International Conference on Artificial Reality and Telexistence
Eurographics Symposium on Virtual Environments (2015)
M. Imura, P. Figueroa, and B. Mohler (Editors)

Bimanual Haptic Simulation of Bone Fracturing for the Training of the Bilateral Sagittal Split Osteotomy

Thomas C. Knott[1] and Torsten W. Kuhlen[2]

[1]Virtual Reality Group, RWTH Aachen University, Germany
[2]Jülich Supercomputing Center, Germany

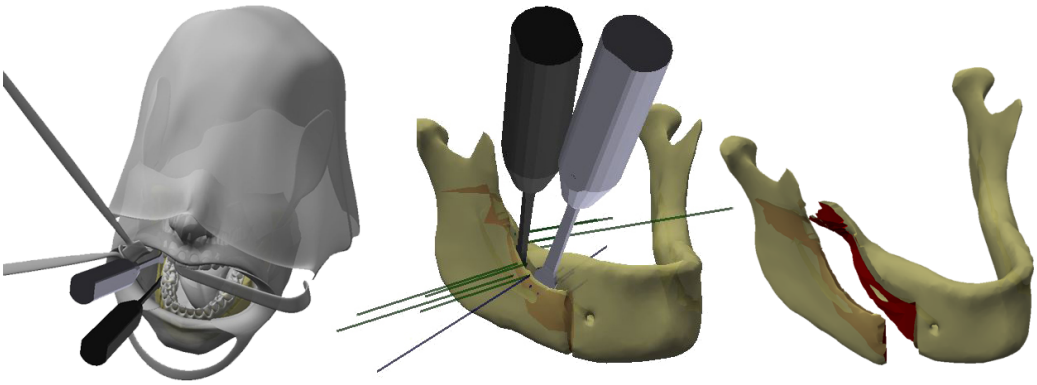

Figure 1: *(Left) Virtual BSSO surgery scenario. (Middle) Surgical chisels inserted into a pre-sawed osteotomy line. The applied load is represented as blue and green lines, and resulted in a partial fracture visible in red inside the bone. (Right) Mandible fragments after a successful fracture.*

Abstract

In this work, we present a haptic training simulator for a maxillofacial procedure comprising the controlled breaking of the lower mandible. To our knowledge the haptic simulation of fracture is seldom addressed, especially when a realistic breaking behavior is required. Our system combines bimanual haptic interaction with a simulation of the bone based on well-founded methods from fracture mechanics. The system resolves the conflict between simulation complexity and haptic real-time constraints by employing a dedicated multi-rate simulation and a special solving strategy for the occurring mechanical equations. Furthermore, we present remeshing-free methods for collision detection and visualization which are tailored for an efficient treatment of the topological changes induced by the fracture. The methods have been successfully implemented and tested in a simulator prototype using real pathological data and a semi-immersive VR-system with two haptic devices. We evaluated the computational efficiency of our methods and show that a stable and responsive haptic simulation of the fracturing has been achieved.

Categories and Subject Descriptors (according to ACM CCS): H.5.2 [Information Interfaces and Presentation]: User Interfaces—Haptic I/O, Benchmarking

1 Introduction

Training opportunities for invasive surgery are usually limited. Therefore, the use of virtual reality-based simulators has become an accepted alternative to traditional methods [CMJ11]. In this paper we present a haptic training simulator dedicated to the learning of a maxillofacial procedure. The Bilateral Sagittal Split Osteotomy (BSSO) according to Obwegeser/Dal Pont [Obw63, Dal58] allows the displacement of the lower jaw and is performed, for instance, in case of a strong pathological under- or overbite. The procedure is carried out in an intraoral approach and starts with the creation of an osteotomy line using a saw or burr (see Fig.2 b). This is followed by a controlled breaking of the mandible by a reversed twisting of one or *two* chisels inserted into the line (see Fig.2 c,d). Afterwards the fragments are rearranged and fixed again. A formal hazard

DOI: 10.2312/egve.20151316

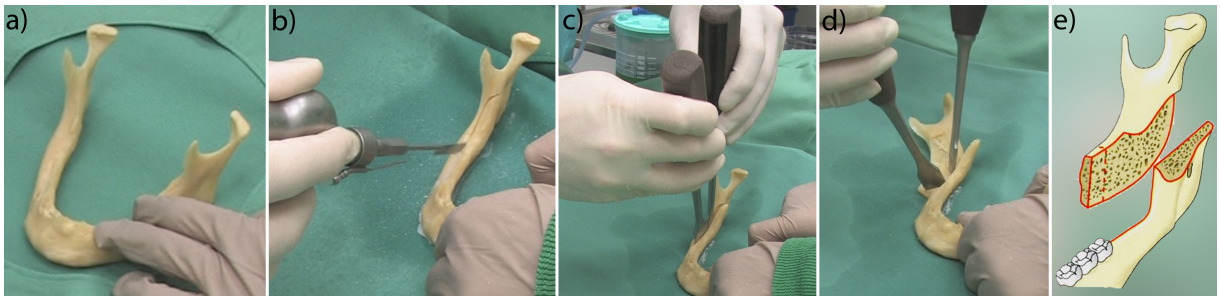

Figure 2: *BSSO according to Obwegeser/Dal Pont [Obw63, Dal58] performed on a mandible specimen: a) mandible, b) sawing of the cracking line, c) chisels inserted into the line, d) successfully broken mandible after twisting of chisels, e) scheme of fragments.*

analysis detailed in [SKD*13] shows that the first two steps are the most delicate ones. In cases of complications, the mandible and the contained neurovascular bundle can be damaged beyond repair with severe consequences. Opposed to the training of sawing or drilling, the manual breaking of bone is seldom addressed by simulators therefore we focus on this part of the procedure.

To this end, we propose a bimanual haptic simulator which allows to train the manual skills needed to perform the split. For the simulation of the behavior of the bone we employ well-founded methods from fracture mechanics by utilizing a framework based on the extended finite element method (XFEM). The main contribution of this paper is the integration of the methods into a haptic simulation and to handle the conflict between the computational complexity and the needed high haptic update rate. To this end, we propose a specialized multi-rate simulation approach and an efficient strategy to solve the occurring mechanical equations. We furthermore introduce remeshing-free techniques for collision detection and visualization which are tailored towards an efficient treatment of the topological changes induced by the fracture.

The remainder of the paper is structured as follows: after the description of the related work, the proposed haptic simulation is presented in section 3. In section 4, we detail our methods for collision detection and visualization. Followed by a description of the virtual medical scenario in section 5 and the evaluation of the simulator in section 6.

2 Related Work

Haptic medical simulations have been created for a vast amount of applications ranging from fundamental techniques to complex surgeries. A good general overview can be found in [CMJ11]. When it comes to surgeries involving hard-tissue as bone, the simulations often address the tasks of drilling or sawing [MSB*06, LWW*14, WCL*14]. Other applications focus more on surgical planing [SSS*14, SIA*09] or the fixation of bone fragments after fracture [BSA07]. To our knowledge, the simulation of the fracturing itself is seldom addressed, especially using well-founded mechanical models, as we do in our simulator.

Nevertheless, we employ techniques related to the ones employed in other simulators. Considering haptic simulation, most rely on a virtual coupling approach to increase the rendering stability [AH99]. The methods can be distinguished into quasi-static approaches, which have no real notion of time, e.g., [BJ08, PDC11] or ours, and approaches incorporating dynamics like [DD06, OTSG09]. The simulations also differ in the way contacts are handled. Some use mathematical bilateral [ZS95] or unilateral [DD06, OTSG09] constraints, others, employ penalty-based approaches, as for instance [WM03, BJ08] or us. Applications simulating tissue deformations often apply the finite element method (FEM) due to its profound mechanical model. For performance reasons, a linear elastic constitutive law is often employed and the FEM equations are updated during the simulation using the so-called co-rotational method to handle large deformations [PDC11, DPD*13]. Such an update is not sufficient when it comes to topological changes in the tissue. When it is cut or it fractures, the mechanical equations need to reflect the dissections. To this end, some cutting simulations perform a remeshing of the employed simulation mesh, as for instance [DGW10], who employ a regular hierarchical-grid for efficient computations. [JK09] spare a remeshing by employing the extended finite element method (XFEM) where the existing mesh is enriched with additional degrees of freedom. Our simulator employs this method as well, but with some significant differences to reflect the additional simulation of the propagation of the crack. Being quite complex, soft-tissue simulation can be too time consuming to be performed on a haptic update rate. Some methods circumvent the problem by reducing the complexity of the objects like [BJ08, DDCB01]. In case high complexity is necessary, often so-called multi-rate approaches are employed. Here, the computationally expensive parts are performed asynchronously on a lower frequency and the fast haptic simulation uses some kind of intermediate representation [OL08]. The latter can be mechanical contact representations, which reflect the actual compliance of the soft-tissue [DD06, DPD*13]. As these are computationally quite expensive for large simulation meshes or complex contact situations, others utilize less expensive geometric representations. In [UMK12], for instance, a simple plane is used as proxy for the haptic simulation which is updated on a slower rate. We also employ a geometric rep-

resentation but use the full geometry of the scene objects as proxy. Furthermore, our multi-rate approach differs from the described ones by having additional components dedicated to the simulation of the crack propagation.

In summary, the realistic haptic simulation of fracturing bone is still an open issue. The main contribution of this paper is therefore an integrated haptic simulator which includes a well-founded mechanical treatment of the breaking bone and addresses the conflict between simulation complexity and haptic real-time constraints.

3 Bimanual Haptic Simulation of Fracture

During the procedure, the surgeon applies via the surgical tools a load on the bone which deforms and finally breaks. To model this behavior, which can be seen as a crack propagation problem, we apply methods from fracture mechanics to simulate the reaction of the bone and how the crack propagates. In our simulator we employ for these tasks a software framework based on the so called *extended finite element method* (XFEM) [BF12, Bay11]. After a brief description of the framework in the following, we will introduce the methods applied for the simulation of the surgical tools and their coupling to the user input. Both are then integrated into one mechanical system utilizing a penalty-based collision response. The resulting mechanical equations contain non-linearities which are treated using a proposed efficient solving strategy. The latter is then integrated into a multi-rate simulation to achieve a stable and responsive haptic interaction.

3.1 Mechanical Model of Bone and Fracturing

The simulation of the bone and its fracture is done within the XFEM framework in two consecutive steps: first, the deformation- and the stress-fields are calculated for a given load which are then used in the second step to decided whether the crack propagates and if, how far and in which direction. For the latter, an *explicit crack representation* (ECR) based on a triangle mesh is used. This describes the crack shape in an undeformed state. The further propagation of the crack is calculated using the *maximum strain energy release rate propagation criterion* (MSERRC) [BF12], resulting in the extension of the ECR. To this end, the deformation field inside the bone needs to be calculated for a load under consideration of the current state of the crack. Applying the classical FEM for this task would mean that the simulation mesh needs to be remeshed along the crack to create the necessary degrees of freedom (DOF) so that both sides of the crack can move apart. This is circumvented in the XFEM. Here, no remeshing is performed and so-called enrichments provide the needed additional DOFs at the *existing* nodes of the simulation mesh. These DOFs are integrated into the classical definition of the FEM deformation field by additional terms:

$$\mathbf{u}(\mathbf{x}) = \sum_{i \in I} N_i(\mathbf{x}) u_i + \sum_{i \in I^{cut}} N_i(\mathbf{x}) \Big[H(\mathbf{x}) - H(\mathbf{x}_i) \Big] a_i, \quad (1)$$

where $u(x)$ defines the displacement for every point x in the undeformed domain $\mathcal{D}$. The first sum is the classic FEM term (see e.g. [ESHD05]), which is applied for each node i of the total node set I with the undeformed positions x_i. Here, the deformations u_i of the DOFs from the FEM are interpolated using the standard linear FEM shape functions N_i. To allow the crack to open up, the deformation field $u(x)$ must be able to contain discontinuities. This is done by the second sum which includes the additional enrichment DOFs a_i for all nodes I^{cut} affected by the crack. The discontinuities in $u(x)$ are created at this by the discontinuous global enrichment function $H(\mathbf{x})$. This evaluates to plus or minus one depending on which side of the crack x is located. To be able to efficiently integrate the ECR into (1), an *implicit crack representation* (ICR) based on level-set functions is employed. To this end, subsets of $\mathcal{D}$ are defined for areas on a "positive" and a "negative" side of the crack as $\mathcal{A}$ respectively $\mathcal{B}$. The most important level set for us is $\Phi(x)$ which is defined as a signed distance function w.r.t. ECR:

$$\Phi(x) = \begin{cases} +dist(x, \text{ECR}) & \text{if } x \in \mathcal{A} \\ -dist(x, \text{ECR}) & \text{if } x \in \mathcal{B} \end{cases} \quad (2)$$

The level sets are discretized on $\mathcal{D}$ via the nodes I and interpolated using N_i. They are used to calculate $H(p)$, determine I^{cut}, and perform the numerical integration resulting in a linear approximation of the internal force of the bone:

$$\boldsymbol{F}_{\text{XF}}(\boldsymbol{x}_{\text{XF}}) \approx \frac{\partial \boldsymbol{F}_{\text{XF}}}{\partial \boldsymbol{x}_{\text{XF}}} \boldsymbol{x}_{\text{XF}}, \quad (3)$$

where $\boldsymbol{x}_{\text{XF}} = [u,a]^T$ are the stacked DOFs including the enrichments. For details on the whole process we refer the interested reader to [Bay11]. Now, the behavior of the bone is computed in a quasi-static approach, thus there is no real notion of time and effects like inertia and damping are neglected. For a fixed load $\boldsymbol{F}_{\text{XF}}$ this would be done by iterating the following steps:

1. Compute the linear approximation (3) for the current crack.
2. Calculate deformation $\boldsymbol{x}_{\text{XF}}$ by solving $\frac{\partial \boldsymbol{F}_{\text{XF}}}{\partial \boldsymbol{x}_{\text{XF}}} \boldsymbol{x}_{\text{XF}} = -\boldsymbol{F}_{\text{XF}}$
3. Propagate crack based on $\boldsymbol{x}_{\text{XF}}$ using MSERRC.

Nevertheless, in our simulation $\boldsymbol{x}_{\text{XF}}$ is not fixed but created via the surgical tools as described in the following.

3.2 Surgical Tool Simulation and Collision Response

The trainee should be able to control the virtual surgical tools, i.e., two chissels, simultaneously via two haptic devices and interact with the bone (see Fig.2 c, d) to apply the forces mentioned above. To couple the tools to the haptic devices we follow the usual virtual coupling (VC) approach to increase stability and haptic rendering quality [AH99]. To this end, we model the virtual tools T_1 and T_2 as rigid bodies and couple them to the input configurations of the devices H_1 respectively H_2 via virtual 6DOF springs. These create coupling forces and torques $\boldsymbol{F}_{\text{VC}}^{T_1}$ and $\boldsymbol{F}_{\text{VC}}^{T_1}$ trying to align the states of the tools $\boldsymbol{x}_{T_1}, \boldsymbol{x}_{T_2}$ with the corresponding inputs $\boldsymbol{x}_{H_1}, \boldsymbol{x}_{H_2}$. To reflect the limited capabilities of the haptic devices we follow the approach of [BJ08] and model $\boldsymbol{F}_{\text{VC}}$ using non-linear functions which

saturate to an upper bound when the force magnitudes reach the maximal displayable values. Furthermore, we simulate the interactions between the virtual objects, i.e., the collision response (CR), with the help of penalty forces $\boldsymbol{F}_{\mathrm{CR}}$ [ESHD05]. These are utilized to model non-penetration constraints, as well as to simulate friction between the objects in contact. For the latter, we employ an approach similar to [MZ90] by using a linear friction model. In order to be compatible to the XFEM framework, we use a quasi-static simulation approach for the whole simulation. To incorporate the simulation of the bone, we combine (3) with the coupling and collision forces by forming a composed mechanical equation system:

$$\left|\begin{array}{rl} \boldsymbol{F}_{\mathrm{XF}}(\boldsymbol{x}_{\mathrm{XF}}) + \boldsymbol{F}^{\mathrm{XF}}_{\mathrm{CR}}(\boldsymbol{x}_{\mathrm{XF}},\boldsymbol{x}_{\mathrm{T}_1},\boldsymbol{x}_{\mathrm{T}_2}) = 0 & \text{(4a)} \\ \boldsymbol{F}^{T1}_{\mathrm{VC}}(\boldsymbol{x}_{\mathrm{T}_1},\boldsymbol{x}_{\mathrm{H}_1}) + \boldsymbol{F}^{T1}_{\mathrm{CR}}(\boldsymbol{x}_{\mathrm{XF}},\boldsymbol{x}_{\mathrm{T}_1},\boldsymbol{x}_{\mathrm{T}_2}) = 0 & \text{(4b)} \\ \boldsymbol{F}^{T2}_{\mathrm{VC}}(\boldsymbol{x}_{\mathrm{T}_2},\boldsymbol{x}_{\mathrm{H}_2}) + \boldsymbol{F}^{T2}_{\mathrm{CR}}(\boldsymbol{x}_{\mathrm{XF}},\boldsymbol{x}_{\mathrm{T}_1},\boldsymbol{x}_{\mathrm{T}_2}) = 0 & \text{(4c)} \end{array}\right|$$

where the $\boldsymbol{F}^{i}_{\mathrm{CR}}$ functions denote the penalty forces on the objects $i = \mathrm{XF},\mathrm{T}_1,\mathrm{T}_2$ with respect to the overall state (we will give further information on this later on). The whole system needs to be solved to calculate the states of the tools and the bone for a given input. $\boldsymbol{x}_{\mathrm{H}_1},\boldsymbol{x}_{\mathrm{H}_2}$.

3.3 Relaxation-based Solving Strategy

Equation system (4) is inherently non-linear due to the penalty forces, which depend on the current contact situation, and the coupling term $\boldsymbol{F}_{\mathrm{VC}}$. Furthermore, [BJ08] show that the saturation of the latter can create singularities in the equation system which have to be taken care of. To this end, they employ a special solving process based on a singular value decomposition (SVD). However, performing a SVD on the whole system (4) would create severe performance issues due to the size of the XFEM part. In order to resolve this conflict, we employ a relaxation-based solving approach in fashion of a non-linear Gauss-Seidel process [Rhe98]. To this end, we separate the system into two parts: (i) equation (4a) and (ii) equation (4b) and (4c). These are then linearized and solved alternatingly under consideration of the latest results from the other part:

1. Perform CD query
2. Setup and solve:
$$\frac{\partial \boldsymbol{F}_{\mathrm{XF}}}{\partial \boldsymbol{x}_{\mathrm{XF}}}\boldsymbol{x}_{\mathrm{XF}} = -\boldsymbol{F}^{CR}_{\mathrm{XF}} \tag{5}$$
3. Update CD with $\boldsymbol{x}_{\mathrm{XF}}$
4. Perform CD query
5. Setup and solve:
$$\begin{bmatrix} \frac{\partial \boldsymbol{F}^{\mathrm{T1}}_{\mathrm{VC}}}{\partial \boldsymbol{x}_{\mathrm{T}_1}} + \frac{\partial \boldsymbol{F}^{\mathrm{T1}}_{\mathrm{CR}}}{\partial \boldsymbol{x}_{\mathrm{T}_1}} & \frac{\partial \boldsymbol{F}^{\mathrm{T1}}_{\mathrm{CR}}}{\partial \boldsymbol{x}_{\mathrm{T}_2}} \\ \frac{\partial \boldsymbol{F}^{\mathrm{T2}}_{\mathrm{CR}}}{\partial \boldsymbol{x}_{\mathrm{T}_1}} & \frac{\partial \boldsymbol{F}^{\mathrm{T2}}_{\mathrm{VC}}}{\partial \boldsymbol{x}_{\mathrm{T}_2}} + \frac{\partial \boldsymbol{F}^{\mathrm{T2}}_{\mathrm{CR}}}{\partial \boldsymbol{x}_{\mathrm{T}_2}} \end{bmatrix} \begin{bmatrix} d\boldsymbol{x}_{\mathrm{T}_1} \\ d\boldsymbol{x}_{\mathrm{T}_2} \end{bmatrix} = - \begin{bmatrix} \boldsymbol{F}^{\mathrm{T1}}_{\mathrm{VC}} + \boldsymbol{F}^{\mathrm{T1}}_{\mathrm{CR}} \\ \boldsymbol{F}^{\mathrm{T2}}_{\mathrm{VC}} + \boldsymbol{F}^{\mathrm{T2}}_{\mathrm{CR}} \end{bmatrix}$$
6. Update CD with $\boldsymbol{x}_{\mathrm{T}_i} = \boldsymbol{x}_{\mathrm{T}_i} + d\boldsymbol{x}_{\mathrm{T}_i}$ for $i = 1,2$
7. Test convergence of $\boldsymbol{x}_{\mathrm{XF}},\boldsymbol{x}_{\mathrm{T}_1},\boldsymbol{x}_{\mathrm{T}_2}$ and repeat when necessary.

Here, the right hand sides in step 2. and 5. denote the values of the corresponding functions evaluated with the latest $\boldsymbol{x}_{\mathrm{XF}},\boldsymbol{x}_{\mathrm{T}_1},\boldsymbol{x}_{\mathrm{T}_2}$. Moreover, a detailed description of the mechanical equations in step 5. can be found in [KLK12]. In order to calculate $\boldsymbol{F}^{\mathrm{XF}}_{\mathrm{CR}}$, which defines the contact forces at the DOFs of the bone, a geometric mapping is used as detailed in section 4.1. Finally, an ad-hoc convergence test as described in [ESHD05] is applied in the last step.

Due to the splitting, we can use, in step 5., the mentioned SVD based strategy and in step 2., a solver suited for the sparse XFEM equations. Experiments with iterative solvers as the conjugate gradient method showed that they do not perform well on the stiff XFEM equations. Therefore, we switched to direct solvers and finally utilized one based on a LLt factorization. Notice that equation (5) in step 2. does not correspond to the full linearization of (4a) as it omits the partial derivative of $\boldsymbol{F}_{\mathrm{CR}}$. Thereby, the matrix of (5) does not depend on the contact situation but only on the state of the bone. Hence, it stays the same until $\boldsymbol{F}_{\mathrm{XF}}$ needs to be updated after the next propagation of the crack. As result, the factorization of the solver can be reused until then. Since the factorization is by far more expensive than the solving, this brings a performance gain outweighing a possibly reduced convergence due to the omitted term. In fact, by employing a warm starting of the process from the previous results, it converged in our experiments (see sec.6) to a solution of the original equation (4) within 5 iterations.

3.4 Haptic Multi-rate Simulation

In order to allow a stable and responsive haptic simulation, the outputted forces, $\boldsymbol{F}^{\mathrm{T1}}_{\mathrm{VC}},\boldsymbol{F}^{\mathrm{T2}}_{\mathrm{VC}}$, need to be computed for the current input $\boldsymbol{x}_{\mathrm{H}_1},\boldsymbol{x}_{\mathrm{H}_2}$ on a high update rate, commonly assumed to be around 1kHz. However, performing the process described in the last section in each haptic step, which would actually be required for this, exceeds the corresponding upper bound of 1ms (see sec.6). To resolve this conflict we follow the idea of a multi-rate simulation [OL08]. Here, the full simulation is performed on a lower frequency while the parts important for the haptic quality are done on the full rate of 1kHz.

To this end, we utilize multiple simulation loops running concurrently (see Fig.3). In a main loop *ML*, runnig at around 50Hz, we update all mechanical DOFs, $\boldsymbol{x}_{\mathrm{XF}},\boldsymbol{x}_{\mathrm{T}_1},\boldsymbol{x}_{\mathrm{T}_2}$, using the process described in the last section. Additionally, we have a haptic loop *HL*, running at 1kHz. Here, we have duplicates of the mechanical states of the tools $\boldsymbol{x}^{HL}_{\mathrm{T}_1},\boldsymbol{x}^{HL}_{\mathrm{T}_2}$. These are updated in *HL* by performing steps 4.-6. using the latest converged results of $\boldsymbol{x}_{\mathrm{XF}}$ from *ML* as input. In doing so, we simulate their interaction with each other and the current deformation state of the bone. Afterwards, the resulting $\boldsymbol{F}^{\mathrm{T1}}_{\mathrm{VC}},\boldsymbol{F}^{T2}_{\mathrm{VC}}$ are send to the haptic devices and the process repeats. Furthermore, to prevent a divergence of the tool states between both loops, we use the latest $\boldsymbol{x}^{HL}_{\mathrm{T}_1},\boldsymbol{x}^{HL}_{\mathrm{T}_2}$ in *ML* to warm start the process. For a stable and smooth haptic simulation it is important that *ML* and *HL* maintain steady high update rates. Therefore, we perform the crack propagation in a third concurrent loop *PL*. Here, we first check whether propagation takes place based on the latest converged results from *ML*. If this is not the case,

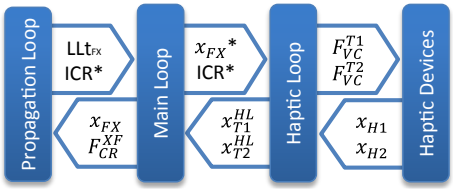

Figure 3: *Scheme of the multi-rate simulation and the exchanged data. The star means that the information is transfered in form of an update of the collision detection.*

we wait for the next results and start over again. In case of propagation, new explicit and implicit crack representations (ECR and ICR) are created followed by the computation of a new linearization $\partial \boldsymbol{F}_{\mathrm{XF}}/\partial \boldsymbol{x}_{\mathrm{XF}}$. For which we then create an $\mathrm{LLt}_{\mathrm{XF}}$-factorization and provide it to *ML*.

4 Remeshing-free Collision Detection and Visualization

When the mechanical state of the bone and the crack changes during the simulation, collision detection (CD) and visualization (VS) have to be updated accordingly. In the following we propose methods which are tailored to take advantage of the representations used by the XFEM framework to perform efficient updates. Taking up the idea of the remeshing free simulation from the XFEM, our proposed CD and VS methods also avoid the remeshing of the employed object representations. Before introducing the individual methods, we will describe an efficient way to embed sampling points into the XFEM simulation mesh and keep them consistent with its changing state during the simulation, which is shared by both methods.

4.1 Geometric Mapping

The deformed position p^d of a sampling point $p \in \mathcal{D}$ is determined by the XFEM deformation field (1). As p^d needs to be calculated for many points and several times per second, we need a fast way to do this. To this end, we will describe in the following an efficient update strategy. First, we determine, for each tetrahedron $T = \{i_1,..,i_4\}$, all sampling points p_j that it encloses. The deformed position p_j^d of the points can then be computed according to (1) by:

$$p_j^d = \sum_{i \in T} \underbrace{(u_i + [H(p_j) - H(x_i)]a_i)}_{d_i(p_j)} N_i(p_j). \tag{6}$$

Note that for a given state u_i, a_i of T, (6) varies for all contained points only in the terms $N_i(p_j)$ and $H(p_j)$. As $H(p_j)$ can only evaluate to plus or minus one, each $d_i(p_j) \in \mathbb{R}^3$ can also have only two different values d_i^+ for $p_j \in \mathcal{A}$ and d_i^- for $p_j \in \mathcal{B}$. We stack these into $D_T^+ = [d_{i_1}^+,..,d_{i_4}^+] \in \mathbb{R}^{3x4}$ (analogously for D_T^-) and compute p_j^d by:

$$p_j^d = \begin{cases} D_T^+ N_{p_j} & \text{if } p_j \in \mathcal{A} \\ D_T^- N_{p_j} & \text{if } p_j \in \mathcal{B} \end{cases} \tag{7}$$

where $N_{p_j} = [N_{i_1}(p_j),..,N_{i_4}(p_j)]^T \in \mathbb{R}^4$. Hence, we perform a matrix-vector product utilizing the SIMD capabilities of the CPU. During the simulation, D_T^+, D_T^- have to be recomputed every time the state of the bone changes. But as they only depend on T we can reuse them to deform all the sampling points used by CD and VS which are enclosed by the same T. Furthermore, N_{p_j} needs only to be computed once for each sampling point during the simulation. As for each point the embedding tetrahedron must be found, we utilize a spatial hashing [THM*03] to speed up the task. Finally, we also need to determine if a point p_j is inside $\mathcal{A}$ or $\mathcal{B}$. This can be done by evaluating $H(p_j)$ or rather the sign of $\Phi(p_j)$ (see (2)). As mentioned before, Φ is provided by the XFEM framework in a discretized form based on the simulation mesh and the shape functions N_i so that we can calculate:

$$\Phi(p_j) = \underbrace{[\Phi(x_{i_1}),..,\Phi(x_{i_4})]}_{\Phi_T \in \mathbb{R}^4} N_{p_j}. \tag{8}$$

The determination to which side a point belongs, is done only on propagation and for points which are enclosed by tetrahedrons influenced by the crack. Notice, that Φ_T needs to be setup only once for each tetrahedron making the evaluation of (8) a fast $\mathbb{R}^4$-scalar product for each contained point.

4.2 Collision Detection

The CD must support queries on a haptic rate and facilitate a fast update when the states of the objects change. This is especially challenging for the CD between the tools and the bone due to the deformations and topological changes of the latter. We employ a representation based on *signed distance fields* (SDF) for the tools, which is rather standard, and details can be found in [ESHD05]. For the bone, we use a representation via a *point shell* (PS) [BJ08] where the surface is sampled with a set of points. This brings along two advantages: (First) sparing any topology, no remeshing is required when the state of the bone changes. Thus, solely an update of the points as described in the last section is necessary. (Second) a point is always completely contained in one tetrahedron which therefore forms a valid bounding volume (BV) for all contained points (this would not hold, e.g., for triangles).

In a straight forward CD test, the SDF would be evaluated for all PS points to compute inter-penetrations. As this would take too long, we cull unnecessary evaluations utilizing a search data-structure for the PS based on bounding volumes (BV). To be able to efficiently update the BVs when the state of the bone changes, we bring the second mentioned advantage into operation. Using the tetrahedrons of the simulation mesh as BVs, their update can be computed directly from the state of the bone without deforming the PS itself. For an unenriched tetrahedron $T = i_1,..,i_4$, we use one BV which is defined via the nodal values $u_{i_1},..,u_{i_4}$. If T is enriched, we employ two BVs for T, one for each side of the crack. To define their extent we take a closer look on (7). Assuming, w.l.o.g., $p_j \in \mathcal{A}$, p_j^d results from the linear interpolation of the columns of D_T^+ using N_j^p. As the values of the latter are in $[0..1]$, any p_d will always be located in the tetrahedron formed by the columns

of D_T^+ taken as nodes. The according BVs for both sides of the crack are organized together with the other BVs in a spatial hash table (HT) [THM*03] to speed up CD tests. For the latter the HT is queried for colliding BVs using the bounding boxes of the surgical tools. Each colliding BV is associated with a list of PS points p_j. First, these are deformed using (7) and then, transformed to the local coordinate system of the SDF using a matrix M_{CT}. Concatenated, we do this by:

$$p_j^{SDF} = M_{CT} D_T^+ N_{p_j} \quad if\ p_j \in \mathcal{A} \tag{9}$$

and analogously for $p_j \in \mathcal{B}$. As M_{CT} and D_T^+ stay the same for all points contained in one BV, a matrix $M = M_{CT} D_t \in \mathbb{R}^{4x4}$ can be computed and used to perform both operations with a single matrix-vector product for all enclosed points. We want to emphasize that only at this stage and only for the involved points, the actual update of the PS is performed. Furthermore, as this is combined with the necessary coordinate transformation, very little costs occur for the update of the PS.

As the CD is embedded in our multi-rate simulation, the update of the HT is performed in *ML* concurrently to the queries in *HL*. To prevent a blocking of the latter during an update, a second version of the HT is updated by *ML* in the background and afterwards exchanged with the one used by *HL* in a glance. To speed up the overall CD process, we furthermore perform the collision tests for each object pair in parallel.

4.3 Visualization

For the visualization we developed a method based on constructive solid geometry (CSG) operations. To avoid an explict remeshing, the operations are performed on the GPU in the image space. To this end, we use two different meshes, one describing the surface of the bone (BM) and one containing an explicit representation of the crack (CM). The nodes of the BM are updated similarly to the point shell points using the geometrical mapping from section 4.1. They will move according to the deformations and breaking of the bone. Since we do not perform any cutting or remeshing, the topology of BM remains the same and no gap will be visible at this point (see Fig.4 a). To this end, we use the CM which is created from the explicit crack representation (ECR) from the XFEM framework. While the ECR is defined in the undeformed domain $\mathcal{D}$, we need for the visualization a crack which deforms and is able to open (see Fig.4 c). Therefore, we use two duplicates of the ECR mesh for visualization and associate each with one side of the crack with respect to the geometrical mapping (see sec.4.1). Thus, when the displacement field is discontinuous, i.e., a crack opens, both side will move apart. Additionally, as we need closed volumes for the CSG operation, the borders of the two duplicates are connected to each other by further triangles to close the gap (see Fig.4 b). To create the final visualization showing the topological changes of the opening bone, CM is subtracted from BM using a CSG operation (see Fig.4 c). This would normally require an expensive and error prone remeshing process. Instead we developed an approach doing this operation during the rendering of the meshes in the image space.

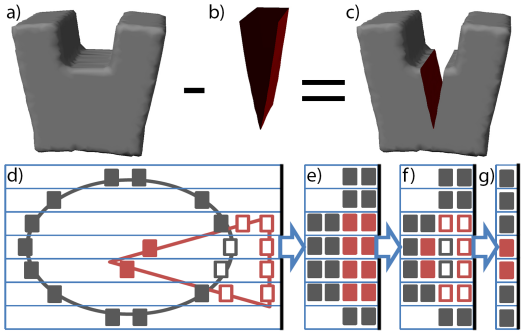

Figure 4: *(Top row) CSG example: Mesh b) is subtracted in image space from mesh a) to create rendering of notch c). (Bottom row) A-Buffer-CSG 2d example: d) the triangle is subtracted from the ellipse during the rendering to the image plane (black line). Created fragments are denoted as boxes. e) Unsorted fragments in A-Buffer. f) Sorted and filtered fragments (unfilled boxes are discarded). g) Resulting pixels.*

To this end, we utilize the capabilities of modern GPUs and the so-called A-buffer concept [Car84], a technique typically employed to render order independent transparency (OIT) [YHGT10]. Here, the fragments generated during the rendering process are not directly stored into the image buffer. Instead, we collect them for each pixel in the A-buffer including the information to which object they belong (see Fig.4 e). After the complete scene is rendered, the fragments of each pixel are sorted according to their depth (see Fig.4 f). Then, we traverse the fragments of each pixel from front to back and keep track of which objects have been entered and left. Thereby, it is possible, during the traversal, to decide for each fragment whether it is located within another object and in which. Based on this information, the CSG operation can be performed by discarding the fragments that have to be cut out. In our scenario, fragments originating from BM are removed if they are located inside CM, while fragments from CM are only kept when they are within BM (see Fig.4 f). The latter is necessary to draw the sides of the crack (see Fig.4 c). After this process, the topmost fragments are copied into the image buffer (see Fig.4 g). Or, in case transparency is wanted, the remaining fragments of each pixel are blended to compute the final pixel color. On the top-right of Figure 4 an according result for an abstract notch can be seen.

5 Bimanual Haptic Simulator Prototype for the BSSO

The development of the whole simulator was done in close cooperation with medical experts utilizing formal methods [SKD*13]. The medical scenario was created from the CT scan of a real pathological patient case. A segmentation served as basis to create a volumetric mesh for the XFEM simulation (553 vertices, 1904 tetrahedrons), a point shell (104567 points) and a surface mesh (9989 vertices, 20000

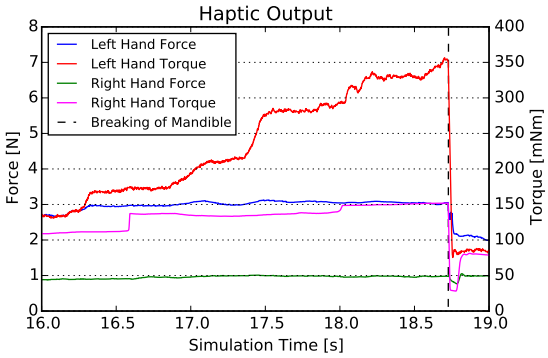

Figure 5: *Plot of the haptic force and torque magnitudes.*

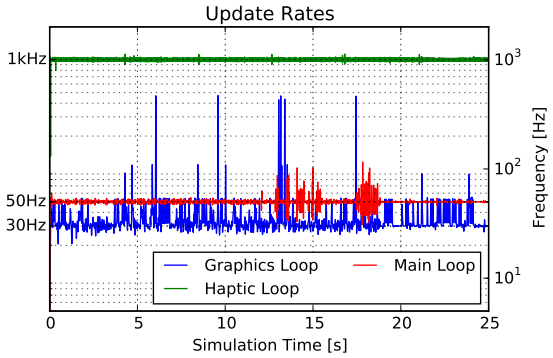

Figure 6: *Plot of the main update frequencies.*

faces). Furthermore, we added a model of a head in procedure pose including further surgical instruments (see Fig.1 left) to reflect the surgeons restricted physical access to the mandible during the procedure. The material properties employed by the XFEM framework are based on literature [SDD03] and differentiate between multiple regions as detailed in [Bay11]. The described concurrency and parallelism was implemented using a software framework based on data-flow concepts [KWHK14]. We employed a stereoscopic monitor SD2220W (21.6″, 1920x1080 pixels) from Planar Systems and two haptic devices. We used one Phantom Premium 1.5/6DOF and, due to the absence of a second 6DOF device, one Phantom Omni®. These were connected to a desktop PC with an Intel® Xeon® E5-1650 v2 CPU(6 cores, 3.5 GHz) with 12 GB of RAM and a NVIDIA GeForce® GTX- 480.

6 Results and Discussion

For the evaluation we let a maxillofacial surgeon expert in the BSSO procedure perform the split with our simulator and recorded his input. Screenshots of the simulation taken during and after the procedure are shown in Figure 1. On the right, one can see the bone fragments resulting from the operation. The magnitudes of the rendered haptic forces and torques are depicted in Figure 5 for a sequence around the moment of break. Beside the stability of the haptic rendering, one can see how particularly the torque magnitudes rise when the surgeon increases the load on the bone by twisting the chisels. Then, when the mandible breaks and moves apart, there is a sudden loss of resistance and the values abruptly decrease.

The recorded trajectories were furthermore used to measure the performance of our simulator. Most crucial are the update rates of the key components depicted in 6. Although the main loop, *ML*, drops at some points below the aimed 50Hz it never sinks below 30Hz and ensures a highly responsive simulation. Furthermore, the maintained critical haptic frequency of 1kHz allows a stable haptic simulation. A fluid visualization is provided via a graphics update rate of 30Hz. As the propagation loop, *PL*, is not an uniform process with a fixed frequency, its performance data is detailed in Table 1 with a breakdown into the comprised steps. In case propagation takes place, the

Component	Mean	SD	Min	Max
Main loop step	4.33	1.27	3.32	15.93
Haptic loop step	0.45	0.16	0.08	0.95
CSG rendering	1.81	0.21	1.11	2.51
Prop. loop step	9.45	15.46	0.12	122.94
- Crack propagation	7.85	13.79	0.10	75.33
- F_{XF} Linearization	65.92	12.84	41.24	83.27
- LLt Factorization	8.41	1.43	6.32	10.96
Update of vis. rep.	0.24	0.05	0.17	0.44
Update of CD rep.	2.20	0.34	1.84	3.22
Collision detection	0.43	0.17	0.00	0.89

Table 1: *Computation times for the main parts in milliseconds.*

overall time is dominated by the propagation computation itself and the linearization of F_{XF}. Another import factor is the fast update of the visualization and the collision detection with respect to deformations and propagation. Table 1 shows that these tasks are performed within few milliseconds. Finally, the time for a collision query always stayed below 0.9ms. This is crucial, as it is part of each *HL* step and would otherwise lead to a violation of the upper bound of 1ms for *HL*. Without the parallelization of the queries (see sec.4.2) the maximal time measured for a query increased to 1.64ms (mean 0.75ms, std 0.38ms) which would lower the haptic frequency drastically. In summary, these results show that interactive simulation rates and a stable haptic rendering is achieved.

7 Conclusion

In this paper we proposed a haptic simulator for the training of a maxillofacial procedure where the surgeon performs a controlled breaking of a mandible bone. A framework for mechanical fracture simulation has been integrated into a bimanual haptic simulation utilizing a specialized multi-rate approach and a dedicated solving strategy. The evaluation showed that the proposed methods successfully solve the conflict between the computationally expensive fracture computations and the required high haptic update rates. Furthermore, techniques for collision detection and visualization have been presented which efficiently handle the topological changes of the bone without a remeshing of

the involved object representations. Although the presented prototype was developed for a specific procedure based on real pathological patient data, the presented techniques can be transfered to other procedures involving fractures.

Acknowledgments

This project has received funding from the European Union's 7th Framework Programme for research, technological development and demonstration under grant agreement 610425.

References

[AH99] ADAMS R., HANNAFORD B.: Stable haptic interaction with virtual environments. *IEEE Trans. on Robotics and Automation 15*, 3 (June 1999), 465–474. 2, 3

[Bay11] BAYDOUN M.: *A Novel Approach to Three-Dimensional Crack Propagation in XFEM with Application to Bilateral Sagittal Split Osteotomy*. PhD thesis, RWTH Aachen University, 2011. 3, 7

[BF12] BAYDOUN M., FRIES T. P.: Crack propagation criteria in three dimensions using the XFEM and an explicit-implicit crack description. *Analysis* (2012), 1–44. 3

[BJ08] BARBIC J., JAMES D.: Six-DoF Haptic Rendering of Contact Between Geometrically Complex Reduced Deformable Models. *IEEE Trans. on Haptics 1*, 1 (2008), 39–52. 2, 3, 4, 5

[BSA07] BLYTH P., STOTT N. S., ANDERSON I. A.: A simulation-based training system for hip fracture fixation for use within the hospital environment. *Injury 38*, 10 (Oct. 2007), 1197–203. 2

[Car84] CARPENTER L.: The A-buffer, an antialiased hidden surface method. In *ACM SIGGRAPH* (1984), vol. 18. 6

[CMJ11] COLES T. R., MEGLAN D., JOHN N. W.: The Role of Haptics in Medical Training Simulators: A Survey of the State of the Art. *IEEE Transactions on Haptics 4*, 1 (Jan. 2011), 51–66. 1, 2

[Dal58] DAL PONT G.: L'osteotomia retromolare per la correzione della progenia. 1, 2

[DD06] DURIEZ C., DUBOIS F.: Realistic haptic rendering of interacting deformable objects in virtual environments. *IEEE TVCG 12* (2006), 36–47. 2

[DDCB01] DEBUNNE G., DESBRUN M., CANI M.-P., BARR A. H.: Dynamic Real-Time Deformations using Space and Time Adaptive Sampling. In *ACM Proc. Siggraph* (2001). 2

[DGW10] DICK C., GEORGII J., WESTERMANN R.: A Hexahedral Multigrid Approach for Simulating Cuts in Deformable Objects. *IEEE Trans. Visual. Comput. Graph. X*, X (Dec. 2010), 1–15. 2

[DPD*13] DERVAUX F., PETERLIK I., DEQUIDT J., COTIN S., DURIEZ C.: Haptic rendering of interacting dynamic deformable objects simulated in real-time at different frequencies. In *IEEE Int. Conf. Intelli. Robots and Systems* (2013), pp. 2010–2016. 2

[ESHD05] ERLEBEN K., SPORRING J., HENRIKSEN K., DOHLMAN K.: *Physics-based Animation*. Charles River Media, Inc., June 2005. 3, 4, 5

[JK09] JERÁBKOVÁ L., KUHLEN T.: Stable Cutting of Deformable Objects in Virtual Environments Using XFEM. *CGA* (2009). 2

[KLK12] KNOTT T., LAW Y., KUHLEN T.: Stable and Transparent Bimanual Six-Degree-of-Freedom Haptic Rendering Using Trust Region Optimization. In *Haptics: Perception, Devices, Mobility, and Communication*. Springer, Tampere, 2012, pp. 270–281. 4

[KWHK14] KNOTT T. C., WEYERS B., HENTSCHEL B., KUHLEN T. W.: Data-flow Oriented Software Framework for the Development of Haptic-enabled Physics Simulations. In *SEARIS* (Minneapolis, USA, 2014), IEEE. 7

[LWW*14] LIN Y., WANG X., WU F., CHEN X., WANG C., SHEN G.: Development and validation of a surgical training simulator with haptic feedback for learning bone-sawing skill. *Journal of Biomedical Informatics 48* (2014), 122–129. 2

[MSB*06] MORRIS D., SEWELL C., BARBAGLI F., SALISBURY K., BLEVINS N., GIROD S.: Visuohaptic Simulation of Bone Surgery for Training and Evaluation. *IEEE CGA*, 6 (2006). 2

[MZ90] MCKENNA M., ZELTZER D.: Dynamic simulation of autonomous legged locomotion. In *ACM SIGGRAPH* (1990), vol. 24, pp. 29–38. 4

[Obw63] OBWEGESER H.: The indications for surgical correction of mandibular deformity by the sagittal splitting technique. *British Journal of Oral Surgery 1* (1963), 157–171. 1, 2

[OL08] OTADUY M. A., LIN M. C.: Introduction to Haptic Rendering Algorithms. In *Haptic Rendering: Foundations, Algorithms, and Applications*. 2008, pp. 159–180. 2, 4

[OTSG09] OTADUY M. A., TAMSTORF R., STEINEMANN D., GROSS M.: Implicit Contact Handling for Deformable Objects. *Computer Graphics Forum 28*, 2 (Apr. 2009), 559–568. 2

[PDC11] PETERLIK I., DURIEZ C., COTIN S.: Asynchronous haptic simulation of contacting deformable objects with variable stiffness. *Intelligent Robots and Systems* (2011), 2608–2613. 2

[Rhe98] RHEINBOLDT W.: *Methods for solving systems of nonlinear equations*. 1998. 4

[SDD03] SCHWARTZ-DABNEY C. L., DECHOW P. C.: Variations in cortical material properties throughout the human mdentate mandible. *A. Journ. Phys. Anthropol. 1* (2003), 120–252. 7

[SIA*09] SOHMURA T., IIDA S., AIKAWA T., KOGO M., IGUCHI Y., YAMAMOTO T., TAKADA K.: Simulation of osteotomy and support for surgery using VR haptic device. *Studies in health technology and informatics 142* (Jan. 2009), 331–6. 2

[SKD*13] SOFRONIA R., KNOTT T., DAVIDESCU A., SAVII G., KUHLEN T., GERRESSEN M.: Failure mode and effects analysis in designing a virtual realityâĂŘbased training simulator for bilateral sagittal split osteotomy. *Int. Journal of Med. Robot. and Comp. Assis. Surg. 9*, December 2012 (2013), e1–e9. 2, 6

[SSS*14] SCHVARTZMAN S. C., SILVA R., SALISBURY K., GAUDILLIERE D., GIROD S.: Computer-aided trauma simulation system with haptic feedback is easy and fast for oral-maxillofacial surgeons to learn and use. *J. oral and maxillofa. surg.* (2014). 2

[THM*03] TESCHNER M., HEIDELBERGER B., MÜLLER M., POMERANETS D., GROSS M.: Optimized spatial hashing for collision detection of deformable objects. *Proc. Vision Modeling Visual.* (2003), 47–54. 5, 6

[UMK12] ULLRICH S., MEMBER S., KUHLEN T.: Haptic Palpation for Medical Simulation in Virtual Environments. *IEEE TVCG 18*, 4 (2012). 2

[WCL*14] WU F., CHEN X., LIN Y., WANG C., WANG X., SHEN G., QIN J., HENG P.-A.: A virtual training system for maxillofacial surgery using advanced haptic feedback and immersive workbench. *Int. J. of Med. Robot. Comput. Assist. Surg. 10*, 1 (2014), 78–87. 2

[WM03] WAN M., MCNEELY W.: Quasi-static approximation for 6 degrees-of-freedom haptic rendering. In *IEEE Visualization 2003* (Oct. 2003), IEEE Computer Society, p. 34. 2

[YHGT10] YANG J. C., HENSLEY J., GRÜN H., THIBIEROZ N.: Real-Time Concurrent Linked List Construction on the GPU. *Computer Graphics Forum 29*, 4 (Aug. 2010), 1297–1304. 6

[ZS95] ZILLES C. B., SALISBURY J. K.: A constraint-based god-object method for haptic display. In *Proceedings ofthe International Conference on Intelligent Robots and Systems* (1995), MIT. 2