END TO END TRAINABLE ACTIVE CONTOURS VIA DIFFERNTIABLE RENDERING

INTRODUCTION-:

- Curve evolution and level sets in image segmentation were helpful in quicker annotation and to predict boundaries. But these had problem in fitting straight edges as they are initialized as curves, polygons would better fit the straight edges. Also it required some user intervention like placing extreme points by annotators to initialize the curve.
- To address the above issues, here we use a neural mesh renderer which is end-to-end trainable and it predicts a polygonal boundary which can fit straight edges.

METHOD-:

- The trained model has encoder (E), decoder (D), differential renderer(R) and triangulation procedure (L)
    - The triangulation procedure followed is Delaunay triangulation, it follows the procedure that the triangles formed by distinct points such that no point lies inside the circumcircle of the triangles.
    - The encoder and decoder model have convolutional, batch normalization and dropout layers with drop out rate of 0.2.
- S is the set of all training and I is an individual image which belongs to R^(channels x height x width).

Steps involved-:

- The image I is passed through a stacked Encoder and decoder model which spits out a dual channel displacement field J, which tells about the location of the

object. It gives the displacement values at each point which helps to update the vertices of the polygon.

- Initial contour is generated by initializing a circle with centre as the centre of the object and diameter of 16 pixels. So it requires no user intervention as it is able to generate initial contour polygon.
    - Number of vertices such that it does not drive down the accuracy, because there is a saturation point to chose number of vertices. Each vertex has a x and y co-ordinate. By following triangulation procedure faces (triangles) are generated. This process is followed in each iteration
    - Initial polygon generated —> points are stored with their co-ordinates —> faces are generated by triangulation procedure —> pixels inside the pixels are labelled as 1 and outside are zero.
- After the initial contour is generated, the vertices are modified by using the displacement field.
-

$$p_j^t = p_j^{t-1} + J(p_j^{t-1})$$

Taken from the paper End to End trainable active contours by differentiable rendering

- here p_j means the vertices and t is the iteration. So each point is updated in each iteration following the displacement field. The final polygon is generated after T iterations.
- Then combining these points a n-vertices polygon is generated and following triangulation procedure faces are generated. So each polygon has P^t = [p_1,p_2,....p_k] at t iteration and k is the number of vertices
- The points which are outside the polygon are truncated
- Then the final segmentation mask (M^t) is generated by rendering the final polygon and generating faces using triangulation procedure. The pixels inside the pixels are labelled as 1 and outside are zero.
- For 3D rendering the segmentation mask is orthogonally projecting the points in z axis by 1 unit

LOSSES INVOLVED-:

- The segmentation mask and the ground truth contours are compared by taking MSE loss of the distance between them. It is called as segmentation loss (L_seg)
- Here M^t is the predicted segmentation mask and M is the true ground truth label

$$\mathcal{L}_{\text{SEG}} = \sum_{t=1}^{T} \|\bar{M}^t - M\|_2$$

Taken from the paper End to End trainable active contours by differentiable rendering

- Ballooning term (L_B), it tries to maximize the polygon area, so if the initial contour is initialized inside the object, by maximizing the area of polygon it can balloon out to fit the boundaries. It is normalized by dividing by the height and width of the segmentation mask.

$$\mathcal{L}_{\mathcal{B}} = \frac{1}{h \times w} \sum_{x} (1 - \bar{M}^t(x))$$

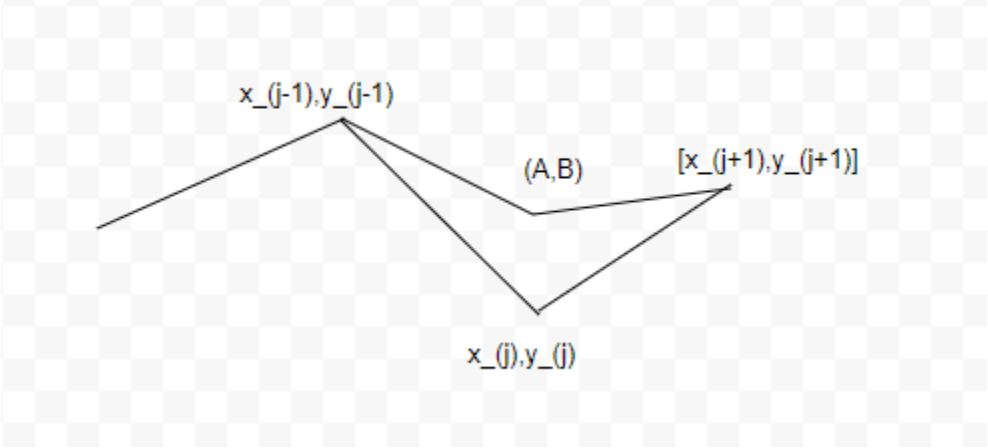
Taken from the paper End to End trainable active contours by differentiable rendering

- h is the height and w is the width of the segmentation mask
- Curvature term (L_k), it tries to minimize the curvature of the the polygon

$$\mathcal{L}_{\mathcal{K}} = \frac{1}{k} \sum_{j} ||p_{j-1}^t - 2p_j^t + p_{j+1}^t||_2$$

Taken from the paper End to End trainable active contours by differentiable rendering

- Intuition behind the above formulation,



- Here the point (A,B) is given by
  - A= [x(j+1) + x(j-1)]/2 and B = [y(j+1) + y(j-1)]/2, so the curvature is decreased by using this formulation so we need to minimize the loss.

- The combined loss function is given by L = L_seg + /lambda_1*L__B + /lambda2* L_k and /lambda_1 and /lambda_2 are applied in all the iteration.

Datasets-:

It is applied in various datasets like Vaihingen, Bing Huts, Toronto city to show its fit in straight edges like buildings. It is also applied in various medical images like cardiac MR, IN-Breast. Also Cityscrapes dataset is used.

- We have optimal resolution of 64x64, less than that will give unclear images and segmentation and 128x128 requires more epochs to train. Also training for more than 3 iterations gives diminishing returns