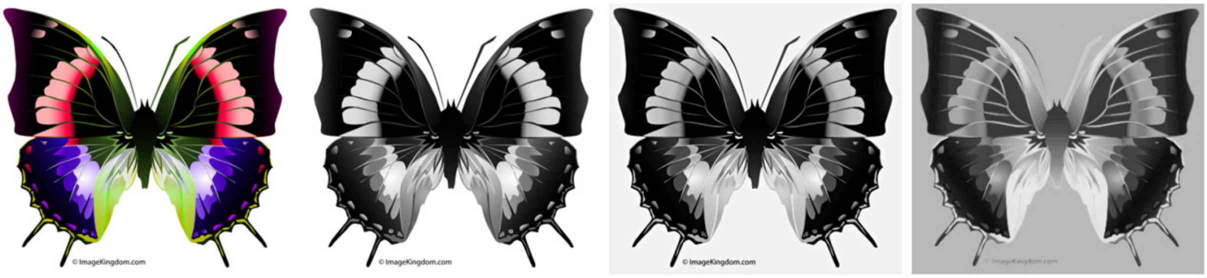


Fig. 7 From *left* to *right*: original color image; decolorize [18]; smith08 [33]; ours, $\lambda = 0.4$

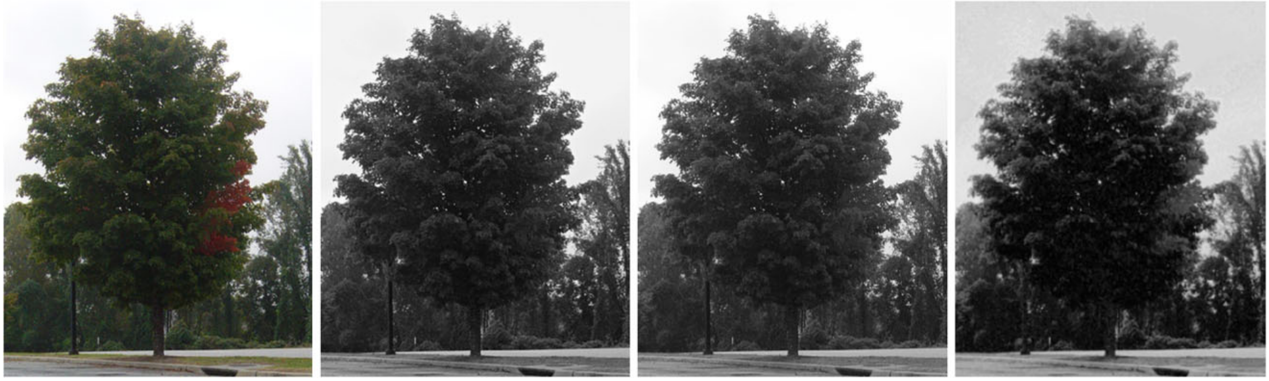


Fig. 8 From *left* to *right*: original color image; smith08 [33]; color2gray [17]; ours, $\lambda = 0.4$

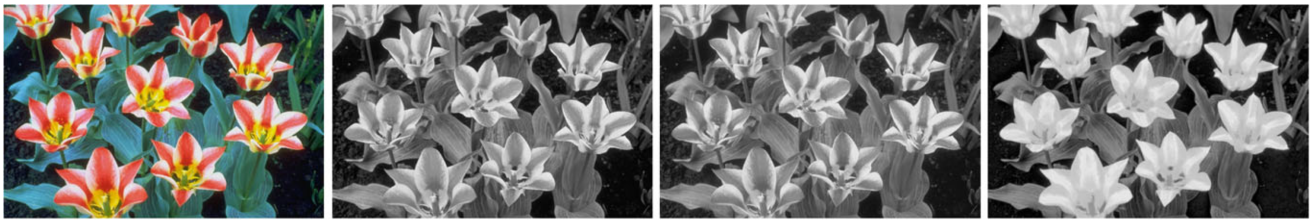


Fig. 9 From *left* to *right*: original color image; smith08 [33]; CIEY [14]; ours, $\lambda = 0.4$

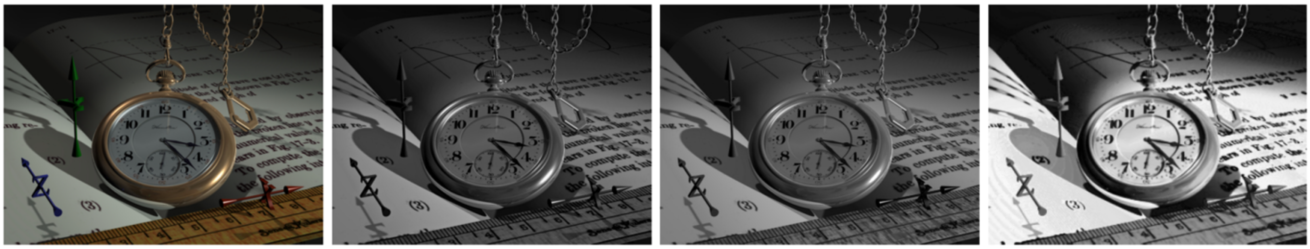


Fig. 10 From *left* to *right*: original color image; smith08 [33]; decolorize [18]; ours, $\lambda = 0.4$

This might be helpful if these circles are real features. At the same time, we also want to point out that the watch face and the printed page seem to be too bright. We might also enhance noise and artifacts instead of real features that are desirable. Results on other test images can be found in the supplementary material to the paper. It is worth noting that we are not doing an objective user study for the results like [7]. Therefore, we are not claiming that our results are always better than the other algorithms. We emphasize that our algorithm can highlight some features of the original images that other algorithms might omit. It is not always the case that our results are the most satisfactory. For example, in Fig. 14, our method does not distinguish the orange-highlighted lines and the pink-highlighted lines very well. More comparisons are shown in the supplementary material. In some cases, our results show salt-and-pepper noise artifacts. We find that the artifacts depend on the parameter k in the k-nearest-neighbors search. We will further discuss this in Sect. 6.3.

6.2 Timing comparison

We implemented our algorithm in Matlab on a 3.6 GHz Xeon processor. The algorithm is of average complexity and takes about 500 lines of code including the display, input, and output routines. While Matlab greatly helps the simplicity of implementation, a complete C++ implementation would be much faster. However, since our algorithm speed is already very competitive, we opted against further low level optimizations and porting our code to C++.

We used color images proposed in previous work to compute an RGB color-to-gray scale mapping. We selected a set of six input images that were picked as examples in previous work, so that we could compare directly to the images produced by other algorithms. In Fig. 1 we compare the visual results of color-blind tests, a linear mapping by using PCA, a mapping using color2gray [17], a mapping using the algorithm by Rasche et al. [27], and our algorithm. We used the C++ implementation from the original authors for color2gray algorithm and Rasche et al. algorithm. We did not use any GPU acceleration of all algorithms.

In Table 1 we show timings of the results. Even though we compare our results in a Matlab implementation, we can outperform the implementation speed of the other algorithms.

In Figs. 11–13 we show more visual comparisons. We would argue that our visual result for the candy image is stronger than the competing algorithms. For the flower and sunrise image we can see advantages and disadvantages in all algorithms.

6.3 Parameter selection

We found that the parameter k in the k-nearest-neighbors search algorithm plays an important role in the final results. The larger the value of k, the higher the quality of the results. An example is shown in Fig. 14. When k is not big enough, there might be some noticeable salt-and-pepper noise artifacts in the result. However, increasing k will significantly increase the computation time. For example, in Fig. 15, when $k = 15$, the computation time is 10.37 s. When $k = 50$, the computation time is 12.53 s. When $k = 200$, the computation time is 17.50 s. Larger k will also make the distance matrix less sparse and significantly increase the storage requirement. For our experiments we fix k to be 50 to balance the speed and the quality.

Table 1 Comparison of computation times in seconds of color2gray, Rasche et al., and our algorithm. Each row of the table are results for one data set whose name is in the leftmost column

	C2G [17]	Rasche [27]	Ours	Image size
Number 6	57	n/a	6	124 ∗ 120
Number 35	52	n/a	6	128 ∗ 125
Number 8	46	n/a	4	120 ∗ 120
Candy	494	400	32	212 ∗ 218
Flower	2100	330	86	311 ∗ 300
Sunrise	1060	n/a	42	295 ∗ 212

7 Discussion

In the following we discuss the results and the comparison to other algorithms.

Local vs. global operator Our mapping algorithm is a global operator. It means that for two input points C_i and C_j, if their values are the same, they will always be mapped to the same grayscale value c_i in spite of their spatial locations in the input image. This is in contrast to other local mapping operators, which will distort the grayscale values to enhance local contrast. We did not include a local enhancement step for two reasons: first, the local enhancement cannot be naturally incorporated in the ISOMAP framework; second, we believe in some cases global operators might be

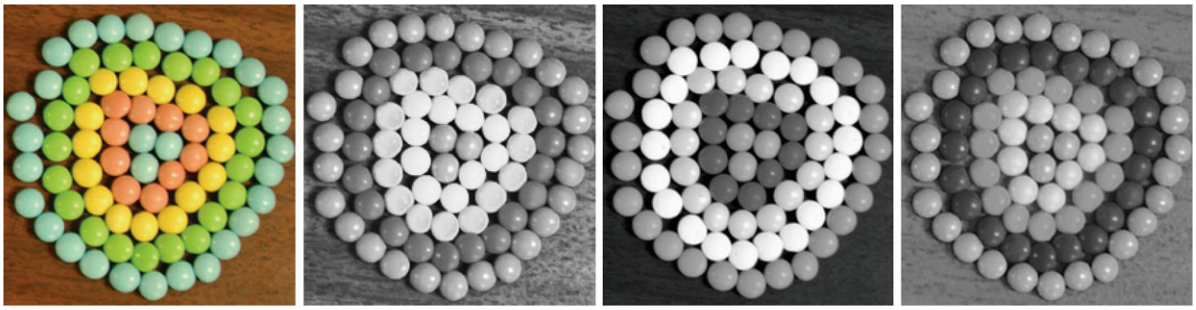


Fig. 11 From *left* to *right*: the original image, color2gray mapping [17]; Rasche et al. mapping [27]; ours, $\lambda = 0.8$

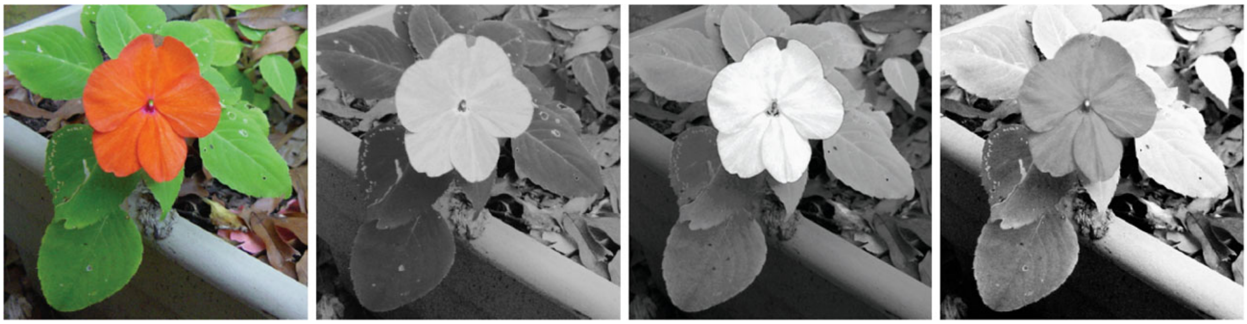


Fig. 12 From *left* to *right*: the original image; color2gray mapping [17]; Rasche et al. mapping [27]; ours, $\lambda = 2.0$

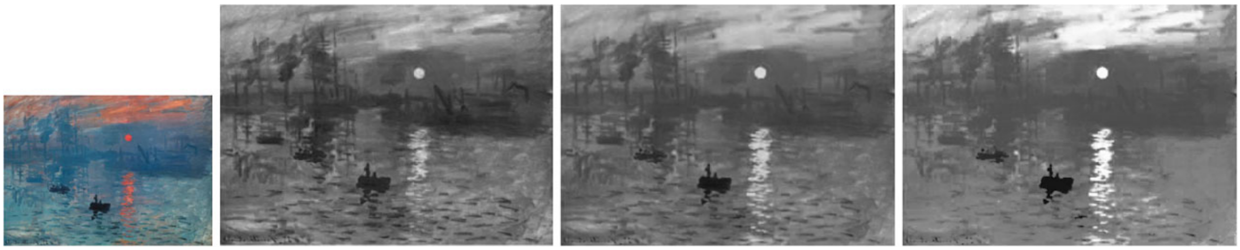
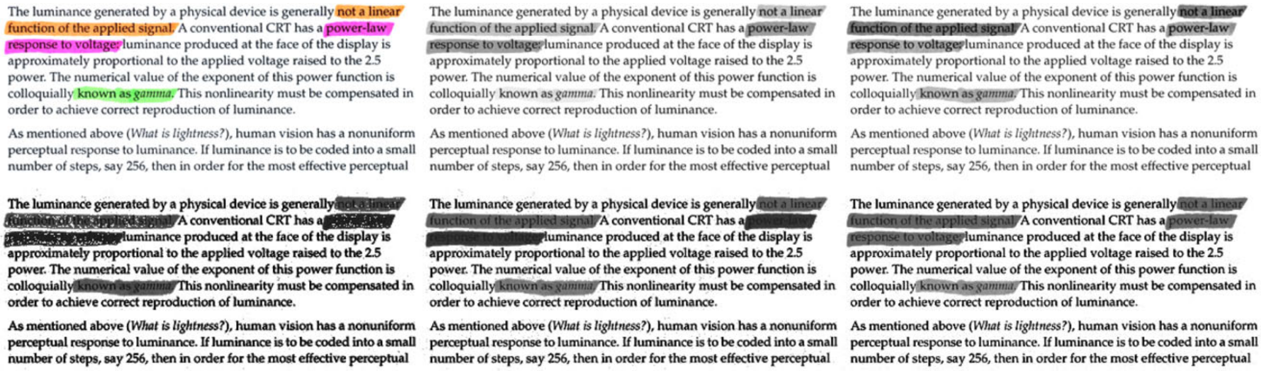


Fig. 13 From *left* to *right*: the original image; decolorize [18]; color2gray [17]; ours, $\lambda = 2.0$



The luminance generated by a physical device is generally not a linear function of the applied signal. A conventional CRT has a power-law response to voltage: luminance produced at the face of the display is approximately proportional to the applied voltage raised to the 2.5 power. The numerical value of the exponent of this power function is colloquially known as *gamma*. This nonlinearity must be compensated in order to achieve correct reproduction of luminance.

As mentioned above (*What is lightness?*), human vision has a nonuniform perceptual response to luminance. If luminance is to be coded into a small number of steps, say 256, then in order for the most effective perceptual

Fig. 14 First row, from left to right: original image; CIEY [14]; decolorize [18]. Second row, comparison of different settings of parameter k for our method. From left to right: $k = 15$, $k = 30$, $k = 50$

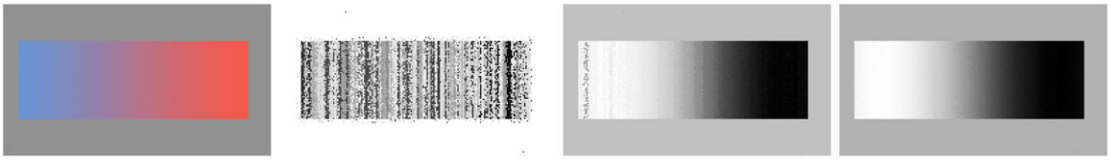


Fig. 15 From *left* to *right*: original color image; our method with $k = 15$, $k = 50$, $k = 200$; all the other parameters are fixed

desirable. For example, if we want to impose tone-mapping on an output grayscale image or mutual information based registration on an output image pair, global operators will be preferable.

Control of the output Similarly to color2gray [17] and Rasche et al. [27], we allow for some control of the output. Similarly to color2gray, we are able to select a direction in $L^*a^*b^*$ color space and decide whether to map this direction to a lighter or darker color. However, we do not have an equivalent for luminance consistency constraints in Rasche et al.'s algorithm. The idea of luminance consistency is to enforce that the relative order of color luminance of similar gamut is enforced. While this control mechanism sounds reasonable in the description, we were not able to verify its importance in the test images and the influence of luminance consistency constraints was also not evaluated in the original paper. In contrast to previous approaches, we provide a parameter λ for contrast management.

Linear vs. nonlinear optimization We think the strength of this algorithm is that it uses a linear operation to map colors from a higher dimensional color space to a lower dimensional one. While the overall mapping is still nonlinear, the nonlinear aspect is due to the sub-manifold detection and geodesic distance computation that precedes the actual mapping step. Our optimization can be computed directly by spectral matrix decomposition and leads to a global optimum. This part of the algorithm is very stable and does not have the many problems of nonlinear optimization. In contrast, previous algorithms [17, 27] use nonlinear optimization. While nonlinear optimization is a powerful tool, it is also very hard to set up and it is not really possible to know if the proposed solution is close to the global optimum.

Sub-manifolds in color images Similarly to previous work our algorithm is beneficial if the color distribution in the original RGB image contains sufficient complexity. Some color images are simple enough so that the simple conversion with Photoshop is sufficient. Even though our algorithm is especially suitable for color distributions that exhibit local or global lower dimensional manifolds, we found the output to be meaningful for all input images.

Limitations The algorithm has several limitations and challenges. First, the sub-manifold detection algorithm uses a k-nearest-neighbor algorithm (KNN) to find neighbors for each color vector in color space. KNN has some known disadvantages: a fixed number of k might not work well on all data sets; KNN search becomes slow in high dimensions. It might be possible to improve results using an adaptive KNN sampling algorithm. This typically comes at the cost of implementation speed. It might actually be more useful to take the opposite approach and use a fast approximate k-nearest-neighbor algorithm instead. The second challenge is the memory consumption of the algorithm. Part of the problem arises from our implementation in Matlab. However, it would be worthwhile to explore more aggressive sub-sampling and clustering strategies for rows and columns.

8 Conclusions

In this paper we explained the ISOMAP algorithm and used it for color image to grayscale conversion. We cast the problem into a dimension reduction problem that has a simple parameter for users to control the level of contrast enhancement naturally. The speed of the algorithm is fast and the quality is competitive with respect to the state-of-the-art global color-to-gray algorithms. There are several interesting avenues for future work. First, we want to implement an out of core algorithm for the mapping, so that a high quality reference solution can be computed for very large images. Second, based on the current solution, we want to improve the quality of the mapping. We are also interested in algorithms for preprocessing textures and shaders in computer games to assist color-blind users.

References

1. Artusi, A., Bittner, J., Wimmer, M., Wilkie, A.: Delivering interactivity to complex tone mapping operators. In: Christensen, P., Cohen-Or, D. (eds.) Rendering Techniques 2003 (Proceedings Eurographics Symposium on Rendering), pp. 38–44. Eurographics, Eurographics Association (June 2003)
2. Bachmann, C.M., Ainsworth, T.L., Fusina, R.A.: Exploiting manifold geometry in hyperspectral imagery. IEEE Trans. Geosci. Remote Sens. **43**(3), 441–454 (2005)
3. Bachmann, C.M., Ainsworth, T.L., Fusina, R.A.: Improved manifold coordinate representations of large scale hyperspectral scenes. IEEE Trans. Geosci. Remote Sens. **44**(10), 2786–2802 (2006)
4. Bala, R., Braun, K.M.: Color-to-grayscale conversion to maintain discriminability. In: SPIE Conference Series. SPIE Conference Series, vol. 5293, pp. 196–202 (December 2003)
5. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural Comput. **15**(6), 1373–1396 (2003)
6. Borg, I., Groenen, P.J.F.: Modern Multidimensional Scaling: Theory and Applications, 2nd edn. Springer, Berlin (2005)
7. Čadík, M.: Perceptual evaluation of color-to-grayscale image conversions. Comput. Graph. Forum **27**(7), 1745–1754 (2008)
8. Cai, N., Younan, S., Du, N., Raksuntorn, Q.: Color representation and classification for hyperspectral imagery. In: IGARSS, pp. 537–540 (Aug 2006)
9. Cui, M., Razdan, A., Hu, J., Wonka, P.: Interactive hyperspectral image visualization using convex optimization. IEEE Trans. Geosci. Remote Sens. **47**(6), 1673–1684 (2009)
10. de Silva, V., Tenenbaum, J.: Global versus local methods in nonlinear dimensionality reduction (2003)
11. de Silva, V., Tenenbaum, B.: Sparse multidimensional scaling using landmark points. Technical Report (2004)
12. Donoho, D., Grimes, C.: Hessian eigenmaps: locally linear embedding techniques for high dimensional data. Proc. Natl. Acad. Sci. **100**(10), 5591–5596 (2003)
13. Durand, F., Dorsey, J.: Fast bilateral filtering for the display of high-dynamic-range images. ACM Trans. Graph. **21**(3), 257–266 (2002)
14. Fairchild, M.D. (ed.): Color Appearance Models. Wiley-IST (2005)
15. Fattal, R.: Edge-avoiding wavelets and their applications. ACM Trans. Graph. **28**(3), 1–10 (2009)
16. Fattal, R., Lischinski, D., Werman, M.: Gradient domain high dynamic range compression. In: SIGGRAPH '02: Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques, pp. 249–256. ACM, New York (2002)
17. Gooch, A.A., Olsen, S.C., Tumblin, J., Gooch, B.: Color2gray: Salience-preserving color removal. In: SIGGRAPH '05: ACM SIGGRAPH 2005 Papers, pp. 634–639. ACM, New York (2005)
18. Grundland, M., Dodgson, N.A.: Decolorize: Fast, contrast enhancing, color to grayscale conversion. Pattern Recogn. **40**(11), 2891–2896 (2007)
19. Han, T., Goodenough, D.G.: Investigation of nonlinearity in hyperspectral remotely sensed imagery: A nonlinear time series analysis approach. In: IGARSS, pp. 1556–1560 (July 2007)
20. Jacobson, N.P., Gupta, M.R.: Design goals and solutions for display of hyperspectral images. **43**(11), 2684–2692 (2005)
21. Ledda, P., Chalmers, A., Troscianko, T., Seetzen, H.: Evaluation of tone mapping operators using a High Dynamic Range display. ACM Trans. Graph. **24**(3), 640–648 (2005)
22. Li, Y., Sharan, L., Adelson, E.H.: Compressing and companding high dynamic range images with subband architectures. ACM Trans. Graph **24**(3), 836–844 (2005)
23. Lischinski, D., Farbman, Z., Uyttendaele, M., Szeliski, R.: Interactive local adjustment of tonal values. In: SIGGRAPH '06: ACM SIGGRAPH 2006 Papers, pp. 646–653. ACM, New York (2006)
24. Mantiuk, R., Myszkowski, K., Seidel, H.-P.: A perceptual framework for contrast processing of high dynamic range images. ACM Trans. Appl. Percept. **3**(3), 286–308 (2006)
25. Nadler, B., Lafon, S., Coifman, R., Kevrekidis, I.: Diffusion maps, spectral clustering and eigenfunctions of Fokker–Planck operators. In: Weiss, Y., Schölkopf, B., Platt, J. (eds.) Advances in Neural Information Processing Systems 18, pp. 955–962. MIT Press, Cambridge (2006)
26. Park, S.H., Montag, E.D.: Evaluating tone mapping algorithms for rendering non-pictorial (scientific) high-dynamic-range images. J. Vis. Commun. Image Represent. **18**(5), 415–428 (2007)

27. Rasche, G.R.K., Westall, J.: Re-coloring images for gamuts of lower dimension. Comput. Graph. Forum **24**(3), 423–432 (2005)
28. Rasche, K., Geist, R., Westall, J.: Detail preserving reproduction of color images for monochromats and dichromats. IEEE Comput. Graph. Appl. **25**(3), 22–30 (2005)
29. Reinhard, E., Stark, M., Shirley, P., Ferwerda, J.: Photographic tone reproduction for digital images. ACM Trans. Graph. **21**(3), 267–276 (2002) (Proceedings of SIGGRAPH 2002 Annual Conference)
30. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science **290**(5500), 2323–2326 (2000)
31. Sha, F., Saul, L.K.: Analysis and extension of spectral methods for nonlinear dimensionality reduction. In: ICML ’05: Proceedings of the 22nd International Conference on Machine Learning, pp. 784–791. ACM, New York (2005)
32. Smith, R. (ed.): Analyzing Hyperspectral Images with TNTmips. Microimages (2006)
33. Smith, K., Landes, P.-E., Thollot, J., Myszkowski, K.: Apparent greyscale: A simple and fast conversion to perceptually accurate images and video. Comput. Graph. Forum **27**(2) (Apr 2008) (Proceedings of Eurographics 2008)
34. Socolinsky, D.A., B Wolff, L.: Multispectral image visualization through first-order fusion. IEEE Trans. Image Process. **11**(8), 923–931 (2002)
35. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science **290**(5500), 2319–2323 (2000)
36. Thomas, B.A., Strickland, R.N., Rodriguez, J.J.: Color image enhancement using spatially adaptive saturation feedback. In: ICIP ’97, vol. 3, p. 30. Washington, DC, USA, 1997. IEEE Comput. Soc., Los Alamitos (1997)
37. Tumblin, J., Rushmeier, H.: Tone reproduction for realistic images. IEEE Comput. Graph. Appl. **13**(6), 42–48 (1993)
38. Tyo, J.S., Konsolakis, A., Diersen, D.I., Olsen, R.C.: Principal-components-based display strategy for spectral imagery. IEEE Trans. Geosci. Remote Sens. **41**(3), 708–718 (2003)
39. Wang, J., Chang, C.I.: Independent component analysis-based dimensionality reduction with applications in hyperspectral image analysis. IEEE Trans. Geosci. Remote Sens. **44**(6), 1586–1600 (2006)

Ming Cui received his B.E. in Civil Engineering and M.Sc. in Computer Science from Zhejiang University, Hangzhou, China in 2002 and 2005, respectively. Currently he is a Ph.D. candidate in Arizona State University and works in Partnership for Research in Spatial Modeling (PRISM) Lab from 2005. His research interests include computer graphics and image processing.

Jiuxiang Hu received the B.Sc., M.Sc. and Ph.D. degrees in Mathematics from Huazhong Normal University in 1988, Huazhong University of Science and Technology, Wuhan, China, in 1991 and Lanzhou University, Lanzhou, China, in 1994, respectively. He is Research Scientist in Imaging and 3D Data Exploitation and Analysis (I3DEA) Lab in the Division of Computing Studies at ASU Polytech campus and Partnership for Research in Spatial Modeling (PRISM) Lab at Arizona State University from 2000. His research interests include computer graphics, visualization, image processing and numerical computation. He has developed and implemented methods to segment biomedical volume data sets, including image statistical and geometric modeling and segmentation techniques with application to structural and quantitative analysis of 2D/3D images.

Anshuman Razdan (M’05) received the B.S. and M.Sc. degrees in Mechanical Engineering and the Ph.D. degree in Computer Science. He is Associate Professor in the Division of Computing Studies and the Director of Advanced Technology Innovation Collaboratory and the I3DEA Laboratory at Arizona State University, Polytechnic campus. He has been a pioneer in computing-based interdisciplinary collaboration and research at ASU.

His research interests include geometric design, computer graphics, document exploitation, and geospatial visualization and analysis. He is the principal investigator and a collaborator on several federal grants from agencies, including the US National Science Foundation (NSF), NGA, and NIH. He is a member of the IEEE.

Peter Wonka (M’08) received his Ph.D. and M.Sc. from the Technical University of Vienna in Computer Science. Additionally he received an M.Sc. in Urban Planning from the same institution. He is currently with Arizona State University (ASU).

Prior to coming to ASU, he was Post-Doctorate Researcher at the Georgia Institute of Technology for two years. His research interests include various topics in computer graphics, visualization, and image processing. He is a member of the IEEE.