Image-Based Palpation Simulation With Soft Tissue Deformations Using Chainmail on the GPU

Dirk Fortmeier[1,2], Andre Mastmeyer[1], Heinz Handels[1]

[1]Institute of Medical Informatics, University of Lübeck [2]Graduate School for Computing in Medicine and Life Sciences, University of Lübeck
fortmeier@imi.uni-luebeck.de

Abstract. Virtual reality surgery simulation can provide an environment for the safe training of medical interventions. In many of these interventions palpation of target organs is common to search for certain anatomical structures in a first step. We present a method for visuo-haptic simulation with tissue deformation caused by palpation solely based on CT data of a patient. Generation of haptic force feedback involves a force parameter image based on distances to the patient's skin and bone. To create a deformed version of the patient's image data, the ChainMail method is applied; bone structures are considered to be undeformable. The simulation can be used to palpate the iliac crest and spinous processes for the preparation of a lumbar puncture or for palpation of the ribcage.

1 Introduction

Finding anatomical structures for the preparation of medical interventions is an aspect of recent visuo-haptic surgery simulator frameworks. In [1], optical markers have been placed in the scene to indicate the location of the iliac crest and spinous processes. In a real intervention, these are palpated before performing a lumber puncture.

The frameworks of [2, 3] use haptic devices to simulate the palpation of the femoral artery to prepare the steps necessary for the Seldinger technique. In these simulations, soft tissue is rendered using triangular surfaces. To display deformations of these surfaces, several methods such as finite element methods or mass-spring models can be used. Another approach is to compute deformed volumetric images based on data obtained by imaging procedures as computed tomography (CT) and visualize these by direct volume rendering. In [4], this was demonstrated for the deformations occurring in a needle insertion simulator. Rendering and calculation of deformations directly on the image data can reduce or circumvent a time-consuming segmentation process that is normally needed for the creation of virtual patients.

In this paper, we present (1) an image-based haptic algorithm for palpation simulation and (2) a ChainMail implementation on graphics hardware based on

H.-P. Meinzer et al. (Hrsg.), *Bildverarbeitung für die Medizin 2013*, Informatik aktuell,
DOI: 10.1007/978-3-642-36480-8_26,

the work of [5, 6] to visualize the deformations occurring in a palpation only using the CT data of virtual patients. The methods presented can be used to palpate the iliac crest and spinous processes for the preparation of a lumbar puncture or for palpation of the ribcage using a Phantom Omni haptic device.

2 Methods and Materials

Visuo-haptic systems rely on two components: Visualization and the simulation of haptic force-feedback. In the following, we present a method for simulation of haptic force-feedback based on a force parameter image and a visualization scheme for rendering of the deformations occuring during the palpation.

2.1 Palpation Simulation

Palpation is simulated by a proxy based volume haptic rendering approach [7]. Here, instead of haptic rendering of the image data, a force parameter image $\mathbf{F} : \mathbb{N}^3 \rightarrow \mathbb{R}$ is created based on the original CT image $\mathbf{I} : \mathbb{N}^3 \rightarrow \mathbb{R}$ in a preprocessing step. Similar to the work of [8], distance maps are used to calculate the values of the force parameter image. Here, these depend on the distance to the patient's skin surface and proximity to bone structures. It is obtained as follows: Using a thresholding operation, bone structures are segmented. For the patient's surface, region growing is used to mask each voxel outside of the patient (giving binary mask $\mathbf{C}$). From these binary masks, euclidean distance maps $\mathbf{A}$ and $\mathbf{B}$ are created (for bone and surface resp.), which assign to each voxel the distance to the nearest voxel of the masks. These are combined together with the binary mask for the body ($\mathbf{C}$) to obtain the force parameter image

$$\mathbf{F} = \alpha \frac{\mathbf{C}}{\beta + \mathbf{A}} + \gamma \mathbf{B} \qquad \alpha, \beta, \gamma \in \mathbb{R} \tag{1}$$

Parameter settings for eq. (1) have been adjusted manually (e.g. for the rib cage scenario, $\alpha = \frac{200}{3}$, $\beta = 2$, $\gamma = \frac{1}{60}$ yielded good results). An example of a slice of a force parameter image can be seen in Fig. 1. Based on the force parameter image and the the current position of the haptic device $\mathbf{x}$, a proxy position $\mathbf{p}$ is computed

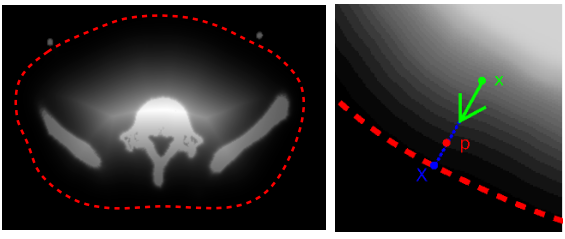

Fig. 1. Left: Force parameter image for the computation of palpation. The patients skin is indicated by a red line. Right: Computation of the proxy $\mathbf{p}$ and surface point $\mathbf{X}$ based on the device tip $\mathbf{x}$ and the normalized gradient of the force parameter image.

$$\mathbf{p} = \mathbf{x} + \frac{\nabla \mathbf{F}(\mathbf{x})}{||\nabla \mathbf{F}(\mathbf{x})||}\mathbf{F}(\mathbf{x}) \tag{2}$$

The proxy position and position of the haptic device is then connected by a virtual spring that is used to calculate the force feedback $\mathbf{f} = k(\mathbf{x}-\mathbf{p})$ by Hooke's law. To simulate friction, the new proxy position is interpolated linearly with the old proxy position.

Furthermore, the surface point $\mathbf{X}$ is computed by finding the intersection of the skin and the line defined by $\mathbf{x}$ and $\mathbf{p}$ (Fig. 1). This point is needed for the visualization of the deformation in the following.

2.2 Visualization of Deformations

The deformations in our simulator are computed by the ChainMail algorithm optimized for the processing with graphics hardware [5]. The main idea is to consider each element of the image as an element in a chainmail-like structure and restrict the positions of each element based on the positions of the neighbors. These restrictions are enforced iteratively until no further elements have been moved. In each iteration, each element checks if one of its six neighbors has been moved. If this is true, and the restrictions are violated, the restrictions are enforced and the element itself is considered as moved in the next iteration.

In Fig. 2 the restrictions imposed by a left neighbor on an element are illustrated. The parameters defining the restricted area are the minimum and maximum distance to the neighbor element and the maximum sheer distance (minDist, maxDist and maxSheer).

We further modify the algorithm in a way that bone structures (elements with a Houndsfield value higher than a certain threshold) are not considered: Each element that is classified as bone omits the enforcement of the restrictions imposed by its neighbors. For the relaxation step, we use a diffusive approach together with a material function [4] to regulate the diffusion rate. Here, a continuous approximation of the Heaviside function with the step at the bone threshold is used to restrict the diffusion to non-bone structures. Now, to calculate and display a deformed image $\mathbf{J}$ based on the undeformed image $\mathbf{I}$, several steps have to be performed (Fig. 3). We implemented these steps on the GPU using NVidia CUDA. First, the surface point $\mathbf{X}$ and the haptic device position $\mathbf{x}$ are used as undeformed and deformed position of the first moved element for the ChainMail computations. The deformed image is then created by relaxation

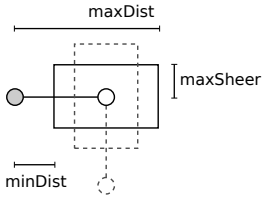

Fig. 2. Restrictions of the movement of the white element imposed by the left neighbor (gray) and bottom neighbor (dashed) as done by [5].

	Average ± Std dev	min	max
1. ChainMail	5.58 ± 2.06	0.22	11.53
2. Relax	52.27 ± 5.02	36.72	65.39
3. Invert	2.55 ± 0.25	1.68	3.26
4. Copy to PBO	0.17 ± 0.02	0.14	0.32
5. Update texture	0.75 ± 0.17	0.56	2.38
6. Reset	3.57 ± 0.19	0.90	4.15
total+overhead	65.07 ± 6.75	44.24	82.23

Table 1. Measured times for the parts of the algorithm in ms. The average is shown together with the corresponding standard deviation.

and inversion of the deformation field. For better performance, each of the steps is only calculated in a cubic volume of interest (VOI) around $\mathbf{x}$ with a fixed edge length. A pixel buffer object (PBO) is then used as the interface between an OpenGL 3D texture and the CUDA data.

Afterwards, the 3D texture can be rendered by a conventional volume renderer (we use the vtkGPUVolumeRayCastMapper provided by the open source Visualization Toolkit). For all results presented in the following, a NVidia GTX 680 has been used.

3 Results

In Fig. 4, a simple cube object and a cube surrounded by bone are deformed by the ChainMail algorithm. Different parameter settings are demonstrated: (p_1) minDist = 0.5, maxDist = 1.5, maxSheer = 0.5; ($p2$) minDist = 0.25, maxDist = 1.5, maxSheer = 1.0. In the first two columns, no relaxation is applied. In the other two 20, resp. 40, iterations of the diffusive relaxation are used. The figure demonstrates that bone is not deformed neither by the ChainMail algorithm nor by the diffusive relaxation.

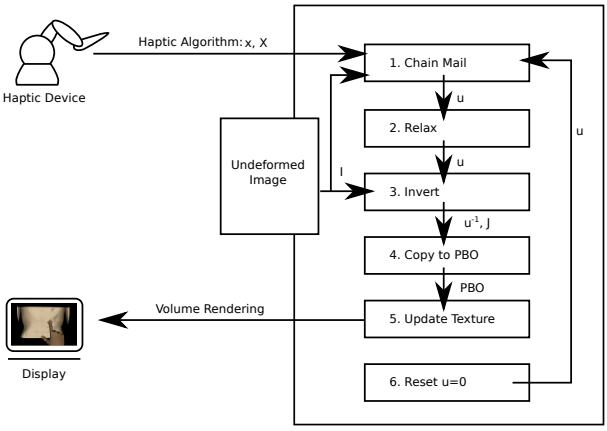

Fig. 3. Data transfer from each part of the deformation visualization algorithm on the GPU. The initial displacement calculated by the haptic algorithm ($\mathbf{X}$, $\mathbf{x}$) is transfered to the GPU based calculation of deformations. After computation of the deformed image, it is rendered via a volume renderer.

Table 2. Measured mean runtimes t_i in ms and number of moved elements n_i for the ChainMail algorithm with parameter setting p_i and initial moved distance d (in voxel edge length). The parameters used are $p_1 = (\text{minDist} = 0.5, \text{maxDist} = 1.5, \text{maxSheer} = 0.5)$ and $p2 = (\text{minDist} = 0.25, \text{maxDist} = 1.5, \text{maxSheer} = 1.0)$.

d	t_1	n_1	t_2	n_2
3	0.22	230	0.22	56
5	0.40	1158	0.39	259
10	1.23	9918	1.08	2120
20	5.98	82238	4.71	17309
30	18.04	280958	13.88	58898

A screenshot of the simulator is presented in Fig. 5. At the tip of the virtual finger, deformed tissue can be seen in a 3D volume rendering and slices of the deformed volume. For a virtual patient with image data consisting of $256 \times 256 \times 236$ elements and a VOI size of 64^3 elements, an interactive total frame rate of 11.50 ± 1.52 Hz has been achieved.

Tab. 1 shows times for each of the components of the visual deformation algorithm. The average processing times with standard deviation and minimum/maximum processing times were measured during a normal use of the simulator. Obviously, the diffusive relaxation is the most time consuming subtask (80% of the total time needed for the whole deformation algorithm). Runtimes of only the ChainMail subtask with parameter settings p_1 and p_2 are displayed in tab. 2: One element has been moved a distance d and the time necessary and the total number of moved elements is shown.

Haptic forces enable the user to distinguish ribs and the space between them in a ribcage palpation scenario. Furthermore, in a lumbar puncture scenario, the iliac crest and spinous processes can be palpated using the methods presented.

4 Discussion

A visual deformation algorithm together with a haptic rendering component have been presented in a simulator framework for the palpation of virtual patients. The effectiveness of the ChainMail algorithm on the GPU using CUDA has

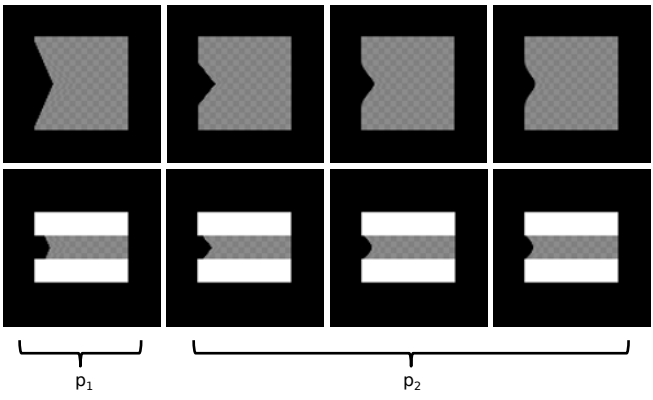

Fig. 4. Several test cases with varying ChainMail parameter settings (p_1, p_2) and relaxation steps (20 resp. 40 in the two columns on the right side).

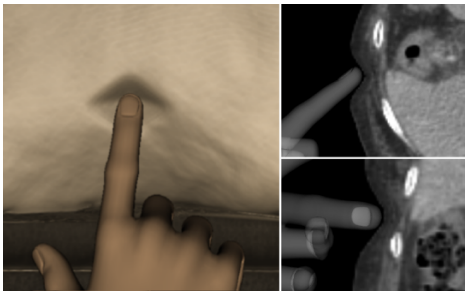

Fig. 5. Screenshot of a simulated palpation of the ribcage. The diamond-shaped deformation caused by the ChainMail algorithm can be seen clearly. On the right, slices of the deformed CT data are shown.

been confirmed, several thousand elements can be computed in just a few milliseconds. Relaxation is the most time consuming part due to the relatively high number of iterations needed to produce smooth deformations (here, 20 iterations were applied). Even with this high number of iterations, the diamond-shaped deformation caused by the limitations of the ChainMail algorithm cannot be compensated completely. Thus, in future work, improvements in this area are desired. Further work has to include a study of face validity, user acceptance and portability to other patients and scenarios. A comparison of the deformations and forces of the haptic simulation to ground truth data (e.g. aquired with a finite element simulation) should be done to verify the realism of the models.

Acknowledgement. This work is supported by the German Research Foundation (DFG HA 2355/10-1) and the Graduate School for Computing in Medicine and Life Sciences funded by Germany's Excellence Initiative (DFG GSC 235/1).

References

1. Färber M, Hummel F, Gerloff C, et al. Virtual Reality Simulator for the Training of Lumbar Punctures. Methods Inf Med. 2009;48(5):493–501.
2. Coles TR, John NW, Gould D, et al. Integrating Haptics with Augmented Reality in a Femoral Palpation and Needle Insertion Training Simulation. IEEE Trans Haptics. 2011;4(3):199–209.
3. Ullrich S, Kuhlen T. Haptic Palpation for Medical Simulation in Virtual Environments. IEEE Trans Vis Comput Graph. 2012;18(4):617–25.
4. Fortmeier D, Mastmeyer A, Handels H. GPU-based Visualization of Deformable Volumetric Soft-Tissue for Real-time Simulation of Haptic Needle Insertion. In: Bildverarbeitung für die Medizin. Berlin: Springer; 2012. p. 117–22.
5. Rössler F, Wolff T, Ertl T. Direct GPU-based Volume Deformation. In: Proceedings of Curac 2008. Leipzig; 2008. p. 65–8.
6. Gibson SF. 3D Chainmail: A Fast Algorithm for Deforming Volumetric Objects. In: Proc Interactive 3D Graphics. New York: ACM; 1997. p. 149–ff.
7. Lundin K, Ynnerman A, Gudmundsson B. Proxy-based Haptic Feedback from Volumetric Density Data. In: Eurohaptics Conference. United Kingdom: University of Edinburgh; 2002. p. 104–9.
8. Bartz D, Gürvit. Haptic Navigation in Volumetric Datasets. In: Proc PHANToM Users Research Symposium. Konstanz: Hartung-Gorre; 2000. p. 43–7.