Homework 6

Wednesday, March 17, 2021 6:53 PM

Equation 1: $C(i,j) = (g(i,j) + g(j,i))^6$ where $g(i,j) = \exp\left(\frac{-\text{grad}_{ij}(i)}{\max(\text{grad}_{\cdot}(k))}\right)$ and $\text{grad}_{ij}(k)$ is the magnitude of image pixel intensity gradient at location k in the direction $i \to j$

Equation 2: $V' = V - V_s + v$

Equation 3: $E' = E - E^{c}(V_s) + E_v$ where $E(V_s) = \bigcup_{v_1 \in V_s} \vee_{v_2 \in V_s} \{(v_1, v_2)\}$ and $E(v) = \bigcup_{u \in V - V_s} \{(u,v), (v,u)\}$ with capacity $c'(v,u) = \sum_i c(v_i, u)$ and $c'(u,v) = \sum_i (u, v_i)$

Equation 4: $s \in \bar{S}$ where $\bar{S} = \hat{S} + \{s\} - S$

Equation 5: $t \in \bar{T}$ where $\bar{T} = \hat{T} + \{t\} - T$

Equation 6: $E_c =$ capacity of $MC(G_i, s_i, t_i) = \sum_{u \in \bar{S}} \sum_{v \in \bar{T}} c(u,v)$

Equation 7: $A \oplus B = \bigcup_{b \in B} A_b$ where A_b is the translation of A by b

```
function image_to_graph(image):
  for each x in image_width:
     for each y in image_height:
        map image[x][y] to a vertex in the image adjacency graph

  for each vertex in V:
    find 8-neighborhood adjacent pixels:
    for each of the 8-neighborhood vertices:
       Connect the current vertex via an undirected edge with weight C(current vertex, neighborhood pixel) using equation 1

  return G(V, E) where G is the image represented as an adjacency graph


function dialate_contour(contour):

    initialize step_size
    if object is large and image is noisy:
       step_size is large
   else if object is small and image is noisy:
       step_size is large
   else:
       step_size is small

  dialate the given contour using equation 7 and obtain the source and sink (inner countour and outer countour) with the obtained step size

  return source, sink


function compute_MC(diallated graph(G_i, s_i, t_i), and contour c_i)
   //node identification to simplify graph and remove cycles
   // Convert G_i(V, E) to G_i'(V', E')

   Convert G_i to G_i' by using equation 2 to compress sets of vertices into one vertex
   and by using equation 3 to convert the edges and compute new edge weights

   Convert initial S and T to S̄ and T̄ using equations 4 and 5
   Calculate the capacity E(c) of the new node identified graph


   return argmin_{c ∈ CN(c_i)} E(c)

function GCBAC(image, initial contour c_0):
   G(V, E) = image_to_graph(image)
   current step i = 0

   while no repeating contours occur:

        source, sink = dialate_contour(c_i)

        c_{i+1} = compute_MC(G_i, source, sink, c_i)

        i = i+1

  return c_i
```