INF 5300 – Lab exercises on segmentation

Anne Solberg (anne@ifi.uio.no)

- Snakes
- Markov random field classification using the ICM algorithm

Matlab exercise - Snakes

- The Matlab scripts and images used can be found on the course web page.
- http://heim.ifi.uio.no/~inf5300/2008/defcont.zip
- They contain mostly a direct implementation of the Kass algorithm from last lecture.
- Start by clearing the workspace and closing all windows. Load a test image.
- Uncomment the test image you want to use.

```
% Start at scratch
% Define external force field
% Read test image and convert to double

 F=imread('circ.tif','tif');
 F=double(F);

% F=imread('square.tif','tif');
% F=double(F);

%F=imread('u.tif','tif');
%F=double(F);
```


Matlab demonstration

```
% Display it

figure
imshow(F,[]);
title('Input image');

% We want to use the negative magnitude of the gradient of this image as external
% force field so we need sobel masks

s_vert=-fspecial('sobel');
s_horz=s_vert';

% Calculate the gradient information.

F_vert=imfilter(F,s_vert,'replicate');
F_horz=imfilter(F,s_horz,'replicate');

% Lets look at these two images

figure
imshow(F_vert,[])
title('Vertical gradients')
figure
imshow(F_horz,[])
title('Horizontal gradients')
```


Matlab demonstration

```
% Now lets calculate the negative magnitude of the gradient.
% This will be the external force field. In order to allow for
% different input images we normalize the gradient image
% to 1

P=sqrt(F_horz.*F_horz+F_vert.*F_vert);
P=P/(max(max(P)));
P=-P;
figure
imshow(P,[])
title('Inverse magnitude of gradient vector')

% Last thing, we need the two spatial derivatives
% of our external force field. Calculate these and
% have a look at them.

P_vert=imfilter(P,s_vert,'replicate');
P_horz=imfilter(P,s_horz,'replicate');

figure
imshow(P_horz,[])
title('X derivative of force field')
figure
imshow(P_vert,[])
title('Y derivative of force field')
```

Take a look at the gradient vectors

```
% Lets take a look at these gradient vectors

[X,Y]=meshgrid([1 4:4:256],[1 4:4:256]);
figure
contour(flipud(P))
hold on
quiver(X,flipud(Y),getmatind(-P_horz,X,Y),getmatind(P_vert,X,Y))
axis image
title('Gradient vectors and contour lines')
```

- Note that close to the border (where the gradient is non-zero), the gradient vector points in the direction of the maximum gradient.
- Acting like a force for the snake this will pull the snake in the direction of the maximum gradient

Matlab demonstration

```
% Now lets define our snake, to begin with lets decide
% on some small number of control points (you can change
% this to your liking, the rest of the program will adapt
% gracefully)

N=20;

% Now we need to give the snake a shape.  Lets make it a circle
% and then "nudge" it a little.

x0=50*cos(0:(2*pi/(N)):(2*pi-(2*pi/(N))))+128
y0=-50*sin(0:(2*pi/(N)):(2*pi-(2*pi/(N))))+128

x0(2)=x0(2)+30;
y0(2)=y0(2)-20;
```


Matlab demonstration

```
% Define the weights given to the two terms in the inner energy
% functional.  The values are NOT arbitrary.

w1=0.000001;
w2=0.01;

% Define constants for the stiffness matrix, do not edit this.

alpha=w2;
beta=-w1-4*w2;
gamma=-2*w1+6*w2;

% Define the step size

lambda = 0.2 % Stiff system
% lambda=0.1; % Unstiff system
```


Matlab demonstration

```
% Define Stiffness matrix. The code below is just a smart way of doing
% this independently of the number of nodes.

% A =[gamma   beta    alpha   0       0       0       alpha   beta;
%     beta    gamma   beta    alpha   0       0       0       alpha;
%     alpha   beta    gamma   beta    alpha   0       0       0;
%     0       alpha   beta    gamma   beta    alpha   0       0;
%     0       0       alpha   beta    gamma   beta    alpha   0;
%     0       0       0       alpha   beta    gamma   beta    alpha;
%     alpha   0       0       0       alpha   beta    gamma   beta;
%     beta    alpha   0       0       0       alpha   beta    gamma];

A=diag(beta,-N+1)+...
  diag(alpha*ones(1,2),-N+2)+...
  diag(alpha*ones(1,N-2),-2)+...
  diag(beta*ones(1,N-1),-1)+...
  diag(gamma*ones(1,N),0)+...
  diag(beta*ones(1,N-1),+1)+...
  diag(alpha*ones(1,N-2),2)+...
  diag(alpha*ones(1,2),N-2)+...
  diag(beta,N-1)
```

Matlab demonstration

```
% Initialise x and y

x=x0';
y=y0';

% The maximum number of iterations

maxiter=500;

% Weight given to external field, set to 0 or 1.

omega=1;  %How much should the gradient information be weighed?

% Display results on top of the input image

figure
imshow(P,[])
title('Snake position')
hold on
```


Matlab demonstration

```
iter=0;
while(iter<maxiter)
   c=rand(1,3); % Randomly color the snake
   %plot(x,y,'*','color',c) % Plot the snake control points
   lplot(x,y,c) % Interconnect the nodes
   iter=iter+1 % Display the iteration number
   x=(inv(A+lambda*eye(N)))*(lambda*x-
            omega*getmatind(P_horz,round(x)+1,round(y)+1));
   y=(inv(A+lambda*eye(N)))*(lambda*y-
            omega*getmatind(P_vert,round(x)+1,round(y)+1));
   dummy=input(['Press return to continue']);
end
```

Remember the equation

getmatind found in defcont.zip

Matlab demonstration

- First, try to run the snake with only internal fources.
- Let's first focus on the "tension force".
- Initialize to a circle.
- Set the weight for the external force (omega) to zero.
- What happens?

A more difficult example

- Try the snake on this image ~inf5300/www_docs/data/seismic_timeslice.mat

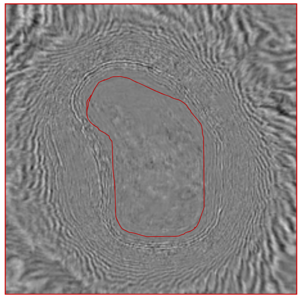
The boundary we are looking for is the texture boundary between the high-frequency texture, and the homogeneous inside.

First – find a feature that has high gradient close to this boundary and low gradient elsewhere.

Use this feature image as input to the snake.

Initialize the snake using a circle both inside and outside the true contour.

Can you make it work?



Markov Random field classification

- Implement the ICM-algorithm from foil 29 from the lecture notes from 5.2.
- Test the effect of contextual classification on the image mar1710_small.mat with training mask mask1710_train1.mat and test mask mask1710_test1.mat.
- The images are available at ~inf5300/www_docs/data
- Experiment with beta and see how the result changes

Mean shift clustering

- Implement a simple mean shift clustering algorithm.
- Try it on the images parrot.png and reine.png
- Expertiment with kernel sizes to get a reasonable segmentation result.
- If time: compare the results to using K-means clustering.
- If you are short of time, download a matlab implementation from Matlab Central.