CONTACT NAME
Peter Yoon: peter.yoon@trincoll.edu



# GPU Accelerated Vessel Segmentation Using Laplacian Eigenmaps

Lin Cheng, Hyunsu Cho, Peter Yoon, and Jiajia Zhao
Trinity College, Hartford, CT 06106

## 1. Abstract

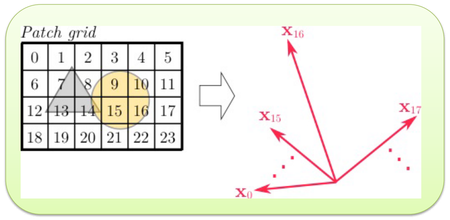
Laplacian eigenmap is a useful technique to improve cluster-based segmentation of multivariate images. However, this approach requires an excessive amount of computations when processing large image datasets. To that end, we present a GPU-based acceleration procedure for vessel segmentation problems.

## 2. Laplacian Eigenmap

As described in Laskaris et. al. [1], the Laplacian Eigenmap is an effective dimensionality reduction method that maps a set of multivariate *features vectors* to points on the real line. Each features vector characterizes a group of neighboring pixels referred to as a *patch*. Once the projection is completed, a conventional method can be used to classify the projected points into one or more clusters.

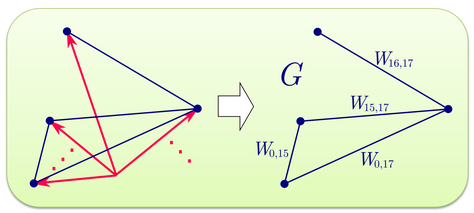
### 1 Generate patches

- Let $\mathbf{x}_i$ be the multi-dimensional vector representing the $i^{th}$ patch.




### 2 Generate the weight matrix *W*

- Build a weighted graph whose nodes are the terminals of the vectors. Let *W* be the associated weight matrix.




- Characteristics of *W*
  - Two endpoints that are close enough should be connected with an edge. → *W* is symmetric.
  - An edge carries more weight as its endpoints are far apart.

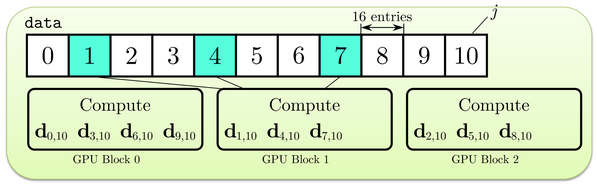
$$W_{ij} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{M\alpha^2}\right)\exp\left(-\frac{\|\mathbf{p}_i - \mathbf{p}_j\|^2}{2\beta^2}\right)$$

- `diff_square()`
  - Compute "difference vectors" $\mathbf{d}_{ij}$ and $\mathbf{f}_{ij}$.

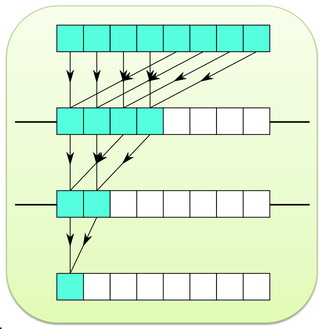
$$\mathbf{d}_{ij}(k) = (\mathbf{x}_i(k) - \mathbf{x}_j(k))^2$$

$$\mathbf{f}_{ij}(k) = (\mathbf{p}_i(k) - \mathbf{p}_j(k))^2$$

  - Divide labor among GPU blocks.
    - Fix index *j* and vary *i* from 0 to *N* – 1, where *N* is the number of features vectors.
    - There is no data dependency between any two difference vectors.




- `reduce()`: *binary reduction*
  - Use multiple GPU threads to compute partial sums in the shared memory.
  - Synchronization is required.

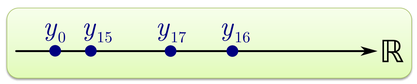


### 3 Compute Laplacian matrix *L*

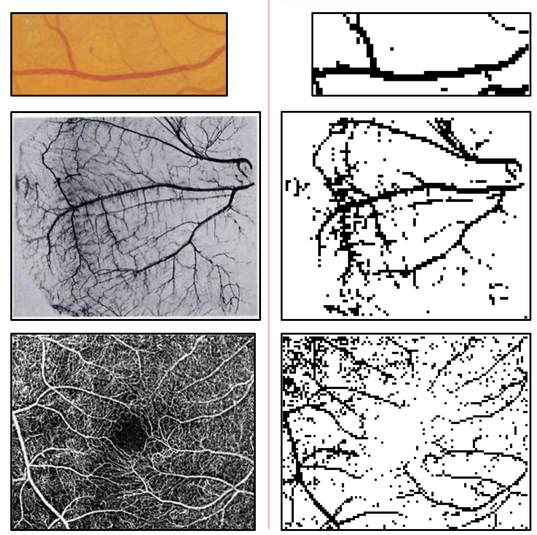
- $L = I - D^{1/2}WD^{1/2}$
  - *D* is a diagonal matrix where each entry is the sum of the corresponding column of *W*.
- Use `cublasDsymm()` for a matrix multiplication.

### 4 Solve the generalized eigenvector problem

- A solution $\mathbf{y} = (y_0, y_1, \cdots, y_{N-1})$ to the generalized eigenvector problem $L\mathbf{y} = \lambda D\mathbf{y}$ defines the map.
- Use `magma_dsyevd()` in MAGMA.
  - Compute all the eigenvalues and eigenvectors of a symmetric matrix.
- Components of **y** are the projections of the multivariate features vectors onto the real line.



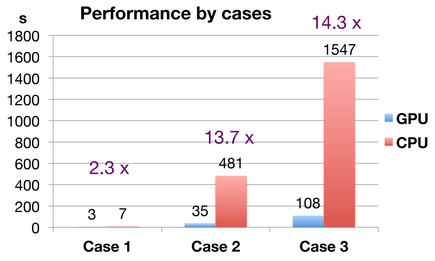

## 3. Results and Performance

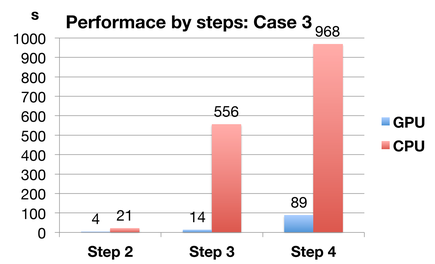



Case 1
A small test vessel image
(256 x 96)

Case 2
Capillary pattern in skeletal muscle
(360 x 300)

Case 3
Capillary pattern in retina
(448 x 352)




- Not only does the framework yield satisfactory segmentation results, but it also presents a significant performance gain, a speedup of 14x for the largest test case.
- Solving the generalized eigenvector problem is the most time-consuming part of the entire segmentation procedure.

**Testing Environment**
- 1 Nvidia Tesla C2050, 3GB memory
- 2 quad-core 2.4GHz Intel Xeon CPU, 12 GB memory
- CUDA 5.0 running on Linux OS

**Reference**
[1] N. Laskaris et. al. "Multivariate image segmentation using Laplacian eigenmaps." EUSIPCO 2004: Proc. of Euro. Signal Processing Conf: 2004.
[2] J. Schmid et. al. "A GPU framework for parallel segmentation of volumetric images using discrete deformable models." Visual Computer: 2011