Contour-based Segmentation

Imaging Lab 9 - May, 2017

Kamyar Nazeri
Student ID: 100633486

Chan-Vese (CV) segmentation

Image segmentation is the process of partitioning a digital image into multiple segments. Here we are using Deformable Models to do segmentation which is evolving a contour Γ that snaps around objects. Chan-Vese Active Contours Model is an example of deformable model and its 2-phase model is:

$$\boldsymbol{\min_{\Gamma}\ E_{CV}[\Gamma|f] = L(\Gamma) + \lambda_{in} \int\limits_{inside\ \Gamma} (f - c_{in})^2 d\vec{x} + \lambda_{out} \int\limits_{inside\ \Gamma} (f - c_{out})^2 d\vec{x}}$$

Where $L(\Gamma)$ is the length of the curve and c is the average gray value of f in each region. By using The Heaviside Function, these equations become:

$$L(\Gamma) = \int\limits_{\Omega} \|\nabla H(\phi)\| d\vec{x} = \int\limits_{\Omega} \delta(\phi)\|\nabla\phi\| d\vec{x}$$

$$c_{in} = \frac{\int\limits_{\Omega} H(\phi) f d\vec{x}}{\int\limits_{\Omega} H(\phi) d\vec{x}} \qquad c_{out} = \frac{\int\limits_{\Omega} (1 - H(\phi)) f d\vec{x}}{\int\limits_{\Omega} (1 - H(\phi)) d\vec{x}}$$

By minimizing the steepest descent to evolve the PDE:

$$\frac{\partial\phi}{\partial t} = \delta(\phi)\left[\nabla \cdot (\frac{\nabla\phi}{|\nabla\phi|}) - \lambda_{in}(f - c_{in})^2 + \lambda_{out}(f - c_{out})^2\right]$$

Where $\delta(\phi)$ is the Dirac delta function, and we use a smooth approximation for it:

$$\delta_\epsilon(t) = \frac{\epsilon}{\pi(\epsilon^2 + t^2)}$$

We initialize segmenting a test image using 1) a large circle in the center of the image and 2) a small circle around a user specified starting point.
To draw a large circle in the center of the image we use the following Matlab code:

```
[x,y] = meshgrid(1:n, 1:m);
D = [(x - n/2).^2 + (y - m/2).^2 <= 100^2];
```

To allow the user to pick a starting point on the image, we use Matlab's *ginput* function, locate point clicked by the user and draw a small circle around it:

```
imshow(uint8(f));
P = ginput(1);
[x,y] = meshgrid(1:n, 1:m);
D = [(x - P(1)).^2 + (y - P(2)).^2 <= 3^2];
```

To prevent a sudden jump in the gradient of the level set, we use the signed distance smooth level set function using Matlab's *bwdist* command:

```
u = bwdist(1-D) - bwdist(D);
u = 0.1 * u;
```

Note that we scale the level set function down, to make our segmentation algorithm faster.

Coding Chan-Vese Segmentation

Listing 1 shows the Chan-Vese Segmentation applied on a grayscale image in Matlab, note that the initial level set function is specified by the user:

```
function [u] = CVINT(f, init , lambda)
    % Chan-Vese segmentation algorithm using level sets

    %Parameters
    dt = 0.2;                % time step
    T = 100;                 % stopping time
    a = 0.1;                 % fudge factor
    e = 0.1;                 % epsilon (Dirac function)
    [m,n] = size(f);         % image size
    f = double(f);           % convert to double
    u = init;

    for t = 0:dt:T
        % u_x = (u(x+1,y) - u(x-1,y)) / 2
        u_x = (u(:,[2:n,n]) - u(:,[1,1:n-1])) / 2;

        % u_y = (u(x,y+1) - u(x,y+1)) / 2
        u_y = (u([2:m,m],:) - u([1,1:m-1],:)) / 2;

        % u_xx = u(x+1,y) - 2u(x,y) + u(x-1,y)
        u_xx = u(:,[2:n,n]) - 2 * u + u(:,[1,1:n-1]);

        % u_yy = u(x,y+1) - 2u(x,y) + u(x,y-1)
        u_yy = u([2:m,m],:) - 2 * u + u([1,1:m-1],:);

        % u_xy = (u(x+1,y+1)+u(x-1,y-1)-u(x-1,y+1)-u(x+1,y-1))/4
        u_xy = (u([2:m, m],[2:n, n]) + u([1,1:m-1], [1,1:n-1]) -
           u([2:m, m],[1, 1:n-1]) - u([1, 1:m-1],[2:n, n])) / 4;

        num = (u_xx.*u_y.^2)-2*(u_x.*u_z.*u_xy)+(u_yy.*u_x.^2);
        denom = (u_x.^2 + u_y.^2).^(3/2) + a;

        % The Dirac delta function, we use a smooth approximation
        drc = e ./(pi * (e^2 + u.^2));

        % the average gray value of f inside the region
        cin =  sum(sum(f .* [ u > 0 ])) / sum(sum([u > 0]));

        % the average gray value of f outside the region
        cout =  sum(sum(f .* [ u < 0 ])) / sum(sum([u <0 ]));

        pde = drc.*(num./denom-lambda*((f-cin).^2+(f-cout).^2));
        u = u + dt * pde;
    end
end
```

Listing 1: Chan-Vese Segmentation function for grayscale images in Matlab

Figure 1 shows segmenting a test image using a large circle in the center of the image:

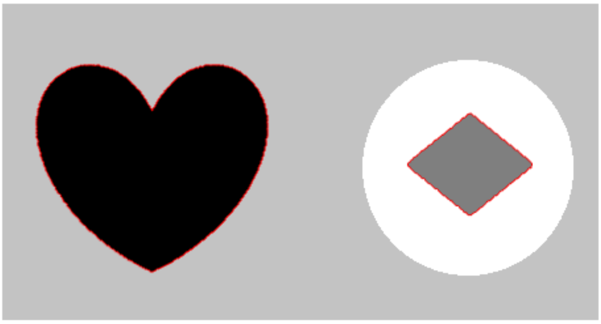

Figure 1: Chan-Vese Segmentation with a large circle in the center as the init level set function.

Figure 2 shows segmenting specific shapes in a test image by drawing a small circle around a user specified starting point as the init level set function:

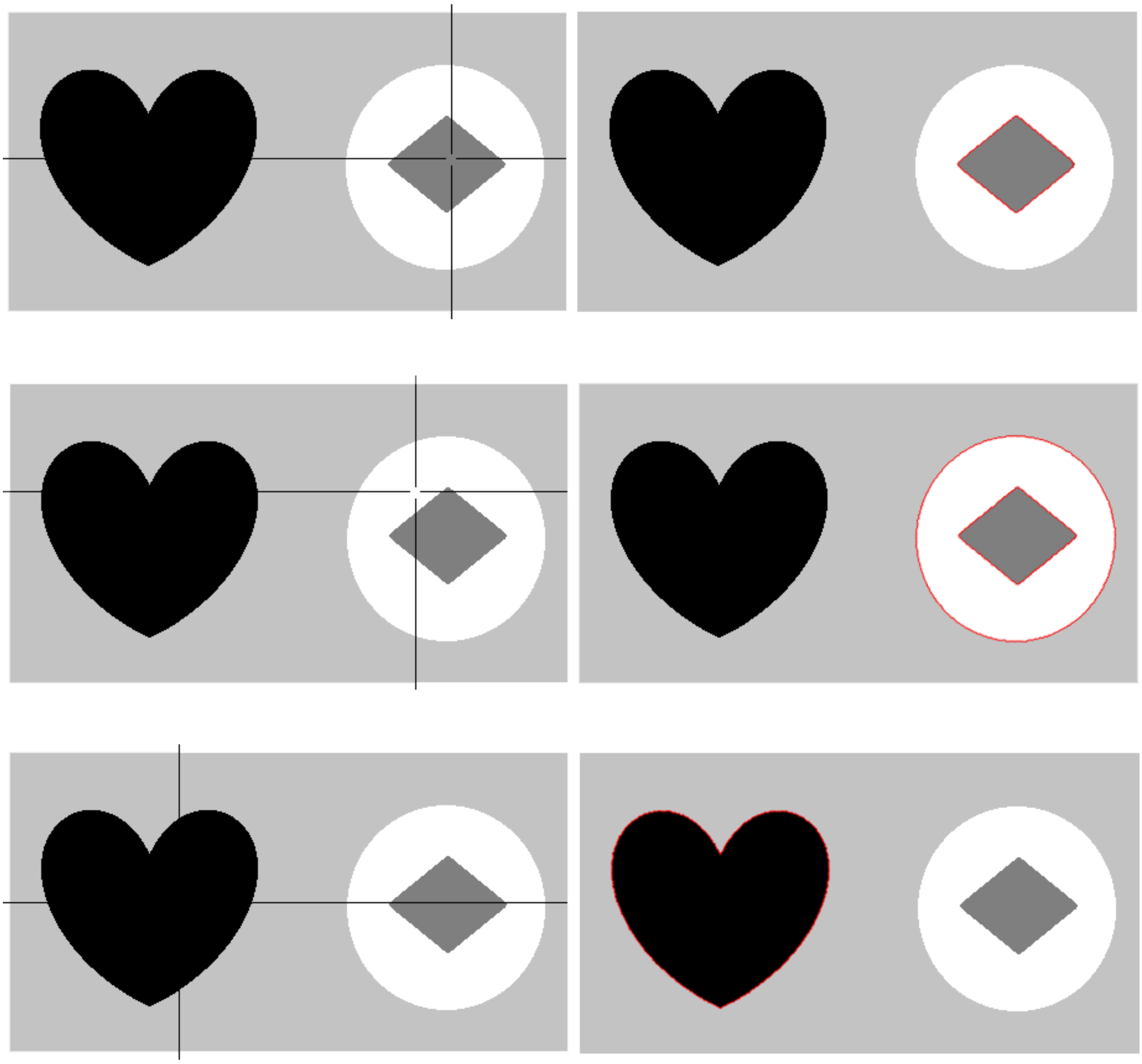

Figure 2: Chan-Vese Segmentation with a small circle around a user specified starting point as the init level set function.