

Anisotropic Diffusion of Hyperspectral Imagery

David Hwang
Project Supervisor: Todd Wittman
June 12, 2009

Abstract

Noise in images often acts as a pitfall in image segmentation. In noisy images, segmentation may yield undesirable results. One approach to remedy this problem is to blur the image in such a way that blurring only occurs between the edges of an image. This can be done using anisotropic diffusion [1]. This paper aims to demonstrate how anisotropic diffusion can be used to improve segmentation of images, in particular hyperspectral imagery.

Background

Anisotropic diffusion can be defined by the following differential equation [2],

$$\frac{\partial u_i}{\partial t} = \nabla \cdot (g(\xi) \nabla u_i), \quad i = 1, \dots, m \quad (1)$$

where $u = (u_1, \dots, u_m)^T$ is an image composed of m bands and $g(\xi)$ is a function where $g(\xi) \rightarrow 0$ when $\|\nabla u\|$ is large. Intuitively, the norm of an image gradient $\|\nabla u\|$ is large on an edge of an image (i.e. a division between forest and concrete). Therefore, diffusion of the edges can be avoided by selecting a proper edge-stopping function $g(\xi)$. Perona-Malik recommends using either one of the choices of $g(\xi)$ below [1],

$$g(\xi) = \frac{1}{1 + \frac{\xi}{p}}, \quad \xi = \|\nabla u\|^2, \quad (2)$$

$$g(\xi) = e^{-\frac{\xi}{p}}, \quad \xi = \|\nabla u\|^2, \quad (3)$$

where p is a constant such that $p > 1$.

A drawback of diffusion techniques is over-diffusion of image where the new image may stray too far from the original. The goal is to remove noise, not to blur the entire image to one color. Therefore, it is advisable to add a fidelity term to the differential equation described in (1),

$$\frac{\partial u_i}{\partial t} = \nabla \cdot (g(\xi) \nabla u_i) - \alpha(u_i - f_i), \quad i = 1, \dots, m, \quad (4)$$

where f is the original image and α acts as a fidelity parameter. The fidelity term forces the equation, at least to an extent, to acknowledge the original image.

It must be noted that these calculations are performed on each band. However, when calculating a new band, contents of the other bands are ignored because a band of an image is treated “separately” (i.e. calculations for one band only use input from the same band). It can be beneficial to perform “coupled” anisotropic diffusion on a band where information from all bands of an image is used to create a new band, as opposed to using the one band “separated”

from the others. To perform, “coupled” anisotropic diffusion, the following differential equation can be used [3],

$$\frac{\partial u_i}{\partial t} = \nabla \bullet (D(u) \nabla u_i), \quad i = 1, \dots, m, \quad (5)$$

where D is a 2x2 tensor matrix. To form D , first, find the eigenvalues and eigenvectors of the following 2x2 matrix,

$$\begin{pmatrix} \sum_{i=1}^m \left(\frac{\partial u_i}{\partial x}\right)^2 & \sum_{i=1}^m \frac{\partial u_i}{\partial x} \frac{\partial u_i}{\partial y} \\ \sum_{i=1}^m \frac{\partial u_i}{\partial x} \frac{\partial u_i}{\partial y} & \sum_{i=1}^m \left(\frac{\partial u_i}{\partial y}\right)^2 \end{pmatrix}. \quad (6)$$

In this “coupled” method, information from all m bands of image u is used to compute the above matrix. Then, denote the eigenvalues and eigenvectors of the aforementioned matrix as λ_1, λ_2 and v_1, v_2 respectively. Finally, the tensor matrix D can be formed,

$$D = \begin{pmatrix} v_{11} & v_{12} \\ v_{21} & v_{22} \end{pmatrix} \begin{pmatrix} g(\lambda_1) & 0 \\ 0 & g(\lambda_2) \end{pmatrix} \begin{pmatrix} v_{11} & v_{12} \\ v_{21} & v_{22} \end{pmatrix}^{-1}. \quad (7)$$

A fidelity term can also be used in the “coupled” version of anisotropic diffusion,

$$\frac{\partial u_i}{\partial t} = \nabla \bullet (D(u) \nabla u_i) - \alpha(u_i - f_i), \quad i = 1, \dots, m. \quad (8)$$

Implementation

MATLAB was used to implement both the “separate” and “coupled” versions of this anisotropic diffusion routine. MATLAB is not known for its speed; however, since the routine works reliably in MATLAB, it should also work reliably using a low level language such as C.

To easily solve a differential equation $f(t, y)$, an iterative numerical method such as Euler’s Method can be used,

$$y_{n+1} = y_n + hf(t_n, y_n).$$

The equation for anisotropic diffusion would act as $f(t, y)$. To obtain $f(t, y)$ for each step, operations would have to be performed on the image, pixel by pixel. Finite differences can be

used to obtain matrices representing $\frac{\partial u}{\partial x}$ and $\frac{\partial u}{\partial y}$. Below is an example of computing finite differences in MATLAB, where u is the image,

```
[m,n] = size(u);
u_x = (u(:, [2:n n]) - u(:, [1 1:n-1]))/2;
u_y = (u([2:m m], :) - u([1 1:m-1], :))/2;
```

To find matrices representing second derivatives (which is necessary to find $\nabla \bullet \nabla u$), finite differences can be applied a second time.

For the edge-stopping function $g(\xi)$, (2) is used in this paper's implementation.

When using a numerical method to solve a differential equation, there are a few issues to consider. One important issue is to consider is what the stopping condition should be. In this paper, we take $\|u_n - u_{n-1}\|$ as the difference. Intuitively, this value represents the amount of pixels that are modified between two iterations. After each iteration, $\|u_n - u_{n-1}\|$ should be expected to be significant as anisotropic diffusion is performed because many pixels should be changing. However, as anisotropic diffusion progresses further $\|u_n - u_{n-1}\|$ will be small enough such that the effects of anisotropic diffusion will be negligible. At that point, it would be advisable to stop. In this paper's implementation, iteration is stopped when $\|u_n - u_{n-1}\|$ goes beneath 1/100 of one percent of the total number of pixels in one band of an image. The other issue is determining a suitable timestep. Although using a small timestep may result in more accurate results, it will cause long computation time, so a somewhat large timestep should be desirable. A timestep should be large, but small enough that it does not result in instability of the differential equation.

Results

The following results shown in this paper were achieved using a computer with an Intel Pentium D 3.2 GHz processor with 2 GB of RAM.

The timestep used is 0.65.

Pines and Urban hyperspectral data are used in these experiments. However, both those datasets contain bands with noise. The noisy bands were removed to prevent noise from skewing results. In the Pines dataset, bands 1-4, 103-110, 149-165, and 216-220 were removed. In the Urban dataset, bands 104-109, 139-153, and 206-210 were removed.

Image segmentation is performed using K-means cosine clustering. For the pines data, seven clusters are taken and for the urban data, four clusters.

The following results may show that "separate" anisotropic diffusion gets better results than "coupled" anisotropic diffusion when time is considered. However, "coupled" anisotropic diffusion can be improved by choosing smaller values of α . In "coupled" anisotropic diffusion, information from all the bands is used to perform anisotropic diffusion on each band, so it is easier for an image to keep its structural integrity with a very small fidelity term. Unfortunately, the same is not true for "separate" anisotropic diffusion as making α very small will blur the image beyond repair.

Separate Anisotropic Diffusion on Pines

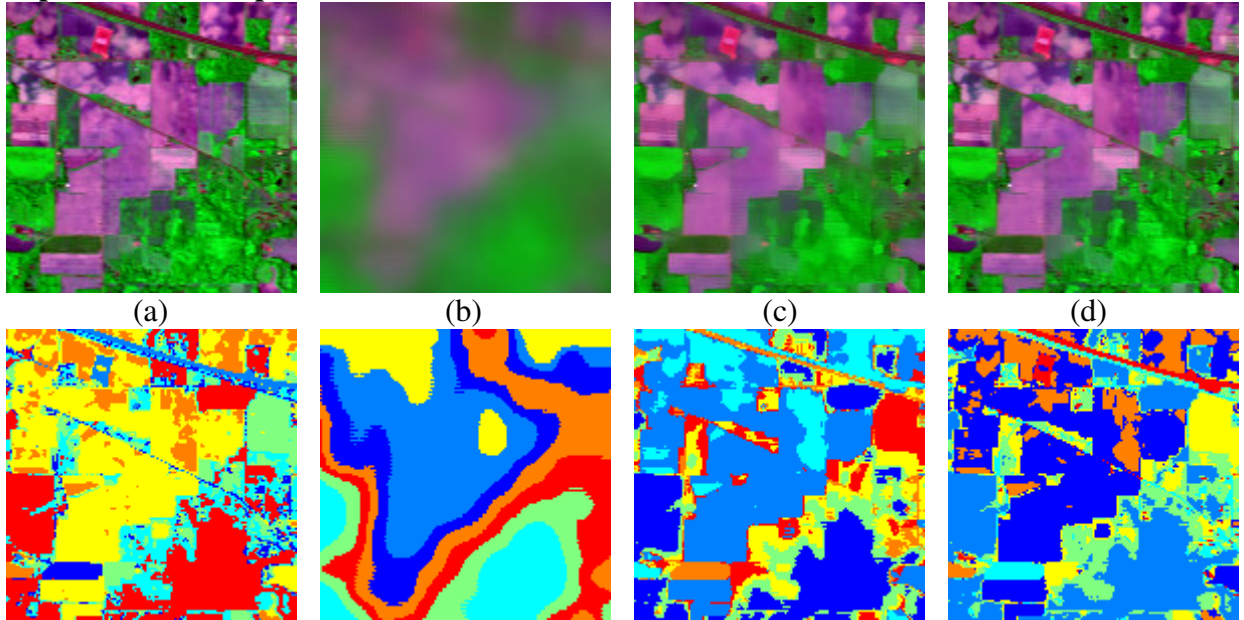


Figure 1. Top. (a) Original (b) $p = 250$, $\alpha = 0.01$ (c) $p = 250$, $\alpha = 0.25$ (d) $p = 250$, $\alpha = 0.75$
Figure 2. Bottom. Results of K-means cosine clustering with 7 clusters.

In Figure 1b, it is evident that setting the fidelity parameter α too low causes excess blurring. When comparing the original image to figure 1c or 1d, it is noticeable that some extraneous details are successfully blurred out. Although segmentation is nowhere near perfect, there exists improvement from the original to Figures 2c and 2d. Unfortunately, Figure 2b is rubbish.

$p \setminus \alpha$	0.01	0.05	0.1	0.25	0.5	0.75
100	510.54	152.26	67.88	25.53	12.46	8.17
175	294.73	126.85	64.29	26.32	12.69	8.63
250	230.79	116.42	65.77	25.91	13.04	8.97

Figure 3. Elapsed times of execution for various values of p and α .

Examining the runtimes of the procedure, in this specific case, it appears lower runtimes were more effective. It takes 231 seconds to get Figure 1b while it takes 26 seconds to get Figure 1c.

Coupled Anisotropic Diffusion on Pines

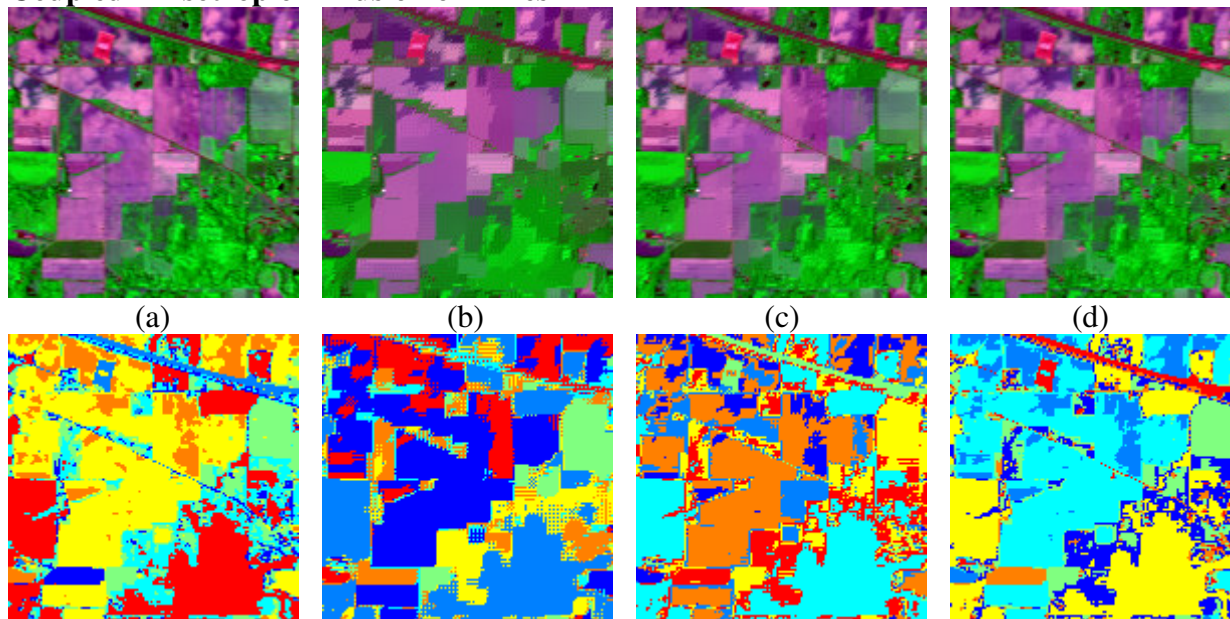


Figure 4. Top. (a) Original (b) $p = 250$, $\alpha = 0.01$ (c) $p = 250$, $\alpha = 0.05$ (d) $p = 250$, $\alpha = 0.1$
Figure 5. Bottom. Results of K-means cosine clustering with 7 clusters.

For the Pines dataset, “coupled” anisotropic diffusion appears to yield more distinct results, as opposed to “separate” anisotropic diffusion. Unfortunately, after segmentation, boundaries appear very unnatural as lattices appear in the segmentation result. In addition, very small values of α were necessary to get sufficient results as segmentation results for Figure 4d are not vastly improved from the original.

$p \setminus \alpha$	0.01	0.05	0.1	0.25	0.5	0.75
100	5478.64	497.76	140.38	36.87	17.07	11.27
175	5635.74	1084.42	218.68	45.08	19.39	12.42
250	4919.31	944.93	249.87	52.03	20.54	13.62

Figure 6. Elapsed times of execution for various values of p and α .

It is necessary to set a very small α , but computation time becomes very long doing so. However, the long computation time yields sharp distinct boundaries in the results.

These results demonstrate one flaw “coupled” anisotropic diffusion has, computation time. In “coupled” anisotropic diffusion, a 2×2 matrix is created for each pixel of each band using information from all of the bands of the image. Memory and processor speed prove to be important issues in these dense calculations.

Segmentation of the Pines dataset is an example of segmenting distinct, contiguous regions; however, anisotropic diffusion of an urban setting proves to be more challenging.

Separate Anisotropic Diffusion on Urban

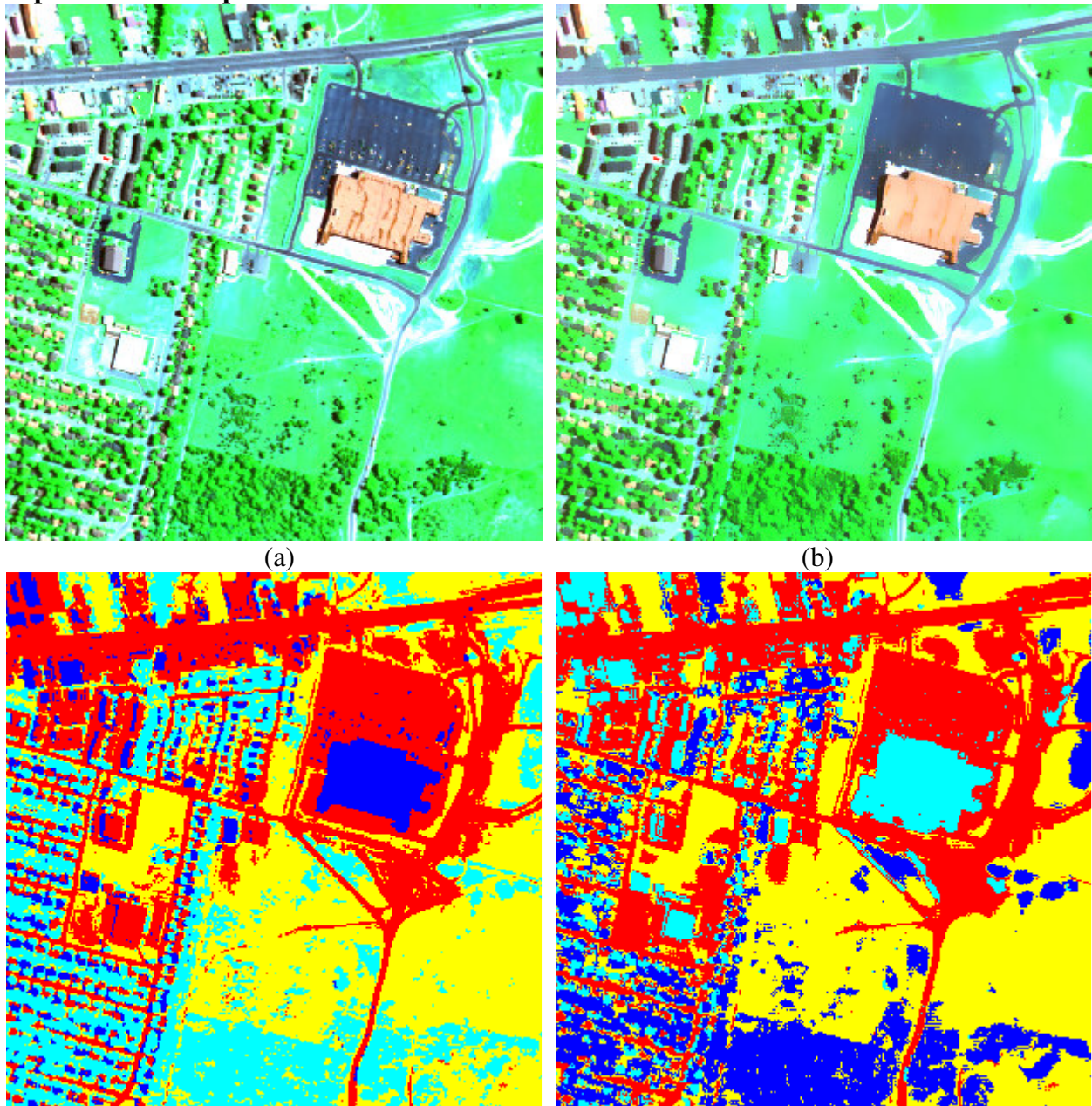


Figure 7. Top. (a) Original Image (b) $p = 100$, $\alpha = 0.1$

Figure 8. Bottom. Results of K-means cosine clustering with 4 clusters.

Diffusion of this image lasted 463.20 seconds. Since this image has much detail, there are few differences between the original and the diffused. Differences are most notable on the roof of the building.

Coupled Anisotropic Diffusion on Urban

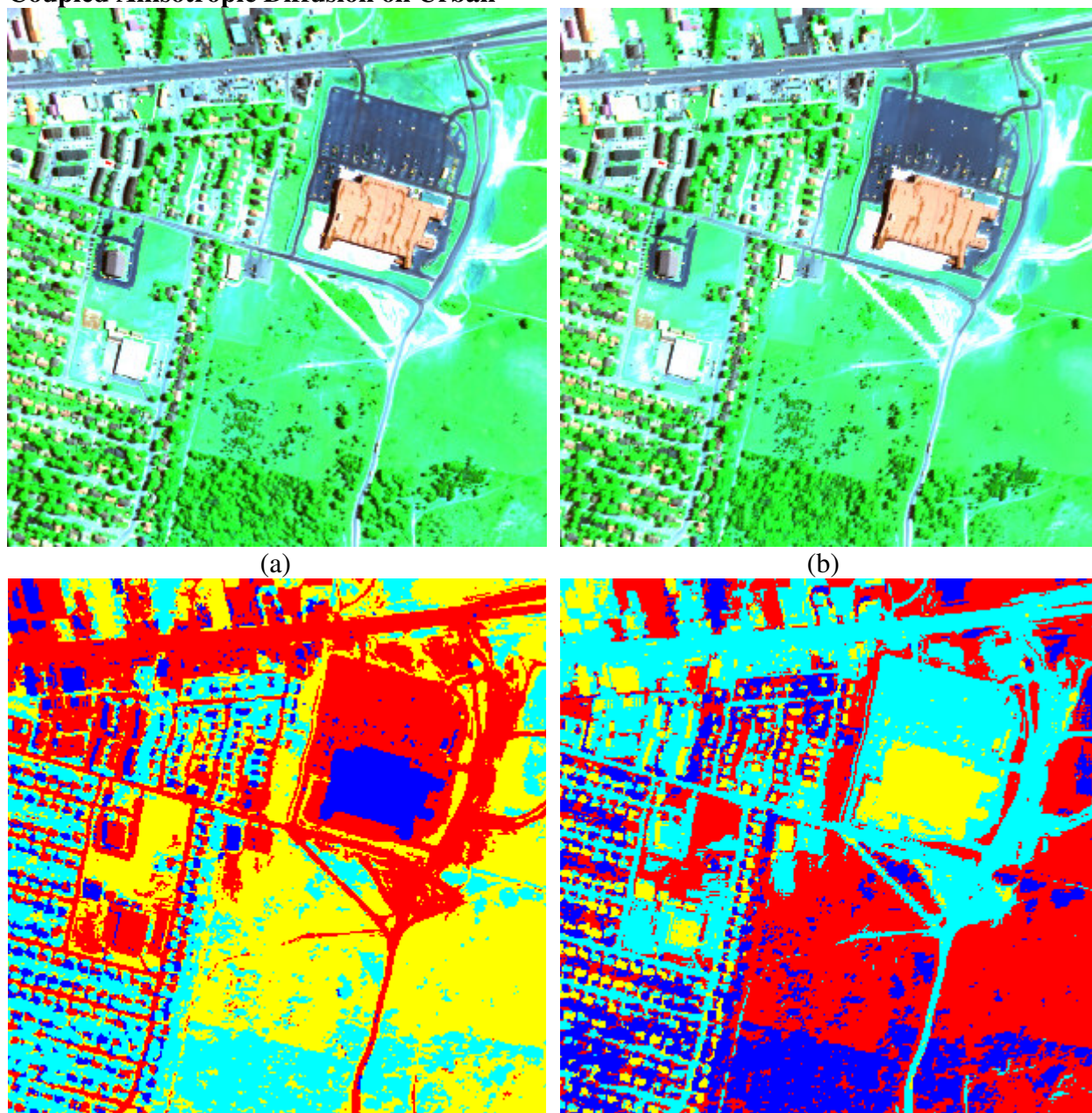


Figure 9. Top. (a) Original Image (b) $p = 750$, $\alpha = 0.1$

Figure 10. Bottom. Results of K-means cosine clustering with 4 clusters.

Elapsed time of computation was 1067.61 seconds. Results for “coupled” anisotropic diffusion on the Urban dataset is not overly impressive as diffusion of the image is barely noticeable. Also, an unfortunate side effect is that segmentation is worse on the diffused image than on the original.

However, there is some good news. A large α was used to retrieve this result. Using smaller values of α will improve results.

References

1. P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **12**(7):629-639, 1990.
2. J. Duarte-Carvajalino, G. Sapiro, M. Velez-Reyes, and P. Castillo, "Fast Multi-Scale Regularization and Segmentation of Hyperspectral Imagery via Anisotropic Diffusion and Algebraic Multigrid Solvers," *Proc. SPIE*, **6565**, 2007.
3. T. Brox and J. Weickert, "Diffusion and Regularization of Vector- and Matrix-Valued Images," *Inverse Problems, Image Analysis and Medical Imaging. Contemporary Mathematics*, **313**:251-268, 2002.