Computers & Graphics 57 (2016) 68–80

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Special Section on VRIPHYS 2015

Accurate and adaptive contact modeling for multi-rate multi-point haptic rendering of static and deformable environments

Thomas C. Knott [a,*], Torsten W. Kuhlen [a,b]

[a] Visual Computing Institute, RWTH Aachen University, Germany
[b] JARA – High-Performance Computing, Germany

ARTICLE INFO

Article history:
Received 18 December 2015
Received in revised form
17 March 2016
Accepted 21 March 2016
Available online 29 March 2016

Keywords:
Haptic rendering
Contact modeling
Physics-based simulation
Bimanual interaction

ABSTRACT

Common approaches for the haptic rendering of complex scenarios employ multi-rate simulation schemes. Here, the collision queries or the simulation of a complex deformable object are often performed asynchronously at a lower frequency, while some kind of intermediate contact representation is used to simulate interactions at the haptic rate. However, this can produce artifacts in the haptic rendering when the contact situation quickly changes and the intermediate representation is not able to reflect the changes due to the lower update rate.

We address this problem utilizing a novel contact model. It facilitates the creation of contact representations that are accurate for a large range of motions and multiple simulation time-steps. We handle problematic geometrically convex contact regions using a local convex decomposition and special constraints for convex areas. We combine our accurate contact model with an implicit temporal integration scheme to create an intermediate mechanical contact representation, which reflects the dynamic behavior of the simulated objects. To maintain a haptic real time simulation, the size of the region modeled by the contact representation is automatically adapted to the complexity of the geometry in contact. Moreover, we propose a new iterative solving scheme for the involved constrained dynamics problems. We increase the robustness of our method using techniques from trust region-based optimization. Our approach can be combined with standard methods for the modeling of deformable objects or constraint-based approaches for the modeling of, for instance, friction or joints. We demonstrate its benefits with respect to the simulation accuracy and the quality of the rendered haptic forces in several scenarios with one or more haptic proxies.

1. Introduction

Haptic force cues are an important feedback channel in humans' interaction with their environment. Haptic rendering algorithms are employed to synthesize these forces computationally. In many applications a physically realistic interaction is required. Here, one goal of the algorithms is to simulate the correct dynamic behavior of the involved objects. Such algorithms typically consist of two main steps: first, the contact situation between the objects is determined and second, a dynamics problem is setup and solved to compute the movements of the objects. The second step is usually done using a fixed set of contacts resulting from the first step. In the so-called "constraint-based" simulation approaches, the contacts are modeled using mathematical constraint equations, which are included into the dynamics problem [1]. These constraints restrict the solutions of the dynamics problem to the so-called *configuration space*. This defines all possible states the objects can take without violating the constraints.

For the simulation of an unilateral contact, a constraint is commonly defined via an inequality. It defines that the distance between two objects, or rather two geometric features, has to be larger or equal to zero. In real-time simulations, these constraints are typically approximated by *linear* inequalities before they are incorporated into the dynamics problem [1]. Such a linear inequality reduces the configuration space by an infinite half-space. Nguyen et al. show in [2] that this can cause problems as the half-space does not approximate the geometric features that the constraint represents very well.

This is especially problematic in geometrically convex contact situations, which generate non-convex problems from a optimization point of view that are usually harder to solve. A simple but still problematic scenario is shown in Fig. 1. Here, the movement of

* Corresponding author.
E-mail address: knott@vr.rwth-aachen.de (T.C. Knott).
URL: http://www.vr.rwth-aachen.de (T.C. Knott).

http://dx.doi.org/10.1016/j.cag.2016.03.007
0097-8493/

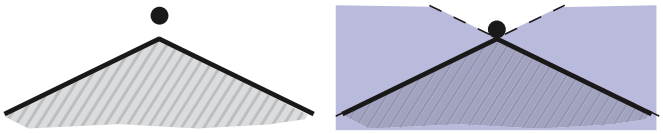

Fig. 1. (Left) Geometrically convex contact situation. (Right) Reduction of the configuration space by two half-space-based constraints. Unavailable space is shown in blue. If the dot just falls down in this example, the constraint set does not prevent the dynamically correct movement. In case it has additionally a slight momentum to the left, the constraint set would lead to an abrupt stopping of the dot outside of the geometry. The correct dynamic behavior could then be simulated using a constraint set that *only* contains a constraint for the left face. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

the dot has to be simulated for the next time-step. To this end, first the constraint set applied to the dynamics problem during the step needs to be determined. However, it is not clear how to model the actual geometry using half-space-like constraints. In the example, one constraint is used for each surface side. Due to their infinite extend, they reach into the actual free space and reduce the configuration space more than necessary. As the constraint set is fixed for the time-step, the point is restricted to stay in the non-blue area until the next step. Although it is possible to choose a different constraint set, it would always forbid movements which are valid from a *kinematic* point of view. In [2] it is demonstrated that this is in concave contact situations always the case. This over-reduction of the configuration space is a problem if it inhibits to simulate the correct *dynamic* behavior of the objects as explained in the example. In general, a badly chosen constraint set can impair the simulation accuracy significantly [1]. Therefore, the choice of the constraint set for the time-step is a critical decision.

Indeed, there are several different approaches to specify the constraint set; each with its own advantages and drawbacks. Anitescu et al., for instance, employ only constraints for geometric features, which are already in a touching or penetrating state [3]. However, Nguyen et al. show that this results in inter-penetrations, which get worse with an increasing time-step length and higher object velocities [2]. Other approaches use a temporal backtracking. Here, the constraint set is iteratively refined until a set is found, which allows for the correct simulation of the dynamic behavior of the objects [1]. This includes the solving of multiple intermediate dynamics problems. As such a process is computationally expensive, it is usually not employed for haptic rendering. A further option is to choose the contact set using an anticipation of the objects' movements. Such an anticipation is typically based on the unconstrained movement of the objects [4]. Consequently, it does not reflect any collision response, which can be problematic when constraints influence each other. There are also approaches trying to prevent the creation of constraint sets with incompatible constraints using the so-called Local Minimal Distance between two meshes [5]. These can help in finding a constraint set that maximizes the available configuration space. Nevertheless, using half-space-like constraints it will in geometrically convex setups always be over-reduced.

The choice of the constraint set gets even more difficult in case of multi-rate haptic rendering methods where the contact configuration is determined at a frequency lower than the one of the haptic simulation. This is a quite common approach, in case collision queries or soft tissue simulations are computationally too expensive to be done at a haptic simulation rate [6,4,7]. Here, a constraint set is determined at a low rate and then used in multiple time-steps at the haptic rate to simulate the user's interaction. Therefore, the constraint set must be appropriate for a *sequence* of simulation steps. As the user interacts with the simulated objects during this sequence, the anticipation of their movements to select a good constraint set is rather difficult. Furthermore, there are situations for which no appropriate constraint set exists. This is the case when the user moves an object over a convex corner during such a sequence. Consider our example and a user that controls the dot and moves it from left to right over the surface; at the beginning, *only* the left constraint should be in the contact set, while at the end, *only* the right one should. In consequence, there is no appropriate *single* constraint set for the complete sequence and, thus, the heuristics for the selection of a constraint set, mentioned in the last paragraph, cannot create one. The resulting rendering artifacts of this problem can be severe, as we show in our experiments (see Section 7).

In conclusion, the problems originate from the bad approximation of the actual object geometry by the linearized constraint set. Furthermore, the implications get more severe in a haptic multi-rate simulation when the set is in use for more then one time-step.

To resolve these problems, we proposed a new efficient contact model in [8], which permits to accurately model the actual object geometry during the solving process of the dynamics problem. The method builds upon an approach presented in [9], which employs the so-called geometrically limited constraints (GLC). Nevertheless, the method using the GLCs has some problems as described in [9] and are further analyzed in this paper. One major problem is that the GLCs only approximate the geometry, which becomes especially problematic in geometrically convex regions. The improved approach proposed in [8] resolves this and other problems of the contact model by incorporating a special treatment of these regions. Furthermore, a method is presented that increases the robustness of the approach by applying techniques from trust region-based optimization.

On key feature of the proposed contact model is that it can be used for a multi-rate haptic rendering for point-based interaction with static and deformable scenes. To this end, a mechanical intermediate contact representation is proposed that is based on the contact model and also incorporates the dynamic behavior of the involved objects, i.e., their compliance. Therefore, a high simulation quality is also guaranteed if the update of the representation is much lower than the haptic frame rate. The evaluation shows that the proposed methods permit an artifact-free haptic rendering of static and deformable scenarios, and demonstrates their haptic real-time capabilities.

One further advantage of the approach is that it does not require a specific technique to model deformable or rigid objects and is compatible with other constraints used, for instance, to model friction or joints.

Nevertheless, the method proposed in [8] is limited to the haptic rendering using only a single point proxy. In this paper, we extend the approach to multiple interaction proxies and permit, for instance, a bimanual haptic interaction. Another shortcoming of the approach is that it uses a fixed size for the region in which geometries were modeled by the intermediate contact representation. This impairs the haptic rendering in case the scene contains geometries with varying mesh resolutions as shown in the results section. Therefore, we propose a method to adaptively change the size of the modeled region depending on the complexity of the geometry currently in the vicinity of a proxy. We show the improved behavior of this extension w.r.t. the haptic rendering in an corresponding experiment.

The remaining paper is organized as follows: after discussing the related work, we describe in Section 3 the fundamental methods for an accurate contact modeling for multi-rate haptic rendering. Thereafter, we show how they can be combined with an efficient simulation of deformable objects in Section 4. Then, we propose how the described approach can be extended to multiple haptic proxies in Section 5 and the usage of a complexity-aware

adaptive contact region creation in Section 6. Finally, we evaluate and discuss our methods in Section 7.

2. Related work

In this section, we provide an overview on the investigated approaches in the area of haptic rendering relevant for our work. Most of the methods rely on a virtual coupling approach where a proxy object is coupled to the input of the haptic device to increase the rendering stability [10]. The type of the proxy object is chosen based on the application. Some use points, as for instance in [11] or lately [12,9] and our approach. Others use rigid bodies [13,6,14] or deformable objects [4,7]. The rendering techniques can be furthermore distinguished by the way contacts are handled. Some apply penalty forces, for instance in [13,14], which have the disadvantage that they result in object inter-penetrations during the simulation. In order prevent these, others use mathematical bilateral [11] or unilateral [4,15] constraints. The latter type is also employed to model frictional contact based on Coulombs law [4,15]. The equations describing the unilateral constraints are commonly linearized in real-time applications. Like detailed in the introduction, this creates problems in geometrically convex contact situations because the configuration space does not properly reflect the geometry. Nguyen et al. tackled this problem for convex geometries in the context of rigid-body simulations [2]. They model each convex object through a set of linear inequalities which are combined using an intersection operation.

The computation of a complete simulation time-step can be too time consuming to be performed at a haptic frame rate. Therefore, the so-called multi-rate approaches perform computationally expensive parts asynchronously at a lower frequency [16,17,7]. For complex scenes, one potentially expensive part is the collision detection. Therefore, Otaduy et al. investigated an approach for the haptic rendering of a static environment in which the collision detection was performed at a lower rate [6]. To increase the rendering quality, a linear update of the penalty-based collision response was proposed. When it comes to scenes including complex deformable objects, the setup and solving of the dynamics problem gets an additional computational burden. Some methods circumvent the problem by reducing the complexity of the objects to be able to simulate them at a high frequency [14,18]. Wang et al., for instance, simulated deformable objects based on sphere-trees augmented with spring-dampers [19]. Although, such methods enable the simulation at haptic rates, they come at the cost of physical accuracy. As this is not suitable in many applications, other methods facilitate complex deformable objects by performing their simulation at a lower rate, while the simulation of the objects controlled by the haptic device is performed at a high haptic rate. Peterlik et al. proposed in [17] a method where these objects are able to interact with each other via an intermediate contact representation. This representation is computed at a lower rate and shared with the high rate. Although the used intermediate representation reflects the compliance of the objects, it allowed only for a quasi-static simulation. The approach was extended in [7] to handle the full dynamics of the involved objects. The used intermediate contact representations are based on linear unilateral constraints. As described in the introduction, the linearization does not permit to accurately model the geometry in geometrically convex contact situations, especially in multi-rate simulation schemes. This creates artifacts in the haptic rendering as we show in Section 7.

The main objective of the approach described in [8] and extended in this paper is to provide a contact model that resolves this problem. The model facilitates a description of a contact configuration, which stays accurate for a larger range of motions than a description based on linear inequalities. The method also fits well with standard mechanical descriptions of deformable objects and constraint-based methods to model, for instance, friction or joints. However, the method described in [8] is limited to a single-point of interaction. In this paper, we extend it to permit multiple haptic proxies and to enable its application to bimanual haptic rendering scenarios. We furthermore propose a method to dynamically adapt the size of the region in which the contact configuration is described proportionally to the complexity of the part of the geometry currently in contact.

3. Accurate contact modeling for multi-rate haptic rendering

In this section, we describe the accurate contact model. First, we introduce the fundamental dynamics equations and notations. Then, we lay out a multi-rate simulation approach (Section 3.2), which gives the needed context for the new contact model detailed afterwards (Section 3.3). The model builds upon the method of geometrically limited constraints [9]. We, then, analyze its problems in our context and resolve them using a local convex decomposition of the contact area and special constraints for geometrically convex areas. Afterwards, we present an iterative solving scheme for the created constrained dynamics problems based on a Projected Gauss–Seidel scheme (Section 3.4). We increase the robustness of this process utilizing techniques from trust region-based optimization (Section 3.5).

3.1. Dynamics fundamentals and notations

This section introduces the notations and equations of dynamics used throughout the paper. For more details and corresponding proofs, we refer to fundamental literature like [1]. The mechanical behavior of the simulated objects is described by the usual equation of dynamics:

$$\mathbf{M}\mathbf{a} = \mathbf{f}(\mathbf{x}, \mathbf{v}), \tag{1}$$

where $\mathbf{a}$ specifies the acceleration, $\mathbf{x}$ and $\mathbf{v}$ are respectively positions and velocities, and $\mathbf{M}$ is a mass matrix. The external and internal forces originating from deformation, gravity etc. are defined by $\mathbf{f}$. Here and throughout the paper we use bold letters for matrices and vectors to differentiate them from scalar values.

Eq. (1) includes terms for one or optionally more haptic proxies (see Section 5). Such a proxy is simulated as mass point with position $\mathbf{x}_{\mathrm{HP}}$ and velocity $\mathbf{v}_{\mathrm{HP}}$ that is connected by a spring-damper to an input device:

$$\mathbf{F}_{\mathrm{VC}}(\mathbf{x}_{\mathrm{HP}}, \mathbf{v}_{\mathrm{HP}}) = -k(\mathbf{x}_{\mathrm{HP}} - \mathbf{x}_{\mathrm{IP}}) - d(\mathbf{v}_{\mathrm{HP}} - \mathbf{v}_{\mathrm{IP}}), \tag{2}$$

where $\mathbf{x}_{\mathrm{IP}}$ and $\mathbf{v}_{\mathrm{IP}}$ are position respectively velocity of the device, while k and d are stiffness respectively damping parameters. In case deformable objects are in the environment, these are also included in (1) using e.g. the finite element method as described in Section 4. All degrees of freedom and corresponding forces are stacked in $\mathbf{x}$ and $\mathbf{v}$ respectively $\mathbf{f}$.

Contact forces are integrated using Signorini's contact model [16] and defined via complementarity conditions:

$$0 \leq \lambda^i \perp \psi^i(\mathbf{x}, \mathbf{v}) \geq 0, \tag{3}$$

where λ^i is the contact force magnitude at the ith contact and ψ^i is a function, which describes the signed distance between the bodies with respect to this contact.

For a stable temporal integration, a backward Euler scheme using linear approximations of the forces is employed. The dynamics problem is formulated with a fixed time-step h on a

velocity level as:

$$\underbrace{\left(\frac{1}{h}\mathbf{M}-\mathbf{B}-h\mathbf{K}\right)}_{\mathbf{D}} d\mathbf{v} = \underbrace{\mathbf{f}(\mathbf{x}_t,\mathbf{v}_t)+h\mathbf{K}\mathbf{v}_t}_{\mathbf{b}}+\mathbf{J}^T\boldsymbol{\lambda}, \tag{4}$$

where $\mathbf{x}_t$ and $\mathbf{v}_t$ are the generalized coordinates and velocities of the whole simulated system at the beginning of the time-step t. $\mathbf{B}=\partial\mathbf{f}/\partial\mathbf{v}$ and $\mathbf{K}=\partial\mathbf{f}/\partial\mathbf{x}$ are the corresponding partial derivatives of the internal forces and $\mathbf{J}=\partial\boldsymbol{\psi}/\partial\mathbf{v}$ is used to apply the constraint forces to the objects. Finally, $d\mathbf{v}$ denotes the change in the velocity that is to be computed. This equation is augmented by complementarity conditions for the unilateral contacts:

$$\mathbf{0}\le\boldsymbol{\lambda}\ \perp\ \mathbf{J}(\mathbf{v}_t+d\mathbf{v})+\frac{\boldsymbol{\psi}_t}{h}\ge\mathbf{0}, \tag{5}$$

where $\boldsymbol{\lambda}$ is a vector containing the stacked contact forces λ^i of all constraints and $\boldsymbol{\psi}_t$ is a vector containing the stacked values of the corresponding constraint functions ψ^i at the beginning of the time-step. We use here and in the following the inequality and complementarity relation for vectors, this common shorthand denotes that the inequality holds for each tuple of the components [1]. The conditions in (5) enforce a penetration free state at the end of the time-step with respect to the incorporated constraints. Other constraint types, e.g. modeling friction, are possible and are considered later on. The last two equations form together a Mixed Linear Complementarity Problem (MLCP). We refer to this as the standard way to model contacts (SCM), which also builds the basis for the contact model proposed in this paper and serves as comparison method in our experiments.

In single-rate simulation schemes, each time-step would include the following two phases: (i) the constraint set is determined based on a proximity query, and (ii), the corresponding MLCP is setup and solved. This conflicts in some simulation scenarios with the real-time requirements in haptic rendering. One reason may be that the proximity queries are computationally too expensive, another, that the mechanics of the simulated objects are too complex, for instance, when deformable objects are simulated. Therefore, a common approach is to perform a multi-rate simulation, in which computational demanding parts are done at a lower rate.

3.2. Multi-rate simulation

In this section, we describe a basic multi-rate simulation scheme, which provides the necessary context for our new contact model. This is described in the next section and integrated into this scheme. Like in [7], our multi-rate scheme performs proximity queries at a low rate to setup a constraint set, which is used for multiple simulation steps at a high rate. At the low rate (*LR*), we linearize the dynamic system and setup the dynamics problem (4) including the constraints (5). At the high haptic rate (*HR*), the resulting dynamics problem is partially updated and solved to compute the new object states. Thus, the temporal integration is only performed at *HR*. For reasons of performance, the problem is reformulated using the Schur complement as a pure Linear Complementarity Problem (LCP). This can then be efficiently solved using iterative methods as the Projected Gauss–Seidel Method [1]:

$$0\le\boldsymbol{\lambda}\ \perp\ \underbrace{\mathbf{J}\mathbf{D}^{-1}\mathbf{J}^T}_{\mathbf{A}_\psi}\boldsymbol{\lambda}+\underbrace{\boldsymbol{\psi}_t/h}_{\mathbf{b}_\psi}+\underbrace{\mathbf{J}(\mathbf{v}_t+h\mathbf{D}^{-1}\mathbf{f}_t)}_{\Delta\mathbf{b}_\psi}\ge\mathbf{0}, \tag{6}$$

where $\mathbf{b}_\psi$ reflects the constraint values at the beginning of a time-step and $\Delta\mathbf{b}_\psi$ how the values would change during this step in case of an unconstrained motion. Furthermore, $\mathbf{A}_\psi$ denotes the constraint response matrix (CRM). It approximates how the values of the constraint functions $\boldsymbol{\psi}$ change for a set of constraint forces $\boldsymbol{\lambda}$ and can be seen as a mechanical intermediate representation for the haptic rendering. We emphasize that the CRM reflects the complete dynamics of the involved objects under consideration of the temporal integration scheme. In case of deformable objects, this also includes their compliant behavior. All in all, the multi-rate simulation algorithm then encompasses the following steps:

Low rate:
L1. Setup dynamics problem $\mathbf{D}\mathbf{x}=\mathbf{b}$
L2. Perform proximity query
L3. Define constraint set and setup Jacobi $\mathbf{J}$
L4. Compute constraint response matrix $\mathbf{A}_\psi$
L5. Share data with high rate

High rate:
H1. Receive *LR* data and haptic device state
H2. Compute $\mathbf{b}_\psi$, $\Delta\mathbf{b}_\psi$
H3. Solve LCP($\mathbf{A}_\psi$, $\mathbf{b}_\psi$, $\Delta\mathbf{b}_\psi$)$\rightarrow\boldsymbol{\lambda}$
H4. Compute new object states using (4)
H5. Send virtual coupling forces to haptic device
H6. Share object states with low rate

Notice that this approach already allows for a simulation with deformable objects. Nevertheless, it would be slow due to the full motion integration in step H4. We address this topic further in Section 4.

In the simulation scheme described above, the contact constraints are setup in *LR*. Therefore, they are used in multiple *HR*-time-steps and for a range of different user inputs. Like detailed in the introduction, problems arise in geometrically convex contact situations from the linearization of the constraints in such a multi-rate application. Here, the constraint set models the involved actual shape of the geometry unsatisfactorily. In the following, we present a novel contact model which resolves the problem.

3.3. Accurate contact modeling

The main objective of the proposed contact model is to facilitate a description of a contact configuration, which stays accurate for a larger range of motions. Thereby, it accurately models the contacts over a longer period of time, i.e., in a sequence of multiple *HR*-time-steps. To this end, we integrate more information about the underlying actual geometry into the contact description, so that it reflects all parts of the geometry, which may be involved in the contact during the sequence. Furthermore, we introduce a special treatment of the problematic geometrically convex areas (see Section 1) to handle these properly.

The proposed approach builds upon the method of geometrically limited constraints (GLC) described by [9]. We furthermore modify it, to resolve the shortcomings of the method in context of our rendering framework. The basic idea behind the GLCs is that as a contact constraint corresponds to a specific geometric feature with a finite size, the corresponding constraints should also be restricted to a corresponding region.

To model such a GLC, a coordinate system is defined around each contact i using the contact normal $\mathbf{n}^i$ and two additional orthogonal tangential vectors $\mathbf{t}_1^i$ and $\mathbf{t}_2^i$. The gap function $\psi^i(\mathbf{x},\mathbf{v},t)$, introduced in Section 3.1, maps from the motions space to the normal direction $\mathbf{n}^i$ of this coordinate system. Analogue to this, we define two additional functions $\tau_1^i(\mathbf{x},\mathbf{v},t)$ and $\tau_2^i(\mathbf{x},\mathbf{v},t)$ per contact, which map from the motion space to the tangential directions $\mathbf{t}_1^i$ and $\mathbf{t}_2^i$. These are then utilized to define constraints that are limited to specific regions. Hence, for a rectangular region the constraint i should only be active if:

$$-s_1^i\le\tau_1^i(\mathbf{x},\mathbf{v},t)\le s_1^i \quad\text{and}\quad -s_2^i\le\tau_2^i(\mathbf{x},\mathbf{v},t)\le s_2^i \tag{7}$$

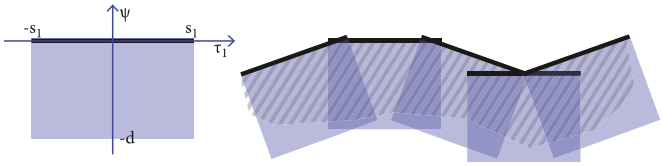

Fig. 2. (Left) GLC with coordinate system (Right) GLC contact set with geometrically concave and convex areas.

holds, where s_1^i and s_2^i specify the size of the active region of GLC i with respect to the two tangential directions (see Fig. 2 left).

GLCs are created for all entities delivered by the proximity queries in step L2 and form the constraint set C_{GLC}. The entities can be triangles, point shell points, or edges. The latter can be used to create GLCs for sharp concave contact areas (see Fig. 2 right); we come back to this topic in Section 7. In contrast to [9], we use the same C_{GLC} in multiple iterations of *HR*. Therefore, it has to be valid over a longer period and farther movements as in a single-rate simulation. Consequently, we also need to perform the proximity queries with a higher proximity threshold (see Section 7).

This higher proximity threshold can create problems with highly convex or thin geometries. If the user interacts, for instance, with the arm of the statue in the top left of Fig. 8, C_{GLC} contains GLCs for the bottom side too. However, in the original method proposed by [9] a GLC is not limited in depth. Consequently, GLCs from opposing parts of the geometry possibly overlap and create conflicts. To prevent this, we additionally limit the active region of a GLC (7) in the direction of the contact normal by depth d^i using ψ^i (see Fig. 2 left):

$$-d^i \leq \psi^i(\mathbf{x}, \mathbf{v}, t) \leq 0. \tag{8}$$

Nevertheless, further issues emerge from overlaps between the active regions of the GLCs in geometrically convex contact situations. A first problem was already described by the authors of [9]. It arises when the borders of the constraints are not perfectly aligned at the surface (see Fig. 3 left). Here, the overlaps create a small concave surface which results in a locking effect as described in the introduction. The impact of the effect on the simulation quality increases with the size of the overlap. Therefore, the simulation quality strongly depends on a good matching sampling of the geometry surface by the GLCs. This can be cumbersome in case of complex geometries, as it requires a high number of small constraints, which reduces the efficiency of the approach. Another problem occurs if constraints are combined with an angle that is too sharp (see Fig. 3 right). Here, the active volumes of the constraints reach into the actually free space. As a result, the configuration space is reduced more than necessary. Nevertheless, even if the constraints do not overlap on the surface, there is an overlap in the inside (see Fig. 3 middle). Therefore, it becomes ambiguous which constraint should be active in case the constraints are already violated. This is possible due to the overall approximative nature of simulations based on linearized dynamics. Furthermore, in case an iterative solving approach is employed, the violation can occur during the solving process as described in Section 3.4.

To resolve the demonstrated problems, we use a method based on a convex decomposition of the active GLCs. To this end, we build upon the approach described in [2]. Here, a convex object involved in a contact is defined via a set of half-spaces $k = 1, .., m$ modeled via linear inequalities $\psi^k(\mathbf{x}, \mathbf{v}, t) \geq 0$. For a complete convex object there is no penetration if at least one of these inequalities is non-violated. Therefore, one can use the following condition to define a non-penetration constraint for the object:

$$\exists k \in 1..m : \psi^k(\mathbf{x}, \mathbf{v}, t) \geq 0 \Leftrightarrow \max_{k=1,..,m} \{\psi^k(\mathbf{x}, \mathbf{v}, t)\} \geq 0. \tag{9}$$

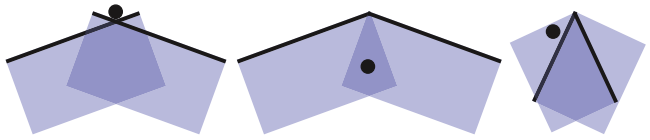

Fig. 3. Problems due to GLC overlaps: (left) locking effect, (middle) constraint ambiguity, (right) too sharp corner.

The second formulation using max was proposed by Nguyen et al. and we use it in the following due to its conciseness.

Nevertheless, to be able to apply this method, a convex decomposition of the involved geometries is needed. The approach described by [2] is based on a complete convex decomposition of the objects. This is not only computationally quite expensive for complex scenarios, but can also degenerate for geometries with a bowl-like shape [20], which tend to occur when pressing on soft materials. As we target a simulation with deformable objects, the decomposition is furthermore necessary in each time-step.

However, we need the convexity information only for the currently active GLCs and, therefore, also perform the decomposition only on this set. To this end, we first determine the subset $C_{GLC}^a \subseteq C_{GLC}$ of the GLCs that are currently in their active region. This is done based on (7) and (8) as described in more detail later on. Second, we perform a local convex decomposition of C_{GLC}^a (see Fig. 8 top right), which results in the disjoint subsets $C_{cs}^l, l = 1, .., m$ of C_{GLC}^a each representing a geometrically convex surface area close to the proxy. As the number of active constraints is usually rather low, we can employ a naive brute force approach without affecting the overall performance significantly. In doing so, we incrementally build up the geometrically convex sub sets $C_{cs}^l, l = 1, .., m$. To this end, we start with a single set C_{cs}^l containing an arbitrary GLC $c \in C_{GLC}^a$. Then we process the other constraints in C_{GLC}^a. For each, we check if it could be added to an existing set without destroying its geometrical convexity as described by [20] using the center points $\mathbf{p}^i$ and normals $\mathbf{n}^i$ of the constraints; if we find a set we add the constraint; if not we create a new set for the constraint. In our experiments (see Section 7), which include also deformable scenarios, this process was not an bottleneck. However, in case it should become a performance issue, one could employ more elaborate methods [1].

Finally, we need to integrate this contact model into our multi-rate simulation scheme. As described, C_{GLC} is setup in *L3* and stays the same in multiple iterations of *HR*. However, the active constraint set C_{GLC}^a changes during these iterations depending on the state of the simulated objects. To test which GLC are currently active, their constraint values $\psi^j, \tau_1^j, \tau_2^j$ have to be checked in *HR* regarding their bounds (7) and (8). As the constraints potentially influence each other and any constraint can be active and apply a force, the influence of each constraint on any other needs to be computable. Similar as in (6), this can be done for $\boldsymbol{\psi}$ on a velocity level by:

$$-\mathbf{d}/h \leq \mathbf{A}_\psi \boldsymbol{\lambda} + \mathbf{b}_\psi + \Delta \mathbf{b}_\psi \leq \mathbf{0}, \tag{10}$$

where we again employ the vector notation for the inequalities to reflect all GLCs. The constraints for which the condition holds are in their active region with respect to their normal direction $\mathbf{n}^i$. Here, inequality (10) incorporates the changes in $\boldsymbol{\psi}$ through active constraint forces $\boldsymbol{\lambda}$ via the constraint response matrix $\mathbf{A}_\psi$. This is defined as in (6).

Using (10) we can check if a constraint is in its active region with respect to the normal direction only. To test for the bounds in the tangential directions we proceed similarly. Therefore, we first of all create linear approximations for the tangential functions τ_1^i and τ_2^i analogue as for $\boldsymbol{\psi}$ in (5):

$$\boldsymbol{\tau} \approx \boldsymbol{\tau}_t + h\mathbf{H}(\mathbf{v}_t + d\mathbf{v}), \tag{11}$$

where $\boldsymbol{\tau}$ denotes the vector of the resulting values of the functions τ_1^i and τ_2^i in a stacked form, $\boldsymbol{\tau}_t$ are the corresponding stacked values of the functions at time t, and $\mathbf{H}_\tau = \partial\boldsymbol{\tau}/\partial\mathbf{v}$. Furthermore, we use analog to (10):

$$-\mathbf{s}/h \leq \underbrace{\mathbf{H}\mathbf{D}^{-1}\mathbf{J}^T}_{\mathbf{A}_\tau}\boldsymbol{\lambda} + \underbrace{\boldsymbol{\tau}_t/h}_{\mathbf{b}_\tau} + \underbrace{\mathbf{H}(\mathbf{v}_t + h\mathbf{D}^{-1}\mathbf{f}_t)}_{\Delta\mathbf{b}_\tau} \leq \mathbf{s}/h \tag{12}$$

to test $\boldsymbol{\tau}$ in *HR* against the bounds $\mathbf{s}$. $\mathbf{A}_\tau$ approximates how the tangential functions $\boldsymbol{\tau}$ of all constraints change for a set of constraint forces $\boldsymbol{\lambda}$. In this, $\mathbf{A}_\tau$ reflects the complete dynamics of the involved objects under consideration of the temporal integration scheme. In our multi-rate simulation, $\mathbf{A}_\tau$ is additionally computed in step L4 and can be seen as part of the intermediate contact representation. $\mathbf{b}_\tau$ and $\Delta\mathbf{b}_\tau$ are calculated in step H2.

The conditions (10) and (12) define together the active set of constraints C^a_{GLC}. A constraint is only in the set if its normal and tangential conditions are fulfilled.

Moreover, we need to integrate the constraints for the convex areas (9) into our dynamics problem. To this end, we replace the constraint equations from (5) by one complementarity condition of the form:

$$0 \leq \lambda \perp \max_{k=1,..,m} \{\underbrace{\mathbf{J}^k(\mathbf{v}_t + d\mathbf{v}) + \psi_t^k/h}_{\psi^k}\} \geq 0, \tag{13}$$

for each geometrically convex area.

Here, the constraint force λ is applied to the dynamics problem (4) using the Jacobi matrix $\mathbf{J}^k = \partial\psi^k/\partial\mathbf{v}$ of the constraint ψ^k with the smallest violation. We give further details on this in the next section, where we present an efficient solving scheme adequate for the resulting constrained dynamics problems, as they do not comply to standard solvers anymore.

In summary, the intermediate contact representation computed at *LR* and shared with *HR* consists of the following parts: the constraint set C_{GLC}, the Jacobi matrices $\mathbf{J}$ and $\mathbf{H}$, and the constraint response matrices $\mathbf{A}_\psi$ and $\mathbf{A}_\tau$. Additionally, for each GLC i in C_{GLC} the position $\mathbf{p}^i$, the coordinate system $\mathbf{n}^i$, $\mathbf{t}_1^i$, $\mathbf{t}_2^i$, and the GLC extends d^i,s_1^i,s_2^i are shared as well. All the data is send at the *LR* in step L5 and received at the *HR* in step H1.

3.4. Iterative solver

In this section, we describe a solver for constrained dynamics problems, which incorporate the previously described contact model. The proposed solver uses an iterative approach and follows a Projected Gauss–Seidel (PGS) scheme. The latter is often applied to solve LCPs in real-time applications. Implementation details on the PGS method itself can be found in [1]. The solver computes penetration resolving constraint forces $\boldsymbol{\lambda}$ with respect to a given constraint set C. These can comprise a set of GLCs, C_{GLC}, but also additional constraints, C_{other}, modeling friction or joints. As additional input, the solver requires the constraint response matrices $\mathbf{A}_\psi$ and $\mathbf{A}_\tau$ plus the corresponding vectors $\mathbf{b}_\psi$, $\Delta\mathbf{b}_\psi$, $\mathbf{b}_\tau$ and $\Delta\mathbf{b}_\tau$.

At the beginning of each iteration, the active GLC subset $C^a_{\mathrm{GLC}} \subseteq C_{\mathrm{GLC}}$ is computed. To this end, we determine for each GLC whether it is in its active range. This is done under consideration of the previous results of the constraint forces $\boldsymbol{\lambda}$ using (10) and (12). The set C^a_{GLC} is then decomposed into its geometrically convex subsets $C^l_{cs}, l = 1,..,m$ as described in the last section. Afterwards, we resolve the constraint violations employing the PGS strategy; hence, we iterate over all sets and resolve them consecutively. To this end, we first compute, for a subset C^l_{cs}, the GLC with the minimal violation. The violation of the GLC is then resolved using the standard PGS method. If there was no violation, the complete subset was non-violated and we proceed to the next subset.

After the GLC subsets are handled, we proceed to the other constraints C_{other}. These are resolved using the standard PGS method. The whole process is repeated until an error threshold is reached and the forces converge (see Section 7). The complete solving scheme is summarized in the following and substitutes the solver in step H3 of the multi-rate simulation.

Iterate until convergence and error threshold is reached
- Compute active GLC Set C^a_{GLC} based on λ
- Set $\lambda_c = 0$ for all $c \in C_{\mathrm{GLC}} \setminus C^a_{\mathrm{GLC}}$
- Compute convex sub sets $C^l_{cs} \subseteq C^a_{\mathrm{GLC}}, l = 1,..,m$
- Iterate over all $C^l_{cs}, l = 1,..,m$
  - Compute $c_k \in C^l_{cs}$ with minimal violation
  - If violated, compute resolving λ_{c_k} for c_k
  - Set $\lambda_c = 0$ for all $c \in C^l_{cs} \setminus \{c_k\}$
- Iterate over all $c \in C_{other}$
  - Compute resolving λ_c for c

During the solving process the constraints with non-zero forces $\boldsymbol{\lambda}$ may change completely several times. A first reason can be that the constraint c_k that defines the maximum of a convex set C^l_{cs} switches. A second, that the proxy leaves the active region of one constraint and enters the one of another. These changes could slow down the convergence or possibly even lead to oscillating behavior. Nevertheless, in our experiments this was never an issue as the solving process always converged in the available time (see Section 7). Furthermore, we propose in the following section a method to increase the robustness of the process.

3.5. Approximation quality awareness & trust regions

Using the approach described above, we can model large contact spaces including geometrically convex areas and incorporate them into our multi-rate simulation. Practically, the size of the modeled area is of course limited by the available computation time as we discuss in Section 6. Furthermore, when we solve the dynamics problem, we are not aware in which area such a contact space description is valid. Therefore, a solution to the problem might lead to movements that go out of the modeled area. In consequence, the movements might result in penetrations of the unmodeled parts of the geometry.

To tackle this problem, we add an additional requirement on the allowed solutions. Following the idea of trust region-based optimization [21], we include a constraint to our dynamics problem, which restricts the results to the region where the employed approximations are trusted. In our case, this region corresponds to the area that is accurately modeled by the current constraint set. To this end, we add in step L3 a box-constraint C_{TR} to our constraint set C. We use the largest possible box that fits into the area covered by the proximity queries. Thereby, it conservatively restricts the solution of the dynamics problem to the area covered by the GLCs. The constraint is defined via:

$$-\frac{r_n}{2} \leq \gamma_n(\mathbf{x}, \mathbf{v}, t) \leq \frac{r_n}{2}, \quad \text{with } n = 1,2,3 \tag{14}$$

where γ_i are functions that map to the box-constraint-coordinates and r_n define the extend of the box. To integrate this constraints into our simulation, we linearize γ_n and add corresponding constraint inequalities for C_{TR} into the dynamics equations (4) and (5). This is done similarly to the other constraints and we therefore spare the details. As result, the solution is not allowed to leave the region modeled by the contact set.

Furthermore, another problem arises during the iterative solving process from the limited range in which an *individual*

constraint should be active. To understand the problem, we take a closer look on how the proposed solver works and interpret the iterative process geometrically.

The algorithm starts with $\lambda_0 = 0$, which means there are zero constraint forces. Considering our simulation scheme, that would lead to an unconstrained motion of the objects. This would result in state:

$$\mathbf{x}_{\text{free}} = \mathbf{x}_{\text{old}} + \Delta\mathbf{x}_{\text{free}}, \tag{15}$$

where $\mathbf{x}_{\text{old}}$ is the result of the last time-step and $\Delta\mathbf{x}_{\text{free}}$ is the motion without constraints. Then, during the successive solving process a sequence of constraint forces λ_k are computed. These correspond to corrective movements:

$$\Delta\mathbf{x}(\lambda_k) = h\mathbf{D}^{-1}\mathbf{J}^T\lambda_k \tag{16}$$

of the objects and define a corresponding sequence of tentative states $\mathbf{x}_k = \mathbf{x}_{\text{free}} + \Delta\mathbf{x}(\lambda_k)$. In this process, the computation of a set of constraint forces λ_k at step k is based on the previous state $\mathbf{x}_{k-1}$; the latter is used to evaluate the constraints and test if an GLC is in its active range.

We now analyze two problematic contact scenarios. In the first one, there are two opposing surfaces and an unconstrained motion, which would tunnel from one side to the other (see Fig. 4 left). Therefore, our solving process starts with a state $\mathbf{x}_{\text{free}}$, which has already passed GLC1 and GLC2. Consequently, no constraint is active in the first iteration and the constraint forces λ become zero. Therefore, no corrective motion is applied and the result of the step is $\mathbf{x}_{\text{free}}$ again. The next iteration then starts under the same conditions an come to the same result. In consequence, the motion tunnels through the object. A first naive approach to resolve the problem would be to start the process with $\mathbf{x}_{\text{old}}$. This fails in the second problematic scenario (see Fig. 4 right). Here, starting from $\mathbf{x}_{\text{old}}$, only the GLC1 is active in the first iteration. This results in the tentative state $\mathbf{x}_{\text{GLC1}}$ which jumps over the GLC2. Hence, no constraint is active in the next iteration, which results in a tunneling through the geometry.

In summary, issues arise when the sequence of tentative states $\mathbf{x}_k$, computed during the solving process, comprises too large movements. More precisely, the problem occurs when the movement from one state to the next, $\Delta\mathbf{x}_k = \mathbf{x}_k - \mathbf{x}_{k-1}$, skips the active region of one of the constraints.

To resolve this problem, we again follow the idea of a trust region. Therefore, we add a *second* smaller trust region into our solving process that reflects the range of the *individual* constraints. It is used in each iteration to limit the change in the tentative state $\Delta\mathbf{x}_k$. In doing so, we want to ensure that we can recover in the next state from penetrations and do not miss out any GLC. The trust region is again implemented as a box-constraint and its size is chosen dynamically with the size of the smallest GLC in the current contact set C_{GLC}. For the first iteration of our solver, we set the center of the box to $\mathbf{x}_{\text{old}}$. After each iteration, the constraint is applied similarly to the other constraints using the PGS method. Differently from the GLCs, the box-constraint cannot be missed as it is defined by an infinite half-space for each side of the box. Furthermore, as we apply it as last step of each iteration k, the final

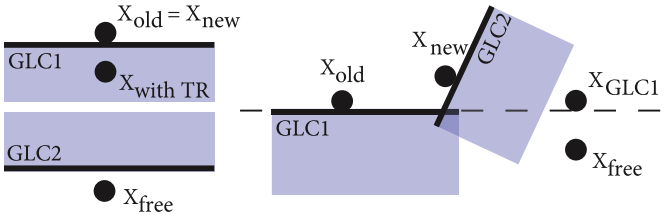

Fig. 4. (Left) Free motion overleaps both GLCs. (Right) GLC1 resolution results in overleap of GLC2.

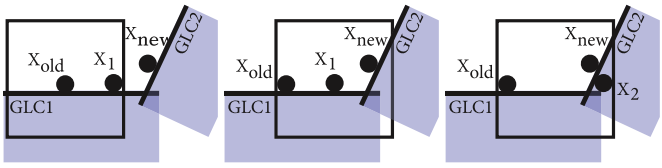

Fig. 5. Illustration of the solving process with a trust region (TR) for the scenario in Fig. 4 (right). The TR is shown as the black frame. (Left) The proxy position x_1 at the end of the first iteration after application of the TR constraint. (Middle) The TR at the beginning of the second iteration. The center is moved to the proxy position x_1. (Right) The proxy position x_2 at the end of the second iteration after the application of the TR. In the next iteration, GLC2 will get active and finally lead to the correct solution. Please note that, to prevent a cluttering of the figure, we omitted in this example the additional GLC, which would normally be used for a geometrically concave corner as described in Section 3.3.

tentative state $\mathbf{x}_k$ at the end of the iteration lies inside the box and thus the currently trusted region. In the next iteration of the solver, a new active set of GLCs C^a_{GLC} is determined based on $\mathbf{x}_k$. Therefore, we can trust, that the region around $\mathbf{x}_k$ is modeled correctly by C^a_{GLC}. This permits us to move the center of the box-constraint to $\mathbf{x}_k$ so that further movements are possible during the iteration. Fig. 5 illustrates the solving process employing the trust region for the example scenario shown in Fig. 4 (right).

As the trust region size is chosen to be smaller than the extent of the smallest GLC, the overall movement $\Delta\mathbf{x}_k$ created by an iteration cannot skip over any constraint. In consequence, the trust region resolves the problems described above. However, the approach restricts the movements that can be done in each iteration. Therefore, it can increase the number of iterations that have to be done to find the final result. Nevertheless, this was not an issue in our experiments as the process always converged to a solution within less than 1 ms (see Section 7.3).

4. Simulation with complex deformable objects

As announced, we want to facilitate, with the contact model, also an accurate haptic rendering of deformable environments. However, the multi-rate simulation described in Section 3.2 performs in step H3 a temporal integration of *all* objects in the scene, which can be too slow for complex deformable objects. To this end, we integrate our contact model into the asynchronous simulation approach described by Dervaux et al. [7].

This approach also performs proximity queries at a lower rate to create a mechanical intermediate contact representation, but instead of performing a temporal integration of all objects at the haptic rate, like we do, the simulation of complex deformable objects is done *solely* at the lower rate. The collision response between the objects simulated at the different rates is calculated using the intermediate contact representation. The representation is based on the classic contact model as described in the introduction and employs a linear complementarity problem similar to (6) to reflect the dynamic behavior of the objects. Therefore, Dervaux et al. also compute a constraint response matrix $\mathbf{A}_\psi$ and a corresponding $\mathbf{b}_\psi$ at the low rate.

Due to the usage of the classic contact model, the simulation suffers from the problems described in the introduction. Therefore, we replace this classic model with our proposed accurate model. To this end, we need to compute the constraint response matrix $\mathbf{A}_\tau$ additionally to $\mathbf{A}_\psi$ at the low rate and use our proposed solver for the dynamics problems including our contact model. As result, the collision response between the objects simulated at the different rates is based on our accurate contact model with all described advantages. We emphasize that no specific method for the modeling of deformable objects is required, it just has to be compatible

to the backward Euler-based temporal integration. For further details on the asynchronous simulation approach, we refer the reader to [7].

5. Simulation with multiple haptic proxies

In this section, we describe how the presented approach can be extended to be used for multiple haptic proxies. This enables, for example, a bimanual haptic interaction as shown in the results section.

In our approach, the proxies are point like and, therefore, have no spatial extend. Consequently, they do not collide and, thus, not directly interact with each other. Therefore, in case of a rigid environment the complete haptic rendering process described in the last section could be done independently for each proxy. Nevertheless, considering the last section, if two proxies are in contact with the same deformable object, they should be able to indirectly interact with each other via the touched object. When, for instance, two proxies press from a different side on the same deformable object (see Fig. 11), they should not only feel the resistance of the object but also the force applied by the other proxy.

To allow such an interaction scenario with our approach, we have to create a combined dynamics problem. To this end, we add for each proxy $p = 1, .., n$ the corresponding additional degrees of freedom to our simulation and setup the corresponding dynamics equation for the virtual coupling using (2). These are combined with the equations for the deformable objects in one mechanical equation system in form of (4). Then, during the simulation, we setup in step L3 a set of GLCs C^p_{GLC} for each proxy p and combine them into a *single* set of constraints $C^{\text{all}}_{\text{GLC}}$. We furthermore use for each proxy its own trust regions and create the corresponding box-constraints C^p_{TR}. In step L4, we then compute *combined* constraint response matrices, $\mathbf{A}^{\text{all}}_{\psi}$ and $\mathbf{A}^{\text{all}}_{\tau}$, for all constraints, including $C^{\text{all}}_{\text{GLC}}$, C^p_{TR} $p = 1, .., n$, and C_{other}. This is done similar to (6) and (12):

$$\mathbf{A}^{\text{all}}_{\psi} = \mathbf{J}_{\text{all}}\mathbf{D}^{-1}\mathbf{J}^T_{\text{all}} \quad \text{resp.} \quad \mathbf{A}^{\text{all}}_{\tau} = \mathbf{H}_{\text{all}}\mathbf{D}^{-1}\mathbf{J}^T_{\text{all}}, \tag{17}$$

where $\mathbf{J}_{\text{all}}$ is the Jacobi for the ψ functions of all constraints and $\mathbf{H}_{\text{all}}$ the Jacobi for the tangential functions τ of all GLCs $C^{\text{all}}_{\text{GLC}}$. The matrices $\mathbf{A}^{\text{all}}_{\psi}$ and $\mathbf{A}^{\text{all}}_{\tau}$ are then used to perform the combined collision response.

The interaction between the different proxies is achieved by using the combined response matrices. As described in Section 3.2, constraint response matrices approximate how the values of the constraint functions would change for a set of constraint forces. By forming response matrices $\mathbf{A}^{\text{all}}_{\psi}$ and $\mathbf{A}^{\text{all}}_{\tau}$ for the constraints of all proxies, the forces applied by one proxy on a deformable object, therefore, influence the constraint values of another proxy that is in touch with the same object. As result, the pressure applied by one proxy can be felt by the other.

Beside the described changes in the mechanical system and the constraint sets, we have to adapt the convex decomposition of the constraint set to work with multiple proxies. As described in Section 3.3 the decomposition is done on the fly during the solving process described in Section 3.4. For the simulation with a single proxy considered up to now, we can simply perform the decomposition of the *complete* active constraint set C^a_{GLC} into the geometrically convex subsets $C^l_{cs}, l = 1, .., m$ by means of the proposed incremental method. However, this approach would result in problems when two or more proxies are used. Here, it could happen that two constraints originating from different proxies are combined in one geometrically convex constraint set C^l_{cs}. Furthermore, during the solving process, a complete convex set C^l_{cs} is treated as non-violated if at least one constraint is non-violated. Hence, if two constraints, which belong to different proxies, would be in the same set C^l_{cs}, this would mean that only one proxy needs to be in a non-penetrating state.

To resolve this problem, we have to ensure that no convex set C^l_{cs} contains GLCs originating from different proxy constraints sets C^p_{GLC}. Hence, each C^l_{cs} always has to be a subset of only one C^p_{GLC}:

$$\forall C^l_{cs},\ \exists C^p_{\text{GLC}} \text{ such that } C^l_{cs} \subseteq C^p_{\text{GLC}}. \tag{18}$$

To be able to enforce this in the decomposition performed during the solving process at *HR*, we add for each GLC the information for which proxy p it was created to the intermediate contact representation that is shared between *LR* and *HR* (see end of Section 3.3). During the incremental decomposition process, we then only combine constraints that originate from the same proxy into the same convex set C^l_{cs}. As result, all proxies are forced to remain in a non-penetrating state, which permits us to simulate multiple haptic proxies using the accurate contact model.

6. Adaptive contact region size

The constraints in the described contact model directly correspond to geometric features. Therefore, the number of constraints needed to represent a part of a surface rises with its complexity. At the same time, the computation times of the contact model strongly depend on the size of the used constraint set, as we show in the results Section 7.4. Here, the computation times are mostly determined by the calculation of the constraint response matrices, the solving procedure and other computations that are done per constraint. Due to the limited available computation time for each simulation step, we therefore can get issues when the geometries to be rendered get very complex. To alleviate this issues, we describe in the following an approach to maintain a real time simulation, by adaptively choosing the *size* of the area modeled by the constraint set.

As described in Section 3.3, the constraint set is created based on the contact entities delivered by a proximity query, which is performed with a given distance threshold. If doing this straight forward, hence, using all given entities, the size of the set would vary depending on the number of entities falling into the specified distance. Therefore, one option to control the constraint set size and, consequently, the computation time, would be to fine tune the proximity threshold. Nevertheless, such a fine tuning would be problematic for two reasons: first, the mesh resolution of the geometries may differ inside one scene and, hence, the same proximity threshold would deliver different amounts of entities depending on the current place of interaction. Second, additional constraints are used for concave edges as described in Section 3.3 and, therefore, more constraints would be created for a geometrically concave region than for a convex region with the same mesh resolution. Consequently, controlling computation times by fine tuning the proximity threshold is not appropriate.

Instead, we therefore directly limit the number of constraints that are created to maintain adequate computation times. To this end, we first empirically determine the upper bound n for the size of the constraint set that could be simulated in haptic real-time on a given hardware. This is done manually at the moment, but could be done automatically during runtime in the future. Then, if the number of entities delivered by the proximity queries exceed the upper bound n, we select only a subset of the entities to create the constraint set.

In doing so, the constraint set resulting from the selection should work well with the trust region approach described in Section 3.5. Here, a trust region is used to restrict the movements of a proxy into an area in that the geometry is completely modeled

by the constraint set. Therefore, to allow for a large range of motions, the selected constraint set should enable a large trust region while being limited in size by n. To select an appropriate set, we search for the largest sphere around the proxy in that all parts of the geometry could be reflected by a constraint set of size n.

Conceptually, we start using a sphere with zero radius and an empty selection of constraints. Then we continuously increase the radius and add constraints that intersect the sphere to our selection. We stop growing the sphere when the upper bound of n constraints is reached.

In our implementation, we perform this efficiently using a sort and sweep approach [1]. To this end, we first compute the distance of the active region of each constraint (see Section 3.3) to the proxy. The distance values are then used to create a sorted list of the constraints. Then, we sweep (iterate) over the list, which can be seen as growing the sphere using discrete steps given by the distances in the list. We finish the sweep when the set reaches the size n. By construction, none of the remaining constraints is closer than the last added constraint. Hence, we can safely use its distance value as radius for the sphere and create the corresponding trust region, while being sure every part of the surface inside is modeled by the constraint set.

As result, the area modeled by the set as well as the size of the accompanying trust region adapts during the simulation to the complexity of the geometry in contact. Hence, real time computation times are maintained for varying mesh resolutions as we show in Section 7.4. We, furthermore, discuss certain trade-offs of the method in Section 8.

7. Results and discussion

In this section, we evaluate the presented contact model in several scenarios. In our first experiments, we show the improved performance of the contact model compared to the classic contact model. Here, we compare two conditions: first, the described multi-rate simulations utilizing the proposed contact model (PCM), and second, the same multi-rate simulations but using the standard contact model (SCM) that is based on the linear inequalities (5) defined in the fundamentals Section 3.2. For PCM, the threshold for the proximity queries was set to a radius of 20 mm around the proxy. In case of SCM, this would create the locking effects described in the introduction and create significant artifacts in the haptic rendering. Therefore, we used only the closest delivered contact in this condition. In both conditions, the proximity queries delivered information on point-triangle and point-edge proximities. We first demonstrate the benefits of the PCM in a simple static and a simple deformable scenario. Afterwards, we show that the advantages are still valid for more complex scenes.

In a further experiment, we then show the possibility to simulate multiple haptic proxies in a bimanual interaction scenario. Finally, we conduct performance measurements in different scenarios to show the real-time capabilities of the presented techniques and the benefits of the adaptive contact region creation.

Throughout the evaluation, we employed a virtual coupling approach. The mass of the simulated point-proxies were set to 1 g while the spring-dampers had a stiffness of 1000 N/m and a damping constant of 0.8 Ns/m. The simulation of friction is done similar to [9]. Here, a standard constraint-based approach is used to model Coulomb's friction cone via a linear approximation. The friction coefficient was set to 0.1. For the simulation of the deformable objects, we employed a hexahedral-FEM. A linear elastic constitutive law was used to create the mechanical equations, which were updated during the simulation using the co-rotational method. In all tests, we employed a multi-rate simulation approach. To this end, the haptic loop used a time-step of 1 ms for the temporal integration and was run with an update rate of 1000 Hz. For the slow loop, we applied a time-step of 33 ms and an update rate of $\sim$30 Hz. This corresponds to a typical update rate, often used in multi-rate applications for the lower rate. The complete simulation is based on in-house developed software frameworks.

Except for the first test case, we recorded trajectories and replayed them for the measurements. For better comparability, we used the same trajectory for both conditions, PCM and SCM. The recordings were made with PCM using a Phantom Premium 1.5 connected to a common workstation with a 4 core 2.53 GHz Intel Xeon CPU with 12 GB RAM.

7.1. Haptic rendering quality

In this section, we demonstrate the increased haptic rendering quality of PCM compared to SCM for geometries with convex and concave areas.

Our first test case uses a static surface created via a sampling of a sine function with an amplitude of 5 mm and a wavelength of 25 mm. For an easier analysis, the haptic trajectory was created using a similar sinus function as for the surface but with an offset of 1 mm. Therefore, the haptic input is quickly swiped from left to right always slightly below the surface. Fig. 6 shows outtakes of the side view on the surface in the z-direction. It furthermore depicts the resulting input and proxy trajectories including their coupling. Additionally, the x- and y-components of the haptic forces are indicated for each proxy position. In the top-left plot, the results of the SCM show that the proxy trajectory regularly moves away from the surface. This happens due to the usage of constraints which are actually already out-dated. Then, the proxy snaps back when an updated constraint set arrives from the low rate. In consequence, the corresponding haptic forces increase unnaturally and are non-smooth in direction and magnitude. The plots for the PCM show a proxy trajectory that always stays on the surface. Therefore, the forces are smoother with respect to their magnitude and direction. The slight bumps reflect the sampling of the rendered geometry and get smaller with a higher resolution of the surface geometry.

To evaluate our proposed contact model in combination with deformable objects, we embedded the above described surface into a deformable mesh (see Fig. 7). The mesh was fixed on the left and the right side. Here, we recorded a haptic trajectory of a fast swipe over the surface from left to right. The results are given in the bottom row of Fig. 6. The plots show that in case of SCM the proxy trajectory does not reflect the sinus shape of the surface. Furthermore, the forces are discontinuous and contain large jumps with respect to magnitude and direction. In case of PCM, the proxy trajectory follows the sinus shape of the surface. This results in rather smooth forces without abrupt changes.

We also performed tests with two less synthetic use cases. The first one employs a “rocker arm” and the second a “Fertility statue” geometry (see Fig. 8). The scenarios are more challenging as more complex contact configurations occur. The transition from the statue to its base, for instance, contains areas which are convex in one direction and concave in another. Such a contact situation is also shown on the top right of Fig. 8; it also depicts the GLC set, which is active at the end of the haptic simulation step. The correct convex decomposition of the set by our approach is illustrated via a color coding.

We evaluated the quality of the haptic forces created by PCM and SCM with a rigid and a deformable version of the “Fertility statue” geometry. The used trajectory is shown as a blue line on the top left of Fig. 8. The haptic forces for the rigid version are

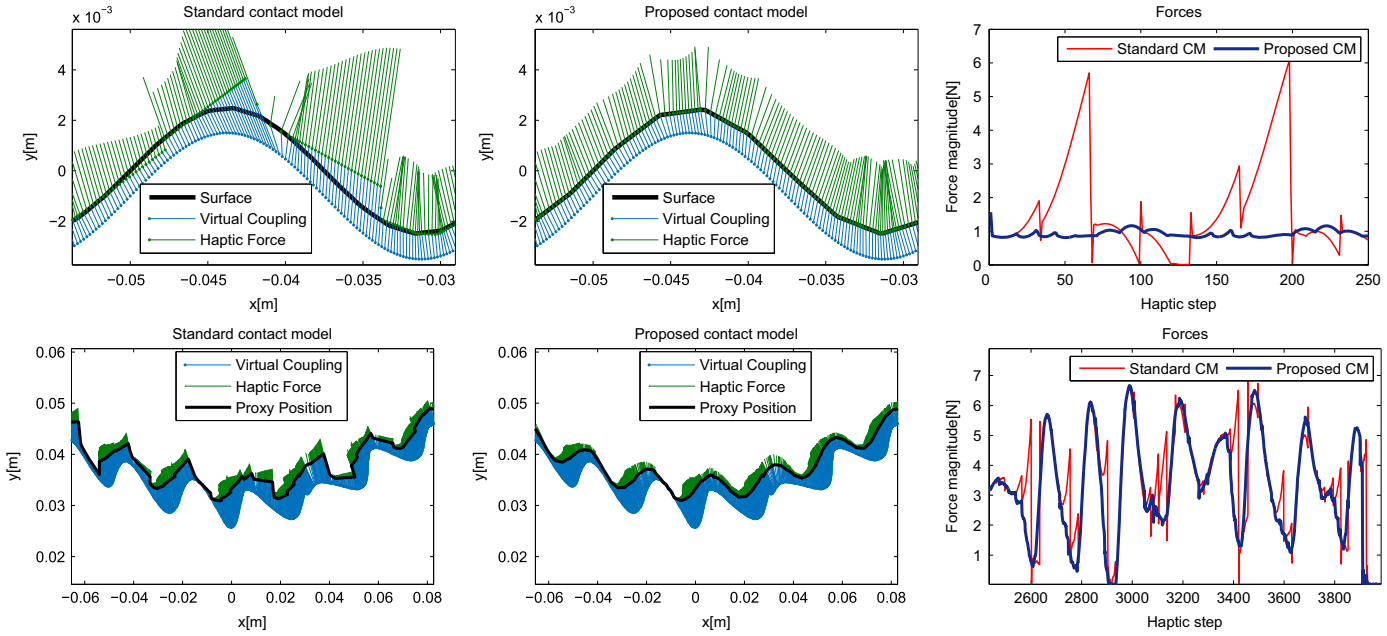

Fig. 6. The top row gives the results for the static scenario and the bottom row for the deformable. The left four plots show outtakes of the simulation trajectories and forces. The input positions are indicated by the blue dots and the proxy positions by green dots. Input and proxy positions from the same time-step are connected by a blue line. The green lines reflect the forces rendered for each proxy position. To this end, the *x*- and *y*-components of the forces are plotted. Although the scaling is arbitrary, the lines provide information about how continuous the forces are with respect to their magnitude and direction. The absolute magnitudes of the forces are reported in the right two plots. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

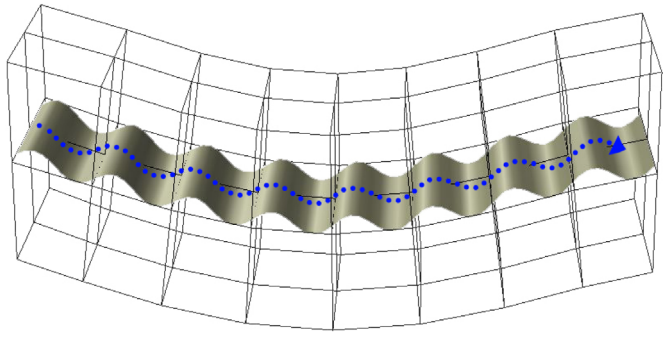

Fig. 7. Surface generated from sine function embedded in a deformable mesh, which is fixed on the left and right side. The input trajectory used in the experiments is illustrated by the dotted blue line. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

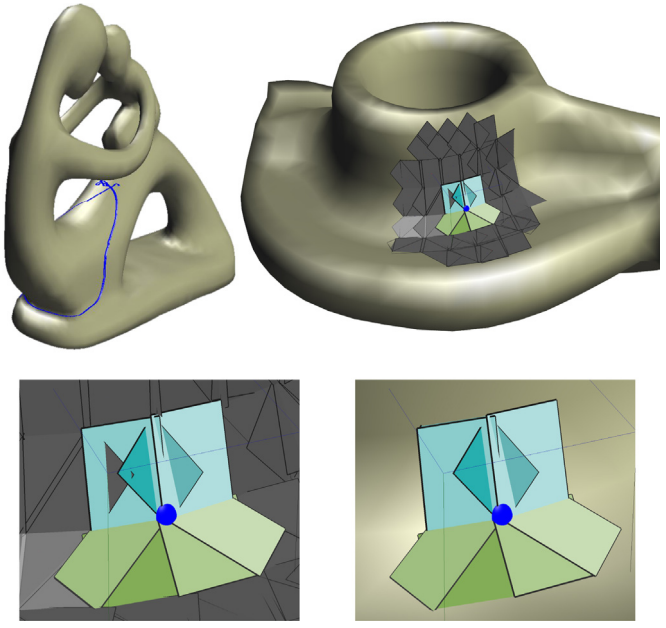

Fig. 8. (Top left) Geometry of "Fertility" statue including the tested haptic trajectory indicated by the blue line. (Top right) Contact situation with a convex–concave surface. The haptic proxy is indicated by the blue dot. The GLCs in the current constraint set C_{GLC} are visualized in gray and the ones which are in their active region C^a_{GLC} are colorized with respect to the convex decomposition. Here, we have two convex sets C^1_{cs} and C^2_{cs} indicated by blue respectively green. The GLCs in each set are, furthermore, distinguished by different levels of saturation. The border of each GLCs is outlined with a black frame. (Bottom left) Close-up view on active region. (Bottom right) Close-up view showing only the convex decomposition. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

reported in Fig. 9 and the ones for the deformable version are given in Fig. 10. In both cases, rigid and deformable, the PCM generated rather smooth forces while the ones generated by SCM contain jumps and hard kinks.

7.2. Simulation with multiple haptic proxies

To show the capabilities of the simulation to handle multiple haptic proxies, we performed a test with a bimanual haptic rendering scenario. In our test, we used a bar like deformable object that is fixed at the left and right side as shown in Fig. 11. To create the test scenario, the input for the haptic proxy of hand two was first moved upwards to push against the object from below and create a deformation of the object. Then, during the measurements, the input for hand two stayed the same, while hand one explored the surface from left to right with a slight pressure using a second haptic proxy. The magnitudes of the resulting haptic forces for both hands are depicted in Fig. 12. The plots show that when hand one comes closer to the second, the forces felt at both hands increase. This demonstrates that the applied pressure is transferred from one hand to the other via the deformable object.

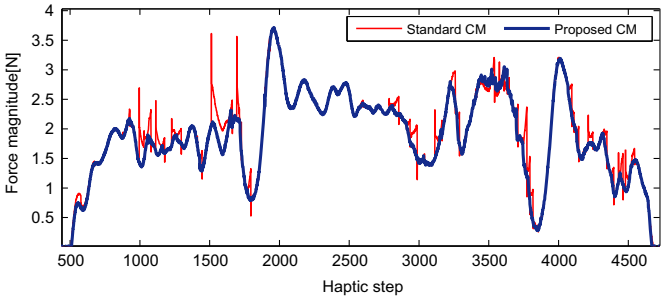

Fig. 9. Haptic forces for complex rigid scenario.

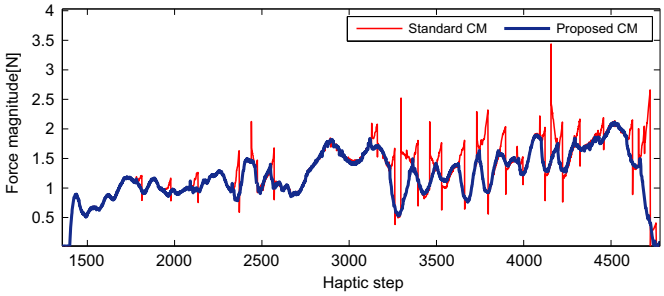

Fig. 10. Haptic forces for complex deformable scenario.

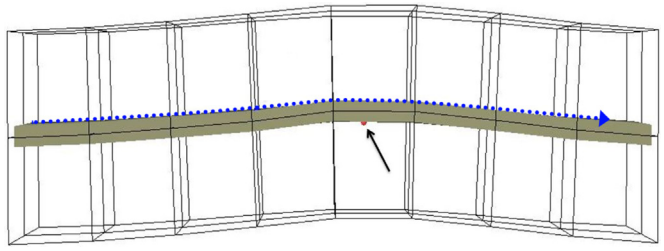

Fig. 11. Bimanual interaction scenario. The input trajectory used in the experiments for the first hand is illustrated by the dotted blue line. The fixed input position for the second hand used during the measurements is shown as red dot below the surface and is indicated by the black arrow. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

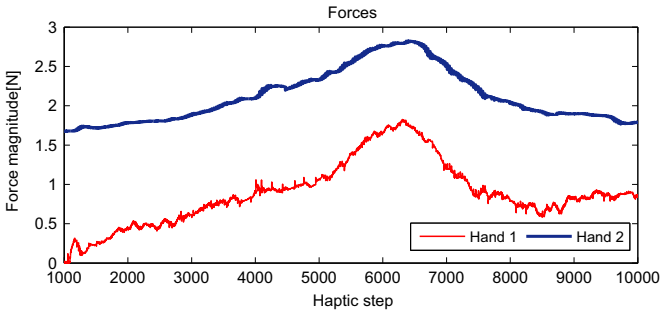

Fig. 12. Haptic forces for bimanual interaction scenario.

7.3. Performance measurements

In the following, we demonstrate the real-time capabilities of our approach in several scenarios.

We first measured the key performance data for our most complex test case, the deformable "Fertility statue". Fig. 13 shows

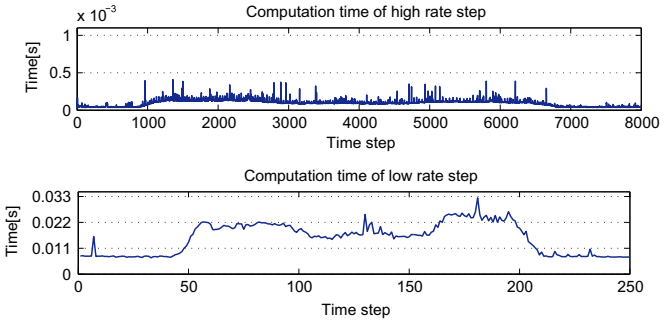

Fig. 13. Computation times for haptic rendering of complex deformable scene with a single proxy.

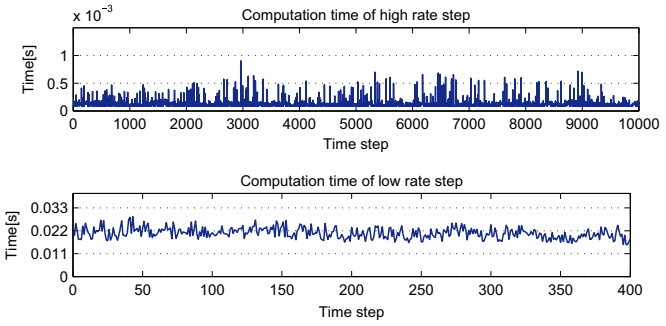

Fig. 14. Computation times for bimanual rendering of deformable scene.

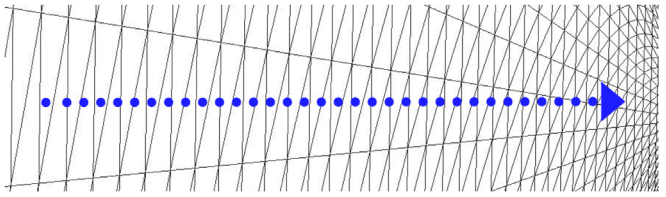

Fig. 15. Geometry with inhomogeneous mesh resolution. The dotted blue line illustrates the input trajectory used in the experiments. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

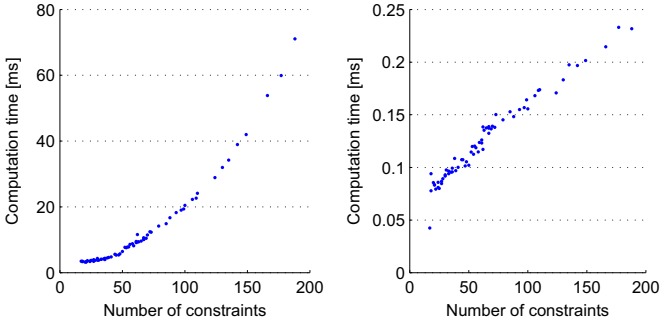

Fig. 16. Scatterplot showing correlation between computation times and the number of constraints. (Left) low rate, (right) high rate.

the computation times measured for the low (*LR*) and for the high (*HR*) simulation rate. Here, the computation time for a *LR* step had a max value of 32.3 ms and a mean of 15.2 ms. Therefore, the aimed update rate of 30 Hz could be maintained. The computation time for a *HR* step had a max value of 0.41 ms and a mean of 0.09 ms. Consequently, the critical haptic update rate of 1 kHz could be easily achieved as well.

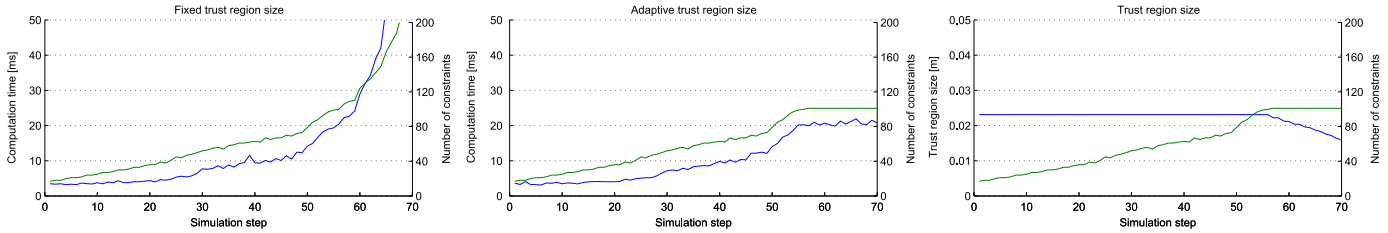

Fig. 17. (Left) Plot showing the number of constraints in green together with the resulting computation times in blue for a simulation with an non-adaptive contact region size. (Middle) Plot showing the results for the simulation with an upper bound for the number of constraints. (Right) Plot showing the size of the contact region in blue and how it decreases when the number of used constraints is bound to an upper limit. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

We furthermore evaluated the performance in the bimanual haptic rendering scenario described in Section 7.2. The measured computation times are shown in Fig. 14. Here, a *LR* step took at max 28.8 ms and on an average 20.8 ms, while the time for a *HR* step was at max 0.9 ms and on average 0.01 ms. Thus, a real time simulation could also be achieved for this scenario.

During our tests, the solver always converged with a constraint error threshold of 0.1×10^{-10} within less than 30 iterations and 0.5 ms. We used an absolute error measure based on the sum of all constraint violations. The number of GLCs in the constraint set C_{GLC} usually ranged between 50 and 100. Furthermore, the constraint set included two additional constraints for each GLC to simulate friction.

7.4. Adaptive contact region size

The geometries of the scenarios tested in the previous section have rather homogeneous mesh resolutions that do not require a dynamic adjustment of the contact region. In the following, we show the benefits of the adaptive contact region approach in case of rendering geometries with an inhomogeneous resolution such as the one shown in Fig. 15. In our tests, we moved the haptic input from the areas with the low resolution to the ones with the high resolution.

As first condition, we measured the computation times with our contact model and a fixed contact region size and, hence, no upper limit for the constraint set size. Fig. 16 shows the correlation between the number of constraints and the computation times of the *HR* and the *LR* simulation step. Besides showing a clear correlation, it helps us choose the upper bound for the number of constraints that could be used without violating the real time constraints. While the upper limit of 1 ms for the high rate simulation is not reached at all, the upper limit of 33 ms for the low rate is reached at about 120 constraints.

As second condition, we then measured the computation times for an adaptive contact region size, which is induced by limiting the constraint set size to 100 constraints. We choose a value below the 120 constraints to be sure that real time is achieved. Fig. 17 shows the computation times of the low rate together with the size of the constraint set for both conditions. While the computation times for the first condition rise above the threshold of 33 ms, the ones for the second condition remain below.

These results show that the adaptive contact region size permits interactive simulation rates also for geometries with varying resolutions by balancing the size of the modeled region and computation time.

8. Conclusion and future work

In this paper, we extended an accurate contact model for multi-rate single-point haptic rendering of static and deformable environments to make the simulation with multiple haptic proxies possible. The contact model facilitates the creation of accurate intermediate contact representations. These enable a correct treatment of geometrically concave and convex contact scenarios at a haptic simulation rate, while reflecting the full dynamics of the involved objects including compliant behavior of deformable objects. Other constraints modeling, for instance, friction, can be easily integrated into the model. The resulting constrained dynamics problems are robustly solved using a novel efficient solving scheme utilizing techniques from trust region-based optimization.

The quality of the haptic rendering is demonstrated for static and deformable scenarios with different complexities. The evaluation furthermore shows that the extension to multiple proxies permits a bimanual haptic rendering of deformable objects. In doing so, both hands can interact with the same deformable object and feel the pressure applied to the object by the other hand. We, furthermore, introduce a method to automatically adapt the region modeled by the intermediate contact representation to the complexity of the geometry in contact. Thereby, we are able to maintain a stable haptic real time simulation for varying mesh resolutions, which is shown several experiments.

However, to allow for this, the haptic proxy is forced to stay in the region modeled by the contact representation. This is a tradeoff necessary to prevent the proxy from tunneling through the unmodeled parts of the geometry and maintain the rendering stability. The downside is that in case of fast movements and a high resolution of the geometry, the proxy may reach the boundary of the modeled region and be stopped there. Therefore, future research should be directed to minimize the effects of these limitations.

However, due to performance reasons, the size of the constraint set that can be used is limited. In the presented approach, this set is used to model a region that is a cubic volume with its center in the starting position of the proxy at the current time-step. This is done under the assumption that, in the period the set is in use, the proxy can possibly move in every direction. Although, this is certainly true, in case the proxy is already moving with some speed in a particular direction, its future movement will probably tend more or less to this direction as well. Therefore, future research could aim at better adapting the position and shape of the modeled region to cover the intended movements of the user based on some kind of heuristic. One possibility would be a rectangular cuboid, which encloses the same number of GLCs as the cubic region used before, but that is located around the line between the starting position of the proxy and its position after a free non-constraint motion (see Section 1). Thereby, it may be possible to reduce the probability that the proxy is stopped without increasing the number of GLCs.

Another interesting research direction could aim at increasing the volume of the region with the same number of GLCs. This could maybe achieved by combining the presented approach with multi-resolution collision detection methods as described by, e.g., Otaduy and Lin for rigid objects [22] or by Barbic and James for

deformable objects [14]. Here, one could try to use a high resolution for the area directly around the proxy and a lower resolution for areas farer away.

Acknowledgments

This project has received funding from the European Union's 7th Framework Program for research, technological development and demonstration under Grant agreement 610425. Furthermore, we thank AIM@SHAPE for providing the data sets used in the evaluation.

References

[1] Erleben K, Sporring J, Henriksen K, Dohlman K. Physics-based animation. Hingham, Massachusetts: Charles River Media, Inc.; 2005 ISBN 1584503807.
[2] Nguyen B, Trinkle J. Modeling non-convex configuration space using linear complementarity problems. In: International conference on robotics and automation; 2010.
[3] Anitescu M, Potra F. Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems. Nonlinear Dynamics 1997;14(93):231–47. http://dx.doi.org/10.1023/A:1008292328909.
[4] Duriez C, Dubois F. Realistic haptic rendering of interacting deformable objects in virtual environments. IEEE Trans Vis Comput Graph 2006;12:36–47.
[5] Johnson D, Willemsen P. Six degree-of-freedom haptic rendering of complex polygonal models. In: 11th symposium on haptic interfaces for virtual environment and teleoperator systems. IEEE Computer Society: Los Angeles; 2003. p. 229–35. ISBN 0-7695-1890-7.
[6] Otaduy M, Lin M. A modular haptic rendering algorithm for stable and transparent 6-DOF manipulation. IEEE Trans Robot 2006;22(4):751–62.
[7] 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 international conference on intelligence robots and systems; 2013. p. 2010–6.
[8] Knott TC, Kuhlen TW. Accurate contact modeling for multi-rate single-point haptic rendering of static and deformable environments. In: Zachmann FJ, Zara F, editors. Workshop on virtual reality interaction and physical simulation. The Eurographics association; 2015. p. 71–80 ISBN 978-3-905674-98-9.
[9] Knott T, Kuhlen T. Geometrically limited constraints for physics-based haptic rendering. In: Haptics: neuroscience, devices, modeling, and applications, Lecture Notes in Computer Science, vol. 8619; 2014. p. 343–51. ISBN 978-3-662-44195-4.
[10] Adams R, Hannaford B. Stable haptic interaction with virtual environments. IEEE Trans Robot Autom 1999;15(3):465–74.
[11] Zilles CB, Salisbury JK. A constraint-based god-object method for haptic display. In: International conference on intelligent robots and systems; 1995.
[12] S. Chan, N. H. Blevins, and K. Salisbury, "Deformable haptic rendering for volumetric medical image data," in 2013 World Haptics Conference, WHC 2013, 2013, pp. 73–78. http://dx.doi.org/10.1109/WHC.2013.6548387.
[13] Wan M, McNeely W. Quasi-static approximation for 6 degrees-of-freedom haptic rendering. In: IEEE Visualization 2003. IEEE Computer Society: Seattle; 2003. p. 34. ISBN 0-7803-8120-3.
[14] Barbic J, James D. Six-DoF Haptic rendering of contact between geometrically complex reduced deformable models. IEEE Trans Haptics 2008;1(1):39–52.
[15] Otaduy Ma, Tamstorf R, Steinemann D, Gross M. Implicit contact handling for deformable objects. Comp Graph Forum 2009;28:559–68.
[16] Otaduy M, Lin M. Haptic rendering: foundations, algorithms, and applications. Wellesley, Massachusetts: AK Peters; 2008.
[17] Peterlik I, Duriez C, Cotin S. Asynchronous haptic simulation of contacting deformable objects with variable stiffness. Intell Robots Syst 2011:2608–13.
[18] Debunne G, Desbrun M, Cani MP, Barr AH. Dynamic real-time deformations using space & time adaptive sampling. ACM SIGGRAPH 2001:31–6.
[19] Wang D, Shi Y, Liu S. Haptic simulation of organ deformation and hybrid contacts in dental operations. IEEE Trans Haptics 2014;7(1):48–60.
[20] Ericson C. Dealing with "Nondecomposable" concave geometry. In: Real-time collision detection; 2005. p. 506–7.
[21] Yuan Y-x. A review of trust region algorithms for optimization. In: Proceedings of international congress on industrial and applied mathematics; 2000. p. 271–82.
[22] Otaduy MA, Lin M. Sensation preserving simplification for haptic rendering. In: Proceedings of ACM Siggraph, vol. 1. San Diego: ACM Press; 2003. p. 543–53.