Learned Snakes for 3D Image Segmentation

Lihong Guo[a], Yueyun Liu[a], Yu Wang[b], Yuping Duan[a,*] and Xue-Cheng Tai[c]

[a]*Center for Applied Mathematics, Tianjin University, China, Tianjin, 300072, China,*
E-mail: {guolihong,yueyunliu}@tju.edu.cn

[b]*Alibaba Group, Hangzhou, China,*
E-mail: tonggou.wang@alibaba-inc.com

[a,*]*Corresponding author. Center for Applied Mathematics, Tianjin University, 300072, China,*
E-mail: yuping.duan@tju.edu.cn

[c] *Department of mathematics, Hong Kong Baptist University, Kowloon Tong, Hong Kong,*
E-mail: XUECHENGTAI@HKBU.EDU.HK

Abstract

Snakes or active contour models are classical methods for boundary detection and segmentation, which deform an initial contour (for 2D image) or a surface (for 3D image) towards the boundary of the desired object. Such snakes models are ideal choices for handling medical image segmentation problems since they are very efficient and require fewer memories by solely tracking the explicit curves or surfaces. However, traditional snakes models solved by the level set method suffer from numerical instabilities and are usually difficult to deal with topological changes. In this paper, we propose a learned snakes model for 3D medical image segmentation, where both the initial and final surfaces are estimated using deep neural networks in end-to-end regimes. The merit of our learned snakes model is that we can realize 3D segmentation by finding a 2D surface based on 2D convolutional neural networks rather than using 3D networks or cutting the volume into 2D slices. Experiments on the Medical Segmentation Decathlon spleen dataset against both 2D- and 3D-based networks demonstrate our model achieving the state-of-the-art accuracy and efficiency, which not only enjoys a 1% higher DSC but also saves more than 90% computational time compared to the well-established elastic boundary projection model [1].

Keywords: 3D segmentation, snakes, active contour, surface evolution, convolutional neural network

1. Introduction

Accurate and automatic organ segmentation is an important prerequisite for computer-assisted diagnosis (CAD), computer-aided surgery (CAS), and radiation therapy (RT), which has been intensively studied in the last several decades [2, 3]. Segmentation aims to extract the organ/object from the given image according to features such as intensities, edges, shape priors,

etc. Various methods have been studied for image segmentation, which either find a partition of the image [4, 5, 6] or detect contours of objects [7, 8, 9].

Mumford and Shah [4] proposed a pioneer segmentation model by finding a closed image of the initial one compounded of several regions with nearly constant intensities. Let $F : \Omega \to \mathbb{R}$ be a given image with Ω being a bounded open subset of $\mathbb{R}^n$. The Mumford-Shah model searches for a pair (U,Γ) by minimizing the following energy functional

$$J_{\mathrm{MS}}(U,\Gamma) = \frac{\lambda}{2}\int_{\Omega}(F-U)^2dx + \int_{\Omega\setminus\Gamma}|\nabla U|^2dx + \alpha\int_{\Gamma}d\sigma, \tag{1}$$

where α, λ are positive parameters, Γ is the set of discontinuities and $\int_{\Gamma}d\sigma$ denotes the length of Γ. Since the Mumford-Shah functional is non-convex, finding the minimizers may trap into local minima. Besides, the Mumford-Shah model (1) needs to be calculated on the entire image domain, the computational costs of which increase dramatically for 3D image segmentation tasks. In Kass, Witkin, and Terzopoulos [7], boundary detection consists of matching a deformable model to an image by considering an energy-minimizing spline guided by external constraint forces and influenced by image forces, which is called *snakes* or *active contours* due to the way of contour evolution. Let $\mathsf{C}(q) : [0,1] \to \mathbb{R}^2$ be a parameterized planar curve. The classical snakes model associated the curve C is to minimize the following energy

$$J(\mathsf{C}) = \underbrace{\int_0^1 |\mathsf{C}'(q)|^2dq + \alpha\int_0^1 |\mathsf{C}''(q)|dq}_{\text{internal energy}} \underbrace{-\lambda\int_0^1 |\nabla F(\mathsf{C}(q))|dq}_{\text{external energy}}. \tag{2}$$

The internal energy guarantees the smoothness of the curve while the external energy attracts the curve toward the edge of the objects. However, the snakes (2) has significant drawbacks. Firstly, it is not intrinsic, which means $J(\mathsf{C})$ depends on the parametrization of C. Even with the same initial curve, different solutions may be obtained by changing the parametrization. Secondly, the energy model is not capable of handling changes of topology, where the final contour has to be as the one of C_0. Lastly, the minimization problem (2) is also numerically problematic such that only a local minimum can be reached. Thus, the initial contour has to been chosen closed enough to the boundary of the object to be detected.

Caselles, Kimmel and Sapiro [8] introduced an intrinsic model, called the *geodesic active contour*. The evolving contours naturally split and merge, which allows detecting several objects and both interior and exterior boundaries simultaneously. Specifically, the energy functional is formulated as follows

$$J_{\mathrm{GAC}}(\mathsf{C}) = \int_0^1 g(|\nabla F(\mathsf{C}(q))|)|\mathsf{C}'(q)|dq, \tag{3}$$

where $g : [0,+\infty] \to \mathbb{R}^+$ is an edge detection function. The steepest descent method was implemented to solve the minimization problem (3) to deform the initial curve $\mathsf{C}(0) = \mathsf{C}_0$ using the following curve evolution equation

$$\frac{\partial \mathsf{C}(t)}{\partial t} = \Big(g\kappa - (\nabla g \cdot \boldsymbol{n})\Big)\boldsymbol{n}, \tag{4}$$

where κ is the Euclidean curvature and $\boldsymbol{n}$ is the unit normal vector. Note that if $g \equiv 1$, the flow (4) reduces to

$$\frac{\partial \mathsf{C}(t)}{\partial t} = \kappa \boldsymbol{n}, \tag{5}$$

which is the mean curvature motion. By regarding the contour as the zero level-set of a 3D function, the geodesic curve computation is reduced to a geometric flow, the steady stage of which gives the detected object. The level set method is the most popular method used to solve the model (4) and (5). However, such kind of methods suffers from high computation burden and numerical instability, requiring re-initialization or additional regularizations.

Convolutional neural networks (CNNs) based segmentation techniques have been extensively studied and gain great success in the last decade. Similar to traditional approaches, we can roughly divide the existing CNNs-based segmentation methods into two categories, i.e., pixel-wise [10, 11, 12, 13] and contour-based methods [14, 15]. Chen *et al.* [16] designed a deep learning framework for 3D image segmentation based on a combination of a fully convolutional network (FCN) and a recurrent neural network (RNN), which are responsible for exploiting the intra-slice and inter-slice contexts, respectively. Zhou *et al.* [17] developed a fixed-point model to use a predicted segmentation mask to shrink the input region, which applied the 2D FCN along three axes and fused the segmentation via majority voting. Yu *et al.* [18] presented a recurrent saliency transformation network in a coarse-to-fine framework for small organ segmentation, where three FCN models were trained separately to incorporate pseudo-3D information into segmentation. Razzak *et al.* [19] developed a two-pathway-group CNN to embed both local and global features without increasing the number of parameters. The aforementioned approaches cut the 3D volume into 2D slices and use a 2D network to deal with each slice, which may lose rich spatial information along the third axis. Thus, 3D pixel-wise networks have also been developed to deal with volumetric data, such as the 3d U-net [20], V-net [21], 3d deep supervised network [22], etc. However, dealing with volumetric data is not only computationally expensive requiring much larger memory consumption, but also less stable to be trained from scratch lacking a pre-trained model for medical purposes. To bridge the gap between 2D and 3D organ segmentation, Xia *et al.* [23] used a 3D volumetric fusion net to fuse the 2D segmentation results from different viewpoints, which still rely on 3D convolutions. Li *et al.* [24] proposed a hybrid densely connected U-net, which consists of a 2D dense U-net for efficiently extracting intra-slice features and a 3D counterpart for hierarchically aggregating volumetric contexts. Liu *et al.* [25] designed a 3D anisotropic hybrid network for 3D medical image segmentation, which can transfer convolutional features learned from 2D images to 3D anisotropic volumes.

On the other hand, edge information has also been deployed in CNN models for medical image segmentation. Chen *et al.* [26] proposed the deep contour-aware network for gland segmentation, where both probability maps of glands and the contours are learned in a multi-task framework. Duan *et al.* [27] used a single nested level set framework to segment cardiac MR images with pulmonary hypertension, which incorporated the image features learned from a deep neural network. Roy *et al.* [28] proposed a synergistic combination of deep learning and

Table 1: The comparison of average time consuming (min) and storage space consuming (Gigabyte) between EBP and LSM for processing one spleen data of the MSD dataset.

	Step I	Step II	Step III	Storage space
EBP	Shells evolution	Pivots processing	3D reconstruction	20
	51.41	4.62	4.27	
LSM	Surface initialization	Surface evolution	3D reconstruction	1.5
	0.50	1.10	3.21	

shape driven level sets for lung nodule segmentation, which achieved better accuracy than both deep network and level set methods. In addition to medical image segmentation, contour-based approaches have also been utilized for instance segmentation such as deep structured active contour [29], curve-GCN [30], deep snakes [31], etc. Moreover, novel loss functions have been studied based on classical active contour models, such as level set loss [32], Mumford-Shah loss [33] and active contour loss [34, 35, 36], which can better combine the geometrical information with region similarity.

At present, studies on 3D contour-based or surface evolution methods are still missing. Ni *et al.* [1] developed the elastic boundary projection (EBP) for 3D medical image segmentation, which places a number of pivot points in the 3D space and determines their corresponding distances to the object boundary along with a dense set of directions. The EBP model achieved good accuracy by bridging the gaps between 2D and 3D approaches such that it can process the 3D data as a whole rather than cutting it into slices. Besides, the EBP can deal with the scenarios of limited data annotations by simply increasing the number of pivots in the training process. However, it consumed an average of 20 Gigabyte disk storage space and 60 minutes to process one spleen data of the Medical Segmentation Decathlon (MSD) dataset (website: http://medicaldecathlon.com/) due to the large numbers of pivots required in training and testing, which greatly limits its practical applications.

In this paper, we propose a novel, end-to-end trainable deep neural network architecture, called the learned snakes model (LSM), to find the desired boundary of 3D objects. Followed the classical snakes, our model contains two parts: 1) generate the initial surface with the similar shape as the desired object; 2) deform the initial surface accurately to match with the object boundaries. More specifically, we use 2D networks to realize both surface initialization and evolution, after which a 3D reconstruction process is performed to recover the 3D volume. Through an efficacious surface initialization, our model requires much fewer pivot points in the surface evolution and 3D reconstruction. Thus, our LSM employs less storage space and performs much faster than EBP, where more than 90% of the disk space and computational time are saved; see Table 1. We evaluate our model on spleen segmentation tasks and demonstrate its promising performance by comparing with the well established 2D and 3D segmentation methods.

To sum up, the main contributions of this work are presented as follows:

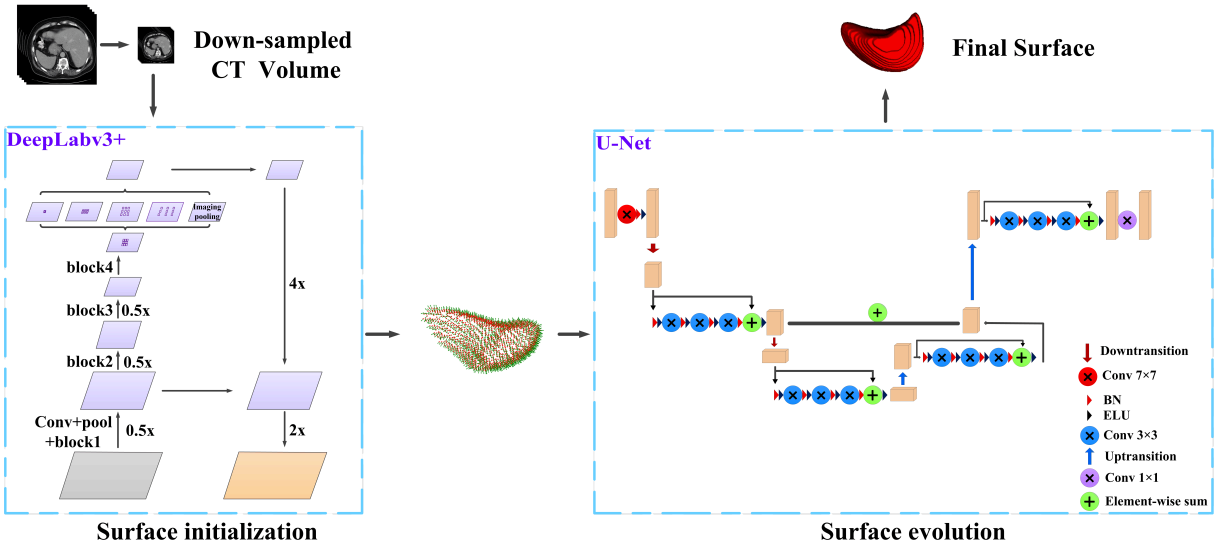


Figure 1: The pipeline of the learned snakes model including two basic modules, i.e., surface initialization and surface evolution.

- We propose a novel segmentation framework based on surface evolution for dealing with 3D image segmentation problems, which is motivated by the classical snakes model to predict the boundary of an object rather than every pixel inside it;
- We develop a surface initialization method by selecting equidistance points from a rough segmented volume to possess similar topological shape as the desired surface, which can guarantee a fast convergence in the surface evolution stage;
- Based on a two-stage surface initialization and evolution formulation, our learned snakes model can realize the 3D segmentation efficiently through 2D deep networks, which can incorporate rich spatial information by consuming much less memories than 3D convolutional neural networks.

The rest of the paper is organized as follows. We present different modules of the proposed learned snakes model in Section 2, including surface initialization, surface evolution, the iteration and inference in testing and 3D reconstruction. Section 3 is dedicated to providing the implementation details of our learned snakes model. Evaluations and experimental results are presented in Section 4. Finally, we conclude the paper and discuss possible future works in Section 5.

2. Learned snakes model

2.1. The overall framework

We consider a 3D CT scan F with size $H_x \times H_y \times H_z$, where the intensity at a specified position is denoted as $F(x,y,z)$. The label data V shares the same dimension with F. Our goal

is to segment a volumetric object U from the 3D image F through end-to-end networks. We follow the traditional snakes model to first estimate an initial surface and then evolve the surface to the desired boundary, both of which are done by deep convolutional neural networks. The pipeline of the proposed learned snakes is illustrated in Figure 1, which contains two separate modules.

2.2. Surface initialization

It is well-known that the initial surface (or contour) is important to the classical snakes model. We propose a fully automatic surface initialization method based on a deep neural network. Assume the initial surface to be S_0, which is used to evolve the final surface.

Since the initial surface is composed of a set of 3D points rather than accurate segmentation, the down-sampled data can be used as inputs to reduce the computational costs in surface initialization. To be specific, we cut down the sizes of 3D CT scans with a fixed step size $dx \times dy \times dz$ along the three dimensions, respectively. Then, we use the well-established segmentation framework DeepLabv3+ [13] as our baseline net, the encoder-decoder structure with atrous spatial pyramid pooling (ASPP) module of which can not only encode multi-scale contextual information but also capture sharper object boundaries. We adopt the ResNet-50 as the network backbone with the *output stride* being 16, where the *output stride* denotes the ratio of input image spatial resolution to the final output resolution. For dealing with small-scale image segmentation problem, we modify the DeepLabv3+ by using stride=1 rather than stride=2 in the first convolution layer to keep more structural information; see Figure 1. Thus, the input features of the ASPP module are with *output stride*=8. Accordingly, the encoder features are upsampled by bilinear interpolation with a factor of 4, and then concatenated with the corresponding low-level features from the first block of the network backbone. Finally, another bilinear upsampling is applied with a factor of 2 to recover the original image size. The whole segmentation process can be expressed as follows

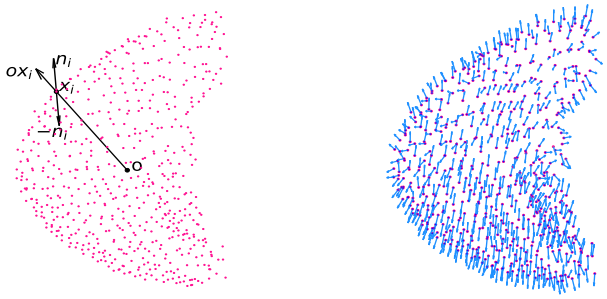
$$P = \textbf{D-net}[F_d; \Theta_{\mathrm{I}}], \tag{6}$$

where $\textbf{D-net}[\cdot; \Theta_{\mathrm{I}}]$ denotes the DeepLabv3+ network with Θ_{I} being network parameters, F_d is the down-sampled data as input of network, and P denotes the probability map outputted by DeepLabv3+. We can obtain the segmentation by binarizing P into W using a given threshold ε, i.e., $W = \mathbb{I}[P \geq \varepsilon]$, where the threshold can be estimated using the automatic clustering methods such as K-means [37, 38]. Here, we simply set $\varepsilon = 0.5$ in numerical experiments, because there is no obvious improvement of segmentation accuracy by varying the threshold.

Then, we estimate the initial surface by selecting equidistant points inside the object of W such as

$$\mathsf{S}_0 = \{\boldsymbol{x} \mid \min_{\boldsymbol{y} \in \partial \mathrm{W}} \mathrm{d}(\boldsymbol{x}, \boldsymbol{y}) = d_o \ \text{ and } \ W(\boldsymbol{x}) = 1\}, \tag{7}$$

where $d(\boldsymbol{x}, \boldsymbol{y})$ denotes the Euclidean distance between the two three-dimensional points $\boldsymbol{x}$ and $\boldsymbol{y}$, and d_o represents the user-specified distance between the initial surface S_0 and the estimated



(a) (b)

Figure 2: Estimation of normal directions. (a) Determination of the outward normal vector. (b) The estimated initial surface S_0 together with the point-wise normal vectors.

boundary $\partial\mathrm{W}$. The initial surface obtained by (7) is a point cloud, i.e., $\mathsf{S}_0 = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_K\}$ with K denoting the total number of points. More importantly, the obtained initial surface S_0 has the similar topological shape as the organ to be segmented. Finally, the coordinates of $\boldsymbol{x}_i \in \mathsf{S}_0$, $i = 1, \ldots, K$, are upsampled to the original resolution using the stepsize $dx \times dy \times dz$ for surface evolution.

2.3. Surface Evolution

Given an initial surface S_0, we develop a learning-based surface evolution method to deform S_0 to match with the desired boundary, which is realized by a light-weighted U-net as follows.

2.3.1. Normal direction estimation

The normal direction of each point in S_0 is estimated as the normal vector of the plane fitting to its local neighborhood. To be specific, we denote the nearest 8 neighbors of $\boldsymbol{x}_i$ as $\mathrm{Neigh}(\boldsymbol{x}_i)$, $\forall \boldsymbol{x}_i \in \mathsf{S}_0$, $i = 1, \ldots K$. Then the fitting plane of $\boldsymbol{x}_i$ can be defined as

$$ax + by + cz + d = 0, \quad \text{s.t.,} \;\; a^2 + b^2 + c^2 = 1, \tag{8}$$

where $a, b, c, d \in \mathbb{R}$ are coefficients chosen to fit with $\mathrm{Neigh}(\boldsymbol{x}_i)$. The normal vector is then given as $\boldsymbol{n}_i = [n_x, n_y, n_z] = \pm[a, b, c]$. In order to obtain the outward normal vector, we select a point $\boldsymbol{o}$ inside the surface S_0 as the viewpoint to choose the normal vectors satisfying

$$\boldsymbol{o}\boldsymbol{x}_i \cdot \boldsymbol{n}_i > 0,$$

as illustrated in Figure 2 (a). We also display the initial surface S_0 associated with the normal directions in Figure 2 (b).

2.3.2. Initial surface encryption

Considering that the number of points in S_0 is much smaller than the one of the desired surface, we encrypt the initial surface in advance. Here, two different methods are adopted using either a hemisphere or a plane to encrypt a point.

- **Hemisphere.** To guarantee the target boundary can be smoothly covered after the evolution, the encrypted points and the associated directions should be dispersed as much as possible. Therefore, we use the spheres to encrypt the pivots. For each pivot $\boldsymbol{x}_i \in \mathsf{S}_0$, $i = 1,2,\ldots,K$, we define a ball $B_i := \{\boldsymbol{b}_{i,j,k}\}$ satisfying

$$\begin{cases} b_{i,j,k}(x) &= \boldsymbol{x}_i(x) + r\sin\alpha_j\cos\beta_k, \\ b_{i,j,k}(y) &= \boldsymbol{x}_i(y) + r\sin\alpha_j\sin\beta_k, \\ b_{i,j,k}(z) &= \boldsymbol{x}_i(z) + r\cos\alpha_j, \end{cases} \tag{9}$$

  where r denotes the radius of the sphere, $\alpha_j \in [0,2\pi]$, $\beta_k \in [0,\pi]$ denote the azimuth angle and elevation angle, and $j = 0,\ldots,2h-1$, $k = 0,\ldots,w-1$ with $2h$ and w being the numbers of azimuth angles and elevation angles, respectively. In order to keep the points uniformly distributed over the sphere, the $2h$ azimuth angles are uniformly distributed, while the w elevation angles have a denser distribution near the equator, i.e., $\alpha_j = 2\pi j/(2h)$ and $\beta_k = \cos^{-1}(2k/(w+1)-1)$. Followed the idea of snakes, we deform the surface along its outward normal direction. Therefore, we rotate the ball B_i to its normal direction $\boldsymbol{n}_i$ and keep the half of the points along the normal direction, which gives an encrypted hemisphere. More specifically, the rotation angle is the angle between z-axis and the normal of $\boldsymbol{x}_i$ such as $\theta = \arccos(\frac{\boldsymbol{z}\cdot\boldsymbol{n}}{|\boldsymbol{z}||\boldsymbol{n}|})$ with $\boldsymbol{z} = (0,0,1)$, and the rotation axis is $\boldsymbol{u} = \boldsymbol{z}\times\boldsymbol{n}$. We can compute the Rodrigues' rotation matrix R by

$$R = \begin{pmatrix} \cos\theta+\boldsymbol{u}_x^2(1-\cos\theta) & -\boldsymbol{u}_z\sin\theta+\boldsymbol{u}_x\boldsymbol{u}_y(1-\cos\theta) & \boldsymbol{u}_y\sin\theta+\boldsymbol{u}_x\boldsymbol{u}_z(1-\cos\theta) \\ \boldsymbol{u}_z\sin\theta+\boldsymbol{u}_x\boldsymbol{u}_y(1-\cos\theta) & \cos\theta+\boldsymbol{u}_y^2(1-\cos\theta) & -\boldsymbol{u}_x\sin\theta+\boldsymbol{u}_y\boldsymbol{u}_z(1-\cos\theta) \\ -\boldsymbol{u}_y\sin\theta+\boldsymbol{u}_x\boldsymbol{u}_z(1-\cos\theta) & \boldsymbol{u}_x\sin\theta+\boldsymbol{u}_y\boldsymbol{u}_z(1-\cos\theta) & \cos\theta+\boldsymbol{u}_z^2(1-\cos\theta) \end{pmatrix}.$$

  Then we rotate B_i by $B_i^T = \{\boldsymbol{b}_{i,j,k}^T \mid \boldsymbol{b}_{i,j,k}^T = R\boldsymbol{b}_{i,j,k}\}$ and select the half of the points on B_i^T satisfying $\boldsymbol{b}_{i,j,k}^T \cdot \boldsymbol{n}_i \geq 0$. We further equip the encrypted points with the unique directions

$$\boldsymbol{d}_{i,j,k} = \boldsymbol{b}_{i,j,k}^T - \boldsymbol{x}_i, \tag{10}$$

  for $j = 0,\ldots,h-1$, $k = 0,\ldots,w-1$.

- **Tangent plane.** We can also encrypt the pivot points using their tangent planes. That is, for each point $\boldsymbol{x}_i \in \mathsf{S}_0$, $i = 1,2,\ldots,K$, we project the encrypted hemisphere onto its tangent plane by carrying on their directions from the hemisphere.

After this process, each pivot point $\boldsymbol{x}_i \in \mathsf{S}_0$, $i = 1,\ldots,K$, is encrypted into a set of points, which is denoted as $X_i = \{\boldsymbol{x}_{i,j,k} \mid j = 0,\ldots,h-1,\ k = 0,\ldots,w-1\}$ associated with the directions $D_i = \{\boldsymbol{d}_{i,j,k} \mid j = 0,\ldots,h-1,\ k = 0,\ldots,w-1\}$. Note that both X_i and D_i are stored as matrixes of the size $h \times w$.

2.3.3. Learned snakes model

Now we develop our learned snakes model, which can deform the initial surface to the desired surface by estimating the distances in between them through a deep neural network.

Evolution distance. We employ the shortest distances in between the encrypted points and the desired boundary to evolve the surface. Suppose Ω^+ is the organ region, Ω^- is the background and $\partial\Omega$ is the interface. The signed distance function is used to estimate the shortest distance of a point $\boldsymbol{x}$ to the boundary such as

$$\Phi(\boldsymbol{x}) = \begin{cases} d(\boldsymbol{x}, \partial\Omega), & \text{if } \boldsymbol{x} \in \Omega^+, \\ 0, & \text{if } \boldsymbol{x} \in \partial\Omega, \\ -d(\boldsymbol{x}, \partial\Omega), & \text{if } \boldsymbol{x} \in \Omega^-, \end{cases}$$

where the sign of $\Phi(\boldsymbol{x})$ indicates the position of point $\boldsymbol{x}$ relative to $\partial\Omega$, i.e., positive values indicating inside $\partial\Omega$, negative values indicating outside $\partial\Omega$, and 0 indicating on $\partial\Omega$. We use the KD-tree algorithm to estimate the distance map and multiply the distance map by -1 for outside points.

Data generation. Given the initial point set $X_i = X_i$, $i = 1, \ldots, K$ and the signed distance function $\Phi(X_i^0)$, we can define an input-output data pair as follows

$$\mathrm{I}_i = F(X_i), \quad \text{and} \quad \mathrm{O}_i = \Phi(X_i),$$

where the inputs are intensities of the points used to incorporate more image features in the learning process. However, owing to the use of fixed directions (10), the initial surface can not reach the desired position by evolving with the shortest distances. Therefore, we use an iterative procedure to generate the input-output pairs to reduce the distances between the estimated and desired surfaces. To be specific, let $X_i^0 = X_i$ and $\Lambda_i^0 = 0$, $i = 1, \ldots, K$, be the initial positions and offsets, respectively, and assume M be the maximal number of iterations. Then we use the fixed directions and the signed distance function to evolve the points as follows

$$X_i^m = X_i^0 + \Lambda_i^m \circ D_i, \quad \text{for } m = 1, \ldots, M, \tag{11}$$

where $\circ$ denotes the point-wise matrix multiplication and the total evolution distance Λ_i^m need to satisfy

$$\Lambda_i^m = \max\left\{\Lambda_i^{m-1} + \Phi(X_i^{m-1}), 0\right\},$$

to guarantee the points not moving towards the opposite directions. Consequently, we generate a series of input-output pairs to train the network, which are

$$\begin{cases} \mathrm{I}_i^m = F(X_i^m), \\ \mathrm{O}_i^m = \Phi(X_i^m). \end{cases} \tag{12}$$

Because the encrypted points are identically determined by the two angles $\{\alpha_j\}_{j=0}^{h-1}$ and $\{\beta_k\}_{k=0}^{w-1}$, we can use 2D convolutional networks to estimate the evolution distances, which means 3D segmentation tasks are realized by 2D networks. In addition, we introduce more channels to input data followed the idea of [1] to enrich information for accurate prediction. Concretely, we sample the boundary and inner volume as follows

$$\begin{cases} X_{i,\ell^A}^m = X_i^0 + (\Lambda_i^m + \ell^A) \circ D_i, & \ell^A \in \{-2, -1, 0, 1, 2\}, \\ X_{i,\ell^B}^m = X_i^0 + \frac{\ell^B}{6} \Lambda_i^m \circ D_i, & \ell^B \in \{1, 2, 3, 4, 5\}. \end{cases} \tag{13}$$

Table 2: Configurations of the U-net architecture in surface evolution, where Conv-BN-ELU represents a sequence of convolution, batch normalization and ELUS.

	kernel size	dilated factor	stride	padding	feature size
Input					16×32×32
Conv-BN-ELU	7×7		1×1	3×3	16 ×32×32
Downtransition	3×3		2×2	1×1	64×16×16
Dilated_Conv-BN-ELU	3×3	2×2	1×1	2×2	64×16×16
Dilated_Conv-BN-ELU	3×3	2×2	1×1	2×2	64×16×16
Dilated_Conv-BN-ELU	3×3	2×2	1×1	2×2	64×16×16
Downtransition	3×3		2×2	1×1	128×8×8
Dilated_Conv-BN-ELU	3×3	2×2	1×1	2×2	128 ×8×8
Dilated_Conv-BN-ELU	3×3	2×2	1×1	2×2	128×8×8
Dilated_Conv-BN-ELU	3×3	2×2	1×1	2×2	128×8×8
Uptransition	4×4		2×2	1×1	64×16×16
Conv-BN-ELU	3×3		1×1	1×1	64×16×16
Conv-BN-ELU	3×3		1×1	1×1	64×16×16
Conv-BN-ELU	3×3		1×1	1×1	64×16×16
Uptransition	4×4		2×2	1×1	16×32×32
Conv-BN-ELU	3×3		1×1	1×1	16×32×32
Conv-BN-ELU	3×3		1×1	1×1	16×32×32
Conv-BN-ELU	3×3		1×1	1×1	16×32×32
Conv-BN-ELU	1×1		1×1	1×1	1×32×32

Besides, we also pull in the directions into the inputs by considering the following 16 channels data

$$\mathrm{I}_i^m = [F(X_{i,\ell^A}^m), D_i, F(X_{i,\ell^B}^m), D_i],$$

which is a typical method used to encourage the 2D deep neural network to implicitly learn more 3D spatial information. On the other hand, the number of the channel for the output O_i^m remains 1.

Optimization. After generating the training data pairs $\{\mathrm{I}_i^m, \mathrm{O}_i^m\},\ i = 1,2,\ldots,K,\ m = 0,1,\ldots,M$, we use an encoder-decoder U-Net pathway with the concatenations of feature maps to estimate the evolution distances (see Figure 1)

$$Q_i^m = \textbf{U-net}[\mathrm{I}_i^m; \Theta_{\mathrm{II}}], \tag{14}$$

where $\textbf{U-net}[\cdot;\Theta_{\mathrm{II}}]$ denotes the U-Net with Θ_{II} being network parameters, and Q_i^m denotes the estimated distance map. The detailed configuration of the CNN architecture is displayed in Table 2, where each convolutional layer is associated with a batch normalization layer to

speed up the convergence of the training process, and an exponential linear units (ELUS) layer [39] is used as the activation function. Besides, we use the dilated convolutions to obtain a larger receptive field and the identity short-cut connection to improve the network representation ability. We configure the network to downsample along each axis until the feature map size for that axis reaches 8 or the maximal number of down-sampling operation reaches 3.

In the training phase, we adopt the following Huber loss to train the U-net to guarantee the better convergence

$$\mathscr{L}_\delta = \begin{cases} \frac{1}{2}\sum_{j,k}(Q_i^m(j,k)-\mathrm{O}_i^m(j,k))^2, & \text{if } |Q_i^m(j,k)-\mathrm{O}_i^m(j,k)| \le \delta, \\ \delta\sum_{j,k}\left(|Q_i^m(j,k)-\mathrm{O}_i^m(j,k)|-\frac{1}{2}\delta\right), & \text{otherwise,} \end{cases} \tag{15}$$

where δ is a hyper-parameter need to be selected through the cross-validation.

2.4. Iteration and inference in testing

The inference stage is similar to the training stage, which is summarized in Algorithm 1. In the inference stage, we fix the parameters Θ_{I} , Θ_{II} and iteration number T to estimate the offsets of Q_i^t, $i = 1,2,\ldots,K$, $t = 1,2,\ldots,T$. By adding the total evolution distances along the pre-defined directions, we obtain the ending point cloud S_T.

Algorithm 1: Learned snakes model

Input: The CT image F;

`/* Surface initialization */`

Obtain F_d by down sampling the CT scan F;

$P \leftarrow$ **D-net**$[F_d;\Theta_{\mathrm{I}}]$, $W = \mathbb{I}[P \ge 0.5]$;

Estimate the initial surface S_0 using (7);

`/* Surface evolution */`

for $\boldsymbol{x}_i \in \mathsf{S}_0$, $i = 1,2,\ldots,K$ **do**

  Estimate the normal vector $\boldsymbol{n}_i$;

  Encrypt $\boldsymbol{x}_i$ to X_i and define the directions D_i;

  for $t = 0,1,\ldots,T-1$ **do**

    Estimate X^t_{i,ℓ^A} and X^t_{i,ℓ^B} using (13);

    $\mathrm{I}_i^t = [F(X^t_{i,\ell^A}), D_i, F(X^t_{i,\ell^B}), D_i]$;

    $Q_i^t \leftarrow$ **U-net**$[\mathrm{I}_i^t;\Theta_{\mathrm{II}}]$;

    $\Lambda_i^{t+1} = \max\{\Lambda_i^t + Q_i^t, 0\}$;

    $X_i^{t+1} = X_i^0 + \Lambda_i^{t+1}\cdot D_i$;

  end

end

Output: The ending point clouds S_T.

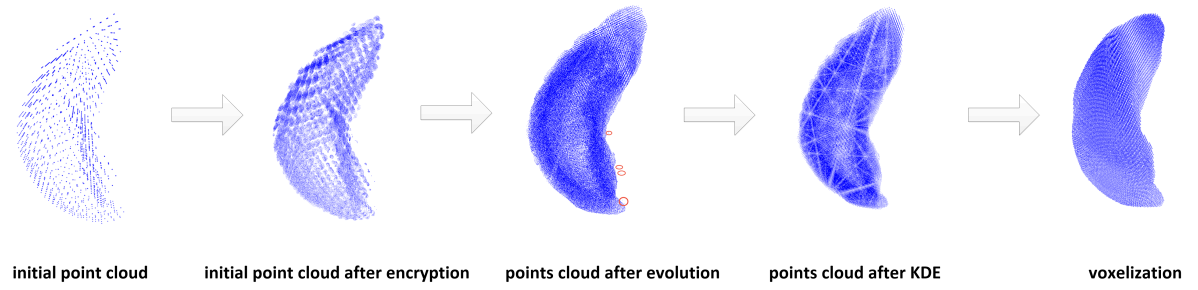


Figure 3: The 3D reconstruction pipeline of our learned snakes model. We start with the initial point cloud and point cloud encryption, then deform the point cloud to the true surface, and finally voxelize the point cloud.

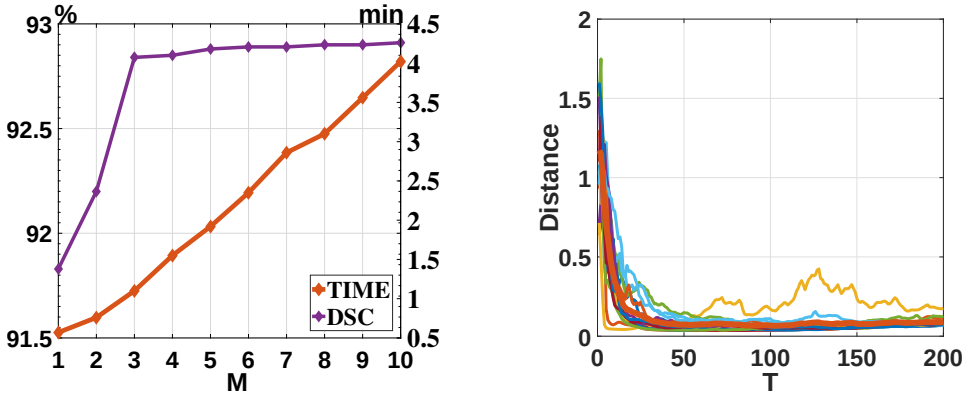
2.5. 3D reconstruction

After the surface evolution, there are still some predicted ending points in S_T that do not converge to the true boundary. Because the initial surface contains outnumbered points after the encryption, we directly use the kernel density estimation (KDE) to remove these isolated points. Specifically, we use the Epanechnikov kernel with bandwidth 1 to estimate a log-likelihood value, and preserve the integer points whose value are not smaller than -14. Then a three-step graphics processing is conducted to reconstruct the 3D image volume from the point cloud. In particular, we first build up the triangle mesh using the Delaunay triangulation and remove the improper tetrahedrons with a circumradius larger than α. Here, α is a threshold used to filter out the tetrahedrons of large sizes, the default value of which is set as $\alpha = 2$ in our implementation. Then we use the subdivide algorithm to obtain the surface using the alpha shape data and we only keep the largest component. Finally, we fill the closed boundary to obtain the segmentation result. The 3D pipeline of our model is illustrated in Figure 3, where the initial point cloud is encrypted using the points locating on the associate tangent planes.

3. Implementation details

We use the spleen subset of the Medical Segmentation Decathlon (MSD) dataset for evaluation. Because the ground truths of the testing dataset are not publicly available, we solely use the training dataset, which contains 41 volumetric CT data. More specifically, we randomly split the 41 data into the training dataset (21 volumes) and testing dataset (20 volumes). The number of slices containing the spleen varies between 31 and 168, and the images are of the size 512×512. Similar to [1], we truncate the image intensities of all scans to the range of [-125,275] and apply the min-max normalization for a fair comparison. Another popular way is to clip the intensities to the [0.5, 99.5] percentiles of the foreground intensities in the training dataset as done by nnU-Net in [40].

The Dice Similarity Coefficient (DSC) is used to evaluate the performance of our learned snakes model and the competitive methods. The DSC measures the ratio between twice the



(a) (b)

Figure 4: The influences of the number of iteration in training phase and numerical convergence in the testing stage. (a) The DSC and computational time versus the number of iterations M; (b) The distances between the estimated surface and the desired surface in the testing stage. Here, each thin curve represents the distance (17) of a randomly selected encrypted point set, while the thick orange curve is the averaged distance of 15 randomly selected point sets.

number of elements common to two sets to the sum of number of elements in each set, i.e.,

$$\mathrm{DSC} = \frac{2|U \cap V|}{|U| \cup |V|}, \tag{16}$$

where $|U|$ and $|V|$ denote the cardinalities of the segmentation and ground truth, respectively. The value of the DSC is within the range [0,1] such that 1 indicates perfect overlap and 0 indicates no overlap between U and V.

In our implementation, we truncate the values of the distance maps in both training and testing within the range $[-\tau, \tau]$ for achieving better results, where τ is chosen as $\tau = 2$ in all the experiments. The value of δ in Huber loss (15) is selected by the cross-validation, the value of which ranges from $\delta \in [0.5, 2]$.

3.1. Model setup

In this subsection, we analyze the choices of the iteration number, the numerical convergence and discuss the influences of the distance between the initial surface and target surface, the radius of the hemisphere, and the number of encrypted points to the performance of our learned snakes model.

3.1.1. The choices of the iteration number

On the first place, we discuss the influences of the number of iteration M in the training phase to the performance of our model. In the experiment, the distance between the initial surface and desired surface is chosen as $d_o = 1$, and the tangent plane encryption is used with the radius of $r = 1$ and the size of $h = w = 32$. We compare the segmentation results obtained by different models trained using input-output data pairs generated by $M = 1$ to $M = 10$. Both

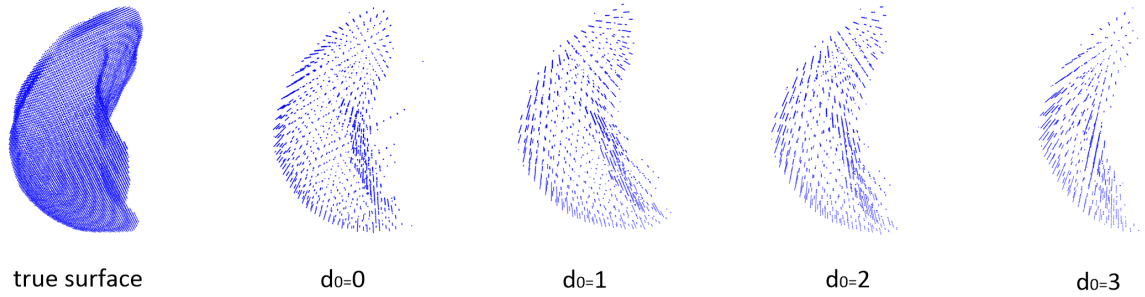


Figure 5: The topological shapes of the initial surfaces obtained using different distance d_o, which reduce as the distance increasing.

the DSC and computational time are displayed in Figure 4 (a). Note that the number of iteration in the testing phase satisfies $T \leq M$, which is chosen for achieving the best results. As can be seen, the value of DSC first increases then becomes stable, while the computational time keeps increasing as M goes from $M = 1$ to $M = 10$. Thus, to balance the efficiency and effectiveness, we choose $M = 3$ unless otherwise specified.

3.1.2. Numerical convergence

For each pivot point $\boldsymbol{x}_i \in \mathsf{S}_0$, $i = 1, \ldots, K$, we use the average distances of the encrypted point sets to measure the convergence in the testing phase, which is defined as

$$l = \frac{1}{h \times w} \sum_{j=1}^{h} \sum_{k=1}^{w} \left| Q_i^t(j,k) \right|. \tag{17}$$

We perform our model trained by $M = 3$ for $T = 200$ iterations in the testing phase and plot both the point-wise distances and the average distance in Figure 4 (b). It is shown that the average distance converges to 0 and remains stable as the iteration goes to 200 iterations, which means the initial surface converges to the objective surface after sufficient iterations. Although more than 50 iterations are required for a nice convergence, i.e., $l < 0.1$, we only evolute $T \leq M$ iterations in our implementation. The reason is that the density of points on the boundary after three evolutions is already enough for surface reconstruction. We can simply remove the non-convergent isolated points during 3D reconstruction.

3.1.3. The influences of the distance between the initial surface and target surface

With the rough segmentation obtained by DeepLabv3+, we generate an initial surface S_0 by selecting equidistant points according to (7). Here, the distance d_o between selected points and the object boundary is another important parameter need to be chosen. Thus, we examine the performance of our LSM with initial surfaces generated by different d_0 including $d_o = 0$, 1, 2, and 3, respectively. As shown in Figure 5, all the initial surfaces have similar shapes as the true surface, and they shrink as the distance d_o increases by losing the structural details. We also evaluate the segmentation accuracy in Table 3, where the distance d_o affects not only the

Table 3: The influences of the initial distance to our learned snakes model, where the models are trained with different numbers of iteration M to achieve the best segmentation accuracies.

Distance d_o	0			1			2			3		
Number of pivots K	1814			1474			1233			1037		
Evaluation metrics	DSC	M	Time	DSC	M	Time	DSC	M	Time	DSC	M	Time
Hemisphere	90.29	4	1.65	92.02	7	2.86	91.7	8	2.23	89.08	8	1.90
Tangent plane	91.8	2	1.21	92.84	3	1.10	91.79	4	0.89	89.27	5	0.78

number of the pivot points in the initial surface set but also the convergence of the model. In particular, we summarize the principles to determine the distance d_o as follows:

- Our LSM requires more iterations to converge as the distance d_o increases for both hemisphere and tangent plane encryption. On the other hand, the computational time is affected by not only the number of pivot points contained in the initial surface but also the number of iteration. On average, the distance d_o should not be too large, e.g., $d_0 \geq 3$.

- Because the segmentation of DeepLabv3+ on the boundaries is not as accurate as of the interior region, it is unwise to choose a too small d_o, e.g., $d_o = 0$, which may result in an inaccurate surface initialization.

- The models with encryption by either hemisphere or tangent plane perform in a similar way. As can be seen, the model using tangent planes encryption gives better segmentation accuracy and converges better during training. Therefore, we suggest to encrypt the pivot points using their tangent planes.

To sum up, we let the distance be $d_o = 1$ and use the tangent plane to encrypt the initial surface for the spleen segmentation task.

3.1.4. The influences of the number of points in encryption

We also evaluate the influences of the radius and the dimension of the encryption to the performance of our LSM on the model trained with $M = 3$. Considering that the distance d_o is fixed as $d_o = 1$, we increase the radius of the tangent plane from $r = 0.5$ to $r = 3$ with the step size of 0.5 and increase the dimension of encryption from $h = w = 32$ to $h = w = 128$. We perform our LSM and record the DSC values in Table 4. We observe that the best accuracy is achieved by $r = 2$ and $h = w = 32$, which gives a DSC of 93.05. Besides, we also have the following conclusions on the choices of the radius and the dimension of the encryption

- As the radius r increases, the DSC first increases and then decreases. When the radius becomes larger and larger, more and more encrypted points lie outside the object resulting in the decline of the DSC.

- By increasing the number of encryption points, the DSC almost remains the same, while the computational time keeps increasing. Therefore, the dimension of the encryption should not be too large.

Table 4: The DSC comparison of encryptions with respect to different radius and number of points, where time is recorded in minute (min).

$h \times w$ \ r	0.5	1	1.5	2	2.5	3	Time
32×32	92.49	92.84	92.86	**93.05**	92.52	91.8	4.82
64×64	92.49	92.85	92.86	92.61	92.02	91.84	8.66
128×128	92.53	92.60	92.92	92.2	91.97	91.79	24.13

Table 5: The comparison of DSCs on models trained by different numbers with iteration M and surface initialization methods.

Initialization \ M	1	2	3	4	5	6	7	8	9	10
Equidistant	90.86	92.21	93.05	93.03	93.06	93.05	93.04	93.06	93.06	93.06
Random	89.25	90.41	91.47	91.65	91.58	91.64	91.6	91.53	91.69	91.67

- The accuracy may decrease when the dimension of the encryption increases. That is because the difficulty to train the U-net in surface evolution increases to a certain extent with more points.

Therefore, we set the radius $r = 2$ and the number of the encrypted points $h = w = 32$ for spleen segmentation.

4. Experimental results

4.1. Comparison with random initialization

To illustrate the importance of the initialization, we compare the performance of our LSM with respect to both equidistant initialization or random initialization. Here the random initialization means to randomly select pivot points inside the spleen volumes obtained by DeepLab-v3+. In this experiment, the same number of the pivot points, i.e., $K = 1474$, and the tangent plane encryption with $r = 2$ and $h = w = 32$ are adopted for all models for a fair comparison. In particular, we compare the two initializations on models trained by different numbers of iteration from $M = 1$ to $M = 10$. As shown in Table 5, the models with equidistant initialization always produce higher DSCs. It is well-known that the classical snakes model is sensitive to the initializations, which may be fail to converge to the desired boundaries for bad initializations. As seen, our LSM works well with both initializations. And of course, the shape-preserving initialization can help to improve the segmentation accuracy since it is easier for the U-net to learn features through data with geometric similarities than the messy data. Besides, selective

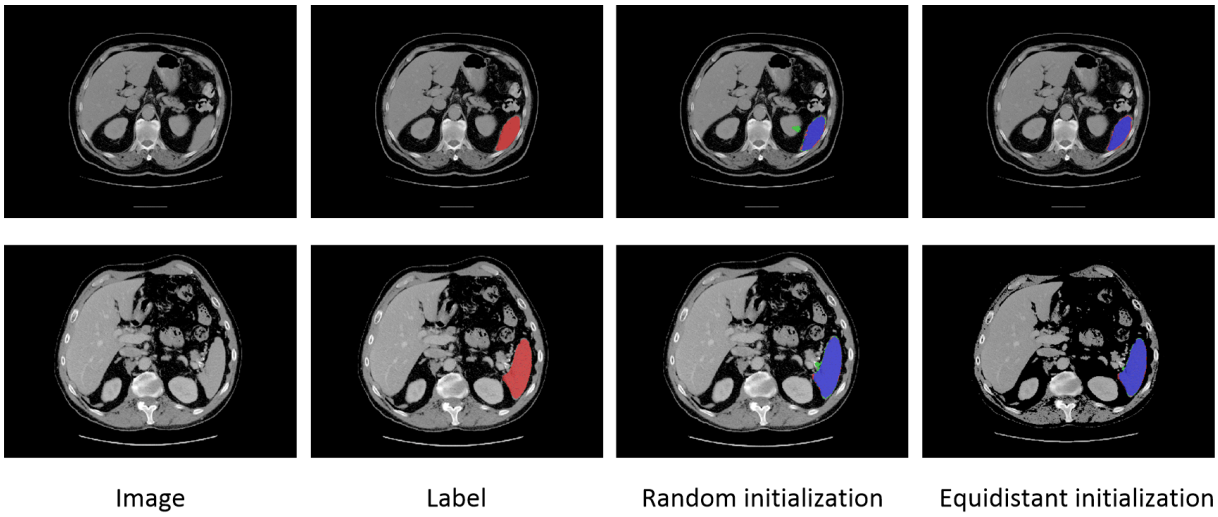


Figure 6: Selective 2D visualization of segmentation results by random initialization and shape-preserving initialization, where red, green and blue indicate the ground truth, prediction and overlapped region, respectively.

visual comparison results are provided in Figure 6 to evaluate the different behaviors of the two models. We observe that the random initialization may result in over-segmentation around the boundaries, and the equidistant initialization can produce better segmentation results with more accurate boundaries.

4.2. Comparison with other methods

We evaluate the performance of our LSM by comparing it with three recently published baseline models named RSTN [18], VNet [21], and EBP [1], which are representative methods for 3D medical image segmentation. The RSTN is a typical 2D based network, which feeds 3 neighboring slices to the 2D network along each dimension and then fuses three views, i.e., coronal, sagittal, and axial view, by majority voting to obtain the final prediction. The VNet is a classical 3D network, which randomly crops the 3D image into $128 \times 128 \times 64$ patches for training due to the limitation of GUP memory. The EBP bridges the gaps between 2D and 3D models, which processes the 3D data as a whole rather than cutting it into slices. In the implementation, both the VNet and EBP need the region-of-interest (ROI) information, while the RSTN and our LSM are realized based on the original raw data. We report the quantitative results including the average DSC with the associated standard deviation, the best and worst accuracies in Table 6. Note that the values of the RSTN, Vnet, and EBP are extracted from [1] since the dataset and settings are all the same. As shown, our model achieves better segmentation accuracy with 1% higher DSC than the EBP model. We also record the segmentation accuracy in surface initialization for a comparison. It is clear that the evolution step is important to increase the segmentation accuracy, both the best/worst and average DSC are significantly improved in surface evolution. Thanks to the initialization, we can estimate the desired surface through a light-weighted network on a point set with much fewer pivots. More importantly, the convergence of our model is faster than EPB as well by saving much computational time. We

Figure 7: An example of the visual results on the MSD spleen dataset, where three planar (coronal, sagittal, and axial) views and the corresponding volumetric view are shown. In the figure, the red and green colors indicate the prediction and ground truth, respectively.

also display a typical segmentation result of our method in Figure 7, the boundaries of which well approximate the desired boundaries in three different views.

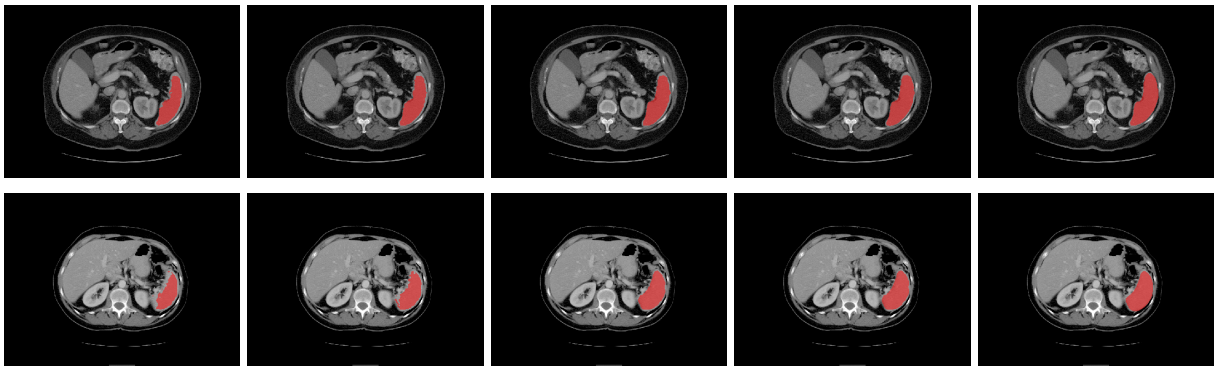
Table 6: Comparison of segmentation results on MSD spleen dataset, where the computational time of our initialization and LSM include the time consumed by 3D reconstruction.

Approach	Average DSC	Max	Min	Parameters	Time (min)	Bounding box
RSTN	89.70±12.60	97.25	48.45	8.1e8	–	N
VNet	92.94±3.58	**97.35**	81.96	4.6e7	–	Y
EBP	92.01±4.50	96.48	77.07	2.3e6	59.87	Y
Initialization	89.87±2.20	92.94	82.67	4.0e7	2.83	N
LSM	**93.05±2.26**	96.14	**88.12**	5.3e5	4.75	N

To better understand the iterative process of our LSM in the testing stage, we further look into the intermediate segmentation results, which are provided in Figure 8. In accord with our formulation, the initial surfaces are chosen inside the 3D object and have a similar topological shape as the desired ones. However, the boundaries of the initial contours may be not continuous and smooth. Thanks to the iterative process in the surface evolution stage, the accuracy of the segmentation is significantly improved, the estimated boundaries of which steadily approach to the desired boundaries.

5. Conclusion

In this paper, we proposed a learned snakes model followed the classical snakes framework to deal with 3D medical image segmentation tasks. To be specific, we used a 2D surface embedded in the 3D space to approximate the surface of the object to be segmented. To overcome the difficulties in surface initialization and evolution, we developed a two-stage segmentation method, both stages of which are realized by deep neural networks. We first applied the modified segmentation network Deeplabv3+ on the down-sampled raw data to obtain coarse segmentations. Then, we generated the initial surfaces by choosing equidistant points from the



(a) Initialization (b) 1st iteration (c) 2nd iteration (d) 3rd iteration (e) Label image

Figure 8: Intermediate segmentation results of our LSM in the testing stage, which are displayed after 3D reconstruction.

segmented volumes. The initial surfaces were further evolved to the final surfaces with the distances estimated by the U-net. We realized the accurate segmentation based on 2D convolutional neural networks without the 3D nets or 3D fusion. Compared to pioneer work EBP, our model provided better accuracy on the MSD spleen segmentation dataset with less than 10% its computational time and storage space.

Currently, most computational costs of our LSM were consumed by 3D reconstruction, which was realized by engineering solutions. Thus, one possible future work is to develop a fast method to achieve the 3D volumes directly from the point cloud data, e.g., the learning-based method [41]. We also would like to improve the performance of the U-net in surface evolution using advanced techniques such as squeeze-and-excitation block [42], convolutional block attention module [43] etc. Alternatively, we may introduce more spatial information in the surface evolution stage to obtain segmentation with smoother boundaries, e.g., curvature regularization [44].

Acknowledgment

The work was supported by the National Natural Science Foundation of China (NSFC 12071345, 11701418), Major Science and Technology Project of Tianjin 18ZXRHSY00160, and Recruitment Program of Global Young Expert. The work of X.C. Tai was supported by Hong Kong Baptist University startup grants RG(R)-RC/17-18/02-MATH and FRG2/17-18/033.

References

[1] T. Ni, L. Xie, H. Zheng, E. K. Fishman, A. L. Yuille, Elastic boundary projection for 3d medical image segmentation, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2019, pp. 2104–2113.

[2] Y. Wang, Y. Zhou, W. Shen, S. Park, E. K. Fishman, A. L. Yuille, Abdominal multi-organ segmentation with organ-attention networks and statistical fusion, Medical Image Analysis 55 (2019) 88–102.

[3] N. Zhao, N. Tong, D. Ruan, K. Sheng, Fully automated pancreas segmentation with two-stage 3d convolutional neural networks, in: Medical Image Computing and Computer Assisted Intervention – MICCAI 2019, Springer International Publishing, 2019, pp. 201–209.

[4] D. Mumford, J. Shah, Optimal approximations by piecewise smooth functions and associated variational problems, Communications on Pure and Applied Mathematics 42 (5) (1989) 577–685.

[5] T. Chan, L. Vese, Active contours without edges, IEEE Transactions on Image Processing 10 (2) (2001) 266–277.

[6] Y. Yang, Q. Zhong, Y. Duan, T. Zeng, A weighted bounded hessian variational model for image labeling and segmentation, Signal Processing 173 (2020) 107564.

[7] M. Kass, A. Witkin, D. Terzopoulos, Snakes: Active contour models, International Journal of Computer Vision 1 (4) (1988) 321–331.

[8] V. Caselles, R. Kimmel, G. Sapiro, Geodesic active contours, International Journal of Computer Vision 22 (1) (1997) 61–79.

[9] S. Biswas, R. Hazra, A new binary level set model using l0 regularizer for image segmentation, Signal Processing 174 (2020) 107603.

[10] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2015, pp. 3431–3440.

[11] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, in: Medical Image Computing and Computer Assisted Intervention - MICCAI 2015, Springer International Publishing, 2015, pp. 234–241.

[12] S. Ren, K. He, R. Girshick, J. Sun, Faster r-CNN: Towards real-time object detection with region proposal networks, IEEE Transactions on Pattern Analysis and Machine Intelligence 39 (6) (2017) 1137–1149.

[13] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, H. Adam, Encoder-decoder with atrous separable convolution for semantic image segmentation, in: Computer Vision – ECCV 2018, Springer International Publishing, 2018, pp. 833–851.

[14] G. Bertasius, J. Shi, L. Torresani, DeepEdge: A multi-scale bifurcated deep network for top-down contour detection, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2015, pp. 4380–4389.

[15] W. Shen, X. Wang, Y. Wang, X. Bai, Z. Zhang, DeepContour: A deep convolutional feature learned by positive-sharing loss for contour detection, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2015, pp. 3982–3991.

[16] J. Chen, L. Yang, Y. Zhang, M. Alber, D. Z. Chen, Combining fully convolutional and recurrent neural networks for 3d biomedical image segmentation, in: Advances in Neural Information Processing Systems, Curran Associates,Inc., 2016, pp. 3036–3044.

[17] Y. Zhou, L. Xie, W. Shen, Y. Wang, E. K. Fishman, A. L. Yuille, A fixed-point model for pancreas segmentation in abdominal CT scans, in: Medical Image Computing and Computer Assisted Intervention - MICCAI 2017, Springer International Publishing, 2017, pp. 693–701.

[18] Q. Yu, L. Xie, Y. Wang, Y. Zhou, E. K. Fishman, A. L. Yuille, Recurrent saliency transformation network: Incorporating multi-stage visual cues for small organ segmentation, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, IEEE, 2018, pp. 8280–8289.

[19] M. I. Razzak, M. Imran, G. Xu, Efficient brain tumor segmentation with multiscale two-pathway-group conventional neural networks, IEEE Journal of Biomedical and Health Informatics 23 (5) (2019) 1911–1919.

[20] Özgün Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, O. Ronneberger, 3d u-net: Learning dense volumetric segmentation from sparse annotation, in: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016, Springer International Publishing, 2016, pp. 424–432.

[21] F. Milletari, N. Navab, S.-A. Ahmadi, V-net: Fully convolutional neural networks for volumetric medical image segmentation, in: 2016 Fourth International Conference on 3D Vision (3DV), IEEE, 2016, pp. 565–571.

[22] Q. Dou, H. Chen, Y. Jin, L. Yu, J. Qin, P.-A. Heng, 3d deeply supervised network for automatic liver segmentation from CT volumes, in: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016, Springer International Publishing, 2016, pp. 149–157.

[23] Y. Xia, L. Xie, F. Liu, Z. Zhu, E. K. Fishman, A. L. Yuille, Bridging the gap between 2d and 3d organ segmentation with volumetric fusion net, in: Medical Image Computing and Computer Assisted Intervention – MICCAI 2018, Springer International Publishing, 2018, pp. 445–453.

[24] X. Li, H. Chen, X. Qi, Q. Dou, C.-W. Fu, P.-A. Heng, H-DenseUNet: Hybrid densely connected UNet for liver and tumor segmentation from CT volumes, IEEE Transactions on Medical Imaging 37 (12) (2018) 2663–2674.

[25] S. Liu, D. Xu, S. K. Zhou, O. Pauly, S. Grbic, T. Mertelmeier, J. Wicklein, A. Jerebko, W. Cai, D. Comaniciu, 3d anisotropic hybrid network: Transferring convolutional features from 2d images to 3d anisotropic volumes, in: Medical Image Computing and Computer Assisted Intervention – MICCAI 2018, Springer International Publishing, 2018, pp. 851–858.

[26] H. Chen, X. Qi, L. Yu, P.-A. Heng, DCAN: Deep contour-aware networks for accurate gland segmentation, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2016, pp. 2487–2496.

[27] J. Duan, J. Schlemper, W. Bai, T. J. W. Dawes, G. Bello, G. Doumou, A. D. Marvao, D. P. O'Regan, D. Rueckert, Deep nested level sets: Fully automated segmentation of cardiac MR images in patients with pulmonary hypertension, in: Medical Image Computing and Computer Assisted Intervention – MICCAI 2018, Springer International Publishing, 2018, pp. 595–603.

[28] R. Roy, T. Chakraborti, A. S. Chowdhury, A deep learning-shape driven level set synergism for pulmonary nodule segmentation, Pattern Recognition Letters 123 (2019) 31–38.

[29] L. Zhang, M. Bai, R. Liao, R. Urtasun, D. Marcos, D. Tuia, B. Kellenberger, Learning deep structured active contours end-to-end, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, IEEE, 2018, pp. 8877–8885.

[30] H. Ling, J. Gao, A. Kar, W. Chen, S. Fidler, Fast interactive object annotation with curve-GCN, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2019, pp. 5252–5261.

[31] S. Peng, W. Jiang, H. Pi, X. Li, H. Bao, X. Zhou, Deep snake for real-time instance segmentation, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2020, pp. 8530–8539.

[32] P. Hu, B. Shuai, J. Liu, G. Wang, Deep level sets for salient object detection, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2017, pp. 540–549.

[33] B. Kim, J. C. Ye, Mumford–shah loss functional for image segmentation with deep learning, IEEE Transactions on Image Processing 29 (2020) 1856–1866.

[34] X. Chen, B. M. Williams, S. R. Vallabhaneni, G. Czanner, R. Williams, Y. Zheng, Learning active contour models for medical image segmentation, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2019, pp. 11632–11640.

[35] S. Gur, L. Wolf, L. Golgher, P. Blinder, Unsupervised microvascular image segmentation using an active contours mimicking neural network, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), IEEE, 2019, pp. 10721–10730.

[36] J. Gu, Z. Fang, Y. Gao, F. Tian, Segmentation of coronary arteries images using global feature embedded network with active contour loss, Computerized Medical Imaging and Graphics 86 (2020) 101799.

[37] X. Li, X. Yang, T. Zeng, A three-stage variational image segmentation framework incorporating intensity inhomogeneity information, SIAM Journal on Imaging Sciences 13 (3) (2020) 1692–1715.

[38] X. Cai, R. Chan, T. Zeng, A two-stage image segmentation method using a convex variant of the mumford–shah model and thresholding, SIAM Journal on Imaging Sciences 6 (1) (2013) 368–390.

[39] D.-A. Clevert, T. Unterthiner, S. Hochreiter, Fast and accurate deep network learning by exponential linear units (elus), https://arxiv.org/abs/1511.07289 (2016).

[40] F. Isensee, J. Petersen, A. Klein, D. Zimmerer, P. F. Jaeger, S. Kohl, J. Wasserthal, G. Koehler, T. Norajitra, S. Wirkert, K. H. Maier-Hein, Abstract: nnU-net: Self-adapting framework for u-net-based medical image segmentation, in: Informatik aktuell, Springer Fachmedien Wiesbaden, 2019, pp. 22–22.

[41] S. Milz, M. Simon, K. Fischer, M. Ppperl, Points2pix: 3d point-cloud to image translation using conditional generative adversarial networks, http://arxiv.org/abs/1901.09280 (2019).

[42] J. Hu, L. Shen, S. Albanie, G. Sun, E. Wu, Squeeze-and-excitation networks, IEEE Transactions on Pattern Analysis and Machine Intelligence 42 (8) (2020) 2011–2023.

[43] S. Woo, J. Park, J.-Y. Lee, I. S. Kweon, CBAM: Convolutional block attention module, in: Computer Vision – ECCV 2018, Springer International Publishing, 2018, pp. 3–19.

[44] Q. Zhong, Y. Li, Y. Yang, Y. Duan, Minimizing discrete total curvature for image processing, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2020, pp. 8280–8289.