Neurodata Algorithms

Jason Yim

October 14, 2017

Abstract

Collection of algorithm pseudocodes, math, proofs, descriptions, etc..

Single Pass Voting

Region-based hierarchical voting in a distance transform map

Proposed by Xing in "Automatic Ki-67 Counting Using Robust Cell Detection and Online Dictionary Learning" as an improvement to the original SPV algorithm. Previous SPV methods localize seeds witha gradient-madnitude weighted majority vote but it is not able to handle cell size and shape variationa very well. It is also sensitive to noise. This method attempts to fix that.

First the variables. Let x, y be the pixel location on the x, y plane and ℓ is the ℓ-th layer of the image:

T	The original image
$\nabla T_\ell(x, y)$	The image gradient
$A_\ell(x, y)$	The cone-shaped voting area
$\frac{-\nabla T(x, y)}{\|\nabla T(x, y)\|}$	Negative gradient direction
$\theta(x, y)$	Angle of the gradient direction
$V_\ell(x, y)$	Confidence map
$g(x, y, \mu_x, \mu_y, \sigma)$	2-variable Gaussian kernel with shared variance
$C_\ell(x, y)$	Distance transform map

More in-depth expressions are given below:

$$\frac{-\nabla T(x, y)}{\|\nabla T(x, y)\|} = -(\cos(\theta(x, y)), \sin(\theta(x, y)) \tag{1}$$

note the angle θ is with respect to the x-axis. The confidence map is calculated as the weighing of the distance transform map with the gaussian kernel:

$$V_\ell(x, y) = \sum_{\ell=0}^{L} \sum_{(m,n)\in S} I[(x, y) \in A_\ell(m, n)] \;\; C_\ell(m, n) \;\; g(m, n, \mu_x, \mu_y, \sigma) \tag{2}$$

where $I(s)$ is the indicator function that returns 1 if s is non-empty and 0 otherwise. S is the set of all voting pixels around the pixel. We can simply choose the euclidean distance transform as C. The voting area of the cone is parameterized by $(r_{\min}, r_{\max})$ which define how short and long we will do our voting and $\Delta \in [0, 2\pi]$ which determines the angle range of the search. The gaussian kernel is parameterized off the voting parameters where

$$(\mu_x, \mu_y) = (x + \frac{(r_{\max} - r_{\min})\cos(\theta)}{2}, y - \frac{(r_{\max} - r_{\min})\sin(\theta)}{2}) \tag{3}$$

The basic algorithm outline is as follows:

1. Calculate the euclidean distance map $C_\ell(x, y)$
2. Calculate the cone shaped voting area $A_\ell(x, y)$ for each pixel
3. For each pixels (x, y) in each layer
   (a) Count the number of pixels in its neighborhood S with voting areas that include (x, y)
   (b) Compute the gaussian kernel at this point
   (c) Set the product of the gaussian kernel, count from (b), and EDT as its confidence map value
4. Run mean-shift clustering to calculate the centers.