

Unsupervised 3D End-to-End Medical Image Registration with Volume Tweening Network

Tingfung Lau[†], Ji Luo[†], Shengyu Zhao, Eric I-Chao Chang, Yan Xu^{*}

arXiv:1902.05020v1 [cs.CV] 13 Feb 2019

Abstract—3D medical image registration is of great clinical importance. However, supervised learning methods require a large amount of accurately annotated corresponding control points (or morphing). The ground truth for 3D medical images is very difficult to obtain. Unsupervised learning methods ease the burden of manual annotation by exploiting unlabeled data without supervision. In this paper, we propose a new unsupervised learning method using convolutional neural networks under an end-to-end framework, *Volume Tweening Network* (VTN), to register 3D medical images. Three technical components ameliorate our unsupervised learning system for 3D end-to-end medical image registration: (1) We cascade the registration subnetworks; (2) We integrate affine registration into our network; and (3) We incorporate an additional invertibility loss into the training process. Experimental results demonstrate that our algorithm is 880x faster (or 3.3x faster without GPU acceleration) than traditional optimization-based methods and achieves state-of-the-art performance in medical image registration.

Index Terms—Registration, unsupervised, convolutional neural networks, end-to-end, medical image

I. INTRODUCTION

Image registration is the process of mapping images into the same coordinate system by finding the spatial correspondence between images (see Figure 1). It has a wide range of applications in medical image processing, such as aligning images of one subject taken at different times. Another example is to match an image of one subject to some predefined coordinate system, such as an anatomical atlas [1].

There has not been an effective approach to generate ground-truth flows for medical images. Widely used supervised methods require accurately labeled ground truth with a vast number of instances. The quality of these labels directly

This work is supported by the National Science and Technology Major Project of the Ministry of Science and Technology in China under Grant 2017YFC0110903, National Science and Technology Major Project of the Ministry of Science and Technology in China under Grant 2017YFC0110503, Microsoft Research under the eHealth program, the National Natural Science Foundation in China under Grant 81771910, the Beijing Natural Science Foundation in China under Grant 4152033, the Technology and Innovation Commission of Shenzhen in China under Grant shenfangai2016-627, Beijing Young Talent Project in China, the Fundamental Research Funds for the Central Universities of China under Grant SKLSD-E-2017ZX-08 from the State Key Laboratory of Software Development Environment in Beihang University in China, the 111 Project in China under Grant B13003. *Dagger* indicates equal contribution. Asterisk indicates the corresponding author.

Tingfung Lau, Ji Luo and Shengyu Zhao are with Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing 100084, China (email: tingflau@outlook.com; geelaw@outlook.com; zsyzzsoft@gmail.com).

Yan Xu is with State Key Laboratory of Software Development Environment and Key Laboratory of Biomechanics and Mechanobiology of Ministry of Education and Research Institute of Beihang University in Shenzhen, Beihang University, Beijing 100191, China (email: xuyan04@gmail.com).

Eric I-Chao Chang and Yan Xu are with Microsoft Research, Beijing 100080, China (email: echang@microsoft.com; xuyan04@gmail.com).

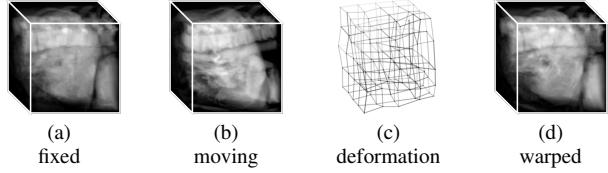


Figure 1: Illustration of 3D medical image registration. Given a fixed image (a) and a moving image (b), a deformation field (c) indicates the displacement of each voxel in the fixed image to the moving image (represented by a grid skewed according to the field). An image (d) similar to the fixed one can be produced by warping the moving image with the flow. The images are rendered by mapping grayscale to white with transparency (the more intense, the more opaque) and dimetrically projecting the volume.

affects the result of supervision, which entails much effort in traditional tasks such as classification and segmentation. But optical flows are dense and ambiguous quantities that are almost impossible to be labeled manually, and moreover, automatically generated dataset (e.g., the Flying Chairs dataset [2]) which deviates from the realistic demands is not appropriate. Consequently, supervised methods are hardly applicable. In contrast, unlabeled medical images are universally available, and sufficient to advance the state of the art through our unsupervised framework shown in this paper.

Previously, there has been much effort on automating image registration. Tools like FAIR [3], ANTs [4] and Elastix [5] have been developed for automated image registration [1]. Generally, these algorithms define a space of transformations and a metric of alignment quality, and then find the optimal transformation by iteratively updating the parameters. The optimization process is highly time-consuming, rendering such methods impractical for clinical applications.

There have been some works tackling medical image registration with CNNs [6]. Miao et al. [7] use CNN to perform 2D/3D registration, in which CNN regressors are used to estimate transformation parameters. Their CNNs are trained over synthetic data and the method is not end-to-end. Wu et al. [8] propose an unsupervised method to train CNN to extract features for 3D brain MRI registration. However, their method is patch-based and needs a feature-based registration method, thus is not end-to-end.

Optical flow prediction is a closely related problem that aims to identify the correspondence of pixels in two images of the same scene taken from different perspectives. FlowNet [2] and its successor FlowNet 2.0 [9] are CNNs that predict optical flow from input images using end-to-end fully convolutional networks (FCN [10]), which are capable of regressing pixel-level correspondence. FlowNet is trained on Flying Chairs, a synthetic dataset that consists of pairs of realistic images

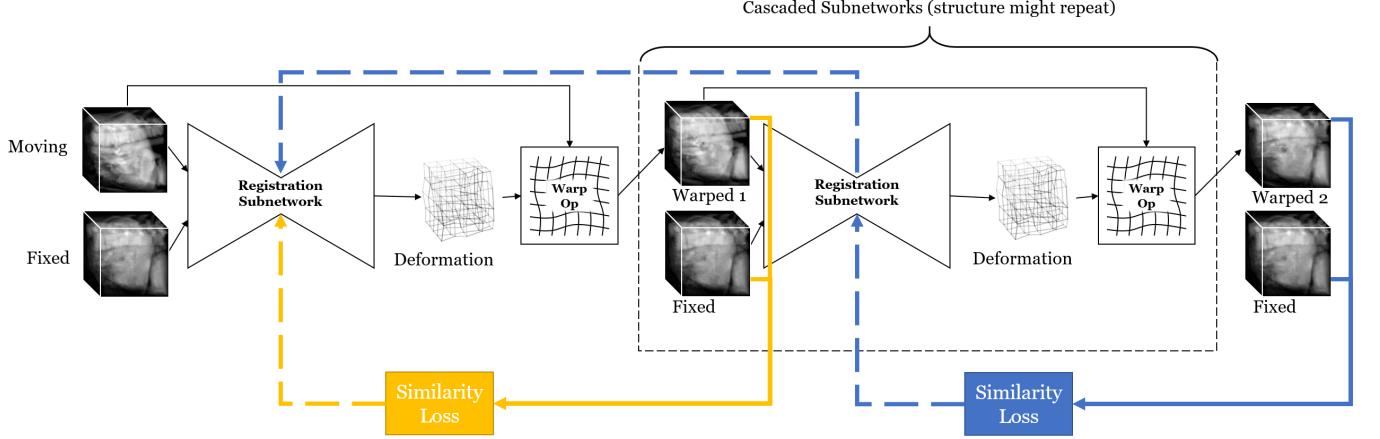


Figure 2: Illustration of an overall structure of Volume Tweening Network (VTN) and how gradients back-propagate. Every registration subnetwork is responsible for finding the deformation field between the fixed image and the current moving image. The moving image is repeatedly warped according to the deformation field and fed into the next level of cascaded subnetworks. The current moving images are compared against the fixed image for a similarity loss function to guide training. There is also regularization loss, but not drawn for the sake of cleanliness. The number of cascaded subnetworks may vary; only two are illustrated here. In the figure, yellow (lighter) indicates the similarity loss between the first warped image and the fixed image, and blue (darker) that between the second warped image and the fixed image. Solid bold arrows indicate how the loss is computed, and dashed bold arrows indicate how gradients back-propagate. Note that the second loss will propagate gradients to the first subnetwork as a consequence of the first warped image being a differentiable function of the first subnetwork.

generated using computer graphics algorithms and the ground-truth flow. However, realistic medical images are hard to generate, which leaves us with the only option of using medical images captured by sensors. The ground-truth flow between medical images is very difficult to obtain, which impedes the employment of supervised learning-based methods.

Spatial Transformer Networks (STN) [11] is a component in neural networks that spatially transforms feature maps to ease back-end tasks. It learns a localization net to produce an appropriate transformation to “straighten” the input image. The localization net is learnt without supervision, though back-end task might be supervised. Given a sampling grid and a transformation, STN applies the warping operation and outputs the warped image for further consumption by deeper networks. The warping operation deals with off-grid points by multi-linear interpolation, hence is differentiable and can back-propagate gradients.

Inspired by FlowNet and STN, we present a network structure (see Figure 2), called *Volume Tweening Network* (VTN), enabling unsupervised training of end-to-end CNNs that perform voxel-level 3D medical image registration. Training does not require ground truth as FlowNet (2.0) does. The moving image is registered and warped, and the warped image is compared against the fixed image to form a similarity loss. There is a rich body of research into similarity losses [12], [13], [14]. The model is trained to minimize a combination of regularization loss and similarity loss. As the method is unsupervised, the performance potentially increases as it is trained with more unlabeled data. The network consists of several cascaded subnetworks, the number of which might vary, and each subnetwork is responsible for producing a transform that aligns the fixed image and the moving one. Deeper layers register moving images warped according to the output of previous layers with the initial fixed image. The final prediction is the composition of all intermediate flows. It turns out that network cascading significantly improves the

performance in the presence of large displacement between the input images. While the idea of cascading subnetworks is found in FlowNet 2.0, our approach does not include as much artificial intervention of network structures as FlowNet 2.0 does. Instead, we employ a natural dichotomy in a subnetwork structure consisting of *affine* and *deformable* registration processes, which is also found in traditional methods, including ANTs (affine and SyN for deformable) [4] and Elastix (affine and B-spline for deformable) [5]. Besides these structural innovations, we also introduce the invertibility loss, similar to left-right disparity consistency loss in [15], to 3D medical image registration. Compared with traditional optimization-based algorithms, ours is 880x faster (or 3.3x faster without GPU acceleration) and achieves state-of-the-art registration accuracy.

To summarize, we present a new unsupervised end-to-end learning system using convolutional neural networks for deformation field prediction between 3D medical images. In this framework, we develop 3 technical components: (1) We cascade the registration subnetworks, which improves performance for registering largely displaced images without much slow-down; (2) We integrate affine registration into our network, which proves to be effective and faster than using a separate tool; (3) We incorporate an additional invertibility loss into the training process, which improves registration performance. The contributions of this work are closely related. An unsupervised approach is very suitable for this problem, as the images are abundant and the ground truth is costly to acquire. The use of the warping operation is crucial to our work, providing the backbone of unsupervision, network cascading and invertibility loss. Network cascading further allows us to plug in different subnetworks, and in this case, the affine registration subnetwork and the deformable ones, enabling us to adapt the natural structure of multiple stages in image registration. The efficient implementation of our algorithm gives a satisfactory speed.

II. RELATED WORKS

A. Directly Related Works

Several existing works that are directly related to ours are discussed below.

FlowNet [2] is an end-to-end FCN [10] that predicts optical flow between realistic images. The network has an encoder-decoder structure that extracts features and predicts optical flow at progressively refined scales, in which skip connections are added to combine high-level and low-level features. FlowNet is trained in a supervised manner with Flying Chairs, a synthetic dataset with ground-truth flow. In contrast, our method is trained without any ground-truth flow, nor does it require synthesizing training data.

FlowNet 2.0 [9], the successor of FlowNet, cascades several carefully crafted versions of FlowNet into one large network. The approach has proven helpful in improving accuracy of estimation. All of the subnetworks in FlowNet 2.0 perform deformable registration and each of them is carefully hand-crafted. Handcrafting the architecture makes the method prone to being task-specific. In contrast, we cascade two types of subnetworks, one performing affine registration, the other dense deformable, and we do not tweak the structure of our subnetworks.

Spatial Transformer Network (STN) [11] is a neural network that performs class-wise alignment. Spatial transformation parameters can be learned by training STN to loss functions without supervision. STN contains a differentiable layer that warps feature maps. One can reconstruct the fixed image by warping the moving one with the flow. STN itself does not register images. We employ the warping operation in STN to enable unsupervised training as well as network cascading. Figure 3 compares STN and a subnetwork (unit of cascading). The localization net is comparable to the convolutional part of our network, which produces the parameters (the flow field). Our grid generator simply skews the lattice on which input images are defined according to the field. The sampler, also known as the warp operation, warps the moving image into the warped image. Both nets are unsupervised. A key difference is that our subnetwork takes two images as input whereas STN takes only one. This difference is not only in design but also philosophical, and adapts to the tasks for each network. Our subnetwork takes one image (the fixed) as the reference image, because the task is to align the other image (the moving) to it. The training is guided by the alignment between the warped image and the fixed image. In STN’s settings, there is no such reference hence only one input image, and the task is to adjust the image to ease whatever task the back-end is to tackle. Its training is guided by the back-end task, which forces it to improve its warping strategy if the back-end network cannot solve the task well because working with the warped image is too hard for it. Philosophically, STN implicitly learns a reference “atlas” for the task at hand.

In an earlier work towards fast and accurate medical image registration by Shan et al. [16], an unsupervised end-to-end learning-based method for deformable medical image registration is proposed. The method in [16] registers 2D images. It is evaluated by registering corresponding sections

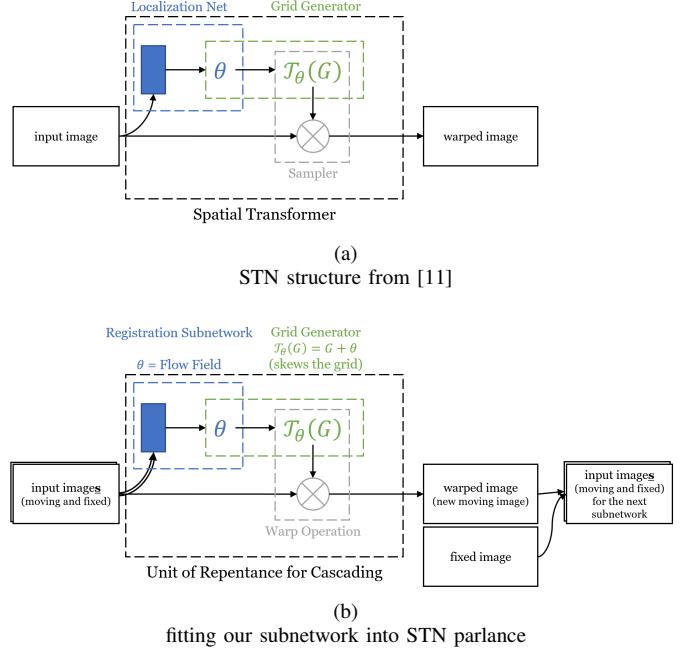


Figure 3: Comparison between STN and a subnetwork in our model. Comparable structures are similarly annotated. However, there is crucial difference between the two, as explained in the article.

of MR brain images and CT liver scans. In contrast, our 3D algorithm directly performs 3D registration, fully exploiting information available from the 3D images. Moreover, we cascade subnetworks to improve the performance. With the help of cascaded subnetworks, affine registration is integrated into our network, which is done out-of-band in [16].

VoxelMorph, proposed by a concurrent work by Balakrishnan et al. [17], is an unsupervised learning-based method for 3D medical image registration. VoxelMorph contains an encoder-decoder structure, uses warping operation to produce warped moving images and is trained to minimize the dissimilarity between the warped image and the fixed image. It is noticeable that both methods use the warp operation to train networks without supervision. This paper exploits the operation in a trio (namely enabling unsupervised training, enabling cascading, and implementing invertibility loss, which will be detailed later). Their method does not consider affine registration and assumes the input images are already affinely aligned, whereas ours embeds the process of affine registration as an integrated part of the network. Furthermore, their algorithm does not work well when large displacement between the images is present, which is common for liver CT scans. Finally, VoxelMorph is designed to take any two images as input, but [17] only evaluates it in the atlas-based registration scenario. Consider a clinical scenario where the brain of a patient is captured before and after an operation. It would be better, in terms of accuracy and convenience, to register these two images, instead of registering both to an atlas. In this paper, we present a more sophisticated and robust design that works well in the presence or absence of large displacement, and we evaluate the methods for general registration among images.

B. Traditional Algorithms

Over recent decades, many traditional algorithms have been developed and studied to register medical images [18], [19], [4], [5]. Most of them define a hypothesis space of possible transforms, often described by a set of parameters, and a metric of the quality of such a transform, and then find the optimum by iteratively updating the parameters.

Selecting an informative metric is crucial to registration quality. The metrics, also called loss functions, often consist of two parts, one measuring the level of correspondence between the input images implied by the transform, the other regularizing the transform itself. Examples of the former part include photo-metric difference, correlation coefficient and mutual information [13], [14] among others [12]. Some of these measures, notably mutual information, require binning or quantizing, which makes gradients vanish thus continuous optimization techniques such as gradient descent inapplicable.

The transformation space can be either parametric or non-parametric. Affine transforms can be described by only a few real numbers, whereas a free-form dense deformable field specifies the displacement for each grid point. Though the latter contains all possible transforms, it is common to apply a multi-stage approach to the problem. For example, ANTs [4] registers the input images with a rigid transform, then a general affine transform, and finally a deformable transform modeled by SyN. In this paper, we also have components for affine/deformable registration, which are modeled by neural networks.

Traditional methods have achieved good performance on several datasets, and are state-of-the-art, but their registration speed is barely practical for clinical applications. These methods do not exploit the patterns that exist in the registration task. In contrast, learning-based methods are generally faster. Computationally, they do not need to iterate over and over before producing the result. Conceptually, they learn those patterns represented by the parameters of the underlying learning model. The trained model is an efficient replacement of the optimization procedure.

C. Supervised Learning Methods

Lee et al. [20] employ Support Vector Machines (SVM) to learn the similarity measure for multi-modal image registration for brain scans. The training process of their algorithm requires pre-aligned image pairs thus the method is supervised. Sokooti et al. [21] develop a patch-based CNN to register chest CT scans and trains it with synthetic data. FlowNet [2], developed by Dosovitskiy et al., is an FCN [10] for optical flow prediction. The network is trained with synthetic data and estimates pixel-level correspondence.

While supervised learning methods achieve good performance, either abundant ground-truth alignment must be available, or synthetic data are used. Generation of synthetic data has to be carefully designed so that the generated data resemble the real ones.

D. Unsupervised Learning Methods

Obtaining the ground truth for registration between 3D medical images is very difficult. To work around this, unsupervised methods are to help.

Both inspired by FlowNet [2], Ren et al. [22] and Yu et al. [23] use STN to warp images according to the optical flow produced by CNN. They use Charbonnier penalty of fixed and warped images to guide the optimization of CNN, eliminating the need for ground-truth flow. Targeting another task, single view depth estimation, Garg et al. [24] develop an FCN that predicts depth from one photo, and use the depth field to reconstruct an image from another perspective, which is then compared against the other photo. The reconstruction loss is used to train CNN without annotated depth data. The method gains a boost in performance with a left-right consistency check [15].

III. METHOD

A. Problem Formulation

The input of an image registration problem consists of two images $I_{1,2}$, both of which are functions $\Omega \rightarrow \mathbb{R}^c$, where Ω is a region of \mathbb{R}^n and c denotes the number of channels. Since this work focuses on 3D medical image registration, we confine ourselves to the case where Ω is a cuboid and $c = 1$ (grayscale image). Specifically, this means $\Omega \subseteq \mathbb{R}^3$ and each image is a function $\Omega \rightarrow \mathbb{R}$. The objective of image registration is to find a displacement field (or flow field) $f_{12} : \Omega \rightarrow \mathbb{R}^3$ so that

$$I_1(x) \approx I_2(x + f(x)), \quad (1)$$

where the precise meaning of “ \approx ” depends on specific application. The field f_{12} is called the flow from I_1 to I_2 since it tells where each voxel in I_1 is in I_2 . We define $\text{warp}(I_2, f)$ as the image I_2 warped according to f , i.e., $\text{warp}(I_2, f)(x) = I_2(x + f(x))$. The above objective can be rephrased as finding f maximizing the similarity between I_1 and $\text{warp}(I_2, f)$.

The image I_1 is also called the fixed image, and I_2 the moving one. The term “moving” suggests that the image is transformed during the registration process.

Consider warping an image twice, first with g_1 then with g_2 . What this procedure produces is

$$\begin{aligned} & \text{warp}(\text{warp}(I, g_1), g_2)(x) \\ &= \text{warp}(I, g_1)(x + g_2(x)) \\ &= I(x + g_2(x) + g_1(x + g_2(x))) \\ &= \text{warp}(I, g_2 + \text{warp}(g_1, g_2))(x). \end{aligned} \quad (2)$$

This motivates the definition of the *composition* of two flows. If we define the composition of the flow fields g_1, g_2 to be

$$g_1 * g_2 = g_2 + \text{warp}(g_1, g_2), \quad (3)$$

Equation (2) can be restated as

$$\text{warp}(\text{warp}(I, g_1), g_2) = \text{warp}(I, g_1 * g_2). \quad (4)$$

It is noticeable that the warp operation in the above formulation should be further specified in practice. Real images as well as flow fields are only defined on lattice points. We continue

them onto the enclosing cuboid by trilinear interpolation as done in [25]. Furthermore, we deal with out-of-bound indices by nearest-point interpolation. That is, to evaluate a function defined on lattice points at any point x , we first move x to the nearest in the enclosing cuboid of those lattice points, then interpolate the value from the 8 nearest lattice points.

B. Unsupervised End-to-End Registration Network

Our network, called *Volume Tweening Network* (VTN), consists of several cascaded registration subnetworks, after each of which the moving image is warped. The unsupervised training of network parameters is guided by the dissimilarity between the fixed image and each of the warped images, with the regularization losses on the flows predicted by the subnetworks.

The warping operation, also known as the sampler in STN [25], is differentiable due to the trilinear interpolation. The warping operation back-propagates gradients to both the input image and the input flow field, which is critical for training our cascaded networks. These subnetworks are trained to work cooperatively in a way that the moving image is successively and gradually aligned.

For better performance, it is common to apply an initial rigid transformation as a global alignment before predicting the dense flow field. Instead of prepending a time-consuming preprocessing stage with a tool like ANTs [4] as what is done in VoxelMorph [17], we integrate this procedure as a top-level subnetwork. The integrated affine registration subnetwork not only works in negligible running time, but also outperforms the traditional affine stage.

C. Loss Functions

To train our model in an unsupervised manner, we measure the (dis)similarity between the moving images warped by the spatial transformer and the fixed image. There is a rich body of research in similarity metrics suitable for medical image registration [26], [12]. Furthermore, regularization losses are introduced to prevent the flow fields from being unrealistic or overfitting. In the following paragraphs, Ω denotes the cuboid (or grid) on which the input images are defined. We will introduce the loss functions that we use in the experiments. Extended discussion of other possible loss functions is available in the appendices.

a) Correlation Coefficient: The covariance between I_1 and I_2 is defined as

$$\text{Cov}[I_1, I_2] = \frac{1}{|\Omega|} \sum_{x \in \Omega} I_1(x) I_2(x) - \frac{1}{|\Omega|^2} \sum_{x \in \Omega} I_1(x) \sum_{y \in \Omega} I_2(y), \quad (5)$$

and their correlation coefficient is defined as

$$\text{CorrCoef}[I_1, I_2] = \frac{\text{Cov}[I_1, I_2]}{\sqrt{\text{Cov}[I_1, I_1] \text{Cov}[I_2, I_2]}}. \quad (6)$$

The images are regarded as random variables whose sample space is the points on which voxel values are available. The range of correlation coefficient is $[-1, 1]$, it measures how much the two images are linear related, and attains ± 1 if and

only if the two are linear function of each other. Applying a non-degenerate linear function to any of the images does not change their correlation coefficient, therefore, this measure is more robust than L_2 loss. For real-world images, the correlation coefficient should be non-negative (unless one of the images is a negative film). The correlation coefficient loss is defined as

$$L_{\text{Corr}}(I_1, I_2) = 1 - \text{CorrCoef}[I_1, I_2]. \quad (7)$$

b) Orthogonality Loss: For the specific task discussed in this paper (medical image registration), it is usually the case that the input images need only a small scaling and a rotation before they are affinely aligned. We would like to penalize the network for producing overly non-rigid transform. To this end, we introduce a loss on the non-orthogonality of $I + A$, where A denotes the transform matrix produced by the affine registration network (see Section IV-B for more details). Let $\lambda_{1,2,3}$ be the singular values of $I + A$, the orthogonality loss is

$$L_{\text{ortho}} = -6 + \sum_{i=1}^3 (\lambda_i^2 + \lambda_i^{-2}). \quad (8)$$

The more deviant $I + A$ is from being an orthogonal matrix, the larger its orthogonality loss. If $I + A$ is orthogonal, the value will be zero.

c) Determinant Loss: We assume images are taken with the same chirality, therefore, an affine transform involving reflection is not allowed. This imposes the requirement that $\det(I + A) > 0$. Together with the orthogonality requirement, we set the determinant loss to be

$$L_{\text{det}} = (-1 + \det(I + A))^2. \quad (9)$$

d) Total Variation Loss (Smooth Term): For a dense flow field, we regularize it with the following loss that discourages discontinuity:

$$L_{\text{TV}} = \frac{1}{3|\Omega|} \sum_x \sum_{i=1}^3 (f(x + e_i) - f(x))^2, \quad (10)$$

where $e_{1,2,3}$ form the natural basis of \mathbb{R}^3 . This is also known as the L2 smooth term.

IV. NETWORK ARCHITECTURE

A. Cascading

Each subnetwork is responsible for aligning the fixed image and the *current* moving image. Following each subnetwork, the moving image is warped with the predicted flow, and the warped image is fed into the next cascaded subnetwork. The flow fields are composed to produce the final estimation. Figure 2 illustrates how the networks are cascaded, how the images are transformed and how each part contributes to the loss. All layers being differentiable, the gradient will back-propagate so that the subnetworks can be trained.

It might be tempting to compare our scheme with that of FlowNet 2.0 [9]. FlowNet 2.0 stacks subnetworks in a different way than our method. It performs two separate lines of flow estimations (large/small displacement flows) and fuses them into the final estimation. Each of its intermediate subnetworks

has inputs of not only the warped moving image and the fixed image, but also the initial moving image, the current flow, and the brightness error. Its subnetworks are carefully crafted, having similar yet different structures and expected to solve specific problems (e.g., large/small displacement) in flow estimation. In contrast, our method does not involve two separate lines of registration, i.e., each subnetwork works on the fixed image and the warped moving image produced by the previous one, and intermediate subnetworks do not get more input than the initial one. We do not interfere much with the structures of subnetworks. Despite the initial affine subnetwork, we do not assign specific tasks to the remaining subnetworks since they share the same structure.

B. Affine and Dense Deformable Subnetworks

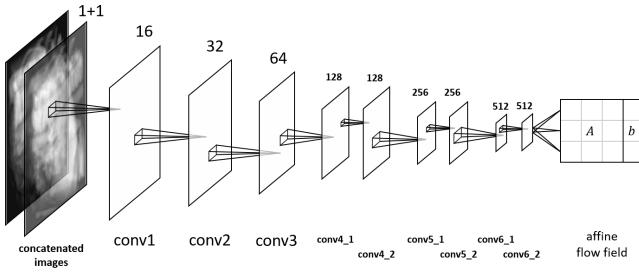


Figure 4: Affine registration subnetwork. The number of channels is annotated above the layer. A smaller canvas means lower spatial resolution.

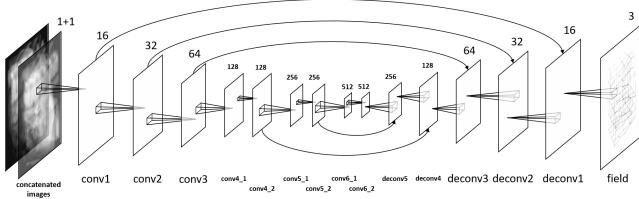


Figure 5: Dense deformable registration subnetwork. Number of channels is annotated above the layer. Curved arrows represent skip paths (layers connected by an arrow are concatenated before transposed convolution). Smaller canvas means lower spatial resolution.

The affine registration subnetwork aims to align the input image with an affine transform. It is only used as our first subnetwork. Its structure, as illustrated in Figure 4, does *not* resemble an auto-encoder suggested by the sandglass shape in Figure 2. This type of subnetwork progressively downsamples the input by doing strided 3D convolution, after which a fully-connected layer is applied to produce 12 numeric parameters, which represents a 3×3 transform matrix A and a 3-dimensional displacement vector b . As a common practice, the number of channels doubles as the length of resolution halves. The flow field produced by this subnetwork is defined as

$$f(x) = Ax + b. \quad (11)$$

The dense deformable registration subnetwork is used as all subsequent subnetworks, each of which refines the registration based on the output of the subnetwork preceding it. Its structure, illustrated in Figure 5, is an auto-encoder and similar

to FlowNet [2]. We use strided 3D convolution to progressively downsample the image, and then use deconvolution (transposed convolution) [10] to recover spatial resolution. As suggested in [2], skip paths connecting the deconvolution layer to the convolution layer with the same spatial resolution are added to help localizing estimation, which results in a structure similar to U-Net [27]. The subnetwork will output the dense flow field, a volume feature map with 3 channels (x, y, z displacements) of the same size as the input.

C. Invertibility

Given two images $I_{1,2}$, going from a voxel in I_1 to the corresponding voxel in I_2 then back to I_1 should give zero displacement. Otherwise stated, the registration should be *round-trip*. In Figure 6, we demonstrate the possible situations. The pair of solid arrows exemplifies round-trip registration, whereas the pair of dashed arrows exemplifies non-round-trip registration. If we have computed two flow fields (back and forth), f_{12} and f_{21} , the composed fields exhibit the round-trip behavior of the registration, as illustrated by the magenta straight arrow in Figure 6. Ideally, round-trip registration should satisfy the equations $f_{12} * f_{21} = f_{21} * f_{12} = 0$. We capture the round-tripness for a pair of images with the invertibility loss, namely

$$L_{\text{inv}} = \|f_{12} * f_{21}\|_2^2 + \|f_{21} * f_{12}\|_2^2. \quad (12)$$

The larger the invertibility loss, the less round-trip the registration. For perfectly round-trip registration, the invertibility loss is zero. We come up with, formulate, and implement the invertibility loss independently of [15]. We use L2 invertibility loss whereas [15] uses L1 left-right disparity consistency loss, which is just a matter of choice. We are the first to incorporate the invertibility loss into 3D images to boost performance on medical image tasks.

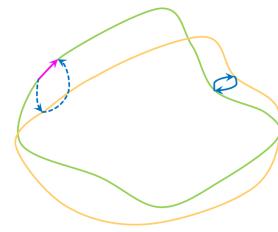


Figure 6: Illustration of how invertibility loss enforces round-trip registration. Green (darker) and yellow (lighter) curves represent two images. Curved arrows: flow fields. Solid curved arrows: flow inversion. Dashed arrows: failure of flow inversion. Straight arrow (magenta, very dark): example vector in the composed flow green \rightarrow yellow \rightarrow green, which is non-zero because the voxel fails to trip back. Best viewed in color.

V. EXPERIMENT

We evaluate our algorithm with extensive experiments on liver CT datasets and brain MRI datasets. We compare our algorithm against state-of-the-art traditional registration algorithms including ANTs [4] and Elastix [5], as well as VoxelMorph [17]. Our algorithm achieves state-of-the-art performance while being much faster. Our experiments prove that the performance of our unsupervised method is improved as more unlabeled data are used in training. We show that cascading

subnetworks significantly improves the performance, and that integrating affine registration into the method is effective.

We evaluate the performance of algorithms with the following metrics:

- **Seg. IoU** is the Jaccard coefficient between the warped liver segmentation and the ground truth. We warp the segmentation of the moving image by the predicted deformable field, and compute the Jaccard coefficient of the warped segmentation with the ground-truth segmentation of the fixed image. “IoU” means “intersection over union”, i.e., $\frac{|A \cap B|}{|A \cup B|}$, where A, B are the set of voxels the organ consists of.
- **Lm. Dist.** is the average distance between warped landmarks (points of anatomical interest) and the ground truth.
- **Time** is the average time taken for each pair of images to be registered. Some methods are implemented with GPU acceleration, therefore there are two versions of this metric (with or without GPU acceleration).

Our model is defined and trained using TensorFlow [28]. We accelerate training with nVIDIA TITAN Xp and CUDA 8.0. We use the Adam optimizer [29] with the default parameters in TensorFlow 1.4. The batch size is 8 pairs per batch. The initial learning rate is 10^{-4} and halves every epoch after the 4th epoch. Each epoch consists of 20000 batches and the number of epochs is 5. Performance evaluation uses the same GPU model. Traditional methods work on CPU. We also test neural-network-based methods with GPU acceleration disabled for a fairer comparison of speed. The CPU model used is Intel® Xeon® CPU E5-2690 v4 @ 2.60GHz (14 Cores).

A. Experiments on Liver Datasets

1) *Settings*: The input to the algorithms are liver CT scans of size 128^3 . Affine subnetworks downsample the images to 4^3 before applying the fully-connected layer. Dense deformable subnetworks downsample the images to 2^3 before doing transposed deconvolution.

We cascade *up to 4* registration subnetworks. The reason we need to cascade multiple subnetworks is that large displacement is very common among CT liver scans and that with more subnetworks, images with large displacement can be progressively aligned. Among these networks, the one with one affine registration subnetwork and three dense deformable registration networks (referred to as “ADDD”) is used to be trained with different amount of data and compared with other algorithms. We use the correlation coefficient as our similarity loss, the orthogonality loss and the determinant loss as regularization losses for the affine subnetwork, and the total variation loss as that for dense deformable subnetworks. The ratio of losses for “ADDD” is listed in Table I. The performance is not very sensitive to the choice of hyperparameters. We suggest that each of the dense deformable subnetworks can automatically learn how to progressively align the images, and that only the final subnetwork and the affine subnetwork need to be trained with similarity loss.

2) *Datasets*: We have three datasets available:

Subnetwork	Loss	Relative Ratio
Affine	Similarity	1
	Determinant	0.1
	Orthogonality	0.1
Dense 1	Similarity	0
	Total variation	1
Dense 2	Similarity	0.05
	Total variation	1
Dense 3	Similarity	1
	Total variation	1

Table I: Ratio of loss functions.

- **LITS** [30] consists of 130 volumes. LITS comes with segmentation of liver, but we do not use such information. This dataset is used for training.
- **BFH** is provided by Beijing Friendship Hospital and consists of 92 volumes. This dataset is used for training.
- **MICCAI** (MICCAI’07) [31] consists of 20 volumes with liver segmentation ground truth. We choose 4 points of anatomical interest as the landmarks¹ and ask 3 expert doctors to annotate them, taking their average as the ground truth. This dataset is used as the test data.

We crop raw liver CT scans to a volume of size 128^3 around the liver, permute (and reflect, if necessary) the axes so that the orientations of the images agree with each other, and normalize them by adjusting exposure so that the histograms of the images match each other. The preprocessing is necessary as the images come from different sources.

3) *Comparison among Methods*: In Table II, “ADDD” is our model detailed in Section V-A1, and “ADDD + inv” is that model trained with additional term of invertibility loss in the central area (the beginning and the ending quaters of each side are removed) with relative weight 10^{-3} . Learning-based methods (VTN and VoxelMorph) are trained on LITS and BFH datasets. All methods are evaluated on the MICCAI dataset. To prove the effectiveness of our unsupervised method, we also train “ADDD” supervised (the row “supervised”), where the output of ANTs is used as the ground truth (using end-point error [2] plus regularization term as the loss function).

Method	Seg. IoU	Lm. Dist.	Time	Time (w/o GPU)
ANTs [4]	0.8124	11.93	N/A	748 s
Elastix [5]	0.8365	12.36	N/A	115 s
VoxelMorph-2 [17]	0.6796	18.10	0.20 s	17 s
VTN ADDD (ours)	0.8868	12.04	0.13 s	26 s
VTN ADDD + inv (ours)	0.8882	11.42	0.13 s	26 s
supervised	0.7680	13.38	0.13 s	26 s

Table II: Comparison among traditional methods, VoxelMorph and our VTN (liver).

VoxelMorph-2 is trained (using code released by [17]) with a batch size of 4 and an iteration count of 5000. Its performance with a batch size of 8 or 16 is worse so a batch size of 4 is used. After 5000 iterations, the model begins to overfit. The reason that it does not perform well on liver

¹The landmarks: (L1) the top point of hepatic portal; (L2) the intersection of the superior and anteroir branches of the right lobe; (L3) the intersection of the superior and inferior branches of the right lobe; and (L4) the intersection of the medial and inferior branches of the left lobe.

datasets might be that it is designed for brain registration thus cannot handle large displacement.

The results in Table II show the vast speed-up of learning-based methods against optimization-based methods. Our methods surpass state-of-the-art registration algorithms in terms of Segmentation IoU and Landmark Distance.

4) Performance with Different Amount of Data: In Table III, the “ADDD” network (see Section V-A1) is trained with different amount of data. The result demonstrates that training with more *unlabeled* data improves the performance. Since we do not need any annotations during the training phase, there are abundant clinical data ready for use.

Training Dataset for VTN ADDD	Seg. IoU	Lm. Dist.
LITS	0.8712	13.11
LITS + BFH	0.8868	12.04

Table III: Comparison of performance of VTN ADDD with different amount of unlabeled training data (liver).

5) Network Cascading: In Table IV, all networks are trained on LITS + BFH. The models whose name does not include “A” have the affine registration subnetwork removed, and the number of “D”的 is the number of dense deformable registration subnetworks. (See Section V-A1 for “ADDD”).

Network	Seg. IoU	Lm. Dist.	Time	Time (w/o GPU)
D	0.8119	14.44	0.08 s	10 s
DD	0.8556	12.97	0.10 s	20 s
DDD	0.8709	12.49	0.12 s	28 s
AD	0.8323	13.20	0.09 s	9 s
ADD	0.8703	12.28	0.11 s	19 s
ADDD	0.8868	12.04	0.13 s	26 s

Table IV: Comparison of performance with different number of cascaded subnetworks (liver).

B. Experiments on Brain Datasets

a) Settings: The input to the algorithms are brain MR images of size 128^3 . For some experiments, the selection of which will be detailed later, the brain scans are preprocessed to be aligned to a specific atlas from LONI [32]. We use ANTs for this purpose. There are two reasons we align the brain scans with ANTs. One is that VoxelMorph [17] requires the input to be affinely registered. The other is that we will compare the performance between “ANTs + deformable registration subnetworks” and “affine registration subnetwork + deformable registration subnetworks”, i.e., to compare the effectiveness of integrating our affine registration subnetwork in place of ANTs.

The following methods will have ANTs-affine-aligned images as input: VoxelMorph methods and VTN without “A” (i.e., “D”, “DD” and “DDD”). A more precise naming is “ANTs + VoxelMorph” and “ANTs + VTN”. The following methods will *not* have ANTs-affine-aligned images as input: ANTs, Elastix, VTN with “A” (i.e., “AD”, “ADD” and “ADDD”). Those methods have affine registration integrated. The comparison inside the former group focuses on dense deformable registration performance, that inside the latter

group on overall performance, and that among the two groups benchmarks affine registration subnetwork versus ANTs affine registration in the context of a complete registration pipeline.

We will show 3 sets of comparisons similar to those for liver datasets. In the tables listed in later paragraphs, the time (with or without GPU) does not include the preprocessing with ANTs even if it is used. Preprocessing an image with ANTs costs 73.94 seconds on average. We will mention this fact again when such emphasis is needed.

Care should be taken when evaluating methods with ANTs affine alignment. For the data to be comparable with those with affine registration integrated, the fixed image should be equivalent. Methods with ANTs affine alignment have both moving and fixed images aligned to an atlas. Those with integrated affine registration never move the fixed image. The affine transform produced by ANTs might not be orthogonal, which is the source of unfair comparison. If the affine transform is shrinking, methods with ANTs affine alignment gain advantage. Otherwise, methods with integrated affine alignment do.

One measure, Segmentation IoU, is not affected, because the volumes of all objects get multiplied by the determinant of the affine transform and the evaluation measure is homogeneous. For Landmark Distance, we perform the inverse of the linear part of the affine transform (which aligns the fixed image to the atlas) to the difference vector between warped landmark and landmark in the (aligned) fixed image, so that the length goes back to the coordinate defined by the original fixed image. This way, we minimize loss of precision to prevent unfairly underevaluating methods with ANTs affine alignment. Speaking of the actual data, the affine transformations produced by ANTs are slightly shrinking. Our correction restores a fair comparison among all methods.

b) Datasets: We use volumes from the following datasets for training:

- ADNI [33] (67 volumes);
- ABIDE-1 [34] (318 volumes): part of data from ABIDE;
- ABIDE-2 [34] (1101 volumes): the rest from ABIDE;
- ADHD [35] (973 volumes).

We acquire the second part of ABIDE after a while when the first part was downloaded and processed, thus the split. This only helps us to understand how performance improves as more data are used for training. For comparison among different methods, it is always the case that all the data mentioned above are used for training.

Raw MR scans are cropped to 128^3 around the brain. Axes are permuted if necessary. The scans are normalized based on the histograms of their foreground color distribution, which might vary because they are captured on different sites.

For evaluation, we use 20 volumes from the LONI Probabilistic Brain Atlas (LPBA40) [32]. LONI consists of 40 volumes, 20 of which have tilted head positions and are

discarded. For the remaining 20 volumes, 18 landmarks² are annotated by 3 experts and the average are taken as the ground truth.

c) *Comparison Among Methods:* In Table V, we compare different methods on brain datasets. All neural networks are trained on all available training data, i.e., ADNI, ABIDE and ADHD. In the table, “supervised” is our “ADD” model supervised with ANTs as the ground truth. Its loss is the endpoint error [2].

Method	Seg. IoU	Lm. Dist.	Time	Time (w/o GPU)
ANTs [4]	0.9387	2.85	N/A	764 s
Elastix [5]	0.9180	3.23	N/A	121 s
VoxelMorph-2* [17]	0.9268	2.84	0.19 s	14 s
VTN DD (ours)*	0.9305*	2.83*	0.09 s	19 s
VTN ADD (ours)	0.9270	2.62	0.10 s	17 s
VTN ADD + inv (ours)	0.9278	2.64	0.10 s	17 s
supervised	0.9060	2.94	0.10 s	19 s

Table V: Comparison among traditional methods, VoxelMorph [17] and our algorithm on brain datasets. Bold: best among all methods. Star: best among methods with ANTs affine pre-alignment.

Methods with “*” use ANTs for affine pre-alignment. The preprocessing time (about 74 seconds) is not included in the table.

Among these methods, our “ADD” achieves the lowest Landmark Distance with a competitive speed. If we compare “ADD” with “DD”, we find that the integration of affine registration subnetwork significantly improves the Landmark Distance, compared to using ANTs for out-of-band affine alignment.

d) *Performance with Different Amount of Data:* In Table VI, we summarize the performance of “DD” (with ANTs affine alignment) trained on different amount of unlabeled data. As more data are used to train the network, its performance in terms of Landmark Distance consistently increases.

Training Dataset for VTN DD	Seg. IoU	Lm. Dist.
ADNI + ABIDE-1	0.9299	2.90
ADNI + ABIDE-1 + ABIDE-2	0.9312	2.86
ADNI + ABIDE-1 + ABIDE-2 + ADHD	0.9305	2.83

Table VI: Comparison of performance of VTN DD (with ANTs affine alignment) with different amount of unlabeled training data (brain).

e) *Network Cascading and Integration of Affine Registration:* Table VII compares the performances of differently cascaded networks. The networks without “A” have ANTs-affine-aligned images as input, whereas the networks with “A” do not.

As one would expect, the performance in each group improves as the model gets more levels of cascaded subnetworks. While the methods with ANTs affine alignment have higher

²The landmarks: (L1) right lateral ventricle posterior; (L2) left lateral ventricle posterior; (L3) anterior commissure corresponding to the midpoint of decussation of the anterior commissure on the coronal AC plane; (L4) right lateral ventricle superior; (L5) right lateral ventricle inferior; (L6) left lateral ventricle superior; (L7) left lateral ventricle inferior; (L8) middle of lateral ventricle; (L9) posterior commissure corresponding to the midpoint of decussation; (L10) right lateral ventricle superior; (L11) left lateral ventricle superior; (L12) middle of lateral ventricle; (L13) corpus callosum inferior; (L14) corpus callosum superior; (L15) corpus callosum anterior; (L16) corpus callosum posterior tip of genu corresponding to the location of the most posterior point of corpus callosum posterior tip of genu on the midsagittal planes; (L17) corpus callosum fornix junction; and (L18) pineal body.

Method	Seg. IoU	Lm. Dist.	Time	Time (w/o GPU)
D*	0.9241	2.91	0.07 s	10 s
DD*	0.9305	2.83*	0.09 s	19 s
DDD*	0.9320*	2.85	0.10 s	28 s
AD	0.9214	2.73	0.08 s	9 s
ADD	0.9270	2.62	0.10 s	17 s
ADDD	0.9286	2.63	0.11 s	26 s

Table VII: Comparison of performance with different number of cascaded subnetworks (brain), and comparison between using ANTs as affine alignment and end-to-end network (integrated affine registration subnetwork). The first group (with “*”) uses ANTs to affinely align input images, whereas the second group does not. Bold: best among all methods. Star: best among methods with ANTs affine pre-alignment. Methods with “*” use ANTs for affine pre-alignment. The preprocessing time (about 74 seconds) is not included in the table.

Segmentation IoU, integrating affine registration subnetwork yields better Landmark Distance. Worth mentioning is that the better Segmentation IoU comes at the price of a rather slow preprocessing phase (74 seconds).

VI. CONCLUSION

In this paper, we present *Volume Tweening Network* (VTN), a new unsupervised end-to-end learning framework using convolutional neural networks for 3D medical image registration. The network is trained in an unsupervised manner without any ground-truth deformation. Experiments demonstrate that our method achieves state-of-the-art performance, and that it witnesses an 880x (or 3.3x without GPU acceleration) speed-up compared to traditional medical image registration methods. Our thorough experiments prove our contributions, each on its own being useful and forming a strong union when put together. Our networks can be cascaded. Cascading deformable subnetworks tackles the difficulty of registering images in the presence of large displacement. Network cascading also enables the integration of affine registration into the algorithm, resulting in a truly end-to-end method. The integration proves to be more effective than using out-of-band affine alignment. We also incorporate the invertibility loss into the training process, which further enhances the performance. Our methods can potentially be applied to various other medical image registration tasks.

ACKNOWLEDGMENT

The authors would like to thank Beijing Friendship Hospital for providing the “BFH” dataset as well as other data provider for making their data publicly available.

REFERENCES

- [1] F. P. M. Oliveira and J. M. R. S. Tavares, “Medical image registration: a review,” *Computer Methods in Biomechanics and Biomedical Engineering*, vol. 17, no. 2, pp. 73–93, 2014.
- [2] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. V. Der Smagt, D. Cremers, and T. Brox, “Flownet: Learning optical flow with convolutional networks,” *International Conference on Computer Vision*, pp. 2758–2766, 2015.
- [3] J. Modersitzki, *FAIR: Flexible Algorithms for Image Registration*. Philadelphia: SIAM, 2009.
- [4] B. B. Avants, N. Tustison, and G. Song, “Advanced normalization tools (ants).” *Insight j*, vol. 2, pp. 1–35, 2009.
- [5] S. Klein, M. Staring, K. Murphy, M. A. Viergever, and J. P. W. Pluim, “elastix: A toolbox for intensity-based medical image registration,” *IEEE Transactions on Medical Imaging*, vol. 29, no. 1, pp. 196–205, Jan 2010.

- [6] G. J. S. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. W. M. van der Laak, B. van Ginneken, and C. I. Sánchez, “A survey on deep learning in medical image analysis,” *CoRR*, vol. abs/1702.05747, 2017. [Online]. Available: <http://arxiv.org/abs/1702.05747>
- [7] S. Miao, Z. J. Wang, and R. Liao, “A cnn regression approach for real-time 2d/3d registration,” *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1352–1363, May 2016.
- [8] G. Wu, M. Kim, Q. Wang, Y. Gao, S. Liao, and D. Shen, “Unsupervised deep feature learning for deformable registration of mr brain images,” in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2013*, K. Mori, I. Sakuma, Y. Sato, C. Barillot, and N. Navab, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 649–656.
- [9] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, “Flownet 2.0: Evolution of optical flow estimation with deep networks,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. [Online]. Available: <http://lmb.informatik.uni-freiburg.de/Publications/2017/IMKDB17>
- [10] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3431–3440.
- [11] M. Jaderberg, K. Simonyan, A. Zisserman, and k. kavukcuoglu, “Spatial transformer networks,” in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 2017–2025. [Online]. Available: <http://papers.nips.cc/paper/5854-spatial-transformer-networks.pdf>
- [12] D. J. H. A. Melbourne, G. Ridgway, “Image similarity metrics in image registration,” pp. 7623 – 7623 – 10, 2010. [Online]. Available: <https://doi.org/10.1117/12.840389>
- [13] J. P. W. Pluim, J. B. A. Maintz, and M. A. Viergever, “Mutual-information-based registration of medical images: a survey,” *IEEE TRANSCATIONS ON MEDICAL IMAGING*, pp. 986–1004, 2003.
- [14] C. Studholme and D. L. G. Hill, “D.j.hawkes. an overlap invariant entropy measure of 3d medical image alignment,” *Pattern Recognition*, no. 32, 1999.
- [15] C. Godard, O. Mac Aodha, and G. J. Brostow, “Unsupervised monocular depth estimation with left-right consistency,” in *CVPR*, vol. 2, no. 6, 2017, p. 7.
- [16] S. Shan, X. Guo, W. Yan, E. I. Chang, Y. Fan, and Y. Xu, “Unsupervised end-to-end learning for deformable medical image registration,” *CoRR*, vol. abs/1711.08608, 2017. [Online]. Available: <http://arxiv.org/abs/1711.08608>
- [17] G. Balakrishnan, A. Zhao, M. R. Sabuncu, J. V. Guttag, and A. V. Dalca, “An unsupervised learning model for deformable medical image registration,” *CoRR*, vol. abs/1802.02604, 2018. [Online]. Available: <http://arxiv.org/abs/1802.02604>
- [18] G. Hermosillo, C. Chefd’Hotel, and O. Faugeras, “Variational methods for multimodal image matching,” *International Journal of Computer Vision*, vol. 50, no. 3, pp. 329–343, Dec 2002. [Online]. Available: <https://doi.org/10.1023/A:1020830525823>
- [19] X. Huang, N. Paragios, and D. N. Metaxas, “Shape registration in implicit spaces using information theory and free form deformations,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 8, pp. 1303–1318, Aug. 2006. [Online]. Available: <https://doi.org/10.1109/TPAMI.2006.171>
- [20] D. Lee, M. Hofmann, F. Steinke, Y. Altun, N. D. Cahill, and B. Scholkopf, “Learning similarity measure for multi-modal 3d image registration,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, June 2009, pp. 186–193.
- [21] H. Sokooti, B. de Vos, F. Berendsen, B. P. F. Lelieveldt, I. Isgum, and M. Staring, “Nonrigid image registration using multi-scale 3d convolutional neural networks,” in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2017*, M. Descoteaux, L. Maier-Hein, A. Franz, P. Jannin, D. L. Collins, and S. Duchesne, Eds. Cham: Springer International Publishing, 2017, pp. 232–239.
- [22] Z. Ren, J. Yan, B. Ni, B. Liu, X. Yang, and H. Zha, “Unsupervised deep learning for optical flow estimation.” in *AAAI*, 2017, pp. 1495–1501.
- [23] J. J. Yu, A. W. Harley, and K. G. Derpanis, “Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness,” in *Computer Vision – ECCV 2016 Workshops*, G. Hua and H. Jégou, Eds. Cham: Springer International Publishing, 2016, pp. 3–10.
- [24] R. Garg, V. K. B.G., G. Carneiro, and I. Reid, “Unsupervised cnn for single view depth estimation: Geometry to the rescue,” in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 740–756.
- [25] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, “Spatial transformer networks,” in *Advances in Neural Information Processing Systems*, 2015, pp. 2017–2025.
- [26] A. Sotiras, C. Davatzikos, and N. Paragios, “Deformable medical image registration: A survey,” *IEEE Transactions on Medical Imaging*, vol. 32, no. 7, pp. 1153–1190, July 2013.
- [27] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2015, pp. 234–241.
- [28] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, “TensorFlow: Large-scale machine learning on heterogeneous systems,” 2015, software available from tensorflow.org. [Online]. Available: <https://www.tensorflow.org/>
- [29] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: <http://arxiv.org/abs/1412.6980>
- [30] “Liver tumor segmentation challenge,” 2017. [Online]. Available: <http://lits-challenge.com/>
- [31] T. Heimann, B. van Ginneken, M. A. Styner, Y. Arzhaeva, V. Aurich, C. Bauer, A. Beck, C. Becker, R. Beichel, G. Bekes, F. Bello, G. Binnig, H. Bischof, A. Bornik, P. M. M. Cashman, Y. Chi, A. Cordova, B. M. Dawant, M. Fidrich, J. D. Furst, D. Furukawa, L. Grenacher, J. Hornegative, D. Kainmüller, R. I. Kitney, H. Kobatake, H. Lamecker, T. Lange, J. Lee, B. Lennon, R. Li, S. Li, H. P. Meinzer, G. Nemeth, D. S. Raicu, A. M. Rau, E. M. van Rikxoort, M. Rousson, L. Rusko, K. A. Saddi, G. Schmidt, D. Seghers, A. Shimizu, P. Slagmolen, E. Sorantin, G. Soza, R. Susomboon, J. M. Waite, A. Wimmer, and I. Wolf, “Comparison and evaluation of methods for liver segmentation from ct datasets,” *IEEE Transactions on Medical Imaging*, vol. 28, no. 8, pp. 1251–1265, Aug 2009.
- [32] D. W. Shattuck, M. Mirza, V. Adisetiyo, C. Hojatkashani, G. Salamon, K. L. Narr, R. A. Poldrack, R. M. Bilder, and A. W. Toga, “Construction of a 3d probabilistic atlas of human cortical structures,” *Neuroimage*, vol. 39, no. 3, pp. 1064–1080, 2008.
- [33] S. G. Mueller, M. W. Weiner, L. J. Thal, R. C. Petersen, C. R. Jack, W. Jagust, J. Q. Trojanowski, A. W. Toga, and L. Beckett, “Ways toward an early diagnosis in alzheimer’s disease: the alzheimer’s disease neuroimaging initiative (adni),” *Alzheimer’s & Dementia*, vol. 1, no. 1, pp. 55–66, 2005.
- [34] A. Di Martino, C.-G. Yan, Q. Li, E. Denio, F. X. Castellanos, K. Alaerts, J. S. Anderson, M. Assaf, S. Y. Bookheimer, M. Dapretto *et al.*, “The autism brain imaging data exchange: towards a large-scale evaluation of the intrinsic brain architecture in autism,” *Molecular psychiatry*, vol. 19, no. 6, p. 659, 2014.
- [35] P. Bellec, C. Chu, F. Chouinard-Decorte, Y. Benhajali, D. S. Margulies, and R. C. Craddock, “The neuro bureau adhd-200 preprocessed repository,” *NeuroImage*, vol. 144, pp. 275 – 286, 2017, data Sharing Part II. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S105381191630283X>
- [36] B. C. Ross, “Mutual information between discrete and continuous data sets,” *PLOS ONE*, vol. 9, no. 2, pp. 1–5, 02 2014. [Online]. Available: <https://doi.org/10.1371/journal.pone.0087357>
- [37] G. Valiant and P. Valiant, “Estimating the unseen: Improved estimators for entropy and other properties,” *J. ACM*, vol. 64, no. 6, pp. 37:1–37:41, Oct. 2017. [Online]. Available: <http://doi.acm.org/10.1145/3125643>

APPENDIX A NAMING OF VOLUME TWEENING NETWORK

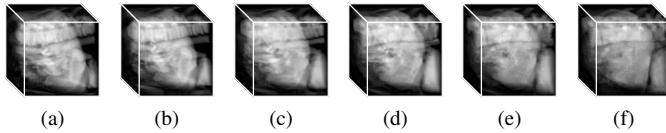


Figure 7: Examples of intermediate warped moving images by a cascaded network consisting of 4 subnetworks. (a) the moving image (a CT liver scan); (b) warped by the affine registration subnetwork; (c/d/e) warped by the first/second/third dense deformable registration subnetwork; (f) the fixed image (another CT liver scan). The images are realistic and are rendered with the same data as those in Figure 1.

In Figure 7, intermediate warped images as well as the fixed and moving images are projected and rendered, giving a straight-forward illustration of how the image is transformed step by step by a cascaded network. This explains the naming *Volume Tweening Network* – due to the cascaded nature of the network, the intermediate images look like frames in a shape tweening animation from the moving image to the fixed image.

APPENDIX B LOSS FUNCTIONS

This section lists some other loss functions that could be used for training the network.

A. L_2 Loss

L_2 loss (or mean square error) measures the difference between intensities of the images at each voxel, it is defined as

$$L_{\text{MSE}}(I_1, I_2) = \frac{1}{|\Omega|} \sum_{x \in \Omega} (I_1(x) - I_2(x))^2. \quad (13)$$

L_2 loss is straight-forward and easy to implement, but it is not very informative if the fixing and moving images have different imaging parameters. The disadvantage manifests itself especially when the images are inter-subject, or are imaged using devices from different manufacturers.

B. Mutual Information

Again we will regard the images as random variables. The entropy of a discrete random variable X is

$$H(X) = - \sum_x p_x(x) \log p_x(x), \quad (14)$$

and the mutual information of two discrete random variables X, Y is

$$I(X; Y) = H(X) + H(Y) - H(X, Y). \quad (15)$$

Mutual information of two images measures how much the two variables are related by *some* function. Presumably the intensities of corresponding anatomical structures in the fixed and moving images should be related via *some* function, which might be non-linear. Therefore, mutual information should be more resistant to non-linear intensity variation. Indeed, variants of mutual information have been the gold standard

among medical image registration metrics [19], [18], [14], [13]. The downside of this family of loss functions is that adapting them in neural networks is not completely trivial.

From the definition, computing mutual information reduces to computing entropies. Given a series of independent observations x_1, \dots, x_n to a random variable X , a naïve estimation of $H(X)$ is

$$H(X) \approx \frac{1}{n} \sum_{i=1}^n \log \frac{|\{j : x_j = x_i\}|}{n}. \quad (16)$$

Though Equation (16) systematically underestimates the entropy and various improved algorithms have been developed [36], [37], it is still the simplest estimator. However, there is one obstacle keeping it from being used in neural networks, “binning.” Traditionally, binning is required for two reasons: that the metric is more robust against random fluctuations in intensities, which will be effaced by binning; and that directly estimating entropy of continuous random variable from unbinned observations is generally more difficult than the binned, discrete alternative. However, the downside of binning is that the operation makes gradient vanish almost everywhere, effectively making the problem of maximizing mutual information a discrete optimization problem, disqualifying gradient descent and other gradient-based optimizers. To tackle the problem, we rewrite Equation (16) as

$$H(X) \approx \frac{1}{n} \sum_{i=1}^n \log \frac{\sum_{j=1}^n z(x_i - x_j)}{n}, \quad (17)$$

where $z(x)$ is 1 if $x = 0$, and 0 otherwise. Now, we are ready to replace z with an approximation, e.g., $z_\lambda(x) = e^{-\lambda|x|}$ for $\lambda > 0$. Furthermore, using this formula boosts the time of computation to the order of number of observations squared. Thus, we need to randomly sample a subset of locations to compute entropy.

APPENDIX C NOTES TO IMPLEMENTATION

A. TensorFlow Custom Operations

We choose TensorFlow [28] as our platform. Several operations need to be implemented as custom operations to reduce memory footprint. These include functionalities to warp images, compute moments of random variables, approximate entropy, etc.

B. Warp and Domain Shrinking

Recall that the domain of a flow is the cuboid Ω on which the image is defined. Consider the flow $f : \Omega \rightarrow \mathbb{R}^3$, define its “valid domain” as

$$\text{vdom } f = \{\omega \in \Omega : \omega + f(\omega) \in \Omega\}, \quad (18)$$

i.e., those points that are kept in Ω when displaced according to the flow. Ideally the valid domain could be the same as Ω . However, sometimes part of the moving image is cropped out in the fixed one, or due to algorithm’s imperfection, the valid domain might unavoidably be a proper subset of Ω , which is called domain shrinking. In our network, an image might need

to be warped multiple times. Warping with a domain-shrinking flow field will introduce glitches on the boundaries. This is especially undesirable if the warping is used to compose inverse flows, which makes values on the boundaries non-zero artefacts. To partially resolve the problem, we assume the central part of the composed flow is still valid, hence only compute invertibility loss in the central area.

C. Flow Composition

If we are composing two dense flows, there is nothing much interesting to do other than following the definition. However, if the first flow to be composed is actually an affine transform, alternative formula for composition could speed up computation, avoid interpolation and eliminate further domain shrinking. Simple calculation gives

$$(Ax + b) \star f = (I + A)f + Ax + b, \quad (19)$$

which does not involve warping at all.

D. Orthogonality Loss

Computing orthogonality loss involves singular values of $I + A$. The square of these singular values are exactly the eigenvalues of $(I + A)^T(I + A)$. Since the loss is a symmetric fraction of these eigenvalues, it can be rewritten as a fraction of the characteristic polynomial of $(I + A)^T(I + A)$ by Viète's theorem. The formula is cumbersome but not difficult to derive.

E. Command Lines for Traditional Methods

In this section we record the command lines used for traditional methods.

For registration (liver and brain) with ANTs [4], the command we use is the following:

```
antsRegistration -d 3 -o <OutFileSpec>
-u 1 -w [0.025,0.975]
-r [<Fixed>,<Moving>,1]
-t Rigid[0.1]
-m MI [<Fixed>,<Moving>,1,32,Regular,0.2]
-c [2000x2000x2000,1e-9,15]
-s 2x1x0 -f 4x2x1
-t Affine[0.1]
-m MI [<Fixed>,<Moving>,1,32,Regular,0.1]
-c [2000x2000x2000,1e-9,15]
-s 2x1x0 -f 4x2x1
-t SyN[0.15,3.0,0.0]
-m CC [<Fixed>,<Moving>,1,4]
-c [100x100x100x50,1e-9,15]
-s 3x2x1x0 -f 6x4x2x1
```

For registration (liver and brain) with Elastix [5], the command we use is the following:

```
elastix -f <Fixed> -m <Moving>
-out <OutFileSpec>
-p Affine -p BSpline_1000
```

For affine pre-alignment (brain) with ANTs, the command we use is the following:

```
antsRegistration -d 3 -o <OutFileSpec>
```

```
-u 1 -w [0.025,0.975]
-r [<Atlas>,<Image>,1]
-t Rigid[0.1]
-m MI [<Atlas>,<Image>,1,32,Regular,0.2]
-c [2000x2000x2000,1e-9,15]
-s 2x1x0 -f 4x2x1 -t Affine[0.1]
-m MI [<Atlas>,<Image>,1,32,Regular,0.1]
-c [2000x2000x2000,1e-9,15]
-s 2x1x0 -f 4x2x1
```

APPENDIX D ADDITIONAL FIGURES

This section includes several example illustrations of results.

A. Figures for Liver Registration

Figure 8 compares the methods listed in Table II except for “ADDD” (which is similar to “ADDD + inv”), where three landmarks are selected and the sections of the volumes at the height of each landmark *in the fixed image* are rendered. This means the red crosses (landmarks in the moving and warped images) indicate the projections of the landmarks onto those planes. It should be noted that though the sections of the warped segmentations seem to be less overlapping with those of the fixed one, the Segmentation IoU is computed for the volume and not the sections. It might well be the case that the overlap is not so satisfactory when viewed from those planes yet is better when viewed as a volume. Similarly, overlapping red and yellow crosses do not necessarily imply overlapping fixed and warped landmarks as they might deviate along *z*-axis.

In Figure 9, we compare ADDD + inv, ADDD, ADD, AD and D, four differently cascaded networks, one with an extra loss term. The data prove cascading networks significantly improves the performance, because it better registers images with large displacement.

Figure 10 illustrates the intermediate flows produced by ADDD network on two CT liver scans. Each subnetwork registers the images better, increasing Segmentation IoU and lowering Landmark Distance.

B. Figures for Brain Registration

Figure 11 exemplifies the methods on two brain scans. Comparison between our methods and traditional methods proves the applicability of our methods to 3D brain registration. Comparison between ADD and DD* shows that integrating affine registration subnetwork is effective.

Figure 12 compares unsupervised and supervised VTN. The figure clearly demonstrates that supervision by the ground-truth generated by traditional methods (ANTs here) does not yield better performance than using our unsupervised training method. Moreover, acquiring the ground-truth is time-consuming, hence not worth the effort.

In Figure 13, there are two dimensions of comparison. Comparing ADD and AD, or D* and DD* shows that performance is gained by cascading more subnetworks. Comparing ADD and DD*, or AD and D* shows that integrating affine registration into the method yields better registration accuracy.

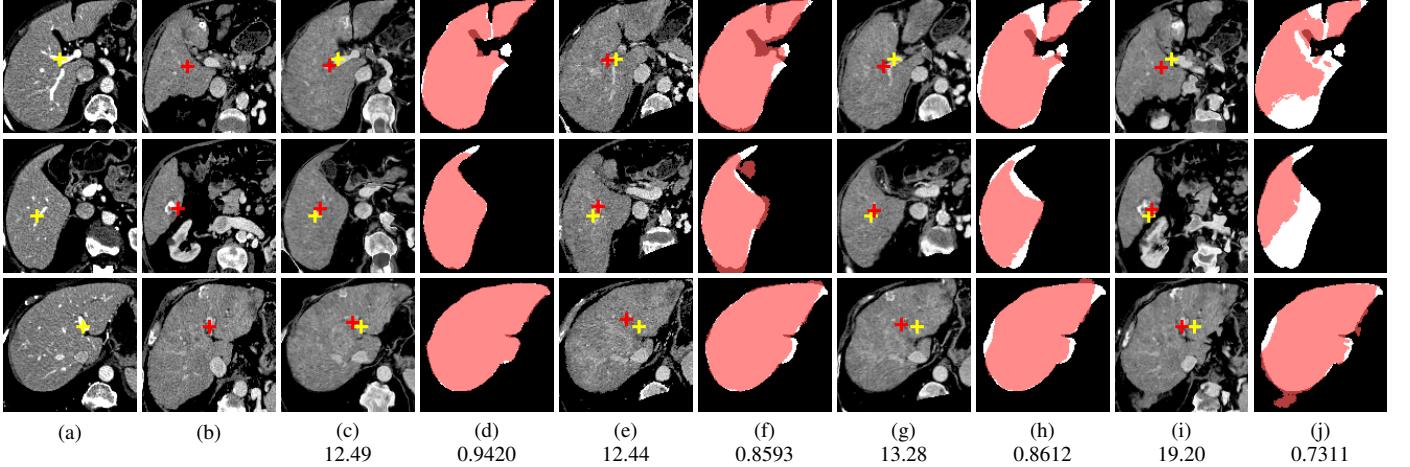


Figure 8: Example comparison among VTN ADDD + inv (c/d), Elastix (e/f), ANTs (g/h) and VoxellMorph-2 (i/j). (a) sections of the fixed image (a CT liver scan); (b) sections of the moving image (another CT liver scan); (c/e/g/i) sections of the warped images and landmark distances; (d/f/h/j) sections of the warped segmentations (white for the fixed and semi-transparent red for the warped) and segmentation IoUs. Crosses indicate the projection of landmarks (L2, L3 and L4 from top to bottom), yellow (lighter) for one in the fixed image, red (darker) for the corresponding one in the moving/warped images. Best viewed in color.

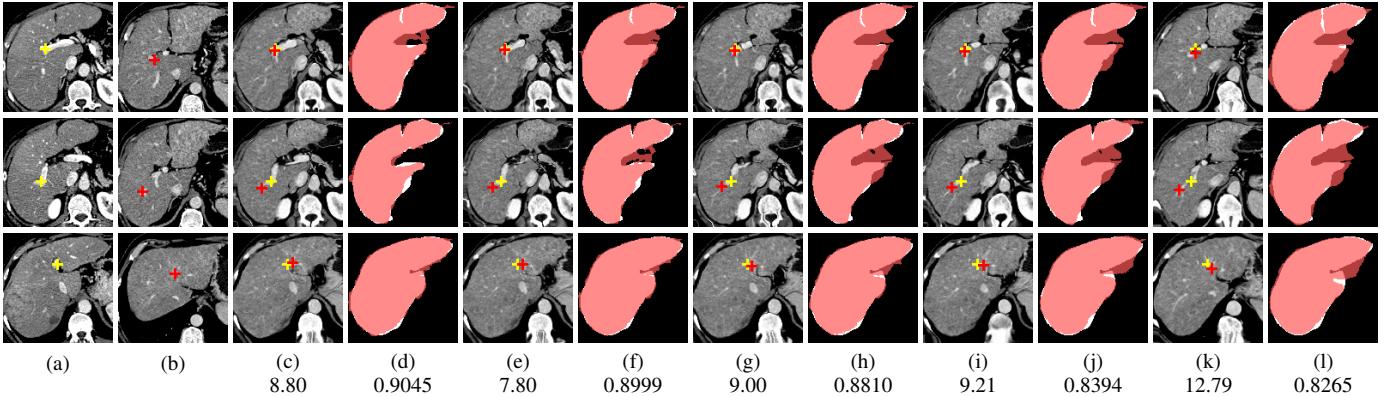


Figure 9: Example comparison among ADDD + inv (c/d), ADDD (e/f), ADD (g/h), AD (i/j) and D (k/l) networks. Columning and coloring are the same as those in Figure 8, except that the fixed image and the moving image are another pair of CT liver scans. Best viewed in color.

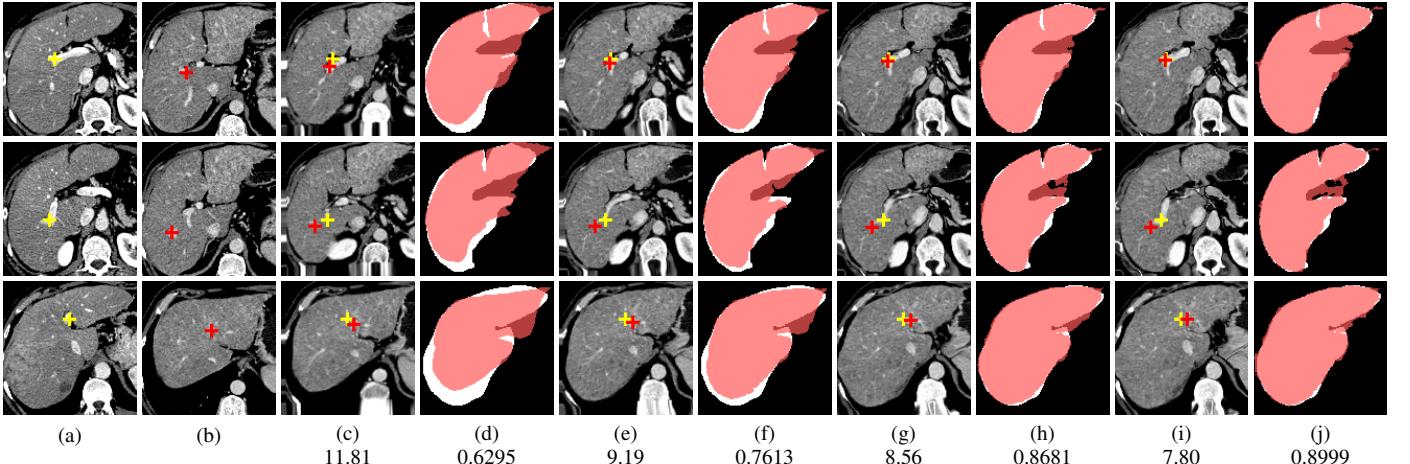


Figure 10: Example intermediate warped moving images by ADDD network. (c/d) warped by the affine subnetwork; (e/f/g/h/i/j) warped by the first/second/third dense deformable subnetwork. Columning and coloring are the same as those in Figure 8, except that the fixed image and the moving image are another pair of CT liver scans. Best viewed in color.

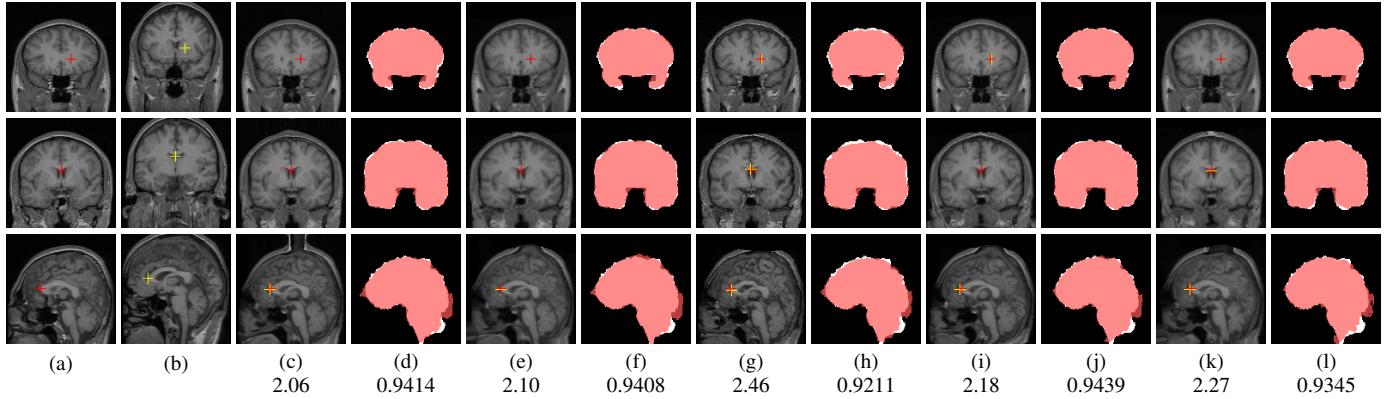


Figure 11: Example comparison among VTN ADD (c/d), VTN DD* (e/f), Elastix (g/h), ANTs (i/j) and VoxelMorph-2* (k/l). The input images to methods with “*” are affinely aligned to a fixed atlas by ANTs and their warped images are transformed backwards according to the affine transformation aligning the fixed image and the atlas for sensible comparison. Columnning and coloring are the same as those in Figure 8, except that the fixed image and the moving image are a pair of MR brain scans and that the landmarks are L7, L12 and L15. Best viewed in color.

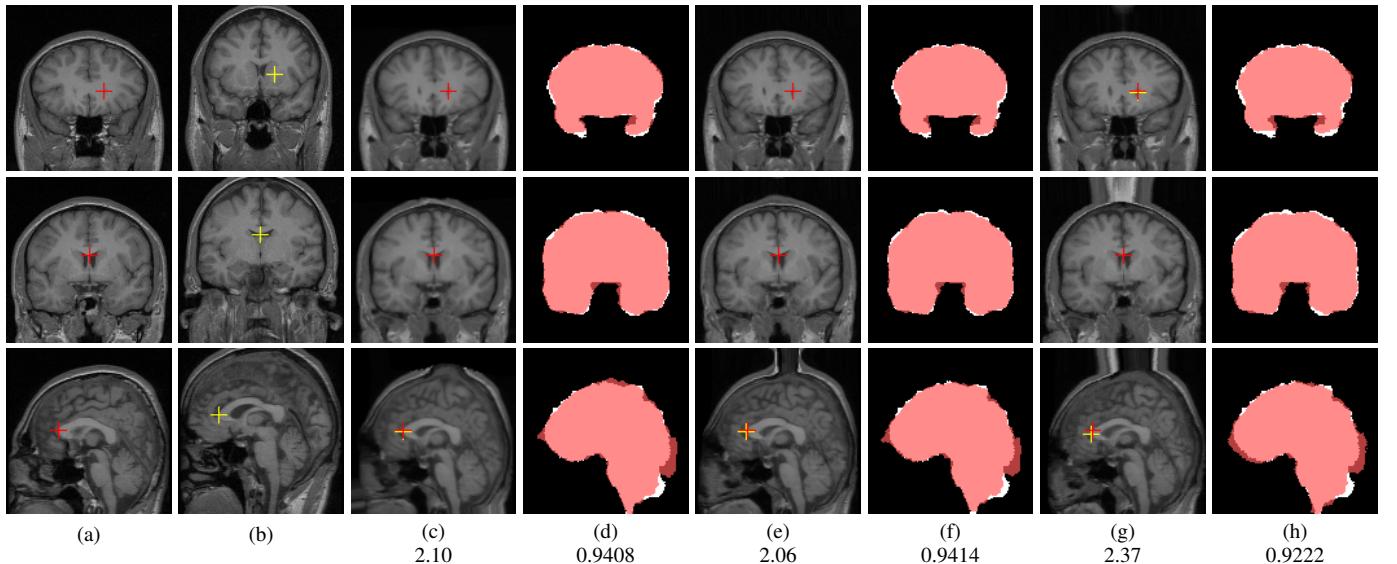


Figure 12: Example comparison among unsupervised and supervised VTN. (c/d) DD*; (e/f) ADD; (g/h) ADD (supervised). Rendering, columnning and coloring are the same as those in Figure 11. Best viewed in color.

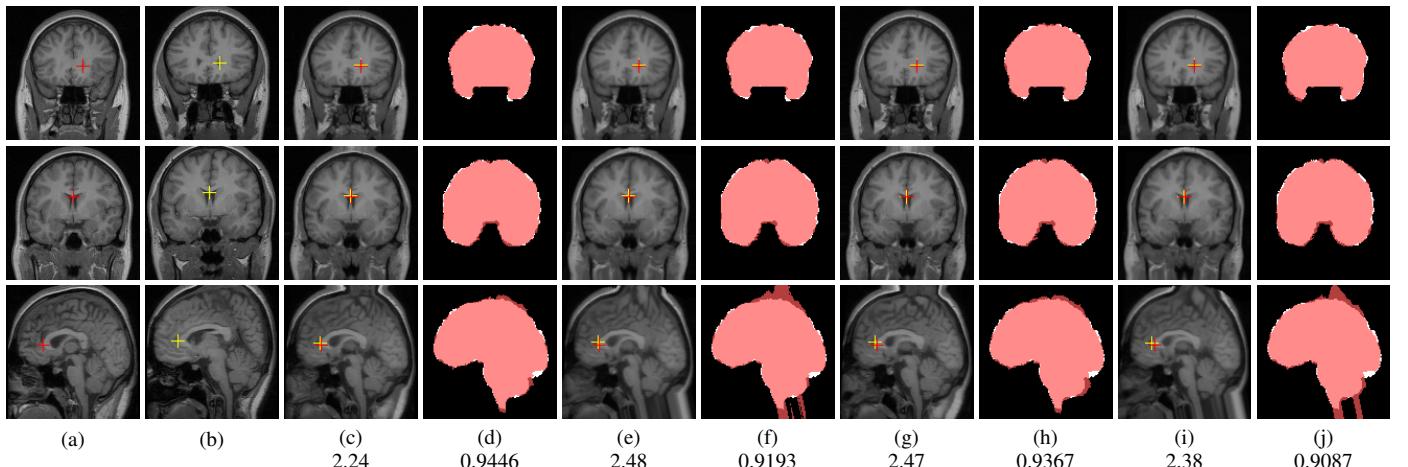


Figure 13: Example comparison among VTN with and without integrated affine registration. (c/d) ADD; (e/f) DD*; (g/h) AD; (i/j) D*. Rendering, columnning and coloring are the same as those in Figure 11, except that the fixed image and the moving image are another pair of MR brain scans. Best viewed in color.