Interactive Hyperspectral Image Visualization Using Convex Optimization

Ming Cui, Anshuman Razdan, *Member, IEEE*, Jiuxiang Hu, and Peter Wonka, *Member, IEEE*

***Abstract*—In this paper, we propose a new framework to visualize hyperspectral images. We present three goals for such a visualization: 1) preservation of spectral distances; 2) discriminability of pixels with different spectral signatures; 3) and interactive visualization for analysis. The introduced method considers all three goals at the same time and produces higher quality output than existing methods. The technical contribution of our mapping is to derive a simplified convex optimization from a complex nonlinear optimization problem. During interactive visualization, we can map the spectral signature of pixels to red, green, and blue colors using a combination of principal component analysis and linear programming. In the results, we present a quantitative analysis to demonstrate the favorable attributes of our algorithm.**

***Index Terms*—Hyperspectral image visualization, linear programming, perceptual color distances, principal component analysis (PCA).**

I. Introduction

HYPERSPECTRAL images contain hundreds of spectral samples per pixel. To visualize such images, the many spectral bands must first be projected to a lower dimensional space, typically the red, green, and blue (RGB) color space of a monitor.

In this paper, we present a framework to *visualize* hyperspectral images. The problem can be formulated as follows. A $w \times h$ hyperspectral image with d bands can be seen as a higher dimensional tensor $T_d \in R^{w \times h \times d}$ where each of the wh pixels is described by a vector X_i with d spectral samples. As output of this algorithm, we want to map the image to a lower dimensional display range, i.e., a tensor $T_k \in R^{w \times h \times k}$ with all entries constrained to lie between 0 and 1 (normalized display range) and $k < d$. In this paper, we consider the $k = 3$ case to map to the display range of a color monitor.

Manuscript received July 7, 2008; revised September 16, 2008 and October 21, 2008. First published April 14, 2009; current version published May 22, 2009. This work was supported in part by the National Geospatial-Intelligence Agency under Grants HM1582-08-BAA-0003 and HM1582-05-1-2004 and by the National Science Foundation under Grant IIS 0612269.

M. Cui and P. Wonka are with the Partnership for Research in Spatial Modeling (PRISM) Laboratory, Department of Computer Science and Engineering, Arizona State University, Tempe, AZ 85287 USA (e-mail: ming.cui@asu.edu).

A. Razdan is with the Imaging and 3-D Data Exploitation and Analysis (I3DEA) Laboratory, Division of Computing Studies, Arizona State University at the Polytechnic Campus, Mesa, AZ 85212 USA.

J. Hu is with the Imaging and 3-D Data Exploitation and Analysis (I3DEA) Laboratory, Division of Computing Studies, Arizona State University at the Polytechnic Campus, Mesa, AZ 85212 USA, and also with the Partnership for Research in Spatial Modeling (PRISM) Laboratory, Department of Computer Science and Engineering, Arizona State University, Tempe, AZ 85287 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TGRS.2008.2010129

We set out by defining three goals for such a visualization: 1) The distances from the input spectral space should be preserved in the output color space to provide a perceptually meaningful visualization; 2) the algorithm should make use of the dynamic range of the display device to show details in the image; and 3) the algorithm should allow for interactive exploration.

Hyperspectral image visualization is usually provided as a functionality in hyperspectral image analysis software such as Multispec [1], ENVI [2], Geomatics [3], TnTlite [4], HyperCube [5], and HIAT [6]. A direct visualization method is to render the image as a 3-D cube [3], [6]. To explore different bands as grayscale images, one set of tools allows a user to cycle through all bands or to flicker between two bands [5]. To extract an RGB color image for visualization, interactive tools can be used to pick three bands and assign them to the RGB channels directly [1], [2]. More sophisticated mappings can be created through user-specified linear combinations of spectral bands [5], data-independent visually meaningful linear combinations [7], or data-dependent automatically computed combinations using principle component analysis (PCA) or minimum noise fraction (a noise reduction version of PCA) [2], [4]. Additionally, independent component analysis (ICA) has been proposed for dimension reduction [8], [9], but ICA is significantly slower than PCA, and it is not clear how to rank the significance of different channels provided by ICA. An alternative idea for visualization would be to borrow from nonlinear methods for dimension reduction, such as locally linear embedding (LLE) and isometric feature mapping (ISOMAP). Two recent nonlinear color mapping approaches by Gooch *et al.* [10] and Rasche *et al.* [11] report running times of minutes to compute a mapping for input images with three spectral samples. In general, we expect these existing nonlinear techniques to require significantly more computation time than linear methods, particularly for $d \gg 3$.

The main problem that we observed is that existing methods map spectral samples to *unbounded* 3-D Euclidean space. After dimension reduction, they all use not only a second nonuniform mapping to color space that creates colorful images but also the illusion of salient features that are not present in the data. Examples are nonuniform scaling, standard deviation stretch, and histogram equalization. Therefore, these algorithms sacrifice the first goal (preservation of spectral distances) to satisfy the second one (using the dynamic range of the display).

In this paper, we propose a novel strategy for hyperspectral image visualization that uses a higher quality mapping. The main idea of our approach is to derive a fast optimization procedure that can perform dimension reduction while considering the boundaries of the hue, saturation, and value (HSV) [12]

color space (see Fig. 4 for a short introduction). During the visualization, a user can interactively explore the hyperspectral data set using spatial and spectral lenses to configure a nonlinear mapping. Our major contribution is as follows.

- We present a high-quality framework for hyperspectral image visualization. We provide the quality of nonlinear methods while preserving much of the interactivity only available with simple linear methods. Both visual and quantitative comparisons suggest that our method satisfies the three goals simultaneously better than existing methods.

II. Related Work

We review related work in three categories: 1) hyperspectral image visualization; 2) color to gray mapping; and 3) dimensionality reduction.

A. Hyperspectral Image Visualization

Traditionally, hyperspectral images have been visualized as a cube with a suite of interactive tools [13]. One set of tools allows a user to extract one spectral band at a time or cycle through spectral bands as an animation. To create RGB images, interactive tools can be used to specify RGB values as linear combinations of spectral bands. This means that an RGB value is computed by a matrix vector multiplication. Along these lines, several authors have suggested methods to automatically create linear combinations of spectral bands to define the RGB blue color channels of a visualization [7], [14]–[16]. In this paper, we compare our results to two such methods: 1) the color-matching function (CMF) algorithm proposed by Jacobson *et al.* [7] and 2) a traditional PCA-based method described in [14, Sec. 2]. The main problem in existing visualization software toolkits is the application of postprocessing algorithms to enhance the image. Examples are nonuniform scaling, standard deviation stretch, and histogram equalization. These algorithms disproportionately enhance minor features. It is worth noting that a visualization can be specifically designed for different applications. For example, a visualization can be used as a postprocess for classification [17].

B. Color to Gray Mapping

In recent years, transforming color images to grayscale attracted the interest of several researchers [10], [11], [18], [19]. The problem is to find a lower dimension embedding of the original data that can best preserve the contrast between the data points in the original data. While these papers are an inspiration for our work, their methodologies do not easily extend to higher dimensions due to memory consumption and computation time.

C. Dimensionality Reduction

There are a larger number of general dimensionality reduction algorithms in the literature. Prominent examples are ISOMAP [20], LLE [21], Laplacian eigenmap embedding [22], Hessian eigenmap embedding [23], conformal maps [24], and diffusion maps [25]. These algorithms are theoretically very strong. However, there are two issues. First, these algorithms assume that the data lie in a nonlinear submanifold in the original space. This assumption must be verified before these nonlinear dimension reduction methods can be used on hyperspectral images. Although previous studies [26], [27] suggest that nonlinearity exists in hyperspectral imagery, the nonlinearity is typically data dependent. Second, nonlinear dimension reduction methods are usually slow and memory intensive. For example, a small 100×100 image gives rise to a distance matrix with $100^4 = 100$ million entries. Computing the singular value decomposition (a typical step needed in these methods) does not scale well to larger images, and a 500×500 image is already out of reach for current workstations. In [28], an accelerated version of ISOMAP is implemented for hyperspectral images. The method greatly enhances the algorithm speed, but running times are still not fast enough for an interactive visualization.

III. Overview

Here, we give an overview of this paper. First, we lay out three goals of the visualization and derive quantitative metrics that we will use to compare our algorithm to previous work. Second, we give the motivation for our algorithm and explain how we derived it. Third, we give a short description of the individual steps of the algorithm.

A. Goals

Preservation of Distances: The first goal of our visualization is to create an image such that the perceptual color distances are similar to the Euclidean distances between the high-dimensional spectral samples. We follow the argumentation which suggests that the Euclidean distance in RGB color space is not a good measure for the perceptual distance (see, for example, [14] and [29]). Therefore, we attempt to preserve Euclidean distances in a perceptual color space, $\mathrm{L^*a^*b^*}$ [12]. To evaluate the preservation of distances in $\mathrm{L^*a^*b^*}$, we define a correlation-based metric similar to that in [7]. Let X be the vector of all pairwise Euclidean distances of the pixels in the high-dimensional spectral space, and let vector Y be the corresponding pairwise Euclidean distances of the pixels in $\mathrm{L^*a^*b^*}$ space. The correlation γ can be calculated using the following formula:

$$\gamma = \frac{X^T Y/|X| - \bar{X}\bar{Y}}{std(X) \cdot std(Y)}. \tag{1}$$

$|X|$ denotes the number of elements in X, and $\bar{X}$ and $std(X)$ denote the mean and standard deviation, respectively. In the ideal case, the normalized correlation equals 1, and the closer the correlation is to 1, the better the distance is preserved. In practice, the images that we consider are too large to consider all pairwise distances, so that we accelerate the computation by subsampling.

It is important to discuss alternative design choices for metric γ. An alternative to Euclidean distances in the original space is to only consider the spectral angles in the original space (e.g., [7]). While spectral angles contain enough information for

endmember identification in classification applications, the radiant intensities in the original space carry additional information about ground textures that reflect shape and contours [7]. This is very important for a series of tasks such as georeferencing [2], registration [30], and change detection [31]. Another idea suggested by Jacobson and Gupta [7] is to ignore the luminance values in $L^*a^*b^*$ space. In our experience, better performance can be achieved when making use of the full color space including the luminance value.

Separability of Features: Correlation alone does not guarantee that colors can be well distinguished. It is still possible that the resulting image is too dark or too bright, because the color space is not efficiently used and too many pixels fall within a smaller part of the color space. Therefore, we use a metric δ that measures how well pixels are mapped to distinguishable colors. The key idea is that the average distance between two pixels in perceptual color space should be as large as possible

$$\delta = |Y|_1/|Y| \tag{2}$$

where Y is from (1), $|Y|_1$ denotes the L1 norm, and $|Y|$ is the number of elements of the vector. Therefore, δ denotes the average pairwise Euclidean distance in the $L^*a^*b^*$ color space. The same metric was independently suggested by Du *et al.* [16]. Larger values of δ indicate a better separability of features, and we therefore try to maximize δ.

Interactive Visualization: For an effective visualization, the data set should be interactively explored by a human user. We consider two aspects of the interactive visualization. First, the computation time should not exceed a few seconds. Second, it is important to have a method that is compatible with interactive tools for data exploration. In this paper, we introduce spatial and spectral lenses as example tools and show how the algorithm can be integrated into an interactive hyperspectral image visualization framework.

B. Design Choices

We first present our analysis of the state of the art and then discuss three possible approaches to improve the visualization. The third approach is the one we follow in this paper.

State of the Art: State-of-the-art techniques use a two-step framework which is shown in the first row of Fig. 2. These techniques map the high-dimensional spectral samples to an unbounded 3-D Euclidean space in the dimension reduction step. Even though the mathematical transformations, such as PCA and ICA, perform typically well when mapping to an infinite space, the color space has an actual boundary that needs to be respected. We call this problem the *boundary problem*. Please note that this problem is similar to the tone reproduction problem in computer graphics [32]. For example, the RGB color space has a cube as a boundary, and the Lab color space is bounded by concave surfaces. Therefore, these methods need a second transformation which actually maps 3-D points to RGB triples for visualization purposes. The simplest transformation is uniformly scaling the point set so that it fits into a cube. The three coordinates of the scaled space can be used as the RGB color values. Usually, the points are sparsely distributed in the cube so that the resulting image tends to be too dark [see Fig. 1(a)]. In the quantitative analysis, this visualization has

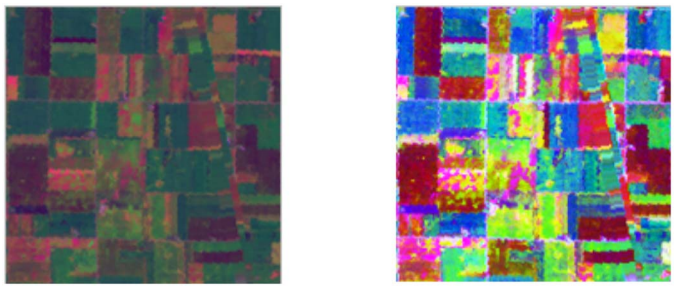


Fig. 1. Comparing two visualization algorithms for a multispectral data set. (a) Visualization with PCA. (b) Visualization with enhanced PCA. Note how smaller features are exaggerated. The first three eigenvectors are mapped to (R, G, B) channels, respectively

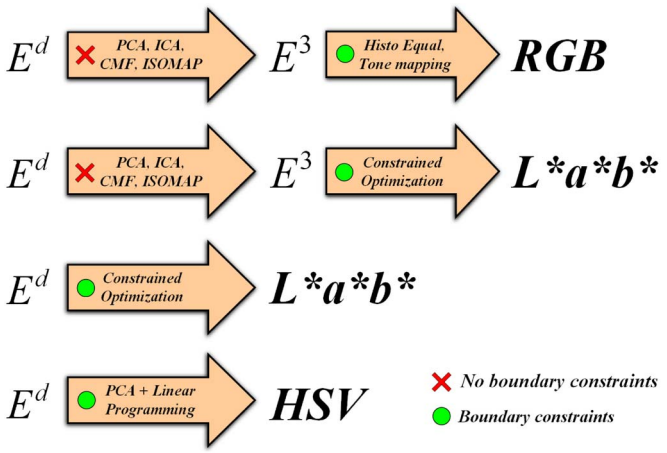


Fig. 2. Four different strategies to address the hyperspectral image visualization problem. The state of the art is shown in the top row. Two possible alternatives are shown in rows 2 and 3. Our approach is shown in row 4 (these alternatives are discussed in Section III-B).

good preservation of distances (γ) but not a high separability of features (δ). Therefore, different alternatives to uniform scaling have been proposed to enhance the separability of features (δ) of the final image such as an exponential transform, nonuniform scaling, standard deviation mapping, the auto normalization transform, histogram equalization [33], and tone mapping [34] techniques. An example of enhanced PCA visualization is shown in Fig. 1(b). However, these techniques work on each color channel separately and therefore distort perceptual color distances nonuniformly. As a result, minor features in the data can be disproportionably exaggerated, and the preservation of distances is poor (γ).

Approach 1: The first idea that we considered was to replace the second mapping with a mapping to $L^*a^*b^*$ color space to better keep perceptual distances (see Fig. 2, row 2). The $L^*a^*b^*$ color space is the most popular choice in this context. However, the envelope of $L^*a^*b^*$ space has a concave and curved boundary, and the cascading of two mappings produced similar undesirable side effects to existing methods.

Approach 2: To avoid complications due to two mappings, we intended to directly map points in E^d to $L^*a^*b^*$ space, posing the problem as a constrained optimization (see Fig. 2, row 3).

Approach 3: Approach 2 seems to be the ideal solution for preserving perceptual distance. However, the $L^*a^*b^*$ color

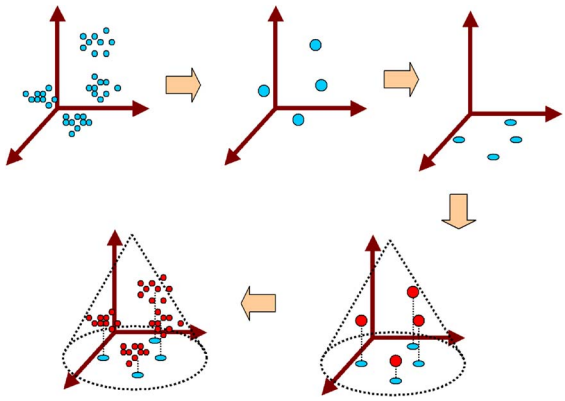


Fig. 3. Pipeline of our framework. (Step 1) The points in high dimension are first clustered, and representatives for clusters are extracted. (Step 2) Representatives are projected to 2-D. (Step 3) The coordinates in the third dimension are computed using linear programming. (Step 4) Interpolation of other points.

space has concave nonlinear boundaries. A constrained optimization that maps the data set into $\mathrm{L^*a^*b^*}$ is very time consuming and does not meet our interactive speed requirement. Therefore, we opted for a third approach (see Fig. 2, row 4) that uses the HSV color space instead of the $\mathrm{L^*a^*b^*}$ color space. Although the HSV color space is not as good as $\mathrm{L^*a^*b^*}$ color space, it is still much better than the RGB color space in preserving perceptual distance. Our experiments will show that even though we optimize color distances in the HSV color space, the distances in the $\mathrm{L^*a^*b^*}$ color space are well preserved according to our correlation metric. Additionally, this simplification allowed us to derive a solution based on a convex optimization that can be solved at interactive speeds. Overall, using the HSV color space is a good compromise between quality and speed requirements. Next, we present an overview of our method and fill in algorithm details in Section IV.

C. Pipeline

The proposed framework includes four steps (see Fig. 3) described in the following.

1) *Preprocessing:* The motivation for preprocessing is to accelerate the computation time. Preprocessing consists of a vector quantization method to cluster the spectral signature of image pixels into M clusters. Optimization is performed on cluster representatives, and interpolation is performed on the remaining spectral samples. We implemented the faster median cut algorithm [35] and the higher quality k-means algorithm [36]. For each cluster, we select one representative point. The outputs of this stage are the representative points and cluster membership information for all the pixels (spectral samples). See Section IV-A for details. The performance and parameter settings for the number of clusters M are evaluated in Section V.
2) *Dimension Reduction:* In this stage, we want to map the representative points to the HSV color space. Our solution is a two-step algorithm that is tradeoff between computation speed and quality according to the metrics γ and δ. First, we project the representative points onto a 2-D plane using PCA. The points are then enclosed by a circle which constitutes the boundary in the hue–saturation plane of the HSV color space. Second, we employ a convex optimization to assign intensity values to the representative points.
3) *Interpolation and Color Mapping:* The hue–saturation components of the remaining points are computed by projecting to the 2-D plane used in the previous step. The intensity component of a point is decided by the distance to its representative point. Finally, all points are mapped into the HSV cone.
4) *Interactive Visualization:* We provide a suite of interactive tools to explore and analyze a hyperspectral data set. We provide three types of tools: 1) linear transformations of the color space; 2) spatial lenses so that the user can interactively select a subregion of the image and recalculate a mapping that enhances the visual discriminability of the features in the subregion; and 3) spectral lenses that enhance the visual discriminability of pixels similar to a user-specified spectral signature.

IV. Methodology

A. Preprocessing

The input to the preprocessing step is the original data set, a high-dimensional tensor $T_d \in R^{w \times h \times d}$ where each of the $w \times h$ pixels is described by a vector X_i with d spectral samples. The output of the preprocessing step is the following clustering information: 1) a $w \times h$ integer map with values ranging from 1 to M; the integer is the cluster ID for the corresponding pixel; 2) the centroid of each cluster, denoted as $\{R_1^d, R_2^d, \ldots, R_M^d\}$; and 3) the average Euclidean distance of all points to their representative point, denoted as $\{r_1, r_2, \ldots, r_M\}$.

We implemented dimension reduction as an option to accelerate the clustering. We use PCA to first reduce the number of bands and keep 99.9% of information from the original data sets. This results in keeping the first 10–20 eigenvectors. The clustering algorithm is conducted in the projected subspace. This acceleration is fairly conservative and has negligible influence on the output. Therefore, we use it for all results in this paper. We use a C implementation of the k-means and the median cut clustering algorithm. For both algorithms, we tried various settings for the number of clusters (M). We found that choosing median cut clustering with $M = 50$ gives a good balance between speed and quality of the mapping. We analyze clustering performance and the influence of parameter settings in Section V-C.

B. Dimension Reduction

1) Problem Formulation: Our method for dimension reduction starts from a multidimensional scaling perspective. Given the spectral samples in original d-dimensional Euclidean space denoted as $\{X_1, X_2, \ldots, X_n\}$, with each of them being a vector in d-dimensions, we try to map them to 3-D color vectors denoted as $\{C_1, C_2, \ldots, C_n\}$. To introduce the problem, we only consider the goal of preservation of distances and ignore the boundaries of the color space, so that we allow any color

vector in unbounded 3-D Euclidean space. That means that we want to find a mapping that minimizes the objective function E

$$E = \sum_{i=1}^{N} \sum_{j=i+1}^{N} \left(D_d(i,j) - D_3(i,j)\right)^2 \tag{3}$$

or in compact form

$$E = \frac{1}{2}\|D_d - D_3\|_F. \tag{4}$$

F denotes the Frobenius norm, D_d and D_3 are matrices so that $D_d(i,j)$ denotes the Euclidean distance between X_i and X_j, and $D_3(i,j)$ denotes the Euclidean distance between C_i and C_j. It is possible to solve this optimization problem using majorization [37] techniques. However, a more popular approach is to center matrices D_d and D_3 first. The centering operator τ can be computed by $\tau(D) = -HSH/2$, where $S = D^2$ and $H = I - 1/N * O$, with O being a matrix of all ones and N being the number of rows of square matrix D. This transforms the problem to minimizing the objective function E

$$E = \|\tau(D_d) - \tau(D_3)\|_F. \tag{5}$$

The importance of the centering operator is that it transforms the problem from a nonlinear optimization to an eigendecomposition problem. The main idea is to replace distances with dot products. The global optimum of (5) is selected as the three eigenvectors of matrix $\tau(D_d)$ that are associated with the largest three positive eigenvalues. Each color vector C_i can map the first three eigenvectors to any combination of RGB. This approach is called classical dimensional scaling. It can be shown that the result is exactly the same as performing PCA on the data set and computing the projection on the three principle components associated with the first three largest eigenvalues [37]. In summary, PCA is one of the best and most popular 3-D projections in the sense of classical multidimensional scaling. However, we need to find a mapping that constrains the vectors C_i to lie within the boundaries of a color space.

2) Dimension Reduction Using Convex Optimization: The main motivation and technical contribution of our approach is to derive a convex optimization for the dimension reduction step. Our main ingredients are the use of the HSV color space and a novel solution to split the dimension reduction in two steps. The envelope of the HSV color space is a right circular cone whose height equals the radius of the base plane. The HSV space is shown in Fig. 4. Our algorithm to map the points from the original d-dimensional space into a 3-D cone shape has the following steps. First, we compute the hue–saturation component of the representative points using linear projection. Second, we assign the intensity component of the representative points. For the second step, we will explain how we model the intensity assignment as a linear programming problem.

3) Projection to 2-D Hue–Saturation Space: The goal of this step is to project the representative points $\{R_1^d, R_2^d, \ldots, R_M^d\}$ in the original d-dimensional space to a 2-D space. The projected points are denoted by $\{R_1^2, R_2^2, \ldots, R_M^2\}$. Note that the superscripts denote the dimensionality of the space where the points reside in. The 2-D space will be parallel to the base plane of the HSV color cone. We use a fast projection

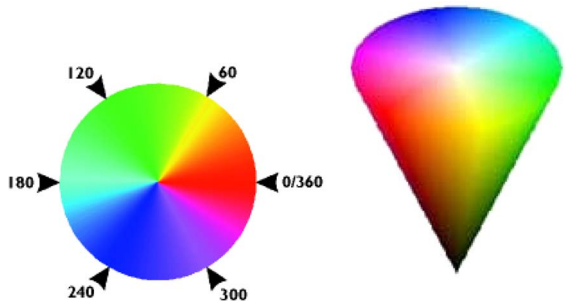


Fig. 4. HSV color space is bounded by a circular cone. (a) We show a cross section of the cone. Hue ranges from 0 to 360 and describes the spectrum of pure colors. Saturation is the distance to the center of the circle and describes color strength. Adding more white will make the color weaker. (b) We show the cone in 3-D. The value coordinate describes how bright or dark the color is.

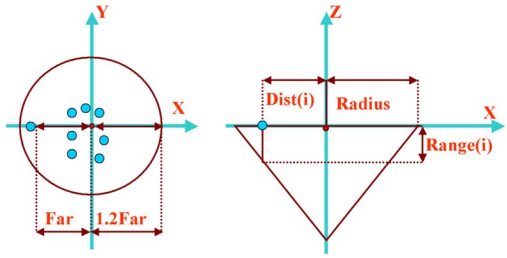


Fig. 5. (a) Two-dimensional projection in a circle. (b) Range of the intensity for each point. The X–Y plane is the *hue–saturation* plane, and the Z-axis corresponds to *value* from the HSV color space.

using PCA for this step. Second, we need to find a circle in the plane that describes the boundary of the HSV cone. We use the centroid of the points $\{R_1^2, R_2^2, \ldots, R_M^2\}$ as middle point of the circle and set $radius = ratio * Far$, where Far denotes the distance of the centroid to the farthest point [see Fig. 5(a)]. We need to slightly enlarge the circle because we only use representative points at this step and other points will still be further away. We use $ratio = 1.2$ as an initial heuristic, but the user can modify this parameter interactively. Please note that we do not want to enclose all projected points in the base circle because this would make our method sensitive to outliers. Up to now, the hue–saturation component of the representative points are decided up to a rotation factor. This degree of freedom can also be set by the user during interactive exploration (see Section IV-D1).

4) Computing the Intensity Component: In this step, we compute the intensity components for all representative points denoted as $\{I_1, I_2, \ldots, I_M\}$. Note that (R_1^2, I_1), $(R_2^2, I_2), \ldots, (R_M^2, I_M)$ together fully describe the final projection in 3-D space $(R_1^3), (R_2^3), \ldots, (R_M^3)$. Please note that we use negative intensity values between 0 and -1 since the cone is upside down (-1 maps to black). The principal goal is to preserve the mutual distances of representative points in the original space as much as possible [see (3)] while respecting several constraints. The problem can be modeled as a linear programming problem. We will describe our solution in five parts. First, we describe the objective function. Second, we show how the objective function can be transformed to map to a convex optimization algorithm. Third, we explain how to incorporate the boundaries of the HSV cone as constraints.

Fourth, we explain how to add cluster separability as constraint. Finally, we show how to solve the problem numerically.

Objective Function: The objective in (3) is to keep D_3 as close to D_n as possible. At this step in the algorithm, we already have a 2-D projection as a partial solution, and we only need to solve for 1-D coordinates in the third dimension. We denote the pairwise distance matrix for $\{R_1^2, R_2^2, \ldots, R_M^2\}$ in 2-D space as D_2 and the pairwise distance matrix for $\{I_1, I_2, \ldots, I_M\}$ as D_1. In the ideal case, $D_1(i,j) = \sqrt{(D_d(i,j))^2 - (D_2(i,j))^2}$ according to the Pythagorean theorem. Therefore, we want to minimize the following objective function E:

$$E = \sum_{i=1}^{M} \sum_{j=i+1}^{M} \left| D_1(i,j) - \sqrt{(D_d(i,j))^2 - (D_2(i,j))^2} \right|. \tag{6}$$

Transforming to a Linear Programming Problem: Even though the term under the square root in (6) is a constant, the resulting optimization is still nonconvex. The reason for this is the absolute value in $D_1(i,j) = |I_i - I_j|$. Therefore, we cannot solve the minimization problem using linear programming. The key insight to bypass this problem is to decide the order of $\{I_1, I_2, \ldots, I_M\}$ using a reasonable criterion beforehand. This order becomes the first set of constraints

$$\text{if} \quad Dist(i) \geq Dist(j), \qquad I_i \geq I_j. \tag{7}$$

We will discuss the meaning of $Dist(i)$ and why we impose these constraints shortly. Now, $D_1(i,j) = I_i - I_j$ when $I_i \geq I_j$, or $D_1(i,j) = I_j - I_i$ when $I_i < I_j$. The absolute value disappears, and the problem is reduced to a linear programming problem.

Cone Boundary Constraints: The second set of constraints is straightforward: We want to restrict the position of the points in the final 3-D space within a cone shape. Let us denote $Dist(i)$ as the distance from point R_i^2 to the origin of the base circle. Since the height of the cone is equal to the radius of base circle, we know that $|I_i| + Dist(i) \leq Radius$ must be satisfied, as shown in Fig. 5(b). Therefore, if $Range(i) = Radius - Dist(i)$, the following two constraints must be applied to make R_i^3 stay in a cone:

$$I_i \leq 0 \quad I_i \geq -Range(i). \tag{8}$$

Transforming to LP Problem Constraints: Now, we look back at the first set of constraints and explain why they make sense. Since the smaller the $Dist(i)$ is, the larger $Range(i)$ is, we intentionally make the final 3-D points appear more like a cone shape by arranging their intensity values in descending order according to $Dist(i)$. It is worth mentioning that other methods exist to decide the order of $\{I_1, I_2, \ldots, I_M\}$. For example, we can use the total energy in the original space: $|R_i^d|_1$ to order them. We tested this setting, and final results do not form a good cone shape.

Cluster Separability Constraints: The third and last set of constraints is designed to separate important points. Remember the we define r_i as the average distance to the ith representative point for all points belonging to that cluster. We define two representative points to be well separated if $r_i + r_j \leq D_d(i,j)$. We want the cluster center to remain well separated in the final 3-D space: $r_i + r_j \leq D_3(i,j)$. This boils down to the following constraint:

$$\text{if} \quad r_i + r_j \leq D_d(i,j), \quad \text{then} \quad \sqrt{(r_i + r_j)^2 - (D_2(i,j))^2} \leq D_1(i,j). \tag{9}$$

Solve Convex Optimization: Now, we have fully set up the linear programming model: Minimize (6) subject to (7)–(9). We solve it with CLP.[1] CLP is a C++ library of several linear programming solvers. We use the primal–dual method implemented in the library. This is not the fastest method, but it works fast enough for our application. Alternatively, instead of minimizing (6), we can minimize the following term $\bar{E}$:

$$\bar{E} = \max_{i=1,\ldots,M, j=i+1,\ldots,M} \times \left| D_1(i,j) - \sqrt{(D_d(i,j))^2 - (D_2(i,j))^2} \right|. \tag{10}$$

This formulation uses the infinity norm instead of the 1-norm. We believe that it gives a better theoretical guarantee. Therefore, we use this objective function for all results in this paper. The derivation of the corresponding linear programming problem is analogous to the derivation presented in this section.

C. Interpolation and Color Mapping

1) Interpolate All Points: At this stage, the coordinates of the representative points are fully determined in 3-D space (HSV color space). In this step, we will map all the remaining points based on the location of the representative points. The coordinates in the hue–saturation plane are decided by projecting all other points in the original space to the same plane using the first two principle components that we have used for the representative points. Note that, alternatively, we can use the first two principle components of all data points with similar computation time. However, performing PCA on all points will bias favorably toward the separability of clusters with a larger number of members. The intensity I_p for a particular pixel P^d is decided by the following formula:

$$I_p = I_i + flag * \sqrt{Edist\left(P^d, R_i^d\right) - Edist\left(P^2, R_i^2\right)}. \tag{11}$$

In the formula, i denotes which cluster P^d belongs to. I_i is the intensity value that we have decided in the previous section. $Edist(p,i)$ denotes the Euclidean distance between p and i in the original space. *flag* is either 1 or -1 denoting whether pixel I_p should be darker or lighter than its representative point intensity I_i, respectively. *flag* is decided by comparing the sum of all band values for this pixel to the sum of all bands for the representative point in the original space. In Fig. 6, we show the mapped cone shape of a real data set (LunarLake02).

2) Fit the Color Cone to Data Cone: Now, the basic idea is to make the color cone big enough to enclose the data cone such that all points are mapped to different colors. This corresponds to the largest red cone shown in Fig. 7(a). To fit the cone so that it encloses all data points might not be the best strategy

[1] https://projects.coin-or.org/Clp

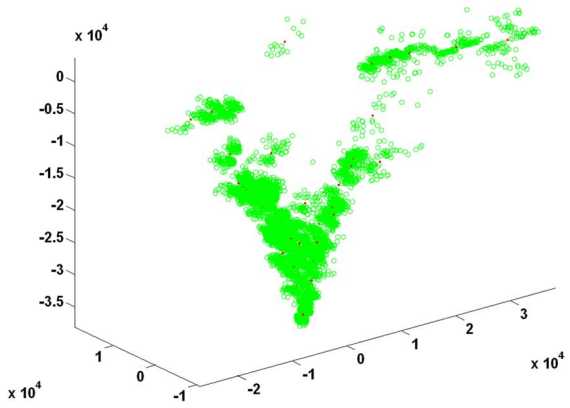


Fig. 6. All data points are mapped to 3-D Euclidean space using our method described in Section IV. Note how they form a cone shape. The units are Euclidean distances in the original data set.

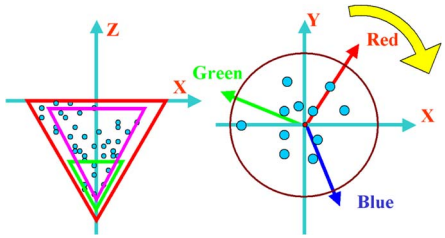


Fig. 7. (Left) How to fit the color cone to the data cone. (Right) HSV cone can be rotated by the user.

due to outliers in the data. The display strategy is to allow the user to shift the color cone up and down along the Z-axis and scale the radius of the cone to enhance different parts of the data. This idea is illustrated using the green and pink cones in Fig. 7(a). Data points outside the color cone will be clamped to the boundary of the cone.

D. Interactive Visualization

In the following, we outline three tools that are useful for interactive visualization: 1) interactive tone rotation; 2) spatial lenses; and 3) spectral lenses.

1) Interactive Tone Rotation: The user can rotate the orientation of the base plane. This is shown in Fig. 7(b). Note that the intensities do not need to be recalculated due to the symmetric structure of the cone. Remapping can be computed in less than a second for millions of points. We found that functionality is very helpful to enhance the contrast and to change the visualization to more aesthetic color choices.

2) Spatial Lens: We propose to use spatial lenses to improve the contrast in selected regions. We can specify a region interactively and recompute a mapping only on that part of the data. The region is treated as an input image itself. Although we still use the same thematic labeling which we got from the preprocessing step for the whole data set, we remap these colors to a cone shape in itself. This allows us to enhance smaller features while locally keeping proportional perceptual distances. A local mapping is also achievable with PCA on the subregion. The PCA will also perform well if the selected region is sufficiently small. This is shown in Fig. 8: We see in CMF that the features are distinguished fairly well; however, the overall contrast can be improved. In the PCA without

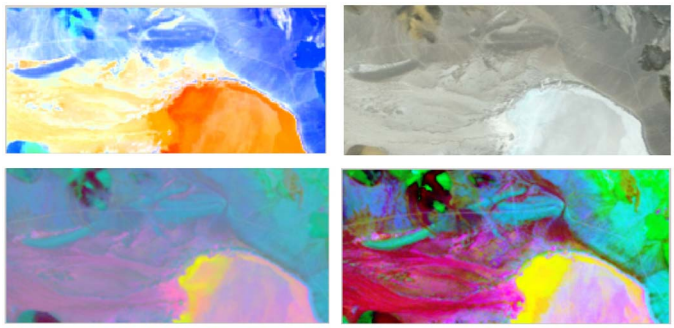


Fig. 8. (Top left) Result using our spatial lens. (Top right) Result of CMF, zoomed in. (Bottom left) Result using PCA. (Bottom right) Result using PCA HE. The rectangular region is selected within the yellow border in the top-left image of Fig. 12. Algorithmic details are described in Section V-A.

enhancement, the features are not easily distinguished, particularly for the slim white branches at the upper right corner. In the PCA with enhancement setting, the results are very colorful, and features are distinguishable. However, the color mapping does not correspond to the true distance and is misleading. For example, at the middle part near the bottom, we see a big contrast of yellow and pink areas, even though these regions have similar spectral signatures. More results for the spatial lens are available in Section V.

3) Spectral Lens: Another tool that we offer is to enable the user to interactively pick one particular pixel in the result image and highlight pixels that have a similar spectral signature to the pixel in the original space. This way, we can use a larger portion of the color space for selected features and their surrounding. We implement this functionality by creating a new cluster whose centroid is set to be the pixel that the user has picked and the members to be all the pixels within some radius in the original space. We call this new center R^d_{M+1}. Note that the thematic labeling map is changed due to insertion of the new cluster. Then, the mapping algorithm is recomputed with a new constraint specifying $I_{M+1} \geq I_i$ for all $i \leq M$. The goal of this constraint is to guarantee that the neighborhood of this pixel is "highlighted." The results are shown in the middle image of the top row of Figs. 11 and 12. Note that there are also some other algorithms available to achieve a similar goal. For example, a well-known technique is to set the hue and saturation components of all the points within a distance directly to the same as that of the selected pixels and keep their intensity values untouched. We argue that our approach is more systematic by intentionally making the rest of the points less "highlighted." Additionally, our approach can provide more hue–saturation variations within the cluster.

V. Results

A. Implementation Details

We implemented our algorithm in Matlab on a 3.6-GHz Xeon processor. The algorithm takes less than 1000 lines of code including the display, input, and output routines. While Matlab greatly helps the simplicity of implementation, a complete C++ implementation would be faster. However, since our algorithm speed is already very competitive, we opted against further low-level optimizations and porting our code to C++.

TABLE I
COMPARISON OF CORRELATION γ

	Our	CMF	PCA	PCA2%	PCA HE
Moffett01	0.96	0.94	0.91	0.68	0.46
Moffett02	0.96	0.79	0.92	0.51	0.35
Moffett03	0.93	0.69	0.96	0.68	0.41
LunarLake01	0.95	0.82	0.92	0.53	0.21
LunarLake02	0.84	0.81	0.95	0.37	0.27
Cuprite01	0.90	0.87	0.91	0.55	0.32
Cuprite02	0.91	0.88	0.95	0.43	0.28

TABLE II
COMPARISON OF AVERAGE DISTANCE δ

	Our	CMF	PCA	PCA2%	PCA HE
Moffett01	38.2	16.4	13.0	50.3	81.4
Moffett02	53.9	30.6	12.4	48.5	77.2
Moffett03	25.5	30.5	9.7	44.7	75.9
LunarLake01	50.9	5.6	10.9	52.5	85.3
LunarLake02	59.8	7.3	15.6	43.8	80.1
Cuprite01	52.1	4.8	10.9	48.4	80.7
Cuprite02	73.0	7.7	13.9	53.2	81.5

We use the AVIRIS data which are available for download online.[2] Each data set has 219 bands ranging from 400 to 2500 nm with uniform steps of 10 nm. The image size is usually around 500×500. We picked seven data sets: scenes 01, 02, and 03 from site Moffett Field, scenes 01 and 02 from site Lunar Lake, and scenes 01 and 02 from site Cuprite.

We compared our method against several previously published algorithms: CMF, PCA, PCA with outlier reduction (PCA 2%), PCA with histogram equalization (PCA HE), spectral band sampling (SBS), and ISOMAP. Note that ISOMAP is a nonlinear dimension reduction method. ISOMAP preserves geodesic distances rather than Euclidean distances. Therefore, we only visually compare ISOMAP results to ours.

CMF linearly projects the data set using three fixed basis vectors which are called CMFs in [7]. The PCA method uses the largest three principle components $(P1, P2, P3)$ to map all pixels to 3-D Euclidean space and then linearly scales the whole data set so that it fits into the (unit) RGB cube. The projection on $(P1, P2, P3)$ is mapped to (R, G, B) channels, respectively. The PCA 2% method also uses PCA and scaling to the RGB cube, but instead of linearly scaling, 2% of the pixels at the ends of each channel are saturated in order to enhance the contrast. A saturated pixel value is one that is moved outside the RGB cube through scaling and subsequently clamped to the boundaries of the cube. Similarly, PCA HE uses PCA for dimension reduction and scaling to the RGB cube but then uses histogram equalization for each color channel. SBS is the simplest algorithm that directly maps the 6th, 20th, and 40th bands to RGB color bands, respectively. Finally, we also implemented a method that uses ISOMAP to do dimension reduction. The implementation of ISOMAP follows [38]. The algorithm takes 0.5 h to get a mapping for an image of $500 \times 500 \times 219$. The implementation of ISOMAP in [28] reports 4.4 h on images with 1.8×10^6 samples, where each sample has 124 bands. Since the algorithm complexity is $O(N \log^2(N))$, we estimate that the running time on an image of $500 \times 500 \times 124$ will take at least 500 s. We therefore believe that current implementations do not meet the interactive display requirement.

[2]http://aviris.jpl.nasa.gov/html/aviris.freedata.html

TABLE III
COMPARING RUNNING TIME IN SECONDS OF DIFFERENT NUMBER OF CLUSTERS AND CLUSTERING METHODS FOR OUR METHOD (UNITS ARE IN SECONDS)

		K-means			Median cut		
Number of clusters (M):		25	50	100	25	50	100
Mof01 25%	Cluster	2.9	7.2	11.8	0.3	0.2	0.2
	Optimize	0.4	5.7	82.6	0.4	6.3	70
	Total	3.3	12.9	94	0.7	6.5	70
Mof01 50%	Cluster	17.4	47.8	86.6	0.9	1.0	1.1
	Optimize	0.4	5.9	84	0.4	7.2	77
	Total	17.8	54	171	1.3	8.2	78
Mof01 100%	Cluster	115	427	917	5.2	4.8	5.4
	Optimize	0.4	6	87	0.6	5.9	66
	Total	115	433	1000	5.8	10.7	71.4
Lak01 25%	Cluster	3.6	8.2	14.2	0.2	0.2	0.2
	Optimize	0.5	5.8	79	0.5	6.3	92
	Total	4.1	14	94	0.7	6.5	92
Lak01 50%	Cluster	23.4	59.2	125	1.0	0.9	0.9
	Optimize	0.4	5.6	79.9	0.6	7.9	75.6
	Total	23.8	60	205	1.6	8.8	77
Lak01 100%	Cluster	126	523	1037	4.6	4.8	4.5
	Optimize	0.4	6.2	86.8	0.5	7.4	89.6
	Total	126	529	1124	5.1	12.2	94

TABLE IV
COMPARING CORRELATION γ AND AVERAGE DISTANCE δ OF DIFFERENT NUMBER OF CLUSTERS AND CLUSTERING METHODS FOR OUR METHOD

		K-means			Kd-tree		
Number of clusters (M):		25	50	100	25	50	100
Mof01 25%	Correlation	0.83	0.90	0.92	0.86	0.92	0.92
	Avg Dist	56.6	58.9	51.7	62.8	54.5	54.6
Mof01 50%	Correlation	0.94	0.94	0.95	0.93	0.97	0.97
	Avg Dist	36.9	34.7	31.4	35.8	39.8	39.7
Mof01 100%	Correlation	0.96	0.95	0.93	0.96	0.96	0.97
	Avg Dist	37.3	37.4	33.9	39.4	38.2	38.8
Lak01 25%	Correlation	0.87	0.87	0.92	0.88	0.88	0.87
	Avg Dist	83.9	84.1	84.1	83.9	84.0	83.9
Lak01 50%	Correlation	23.4	59.2	125	1.0	0.9	0.9
	Avg Dist	0.4	5.6	79.9	0.6	7.9	75.6
Lak01 100%	Correlation	0.93	0.93	0.96	0.95	0.95	0.95
	Avg Dist	49.7	50.5	49.9	50.1	50.9	50.7

B. Quantitative Comparison

We measure the quality of the mapping based on the two metrics γ, indicating the preservation of distances [see (1)], and δ, indicating the separability of features [see (2)]. The result for γ is shown in Table I, and that for δ is shown in Table II. Note that a good mapping should have γ close to 1 and δ as high as possible. To accelerate the quantitative comparison, we randomly subsample pixels so that, in each row and column, only every fifth pixel is used in the computation of the pairwise distances. That means that only 4% of the pixels are used.

The values of SBS are not competitive, and we did not include them in the table. Since ISOMAP preserves geodesic distances rather than Euclidean distances, it is not meaningful to apply the correlation metric to ISOMAP results.

The comparison of γ values reveals that PCA 2% and PCA HE strongly exaggerate features and therefore have a low correlation score. CMF produces solid results, but our method and PCA are generally better than CMF. Even though our method is better than PCA in some cases, we consider PCA to be the most stable according to γ and, therefore, the best method to preserve the distances. We consider our method the second best.

However, the comparison of δ shows the significant drawback of PCA. The separability of features is low, and this

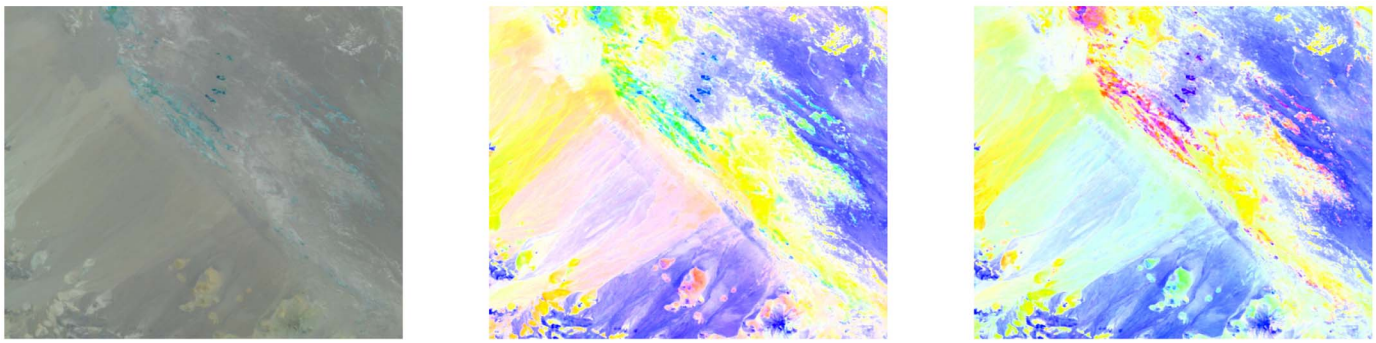


Fig. 9. (Left) CMF result: $\gamma = 0.82$ and $\delta = 5.6$. (Middle) Our result with Kd-tree: $\gamma = 0.95$ and $\delta = 50.9$. (Right) Our result with k-means: $\gamma = 0.95$ and $\delta = 50.5$.

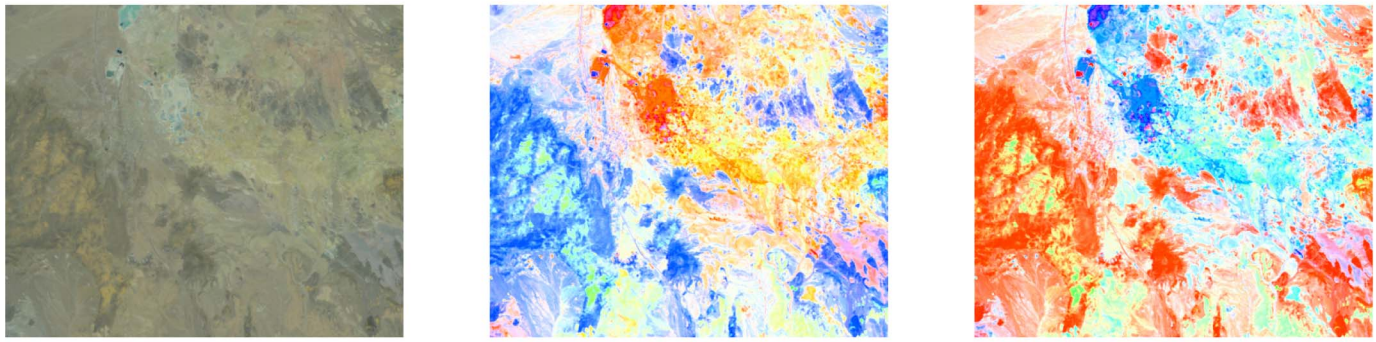


Fig. 10. (Left) CMF result: $\gamma = 0.88$ and $\delta = 7.7$. (Middle) Our result with Kd-tree: $\gamma = 0.91$ and $\delta = 73.0$. (Right) Our result with k-means: $\gamma = 0.90$ and $\delta = 76.2$.

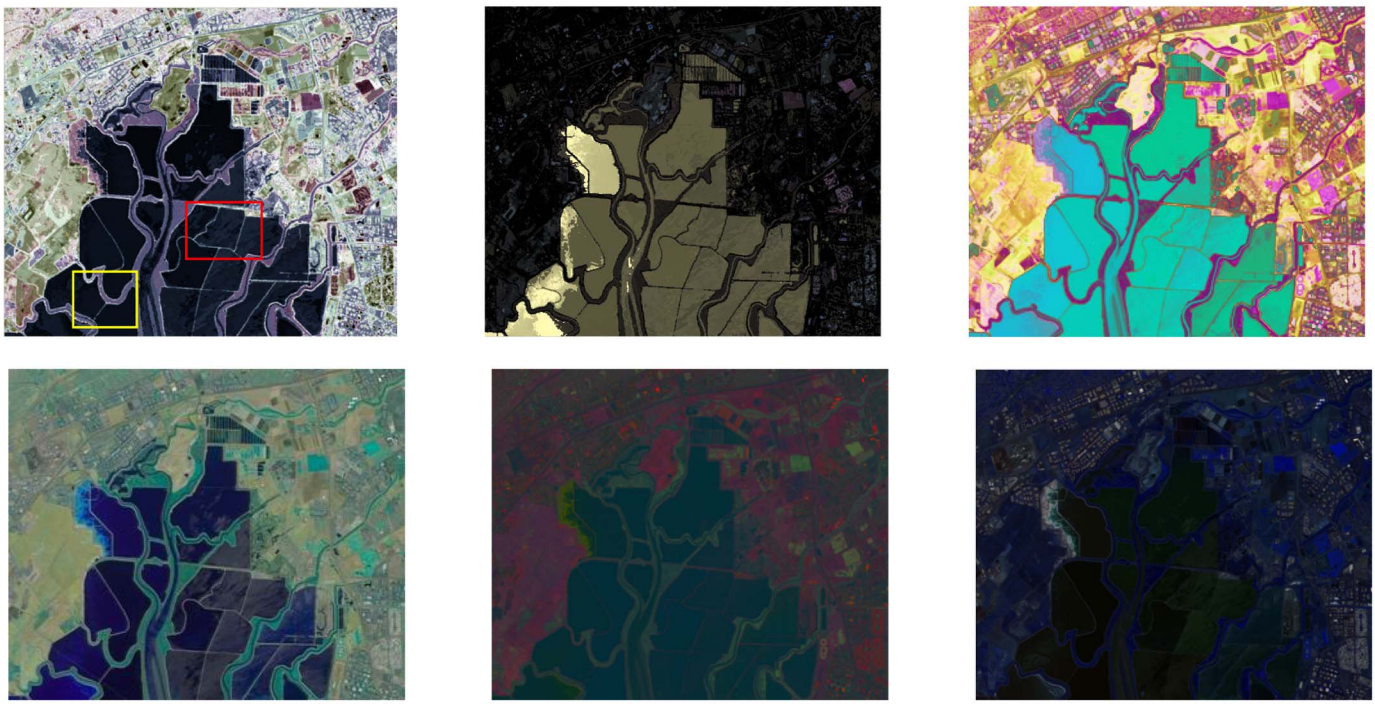


Fig. 11. (Top left) Result using our method. (Top middle) Result using our spectral lens. (Top right) Result using ISOMAP. (Bottom left) Stretched CMFs [7]. (Bottom middle) PCA without enhancement. (Bottom right) Result using three bands directly chosen from the original data. The data set is scene 2 from Moffettfield.

results in dark images that are not useful for visualization. The δ values are a factor of three to five times lower than our method. The δ values of CMF are comparable to that of our method for the first three data sets, but other data sets exhibit δ values that are up to ten times lower. As expected, PCA 2% is comparable to our method, and PCE HE has the best δ

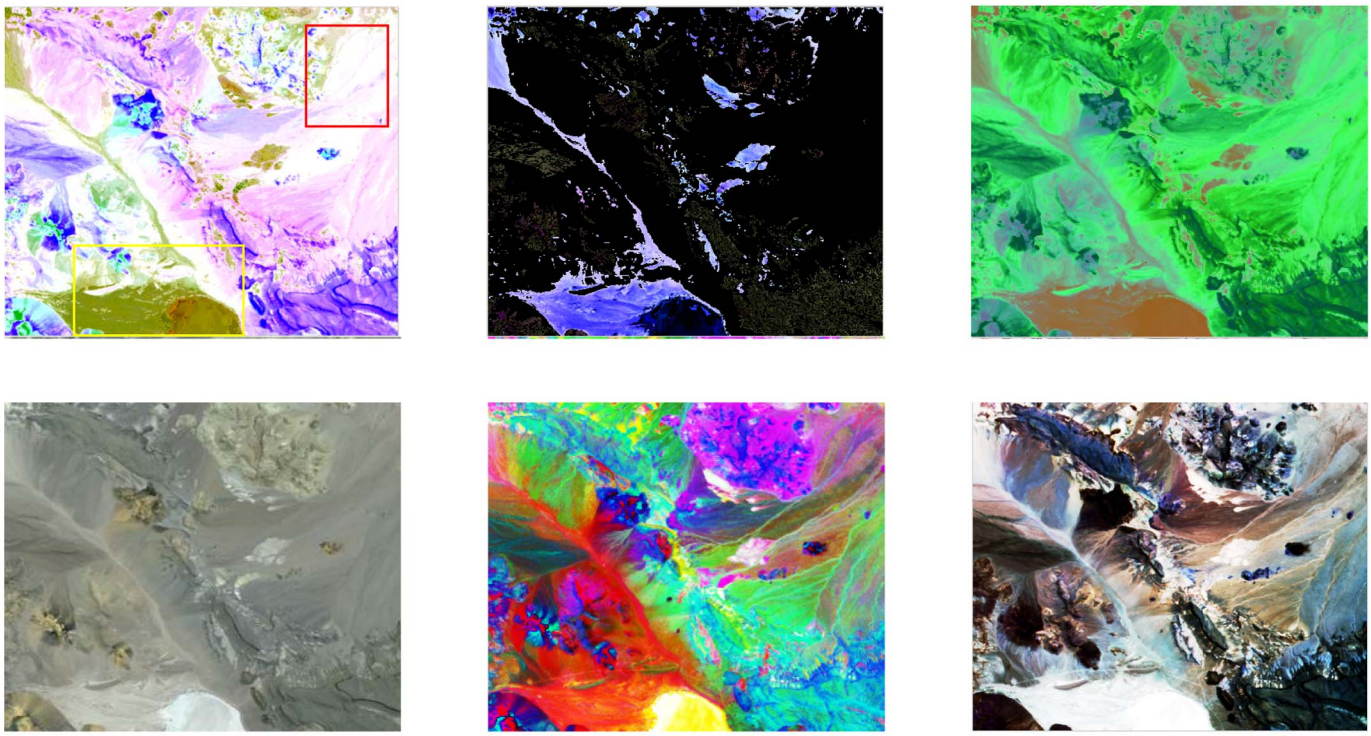


Fig. 12. (Top left) Result using our method. (Top middle) Result using our spectral lens. (Top right) Result using ISOMAP. (Bottom left) Stretched CMFs [7]. (Bottom middle) PCA HE. (Bottom right) Result using three bands directly chosen from the original data. The data set is scene 2 from Lunarlake.

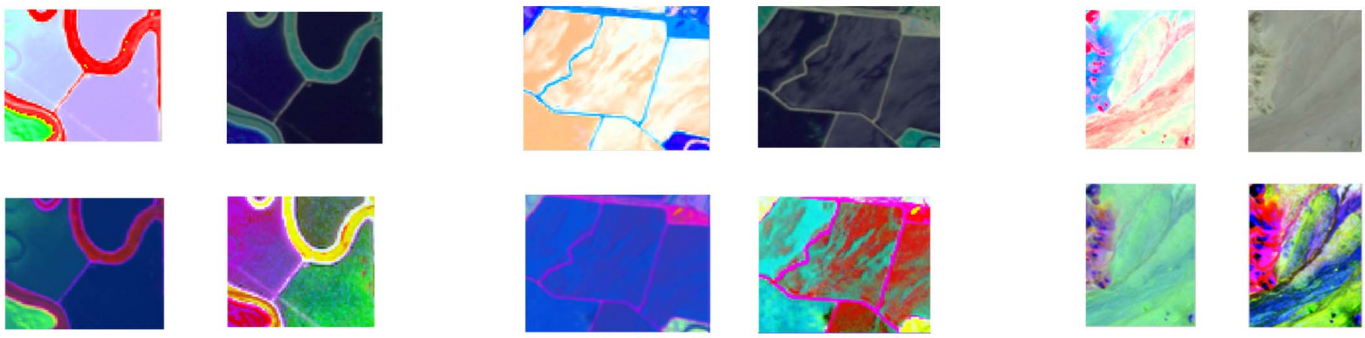


Fig. 13. Application of the spatial lens on three examples. Each of the three examples compares four methods: our algorithm in the top left, CMF in the top right, PCA in the bottom left, and PCA HE in the bottom right. (Left) Spatial lens is applied to the yellow rectangle shown in the left image of Fig. 11. (Middle) Red rectangle in top-left image of Fig. 11. (Right) Red rectangle in the top-left image of Fig. 12. For each example, the layout is the same as that in Fig. 8.

values. However, note that these two algorithms did not perform well according to our metric γ and are generally not distance preserving.

We conclude that our method is the best tradeoff and achieves both goals of preserving spectral distances and separating features in the visualization.

C. Parameter Selection for Our Algorithm

We evaluated the parameter for the number of clusters M and the two implemented clustering algorithms k-means and median cut. We use two data sets Moffett01 and LunarLake01 for the evaluation, and for each of the two data sets, we additionally create 25% and 50% downsampled versions (in spatial dimensions only), giving a total of six data sets. The original size of both the data sets is $512 \times 614 \times 219$.

In Table III, we compare the running time of different settings for the number of clusters, clustering algorithm, and input image size. The two most time-consuming steps in our algorithm are clustering and convex optimization on representative points (interactive visualization tools, such as the rotation of the color wheel and the spatial lens on a small subset of the data, have response times of less than 0.1 s). The table shows that k-means clustering is much slower than median cut. Note that our recommended setting of $M = 50$ results in visualization time of about 10 s which is reasonable for interactive software. We also see that increasing the image size has less impact on the running time than increasing the number of clusters.

In Table IV, we present a quantitative evaluation of the parameter setting. We can observe that the quality for using 50 clusters is only slightly worse than using 100 clusters.

D. Visual Comparison

In Figs. 9 and 10, we show the comparison between our algorithm and the CMF algorithm. While CMF tries to preserve the influence of the visible spectral bands, we can see that several features are lost in the visualization.

In Fig. 11, we see the results of different algorithms on data set Moffettfiled02. In the top-left image, which is our result, the urban area looks very clear, with several details being nonobservable in other methods. The lake is relatively dark but can be made clearer with a spatial or spectral lens. Please also note how the PCA without any enhancement produces a visualization that is too dark, as indicated by low values of δ in our quantitative comparison. In Fig. 12, we show a visual comparison of selected algorithms on the Lunarlake02 data set. See that, in our result, features are again easily distinguishable. For this data set, we use PCA HE. Note that histogram equalization can provide colorful results but that the interpretation is difficult because the distances in the original spectral space are not preserved resulting in low values of γ.

In Fig. 13, we see three more examples of the application of a spatial lens. For each example, we compare our method to CMF, PCA, and PCA HE. We have the following few remarks about the results. In example 1, the curved strip in the left middle part of the image is very clear, while CMF fails to show this. The straight strip does not show up in the PCA without enhancement. In the PCA with enhancement, the body of two parts in the lake has too much perceptual difference: One part is pink, and the other part is green. From the original data, we know that they should not be that dissimilar to each other. In example 2, both our method and CMF appear to do a good job, but in the PCA without enhancement, the features are less clear. In PCA HE, the two different kinds of materials are mapped to blue and red, respectively, which is not desirable. In example 3, CMF cannot distinguish the features very well, while our method and the PCA without enhancement get similar results.

VI. Discussion

Advantages: This new framework provides a good visualization result for hyperspectral images while avoiding distortion of significant features. It also provides real-time interaction to further facilitate exploration. Based on the visual and quantitative comparisons, we argue that we outperform state-of-the-art techniques.

Limitations and Future Work: There are several aspects of our algorithm that we want to improve in future work. One limitation of the current algorithm is that it does not try to map high-dimensional pixel signatures to natural colors. Although we can partially meet this requirement by rotating the color wheel to make a particular part of the image look natural, we do not have a systematic way to guarantee that all features satisfy this criterion at the same time. We would also like to experiment with using ICA instead of PCA for projecting the colors to the 2-D plane.

It is also worth mentioning that the two goals that we set up at the beginning, namely, preserving spectral distance and obtaining high feature separability, may be two contradictory goals. In the current algorithm, these two goals are unified in the optimization process by explicitly setting the preservation of spectral distances as the objective function and casting feature separability as a set of constraints. We would like to explore the possibility of putting both goals in the objective function with a parameter to balance their relative weights in the future.

VII. Conclusion

In this paper, we propose a new framework for hyperspectral image visualization. We are the first to consider the final color space in our computation, and therefore, we are able to derive a higher quality mapping than previous work. Experiments show that the visual quality of the final mapping improves over state-of-the-art approaches. The framework also provides some interaction abilities which are important for a human analyst to explore the data.

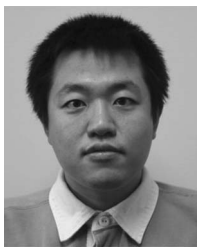
Acknowledgment

The authors would like to thank J. Ye and S. Jeschke for helpful discussions.

References

[1] *MultiSpec*. [Online]. Available: http://cobweb.ecn.purdue.edu/biehl/multispec/description.html
[2] *ENVI*. [Online]. Available: http://www.ittvis.com/envi/
[3] *Geomatics*. [Online]. Available: http://www.pcigeomatics.com/
[4] *TnTlite*. [Online]. Available: http://www.microimages.com/tntlite/
[5] *HyperCube*. [Online]. Available: http://www.tec.army.mil/hypercube/
[6] *HIAT*. [Online]. Available: http://www.censsis.neu.edu/software/hyperspectral/hyperspectral.html
[7] N. Jacobson and M. Gupta, "Design goals and solutions for display of hyperspectral images," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 11, pp. 2684–2692, Nov. 2005.
[8] J. Wang and C.-I Chang, "Independent component analysis-based dimensionality reduction with applications in hyperspectral image analysis," *IEEE Trans. Geosci. Remote Sens.*, vol. 44, no. 6, pp. 1586–1600, Jun. 2006.
[9] Y. Zhu, P. K. Varshney, and H. Chen, "Evaluation of ICA based fusion of hyperspectral images for color display," in *Proc. 10th Int. Conf. Inf. Fusion*, Jul. 9–12, 2007, pp. 1–7.
[10] A. A. Gooch, S. C. Olsen, J. Tumblin, and B. Gooch, "Color2gray: Salience-preserving color removal," in *Proc. ACM SIGGRAPH*, 2005, pp. 634–639.
[11] K. Rasche, R. Geist, and J. Westall, "Re-coloring images for gamuts of lower dimension," *Comput. Graph. Forum*, vol. 24, no. 3, pp. 423–432, Sep. 2005.
[12] D. A. Forsyth and J. Ponce, *Computer Vision: A Modern Approach*. Englewood Cliffs, NJ: Prentice-Hall, 2002.
[13] *Analyzing Hyperspectral Images With TNTmips*, R. Smith, Ed. Lincoln, NE: Microimages, 2006.
[14] A. Tyo, A. Konsolakis, D. Diersen, and R. Olsen, "Principal-components-based display strategy for spectral imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 41, no. 3, pp. 708–718, Mar. 2003.
[15] J. Wang and C. Chang, "Independent component analysis-based dimensionality reduction with applications in hyperspectral image analysis," *IEEE Trans. Geosci. Remote Sens.*, vol. 44, no. 6, pp. 1586–1600, Jun. 2006.
[16] Q. Du, N. Raksuntorn, S. Cai, and R. Moorhead, "Color display for hyperspectral imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 6, pp. 1858–1866, Jun. 2008.
[17] S. Cai, Q. Du, and R. Moorhead, "Hyperspectral imagery visualization using double layers," *IEEE Trans. Geosci. Remote Sens.*, vol. 45, no. 10, pp. 3028–3036, Oct. 2007.
[18] K. Rasche, R. Geist, and J. Westall, "Detail preserving reproduction of color images for monochromats and dichromats," *IEEE Comput. Graph. Appl.*, vol. 25, no. 3, pp. 22–30, May/Jun. 2005.
[19] R. Mantiuk, K. Myszkowski, and H.-P. Seidel, "A perceptual framework for contrast processing of high dynamic range images," *ACM Trans. Appl. Percept.*, vol. 3, no. 3, pp. 286–308, Jul. 2006.

[20] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, Dec. 2000.
[21] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, Dec. 2000.
[22] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Comput.*, vol. 15, no. 6, pp. 1373–1396, Jun. 2003.
[23] D. Donoho and C. Grimes, "Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data," in *Proc. Nat. Acad. Sci. U.S.A.*, May 2003, vol. 100, pp. 5591–5596.
[24] F. Sha and L. K. Saul, "Analysis and extension of spectral methods for nonlinear dimensionality reduction," in *Proc. 22nd ICML*, 2005, pp. 784–791.
[25] B. Nadler, S. Lafon, R. Coifman, and I. Kevrekidis, "Diffusion maps, spectral clustering and eigenfunctions of Fokker–Planck operators," in *Advances in Neural Information Processing Systems*, vol. 18, Y. Weiss, B. Schölkopf, and J. Platt, Eds. Cambridge, MA: MIT Press, 2006, pp. 955–962.
[26] C. M. Bachmann, T. L. Ainsworth, and R. A. Fusina, "Exploiting manifold geometry in hyperspectral imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 3, pp. 441–454, Mar. 2005.
[27] T. Han and D. Goodenough, "Investigation of nonlinearity in hyperspectral remotely sensed imagery: A nonlinear time series analysis approach," in *Proc. IGARSS*, Jul. 2007, pp. 1556–1560.
[28] C. M. Bachmann, T. L. Ainsworth, and R. A. Fusina, "Improved manifold coordinate representations of large-scale hyperspectral scenes," *IEEE Trans. Geosci. Remote Sens.*, vol. 44, no. 10, pp. 2786–2802, Oct. 2006.
[29] P. K. Robertson and J. F. O'Callaghan, "The application of perceptual color spaces to the display of remotely sensed imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 26, no. 1, pp. 49–59, Jan. 1988.
[30] J. Kern and M. Pattichis, "Robust multispectral image registration using mutual-information models," *IEEE Trans. Geosci. Remote Sens.*, vol. 45, no. 5, pp. 1494–1505, May 2007.
[31] N. C. Rowe and L. L. Grewe, "Change detection for linear features in aerial photographs using edge-finding," *IEEE Trans. Geosci. Remote Sens.*, vol. 39, no. 7, pp. 1608–1612, Jul. 2001.
[32] P. Shirley, M. Ashikhmin, M. Gleicher, S. Marschner, E. Reinhard, K. Sung, W. Thompson, and P. Willemsen, *Fundamentals of Computer Graphics*, 2nd ed. Natick, MA: AK Peters, Ltd., 2005.
[33] S. M. Pizer, E. P. Amburn, J. D. Austin, R. Cromartie, A. Geselowitz, T. Greer, B. T. H. Romeny, and J. B. Zimmerman, "Adaptive histogram equalization and its variations," *Comput. Vis. Graph. Image Process.*, vol. 39, no. 3, pp. 355–368, Sep. 1987.
[34] R. Fattal, D. Lischinski, and M. Werman, "Gradient domain high dynamic range compression," in *Proc. 29th Annu. Conf. Comput. Graph. Interactive Tech. SIGGRAPH*, 2002, pp. 249–256.
[35] P. Heckbert, "Color image quantization for frame buffer display," *ACM SIGGRAPH Comput. Graph.*, vol. 16, no. 3, pp. 297–307, Jul. 1982.
[36] J. A. Hartigan and M. A. Wong, "A k-means clustering algorithm," *Appl. Stat.*, vol. 28, no. 1, pp. 100–108, 1979.
[37] T. Cox and M. Cox, *Multidimensional Scaling*. London, U.K.: Chapman & Hall, 1994.
[38] V. de Silva and J. B. Tenenbaum, "Global versus local methods in nonlinear dimensionality reduction," in *Proc. NIPS*, 2002, pp. 705–712.



Ming Cui received the B.E. degree in civil engineer and the M.S. degree in computer science from Zhejiang University, Hangzhou, China, in 2002 and 2005, respectively. He is currently working toward the Ph.D. degree at Arizona State University, Tempe.

He has been with the Partnership for Research in Spatial Modeling (PRISM) Laboratory, Division of Computing Studies, Arizona State University, Polytechnic Campus, Mesa, since 2005. His research interests include computer graphics and image processing.

Anshuman Razdan (M'05) received the B.S. degree in mechanical engineering from the Regional Engineering College, Kurukshetra, India, and the M.S. degree in mechanical engineering and the Ph.D. degree in computer science from Arizona State University (ASU), Tempe.

He is currently an Associate Professor with the Division of Computing Studies and the Director of the Advanced Technology Innovation Collaboratory and the I3DEA Laboratory, ASU, Polytechnic Campus, Mesa. 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 National Science Foundation, the National Geospatial-Intelligence Agency, and the National Institutes of Health.

Jiuxiang Hu received the B.S. degree in mathematics from the Huazhong Normal University, Wuhan, China, in 1988, the M.S. degree in mathematics from the Huazhong University of Science and Technology, Wuhan, in 1991, and the Ph.D. degree in mathematics from Lanzhou University, Lanzhou, China, in 1994.

He has been a Research Scientist with the Imaging and 3-D Data Exploitation and Analysis (I3DEA) Laboratory, Division of Computing Studies, Arizona State University, Polytechnic Campus, Mesa, and with the Partnership for Research in Spatial Modeling (PRISM) Laboratory, Arizona State University, Tempe, since 2000. His research interests include computer graphics, visualization, and 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 2-D/3-D images.

Peter Wonka (M'08) received the M.S. degree in urban planning and the M.S. and Ph.D. degrees in computer science from the Technical University of Vienna, Vienna, Austria.

He is currently with the Research in Spatial Modeling (PRISM) Laboratory, Department of Computer Science and Engineering, Arizona State University (ASU), Tempe. Prior to coming to ASU, he was a Postdoctoral Researcher with the Georgia Institute of Technology, Atlanta, for two years. His research interests include various topics in computer graphics, visualization, and image processing.