Interactive Deformation of Volume Images for Image Registration

Filip Malmberg[1,2]*, Robin Strand[1,2], and Joel Kullberg[2]

1) Centre for Image Analysis, Dept. of Information Technology, Uppsala University, Sweden.
2) Section of Radiology, Dept. of Surgical Sciences, Uppsala University, Sweden.

Abstract. Deformable image registration, the task of finding a spatial transformation that aligns two or more images with each other, is an important task in medical image analysis. To a large extend, research on image registration has been focused on automatic methods. This is in contrast to, e.g., image segmentation, where interactive semi-automatic methods are common. Here, we propose a method for interactive editing of a deformation field aligning two volume images. The method has been implemented in a software that allows the user to click and drag points in the deformed image to a new location, while smoothly deforming surrounding points. The method is fast enough to allow real-time display of the deformed volume image during user interaction, on standard hardware. The resulting tool is useful for initializing automatic methods, and to correct errors in automatically generated registrations.

Keywords: Deformable image registration, Medical image processing, Interactive registration

1 Introduction

Deformable image registration is an important task in medical image analysis [11] with applications in, e.g., interactive longitudinal analysis (e.g., interactive change detection and quantification) [9, 8] and interactive analysis of multimodal magnetic resonance images and ultrasound [10]. It can be defined as the task of finding a spatial transformation that aligns two or more images with each other. Here, we consider registration of two images – one referred to as the *source* image and the other referred to as the *target* image. In this case, the deformable registration problem consists of finding a deformation field that indicates, for each voxel in the target image, the location of the corresponding voxel in the source image.

For many image analysis tasks, e.g. image segmentation, semi-automatic interactive methods have proven to be a valuable complement to fully automatic methods [5, 6]. In contrast, research in image registration has to a large extent focused on development automatic methods [11]. Even basic interactive interventions, such as manual editing of deformation fields, remain largely unexplored.

* Send correspondence to `filip.malmberg@it.uu.se`.

Here, we present a method for interactive free-form editing of a deformation field aligning two volume images. This tool can be used for, e.g.,:

- Finding an approximate alignment to initialize automatic registration methods.
- Correcting errors in automatically generated registration results.

We have implemented the proposed method in a software that allows the user to load a source and a target image, optionally together with a deformation field generated by an automatic method, and quickly assess the alignment between the two images. The user can modify the deformation field interactively by applying standard affine transformations, i.e., translation, scaling and rotation. Additionally, a free-form deformation tool allows the user click on any point in the image and drag it to a new location, while smoothly deforming surrounding points.

To describe the technical problem solved by our proposed method, we note that from an implementation point of view, it is important to consider the direction of the transformation between the source and target images: A *forward* transform maps points in the source image to points in the target image, while a *backward* transform maps points in the target image to points in the source image. The main difference between the two approaches is the difficulty of the interpolation problem that needs to be solved to calculate the deformed source image. With a backward transform this calculation is straightforward; for every voxel in the target image, we can simply interpolate the value at the corresponding location in the source image. In contrast, when a forward transform is used, a scattered data interpolation problem needs to be solved [11]. For this reason most registration methods, including the method presented here, uses backward transforms.

Although a backward transform is used to represent the deformation internally, it still appears from the users perspective as if the source image is being deformed to match the target image. Therefore, an intuitive user interface requires us to formulate the editing operations available to the user in terms of forward transformations. To integrate this type of user interaction with the backward transform representation we need to compute, for every editing operation, the inverse of the transform given by the user. For affine transformations, the desired inverse transforms are easy to calculate analytically. For the proposed free-form deformation tool, however, this is not the case. Rather than attempting to invert the free-form deformation analytically, we calculate the inverse numerically, using a simple and fast iterative method. Our implementation demonstrates that this approach is fast enough to allow real-time manipulation of the deformation field on standard hardware.

2 Related work

As mentioned in the introduction, interactive methods for image registration is a relatively unexplored field. In this section, we briefly review related work in this area.

A method for interactive manual editing of volumetric deformation fields is described in a patent by Declerck [2]. The patent mentions the need for using inverse transforms, as discussed in Section 1, although no details are given on how to perform the inversion in practice.

An alternative method for interactive deformation of volume images was proposed by Gascon et al. [3]. In their method the deformation field represents a forward transformation, and is sampled on a tetrahedral mesh of lower resolution than the volume image itself. A fast rasterization algorithm is used to visualize the deformed volume. The authors report real-time performance for meshes consisting of up to 5000 tetrahedra.

A framework for semi-automatic registration was proposed by Kolesov et al. [4]. In this framework, existing automatic algorithms are considered as open loop system. Sparse interactions from the user guides the automatic algorithm, providing global anatomical knowledge.

3 Notation and Definitions

We consider registration between two volume images – one image referred to as the source or moving image, and the other referred to as the target or fixed image. Formally, we define a gray-scale volume image I as a mapping from $\mathbb{R}^3$ to the interval $[0, 1]$. Following the notation of Sotiras et al. [11], we denote the source image by S and the target image by T.

The two images are related by a transformation $W : \mathbb{R}^3 \to \mathbb{R}^3$. As stated in Section 1, this transformation is assumed to map the target image to the source image, i.e., a backward transform. The transformation at every position $\mathbf{x}$ is given as the addition of an identity transform with a displacement field $\mathbf{u}$, i.e.,

$$W(\mathbf{x}) = \mathbf{x} + \mathbf{u}(\mathbf{x}) \ . \tag{1}$$

In our implementation, we store $\mathbf{u}$ for each voxel in the target image. Note that by Equation 1, it is trivial to transform between $\mathbf{u}$ and W.

Let V and W be transformations. The *composition* $V \circ W$ of V and W is defined as

$$(V \circ W)(\mathbf{x}) = W(V(\mathbf{x})) \ . \tag{2}$$

A transformation W is *invertible* if there exists a transformation W^{-1}, called the *inverse* of W, such that $W \circ W^{-1}$ is the identity transform.

4 User Interface

In our implementation of the proposed method, the image data is presented to the user in three orthogonal views. The deformation tools can be used in any of the views, and the results are immediately updated in all views. The user can scroll between slices using the mouse wheel. It is also possible to click on a point

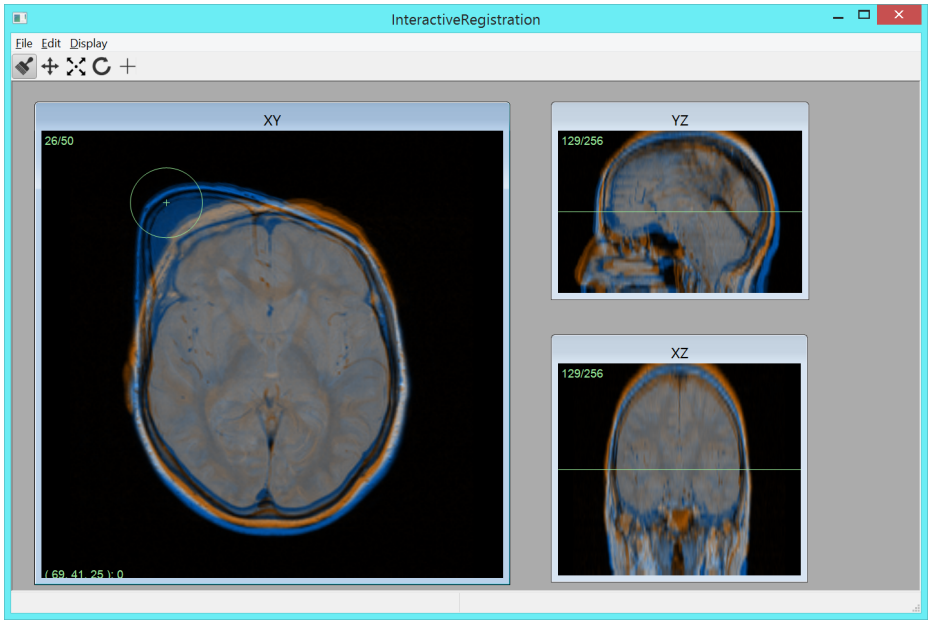

Fig. 1. A screen-shot of the user interface of our software for interactive editing of a deformation field aligning two images. The target image and the transformed source image are displayed using a colored overlay in three orthogonal views. The user can deform the source image by clicking and dragging in any of the views, as shown in the screen-shot.

in one of the views, and have the other views scroll to that point. A screen-shot of the user interface is shown in Figure 1.

To visualize both the target image and the transformed source image simultaneously, we overlay the two images using slightly different colors. Representing a screen color as an RGB-triple in $[0,1]^3$, the color of each display pixel is given by

$$S(W(\mathbf{x}))(1, 0.5, 0) + T(\mathbf{x})(0, 0.5, 1) \ . \tag{3}$$

This means that in areas where $S(W(\mathbf{x}))$ is darker than $T(\mathbf{x})$ the displayed image will have a blue tint, and in areas where $S(W(\mathbf{x}))$ is brighter than $T(\mathbf{x})$ the displayed image will have a red tint. When $S(W(\mathbf{x}))$ and $T(\mathbf{x})$ are equal, the displayed image will be gray-scale. Assuming that the images to be registered have similar intensities, this means that when a good registration has been found, $S(W(\mathbf{x}))$ and $T(\mathbf{x})$ should be almost equal over the entire volume. Even when this assumption is violated however, we have found that this method of visualization makes it quite easy to identify registration errors and determine how the current transformation should be changed to improve the registration.

5 Deformation tools

Our software implements a number of tools for manipulating the transformation between the source and target images. In this section, we give a detailed description of these tools. We provide standard affine transformations: translation, scaling and rotation. Additionally, we provide a free-form deformation tool, that allows the user to locally deform the image.

As stated in Section 1, our software represents the mapping between the source and target images as a backward transformation, that maps the target image to the source image. The operation of the proposed tools, however, is most easily described in terms of a forward transform, i.e., a mapping from the source image to the target image. Let us therefore pretend for a moment that the inverse W^{-1} of the current transformation exists and is known. When the user applies a tool, the current transformation W is replaced by a transformation W_{new}, defined in terms of its inverse:

$$W_{new}^{-1} = W^{-1} \circ N \ , \tag{4}$$

where the appended transform N represents the deformation indicated by the user.

In practice, we are of course not interested in finding the forward transform W_{new}^{-1} but rather the corresponding backward transform W_{new}. It is easy to verify that $W_{new} = N^{-1} \circ W$. In this expression, W is known but not N^{-1}. For affine transformations, N^{-1} can easily be determined analytically. Thus, the only remaining challenge for implementing affine transforms is to define an intuitive interface for specifying the transforms, e.g., mapping mouse actions to transforms. Such interfaces can defined in many different ways, and this issue will not be discussed further here. Instead, the remainder of this section will focus on the free-form deformation tool. For this tool, an analytical inverse is not available. Instead, we proposed to calculate the inverse numerically, using a simple and fast iterative method.

5.1 Free-form deformation

With this tool, the user can click on a point in the image to move it to a new location, while smoothly deforming surrounding points. When the user clicks on a point $\mathbf{p}$ and drags this point to a new location $\mathbf{q}$, the current transformation is updated according to Equation 4, with the appended transform $N_{\mathbf{p},\mathbf{q}}$ defined as

$$N_{\mathbf{p},\mathbf{q}}(\mathbf{x}) = \mathbf{x} + \alpha(|\mathbf{x} - \mathbf{p}|)(\mathbf{q} - \mathbf{p}) \ . \tag{5}$$

In the above equation, α is a radially decreasing function based on the *smoothstep* function commonly used in computer graphics [7]. Formally, α is defined as

$$\alpha(t) = \begin{cases} 0 \text{ if } t > R; \\ 1 \text{ if } t < 0; \\ 1 - (3(t/R)^2 - 2(t/R)^3) \text{ otherwise} \end{cases} \ , \tag{6}$$

where R is the *influence radius* of the tool. This function smoothly maps distances between 0 and R to values between 1 and 0, producing an "S"-shaped falloff curve.

The influence radius R is computed as

$$R = R_0 + 2|\mathbf{q} - \mathbf{p}| \ , \tag{7}$$

where R_0 is a user defined tool radius. In other words, the influence radius depends on the distance the user has dragged the mouse – the further the user moves a point, the larger the surrounding area that needs to be deformed in order to avoid having the deformation field "fold over" itself.

Rather than trying to find the inverse of $N_{\mathbf{p},\mathbf{q}}$ analytically, we use the numerical method proposed by Chen et al. [1] to calculate $N^{-1}_{\mathbf{p},\mathbf{q}}$ at every voxel. A proof that $N_{\mathbf{p},\mathbf{q}}$ is invertible and satisfies the conditions needed for the algorithm of Chen et al. to converge to the correct solution is given in the appendix of this manuscript.

6 Computational optimizations

To achieve real-time visualization of the new deformation field as the user drags the cursor, a number of computational optimizations are implemented:

- During dragging, the new deformation is only calculated in the currently displayed 2D slices. Only when the user releases the mouse cursor is the entire 3D deformation field updated. This approach was also suggested by Declerck [2].
- For all tools the computation of the updated transform W_{new}, including the inversion of the user specified transform, can be performed independently for all voxels. This makes it trivial to perform these computations in parallel, utilizing all available processors.
- For the free-form deformation tool, it holds that for all voxels outside the radius of influence $N_{\mathbf{p},\mathbf{q}}$, and therefore also $N^{-1}_{\mathbf{p},\mathbf{q}}$, equals the identity transform. Therefore, we only need to calculate $N^{-1}_{\mathbf{p},\mathbf{q}}$ for voxels within the radius of influence.

With these optimizations in place, we achieve real-time performance on a commodity laptop computer, even for large volume images.

7 Conclusion

We have described a method for manual editing of a deformation field aligning two volume images – a *target* image and a *source* image. The proposed method has been implemented in a software that allows the user to interactively modify the deformation field aligning two images by clicking and dragging points in the deformed source image. The proposed method is fast enough to allow real-time

display of the deformed source image during user interaction, on a commodity laptop computer.

The method presented here has some similarities to that described in a patent by Declerck [2]. The authors therefore advise anyone implementing the methods presented here to carefully review this patent to determine possible conflicts.

Appendix

We prove that the inverse deformation field $N^{-1}_{\mathbf{p},\mathbf{q}}$ exists, and can be calculated using the iterative algorithm proposed by Chen et al. [1]. Let $\mathbf{n}$ be the deformation field associated with the transformation $N_{\mathbf{p},\mathbf{q}}$, i.e.,

$$\mathbf{n}(\mathbf{x}) = \alpha(|\mathbf{x}-\mathbf{p}|)(\mathbf{q}-\mathbf{p}) \ . \tag{8}$$

According to [1], the simple fixed point iteration algorithm converges to the true inverse of $\mathbf{N}_{\mathbf{p},\mathbf{q}}$ if the following conditions are satisfied for all points $\mathbf{x}$,$\mathbf{y}$:

$$\exists M > 0 \text{ s.t. } |\mathbf{n}(\mathbf{x})| \leq M \ . \tag{9}$$

$$\exists a < 1 \text{ s.t. } |\mathbf{n}(\mathbf{x}) - \mathbf{n}(\mathbf{y})| \leq a|x-\mathbf{y}| \ . \tag{10}$$

The first condition says that $\mathbf{n}$ is bounded, and is trivially satisfied by, e.g., $M = |\mathbf{q}-\mathbf{p}|$. Next, we will show that the second condition is also satisfied. From Equation 8, it follows that

$$|\mathbf{n}(\mathbf{x}) - \mathbf{n}(\mathbf{y})| = |\mathbf{q}-\mathbf{p}||\alpha(|\mathbf{x}-\mathbf{p}|) - \alpha(|\mathbf{y}-\mathbf{p}|)| \ . \tag{11}$$

The function α is monotonically decreasing, i.e., $\alpha'(t) \leq 0$ for all t. Moreover, the derivative of α has a single minimum at $\alpha'(R/2) = -3/(2R)$. Therefore, it holds that

$$|\alpha(|\mathbf{x}-\mathbf{p}|) - \alpha(|\mathbf{y}-\mathbf{p}|)| = \tag{12}$$

$$\left| \int_{|\mathbf{y}-\mathbf{p}|}^{|\mathbf{x}-\mathbf{p}|} \alpha'(t)\,\mathrm{d}t \right| \leq \tag{13}$$

$$\left| \int_{|\mathbf{y}-\mathbf{p}|}^{|\mathbf{x}-\mathbf{p}|} \frac{3}{2R}\,\mathrm{d}t \right| = \tag{14}$$

$$\frac{3}{2R}||\mathbf{x}-\mathbf{p}| - |\mathbf{y}-\mathbf{p}|| \leq \tag{15}$$

$$\frac{3}{2R}|\mathbf{x}-\mathbf{y}| \tag{16}$$

Inserting this into Equation 11, we obtain

$$|\mathbf{n}(\mathbf{x}) - \mathbf{n}(\mathbf{y})| \leq \frac{3|\mathbf{q}-\mathbf{p}|}{2R}|\mathbf{x}-\mathbf{y}| \ . \tag{17}$$

Since $\frac{3|\mathbf{q}-\mathbf{p}|}{2R} = \frac{3|\mathbf{q}-\mathbf{p}|}{2(R_0+2|\mathbf{q}-\mathbf{p}|)} \leq \frac{3|\mathbf{q}-\mathbf{p}|}{4|\mathbf{q}-\mathbf{p}|} = 3/4 < 1$, the second condition holds.

References

1. Chen, M., Lu, W., Chen, Q., Ruchala, K.J., Olivera, G.H.: A simple fixed-point approach to invert a deformation field. Medical physics 35, 81 (2008)
2. Declerck, J.: Interactive manual deformable registration of images (Apr 24 2012), US Patent 8,165,425
3. Gascon, J., Espadero, J.M., Perez, A.G., Torres, R., Otaduy, M.A.: Fast deformation of volume data using tetrahedral mesh rasterization. In: Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation. pp. 181–185. ACM (2013)
4. Kolesov, I., Zhu, L., Karasev, P., Tannenbaum13, A.: A control framework for interactive deformable image registration
5. Malmberg, F.: Graph-based methods for interactive image segmentation (2011)
6. Malmberg, F., Nordenskjöld, R., Strand, R., Kullberg, J.: Smartpaint: a tool for interactive segmentation of medical volume images. Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization (ahead-of-print), 1–9 (2014)
7. Musgrave, F.K., Peachey, D., Perlin, K., Worley, S., Ebert, D.S.: Texturing and modeling: a procedural approach. Academic Press Professional, Inc. (1994)
8. Pohl, K.M., Konukoglu, E., Novellas, S., Ayache, N., Fedorov, A., Talos, I.F., Golby, A., Wells, W.M., Kikinis, R., Black, P.M.: A new metric for detecting change in slowly evolving brain tumors: validation in meningioma patients. Neurosurgery 68(1 Suppl OPERATIVE), 225 (2011)
9. Ridha, B.H., Barnes, J., Bartlett, J.W., Godbolt, A., Pepple, T., Rossor, M.N., Fox, N.C.: Tracking atrophy progression in familial alzheimer's disease: a serial MRI study. The Lancet Neurology 5(10), 828–834 (2006)
10. Roche, A., Pennec, X., Malandain, G., Ayache, N.: Rigid registration of 3-d ultrasound with MR images: a new approach combining intensity and gradient information. Medical Imaging, IEEE Transactions on 20(10), 1038–1049 (2001)
11. Sotiras, A., Davatzikos, C., Paragios, N.: Deformable medical image registration: A survey. Medical Imaging, IEEE Transactions on 32(7), 1153–1190 (2013)