Jerry Chen

CS558

Computer Vision

Prof. Dunn

HW 3

For k means, picked 10 random triplets as seeds and computed the distance between the 10 points and every other point on the image. Picked the minimum distance to assign each point to a center and then recomputed new center using the average. Repeated until centers have converged and stopped changing after iteration. For slic, initialized with 50\*50 blocks and shifted centers in 3\*3 windows using least gradient. Then computed 5d distance for every point on the image and assigned each to its centroid with least distance. Repeated until convergence or max iterations.

A picture containing graphical user interface

Description automatically generated

A picture containing photo, person

Description automatically generated

%run k-means on image

I = imread('white-tower.png');

I = im2double(I);

k = 10;

[RGB\_map,Seg] = kmeans(I,k);

montage({I,Seg})

title("Original Image Vs. Segmented Image")

%slic segmentation

Im = imread('wt\_slic.png');

Im = im2double(Im);

max\_it = 3;

[li,slic\_seg] = slic(Im,max\_it);

montage({slic\_seg})

title("SLIC")

function [RGB\_map,seg] = kmeans(img,k)

%UNTITLED Summary of this function goes here

% Detailed explanation goes here

seg = img;

%randomly choose 10 points

[ht,wd,ch] = size(img);

x = randi(ht,1,k);

y = randi(wd,1,k);

RGB\_map = zeros(k,ch);

new\_RGB = zeros(k,ch);

map = zeros(ht,wd,k);

for i = 1:k

RGB\_map(i,:) = img(x(i),y(i),:);

end

converging = true;

while converging

%calculate distance

for i = 1:k

for h = 1:ht

for w = 1:wd

map(h,w,i) = sqrt((img(h,w,1)-RGB\_map(i,1)).^2+(img(h,w,2)-RGB\_map(i,2)).^2+(img(h,w,3)-RGB\_map(i,3)).^2);

end

end

end

[~,I] = min(map,[],3);

%new center

for i = 1:k

count = 0;

for h = 1:ht

for w = 1:wd

if I(h,w) == i

new\_RGB(i,1) = new\_RGB(i,1) + img(h,w,1);

new\_RGB(i,2) = new\_RGB(i,2) + img(h,w,2);

new\_RGB(i,3) = new\_RGB(i,3) + img(h,w,3);

count = count + 1;

end

end

end

new\_RGB(i,:) = new\_RGB(i,:)/count;

end

difference = abs(RGB\_map-new\_RGB);

%check if converged

if sum(difference) < 0.05

converging = false;

%produce segmented image

for i = 1:k

for h = 1:ht

for w = 1:wd

if I(h,w) == i

seg(h,w,:) = RGB\_map(i,:);

end

end

end

end

else

RGB\_map = new\_RGB;

end

end

end

function [I\_map,seg] = slic(img,max\_it)

%UNTITLED3 Summary of this function goes here

% Detailed explanation goes here

[ht,wd,~] = size(img);

seg = img;

%Divide into 50\*50 blocks

b\_num = (ht\*wd)/2500;

%[x,y,R,G,B]

c\_map = zeros(b\_num,5);

converging = true;

it = 1;

for i = 1:ht/50

x = 25+50\*(i-1);

for j = 1:wd/50

y = 25+50\*(j-1);

idx = 15\*(i-1)+j;

c\_map(idx,1) = x;

c\_map(idx,2) = y;

c\_map(idx,3:5) = img(x,y,:);

end

end

while converging

%compute gradient

for j = 1:b\_num

window = zeros(3,3);

for x = -1:1

for y = -1:1

window(x+2,y+2) = rgb\_grad(c\_map(j,1)+x,c\_map(j,2)+y,img);

end

end

[minw,idx] = min(window);

[~,miny] = min(minw);

minx = idx(miny);

c\_map(j,1:2) = c\_map(j,1:2) + [minx-2,miny-2];

c\_map(j,3:5) = img(c\_map(j,1),c\_map(j,2),:);

end

d\_map = zeros(ht,wd,b\_num);

for i = 1:b\_num

for h = 1:ht

for w = 1:wd

d = c\_map(i,:);

d\_vect = [(h-d(1))/2,(w-d(2))/2,(img(h,w,1)-d(3)),(img(h,w,2)-d(4)),(img(h,w,3)-d(5))];

d\_map(h,w,i) = norm(d\_vect);

end

end

end

[~,I\_map] = min(d\_map,[],3);

newc\_map = zeros(b\_num,5);

cluster\_rgb = zeros(b\_num,3);

for i = 1:b\_num

count = 0;

for h = 1:ht

for w = 1:wd

if I\_map(h,w) == i

rgb = img(h,w,:);

newc\_map(i,1) = newc\_map(i,1) + h;

newc\_map(i,2) = newc\_map(i,2) + w;

cluster\_rgb(i,1) = cluster\_rgb(i,1) + rgb(1);

cluster\_rgb(i,2) = cluster\_rgb(i,2) + rgb(2);

cluster\_rgb(i,3) = cluster\_rgb(i,3) + rgb(3);

count = count + 1;

end

end

end

newc\_map(i,1:2) = floor(newc\_map(i,1:2)/count);

colors = img(newc\_map(i,1),newc\_map(i,2),:);

newc\_map(i,3:5) = colors;

cluster\_rgb(i,:) = cluster\_rgb(i,:)/count;

end

eq = isequal(c\_map,newc\_map);

if ~eq && it < max\_it

c\_map = newc\_map;

it = it + 1;

else

converging = false;

for i = 1:b\_num

for h = 1:ht

for w = 1:wd

if I\_map(h,w) == i

if border(I\_map,h,w)

seg(h,w,:) = [0,0,0];

else

seg(h,w,:) = cluster\_rgb(i,:);

end

end

end

end

end

end

end

end

function grad = rgb\_grad(x,y,img)

%UNTITLED4 Summary of this function goes here

% Detailed explanation goes here

r\_g = norm(img(x+1,y,1)-img(x-1,y,1),img(x,y+1,1)-img(x,y-1,1));

g\_g = norm(img(x+1,y,2)-img(x-1,y,2),img(x,y+1,2)-img(x,y-1,2));

b\_g = norm(img(x+1,y,3)-img(x-1,y,3),img(x,y+1,3)-img(x,y-1,3));

grad = norm([r\_g,g\_g,b\_g]);

end

function bool = border(i\_map,x,y)

%UNTITLED Summary of this function goes here

% Detailed explanation goes here

bool = false;

x\_range = x+1;

y\_range = y+1;

[ht,wd] = size(i\_map);

if x\_range > ht

x\_range = ht;

end

if y\_range > wd

y\_range = wd;

end

pixel1 = i\_map(x\_range,y);

if pixel1 ~= i\_map(x,y)

bool = true;

end

pixel2 = i\_map(x,y\_range);

if pixel2 ~= i\_map(x,y)

bool = true;

end

end