Non-Rigid Volume Registration For DCE-MRI Using Deep Learning

Jonatan Ferm

May 2021

Abstract

In dynamically contrast enhanced magnetic resonance imaging (DCE-MRI) multiple images are generated for a single scan. Before further analysis can be done, the images needs to be registered. Traditional registration techniques can yield impressive results, but are often very computationally expensive.

In this thesis a neural network model using spatial transformer networks is produced. This model aims to solve the registration problem, while not being too computationally expensive.

Several versions of the model are trained, on a limited data set of DCE-MRI scans of rabbit livers. The registration accuracy of these versions are then compared to each other, and to two baseline methods: ANTs and Elastix.

The model is shown – with a similarity metric and visual inspection – to register the volumes less accurately than ANTs, but more accurately than Elastix. Notably this is done with orders of magnitude less computational expense after training.

Acknowledgements

This Master Thesis Project was made possible with collaboration from Dana Peters at the Yale School of Medicine, who I would like to thank for all the invaluable advice and help she has given me throughout this process.

I would also like to thank my advisor as LTH, Anders Heyden, who has especially helped me form the report.

Contents

Introduction

Registration is the task of aligning different images or volumes with each other. This is a fundamental task in medical image analysis. In some cases a translation or rotation is enough to align two images, this is called a rigid registration. That is however, often not enough in medical image analysis, where images can differ in more complex ways due to anatomical, physiological, or pathological reasons. In such cases non-rigid registration method, which can capture local deformations, are needed.

One application where a non-rigid approach may be necessary is dynamic contrast enhanced magnetic resonance imaging (DCE-MRI). It is an imaging method where a series of images are captured over time, while the patient is often breathing freely. The breathing results in complex motion which, if not corrected for, will impact analysis.

There are traditional non-rigid methods; such as Advanced Normalization Tools (ANTs) [2], or Elastix [6]. These tools do a better job than rigid registration methods to solve this problem. However, one problem with many such methods is that they are often prohibitively computationally intensive. Using a regular workstation you might not be able to register more than a single scan in a day.

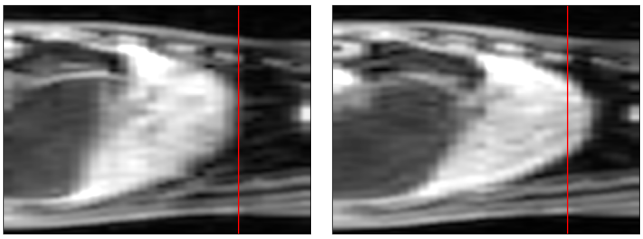


Figure 1.1: Two MR scans of a liver taken in sequence. Breathing motion has displaced parts, and no simple geometric transformation can align them.

The last decade have shown deep learning to be a method capable of solving a multitude of problems in the image analysis "space". While training deep networks is not a quick process, once they are trained they are capable of performing their task in seconds.

This thesis applies recent advances in deep learning to the problem of registering volumes of DCE-MRI scans of livers. The goal is to develop a method that can achieve similar or better registration accuracy than traditional methods, and doing so faster. A model will be trained on a training set of scans of rabbit livers, where breathing motion is extensive. This model will then register test set of scans, and the results will be compared to a registration of the same test set using ANTs and Elastix.

Theory

This section aim to give a short overview of the science behind capturing DCE-MRI scans, and the deep learning methods used to register the scans.

2.1 Magnetic Resonance Imaging

Magnetic resonance (MR) scanners work by listening to radio signals transmitted by your body. Transmitting coherent radio signals is unfortunately not something your body does without outside help. Thus before being able to listen, the scanner must first help your body create something to listen to.

Radio signals are structured propagating changes in the electromagnetic field. One ubiquitous aspect of living beings that interacts with the electromagnetic field is the hydrogen nucleus, which is especially abundant in water and fat. The proton in the hydrogen nucleus has a weak magnetic moment due to its charge and inherent angular momentum (spin).

However, under normal circumstances the changes to the electromagnetic field created by the nuclei in your body cannot really be measured. The spins of different nuclei point in different directions, creating fields that cancel each other out. This results in a net field that is functionally zero. By aligning the spins, their fields will no longer cancel each other out. This is done by applying a very strong outside magnetic field, B_0 (usually in the range of 1 to 7 Tesla). The nuclei will either align with the field or point against it, as these are lower energy states. A very small majority (based on the Boltzmann distribution) will align with the field, creating a measurable effect.

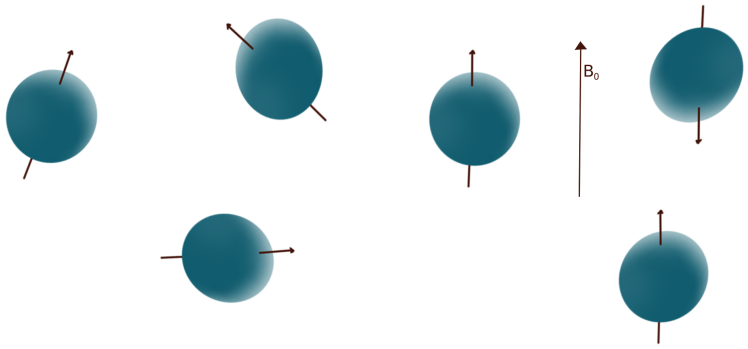


Figure 2.1: Representation of hydrogen nuclei before and after being aligned by an external magnetic field.

Once there is a measurable field; there needs to be changes to measure. Preferably these changes should only happen in the part of the body that is to be imaged, getting signal from other parts is not very helpful.

To differentiate parts of the body the property of Larmor precession frequency is used. Precession is a phenomena that can occur in spinning object, where the axis of rotation itself rotates along a second axis. This can for example be seen in toy spinning tops, which especially start to precess (or wobble) if you poke them. When a magnetic axis rotates along a second axis it is called Larmor precession. The Larmor precession frequency of a hydrogen nuclei is only dependant one thing: the field strength of the background magnetic field. Thus by applying a second magnetic field, that varies in strength along the body, different parts will have different precession frequency.

But like a well spun spinning top, the precession will not be noticeable unless we poke it. To increase the angle of precession (and make it noticeable) a third magnetic field is applied. This fields needs to have two important properties different than the first two fields: It should be perpendicular to the first two fields; and it should pulse with a frequency that matches the Larmor frequency of the region of interest.

How this increases the angle of precession merits an explanation. The nuclei precess along an axis aligned to the background magnetic field. When the perpendicular third field is applied the background field changes orientation slightly, tilting the precession axis. This separates the axes of magnetic moment and precession, creating an angle between them. Since the field is pulsing at a frequency matching the precession frequency, the induced tilt will follow the precession. This will create an compounding effect, increasing the angle of precession until the perpendicular pulse stops. Outside the region of interest the precession frequency will be different, not matching the frequency of the axis tilting field, and thus not compounding the increased precession angle.

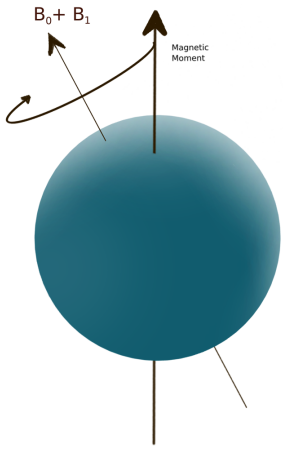


Figure 2.2: When the precession inducing B_1 field is added to the aligning B_0 field, the direction of the magnetic momentum no longer matches the background field. This mismatch causes the nuclei to precess, like a spinning top would if the table it is on is gets tilted.

When the pulse does stop, two things will happen: The spins will start realigning with the background field, and the precession of the different nuclei will start dephasing. This affects the strength of the field, and thus the signal, they create. How quick the nuclei is realigned and dephased is know as relaxation time, and it is dependant on its substrate (for example a nuclei in water will realign slower than one in fat).

The changing field created by the nuclei, as they realign and dephase, is the signal picked up by the scanner. To create an image from the signal, this process is repeated several times, with additional gradient fields applied. These gradient fields will further alter the Larmor frequency, affecting the frequency content of the signal created by the nuclei. Frequency analysis techniques can then be used to convert the collected signals to an image.

2.1.1 Dynamic Contrast Enhanced MRI

In DCE-MRI a contrast agent is injected into the patient before scanning. This is some substance that alters the relaxation time of tissue in its proximity. Gadolinium based contrast agents are common, which has paramagnetic properties that shorten the relaxation time.

After injection, the patient is scanned multiple times, over anywhere from a minute to several hours. By comparing the scans, and how the signal strength changes, the dynamics of how the contrast agent moves through the body can be inferred.

One usage for this is to identify tumors. Without contrast enhanced methods, the signal from tumor tissue might not differ much from healthy or necrotic

tissue. But since tumors tend to have an abnormal inflow of blood, they will also have an abnormal inflow of contrast agent. This makes for a much higher contrast in the image between different tumor and non-tumor tissue.

2.2 Image Registration

Image registration is the task of aligning different images to each other using matching features. Even though two different images may be of the same subject they most likely will differ. Either the orientation of the camera, or the orientation of subject will be different.

Figure 2.3 shows two images of the same triangle-shaped floral pattern. Compared to the left image, the right image is distorted. This may either be due to changing the alignment of the fictitious camera, or changing the triangle – the results are the same.

To align these images a transform needs to be found that matches features of the images. This is a relatively simple example where a rotation and scaling of one axis is enough to exactly match the images. Such a transformation could be implemented with a sampling grid, visualized in Figure 2.4. If every point in right image is sampled where there are grid points in the grid, and those points are then mapped to a Cartesian grid, one would end up with the left image. In this case with a much poorer resolution since the grid is pretty sparse for visualisation purposes, in a real scenario a much finer grid would be used.

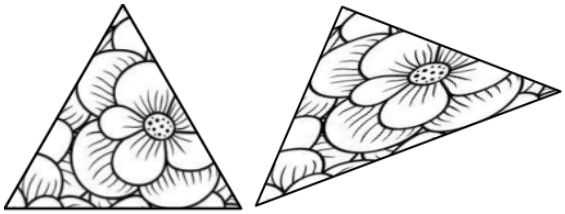


Figure 2.3: Two images of a patterned triangle.

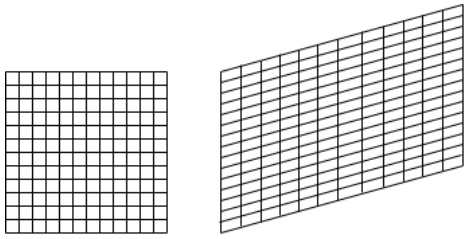


Figure 2.4: A sampling grid, and the corresponding Cartesian grid, that could be used to align the images in Figure 2.3.

The above example only needs two parameters to register the images. Sadly this is seldom enough. In physiological use cases you may need as many transformation parameters as pixels in the image. A more complex deformation of the example triangle can be seen in Figure 2.5. Here parts of the image has been non-uniformly stretched. To create a sampling grid to return to the original image there is no simple transform requiring only a couple of parameters. To describe the corresponding transformation grid in Figure 2.6 one needs to specify the displacement of each grid point. To avoid "too much" under-determination, which would enable any image to be registered to any other image, artificial constraints may need to be introduced. Some constraint on the smoothness of the transform could for example be used, if it is reasonable that the cause of the deformation causes a somewhat smooth deformation.

The problem of registration can be summarized as finding the grid in Figure 2.6 given the triangles in Figure 2.5 and 2.3. How this is achieved differs significantly between different methods.

Figure 2.5: The patterned triangle deformed non-rigidly.

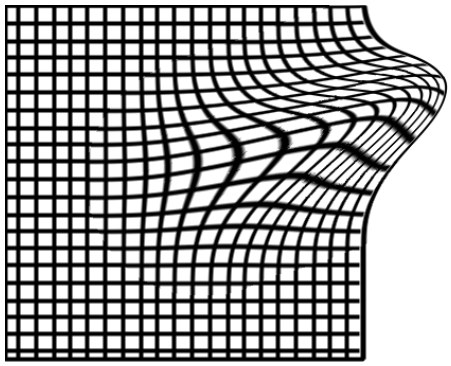


Figure 2.6: A sampling grid that could transform the deformed triangle from Figure 2.5 to be registered to the one from Figure 2.3

2.3 Artificial Neural Networks

An artificial neural network (ANN) is a computational model inspired by the biological network work of neurons found in living animals nervous systems. The goal of an ANN is to generate some helpful output from a given input. To do this large training data sets with inputs and matching desirable outputs are often used. A training algorithm iterates over the data, modifying the network to produce output matching the know desirable output.

The building block of ANNs are nodes that take some input and generate an output. The output is generated by calculating a weighted sum of the input which is then passed to an activation function. This is described by:

$$a = \begin{bmatrix} 1 & x_1 & ... & x_n \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ ... \\ w_n \end{bmatrix}$$

$$y = f(a)$$

Where the input x and a static bias is weighted by w and passed to the activation function f. Given a set of input with known output, a set of weights can be found by an optimization algorithm, commonly stochastic gradient descent.

In ANNs the nodes are divided into layers. The values of the nodes in the first layer are the input to the network. The consequent layers uses previous layers as input, with the last layer being the output of the network. Figure 2.7 show a very simple 3 layer network with 2 input nodes, 3 hidden layer nodes, and 1 output node. A practical network, like the one used for this report, can have millions of nodes per layer over a dozen layers.

The output of the final layer represents the output of the ANN. To compare this output to the known desired output a loss function is needed. The loss

function is some function takes two outputs and produces a single number. A loss function for images might calculate the squared difference for each pixel, and then return the mean value of that.

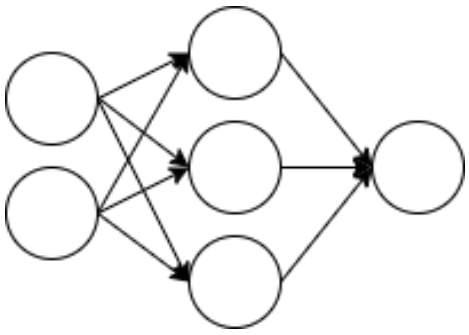


Figure 2.7: Nodes in a network, divided into 3 layers.

2.3.1 Activation function

One of the most common activation functions have historically been the rectified linear unit (ReLU). ReLU is defined by

$$y = \begin{cases} 0, & \text{if } x \leq 0 \\ x, & \text{if } x > 0 \end{cases}$$

However, this activation function has the limitation that the gradient is lost when $x < 0$. It is very common that the optimizer uses the gradient to optimize the weights.

In recent years a method that surmounts this problem has gained popularity. Leaky ReLU (LReLU) is a similar activation function that, rather than completely nullifying values less than 0, leaks a little. LReLU is defined by

$$y = \begin{cases} kx, & \text{if } x \leq 0 \\ x, & \text{if } x > 0 \end{cases}$$

Where k some, usually small (< 1) positive, value that can either be set beforehand as a hyperparameter or learned by the network. By keeping x as a factor below 0 a non-zero gradient is preserved, and can be used by the optimizer.

2.3.2 Convolutional Neural Networks (CNN)

In the example given above the nodes in a layer are connected to all nodes in the previous layer, for most applications this is not what is usually done. Either due to the vast amounts of weights that would need to be stored and updated, or due to the approach being suitable to the nature of the application.

In image analysis a common approach that the nodes only have local connection to the previous layer. This is visualized below in Figure 2.8. Here each node in the second layer takes input from 3 nodes in the previous layer.

This represents a one-dimensional convolution with a kernel size of 3. In image analysis 2-dimensional data is common, then a kernel size of 3 represents a 3×3 kernel. Images can also have multiple channels, which is common when dealing with color images. Color images often have 3 channels, one each for: red, green, and blue.

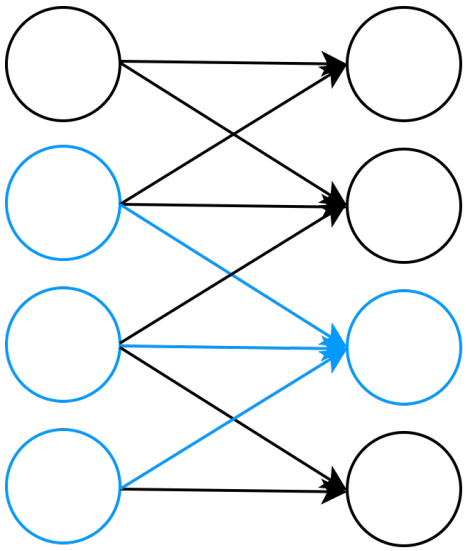


Figure 2.8: Two layers in a convolutional network. The connections to one node is highlighted in blue. The weights used for the highlighted connections are used for all other nodes as well.

2.3.3 Group Normalization

Normalizing data is a staple of machine learning and makes training deep networks possible. As the data is transformed by the layers of the network, it will generally become more and more disjointed. Some nodes might output huge values while some output very small values. The output distributions might also vary significantly in shape and mean. This makes finding weights that successfully combine these distributions harder. Some weights might need to be very big, while others very small; The output might be very sensitive to changes in some weights, while barely affected by changes to others.

Traditionally batch normalization has been a popular approach to normalizing data. Batch normalization works by, for each feature (pixel in an image), finding a mean and variance for that feature over a batch of training data. This works great as long as batch sizes are reasonable large. If batch size is constrained, this is no longer a viable option.

Group normalization [8] works by grouping a number of features and finding a mean and variance of one data entry. For images this means we find the mean and variance of the pixels in part of the image, rather than the mean and variance per pixel over many images.

This works under the assumption that the sets of features grouped together share somewhat similar distributions, which is a more lax assumption the smaller the groups are. But smaller groups leads us back to the problem we tried to escape, not enough data to find a robust mean and variance. The original paper on group normalization [8] suggested that: a good compromise for this tension is to use one forth as many groups as there are channels. This thesis uses 4 groups, which is in the range of suggested groups in the aforementioned paper.

2.3.4 Spatial Transformer Networks

One of the biggest advancements in image analysis in recent years have been spatial transformer networks [4]. A spatial transformer network is used as a layer in a neural network, just like a convolutional layer.

A spatial transformer network works by taking some input feature map U, then processing through a localization net to produce a set of transformation parameters θ for some transform T. The parameter set θ might for example be translation, scale, shear, and rotation of an affine transform. This transform should produce a sampling grid that can be used to sample U to produce V, the transformed feature map.

This could for example be used in a network that identifies the color of birds, from a data set of bird pictures. Then the input image U would be a picture of a bird. Which is passed to the localisation net which identifies where in the image the bird is. In this example the transform could be as simple as cropping a small image centered at the bird, which would just make the localisation net output the x and y coordinates as θ. The cropped image (V) can then be passed on for further processing.

It is important that the transform T is differentiable with respect to θ, otherwise the optimizer used to train the network might not work.

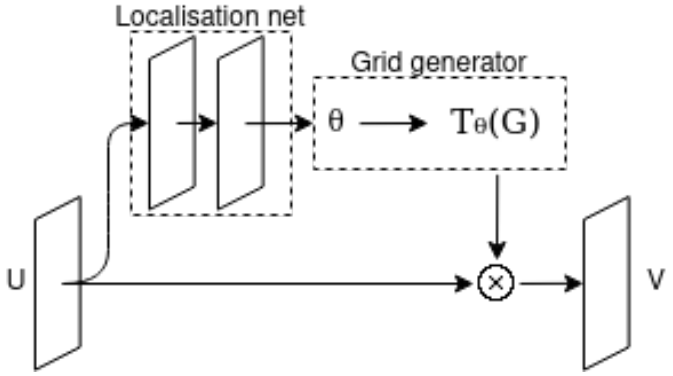


Figure 2.9: A spatial transformer network layer, taking an input feature map U, producing a set of transformation parameters θ, for the transformation T, which take some input G, to produce a transformed output map $T_\theta(U) = V$.

2.3.5 Grid generator

As mentioned, it is important that the transformation is differentiable. Fortunately there are methods that produce differentiable transforms. My implementation uses the method described in [3].

This method works by creating a sampling grid, that when used to sample an input performs a differentiable transformation. The grid is created by taking a Cartesian grid and integrating it over a stationary velocity field. This is described by the differential equation

$$\frac{\partial \boldsymbol{\phi}^{(\boldsymbol{t})}}{\partial t} = \boldsymbol{v}(\boldsymbol{\phi}^{(t)})$$

where $\boldsymbol{v}$ is a stationary velocity field. $\boldsymbol{\phi}^{(t)}$ is the grid at time t, which is initially $(\boldsymbol{\phi}^{(0)})$ a Cartesian grid. The grid used for the transformation is $\boldsymbol{\phi}^{(1)}$. Given a stationary velocity grid, which is what the localisation network outputs, the solution can be solved for numerically. My implementation does this using the method scaling and squaring, a method for numerically calculating matrix exponentials. The method can also be used to integrate vector fields, as described by Arsigny, et al. [1].

The integration of a Cartesian grid over a velocity field, to generate a sampling grid used as a registration transform, is visualized below in Figure 2.10

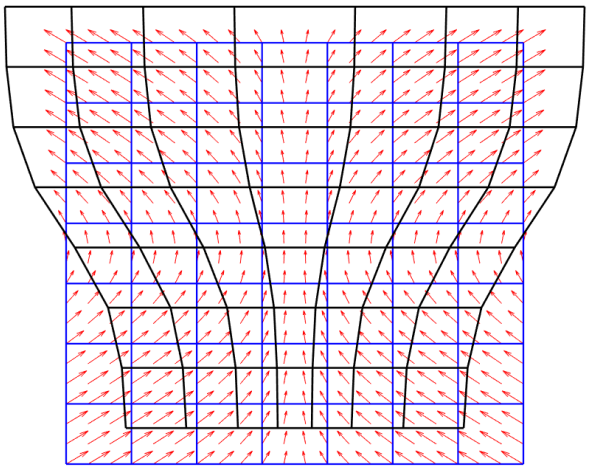


Figure 2.10: A sampling grid (black) generated by integrating a Cartesian grid (blue) over a stationary velocity field (red).

Method

3.1 Data set

The data set consist of 65 DCE-MRI scan series. The data was obtained using a 3 Tesla Siemens Prisma scanner. 80 scans were performed at 2.7s intervals. These scans have a field of view of $200 \times 119 \times 80$ mm, a resolution of $384 \times 228 \times 32$, a repetition time of 3.45 ms, an echo time of 1.28 ms, and a 9° flip angle.

The imaging was performed during bolus injection of 0.1 mmol/kg Gd-DTPA contrast agent. The rabbits consisted of control rabbits, and rabbits with orthotopic VX2 liver (REF) tumors, at about 3 weeks of tumor growth, and one or two weeks after therapy with trans-arterial chemo-embolization (TACE).

3.1.1 Data preprocessing

The data was cropped to a square, in the XY plane, over the region of interest. It was then re-sampled to a resolution of $256 \times 256 \times 32$. Data set entries need to be pairs of scans, one moving and one fixed volume. 10 pairs were created for each scan in the series. This was done by pairing them with the 5 scans before, and 5 scans after them in the series.

3.2 Localization Net Structure

The localization net uses a structure similar to U-net[7], with convolutions and skip connections. Mainly the network utilizes blocks of $3 \times 3 \times 3$ convolutions, group normalization, and $2 \times 2 \times 2$ max pooling layers to reduce the data dimension. The reduced data is then resized with convolutions, group normalization, and transposed convolutions. This structure is visualized in Figure 3.1 below. The input of the network is a $256 \times 256 \times 32 \times 2$ volume. The last dimension are the channels, one channel is for the fixed volume and the other channel for the moving volume. The final output of the network is a $256 \times 256 \times 32 \times 3$ volume, here the channels represent the x, y, and z-velocity of the transformation.

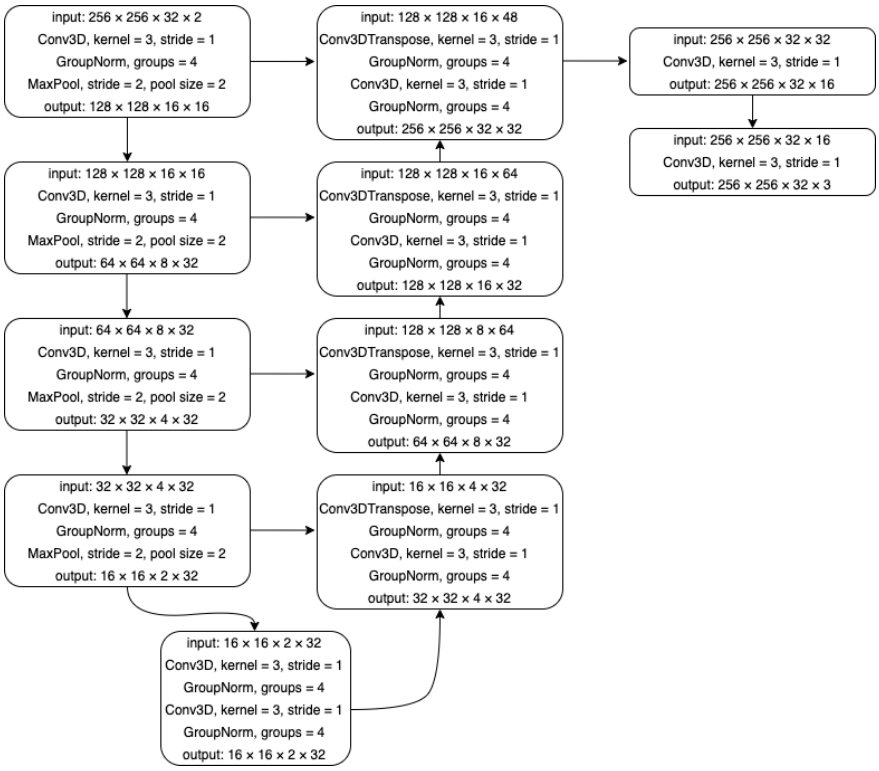


Figure 3.1: Structure of the localization net used.

3.3 Training parameters

The network was trained over 10 epoch using a Nvidia GTX 1080 Ti, the ADAM[5] optimizer. The learning rate started at 10^{-4} and began decreasing after the first 3 epoch to finally reach 10^{-6} at the last epoch. 10% of the data set was separated into a validation data set.

3.4 Loss Function

The training used the loss function with two parts: one for the difference between fixed and registered volume, and one for rewarding smoothness of the transform. If smoothness is not rewarded the resulting transforms will pick up on a lot of noise in the images and result in transforms capturing other things than the motion we are interested in corrected. The loss function is defined by:

$$||\mathrm{F} - \mathrm{T}_\theta(\mathrm{M})||_2^2 + \lambda||\nabla \mathrm{T}_\theta||_2^2$$

where F is the fixed volume, T is the registration transformation with parameters θ, λ is a tuning parameter to regulate the importance of smoothness, $\nabla \mathrm{T}_\theta$ is

the gradient of the transformation grid.

In short this loss function tries to find a transformation that matches two volumes as closely as possible while maintaining a level of smoothness decided by the parameter λ.

3.4.1 Volume Series Registration

The network is trained to register one volume with another one. The DCE scans used have around 80 scans each. To register the whole series one reference volume from the series is chosen, then all other volumes are registered to that volume.

3.4.2 Evaluating the result

The network is trained several times with different values for the smoothness parameter λ. Results with different λ will be compared to each other to investigate its impact, and to find be best value. These results will then be compared to registrations using ANTs and Elastix. Unfortunately only a few baseline scans are done. This is due to the inherent problem we are trying to solve, it takes a very long time to register scans using these methods.

The success will be measured by visual inspection, and a similarity metric is improved by registering. Some examples will be further investigated by visual comparison, of both the registration and visualisation of the registration transform.

Similarity metric

The similarity metric used to compare how well a series of scans have been registered is the mean mean square error.

$$\frac{1}{N-1}\sum_{i}||\mathrm{F}-\mathrm{T}_{\theta}(\mathrm{M}_{i})||_{2}^{2}$$

Each frame in a series of N scans are registered to a reference frame F. For each scan the voxel mean squared difference in voxel intensity is calculated, then the mean value for the whole series is used as the similarity metric.

This is a simple metric, that unfortunately is not without faults, which are discussed in section 5.4.

Baseline methods parameters

ANTs is run with the parameters "type of transform" = 'SyN', and "reg iterations" = [160, 80, 40]; Elastix in run with "MaximumNumberOfIterations" = 500, and "FinalGridSpacingInVoxels" = 4.

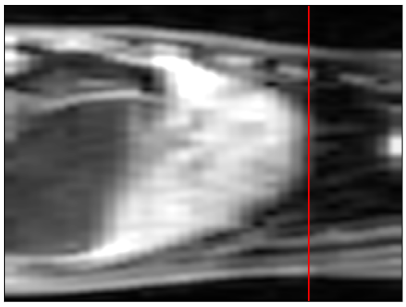
Results

Figure 4.1 shows examples of registrations made from networks trained using a different value for the smoothness parameter λ. The images are slices of the three dimensional volumes. The notable difference between the moving and fixed image before registration is the position of the liver. A red reference line has been drawn in the images at the position where the top of the liver is located in the fixed image. The sampling grids used for the transformation is also shown in the right column. Figure 4.2 show the same slice registered using ANTs and Elastix.

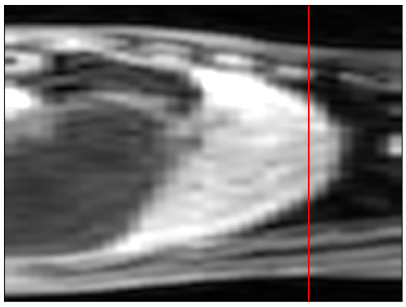
Table 4.1 show how the model performs with different values for λ over the full data set. The value shown is the difference in the mean of the mean squared error. This value is calculated by first taking the mean squared error between all volumes in a scan to the reference volume in that series, before and after registration. The difference between these values are then calculated as an indicator how how much of an improvement the registration was. The mean value of these value for all the scans are shown in the table with a 95% confidence interval.

Figure 4.3 show scatter plots of the mean of the mean squared errors before and after registration. The blue dotted line if a reference line showing where no improvement is made. Under this line improvement is made and over it the attempt at registration has increased the mean MSE. Notably there are some data points over the blue line, an example of a registration from one of these is shown in Figure 4.4.

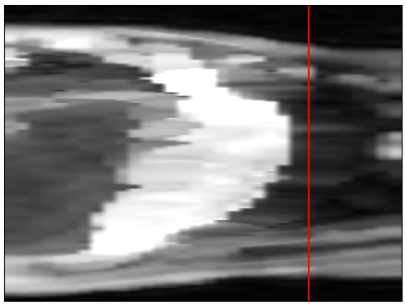
Fixed image.



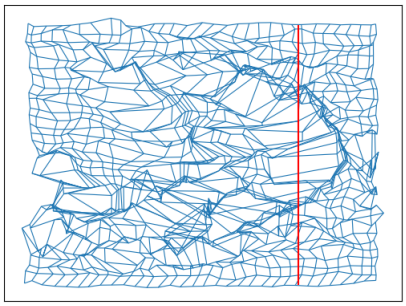
Moving image.



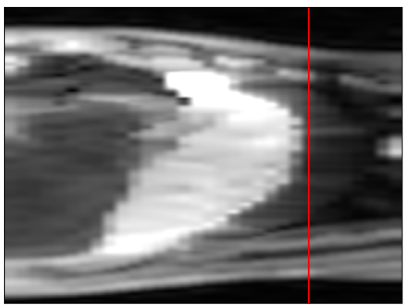
Registered image.
$\lambda = 0$

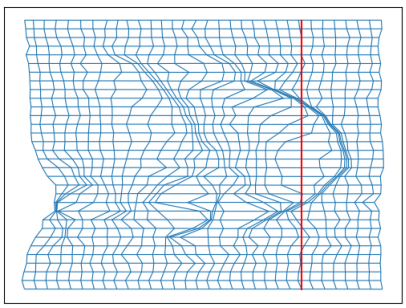


Sampling grid.

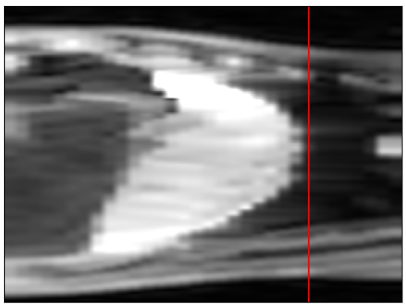


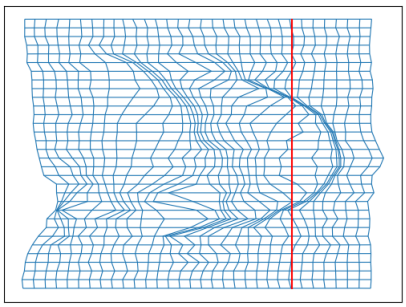
$\lambda = 0.05$





$\lambda = 0.1$





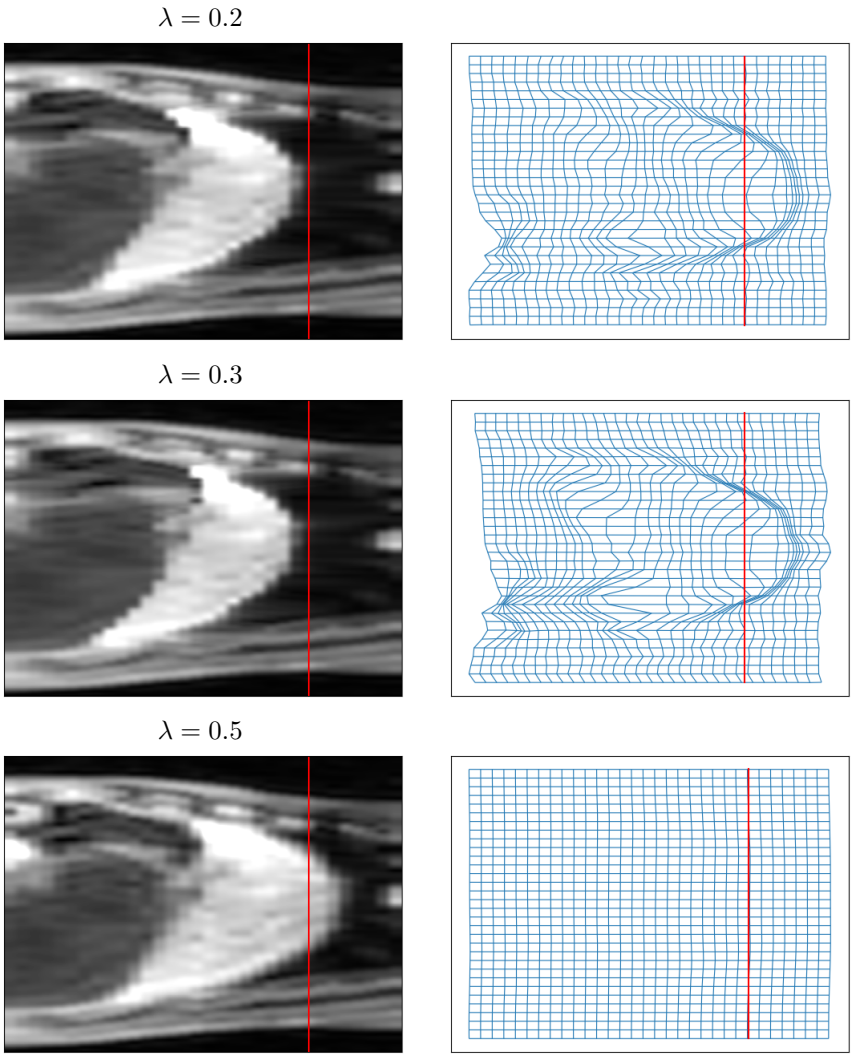
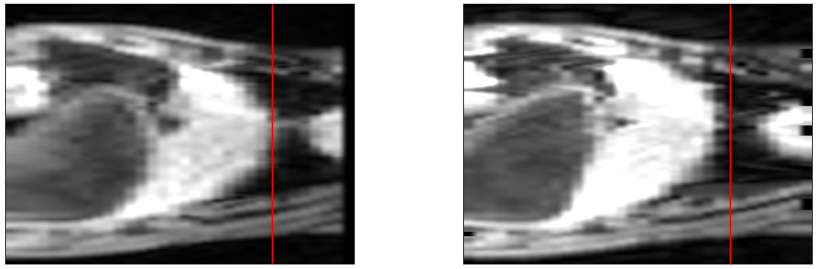


Figure 4.1: A slice of scan registered with models trained with different values for λ. The fixed and moving images have the liver in different positions. The red line shows denotes the edge of the liver in the fixed image, for reference. The right column shows the sampling grid used to transform the moving image.



(a) ANTs registration (b) Elastix registration

Figure 4.2: Examples of ANTs and Elastix registration on the same volume as in Figure 4.1

λ	mean MSE difference
.05	.460 [±.064]
.1	.475 [±.054]
.2	.192 [±.042]
.3	.078 [±.048]
.5	.041 [±.005]

Table 4.1: The mean mean squared error difference for different values of λ. After the scans in a series have been registered to a time point, the MSE is computed for those scans before and after registration. This is the difference mean MSE is the whole series, with a 95% confidence interval. As hinted in previous figures, the smoother transforms change the volumes less and thus produces smaller differences.

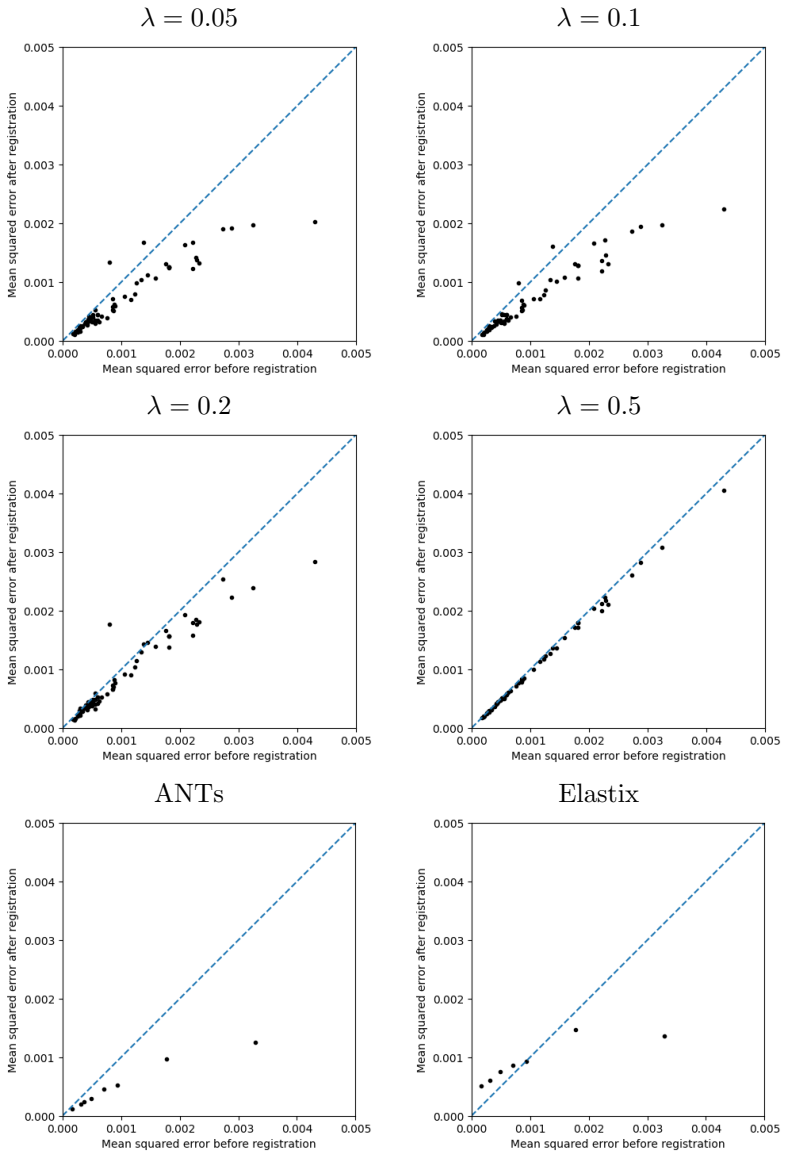


Figure 4.3: Mean squared error before and after registration, for different values of λ and the two baseline methods. A lower value for λ, and thus a less smooth transform, show a clear correlation to a more closely matched registration. The two baseline methods seem to perform overall better for the scans are more mismatched to begin with. Fewer values for ANTs and Elastix have been computed due to computational constraints.

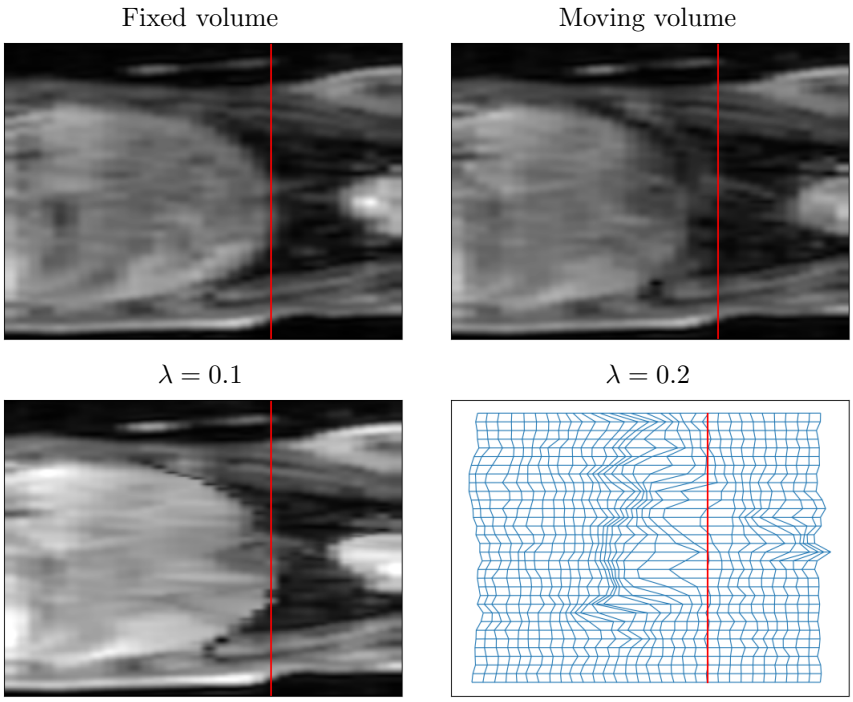


Figure 4.4: An example of when the model produces a result that is less similar than before registration. The registered scan has an increased average pixel intensity compared to the fixed volume. Highly likely caused by this scan having a lower average voxel intensity than the rest of the data set.

Discussion

5.1 Baseline comparison

Looking at the scatter plots in Figure 4.3, it is clear that the deep learning method performs more accurate registrations when λ is decreased. However, even for the lower values the registration is not as accurate as ANTs, which performs well both looking at the mean MSE metric and under visual inspection. Visual inspection also indicate that the lower λ values produce transformations that are not consistent with anatomy, while the higher one does. A value around .1 to .3 seem to produce a reasonable registration while respecting anatomy.

As seen in Figure 4.2 Elastix does move the edge of the liver towards the reference line. However, it is done at the cost of an image that is more deformed than any other example. The slice shown is one of the scans where Elastix performs relatively well, indicating that it is not very well suited for this purpose, at least given the used parameters.

5.2 Folding

Under closer inspection of the transform using $\lambda = 0$, seen in Figure 4.1, there is some folding of the mesh. This is a clear sign that the transform is not bijective, something that the transform generating method was suppose to ensure.

This is very likely caused by the numeric integration method used. Scaling and squaring relies on the scaled velocity field being sufficiently small. As the transform is not controlled by the loss function the $\lambda = 0$ case the velocity field clearly becomes too big.

This problem could be solved by increased the number of steps used in by the method, thus scaling the field to be smaller. But since this is not an issue when $\lambda > 0$ and $\lambda = 0$ is a sub optimal choice, due to the noisy and volatile transform it produces, increasing the steps does not seem necessary to produce the optimal transform in this setting.

5.3 Volumes registering less similar

As seen in the scatter plots a couple of the volumes are less similar after the registration. A slice of the actual scan can be seen before and after registration in Figure 4.4.

From looking at the scan it becomes clear that the reason for this that the mean intensity is increased after registration. The pre-registration image clearly has a lower intensity than both the post-registration image and the other scans in this report. The registration has still managed to position the liver to the correct location.

As the images are normalized one-by-one this is most likely due to some extremely bright spot in the image lower the relative intensity of the other voxels. Normalizing all the images together could solve this problem, but this would bring other problems as the MR signal can vary a lot since it is dependant of many factors that vary between different scans. The coil sensitivity in particular is very unlikely to be the same between two scans.

That the training distribution is similar to the test distribution is an underlying assumption needed to make deep-learning work. Deviations from this assumption can cause errors. This can be solved by training with more data, to have the training data distribution encompass a wider range. Possibly this could be solved by using some other normalization scheme that conforms the data better.

5.4 The similarity loss function

There might be some problem using the mean squared error as a measure for a successful registration for this problem. The whole point of DCE-MRI is that the values in the scans should change as the contrast agent propagates. Thus using this loss function might suppress the very effect that we are looking for and trying to measure. It could be worth exploring how much of a problem this is and if another loss function, like a cross-correlation model, might mitigate it.

Another approach that could overcome this, at the cost of some manual work, is to mark some landmarks in the liver and then measure success by how closely those are matched after registration.

5.5 Density correction

After an image has been registered, it is possible that the density changes more than what is physiologically feasible. Take the example images in the registration theory as an example, Figures 2.5 and 2.6. Here density in part of the image will be significantly changed after registration. There is no guarantee that this would not happen when registering the live data. However, the smoothness constraint used may mitigate the risk. Looking at the registered images

in Chapter 4 there does not seem to be any physiologically non-feasible density changes for the preferred λ-values.

There are methods to correct for possible density deformation, such as Jacobian determinant density correction used by Zha, Wei, et al. [9]. This is not done in this model due to time constraints. If this method were to be developed further, some method for density correction should be implemented before clinical use, to ensure more robust results.

5.6 Memory constraints

The hardware I had available to train this model put some constraints on it. The model presented in this report was very close to maxing out the 10 Gb memory of the Nvidia GTX 1080 Ti used. Using more layers or channels was not possible with this setup and might have made performance better.

Memory constraints also prevented the model registering more than two frames at once. Some model that can be fed a full scan series where the cyclic nature of breathing motion could be leverage would likely outperform a model where that information is not available.

5.7 Size of data set

The data set used contains about 100 scan series. From this a few ten thousands of data points are created by combining different scans in the series. It is well established that deep learning methods scale well with, and usually require, larger data sets. It is therefor likely that the deep learning method does not perform as well as it could have, given access to a larger data set. Unfortunately this would not have been feasible for this report, even if the data had been available. The training step would have taken a prohibitively long time with the hardware used. This highlights one of the weaknesses of the deep learning approach.

Conclusion

Compared to the baseline methods it is clear that the deep learning performs in the middle. ANTs performs better; Elastix performs worse. Overall the performance of the methods show that this is a hard problem to solve inherently.

The deep learning model has a couple of advantages and could possibly surpass ANTs given different circumstances.

Training the deep learning model is as time consuming as registering a handful of scans with ANTs. After the training is complete registering a scan can be done in a matter of seconds, where ANTs would require hours.

The deep learning approach would also benefit from more data, while ANTs would not. However, to draw the benefit some combination of more time and more hardware would be needed.

Bibliography

[1] Vincent Arsigny, Olivier Commowick, Xavier Pennec, and Nicholas Ayache. A log-euclidean framework for statistics on diffeomorphisms. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 924–931. Springer, 2006.

[2] Brian B Avants, Nicholas J Tustison, Michael Stauffer, Gang Song, Baohua Wu, and James C Gee. The insight toolkit image registration framework. *Frontiers in neuroinformatics*, 8:44, 2014.

[3] Adrian V Dalca, Guha Balakrishnan, John Guttag, and Mert R Sabuncu. Unsupervised learning for fast probabilistic diffeomorphic registration. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 729–738. Springer, 2018.

[4] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025, 2015.

[5] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[6] Stefan Klein, Marius Staring, Keelin Murphy, Max A Viergever, and Josien PW Pluim. Elastix: a toolbox for intensity-based medical image registration. *IEEE transactions on medical imaging*, 29(1):196–205, 2009.

[7] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[8] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.

[9] Wei Zha, Stanley J Kruger, Kevin M Johnson, Robert V Cadman, Laura C Bell, Fang Liu, Andrew D Hahn, Michael D Evans, Scott K Nagle, and Sean B Fain. Pulmonary ventilation imaging in asthma and cystic fibrosis using oxygen-enhanced 3d radial ultrashort echo time mri. *Journal of Magnetic Resonance Imaging*, 47(5):1287–1297, 2018.