

Definition of Volume Transformations for Volume Interaction

Thomas Schiemann and Karl Heinz Höhne

Institute of Mathematics and Computer Science in Medicine (IMDM),
University Hospital Hamburg-Eppendorf, Germany
e-mail: schiemann@uke.uni-hamburg.de

to be published in: Duncan, J., Gindi, G. (Eds.): *Information Processing in Medical Imaging (IPMI)*,
Lecture Notes in Computer Science, Springer Verlag, Heidelberg – New York, 1997

Abstract. Volume transformations of medical images play an important role for many applications such as registration of different modalities, mapping atlases onto clinical data, or simulation of surgical procedures. While registration and atlas mapping can for the major number of applications be performed without tight time constraints, it is essential for simulation systems that they allow real-time interaction. As any computational method in volumes is usually very time consuming, current approaches do mainly concentrate on surface manipulations instead of transforming the entire volume. This paper describes an approach, which overcomes this problem by first defining the volume manipulation on basis of surface models, which ensure real time performance, and in a second step the transformation is applied to the entire volume by interpolating the mapping parameters using scattered data interpolation methods.

1 Introduction

Geometric volume transformations often are a matter of interest when handling tomographic images in medicine. Currently, registration procedures are the main area of application and have already entered clinical routine. For the purpose of multi-modal registration, a global affine transformation of one of the volumes is usually sufficient [5, 15] if we assume that image distortions introduced by the scanning devices have been corrected in advance. The situation gets much more complex if we enter the field of atlas registration, which requires non-linear transformations for compensation of morphological differences between the atlas and the subject under consideration.

Simulation procedures build another large field where volume transformations can be used and cover a variety of medical applications:

- Simulation of classical surgical procedures: Open the operation field and shift, remove, replace, or sculpture structures.
- Simulation of endoscopic procedures: Blow up the area of investigation and deform the esophagus, the stomach, or the intestine as the endoscope is pushed through them.
- Simulation of growing morphological structures: Select a seed-area and let the included structures grow (for study of genesis), or insert a seed point for a lesion and see the effects of its growth.
- Assessment of inter-individual variation: Select corresponding structures of different individuals and map them onto each other or onto a common reference.

It is obvious that non-linear volume transformations are again required for all such procedures. These transformations usually lead to expensive computations, while for simulation of surgery or endoscopy real-time execution is required for enabling realistic feedback. Therefore current systems [1, 3, 8, 13] concentrate on transformation of surfaces only, which introduces a number of drawbacks, because all structures are hollow and can hence not represent reality.

In this paper we describe a method, which offers true volume transformations at any stage of a simulation procedure. While the definition of the transformation is still based on surface models in order to provide real-time interaction, the volume transformation is computed in the entire volume by interpolating the mapping parameters using scattered data interpolation methods.

2 Method

This section covers three different topics, which are necessary to apply volume transformations within an volume interaction environment [6], which is suitable for different applications. The volume interaction

environment allows segmentation [11] of volume objects from different tomographic sources (CT, MRI, histological slices) and exploration of these objects via comprehensive 3D visualization techniques [9, 14].

The first subsection describes, how to introduce surface models into the volume context, for example models of medical devices like the tip of an endoscope, or surface representations of morphological structures, which are contained in the volume. The different ways of interaction with the surface model within the volume context are described in second subsection, and the third subsection shows how to extend the definition of the volume transformation from the surface models to the entire volume.

2.1 Surface model

The first objective is to set up a surface model for real-time interaction in the context of the volume. The design of the model shall have several degrees of freedom such that it can be adapted to different applications. We are therefore using a model M with two layers of parameters. It can be formalized as

$$M(P_0, n_1, n_2, n_3; \Sigma, \alpha).$$

The first layer describes an orthogonal coordinate system with origin at P_0 and directions n_i . This coordinate system spans a hull with shape parameters Σ and α . The parameter Σ can take discrete values from a limited set, where each element corresponds to a fixed basic shape of the model (tab. 1). There are three classes of shapes corresponding to elemental solids \mathcal{E} (e.g. ellipsoid or cone), compositions \mathcal{C} of solids, which are usually approximations of medical devices, and triangular meshes \mathcal{T} for keeping surface representations of volume objects.

The parameter α is a tuple of variable length with boolean, integer, or float elements. Their number and type is depending on the value of Σ . Typical parameters, which are determined by α are extent, mesh width, number of elements composed (for class \mathcal{C}), or whether the solids are closed or open. The possible values for Σ , which have been realized so far are listed in table 1, and figure 1 shows the model's appearance for different values of Σ and α .

The parameter value $\Sigma = \mathcal{T}$ differs from all other parameters in the way, that the corresponding model is not determined by a closed analytical description, but instead uses a pure enumeration of surface elements. This option is necessary for storing arbitrary surface models, which are usually obtained from volume objects by triangulation. For this purpose we are using the marching cubes technique [7] extended by a method for reducing the number of triangles [16].

Σ	Shape	Σ	Shape	Σ	Shape	Σ	Shape
\mathcal{E}_1	Plane	\mathcal{E}_5	Star	\mathcal{C}_1	Arrow	\mathcal{T}	Triangular mesh
\mathcal{E}_2	Ellipsoid	\mathcal{E}_6	Torus	\mathcal{C}_2	Dumbbell		
\mathcal{E}_3	Cone	\mathcal{E}_7	Cylinder	\mathcal{C}_3	Ultrasound-sensor		
\mathcal{E}_4	Cube			\mathcal{C}_4	Head of endoscope		

Table 1. Values for the basic shape parameter Σ and corresponding appearances of the model. The values \mathcal{E}_i represent elemental solids, \mathcal{C}_i represent compositions of solids, and \mathcal{T} stands for triangular meshes of arbitrary shape.

For later use of the model it is essential that it can be displayed very fast. Therefore the model is also represented in the typical fashion of computer graphics as three lists of points $\{P_i\}$, lines $\{L_i\}$, and triangles $\{T_i\}$.

$\{P_i\}$ ($i = 1, \dots, n_P$, $P_i \in \mathbb{R}^3$) includes the hull-points after discretization of the model according to Σ and α .

$\{L_i\}$ ($i = 1, \dots, n_L$, $L_i \in \{1, \dots, n_P\}^2$) is a list of line-elements and determines, which two points from $\{P_i\}$ shall each be linked by a line, if the model is displayed as a wire-frame.

$\{T_i\}$ ($i = 1, \dots, n_T$, $T_i \in \{1, \dots, n_P\}^3$) is a list of triangle-elements and determines, which three points from $\{P_i\}$ shall each build a triangular patch, if the model is displayed with shaded surfaces.

In figure 1 the different appearances of the model are shown as wire-frames without any further context in order to enhance the method of construction. But as the model shall serve as an aid for definition of

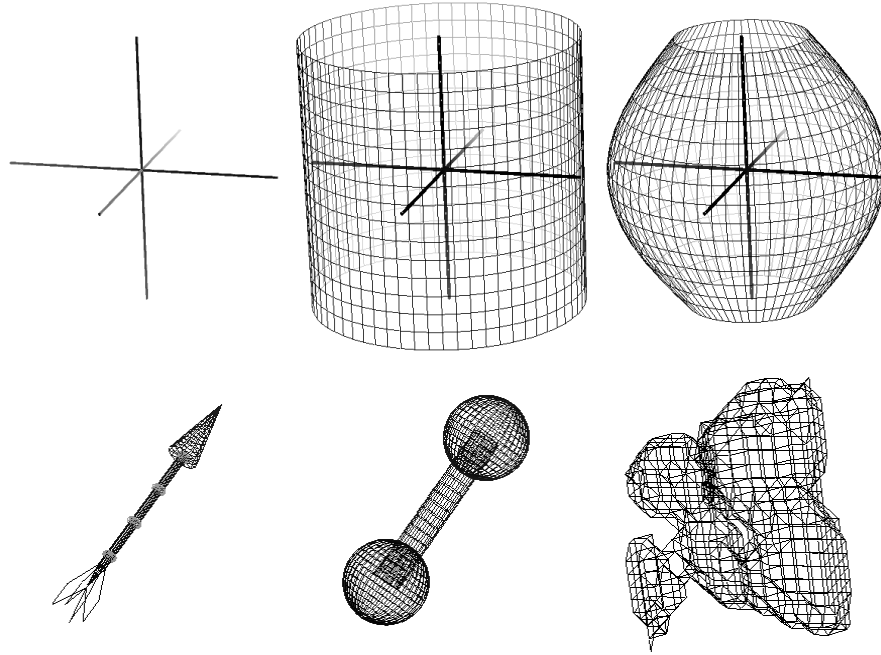


Fig. 1. Different appearances of the surface model. Top row: Coordinate system without hull and with two different cylindrically shaped hulls ($\Sigma = \mathcal{E}_7$) corresponding to two different settings for the parameter-vector α , which in this case determines the curvature. Bottom row: Two compositions of elemental solids (arrow $\Sigma = \mathcal{C}_1$, and dumbbell $\Sigma = \mathcal{C}_2$), and a surface representation of a part of the intestine ($\Sigma = \mathcal{T}$).

volume transformations, it is essential that it can be displayed in a natural way within the context of the volume. The rendering routines for the model are hence considering all geometric mapping parameters, which are used for any image (2D or 3D) obtained from the volume. The surface rendering uses the z-buffer of 3D images in order to be consistent regarding hidden surfaces. If the model is visualized with shaded surfaces, the illumination parameters are also equivalent to those of the volume based image (fig. 2 left). When the model is to be displayed on 2D slices of the volume, this can be performed by orthogonal 3D visualization (fig. 2 middle), or the list of triangles $\{T_i\}$ can be used to compute the intersection of the model with the slice for showing the outline of the intersection (fig. 2 right).

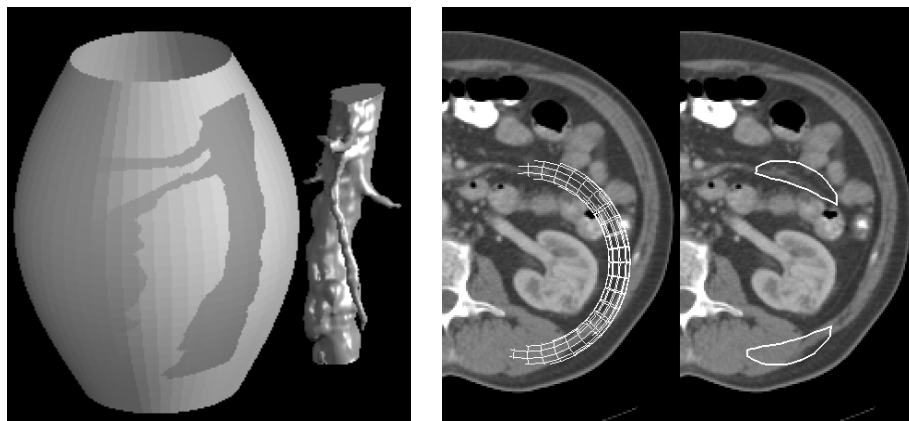


Fig. 2. Visualization of the surface model on images obtained from the volume. Left: Perspective 3D rendering within the context of volume objects. Hidden surfaces and illumination are computed on common parameters in order to achieve a consistent appearance of the model. Right: Display of the model on planes of the volume either by orthogonal 3D rendering (left) or outlining of the intersection area (right).

2.2 Interaction

The last subsection described the static construction of the surface model and how to adapt it to different circumstances. For its primary use of defining volume transformations, the second important question is how to modify the surface model intuitively and quickly. As we want to use the method on a common workstation, the only input device we can use is a standard mouse. Any interaction with the model must therefore be performed in the following way:

1. User action (= movement of mouse)
2. Derivation of a 3D modification of the model from the 2D movement
3. Computation of the transformation of the whole model
4. Output of the transformed model (on the screen)

Steps 1 and 4 do not need much discussion, as step 1 is a natural method for any interaction with standard computers, and step 4 means the 3D rendering procedures, which are based on well known computer graphics techniques [4]. Steps 2 and 3 are however critical for the goal of an intuitive and fast method.

Interaction with the surface model For transformation of the model according to the user action with the mouse there is a large set of modes, which can be grouped into three different classes (fig. 3):

- Global functions of different sub-classes are used for an overall general modification of the model: Polynomial functions up to second order are used to perform global movements (first order) or simple distortions (second order). More general mappings are used to model elastic deformations (e.g. via thin-plate-splines or warping schemes).
- Local displacements of the model's hull are expressed by displacements of the mesh points. Any mesh point can be selected for interactive displacement, and several points can be grouped for common modifications.
- Parameter modifications of the parameter vector α . For these modes mouse movement will result in modification of one or several elements of α , e.g. for changing the geometry of a model for a surgical device.

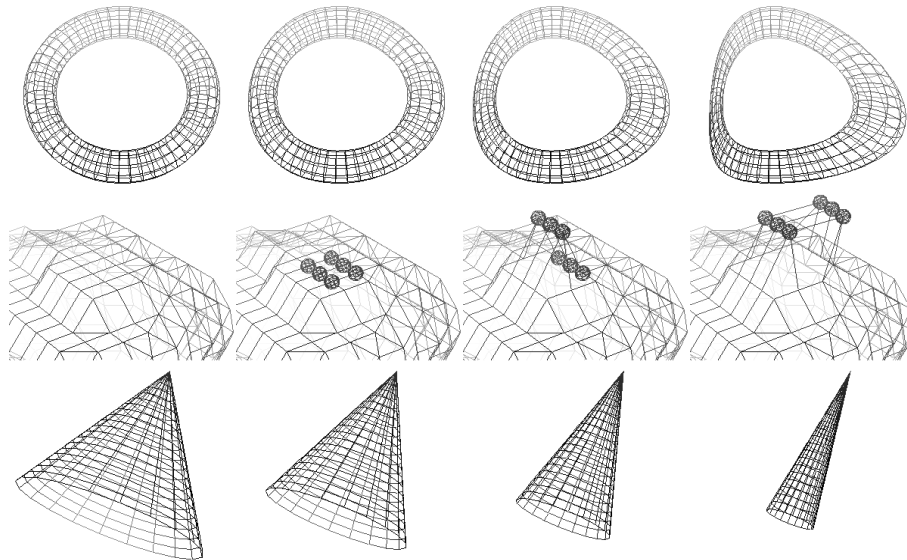


Fig. 3. Different classes of interaction, where each image represents a state during a continuous interaction. The model can be transformed via global functions (top), local modifications of the mesh points (middle), or changes of the parameter vector α (bottom).

The main problem, when having only a 2D mouse for definition of a 3D transformation is the lack of a third dimension. The common way to solve this problem is to use the keyboard for switching to

the missing dimension. For example, a free 3D rotation could in this way be defined by assigning the X-rotation to the up-down movement, the Y-rotation to the left-right movement, and the Z-rotation to the left-right movement, while a certain key is pressed. This is of course not intuitive and causes major difficulties for many users. Hence we prefer to use one of the following two methods for definition of 3D actions:

- The number of degrees of freedom for the model’s transformation can be reduced. In this way every dimension of the transformation can be assigned to one of the mouse directions. A 3D translation could e.g. be restricted to a plane or to the direction perpendicular to an object surface. The restricting elements can usually be determined from the context of the application.
- The 3D transformation could be defined indirectly by definition of an auxiliary item, which can be defined more easily. For example, a free rotation could be defined by selection of a point, towards which a certain axis of the model shall be rotated.

Computation of the transformation of the model The method described above ensures, that the interaction with the model can be performed as intuitive as possible with a 2D mouse. For real-time execution it is now critical, that the transformation can be applied with as little computational expense as possible. The detailed way of computing the transformation of the model depends on the actually selected transformation mode. In principle, there are the following two main steps, which have to be performed:

1. Transformation of all hull points according to the transformation.
2. Transformation of all hull points from object space into the space of the image, in which the model is to be displayed. This means multiplication of every point with a homogeneous (4×4) -matrix.

As there is a large variety of possible volume transformations to be defined, it is impossible to make a clear general statement on the computational costs. For affine transformations, the computation is very cheap, because for the transformation itself it is sufficient to transform the coordinate system of the model only, resulting in a new matrix, which can be combined with the image matrix of the second step. For general non-linear transformations, the expense depends directly on the complexity of the active transformation mode: While displacing groups of mesh points is cheap, because only a local small portion of the model needs to be transformed, polynomial mappings require by far more computation time, and parameter modifications may be expensive if the model’s geometry is complex.

One possible way of saving time, which can be taken under any circumstances, is to keep the complexity regarding the model’s number of surface elements as low as possible. Especially during the initial state of an application it is usually acceptable to work with models in lower resolution.

2.3 Volume interpolation

The model described in the previous two subsections offers the functionality of interactively defining volume transformations in a very general fashion. As our main intention is to obtain true volume transformations, the transformation has to be spread from the surfaces of the model to every single voxel in the volume in order to assign a displacement vector to every discrete point. Techniques, which could be used for this are known as scattered data interpolation methods, because the given data points are not well ordered for parameter-guided standard interpolation. In general, we have to deal with the following interpolation problem:

We have a set of source points $\{A_i\}$, which are taken from the surface model in the original state, and a set of target point $\{Z_i\}$, which are taken from the transformed surface model. The problem is to find an at least steady function

$$f(P) : \mathbb{R}^3 \rightarrow \mathbb{R}^3 \quad \text{with } f(A_i) = Z_i \text{ for all } i.$$

We have investigated two different interpolation schemes for this problem:

Interpolation with weighted local interpolators The most obvious approach is to construct f as a weighted sum of locally interpolating functions f_i :

$$f(P) = \sum_{i=1}^n \omega_i(P) f_i(P),$$

where

$$f_i : \mathbb{R}^3 \rightarrow \mathbb{R}^3, \quad f_i(A_i) = Z_i \text{ for all } i,$$

and

$$\omega_i : \mathbb{R}^3 \rightarrow \mathbb{R}$$

are normalized weight functions with

$$\omega_i(A_i) = 1 \text{ for all } i, \quad \omega_i(A_j) = 0 \text{ for all } i \neq j,$$

$$\omega_i(P) \geq 0 \text{ and } \sum_{i=1}^n \omega_i(P) = 1 \text{ for all } P \in \mathbb{R}^3.$$

This approach has originally been proposed in [12]. For our method we followed a choice for f_i and ω_i , which is described in [10]:

$$f_i(P) = Z_i + L(P - A_i),$$

where $L : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ is a linear functional, which minimizes a global error function. The weights are chosen as

$$\omega_i(P) = \frac{\sigma_i(P)}{\sum_{j=1}^n \sigma_j(P)}$$

with

$$\sigma_i(P) = \frac{1}{\|P - A_i\|^2}$$

The construction of this interpolation scheme follows basically geometric ideas. It is therefore not surprising, that artifacts are occurring, if the range of displacements is increasing. This problem does not occur with the second approach:

Interpolation with thin-plate splines Interpolation with thin-plate splines has successfully been introduced to 2D image processing [2] and can directly be generalized for 3D applications. The construction of thin-plate splines origins from mechanics, which results in much better behavior for even strong displacements compared to interpolation with linear local interpolators. The interpolating function f is constructed as a linear combination of functions of the form

$$U(r) = r^2 \log r^2 \quad (r \in \mathbb{R}, r > 0),$$

and an affine correction term:

$$f(P) = \sum_{i=1}^n \Omega_i U(\|P - A_i\|) + L(P) + Q_0,$$

where $\Omega_i, Q_0 \in \mathbb{R}^3$, and $L : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ is again a linear functional. The function U is a fundamental solution of the biharmonic equation $\Delta^2 U = 0$. The Vectors Ω_i and Q_0 and the coefficients of L are determined by solving the following system of linear vector equations, which formalize the interpolation condition and normalization:

$$\begin{aligned} \sum_{i=1}^n \Omega_i U(\|A_j - A_i\|) + L(A_j) + Q_0 &= Z_j \quad j = 1, \dots, n \\ \sum_{i=1}^n \Omega_i &= 0 \\ \sum_{i=1}^n \Omega_i x_i &= 0 \\ \sum_{i=1}^n \Omega_i y_i &= 0 \\ \sum_{i=1}^n \Omega_i z_i &= 0, \quad \text{where } A_i = (x_i, y_i, z_i)^T \end{aligned}$$

Volume resampling The remaining task is now, to actually perform the volume transformation by resampling the new volume. As the integer-coordinates of the input volume are mapped onto float-coordinates in the output-volume, a single input voxel will usually be spread on several output voxels, or several input voxels will share one output voxel (for shrinking transformations). Different resampling solutions have been published for this problem [17]. In the current phase of this project we are using a straight-forward approach, which transforms the 6 neighbors of every voxel and fills the volume spanned by these neighbors in the output volume with the value of the central voxel. This filling is weighted according to the percentage, which is covered in the respective output voxel.

3 Results

The proposed methods have been implemented in ANSI-C on standard UNIX-workstations (e.g. DEC alpha). They have been embedded in a comprehensive medical volume interaction system (VOXEL-MAN) and can thus be invoked in conjunction with many different applications. The following subsections describe two different areas, in which the method can be applied.

3.1 Normalization of Shape

For comparison of different individuals it is useful to transform different volumes into a common general shape. Such procedures are frequently proposed for brain studies, especially in the field of computerized atlases.

If the brain is assumed to be roughly ball shaped, different brains can be transformed to spheres and can hence be compared in their new normalized shape. For this purpose we are using the described model in star-shape ($\Sigma = \mathcal{E}_5$). The center of the star is located in the center of the brain, and every ray gets the length, which makes it touch the outer surface. The rays have a certain width, which ensures that they are long enough to leave the sulci. Figure 4 shows a 3D image of a brain with the corresponding ray ends each marked by small balls. The ray end points form the set $\{A_i\}$ of source points.

We then take a star with equal number of rays but uniform length of 100 voxels and take their end points as the set of target points $\{Z_i\}$. After computation of the thin-plate spline the volume is transformed and we end up with a ball shaped brain as shown in figure 5.

While this application does in general not require real-time definition of the mapping, our approach has the advantage, that corrections on the set of source points $\{A_i\}$ can easily and quickly be performed. Such corrections might e.g. be necessary when pathological changes in morphology are present.

3.2 Simulation of surgical procedures

For simulation of surgical procedures the described method offers the choice of two different ways of interaction depending on the question, whether the surface model represents the surgical device or the examined structure. We illustrate this application with the example of a needle examination of a kidney.

The first principle is to use the surface model as a representative for the needle by using the cylindrical shape ($\Sigma = \mathcal{E}_7$) with the interaction modes of free rotation and translation along a fixed direction (determined by the rotation). Hence the needle can be rotated and shifted along the specified axis within the context of the volume objects (fig. 6 left). The mesh points around the tip of the needle are "sensitive" and recognize, when they hit volume structures. Hence two sets of points can be obtained after having shifted the needle into the kidney: Those points on the boundary of the kidney, which were hit by the sensitive grid points when the tip of the needle penetrated the kidney, form the source points $\{A_i\}$ for the coming volume transformation. The current positions of the sensitive needle points form the target points $\{Z_i\}$. After computation of the volume transformation we obtain images like in figure 7, which show the deformed kidney.

The second principle is to fill the surface model with a triangular description of the considered kidney ($\Sigma = \mathcal{T}$, fig. 6 right). The interaction mode is set to translation perpendicular to the last hit surface element. When now moving the mouse, we immediately see a feedback of the deformed kidney surface, because for the surface model the transformation can be computed very fast. Again the computation of the true volume transformation can be released at any time of the procedure, e.g. in order to assess internal structures (fig. 7).

While both principles have the advantage of real-time interactivity, they both also have the disadvantage of showing only one aspect of the procedure at a time - either the surgical instrument or the considered structure. This problem can be handled by switching from one principle to the other, which can be done at any stage of the procedure.

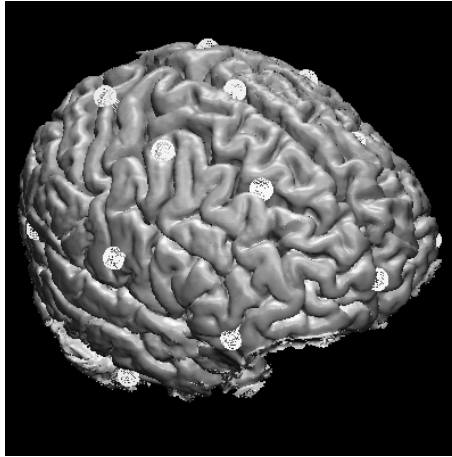


Fig. 4. 3D image of a brain obtained from MRI. The markers show the end points of the rays corresponding to the star shaped model.

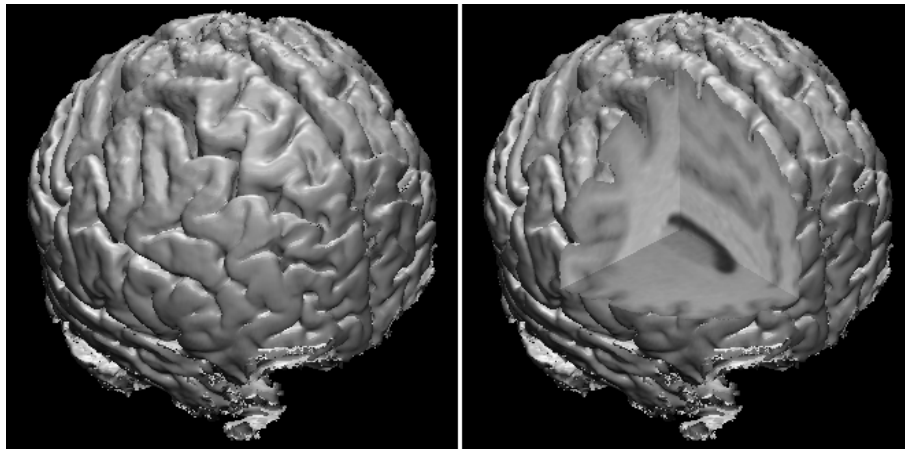


Fig. 5. Brain after volume transformation into a uniform shape. On the right a portion of the brain has been removed for showing the effect of true volume instead of surface transformation.

4 Conclusion

We have described a framework for interactive definition and execution of volume transformations. As it is not based on any conditions on the type of the considered volumes or their contents, the method has a very large range of possible applications. This is one of the major differences to existing approaches, which are usually suited to single questions. Another difference is the fact, that our approach results in true volume transformations, which are important for a comprehensive assessment of the structures contained in the volume.

It is clear that such a complex system has many points, which are going to be subject for improvement:

- The set of interaction modes has to be expanded towards algorithms for mechanical simulations like finite element methods. This will improve the realism of the local deformations, which are currently based on mesh point displacements.
- While the interaction times are in the order of real-time (for model complexity within a reasonable range), the resampling procedure for the volume transformation has to be performed more efficiently in order to apply the true volume transformation more frequently during the procedure.
- The surface model will be subject for automatic procedures for adaption to different applicational contexts, e.g. for automatically finding an optimal laparoscopic path. While there is no doubt that many applications would benefit from such algorithms, this also means to leave the scope of very general applicability.

The two applications shown in the previous section are only first attempts in showing the practicability

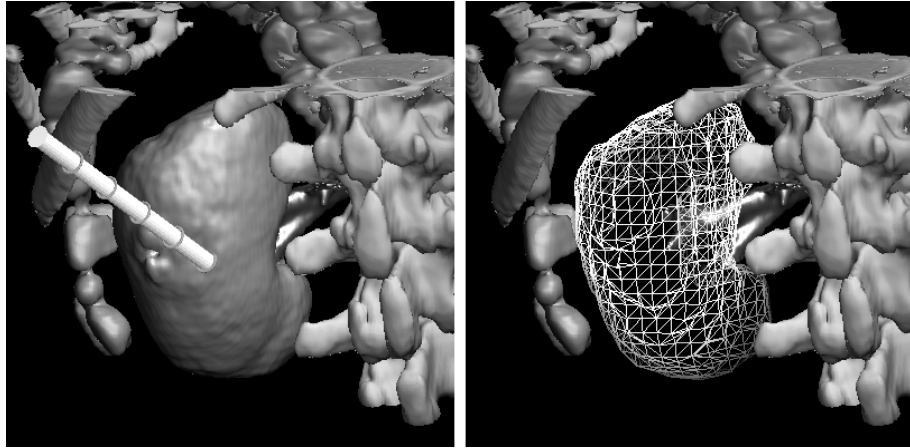


Fig. 6. View of the left kidney and surrounding structures (bone, intestine) obtained from a spiral CT dataset of an upper abdomen. The surface model mimics a needle and can be moved within the volume scene (left). When the needle touches the kidney, the deformation can not be visualized immediately, because the kidney is present in its volume representation. This situation changes, if the kidney is replaced with the corresponding surface model (right) in order to allow real-time manipulations. (The kidney is displayed as a wire-frame only instead of shaded surfaces in order to enhance its different representation compared to the volume objects).

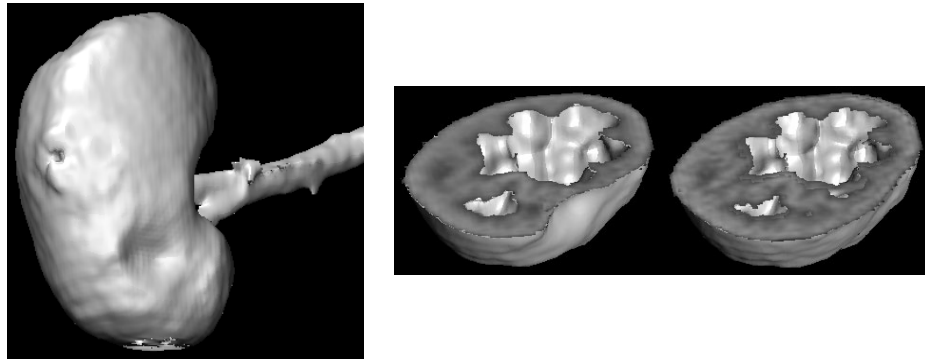


Fig. 7. Volume based renderings of the kidney after computation of the true volume transformation defined in figure 6. The imprint of the needle is clearly visible in the lower section of the organ. The cuts through the kidney show the treatment's effects on internal structures. For comparison the very right image shows the corresponding structure without deformation.

of the method. Other possible applications could only briefly be mentioned and will - as the two others - be subject of further developments.

5 Acknowledgments

We are grateful to all members of our department, who have supported this work. The described methods have been realized within the VOXEL-MAN framework, which is the main project of our research group: Bernhard Pflessner, Andreas Pommert, Kay Priesmeyer, Martin Riemer, Rainer Schubert, and Ulf Tiede.

References

1. Ayache, N.: Medical computer vision, virtual reality and robotics. *Image and Vision Computing* 13, 4 (1995), 295–313.
2. Bookstein, F. L.: *Morphometric tools for landmark data*. Cambridge University Press, Cambridge, 1991. (ISBN 0-521-38385-4).
3. Cotin, S., Delingette, H., Ayache, N.: Real Time Volumetric Deformable Models for Surgery Simulation. In Höhne, K. H., Kikinis, R. (Eds.): *Visualization in Biomedical Computing 1996*. Lecture Notes in Computer Science 1131, Springer-Verlag, Heidelberg, 1996, 535–540.
4. Glaeser, G.: *Fast Algorithms for 3D-Graphics*. Springer-Verlag, Heidelberg, 1995. (ISBN 3-540-94288-2).

5. Hill, D. L., Studholme, C., Hawkes, D. J.: Voxel Similarity Measures for Automated Image Registration. In Robb, R. A. (Ed.): *Visualization in Biomedical Computing 1994, Proc. SPIE 2359*. Rochester, MN, 1994, 205–216.
6. Höhne, K. H., Pflesser, B., Pommert, A., Riemer, M., Schiemann, T., Schubert, R., Tiede, U.: A new representation of knowledge concerning human anatomy and function. *Nature Med.* 1, 6 (1995), 506–511.
7. Lorensen, W. E., Cline, H. E.: Marching cubes: A high resolution 3D surface construction algorithm. *Comput. Graphics* 21, 4 (1987), 163–169.
8. MacDonald, D., Avis, D., Evans, A. C.: Multiple Surface Identification and Matching in Magnetic Resonance Images. In Robb, R. A. (Ed.): *Visualization in Biomedical Computing 1994, Proc. SPIE 2359*. Rochester, MN, 1994, 160–169.
9. Pommert, A., Bomans, M., Höhne, K. H.: Volume visualization in magnetic resonance angiography. *IEEE Comput. Graphics Appl.* 12, 5 (1992), 12–13.
10. Ruprecht, D., Müller, H.: Free form deformations with scattered data interpolation methods. In Farin, G. et al. (Eds.): *Geometric Modelling (Computing Suppl. 8)*. Springer-Verlag, 1993, 267–281.
11. Schiemann, T., Bomans, M., Tiede, U., Höhne, K. H.: Interactive 3D-segmentation. In Robb, R. A. (Ed.): *Visualization in Biomedical Computing II, Proc. SPIE 1808*. Chapel Hill, NC, 1992, 376–383.
12. Shepard, D.: A two-dimensional interpolation function for irregularly spaced data. In *Proc. of the 23rd National Conference of the ACM*. ACM-Press, New York, 1968, 517–524.
13. Thompson, P., Toga, A. W.: A Surface-Based Technique for Warping Three-Dimensional Images of the Brain. *IEEE Trans. Med. Imaging* 15, 4 (1996), 402–417.
14. Tiede, U., Schiemann, T., Höhne, K. H.: Visualizing the Visible Human. *IEEE Comput. Graphics Appl.* 16, 1 (1996), 7–9.
15. van den Elsen, P. A., Pol, E.-J. D., Viergever, M. A.: Medical image matching — A review with classification. *IEEE Engng. Med. Biol.* 12, 1 (1993), 26–39.
16. Wilmer, F., Tiede, U., Höhne, K. H.: Reduktion der Oberflächenbeschreibung triangulierter Oberflächen durch Anpassung an die Objektform. In Fuchs, S., Hoffmann, R. (Eds.): *Mustererkennung 1992, Proc. 14. DAGM-Symposium*, Springer-Verlag, Berlin, 1992, 430–436.
17. Wolberg, G.: *Digital Image Warping*. IEEE Computer Society Press, Los Alamitos, CA, 1990.