applied sciences

MDPI

Article

Hands-On Deformation of Volumetric Anatomical Images on a Touchscreen

Rosell Torres, Alejandro Rodríguez and Miguel Otaduy *

Universidad Rey Juan Carlos, Dept. of Computer Sience, 28933 Móstoles-Madrid, Spain; rose2torres@gmail.com (R.T.); alejandroRA88@gmail.com (A.R.)
* Correspondence: miguel.otaduy@urjc.es

Abstract: In this work, we propose a novel metaphor to interact with volumetric anatomical images, e.g., magnetic resonance imaging or computed tomography scans. Beyond simple visual inspection, we empower users to reach the visible anatomical elements directly with their hands, and then move and deform them through natural gestures, while respecting the mechanical behavior of the underlying anatomy. This interaction metaphor relies on novel technical methods that address three major challenges: selection of anatomical elements in volumetric images, mapping of 2D manipulation gestures to 3D transformations, and real-time deformation of the volumetric images. All components of the interaction metaphor have been designed to capture the user's intent in an intuitive manner, solving the mapping from the 2D touchscreen to the visible elements of the 3D volume. As a result, users have the ability to interact with medical volume images much like they would do with physical anatomy, directly with their hands.

Keywords: tangible interaction; medical images; deformable models

Citation: Torres, R.; Rodríguez, A.; Otaduy, M. Hands-On Deformation of Volumetric Anatomical Images on a Touchscreen. *Appl. Sci.* **2021**, *1*, 0. https://doi.org/

Academic Editor: Qi Sun, Ana Serrano, Diego Gutierrez, Enrico Vezzetti

Received: 3 September 2021
Accepted: 8 October 2021
Published:

1. Introduction

Even though the human body is composed mostly of flexible, dynamic tissue, medical volume images—such as magnetic resonance imaging (MRI) or computed tomography (CT) scans—are displayed as static, non-interactive datasets. Clinicians would enjoy new opportunities if they had the ability to inspect and interact with medical volume images in a tangible manner, much like they would do with a physical body had they access to internal anatomy in a harmless, unconstrained manner. This could be implemented through a tangible interaction tool where users may manipulate and deform anatomical elements naturally with their hands, and thus intuitively scrutinize shape, appearance and mechanical properties, as well as the connections and contacts between elements. Such tool can facilitate clinical tasks such as pre-operative planning, by largely simplifying usability and interaction.

The classical approach to simulating anatomical deformation is to (i) segment anatomical elements, (ii) mesh these elements separately, and (iii) perform a contact mechanics simulation between the anatomical elements. Selection, manipulation and deformation are all executed on the segmented anatomy. Therefore, the ability to select, manipulate and deform individual anatomical elements is limited by the accuracy and robustness of segmentation. This is to date an extremely hard and laborious problem, and it hinders largely the resolution and applicability of anatomical deformation.

We propose instead to carry out all interactions (i.e., selection, manipulation and deformation) directly on the input medical images, and thus retain the resolution of the original data. However, this is not free of challenges. The dense three-dimensional nature of medical volume images poses inherent difficulties for the tangible manipulation of the underlying anatomy. Display and interaction devices are still predominantly two-dimensional, and appropriate mappings must be defined between 2D and 3D to enable direct manipulation of volume datasets. For visualization, volume rendering provides

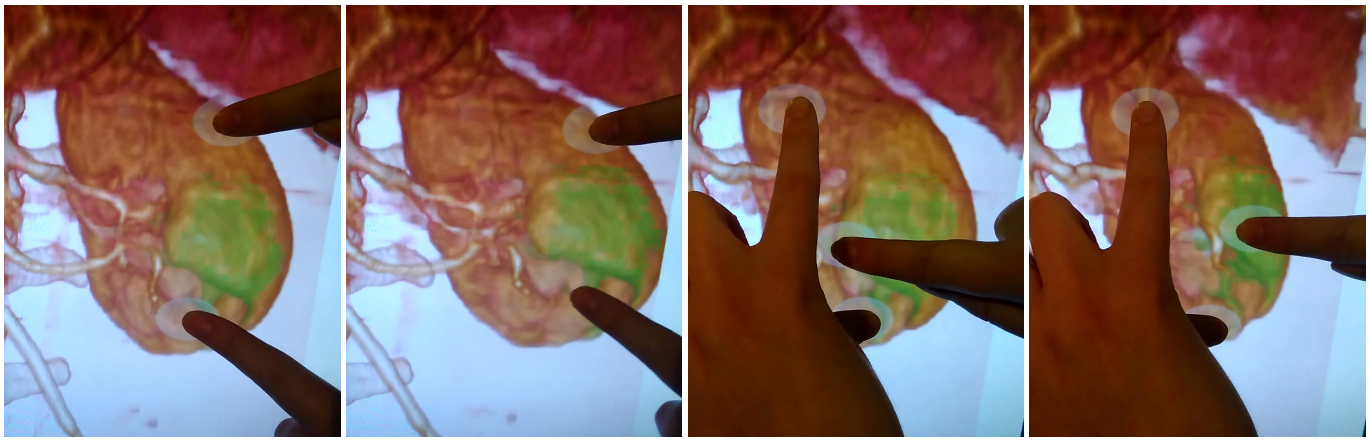

Figure 1. Intuitive manipulation and deformation of a kidney in a CT-scan. Our method allows users to directly touch and interact with anatomical elements in a medical volume image much like they would do in the real world, through natural manual actions. We use a rigid handle metaphor, highlighted in green in the images, to map user actions into the manipulation of the volume. The first two images show in-plane manipulation with two fingers, and the last two images show out-of-plane manipulation with three fingers.

a successful solution for the definition of visible surfaces within the dense 3D dataset. For deformation, on the other hand, there is no effective method for conveying 3D selection and manipulation of volume data in a natural and intuitive manner using a 2D interface.

In this work, we propose a metaphor of natural interaction with medical volume images through tangible direct interaction on a touchscreen. We let users deform the anatomy by touching it directly and performing manipulation operations with their fingers, much like they would do with real anatomy. A key ingredient of our method is to map the 2D touchscreen interaction to 3D volumetric deformation through a rigid handle metaphor. Then, we decompose the interaction process into three operations: selection of handles, mapping of handle transformations, and computation of handle-driven deformations.

First, in Section 4, we describe an intuitive, user-guided method for the selection of anatomy elements as volumetric handles. We use an intuitive mapping to define selection parameters directly on the touchscreen, based on visible anatomy elements. Furthermore, we apply a region growing method based on visual settings to segment an anatomical element and conform a handle.

Second, in Section 5, we describe an optimization-based method for the computation of rigid handle transformations. We define intuitive cues for direct-touch manipulation of the handles on a touchscreen, and we compute corresponding handle transformations by solving efficiently a closed-form optimization problem.

Third, in Section 6, we summarize how we drive the deformation of the full anatomy from the transformation of the handle. To this end, we adapt a corotational coarsening deformation method [1] that solves an elastostatic deformation problem on a coarse finite-element mesh, while respecting the material distribution and boundary conditions imposed on a fine mesh, and then applies the deformation to the embedded medical image through volume resampling [2]. We extend the original corotational coarsening deformation method by supporting additional types of high-resolution boundary conditions, to accommodate handle-based manipulation.

In our results, such as Figure 1, we show multiple examples where the user deforms and inspects the anatomy in a medical volume image. We demonstrate that our method is applicable to a wide range of anatomical elements, from large organs to fine elements such as blood vessels. The user performs selection and manipulation actions in a natural way, with smooth transitions. Immediate tangible and visual feedback makes interaction and inspection intuitive and effective.

We believe that our methods can serve several clinical applications. The most evident one is pre-operative planning, but our conversations with clinical experts also point to potential use for exploration and diagnosis. At present, the proposed system barely achieves

interactive performance, due to the high computational cost of soft-tissue deformation and volume resampling. While the current performance is sufficient for demonstrating novel interaction features, it is not yet acceptable for practical application. Therefore, our work currently lacks user evaluation, which is an utmost requirement for full validation of the proposed methods.

2. Related Work

Our work builds on previous methods that address diverse challenges on 3D interaction, volume exploration, and volume deformation. More than twenty years ago, Hinckley et al. [3] identified the potential benefits of tangible interaction on touchscreens for the exploration of medical volume images. Visualization and interaction should go hand in hand to overcome the challenges for exploring and manipulating complex datasets [4].

2.1. 3D Interaction on Touchscreens

Direct 3D interaction using 2D cues on touchscreens has been studied by multiple authors. The different proposed methods aim for different goals, such as maximizing the intuitiveness of the interaction metaphors, maximizing user performance, or minimizing ambiguity. The study by López et al. [5] investigates the combination of various interaction and navigation metaphors. A recent survey by Mendes et al. [6] discusses the pros and cons of multiple methods, and goes beyond 2D interaction to discuss also immersive 3D interaction.

Hancock et al. [7] solved the ambiguity problems for 6-DoF interaction by factoring the full 6-DoF interaction into simpler actions. Cohé et al. [8] designed a simple widget that supports 3D transformations naturally through touchscreen actions. Reisman et al. [9] computed 3D transformations by formulating an optimization problem, such that the 2D projections of the 3D points touched by the user follow the actions of the fingers as close as possible. Liu et al. [10], on the other hand, tried to limit 2D cues to just two fingers, to simplify interaction, and disambiguate the intended 3D transformation based on the actions of the fingers.

Given the diversity of methods, several studies have compared multiple techniques. Martinet et al. [11] concluded that user performance was improved when separating translation and rotation, as interactions were simpler and more accurate. Klein et al. [12] focused their study on the analysis of several interaction techniques for exploration of 3D data in scientific visualization.

As our goal is to maximize the similarity with real-world manipulation of anatomy, we follow the optimization formulation by Reisman et al. [9] for direct manipulation. This method enables interaction metaphors that resemble best the grasping actions that humans carry out in 3D, and it requires no widgets. Moreover, the method of Reisman et al. enables a seamless transition from selection to deformation, all in a similar way to real life interaction. We opt for this natural concatenation of actions, at the expense of the possibly improved user performance of widget-based manipulation. However, we modify the optimization approach of Reisman et al., by formulating a simpler linear problem with a closed-form solution. In a similar spirit to early approaches for interactive camera control [13], we linearize the relationship between 2D and 3D actions.

In our work, we assume that the touchscreen is large enough for uncluttered interaction. Therefore, small screens, such as those in mobile phones, are not well suited for our method. Dedicated methods have been designed for effective interaction on small screens [14]. Interaction with large models on mobile devices often also requires network architectures for adaptive rendering [15].

Intuitive interaction through touchscreens is not the only alternative for natural 3D interaction. One approach is to provide colocated interaction thanks to a see-through display [16], or to combine colocation and tangible actions for interaction on a visualization workbench [17]. Another solution, as proposed by Marton et al. [18], is indirect interaction, where the user may control the camera on a separate touch-enabled surface, while the

synchorized actions are displayed on a larger projection system. Coffey et al. [19] explore the idea of visualizing a world-scale representation of the data, while interacting with a miniature-scale representation of the data. Bruder et al. [20] have compared 2D touchscreen interaction vs. 3D mid-air interaction on stereoscopic displays, showing that the 2D option is more efficient close to the screen, whereas the 3D option is more efficient for targets further away from the screen. Besançon et al. [21] have also evaluated various interaction methods, including comparisons of indirect vs. tactile and tangible methods.

2.2. Exploration and Selection of Volume Data

The exploration of volume data has typically been addressed from two major angles. One is the definition of rendering parameters, and the other is the definition of views.

Kniss et al. [22] designed complex transfer functions for volume rendering, and along with them, direct manipulation widgets which make specifying transfer functions intuitive and convenient. Continuing this line of work, others have also explored user interfaces for defining complex transfer functions [23], or user interfaces for choosing more generic rendering parameters [24].

As mentioned earlier, one way to optimize the exploration of the volume is to find best views. Weiskopf et al. [25] designed methods to combine complex clipping planes. Other works employ various visualization metaphors for illustrative visualization of medical datasets [26], or connect touchscreen-based interaction with the exploration of volume data [27]. Zorzal et al. [28] propose an augmented reality 3D display for segmentation operations, but we limit ourselves to 2D touchscreens. In contrast to these works, our approach is not limited to the exploration of static volume data. Instead, we design manipulation methods that allow the user to explore the deformation of the volume data.

One of our proposed techniques addresses the selection problem. To this end, Wiebel et al. [29] have investigated the problem of correctly picking the visible isosurface. We rely on state-of-the-art solutions to this problem, but we further address the challenge of growing the picked selection to cover a visible volume of interest. Owada et al. [30] try to select a region of interest by drawing a stroke, followed by an automated region segmentation. Our approach is similar in the sense that it combines a cue and automated segmentation, but our cue resembles a more natural grab action. Direct selection methods can be augmented with topological information through graph optimization approaches, to avoid the depth discontinuities that appear in simple ray casting. This idea has been applied to visible line selection by Wiebel et al. [31], and visible surface patch selection by Stoppel et al. [32]. In our work, we are concerned with volume selection beyond visible geometry, and an interesting direction for future work is to accommodate graph optimization approaches into this volume selection. Volume selection can also be addressed by controlling simple volumetric primitives with Boolean operations. Stoppel et al. [33] applied this idea to create illustration widgets. Besançon et al. [34], on the other hand, proposed to draw 2D shapes and then explicitly control how they extend into 3D. They also discussed an extensive taxonomy of selection methods, which we point to for further reading on the topic.

2.3. Volume Image Deformation

The deformation of volume images has been addressed using two major families of methods. One family is based on volume rendering with 3D texture mapping, and encoding the deformation as the mapping function. The other family is based on resampling the original volume into a new regular grid using the deformation.

Early methods performed volume rendering using slicing planes, and transformed voxels to the original dataset using GPU-based 3D texture mapping [35]. The various methods differed in terms of the approach to encode the deformation, such as trilinear interpolation based on free-form deformation [36], volumetric skinning of articulated motions [37], or scattered data interpolation [38]. This approach has been used within procedural manipulation operations on the volume, such as cutaway, for illustration

purposes [39]. As a variant of the mapping approach, Georgii and Westermann [40] presented a method for volume rendering high-resolution irregular tetrahedral meshes, which enables arbitrary encoding of deformations within the mesh. Correa et al. [41] also used a mapping function to render deformable volume images. They designed displacement fields with editing capabilities, subject to feature preservation constraints, to ease illustrative visualization. Kretschmer et al. [42], on the other hand, reformatted volume images into planar views to aid in diagnostic explorations.

Thanks to the increase of performance of graphics hardware, real-time resampling of the full volume image is now possible. This enables the use of richer volume rendering methods, while handling the deformation as part of a resampling step. The approach was first used with deformations defined on coarse tetrahedral meshes and voxel-level parallelization [43], and it was then accelerated through tetrahedron-level parallelization [2]. Further extensions have considered different deformation models, such as a chain-mail algorithm [44], the corotational coarsening model used in our work [1], or the addition of respiratory models [45].

As discussed in the introduction, the classical approach to simulating anatomical deformation is to segment and mesh the individual anatomical elements. Selection and manipulation actions are then largely simplified, as they can leverage the explicit boundaries of the elements. The simulation of deformations is solved using state-of-the-art contact mechanics formulations, and there are even open frameworks that facilitate this task [46]. A sample of diverse representative medical applications can be found in the review of Talbot et al. [47]. The classical approach is largely complementary to our work, as we can leverage segmented data when available. However, in the classical approach all relevant anatomical elements must be individually segmented. Due to the complexity of this task, the approach is either rejected or combined with laborious manual intervention. In our approach, instead, all anatomical elements remain present, as all interaction is defined on the original volume data.

3. Methods: Overview

Medical images contain the raw information of a patient's anatomy. Therefore, to empower clinicians to interact with this anatomy while remaining faithful to the true image data, our proposed manipulation and deformation methods are executed directly on the input medical images. This approach poses challenges for the design of selection, interaction, and deformation methods that work directly on volumetric images.

We rely on state-of-the-art techniques [1,2] for the computation of physics-based volumetric image deformation. These techniques combine robust mechanical response with efficient high-resolution deformation. Our novel image manipulation techniques, on the other hand, enable intuitive mapping of 2D gestures on a touchscreen to the volumetric deformation. The connecting bridge is the use of a rigid handle metaphor. By means of the interaction techniques, we manipulate rigid handles within the volumetric anatomy, and these handles serve as boundary conditions for the full anatomy deformation.

At runtime, the full image manipulation process requires the following steps, which are also identified in Figure 2:

1. Selection of rigid volumetric handles.
2. Computation of 3D handle transformations from 2D gestures.
3. Computation of boundary conditions for the deformation model.
4. Computation of coarse volumetric image deformation.
5. Interpolation to high-resolution image deformation.
6. Resampling of the deformed volumetric image.

Steps 1, 2 and 3 in the process are the proposed novel interaction techniques, and are described in detail in Sections 4–6, respectively. Steps 4, 5 and 6 are implemented using state-of-the-art volume image deformation techniques, as introduced above. In particular, we adopt a deformation technique called corotational coarsening [1], which uses two simulation meshes of different resolution. These two meshes and their mechanical

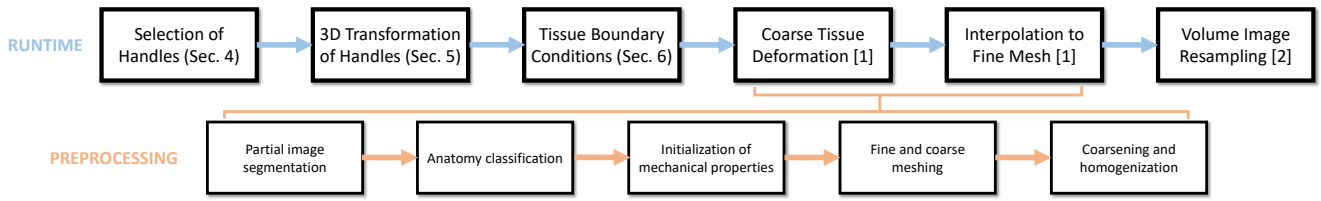

Figure 2. Runtime and preprocessing steps of our volume image manipulation method. At runtime, it combines novel interaction techniques with state-of-the-art volume image deformation techniques. A user selects handles within the volume, applies 3D transformations to the handles through 2D gestures, and the transformed handles translate into boundary conditions for an anatomy deformation model. We use the corotational coarsening method to deform the volume image [1], which entails three steps: low-resolution deformation, interpolation of this deformation to a high-resolution mesh, and resampling of the volume image [2]. In addition, the volume image deformation approach requires several preprocessing steps for the generation of a highly accurate low-resolution interactive model: partial segmentation of the volume image, classification of the anatomy elements based on image values, initialization of mechanical properties from image values, high-res and low-res meshing, and coarsening from the high-res deformation model to a homogenized low-res deformation model.

properties are defined at preprocessing stage, as indicated in the pipeline Figure 2. This preprocessing requires a partial segmentation of the volume image and the mapping from image values to mechanical properties based on anatomical classification (Please see the work of Torres et al. [1] for details). The fine mesh captures the desired level of detail of the anatomy, and represents mechanical deformations accurately, but it cannot be simulated in real time. The coarse mesh, on the other hand, does not respect anatomical detail explicitly, but it allows real-time simulation. Corotational coarsening takes the properties of the fine mesh and computes nonlinear shape functions and homogenized mechanical properties of the coarse mesh, such that the accuracy of the fine mesh is well preserved. At runtime, the solver simulates mechanical deformations on the coarse mesh, interpolates these simulations in a nonlinear fashion to the fine mesh, and finally to the full volumetric image thanks to a resampling method [2]. The original corotational coarsening method does not support as boundary conditions the transformation of arbitrary rigid handles. We extend the method to support such boundary conditions efficiently, and hence smoothly connect the handle metaphor from selection to deformation.

4. Methods: User-Guided Handle Selection

Similar to work in geometric modeling [48], we deform the anatomy by manipulating a set of handles. This metaphor is natural and intuitive, as it mimics the real-world action of pinching or grasping part of an object with our hands, and then translating and rotating the hands as rigid tools to deform the object.

The first necessary step in handle-based deformation is the selection of handles. Three aspects make this step challenging in our setting. One is the limitation of 2D interaction posed by the touchscreen, and the other two are the dense volumetric nature and the topological complexity of medical images.

In this section, we describe our intuitive, user-guided method for the selection of volumetric handles. First, we describe the definition of selection parameters directly through intuitive actions on the touchscreen. Next, we describe a region growing method to segment the volume and thus select each handle. Finally, we discuss handle refinement operations and the management of handle state to define various types of boundary conditions in the deformation of the anatomy.

4.1. Mapping of Selection Cues

The problem of handle selection can be posed as selecting a subset of the voxels in the volume dataset, based on some user input. This problem has been tackled before in medical imaging, as part of image segmentation. In that context, user control can be maximized by

displaying supplementary cross sections, and letting the user draw directly on such cross sections. However, with cross sections, interaction does not mimic real life actions.

Instead, we wish to let users select handles in a tangible manner, letting them express their intent directly on the volume image. To this end, we design a two-finger selection metaphor that exploits isosurface raycasting to map the 2D user's actions into 3D cues in a natural manner. In the following, we define the 2D and 3D positions associated with the two fingers, and we describe the mapping between finger actions and selection parameters.

We assume that user interaction is carried out using the index finger and the thumb. We define as $\mathbf{p}_i$ and $\mathbf{p}_t$ the screen-space positions of the index and the thumb, respectively. These points map along their corresponding viewing rays to 3D finger points $\mathbf{q}_i$ and $\mathbf{q}_t$ on the visible isosurface, respectively. In our implementation, we find the two 3D finger points through independent ray casting, but a graph-based optimization could provide more robust results [31].

When the user first touches the screen with both fingers, we define the handle seed $\mathbf{s}$ as the average of the corresponding 3D finger point positions, $\mathbf{s} = \frac{\mathbf{q}_i + \mathbf{q}_t}{2}$. Based on the 3D depth of the seed and the current perspective projection, we also compute the scale factor s between screen-space displacements and 3D displacements.

The user may then touch the screen with both fingers, and vary the distance between their screen-space positions, indicating a grow or zoom action. At any time, we define the handle extent d by scaling the distance between the finger screen-space positions, i.e., $d = s \, \|\mathbf{p}_i - \mathbf{p}_t\|$.

The various components of the mapping of selection cues are depicted in Figure 3. As soon as the user lifts one finger off the screen, the selection process is terminated. As shown in Figure 4, the selection is displayed at all times using a color mask.

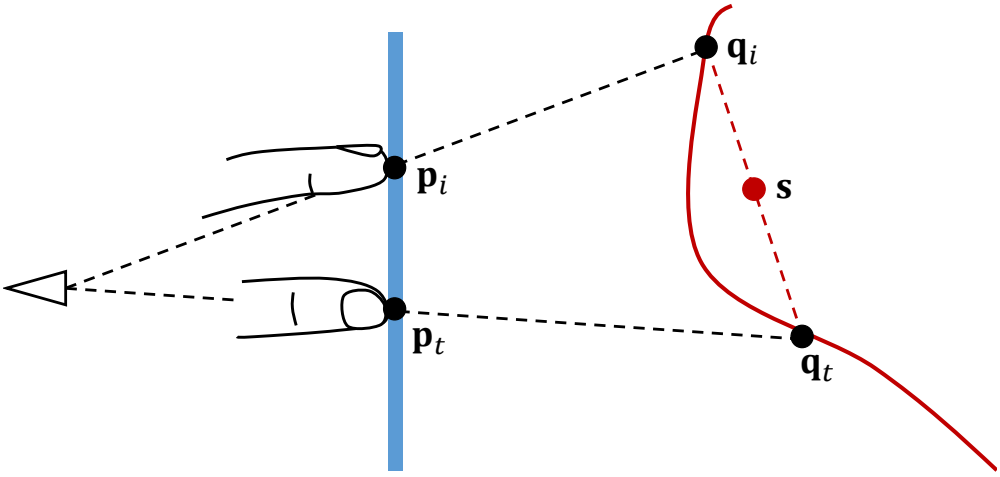

Figure 3. For handle selection, the 2D positions of the thumb and index on the touchscreen ($\mathbf{p}_t$ and $\mathbf{p}_i$, respectively) are mapped to 3D positions ($\mathbf{q}_t$ and $\mathbf{q}_i$, respectively) through isosurface raycasting. The selection is initialized at the seed $\mathbf{s}$.

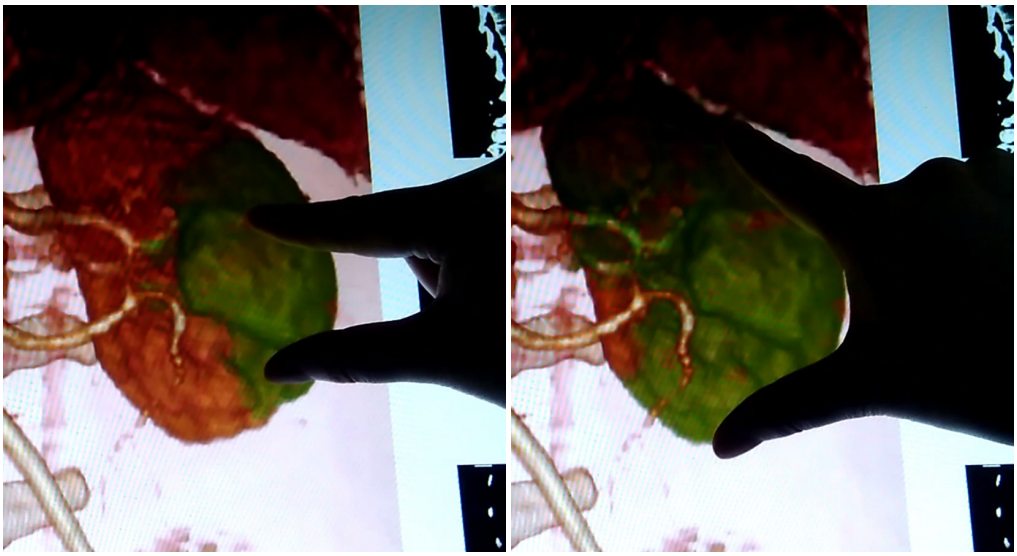

Figure 4. A mask is rasterized on the volume data to depict the selection of a handle. In the two snapshots, the extent indicated with the fingers is mapped to the selection.

4.2. Region Growing for Handle Selection

The seed and extent parameters defined in the previous section let users define their selection intent in an intuitive manner. The actual selection, however, requires identifying the visible part of the organ that is being touched, within the intended extent. This selection becomes a difficult task due to the complex topology of the anatomy embedded in the volume images and the lack of explicit definition of organ boundaries and connectivity.

To tackle the selection problem, we resort to image segmentation techniques, in particular the well-known seeded region growing algorithm [49]. This algorithm aligns perfectly with our interaction approach, as it allows us to efficiently identify a region of the volume that is connected to the seed and fulfills some homogeneity criterion. It considers the distribution of image density values around the seed voxel, and it grows a selected region incrementally by adding other voxels whose density is within a range of the density distribution at the seed.

Specifically, we start by registering the density δ_{seed} at the seed voxel and computing the standard deviation σ_{seed} of density values in the 1-ring neighborhood of the seed voxel. Then, we start a breadth-first growth from the seed. For every candidate voxel that we visit, we compute a homogeneity ratio h based on its density δ as:

$$h = \frac{|\delta - \delta_{\text{seed}}|}{\sigma_{\text{seed}}}. \tag{1}$$

If the homogeneity ratio is below a threshold $h_{\max}$, we accept the candidate voxel into the selected region, and we add as candidates the unvisited voxels within its six direct neighbors. In all our examples, we have used $h_{\max} = 1.1$ as homogeneity threshold, indicating that we accept voxels whose density difference w.r.t. the seed voxel is less than 110% of the standard deviation of the density around the seed. The threshold value was selected by preprocessing the data and analyzing the homogeneity of the organs present in the volume. By comparing the density differences to the standard deviation around the seed, we automatically adapt the selection criterion to the level of noise in the density data, and the user only needs to specify the seed and the extent of the selection. Nevertheless, our selection method assumes that the seed and its neighborhood are representative of the whole region of interest, and distinctive with respect to other adjacent anatomy. If the contrast in the image data is not sufficient, the method may have trouble growing the selection correctly.

In contrast to other segmentation approaches, region growing enables simple user-guided definition of the selection extent, following the interaction metaphor described in

Section 4.1. Moreover, region growing also allows for highly efficient parallel implementation on GPUs. We follow a queue-based implementation of the region growing algorithm, where the queue is initialized with the six direct neighbors of the seed, and each pass of breadth-first search flushes the queue and repopulates it with candidate neighbors of visited voxels. We stop the growth process when the number of passes over the queue has reached the extent, or if there are no more candidate voxels.

On the GPU, we launch one thread per candidate voxel in the queue. With this parallel implementation, the algorithm constitutes a parallel stencil-computation problem, which adapts perfectly to the computational paradigm of modern GPUs. However, due to the nature of the growth process, only a small fraction of the voxels are actually candidates and must be visited on each pass. Then, a simple dispatch mechanism of the parallel jobs would yield many launched threads that would perform futile computation. We succeed to drastically reduce the number of unnecessary threads that are launched through an efficient blocking scheme [44] that exploits the sparsity of the computation. Thanks to this highly efficient implementation, the selected region grows or shrinks in just a few milliseconds, providing visual feedback to the user and allowing a very fast and intuitive selection of the desired handle.

4.3. Handle Management

In addition to the procedure for handle selection described above, we provide methods to refine the handles and manage their status. Multiple handles can be selected at the same time, as shown in Figure 5, and each of them can be controlled independently. To do this, the user may pick one or several handles out of the currently existing set. To pick one handle, the user simply needs to press the touchscreen on the 2D projection of the handle.

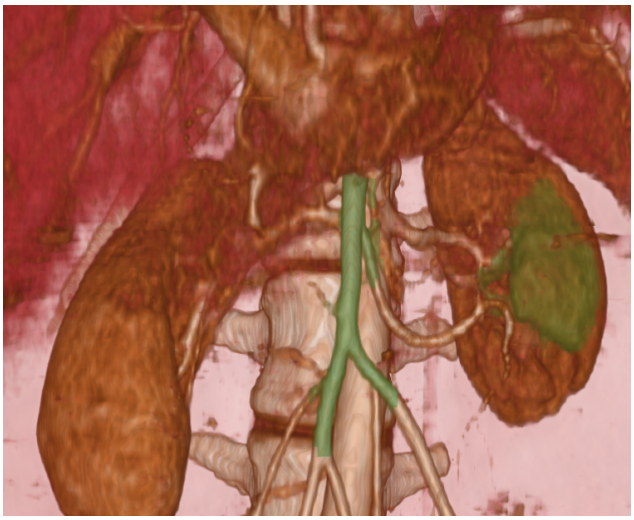

Figure 5. Multiple handles can be selected at the same time, and they can be controlled independently.

We enable handle refinement through union and difference set operations. Then, to locally enlarge or reduce a handle, the user creates a second handle selection, picks both the original and the new handle, and then selects the union or difference operation to enlarge or reduce the original handle, respectively.

Our soft-tissue deformation method, described in Section 6, defines boundary conditions on the fine tetrahedral mesh where the volume image is embedded. Then, even though handle selection and refinement are executed on the volume image, the selection must be applied to the nodes of the fine tetrahedral mesh too. When we select or refine a handle, we mark all the mesh tetrahedra that contain selected voxels, and we set the nodes of the handle as the nodes of the tetrahedra. We let each node belong to only one handle, the last one that claimed ownership of the node. In a refinement operation, we reset the nodes of the two handles involved before applying the union or difference operation.

In our hands-on manipulation operations, we define three possible status for handles: active, fixed, or idle. When a handle is active, its nodes undergo the manipulation operations described in Section 5, and then their positions are set as Dirichlet boundary conditions for soft-tissue deformation. When a handle is fixed, the positions of its nodes are also set as Dirichlet boundary conditions during soft-tissue deformation. When a handle is idle, its nodes are treated as regular free nodes during soft-tissue deformation, but the handle remains available to be active or fixed later.

5. Methods: Direct Handle Manipulation

Once handles are selected, the user may manipulate them in a natural and intuitive manner. Our solution for handle transformation tries to replicate the manipulation of real-world objects with the hands, with the constraint that user interactions are carried out on a 2D touchscreen. We propose a solution that maps 2D actions automatically into 3D transformations of the handles. Our solution is based on the numerical optimization of 3D rigid transformations, such that the visual projections of the handles best follow the actions of the user.

We start this section with the statement of the numerical optimization problem, which follows previous work on direct manipulation [9]. However, in contrast to their solution, we derive a closed-form solution for linearized manipulations. This solution is extremely efficient, hence the user obtains immediate visual feedback of the manipulation actions. We conclude the section by discussing variants of the optimization formulation.

5.1. Problem Statement and Formulation

We pose handle manipulation as a multi-finger interaction. The user touches a handle at multiple locations simultaneously, and slides the fingers on the screen to convey the motion of the projected handle. This projected motion is automatically transformed into an optimal 3D motion, providing the user with a sense of natural and intuitive manipulation of the handle directly in 3D.

When the user touches a handle, we identify the screen-space positions of the fingers, and we map them along their viewing rays to 3D positions on the visible isosurface, same as for handle selection as described in Section 4.1. We initialize these 3D positions as the handle points. Once a handle is transformed during manipulation, the positions $\{\mathbf{q}_j \in \mathbb{R}^3\}$ denote the updated positions of the handle points in view space, i.e., expressed in the reference system of the camera. We define as $\{\mathbf{p}_j \in \mathbb{R}^2\}$ the target 2D projections for handle points, expressed in normalized device coordinates. When the user moves the fingers on the screen, we record their current positions as the target 2D points.

Following Reisman et al. [9], we pose handle manipulation as the computation of the optimal 3D transformation of handle points $\{\mathbf{q}_j\}$ such that their projections match the target projections $\{\mathbf{p}_j\}$. We express the 3D transformation with a translation vector $\mathbf{t}$ and a rotation matrix $\mathbf{R}$, and the 3D-to-2D projection as a function $\mathrm{Proj}(\cdot)$. Then, we define the following error term for each handle point:

$$\mathbf{f}_j = \mathrm{Proj}\big(\mathbf{R}\,\mathbf{q}_j + \mathbf{t}\big) - \mathbf{p}_j. \tag{2}$$

And we formally define handle manipulation as the optimization of an objective function f, i.e.,

$$\{\mathbf{t}, \mathbf{R}\} = \arg\min f, \quad f = \sum_j \mathbf{f}_j^T\,\mathbf{f}_j. \tag{3}$$

The projection function consists of a linear transformation $\mathbf{P}$ from 3D to 2D, followed by a division by the 3D Z coordinate. With a unit vector $\mathbf{k} = (0,0,1)^T$, the projection can be written as

$$\mathrm{Proj}(\mathbf{v}) = \frac{1}{\mathbf{v}^T\,\mathbf{k}}\,\mathbf{P}\,\mathbf{v}. \tag{4}$$

5.2. Optimal Linearized Manipulation

For smooth and intuitive manipulation, we visualize the transformed handle continuously as the user moves fingers on the screen. After each finger motion, we compute the optimal transformation $\{\mathbf{t}, \mathbf{R}\}$, and we update the 3D handle points as $\mathbf{q}_j \leftarrow \mathbf{R}\,\mathbf{q}_j + \mathbf{t}$. In this way, each incremental finger motion results in a small rigid transformation.

The optimization (3) constitutes a nonlinear least-squares problem. In previous work [9], this problem was solved using an iterative Levenberg–Marquardt solver. However, under the assumption of small transformations, we can safely linearize the error terms (2), and solve a linear least-squares problem instead. This amounts to performing one Gauss–Newton iteration of the original nonlinear problem.

Two nonlinear functions must be linearized: the rotation and the perspective projection. The gradient of a projected vector can be derived from (4) as:

$$\frac{\partial(\mathrm{Proj}(\mathbf{v}))}{\partial \mathbf{v}} = \frac{1}{\mathbf{v}^T\,\mathbf{k}}\,\mathbf{P}\left(\mathbf{I} - \frac{\mathbf{v}\,\mathbf{k}^T}{\mathbf{v}^T\,\mathbf{k}}\right). \tag{5}$$

To linearize the rotation, it is convenient to express it using an axis-angle representation $\mathbf{r}$, where the magnitude $\|\mathbf{r}\|$ represents a rotation angle, and the direction $\frac{\mathbf{r}}{\|\mathbf{r}\|}$ represents the axis of rotation. Then, under small rotations, a rotation operation can be linearly approximated as

$$\mathbf{R}\,\mathbf{q} \approx (\mathbf{I} + \mathrm{Skew}(\mathbf{r}))\,\mathbf{q} = \mathbf{q} - \mathrm{Skew}(\mathbf{q})\,\mathbf{r}, \tag{6}$$

where $\mathrm{Skew}(\cdot)$ denotes a skew-symmetric matrix representing the cross-product linear transformation.

The gradient of a rotated vector can easily be derived from this linear approximation, resulting in:

$$\frac{\partial(\mathbf{R}\,\mathbf{q})}{\partial \mathbf{r}} = -\mathrm{Skew}(\mathbf{q}). \tag{7}$$

After linearizing the projection and the rotation, the gradients of error terms (2) w.r.t. the translation and the rotation can be easily derived, resulting in:

$$\frac{\partial \mathbf{f}_j}{\partial \mathbf{t}} = \frac{1}{\mathbf{q}_j^T\,\mathbf{k}}\,\mathbf{P}\left(\mathbf{I} - \frac{\mathbf{q}_j\,\mathbf{k}^T}{\mathbf{q}_j^T\,\mathbf{k}}\right), \tag{8}$$

$$\frac{\partial \mathbf{f}_j}{\partial \mathbf{r}} = \frac{1}{\mathbf{q}_j^T\,\mathbf{k}}\,\mathbf{P}\left(\frac{\mathbf{q}_j\,\mathbf{k}^T}{\mathbf{q}_j^T\,\mathbf{k}} - \mathbf{I}\right)\mathrm{Skew}(\mathbf{q}_j). \tag{9}$$

With these ingredients, we solve the linearized least-squares problem (3) as a small linear system. Grouping the unknown translation and rotation as $\mathbf{x} = \begin{pmatrix} \mathbf{t} \\ \mathbf{r} \end{pmatrix}$, the resulting linear system can be written as:

$$\mathbf{A}\,\mathbf{x} = \mathbf{b}, \tag{10}$$

$$\text{with } \mathbf{A} = \sum_i \begin{pmatrix} \frac{\partial \mathbf{f}_i}{\partial \mathbf{t}} & \frac{\partial \mathbf{r}_i}{\partial \mathbf{t}} \end{pmatrix}^T \begin{pmatrix} \frac{\partial \mathbf{f}_i}{\partial \mathbf{t}} & \frac{\partial \mathbf{r}_i}{\partial \mathbf{t}} \end{pmatrix} \tag{11}$$

$$\text{and } \mathbf{b} = -\sum_i \begin{pmatrix} \frac{\partial \mathbf{f}_i}{\partial \mathbf{t}} & \frac{\partial \mathbf{r}_i}{\partial \mathbf{t}} \end{pmatrix}^T (\mathrm{Proj}(\mathbf{q}_i) - \mathbf{p}_i). \tag{12}$$

5.3. Constrained Transformations

Our direct manipulation method, formulated as an optimization, admits the use of an arbitrary number of fingers. However, if the number of fingers is less than three, then the linear problem (10) is underconstrained.

In unconstrained situations, when the user interacts using only one or two fingers, we assume that the user's intent is to perform constrained transformations. One-finger

interaction is naturally mapped to XY translation, and two-finger interaction is naturally mapped to XYZ translation and Z rotation. We also let the user constrain the transformation at will, to prevent motions in undesired axes. Some useful examples are to eliminate Z translation, Z rotation, or XY translation.

Constrained transformations are easily implemented by selecting the appropriate rows and columns in the linear problem (10). Figure 1 shows examples of full and constrained transformations.

6. Methods: Volumetric Deformation

The human anatomy captured in a medical volume image exhibits high-resolution heterogeneity. Popular methods for the simulation of anatomical deformations would require a fine simulation mesh to correctly resolve the anatomical elements present in the medical image, but this choice would prevent interactive deformations. As an alternative, numerical coarsening methods [50,51] simulate deformations on coarse meshes, while maximizing the similarity w.r.t. the behavior produced by fine meshes. Among such methods, corotational coarsening [1] supports accurate handling of high-resolution boundary conditions under a linear corotational material model.

We start this section with a brief summary of the corotational coarsening method. The original method supports accurately Dirichlet boundary conditions for high-resolution points that are fixed at their rest position, but it does not handle other types of Dirichlet boundary conditions. We extend the corotational coarsening method to handle Dirichlet boundary conditions for rigidly transformed high-resolution points. In this way, we support precise handle-based manipulation of highly detailed anatomical elements, while the remaining anatomy is deformed interactively as dictated by the corotational coarsening model. We conclude the section by summarizing how the mesh deformation is applied to the embedded medical volume image through volume resampling.

6.1. Corotational Coarsening

The corotational coarsening model takes as input two tetrahedral meshes, one fine and one coarse, that discretize the same deformable volume, where the coarse nodes are a subset of the nodes of the fine mesh. Given a heterogeneous linear corotational FEM model [52] defined on the fine mesh, corotational coarsening computes homogenized stiffness matrices and nonlinear shape functions on the coarse mesh, such that the behavior of the fine mesh is closely approximated. Coarsening is designed for each coarse element independently, and the result is assembled to produce the coarse model on the full coarse mesh. The method pays special attention to the accurate approximation of Dirichlet and Neumann boundary conditions applied on the fine mesh. However, for clarity, and w.l.o.g., in the remainder we ignore the presence of fine nodes with non-zero Neumann boundary conditions (i.e., non-zero external forces), and we consider only fine nodes with Dirichlet boundary conditions (i.e., constrained positions).

Similar to Torres et al. [1], we introduce a *block notation* $[\cdot]$ to indicate replication of a 3×3 matrix into a block-diagonal matrix, or replication of a 3×1 vector into a long vector. The size of $[\cdot]$ can be inferred in each case from the terms it multiplies.

Let us denote with $\mathbf{x}_c$ and $\mathbf{x}_d$ vectors that concatenate the positions of all coarse nodes and all fine Dirichlet nodes in a coarse element, respectively. We denote with an overline the rest-state positions for the corresponding nodes. Let us also denote as $\mathbf{R}_c$ the best-fit rotation matrix of a coarse element. Then, using the block notation, the homogenized linear corotational problem on a coarse element can be expressed as:

$$[\mathbf{R}_c]\,\mathbf{K}_h\left([\mathbf{R}_c]^T\mathbf{x}_c - \overline{\mathbf{x}_c}\right) = \mathbf{f}_c + \mathbf{f}_d, \tag{13}$$

$$\text{with } \mathbf{f}_d = -[\mathbf{R}_c]\,\mathbf{K}_{hd}\left([\mathbf{R}_c]^T\,\mathbf{x}_d - \overline{\mathbf{x}_d}\right). \tag{14}$$

The matrices $\mathbf{K}_h$ and $\mathbf{K}_{hd}$ denote the homogenized stiffness matrices of the coarse element, and we refer the reader to the work of Torres et al. [1] for details about their

computation. The vectors $\mathbf{f}_c$ and $\mathbf{f}_d$ denote, respectively, external forces applied on the coarse nodes and coarse forces produced by the fine Dirichlet nodes.

The strength of the corotational coarsening method is that it imposes a small computational overhead w.r.t. a pure coarse FEM model, but it achieves accuracy comparable to a fine FEM model under certain types of boundary conditions. Next, we extend the method to handle accurately and efficiently rigid transformations of fine nodes, as produced by our handle manipulation method.

6.2. Coarsening of High-Resolution Rigid Transformations

In our setting, fine Dirichlet nodes are rigidly transformed with rotation $\mathbf{R}_m$ and translation $\mathbf{t}_m$, as computed by our direct manipulation method in Section 5. Note that, in contrast to the incremental transformations discussed in Section 5, here we refer to cumulative transformations from the rest-state configuration $\overline{\mathbf{x}_d}$. Using the block notation, the coarse forces (14) produced by rigidly transformed fine nodes are:

$$\begin{aligned}\mathbf{f}_d &= -[\mathbf{R}_c]\,\mathbf{K}_{hd}\left([\mathbf{R}_c]^T\left([\mathbf{R}_m]\,\overline{\mathbf{x}_d}+[\mathbf{t}_m]\right)-\overline{\mathbf{x}_d}\right)\\ &= -[\mathbf{R}_c]\,\mathbf{K}_{hd}\left(\left[\mathbf{R}_c^T\,\mathbf{R}_m-\mathbf{I}\right]\overline{\mathbf{x}_d}+\left[\mathbf{R}_c^T\,\mathbf{t}_m\right]\right).\end{aligned} \tag{15}$$

The computation of the resulting coarse forces appears at first computationally expensive, due to the density of matrix $\mathbf{K}_{hd}$. However, as noted by Torres et al. [1], and as a result of the block-replication structure, the product $\left[\mathbf{R}_c^T\,\mathbf{R}_m-\mathbf{I}\right]\overline{\mathbf{x}_d}$ amounts simply to a linear transformation of the nine distinct elements of the block-replicated matrix. Therefore, the product can be rewritten as a constant linear transformation of the nine-vector formed with the elements of the block-replicated matrix. We denote the resulting transformation as:

$$\left[\mathbf{R}_c^T\,\mathbf{R}_m-\mathbf{I}\right]\overline{\mathbf{x}_d}=\text{Reshape}(\overline{\mathbf{x}_d})\cdot\text{Unroll}\left(\mathbf{R}_c^T\,\mathbf{R}_m-\mathbf{I}\right), \tag{16}$$

where $\text{Unroll}(\cdot)$ transforms a 3×3 matrix into the nine-vector with all its elements, and $\text{Reshape}(\cdot)$ transforms a $3n$ vector into the corresponding $3n\times 9$ matrix.

The product $\mathbf{K}_{hd}\left[\mathbf{R}_c^T\,\mathbf{t}_m\right]$ also admits an efficient implementation thanks to block-replication, where columns of $\mathbf{K}_{hd}$ are first added together in three groups, and the result is multiplied by the block-replicated vector. We denote the resulting transformation as:

$$\mathbf{K}_{hd}\left[\mathbf{R}_c^T\,\mathbf{t}_m\right]=\text{ColumnAdd}(\mathbf{K}_{hd})\cdot\mathbf{R}_c^T\,\mathbf{t}_m, \tag{17}$$

where $\text{ColumnAdd}(\cdot)$ adds all columns of a matrix into three groups.

From (16) and (17), the computation of coarse forces due to fine Dirichlet nodes amounts to:

$$\begin{aligned}\mathbf{f}_d = &-[\mathbf{R}_c]\,\mathbf{K}_{hd}\cdot\text{Reshape}(\overline{\mathbf{x}_d})\cdot\text{Unroll}\left(\mathbf{R}_c^T\,\mathbf{R}_m-\mathbf{I}\right)\\ &-[\mathbf{R}_c]\cdot\text{ColumnAdd}(\mathbf{K}_{hd})\cdot\mathbf{R}_c^T\,\mathbf{t}_m.\end{aligned} \tag{18}$$

The product $\mathbf{K}_{hd}\cdot\text{Reshape}(\overline{\mathbf{x}_d})$ can be computed just once when handle selection is finalized. Overall, thanks to the efficient rearrangement of operations, the complete computation of coarse forces can be executed efficiently with a cost independent of the fine mesh discretization.

6.3. Volume Resampling

The corotational coarsening method can be efficiently implemented with all coarse-mesh-level computations on the CPU and all fine-mesh-level and volume-level computations on the GPU. On every handle-based manipulation step, the method first gathers force computations due to fine nodes, then executes a quasi-static deformation solve on the coarse mesh, and transfers the resulting soft-tissue deformation to the fine nodes. This

step uses precomputed nonlinear shape functions that respect the heterogeneity of the underlying anatomy.

The next step of the method requires transferring the deformation of the fine mesh to the actual volume image. This step is executed efficiently on the GPU as a tetrahedral rasterization operation, with the original volume image as a 3D texture map. We refer the reader to the work of Torres et al. [1] for full details. The final step is to rasterize the tetrahedra of the handles, to apply the visualization mask on the volume of the handles.

7. Results

In the images, we show several experiments that demonstrate the features of our hands-on deformation methods. We carry out exploration and deformation operations on multiple anatomical elements of the abdomen, such as a kidney, the stomach, or blood vessels.

7.1. Implementation Details and Performance

We have implemented our work on a compute platform consisting of a 3.1 GHz Quad-core Intel (Santa Clara, CA, USA) Core i7-3770S CPU with 16 GB of memory, and a NVIDIA (Santa Clara, CA, USA) GTX670 graphics card. We have used a HANNspree (Taipei, Taiwan) 23-inch touchscreen with full 10-finger tracking. We perform volume rendering using VTK [53], and we also employ its ray-casting picking functions.

For all the example manipulations, we have used an abdomen CT-scan with 5 million voxels. This dataset is publicly available under the OsiriX DICOM image library (ref. BREBIX) [54]. As a preprocess, we have scanned the opacity range of the model to identify the major anatomical elements and define transfer functions, as well as their mechanical parameters (i.e., Young modulus and Poisson's ratio). To apply the corotational coarsening deformation model, we have created coarse and fine tetrahedral meshes in the following way. First, we have created the coarse mesh as a regular tetrahedral mesh covering the volume image, producing a total of 6000 tetrahedra. Next, we have segmented bone material based on its opacity range, we have meshed the resulting bone surfaces, and we have created the fine tetrahedral mesh using TetGen by combining the bone nodes with a regular high-resolution grid. The fine mesh has a total of 281,000 tetrahedra.

In the examples shown in the paper, the average performance of our method is of 10 fps. The computational cost is dominated by CPU-GPU data transfers, which take about 50 ms per frame, and are motivated by the lack of low-level access of the VTK API to the volume data. This cost could be drastically reduced with a fully dedicated implementation, and thus improve the overall runtime performance. With the current performance, and same as in other applications, latency makes users move more slowly than they would with a more responsive system.

Within our algorithm, the cost of deformation is below 30 ms per frame, and resampling takes just 15 ms. The cost of other components, such as the selection of handles and the mapping of 2D gestures to 3D transformations, is negligible, thanks to the high efficiency of the methods we propose. For performance scalability, we refer the reader to the analysis carried out by Torres et al. [1]. During manipulation and deformation, our method adds just two operations to their pipeline: (i) the mapping of 2D gestures to the 3D transformation of a handle, and (ii) the computation of forces due to the rigidly transformed handle, as in (10). The cost of these two operations is negligible compared to the rest of their pipeline.

7.2. Selection Examples

Figure 6 shows two examples of dynamic growth of selected handles. The images correspond to real-time screen captures, and they were generated through hands-on interaction on the touchscreen. The examples highlight the success of our selection technique on both large and thin anatomy. The user naturally guides the extent through the index and thumb (denoted by the black circles in the images), but the selection is constrained to the organ of interest.

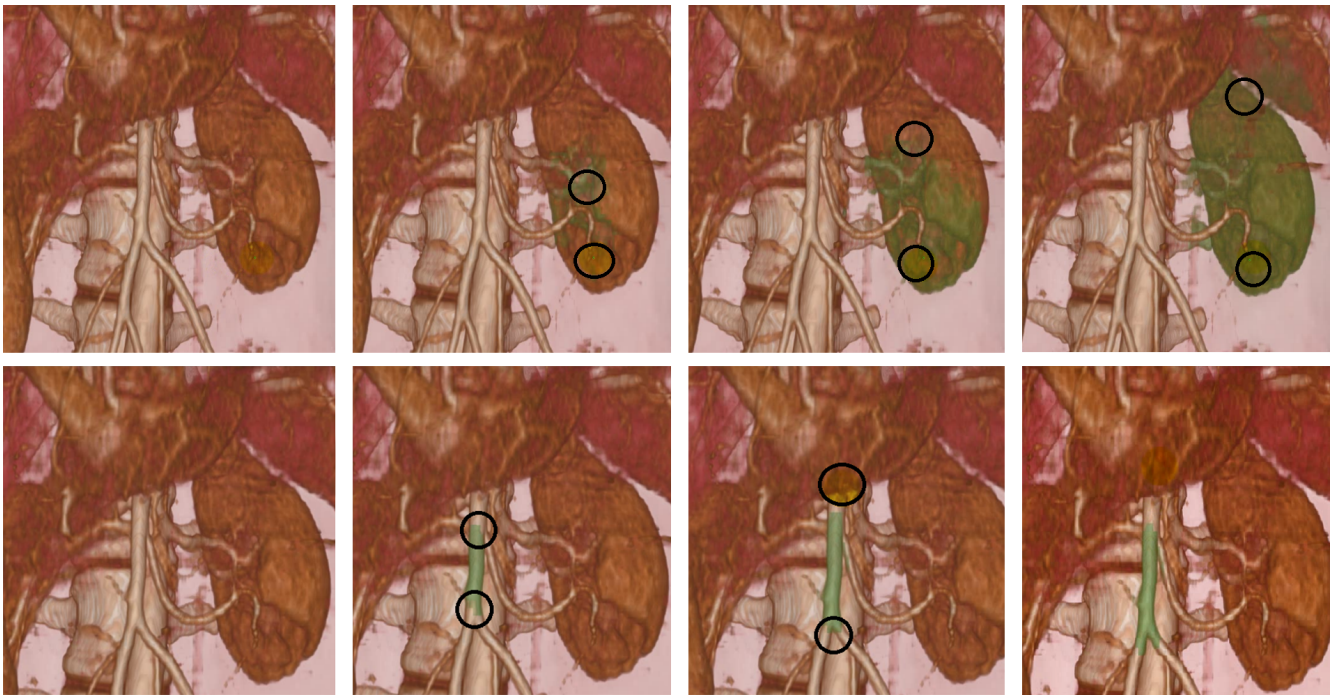

Figure 6. Two examples of dynamic growth of selected handles, in green, both on large and thin anatomy. The black circles represent $\mathbf{p}_i$ and $\mathbf{p}_t$, i.e., the screen-space positions of the index and the thumb. Note that the change of color occurs in the volumetric medical image representation; it is not a simple change of color of the rendered image. As the green color is constrained to the selected anatomical elements (kidney and vein), the figure demonstrates the success of the volumetric selection, i.e., no green pixels bleed to other regions of the data set.

7.3. Deformation Examples

Figures 7 and 8 show multiple examples of natural hands-on deformation of volumetric anatomy. The images combine real-time screen captures and recordings of the user interacting on the actual touchscreen. The top row in Figure 7 shows an in-plane translation of a vein using only one finger. This example demonstrates the ability to interact with fine anatomical features. Note that the blood vessels cannot be resolved by the coarse simulation mesh, but they are correctly handled thanks to the corotational coarsening deformation model described in Section 6. We have augmented the original deformation model to apply rigid transformations on fine features such as the blood vessels.

The bottom row in Figure 7 and the top row in Figure 8 show in-plane transformations (translation and rotation) of a kidney using two fingers. The bottom row in Figure 8, on the other hand, shows out-of-plane rotation of the kidney using three fingers. The arrows show the direction of motion. As shown also in Figure 1, the user executes a selection that encloses the volume of the kidney in a simple way, without explicit knowledge of its boundary, and then enjoys the ability to apply both in-plane and out-of-plane transformations in an intuitive manner.

8. Discussion

Our work is best classified as an enabling technology that opens the door to new possibilities. The two major areas of potential application are exploration for diagnosis and pre-operative planning. Next, we discuss in more detail how the features of our method could be leveraged in some specific applications.

Our direct interaction methods provide the ability to easily move and rotate anatomical elements, thus uncovering back surfaces or enabling access to hidden areas. This is particularly useful if the region of interest is on the backside of an anatomical element, or hidden behind layers of anatomy. The traditional approach is to navigate the view and adapt the volume rendering settings to display the region of interest, but in some cases the

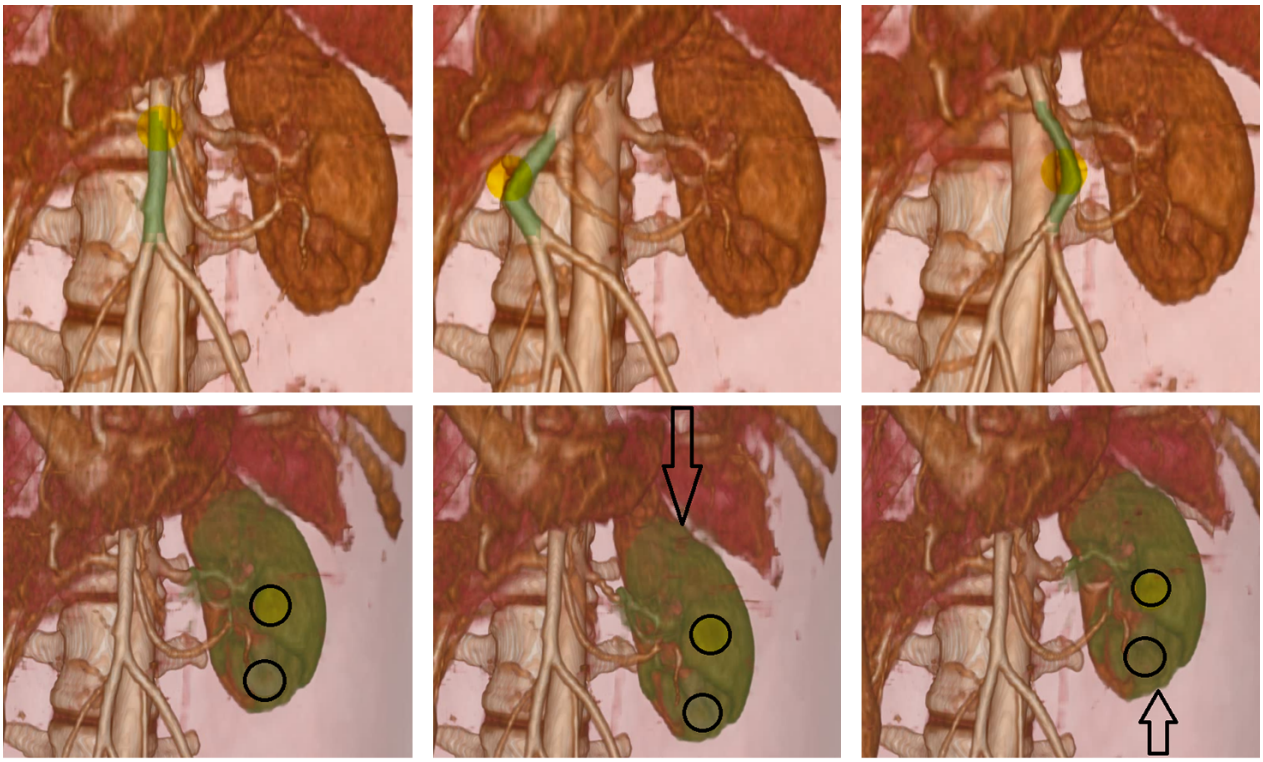

Figure 7. Real-time screen captures of natural hands-on deformation of volumetric anatomy. Top row: in-plane translation of a vein using only one finger. Bottom row: in-plane translation of the selected kidney using two fingers. The arrows show the direction of motion.

clinician may want to retain a certain overall view and/or combine viewing of the region of interest with motion of the anatomy.

One example is exploration through motion, deformation and palpation of the anatomy. Provided that the mechanical behavior is accurate enough, the clinician may virtually palpate the anatomy and identify clinically relevant features. Palpation can aid in the detection and understanding of, e.g., cancerous tissue [55].

Another example is planning of interventions when the anatomy is hidden from the point of entry of the surgical tools. Some of the decisions that could be planned include the point of entry itself, directions for cutting, or the amount of admissible motion and deformation of organs under safety conditions. For the specific case of the abdominal anatomy shown in our examples, particular clinical cases that could leverage our tools include resection of retroperitoneal tumors with complicated accessibility, e.g., partial nephrectomy [56], tumor resection in the adrenal gland hidden by the pancreas [57], or liver surgery in back/hidden areas of the liver.

With our proposed interaction metaphor, the clinician may simply move organs around, separate them, and rotate them to inspect various views, all through a simple "grab and move" action, much like we manipulate objects in the real world. In open surgery, the region of interest may be behind other anatomy layers. It is not enough to just "erase" these layers in the view of the volume image, as in reality they are cut and moved, and when moved they pull from other anatomy, possibly also at the region of interest. Our prototype implementation of tangible interaction is a first step toward mimicking these operations in an effective manner. Note that, for planning, it is not necessary to move the anatomy using realistic tools; it is sufficient to "grab" the anatomy and move it, with the connected tissue following in a physically accurate manner. The extent of the handle selection (Section 4) defines the part of an organ that is "grabbed". Once the clinician moves a handle, it may be kept in place to keep clear access to the region of interest. The clinician could rotate an organ and visualize and explore blood vessels in the region

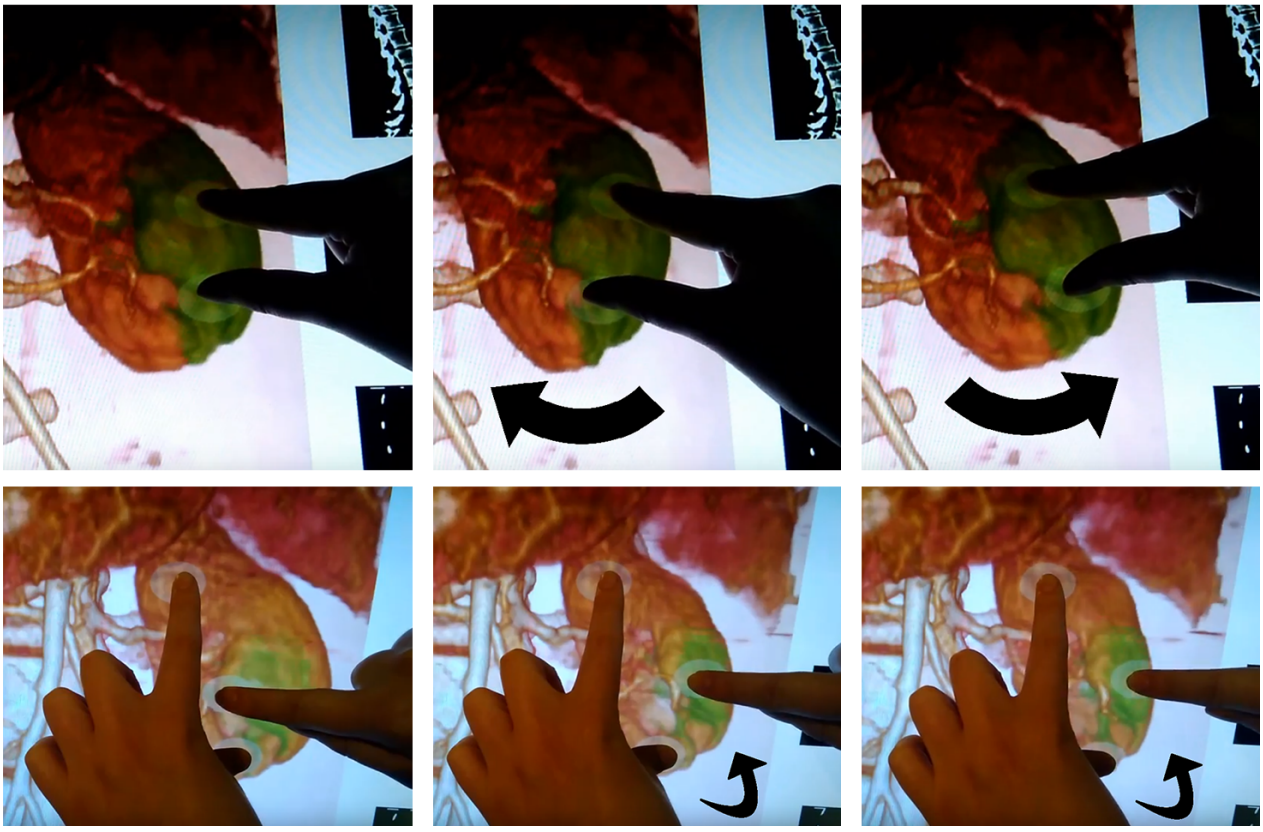

Figure 8. Video images of the user performing natural hands-on deformation of volumetric anatomy, interacting with the actual touchscreen. Top row: in-plane translation and rotation of the kidney using two fingers. Bottom row: out-of-plane rotation of the kidney using three fingers. The arrows show the direction of motion.

of interest. The amount of admissible rotation depends on patient-specific mechanical response and requires a physics-based deformation model. Then, this rotation could be planned without compromising the safety of blood vessels due to excessive tension.

9. Conclusions

In this paper, we have presented a novel metaphor for interacting with medical volume images. Users are able to explore and manipulate the anatomical elements in a medical image, much like they would do in the real world, simply by touching and moving them with their hands. This novel interaction metaphor is possible thanks to three novel technical components: user-guided selection of handles following a region growing approach, direct manipulation of handles through optimization-based computation of linearized transformations, and extension of a corotational coarsening deformation model to handle high-resolution rigid transformations.

In future work, the different elements of the method should be evaluated from a user-performance point-of-view, and the applied impact should be evaluated on practical cases. As mentioned in the introduction, and evidenced by the computational performance metrics and the accompanying video, the latency of the current prototype implementation limits its practical use. While we succeeded to demonstrate the selection and manipulation metaphors, the existing latency required slow and careful actions by the user. With faster user actions, the amount of soft-tissue deformation between system updates soon becomes too large, and the simulation suffers artifacts. Faster computational response and safety limits on the deformation are needed for practical evaluation by end users. Moreover, even though the previous section discusses practical use of the proposed interaction metaphors, it remains unknown whether the interaction metaphors are truly useful for clinicians.

Further work should be devoted to testing the effectiveness of our hands-on interaction metaphors on clinically motivated use cases.

Alongside its innovative nature, our work also entails multiple limitations. In addition to user evaluation, future work directions could address such limitations.

The selection and deformation methods we have developed do not require an explicit segmentation of the medical image. This is a powerful feature, as it enables rapid access to functional data, without the need for time-consuming data preparation. If higher-quality results are needed, however, segmentation can be applied to the data, and this knowledge can be exploited by both the selection and deformation methods. The quality of our methods is higher with high-contrast images, such as the CT-scan shown in our results. For other types of image modalities, with worse contrast, the selection process and the assignment of mechanical properties would suffer errors. Note that this is not worse than for approaches based on segmentation.

The deformation model used in this work also suffers several limitations. Most importantly, it is limited to linear corotational materials, and it does not handle arbitrary types of boundary conditions. Real-time deformation of highly complex and heterogeneous datasets is still an open research challenge.

A central feature of our hands-on deformation method is the use of a touchscreen as interface. Touchscreens are nowadays commodity technology, which maximizes the impact of the methods. However, 2D interaction poses limitations in contrast to 3D immersive interaction. Our methods could be extended to 3D, but the interaction with dense volumetric datasets would pose new challenges. Immersing the hands into a volume image would produce ambiguities on what elements should be selected and should or should not interact with the hands. Within touchscreens, our work is limited to moderately sized screens, where the fingers of the user do not produce excessive clutter. It would be interesting to evaluate performance with respect to screen size, and compare our methods to indirect interaction.

Author Contributions: Conceptualization, R.T. and M.O.; methodology, R.T., A.R. and M.O.; software, R.T. and A.R.; validation, R.T.; formal analysis, R.T. and M.O.; investigation, R.T., A.R. and M.O.; resources, M.O.; data curation, R.T.; writing—original draft preparation, M.O.; writing—review and editing, R.T., A.R. and M.O.; visualization, R.T. and A.R.; supervision, M.O.; project administration, M.O.; funding acquisition, M.O. All authors have read and agreed to the published version of the manuscript.

Funding: This project has received funding from the European Research Council (ERC Consolidator Grant 772738 TouchDesign), and from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No. 764644, Rainbow. This paper only contains the author's views and the Research Executive Agency and the Commission are not responsible for any use that may be made of the information it contains.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The human torso dataset was obtained from the OsiriX DICOM Viewer's site.

Acknowledgments: The authors would like to thank the members of the MSLab at URJC for their help and comments.

Conflicts of Interest: The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Torres, R.; Rodríguez, A.; Espadero, J.M.; Otaduy, M.A. High-resolution Interaction with Corotational Coarsening Models. *ACM Trans. Graph.* **2016**, *35*, 211:1–211:11.
2. Rodríguez Aguilera, A.; León Salas, A.; Martín Perandrés, D.; Otaduy, M.A. A parallel resampling method for interactive deformation of volumetric models. *Comput. Graph.* **2015**, *53*, 147–155.

3. Hinckley, K.; Goble, J.C.; Pausch, R.; Kassell, N.F. New Applications for the Touchscreen in 2D and 3D Medical Imaging Workstations. In Proceedings of the SPIE Medical Imaging '95, San Diego, CA, USA, 26 February–2 March 1995.
4. Keefe, D.F. Integrating Visualization and Interaction Research to Improve Scientific Workflows. *IEEE Comput. Graph. Appl.* **2010**, *30*, 8–13.
5. López, D.; Oehlberg, L.; Doger, C.; Isenberg, T. Towards An Understanding of Mobile Touch Navigation in a Stereoscopic Viewing Environment for 3D Data Exploration. *IEEE Trans. Vis. Comput. Graph.* **2016**, *22*, 1616–1629.
6. Mendes, D.; Caputo, F.M.; Giachetti, A.; Ferreira, A.; Jorge, J. A Survey on 3D Virtual Object Manipulation: From the Desktop to Immersive Virtual Environments. *Comput. Graph. Forum* **2019**, *38*, 21–45.
7. Hancock, M.; Carpendale, S.; Cockburn, A. Shallow-depth 3D Interaction: Design and Evaluation of One-, Two- and Three-touch Techniques. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, San Diego, CA, USA, 28 April–3 May 2007; pp. 1147–1156.
8. Cohé, A.; Dècle, F.; Hachet, M. tBox: A 3D Transformation Widget Designed for Touch-screens. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Vancouver, BC, Canada, 7–12 May 2011; pp. 3005–3008.
9. Reisman, J.L.; Davidson, P.L.; Han, J.Y. A Screen-space Formulation for 2D and 3D Direct Manipulation. In Proceedings of the 22Nd Annual ACM Symposium on User Interface Software and Technology, Vancouver, BC, Canada, 4–7 October 2009; pp. 69–78.
10. Liu, J.; Au, O.K.; Fu, H.; Tai, C. Two-Finger Gestures for 6DOF Manipulation of 3D Objects. *Comput. Graph. Forum* **2012**, *31*, 2047–2055.
11. Martinet, A.; Casiez, G.; Grisoni, L. Integrality and Separability of Multitouch Interaction Techniques in 3D Manipulation Tasks. *IEEE Trans. Vis. Comput. Graph.* **2012**, *18*, 369–380.
12. Klein, T.; Guéniat, F.; Pastur, L.; Vernier, F.; Isenberg, T. A Design Study of Direct-Touch Interaction for Exploratory 3D Scientific Visualization. *Comput. Graph. Forum* **2012**, *31*, 1225–1234.
13. Gleicher, M.; Witkin, A. Through-the-lens Camera Control. In Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques, Chicago, IL, USA, 27–31 July 1992; pp. 331–340.
14. Wobbrock, J.O.; Forlizzi, J.; Hudson, S.E.; Myers, B.A. WebThumb: Interaction Techniques for Small-Screen Browsers. In Proceedings of the 15th Annual ACM Symposium on User Interface Software and Technology, Paris France, 27–30 October 2002; Association for Computing Machinery: New York, NY, USA, 2002; pp. 205–208.
15. Rodríguez, M.B.; Agus, M.; Marton, F.; Gobbetti, E. HuMoRS: Huge Models Mobile Rendering System. In Proceedings of the 19th International ACM Conference on 3D Web Technologies, Vancouver, BC, Canada, 8–10 August 2014; Association for Computing Machinery: New York, NY, USA, 2014; pp. 7–15.
16. Hilliges, O.; Kim, D.; Izadi, S.; Weiss, M.; Wilson, A. HoloDesk: Direct 3D Interactions with a Situated See-through Display. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Austin, TX, USA, 5–10 May 2012; pp. 2421–2430.
17. Spindler, M.; Büschel, W.; Dachselt, R. Use Your Head: Tangible Windows for 3D Information Spaces in a Tabletop Environment. In Proceedings of the 2012 ACM International Conference on Interactive Tabletops and Surfaces, Cambridge, MA, USA, 11–14 November 2012; pp. 245–254.
18. Marton, F.; Rodriguez, M.B.; Bettio, F.; Agus, M.; Villanueva, A.J.; Gobbetti, E. IsoCam: Interactive Visual Exploration of Massive Cultural Heritage Models on Large Projection Setups. *J. Comput. Cult. Herit.* **2014**, *7*, 1–24.
19. Coffey, D.; Malbraaten, N.; Le, T.; Borazjani, I.; Sotiropoulos, F.; Erdman, A.G.; Keefe, D.F. Interactive Slice WIM: Navigating and Interrogating Volume Data Sets Using a Multisurface, Multitouch VR Interface. *IEEE Trans. Vis. Comput. Graph.* **2012**, *18*, 1614–1626.
20. Bruder, G.; Steinicke, F.; Sturzlinger, W. To Touch or Not to Touch?: Comparing 2D Touch and 3D Mid-air Interaction on Stereoscopic Tabletop Surfaces. In Proceedings of the 1st Symposium on Spatial User Interaction, Los Angeles, CA, USA, 20–21 July 2013; pp. 9–16.
21. Besançon, L.; Issartel, P.; Ammi, M.; Isenberg, T. Mouse, Tactile, and Tangible Input for 3D Manipulation. In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems, Denver, Co, USA, 6–11 May 2017; Association for Computing Machinery: New York, NY, USA, 2017; pp. 4727–4740.
22. Kniss, J.; Kindlmann, G.; Hansen, C. Interactive Volume Rendering Using Multi-dimensional Transfer Functions and Direct Manipulation Widgets. In Proceedings of the Conference on Visualization '01, San Diego, CA, USA, 21–26 October 2001; pp. 255–262.
23. Salama, C.R.; Keller, M.; Kohlmann, P. High-Level User Interfaces for Transfer Function Design with Semantics. *IEEE Trans. Vis. Comput. Graph.* **2006**, *12*, 1021–1028.
24. Tory, M.; Potts, S.; Moller, T. A parallel coordinates style interface for exploratory volume visualization. *IEEE Trans. Vis. Comput. Graph.* **2005**, *11*, 71–80.
25. Weiskopf, D.; Engel, K.; Ertl, T. Interactive clipping techniques for texture-based volume visualization and volume shading. *IEEE Trans. Vis. Comput. Graph.* **2003**, *9*, 298–312.
26. Svakhine, N.; Ebert, D.S.; Stredney, D. Illustration motifs for effective medical volume illustration. *IEEE Comput. Graph. Appl.* **2005**, *25*, 31–39.
27. Song, P.; Goh, W.B.; Fu, C.W.; Meng, Q.; Heng, P.A. WYSIWYF: Exploring and Annotating Volume Data with a Tangible Handheld Device. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Vancouver, BC, Canada, 7–12 May 2011; pp. 1333–1342.

28. Zorzal, E.R.; Sousa, M.; Mendes, D.; dos Anjos, R.K.; Medeiros, D.; Paulo, S.F.; Rodrigues, P.; Mendes, J.J.; Delmas, V.; Uhl, J.F.; et al. Anatomy Studio: A tool for virtual dissection through augmented 3D reconstruction. *Comput. Graph.* **2019**, *85*, 74–84.
29. Wiebel, A.; Vos, F.M.; Foerster, D.; Hege, H. WYSIWYP: What You See Is What You Pick. *IEEE Trans. Vis. Comput. Graph.* **2012**, *18*, 2236–2244.
30. Owada, S.; Nielsen, F.; Igarashi, T. Volume Catcher. In Proceedings of the 2005 Symposium on Interactive 3D Graphics and Games, Washington, DC, USA, 3–6 April 2005; Association for Computing Machinery: New York, NY, USA, 2005; pp. 111–116.
31. Wiebel, A.; Preis, P.; Vos, F.M.; Hege, H.C. 3D Strokes on Visible Structures in Direct Volume Rendering. In *EuroVis—Short Papers*; Hlawitschka, M., Weinkauf, T., Eds.; The Eurographics Association: Geneva, Switzerland, 2013.
32. Stoppel, S.; Hege, H.C.; Wiebel, A. Visibility-Driven Depth Determination of Surface Patches in Direct Volume Rendering. In *EuroVis—Short Papers*; Elmqvist, N., Hlawitschka, M., Kennedy, J., Eds.; The Eurographics Association: : Geneva, Switzerland, 2014.
33. Stoppel, S.; Bruckner, S. Smart Surrogate Widgets for Direct Volume Manipulation. In Proceedings of the 2018 IEEE Pacific Visualization Symposium (PacificVis), Kobe, Japan, 10–13 April 2018; pp. 36–45.
34. Besançon, L.; Sereno, M.; Yu, L.; Ammi, M.; Isenberg, T. Hybrid Touch/Tangible Spatial 3D Data Selection. *Comput. Graph. Forum* **2019**, *38*, 553–567.
35. Westermann, R.; Rezk-Salama, C. Real-Time Volume Deformations. *Comput. Graph. Forum* **2001**, *20*, 443–451.
36. Rezk-Salama, C.; Scheuering, M.; Soza, G.; Greiner, G. Fast Volumetric Deformation on General Purpose Hardware. In Proceedings of the ACM SIGGRAPH/EUROGRAPHICS Workshop on Graphics Hardware, Los Angeles, CA, USA, August 2001; pp. 17–24.
37. Rhee, T.; Lewis, J.P.; Neumann, U.; Nayak, K. Soft-Tissue Deformation for In Vivo Volume Animation. In Proceedings of the 15th Pacific Conference on Computer Graphics and Applications, Maui, HI, USA, 29 October–2 November 2007; pp. 435–438.
38. Correa, C.D.; Silver, D.; Chen, M. Volume Deformation via Scattered Data Interpolation. In Proceedings of the Sixth Eurographics/IEEE VGTC Conference on Volume Graphics, Prague, September 3-4, 2007; pp. 91–98.
39. Correa, C.; Silver, D.; Chen, M. Feature Aligned Volume Manipulation for Illustration and Visualization. *IEEE Trans. Vis. Comput. Graph.* **2006**, *12*, 1069–1076.
40. Georgii, J.; Westermann, R. A Generic and Scalable Pipeline for GPU Tetrahedral Grid Rendering. *Vis. Comput. Graph. IEEE Trans.* **2006**, *12*, 1345–1352.
41. Correa, C.D.; Silver, D.; Chen, M. Constrained illustrative volume deformation. *Comput. Graph.* **2010**, *34*, 370–377.
42. Kretschmer, J.; Soza, G.; Tietjen, C.; Suehling, M.; Preim, B.; Stamminger, M. ADR—Anatomy-Driven Reformation. *IEEE Trans. Vis. Comput. Graph.* **2014**, *20*, 2496–2505.
43. 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 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, Anaheim, CA, USA, 19–21 July 2013; pp. 181–185.
44. Rodríguez, A.; León, A.; Arroyo, G. Parallel deformation of heterogeneous ChainMail models: Application to interactive deformation of large medical volumes. *Comput. Biol. Med.* **2016**, *79*, 222 – 232.
45. Fortmeier, D.; Wilms, M.; Mastmeyer, A.; Handels, H. Direct Visuo-Haptic 4D Volume Rendering Using Respiratory Motion Models. *IEEE Trans. Haptics* **2015**, *8*, 371–383.
46. Faure, F.; Duriez, C.; Delingette, H.; Allard, J.; Gilles, B.; Marchesseau, S.; Talbot, H.; Courtecuisse, H.; Bousquet, G.; Peterlik, I.; Cotin, S. SOFA: A Multi-Model Framework for Interactive Physical Simulation. In *Soft Tissue Biomechanical Modeling for Computer Assisted Surgery*; Payan, Y., Ed.; Studies in Mechanobiology, Tissue Engineering and Biomaterials; Springer: Berlin/Heidelberg, Germany, 2012; Volume 11, pp. 283–321.
47. Talbot, H.; Haouchine, N.; Peterlik, I.; Dequidt, J.; Duriez, C.; Delingette, H.; Cotin, S. Surgery Training, Planning and Guidance Using the SOFA Framework. In *EG 2015—Dirk Bartz Prize*; Hege, H.C., Ropinski, T., Eds.; The Eurographics Association: Geneva, Switzerland, 2015.
48. Botsch, M.; Kobbelt, L. An Intuitive Framework for Real-time Freeform Modeling. *ACM Trans. Graph.* **2004**, *23*, 630–634.
49. Adams, R.; Bischof, L. Seeded region growing. *IEEE Trans. Pattern Anal. Mach. Intell.* **1994**, *16*, 641–647.
50. Kharevych, L.; Mullen, P.; Owhadi, H.; Desbrun, M. Numerical Coarsening of Inhomogeneous Elastic Materials. *ACM Trans. Graph.* **2009**, *28*, 51:1–51:8.
51. Nesme, M.; Kry, P.G.; Jerábková, L.; Faure, F. Preserving Topology and Elasticity for Embedded Deformable Models. *ACM Trans. Graph.* **2009**, *28*, 52:1–52:9.
52. Müller, M.; Gross, M. Interactive Virtual Materials. *Proc. Graph. Interface* **2004**, *4*, 239–246.
53. Schroeder, W.; Martin, K.; Lorensen, B. *The Visualization Toolkit: An Object-Oriented Approach to 3D Graphics*, 3rd ed.; Technical Report; Kitware Inc.: New York, NY, USA, 2004.
54. OsiriX. DICOM Image Library. 2020. Available online: (accessed on 30 August 2021).
55. Kolb, T.M.; Lichy, J.; Newhouse, J.H. Comparison of the Performance of Screening Mammography, Physical Examination, and Breast US and Evaluation of Factors that Influence Them: An Analysis of 27,825 Patient Evaluations. *Radiology* **2002**, *225*, 165–175.
56. Gill, I.S.; Delworth, M.G.; Munch, L.C. Laparoscopic retroperitoneal partial nephrectomy. *J. Urol.* **1994**, *152*, 1539–1542.
57. MacGillivray, D.C.; Whalen, G.F.; Malchoff, C.D.; Oppenheim, D.S.; Shichman, S.J. Laparoscopic resection of large adrenal tumors. *Ann. Surg. Oncol.* **2002**, *9*, 480–485.