Run kmeans_tower.m

Time:1min–

```
    66    93   102
   119   127   127
    24    21    16
   127   127   107
   127   127   127
    97    86    64
    96   122   127
    80   107   120
   127   127   127
    64    58    44
```

Input RGB matrix, random start points k and iterators can end.

Out put clusters and centers of them.

First, sample k number to select starting point.

Second, calculate all points to these centers. Meanwhile, choose the nearest center to cluster.

Third, update each cluster. Update their new center as the average of them.

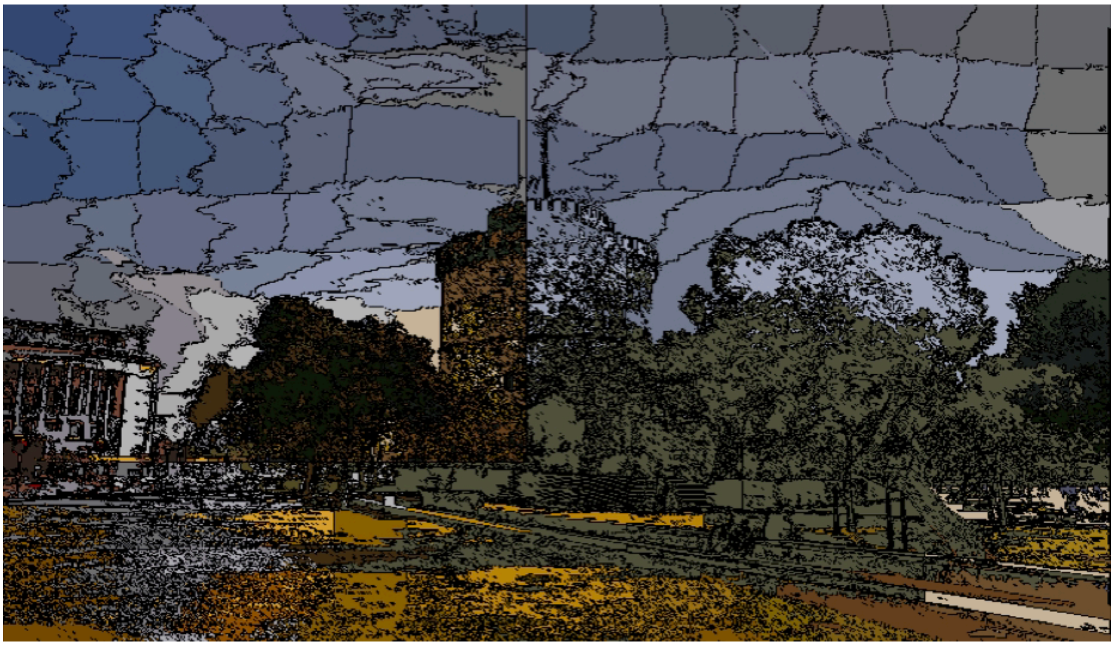
After converge or reach the max iters, return the function.

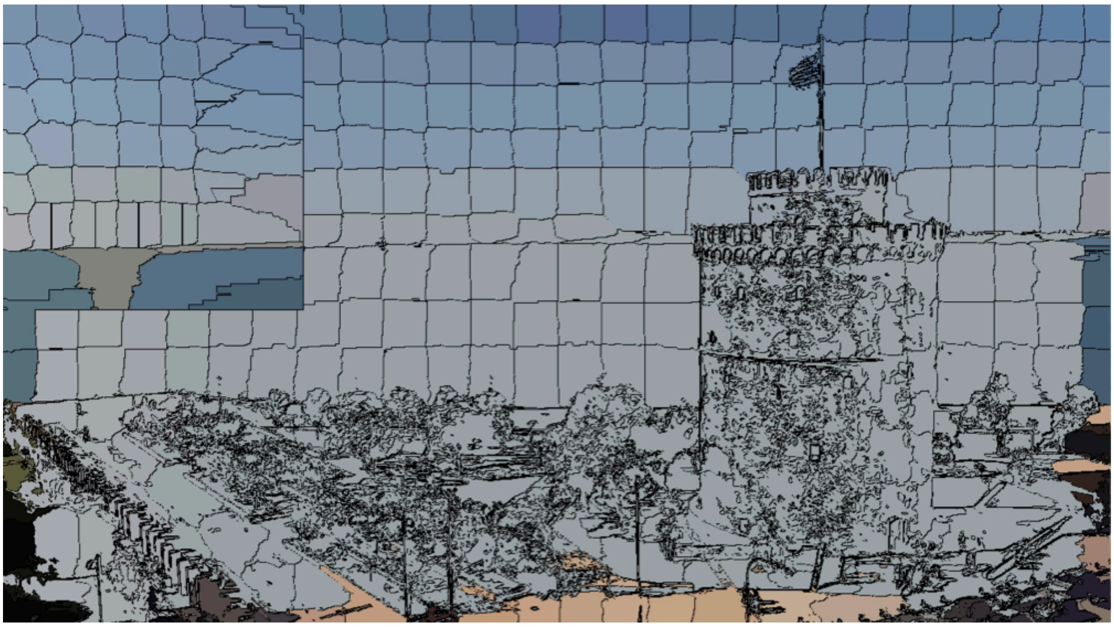
Run slic_tower.m

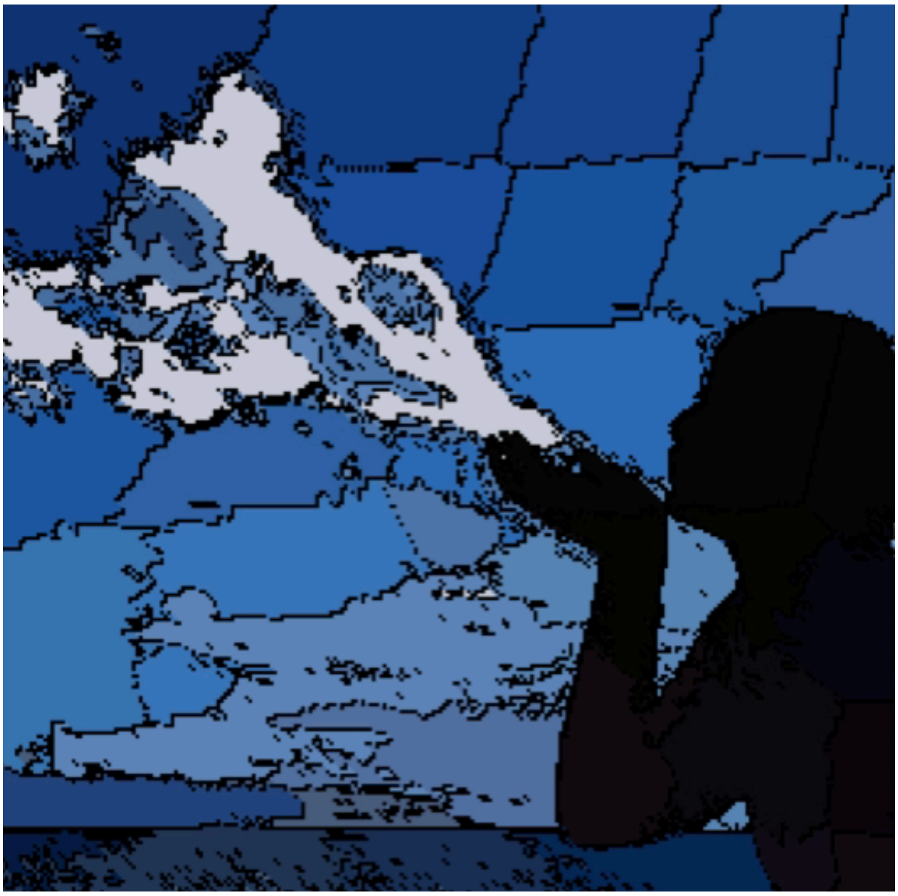
White_towel.jpg Time:30min+

Wt.jpg Time:10min+

1.jpg Time:5min+







Input the path of image.

Output the image after deal.

First, init the param like height and width. Create 50*50 block in order. Init the center of them.

Second, use a 3*3 slide window to update the center by gradient of RGB value.

Third, calculate each pixel to cluster.

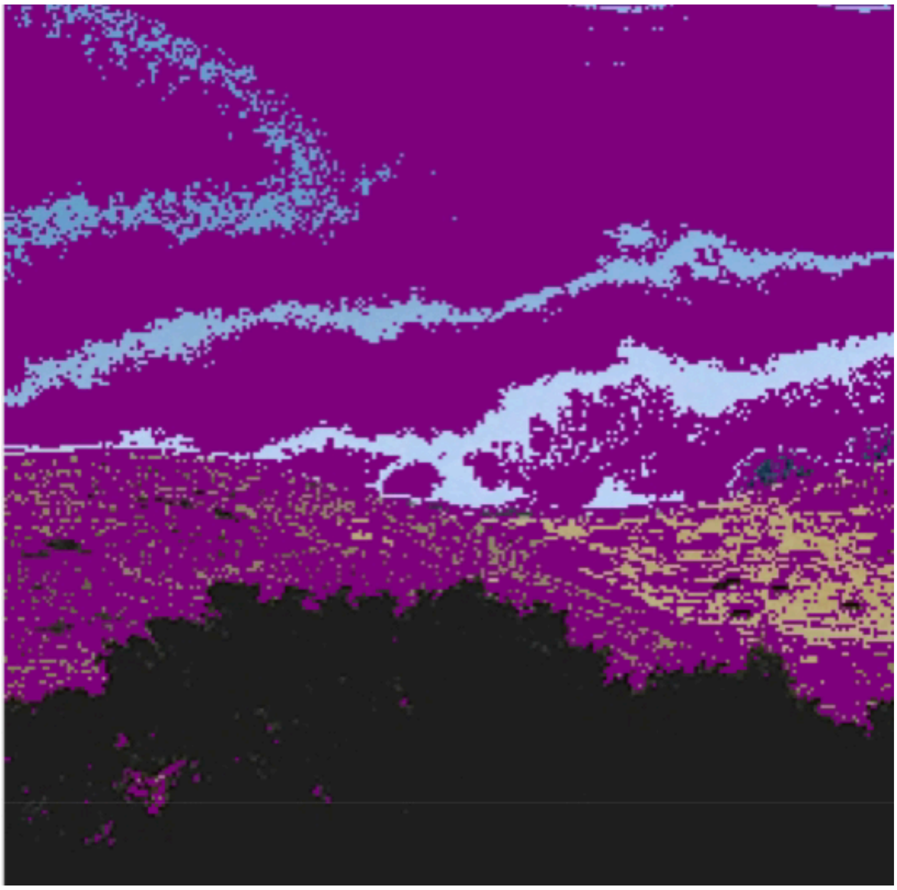
Forth, at the same time, exclude section 100 pixel away.

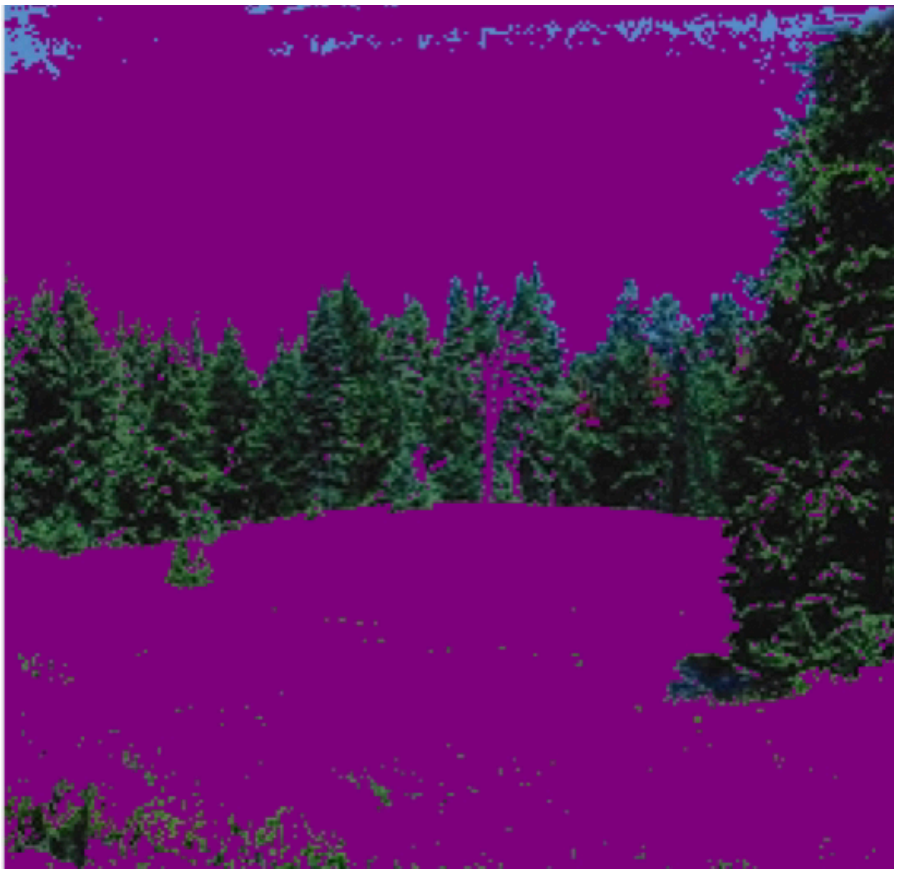
Fifth, judge if converge and increate iter.

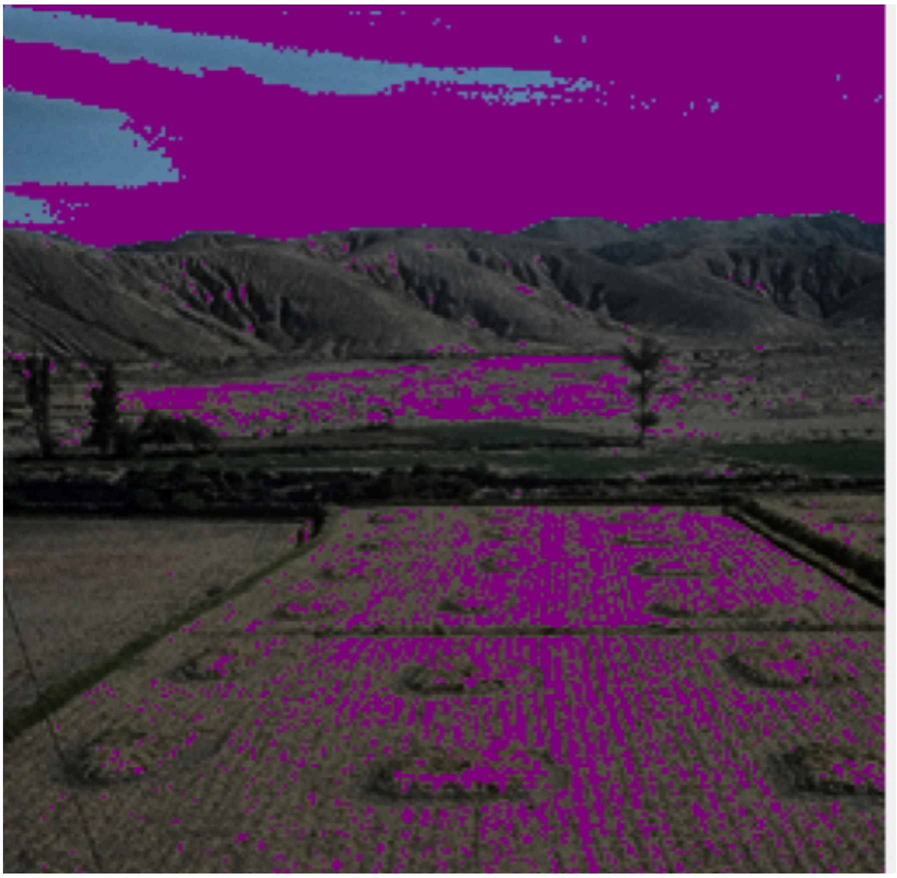
Lastly, paint the border to blank, otherwise the average color.

Run classification.m

Time:5min+







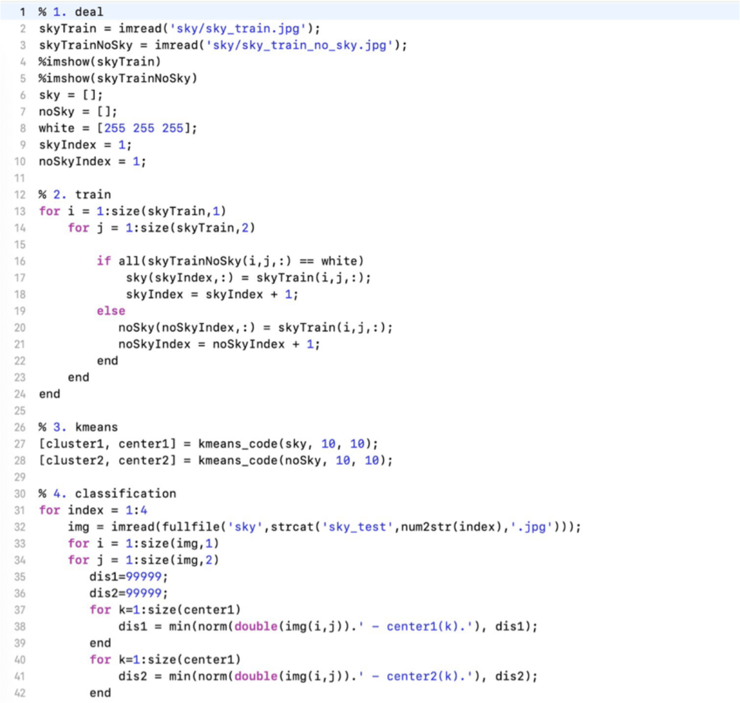
First, remove the sky of the picture.

Second, add the sky part of picture to one kmeans process, otherwise add to another.

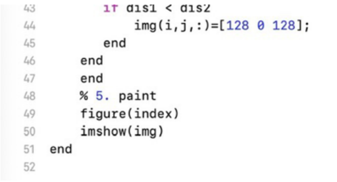
Third, train them by kmeans.

Forth, scan the newly picture. If one pixel is close to sky clusters, then paint it to purple. Otherwise, do nothing.

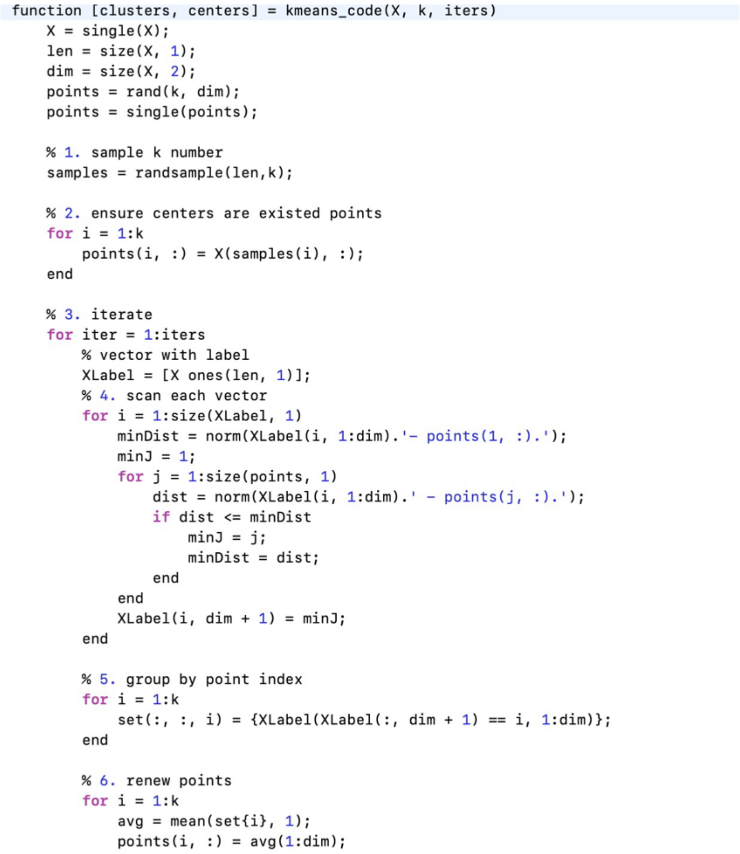
Code



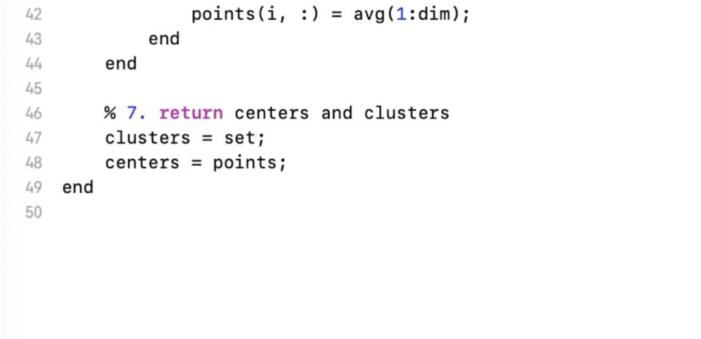
```
% 1. deal
skyTrain = imread('sky/sky_train.jpg');
skyTrainNoSky = imread('sky/sky_train_no_sky.jpg');
%imshow(skyTrain)
%imshow(skyTrainNoSky)
sky = [];
noSky = [];
white = [255 255 255];
skyIndex = 1;
noSkyIndex = 1;

% 2. train
for i = 1:size(skyTrain,1)
    for j = 1:size(skyTrain,2)

        if all(skyTrainNoSky(i,j,:) == white)
            sky(skyIndex,:) = skyTrain(i,j,:);
            skyIndex = skyIndex + 1;
        else
           noSky(noSkyIndex,:) = skyTrain(i,j,:);
           noSkyIndex = noSkyIndex + 1;
        end
    end
end

% 3. kmeans
[cluster1, center1] = kmeans_code(sky, 10, 10);
[cluster2, center2] = kmeans_code(noSky, 10, 10);

% 4. classification
for index = 1:4
    img = imread(fullfile('sky',strcat('sky_test',num2str(index),'.jpg')));
    for i = 1:size(img,1)
    for j = 1:size(img,2)
       dis1=99999;
       dis2=99999;
       for k=1:size(center1)
           dis1 = min(norm(double(img(i,j)).' - center1(k).'), dis1);
       end
       for k=1:size(center1)
           dis2 = min(norm(double(img(i,j)).' - center2(k).'), dis2);
       end
```



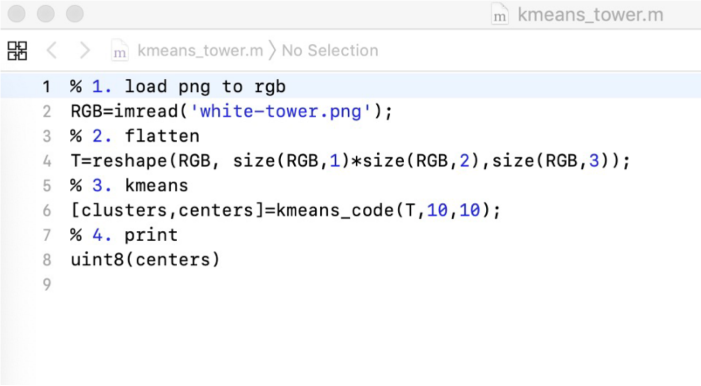
```
       if dis1 < dis2
           img(i,j,:)=[128 0 128];
       end
    end
    end
    % 5. paint
    figure(index)
    imshow(img)
end
```



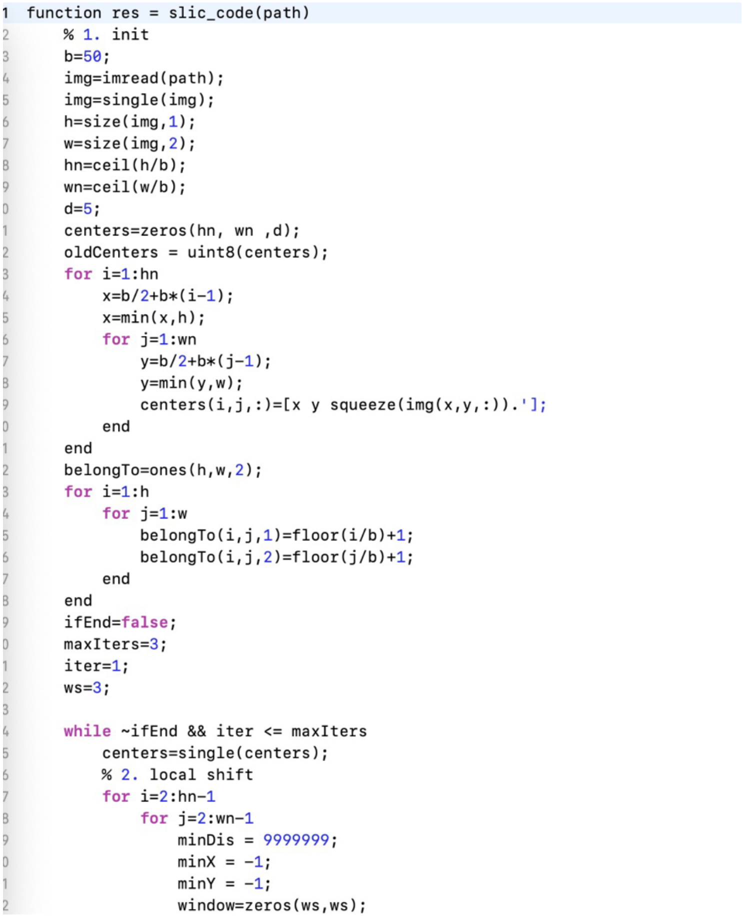
```
function [clusters, centers] = kmeans_code(X, k, iters)
    X = single(X);
    len = size(X, 1);
    dim = size(X, 2);
    points = rand(k, dim);
    points = single(points);

    % 1. sample k number
    samples = randsample(len,k);

    % 2. ensure centers are existed points
    for i = 1:k
        points(i, :) = X(samples(i), :);
    end

    % 3. iterate
    for iter = 1:iters
        % vector with label
        XLabel = [X ones(len, 1)];
        % 4. scan each vector
        for i = 1:size(XLabel, 1)
            minDist = norm(XLabel(i, 1:dim).'- points(1, :).');
            minJ = 1;
            for j = 1:size(points, 1)
                dist = norm(XLabel(i, 1:dim).' - points(j, :).');
                if dist <= minDist
                    minJ = j;
                    minDist = dist;
                end
            end
            XLabel(i, dim + 1) = minJ;
        end

        % 5. group by point index
        for i = 1:k
            set(:, :, i) = {XLabel(XLabel(:, dim + 1) == i, 1:dim)};
        end

        % 6. renew points
        for i = 1:k
            avg = mean(set{i}, 1);
            points(i, :) = avg(1:dim);
```



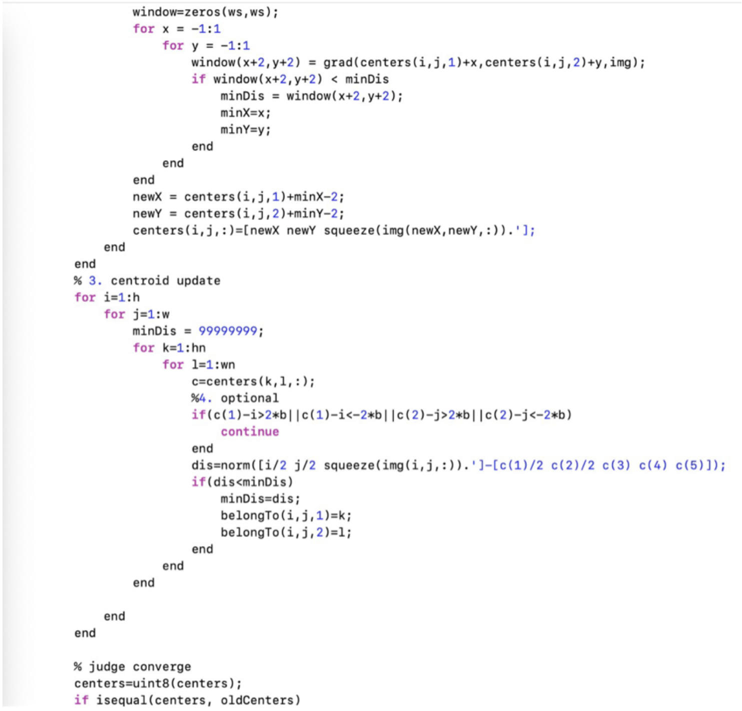
```
            points(i, :) = avg(1:dim);
        end
    end

    % 7. return centers and clusters
    clusters = set;
    centers = points;
end
```



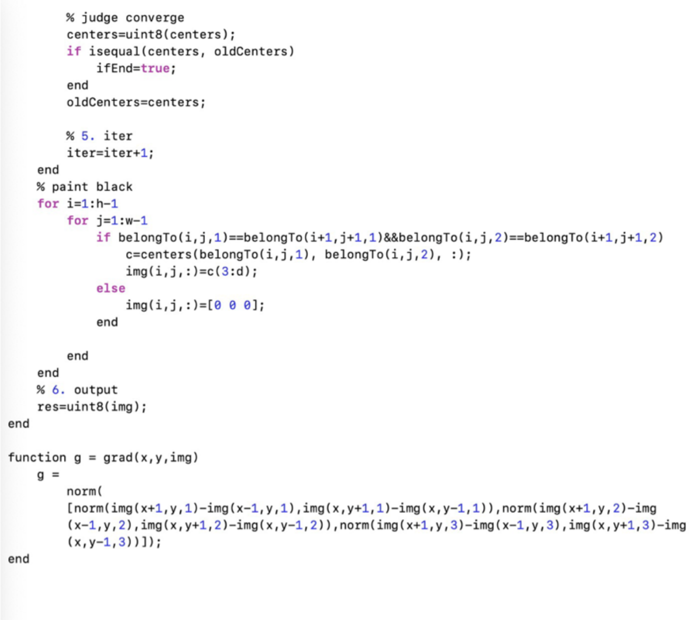
```
% 1. load png to rgb
RGB=imread('white-tower.png');
% 2. flatten
T=reshape(RGB, size(RGB,1)*size(RGB,2),size(RGB,3));
% 3. kmeans
[clusters,centers]=kmeans_code(T,10,10);
% 4. print
uint8(centers)
```



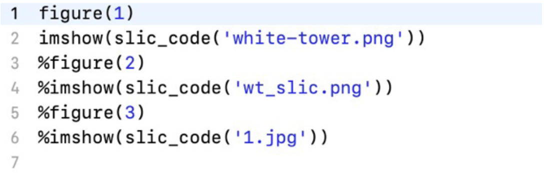
```
function res = slic_code(path)
    % 1. init
    b=50;
    img=imread(path);
    img=single(img);
    h=size(img,1);
    w=size(img,2);
    hn=ceil(h/b);
    wn=ceil(w/b);
    d=5;
    centers=zeros(hn, wn ,d);
    oldCenters = uint8(centers);
    for i=1:hn
        x=b/2+b*(i-1);
        x=min(x,h);
        for j=1:wn
            y=b/2+b*(j-1);
            y=min(y,w);
            centers(i,j,:)=[x y squeeze(img(x,y,:)).'];
        end
    end
    belongTo=ones(h,w,2);
    for i=1:h
        for j=1:w
            belongTo(i,j,1)=floor(i/b)+1;
            belongTo(i,j,2)=floor(j/b)+1;
        end
    end
    ifEnd=false;
    maxIters=3;
    iter=1;
    ws=3;

    while ~ifEnd && iter <= maxIters
        centers=single(centers);
        % 2. local shift
        for i=2:hn-1
            for j=2:wn-1
                minDis = 9999999;
                minX = -1;
                minY = -1;
                window=zeros(ws,ws);
```



```
                window=zeros(ws,ws);
                for x = -1:1
                    for y = -1:1
                        window(x+2,y+2) = grad(centers(i,j,1)+x,centers(i,j,2)+y,img);
                        if window(x+2,y+2) < minDis
                            minDis = window(x+2,y+2);
                            minX=x;
                            minY=y;
                        end
                    end
                end
                newX = centers(i,j,1)+minX-2;
                newY = centers(i,j,2)+minY-2;
                centers(i,j,:)=[newX newY squeeze(img(newX,newY,:)).'];
            end
        end
        % 3. centroid update
        for i=1:h
            for j=1:w
                minDis = 99999999;
                for k=1:hn
                    for l=1:wn
                        c=centers(k,l,:);
                        %4. optional
                        if(c(1)-i>2*b||c(1)-i<-2*b||c(2)-j>2*b||c(2)-j<-2*b)
                            continue
                        end
                        dis=norm([i/2 j/2 squeeze(img(i,j,:)).']-[c(1)/2 c(2)/2 c(3) c(4) c(5)]);
                        if(dis<minDis)
                            minDis=dis;
                            belongTo(i,j,1)=k;
                            belongTo(i,j,2)=l;
                        end
                    end
                end

            end
        end

        % judge converge
        centers=uint8(centers);
        if isequal(centers, oldCenters)
```



```
        % judge converge
        centers=uint8(centers);
        if isequal(centers, oldCenters)
            ifEnd=true;
        end
        oldCenters=centers;

        % 5. iter
        iter=iter+1;
    end
    % paint black
    for i=1:h-1
        for j=1:w-1
            if belongTo(i,j,1)==belongTo(i+1,j+1,1)&&belongTo(i,j,2)==belongTo(i+1,j+1,2)
                c=centers(belongTo(i,j,1), belongTo(i,j,2), :);
                img(i,j,:)=c(3:d);
            else
                img(i,j,:)=[0 0 0];
            end

        end
    end
    % 6. output
    res=uint8(img);
end

function g = grad(x,y,img)
    g =
        norm(
        [norm(img(x+1,y,1)-img(x-1,y,1),img(x,y+1,1)-img(x,y-1,1)),norm(img(x+1,y,2)-img
        (x-1,y,2),img(x,y+1,2)-img(x,y-1,2)),norm(img(x+1,y,3)-img(x-1,y,3),img(x,y+1,3)-img
        (x,y-1,3))]);
end
```



```
figure(1)
imshow(slic_code('white-tower.png'))
%figure(2)
%imshow(slic_code('wt_slic.png'))
%figure(3)
%imshow(slic_code('1.jpg'))
```