Active Contour

Theory

In class, we already learnt that we can use a flexible line s(p)=(x(p),y(p)) moving around to minimize the error function.

The error function for active contour include two parts, internal energy and external energy

$$E_{int} = \alpha E_{cont} + \beta E_{curv} = \frac{1}{2}[a|s'|^2 + \beta|s''|^2]$$

To minimize the energy, using Euler-Lagrange equation, the energy function can be rewrite as

$$as'' - \beta s'''' - \nabla E_{ext} = 0$$

We are going to use gradient decent method to find out the minimal energy, so we can convert s into a function of time, the equation is converted to

$$\partial s/\partial t = as'' - \beta s'''' - \nabla E_{ext}$$

Since s has x and y component, we can write them separately.

$$x''[i] = x[i-1] - 2x[i] + x[i+1]$$
$$x''''[i] = x[i-2] - 4x[i-1] = 6x[i] - 4x[i+1] + x[i+2]$$

The internal part is a matrix

$$A = \begin{pmatrix} -2a - 6\beta & a + 4\beta & -\beta & 0 & 0 \cdots 0 & -\beta & a + 4\beta \\ \vdots & & & \ddots & & & \vdots \\ a + 4\beta & -\beta & & \cdots & & & -2a - 6\beta \end{pmatrix}$$

Every row, there are only 5 columns not zero, and their values are shifted one column per row.

$$\frac{\partial x}{\partial t} = Ax - \nabla_x E_{ext}$$

Discretize the time parameter

$$\frac{\partial x}{\partial t} = (x_t - x_{t-1})/r$$

In the y direction, it's similar.

The problem is simplified

$$X_t = (1 - rA)^{-1}[X_{t-1} + rf_x(X_{t-1}, Y_{t-1})]$$
$$Y_t = (1 - rA)^{-1}[Y_{t-1} + rf_y(X_{t-1}, Y_{t-1})]$$

fx,fy are x, y component of derivative of external energy.

Ref: https://www.crisluengo.net/archives/217

Code description
1) Image could be imported either from the camera or read from file.
2) The original image was first displayed in the window, press esc to close window, the image would be converted to gray scale image. Depends on images, you can choose to add noise to image. Image will be smoothed before further processed.
3) after preprocessing, the gray image will be displayed in the window, click mouse to draw a initial contour on the image. Due to the limitation of my algorithm, the contour should be not far away from the actual contour. Press ESC to close the window.
4) gray image without drawing contour will once again displayed in the window, press keyboard for different options. The text file is saved in snake.txt.
s-show active contour of image
w-save current image
c-show image with mouse drawing initial contour
o-show original image after converting to gray scale, smoothing and for some image adding noise
h-show help menu
esc-close window
5) important function description
captureVid() to capture images from video.
showImg(img), grayImg(img), noise(img), smooth(img) for images preprocessing
draw_circle, mouse callback function
gradient(), return derivatives of external engery
createA() creates the matrix A
xcheckBounds, ycheckBounds, check if the x, y coordinates still within the image
snake, perform the active contour algorithm
parameters
a= 0.006
b = 0.001
r = 100
iterations = 100

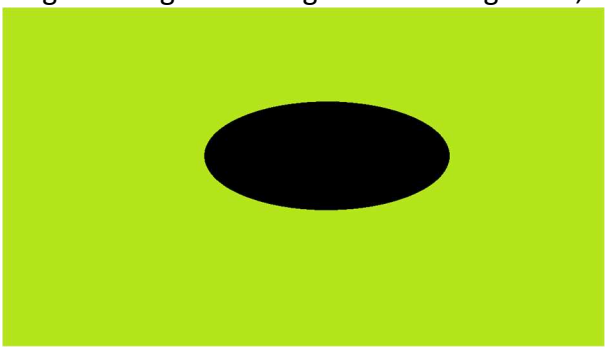
```
def snake(x, y, N, a, b, gx, gy, gamma, n_iters,img):

    A = createA(a,b,N)
    B = np.linalg.inv(np.identity(N) - gamma*A)
    snakes=[]

    for i in range(n_iters):
        if(i%5==0):
            plt.plot(x,y)

        fx=np.zeros(x.shape[0])
        fy=np.zeros(x.shape[0])
        for j in range(0, x.shape[0]):
            fx[j]=gx[y[j]][x[j]]
```

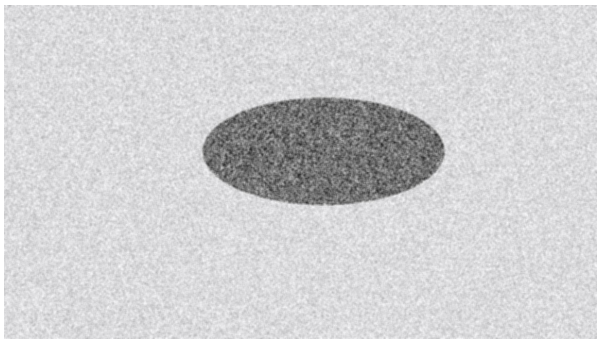
```
            fy[j]=gy[y[j]][x[j]]
        px = np.matmul(B, x + gamma*fx)
        px=xcheckBounds(px,img)
        py = np.matmul(B, y + gamma*fy)
        py=ycheckBounds(py,img)
        snakes.append( (py.copy(),py.copy()) )
        x, y = px.copy(), py.copy()
    return x,y
```

Result

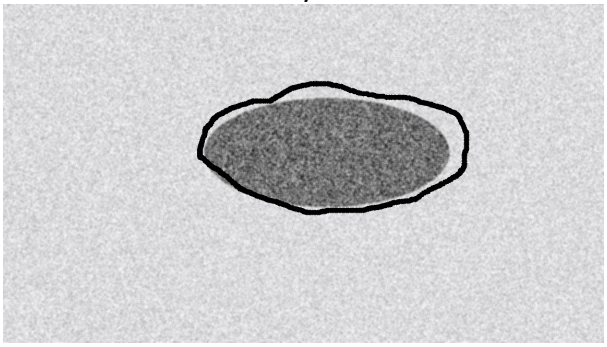
Start with a simple case, I draw a eclipse filled in paint. Convert to gray scale, to make the case more challenge, random noise was added to the picture.

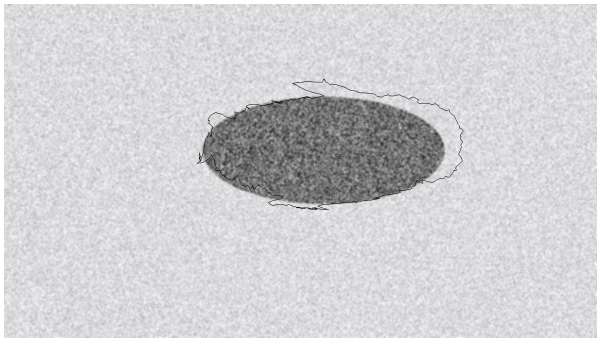
Original image and image after adding noise, smoothing, converted to grayscale



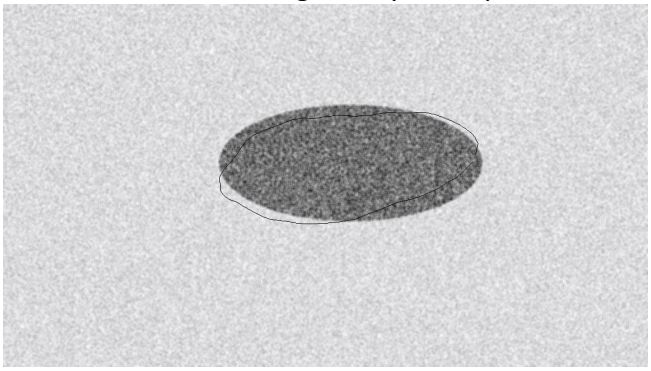


Initial contour drawn by mouse and final results.

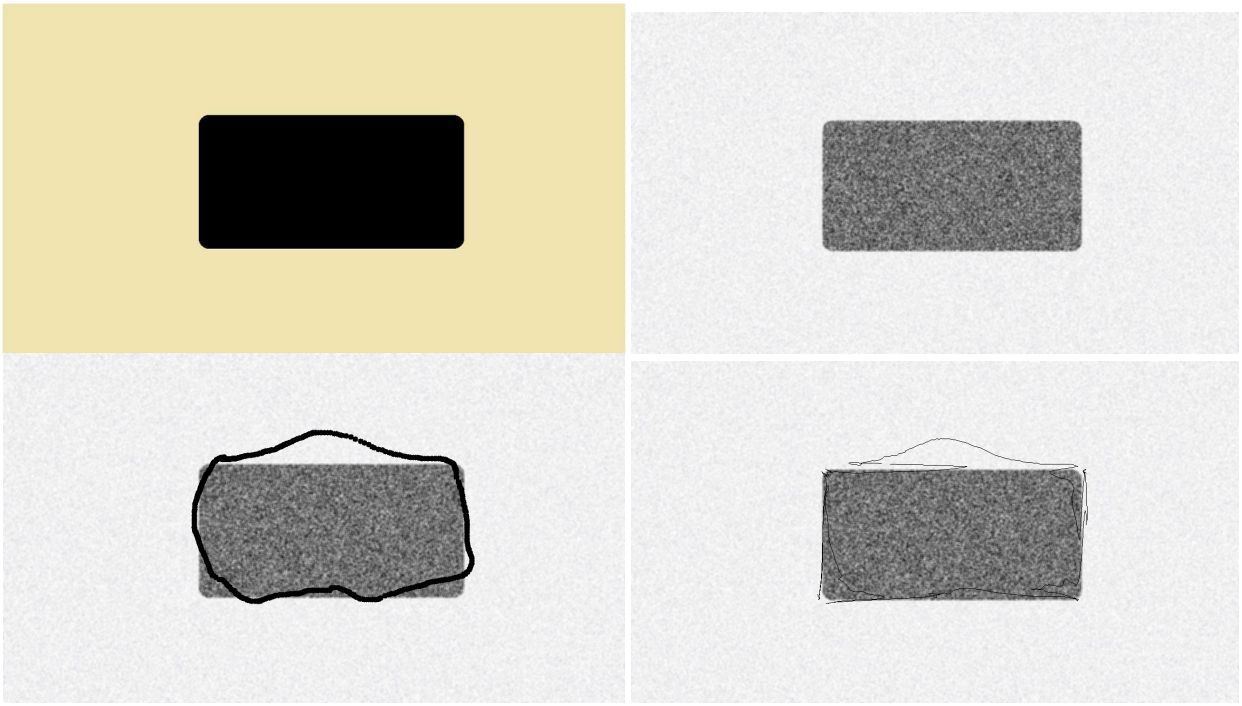


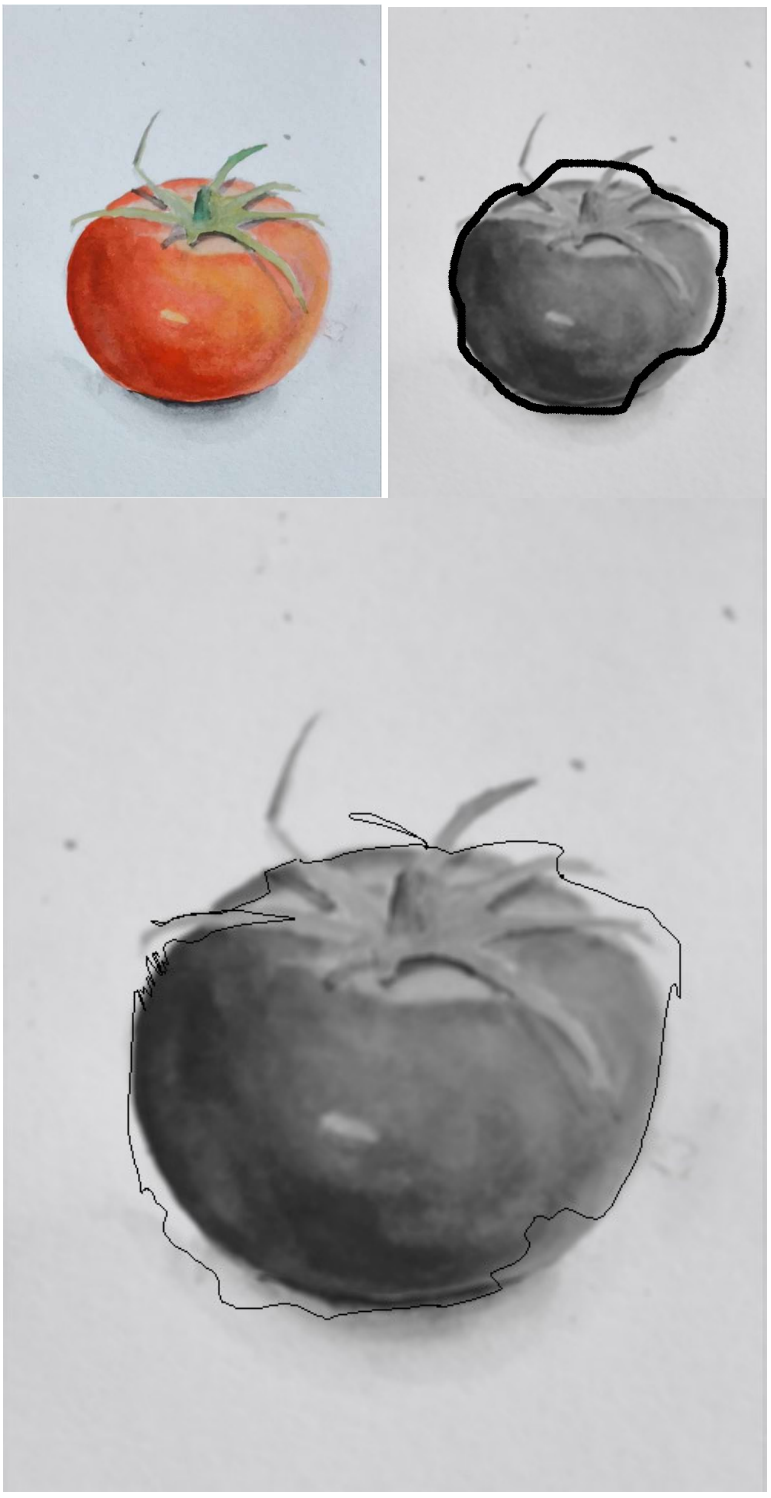


If the initial contour is given by a ecllips close to the object, the result will be better



More testing





Analysis
Based on my tests, my program can find the contour of the objects. We can find part of human's head, tomato leaves.
However, there are some limition

1) The initial contour should be close to the object. Since the background of my images are single color, the algorithm can't find the direction to move to reduce the error function
2) My parameters are constant and can't be changed in the middle of the program, for example, I can't change beta to allow segmentation.