A Hybrid Constraint-Penalty Proxy Method for Six Degree-of-Freedom Haptic Display of Deforming Objects

Stuart A. Bowyer Ferdinando Rodriguez y Baena*
Imperial College London

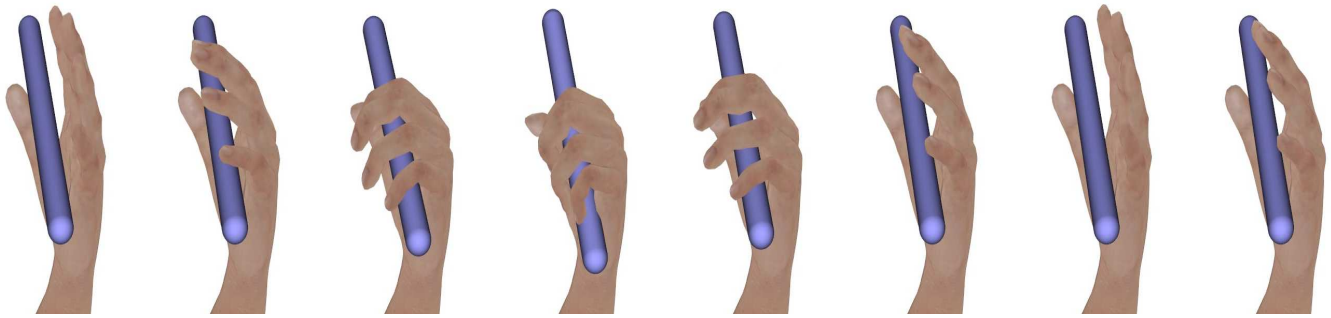


Figure 1: *Sequence of frames showing the hybrid proxy (purple) interacting with a simulated hand model making a gripping movement. The master device pose is shown in green for frames one to three.*

Abstract

There are many applications and tasks in which the precise, high-fidelity haptic display of deforming objects is required. A crucial element in haptic rendering is the definition of a proxy pose that follows the motion of the user, while respecting the geometry of the object being displayed. Conventional methods for computing the dynamics of a proxy interacting with a deforming object suffer from several issues relating to numerical instabilities when the proxy becomes over-constrained and high computational demands. This paper presents a novel hybrid proxy that combines modified versions of constraint-based and penalty-based proxies together to give high fidelity rendering with reduced computational requirements and robustness to situations where the proxy becomes enclosed. Experimental analysis of the proposed method shows that it can efficiently compute proxy dynamics that faithfully render the required object. This research forms a basis for further development of novel hybrid dynamic proxies for haptics and allows for increasingly complex deforming geometries to be rendered.

Keywords: Haptics, proxy, god-object, gauss' principle of least constraint, deforming objects

Concepts: •**Human-centered computing** → **Haptic devices;** •**Computing methodologies** → **Physical simulation; Virtual reality;**

*e-mail:f.rodriguez@imperial.ac.uk

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org. © 2016 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM Symposium on Virtual Reality Software and Technology (VRST), November 2-4, 2016, Münich, Germany.

1 Introduction

Though they are widespread in every-day-life, the haptic rendering of deforming objects presents a significant challenge for the majority of haptic algorithms currently described within the literature. It is generally necessary to trade haptic fidelity against computational performance and numerical robustness.

Proxies, also known as 'god objects', were initially proposed for haptic rendering in virtual environments, and involve computing the motion of a simulated version of a tool which follows the actual tool, while not penetrating any virtual objects within its workspace [Zilles and Salisbury 1995; Ruspini et al. 1997]. In general, proxy methods employ a virtual linkage formed of a spring, and possibly a damper, between the actual tool and the proxy tool to define its ideal motion, and then restrict the motion using a range of techniques.

Proxies present many advantages for the haptic rendering of complex objects. One of the most important is that in most methods there is temporal coherence between the poses generated using a proxy. The simulated dynamics of the proxy means that it is unlikely to make large discrete jumps in position within the workspace. This ensures that the feeling to the user is realistic and pleasant.

Another advantage of proxies is that they prevent 'pop-through', when the actual tool reaches the middle of a constrained region. Pop-through is where the target pose suddenly switches sides of a constrained region, such that users pass completely through it, rather than retract from the side of the object they were initially interacting with.

There are three primary approaches within the haptics literature for modulating the effects of the virtual linkage in such a way that the proxy does not penetrate any virtual objects. 'Penalty-based methods' compute a set of penalty forces upon the proxy, based on the magnitude of its penetration into the other objects [McNeely et al. 1999; Otaduy and Lin 2006; Barbic and James 2008]. By combining the penalty forces from the proxy penetration points with the forces from the virtual linkage, the overall motion of the proxy can be computed which minimises penetration while following the actual tool. Though penalty-based methods can be simple to implement, the fact that there must be some penetration before the

penalty forces are generated means that there will be some inaccuracy in the proxy placement, and hence reduction in fidelity. An additional issue with certain penalty based methods in a dynamic environment is that the movement of a surface can suddenly release an embedded proxy. When this happens the proxy can suddenly accelerate producing discontinuities in the rendering. 'Impulse-based methods' generate impulse response forces from each contact between the proxy and rendered objects [Mirtich and Canny 1995]. These methods perform well in certain configurations; however, it is generally necessary to integrate penalty forces in order to prevent slow penetrations during resting contacts [Constantinescu et al. 2004; Constantinescu et al. 2005].

When the smoothness of proxy motion is paramount, and all proxy penetration of the rendered object should be avoided to ensure fidelity, a 'constraint-based method' for computing proxy motion can be used. Within a constraint-based method, contact information is used to mathematically constrain the proxy's motion along the surface, preventing any penetration [Baraff 1989; Salisbury and Tarr 1997; Redon et al. 2002; Ortega et al. 2007; Wang et al. 2011]. The primary advantage of constraint-based methods is that they guarantee that the proxy will not penetrate any objects and its motion will be smooth and consistent. However, the process generally requires convex optimisation to establish the constrained projection of the unconstrained motion, and often continuous collision detection, both of which can be computationally demanding.

The fact that constraint-based methods do not penetrate the objects being rendered gives them a significant advantage for haptic rendering as they can more precisely display the object's geometry. However, for deforming geometries, their disadvantages, in terms of computational demand and lack of numerical robustness are challenging.

This paper describes how a hybrid combination of a constraint-based proxy and a penalty-based proxy can be used to overcome these challenges with rendering deforming objects.

2 Method

2.1 Computing Constraint-based Motion using Gauss Principle

A well established method for computing constraint-based proxy motion with six degrees of freedom is described in [Ortega et al. 2007], based on [Redon et al. 2002]. This method is briefly described here for completeness. The first stage in defining the motion of a proxy, at a given time-step, is to compute its unconstrained acceleration $\mathbf{a}_u \in \mathbb{R}^6$, due to the virtual linkage between it and the physical tool, as follows:

$$\mathbf{a}_u = \mathbf{f}_{vl} = K_{vl}\mathbf{x}_{vl}, \tag{1}$$

assuming unit mass and inertia, where $\mathbf{a}_u \in \mathbb{R}^6$ is the virtual linkage force, $K_{vl} \in \mathbb{R}^{6\times6}$ is the diagonal stiffness matrix for the virtual linkage and $\mathbf{x}_{vl} \in \mathbb{R}^6$ is the extension of the virtual linkage, computed from the poses of the proxy and the physical tool:

$$\mathbf{x}_{vl} = \left[\mathbf{x}_{t,vl}, \mathbf{x}_{r,vl}\right], \tag{2}$$

where $\mathbf{x}_{t,vl} \in \mathbb{R}^3$ is the translational extension and $\mathbf{x}_{r,vl} \in \mathbb{R}^3$ is the rotational extension in SO(3).

From Gauss' principle, it is known that the constrained acceleration of an object $\mathbf{a}_c \in \mathbb{R}^6$ (in this case the proxy) is the one which minimises the, so-called, 'kinetic distance' between the unconstrained and constrained accelerations:

$$\mathbf{a}_c = \underset{\mathbf{a}\in A_c}{\arg\min} \frac{1}{2}(\mathbf{a}-\mathbf{a}_u)^T M(\mathbf{a}-\mathbf{a}_u), \tag{3}$$

where A_c is the set containing all permissible accelerations of the body which do not penetrate the surrounding objects and $M \in \mathbb{R}^{6\times6}$ is the proxy's diagonal mass matrix.

Within the constraint-based proxy method, the set A_c is defined using a contact Jacobian $J_c \in \mathbb{R}^{6\times n}$ which defines the separating acceleration of the proxy, along the virtual object surface normals, at n discrete points where it makes contact with the virtual objects within its workspace. i.e. when the contact Jacobian is multiplied by the acceleration of the centre of mass of the proxy, it produces a vector of length n which defines the linear acceleration of the proxy away from the surrounding virtual objects at each contact point, along the contact normals. To ensure that the proxy does not penetrate the virtual objects, it is necessary for all elements of the produced vector to be non-negative. Formally, J_c defines A_c as follows:

$$A_c = \left\{\mathbf{a} : \mathbf{a} \in \mathbb{R}^6, J_c\mathbf{a} \geq 0\right\}. \tag{4}$$

Establishing $\mathbf{a}_c$ from equations (3) and (4) is a quadratic programming problem and can be solved in a variety of ways, e.g. [Wilhelmsen 1976; Boyd and Vandenberghe 2004]. Once $\mathbf{a}_c$ is known at a given time, it is necessary to integrate the motion of the proxy across the subsequent time-step, in the direction of $\mathbf{a}_c$. As the constraint-based method relies on surface contacts, continuous collision detection is used for the motion across the time-step to ensure that its motion is stopped if the object makes contact with any other objects in its workspace. If collisions are detected, then J_c is recomputed from the new contact points and a new value for $\mathbf{a}_c$ is solved.

This process of computing constrained acceleration and continuous collision detection for a simplified proxy is shown in figure 2. It can be seen that the proxy moves in discrete steps, ensuring that it does not penetrate the object being rendered.

2.2 Discrete Range Queries

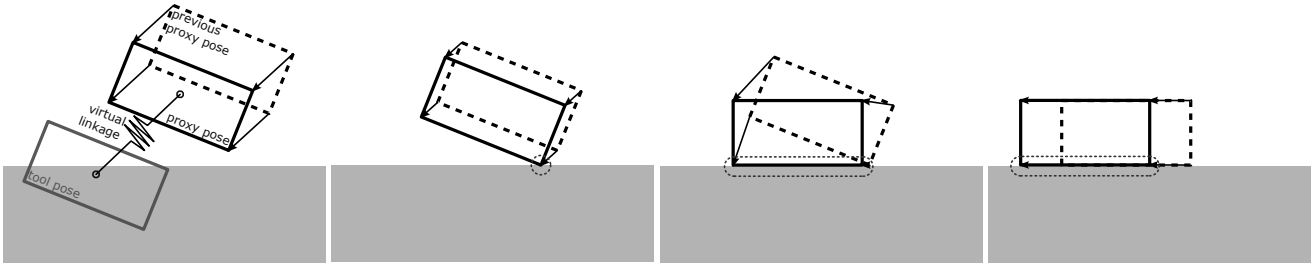
There are two reasons which make implementing the above method with a high resolution deforming object challenging. The first is that the continuous collision detection is computationally demanding for a deforming body. As explained in [Ericson 2004], collision detection is difficult in deforming bodies because conventional search structures become invalidated after deformation. Most continuous collision detection methods rely on repeated discrete collision detection searches and this can be prohibitively complex.

The second reason is that the deformation of the body can cause a proxy to become embedded within the virtual object when it deforms, and the algorithm needs to be able to resolve this. The definition of A_c in equation (4) only ensures that surface contacts do not become penetrations due to the proxy motion, it does not eliminate penetrations which have already occurred.

A solution to these problems, proposed here, is to linearly approximate the proxy's motion and compute the contact Jacobian based on a wider set of points which are 'close to' contacts, or already in penetration. Based on the expected velocity of the constrained tool, it is possible to estimate an upper bound for the displacement of all points on the surface of the proxy in a single time-step Δx_{max}, as follows:

$$\Delta x_{max} = \Delta t \left(\dot{x}_{max} + \omega_{max} r_{max}\right), \tag{5}$$

where Δt is the time-step, $\dot{x}_{max}$ is the estimated maximum translational velocity of the tool's centre of mass, ω_{max} is the estimated maximum angular velocity of the tool's centre of mass and r_{max} is the maximum radius of any point on the tool from its centre of mass.



(a) $t = t_1$ **(b)** $t = t_2$ **(c)** $t = t_3$ **(d)** $t = t_4$

Figure 2: *Illustration of the conventional proxy motion scheme used in [Ortega et al. 2007], where the actual tool (not shown in frames (b)-(c)) is moving down and to the left while a virtual linkage pulls the rectangular proxy towards it. The dark grey region represents the object being haptically rendered. In this method, continuous collision detection is used between each time step to ensure that the proxy does not penetrate the surface. Highlighted at each time-step are the contacts between the proxy and surface which will be used to define the constrained acceleration of the proxy at the subsequent step.*

By identifying all primitive pairs between the proxy and virtual objects which are within the upper displacement limit, assuming this is small, a contact Jacobian can be assembled which ensures that none of these point pairs penetrate during the following time-step. As all potential contact locations for the subsequent time-step are considered by the single contact Jacobian, continuous collision detection is not required and the proxy should not enter the virtual object being rendered.

As close contact point pairs are not necessarily collocated at the start of the time-step, simply ensuring their relative acceleration is non-negative is not sufficient to generate correct motion which follows the surface of the virtual object. Instead, the accelerations at each close contact point must be set individually, based on their proximity, such that, at the end of the next time-step, they remain non-penetrating. This formulation for constraining the acceleration of the proxy also resolves the issue of penetrations due to surface deformation because it can recover point pairs which have penetrated. Rather than having negative limits for the acceleration for each contact point, which prevent penetrations from occurring during the subsequent time-step, a positive limit can be set for the acceleration of an already penetrating contact point which ensures that the proxy accelerates away from the contact such that the penetration is eliminated by the end of the time-step. This is similar to the method presented in [Ortega et al. 2007] for recovering surfaces after integration errors.

In the likely case that the object being rendered is represented as a triangular mesh, a range query can identify all triangles within the mesh which are within a pre-defined distance of the proxy. If the range value r is the proxy displacement upper bound:

$$r = \Delta x_{max}, \tag{6}$$

then all triangles found by the range query should have close contact points included within the contact Jacobian so that the constrained acceleration ensures that they are non-penetrating by the end of the following time-step. This range query variation on the conventional constraint-based method is illustrated in figure 3.

To implement this method using Gauss' Principle it is necessary to substitute the set of permissible accelerations based on contacts, A_c, in equation (3) with a set of permissible accelerations based on close contacts, A_p, defined as follows:

$$A_p = \left\{ \mathbf{a} : \mathbf{a} \in \mathbb{R}^6, J_p \mathbf{a} \geq \mathbf{b} \right\}, \tag{7}$$

where $J_p \in \mathbb{R}^{6 \times m}$ is the proximity Jacobian for the m points which are within the close contact region and $\mathbf{b} \in \mathbb{R}^m$ is the acceleration limit vector for each of the close contact points. The i-th element of the acceleration limit vector $\mathbf{b}_i$ can be computed from the close contact point details, based on the standard equations of motion, as follows:

$$\mathbf{b}_i = 2 \frac{\Delta x_{pr,i} - \dot{x}_{pr,i} \Delta t}{\Delta t^2}, \tag{8}$$

where $\Delta x_{pr,i} \in \mathbb{R}$ is the close contact point pair separation projected onto the virtual object surface normal $\hat{n}_i$, $\dot{x}_{pr,i} \in \mathbb{R}$ is the close contact point pair relative velocity projected onto the virtual object surface normal and Δt is the time-step. The motion projections are computed using the following equations:

$$\Delta x_{pr,i} = (\mathbf{x}_{ac,i} - \mathbf{x}_{pr,i}) \cdot \hat{\mathbf{n}}_i \tag{9}$$

$$\dot{x}_{pr,i} = -\dot{\mathbf{x}}_{pr,i} \cdot \hat{\mathbf{n}}_i, \tag{10}$$

where $\mathbf{x}_{ac,i} \in \mathbb{R}^3$ is the Cartesian position of close contact point i on the object mesh, $\mathbf{x}_{pr,i} \in \mathbb{R}^3$ is the Cartesian position of close contact point i on the proxy and $\dot{\mathbf{x}}_{pr,i} \in \mathbb{R}^3$ is the Cartesian velocity of close contact point i on the proxy.

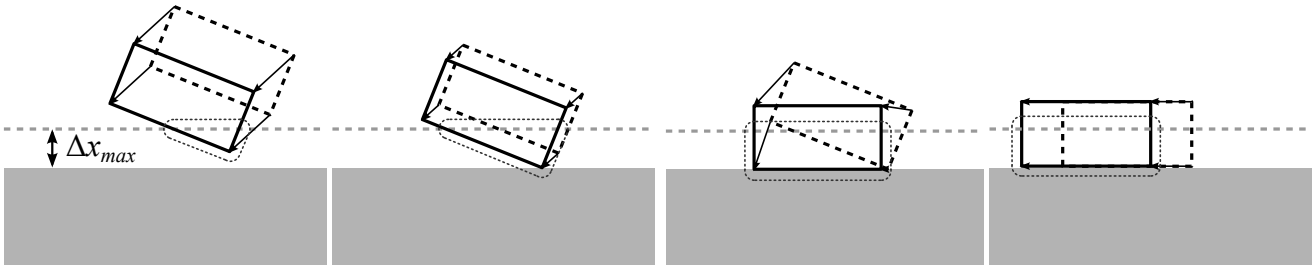
To ensure that the proxy's motion along the surface was smooth, two adaptations were made to the above formulation. The first was that edge normals were computed explicitly in equations (9) and (10) when the close point on the object mesh $\mathbf{x}_{ac,i}$ was on the edge of a triangle rather than using the surface normal of the triangle. The edge normals were computed using the following equation:

$$\hat{\mathbf{n}}_i = \frac{\mathbf{x}_{pr,i} - \mathbf{x}_{ac,i}}{\|\mathbf{x}_{pr,i} - \mathbf{x}_{ac,i}\|}. \tag{11}$$

This formulation for the normals ensured that the convex edges of the object were rounded rather than sharp and did not cause erratic motion in the proxy. The second modification was that the set of close contacts identified by the range query were filtered to remove any which had inconsistent normals. This was achieved by searching the set of close contacts and removing any which had closest-point vectors that intersected other triangles that were also in the close contact set.

2.3 Inconsistent Contact Constraints

Inconsistent contact constraints occur when the deformation in one of the contacting objects encapsulates the other object. Physically, this is representative of gripping or clamping one object by the other; however, in the case of (modelled) incompressible objects, there is no acceleration which can prevent penetration (i.e.



(a) $t = t_1$ **(b)** $t = t_2$ **(c)** $t = t_3$ **(d)** $t = t_4$

Figure 3: *Illustration of the proxy motion scheme used here which avoids the necessity for continuous collision detection. The actual tool (not shown) is moving down and to the left while a virtual linkage pull the rectangular proxy towards it. The dark grey region represents the object being haptically rendered and the dashed grey line represents the boundary inside which close contacts are considered, Δx_{max} from the object's surface. In this method, range queries are used between the proxy and object's surface to identify close contacts, as highlighted, which are used to form the contact Jacobian.*

$A_p = \emptyset$). For a given geometric configuration of a contacting proxy, it is possible to deduce whether the contact constraints form an inconsistent equality using the method described in [Kuhn 1956].

As described previously, there are many advantages to constraint-based haptic rendering with proxies. Therefore, it was decided that a hybrid approach would be beneficial which switches between proxy types dependent on the constraint configuration. This meant that the advantages of constraint-based rendering could be used for the majority of the time, and then a more robust method could be used when the proxy was gripped/clamped by the environment. The penalty method described in [McNeely et al. 1999] was adapted for this purpose. As long as the contact stiffnesses are suitably set, this penalty based method is robust and smooth when the proxy becomes encapsulated.

Specifically, the summed penalty force/torque vector $f_p \in \mathbb{R}^6$ was computed as follows:

$$f_p = -J_p \Delta x_{pr} + K_{vl} \mathbf{x}_{vl} \tag{12}$$

Again, assuming unit mass and inertia, the penalty-based constrained acceleration is therefore:

$$a_p = -J_p \Delta x_{pr} + K_{vl} \mathbf{x}_{vl} \tag{13}$$

The switching logic of the hybrid approach was initially set to change from the constraint-based method to the penalty-based method whenever it was computed that no solution existed to equation (7). However, this led to unsatisfactory oscillations of the proxy as the two methods had different steady-state poses for the proxy. The solution to this was to swap from the constraint-based method to the penalty-based method whenever an invalid configuration was detected, but to ensure that the system could not change back for at least 50 ms (determined empirically). This latency in swapping ensured the proxy was fully free from the object before the constraint-based dynamics were reintroduced.

2.4 System Implementation

Figure 4 illustrates a system for the use of the hybrid proxy in haptic rendering. The system proposed is a multi-threaded one, in which three primary components run asynchronously on a multi-core CPU. The three threads are for: haptic rendering; hybrid proxy simulation; and deformation modelling. This system is similar to

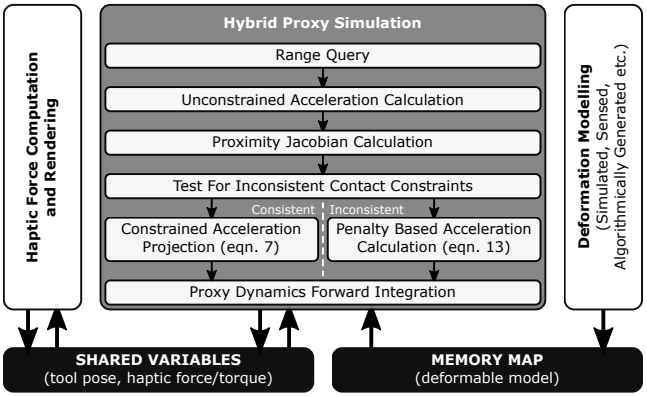


Figure 4: *Illustration of a multi-threaded haptic rendering implementation using the hybrid proxy.*

that in [Ortega et al. 2007]; however, the force computation and rendering are combined into a single thread and the deformation modelling is included. Information is exchanged between the threads using simple shared variables and memory maps. For the experiments in the following section, this system was implemented using the Open Robot Control Software (OROCOS) [Bruyninckx 2001].

2.4.1 Deformation Modelling Thread

The deformation modelling thread defines object geometries in some way and writes them to shared memory maps for use by the proxy simulation.

The deforming geometries rendered using the hybrid proxy can be generated with a range of methods. The surfaces can come from techniques such as: physical simulation [Allard et al. 2007]; sensing [Rydén and Chizeck 2013]; or by algorithmic generation [Le and Deng 2014]. The only requirement of the method proposed here is that the geometries are represented in a form that can be efficiently range queried using an appropriate method. For the experiments in section 3, the meshes were generated algorithmically.

2.4.2 Haptic Force Computation and Rendering Thread

The haptic force computation and rendering thread takes the proxy dynamics (computed by the proxy simulation) and the current tool dynamics and computes the haptic force/torque, $\mathbf{f}_h \in \mathbb{R}^6$. This

force/torque command is then set in the haptic device's low-level controller and the current tool pose is written to shared memory.

The force computation can be performed in a number of different ways. For the following experiments, a simple visco-elastic virtual linkage was implemented, as described in equation (14).

$$\mathbf{f}_h = -K_{vl}\mathbf{x}_{vl} - B_{vl}\dot{\mathbf{x}}_{vl}, \tag{14}$$

where $B_{vl} \in \mathbb{R}^{6\times6}$ is the diagonal damping matrix for the virtual linkage, $\dot{\mathbf{x}}_{vl} \in \mathbb{R}^6$ is the first derivative of $\mathbf{x}_{vl}$ and the remaining symbols are as defined previously.

Similarly, the constraint-based force computation from [Ortega et al. 2007], or other linear and non-linear haptic controllers [Bowyer and Rodriguez y Baena 2015] could be used here without requiring modification to the method.

2.4.3 Hybrid Proxy Simulation Thread

The hybrid proxy simulation thread takes the object geometry and tool pose and simulates a time-step of the proxy dynamics. This thread implements the method described in section 2. The overall procedural flow of the simulation is illustrated inside the 'hybrid proxy simulation' block of figure 4.

The range querying at the beginning of the proxy simulation can be performed using many techniques described in the literature [Hoff et al. 2002; Je et al. 2012], dependent on aspects such as the format of the object's geometry and the type of deformation. For the experimental validation in the following section, a Deformation Invariant Bounding Sphere (DIBS) hierarchy [Bowyer and Rodriguez y Baena 2014] was used. This is a bounding volume hierarchy that is optimised for collision detection, proximity querying and range querying between fast-deforming mesh objects with fixed topologies and rigid tool geometries. These are common requirements in the haptic exploration of simulated or algorithmically generated geometries and therefore this method is appropriate here. In the case of small object penetrations, the penetration magnitude and direction was computed by projecting radially from the haptic tool interacting with the object.

3 Experimental Evaluation and Results

Three dynamic objects were haptically rendered with the hybrid proxy method. The method was implemented in C++, as described in the previous section, and visual feedback was provided using VTK (Kitware, Inc.). A Force Dimension sigma.7 (six degree-of-freedom) haptic device was used to control the tool pose within the simulation. In all tests, a cylinder was used to define the haptic probe that was manipulated by the user.

To asses the proposed method, the following metrics were recorded for each experiment:

- *Proxy method* - the type of proxy (constraint/penalty) in use at any instant
- *Number of close contacts* - the number of triangle-tool contacts considered in the solver
- *Computation time* - for each range query, proxy simulation and in total
- *Proxy-object separation* - the distance between the proxy and object, such that positive values identify a separation and negative values identify a penetration
- *Proxy-tool separation* - the distance (translational and rotational magnitudes) between the proxy and actual tool

The first experiment involved rendering a beating heart [Kwok et al. 2010]. The tool was forced to intersect the heart so that the proxy slid on the pulsating surface. The proxy moved in and out with the beating of the heart at 30 Hz whilst moving down. Figure 5 shows the resulting metrics for this experiment. As the proxy only rested on the outer surface of the heart, it never became trapped, and therefore the constraint-based method was always used.

The second experiment involved rendering a block with a hole in the middle that periodically deformed, figure 6. The hole in the block was initially large enough for the proxy to fit inside; however, at the peak of the cycle the hole became too small for the proxy and clamped it in place. This tested for the method's ability to swap between modes.

The third experiment was similar to the second, but simulated a human hand gripping the proxy, figures 1 and 7. This test involved a much more complicated contact profile. The human hand was generated in MakeHuman and animated in VTK. The range of motion of the grip was sufficient that it completely enclosed the proxy.

4 Discussion

The results for experiment 1 show that the dynamic formulation of Guass' principle of least constraint based on 'close contacts' is valid. The motion profile shows that the proxy moves in time with the beating heart. No penetrations were recorded between the proxy and the heart throughout the trial, even though the haptic device tried to pull the proxy inside. This shows that the use of positive constraints on acceleration in equation (7) are functional and they do not suffer from issues due to the linear approximation. Qualitatively, the movement of the proxy along the surface of the heart was smooth and produced good tactile information regarding the subtle detail on the surface. The total computation time for each step in the simulation was generally in the range of 0.1-0.2 ms; of which, the constraint-based simulation was a small component. This shows that the method, even with over 45 contacts under consideration, still runs sufficiently fast for effective haptic rendering. As the proxy and object came into contact (time = 0.8 s, proxy-object separation = 0 mm), the proxy-tool separation started to increase as the tool penetrated while the proxy remained on the surface. As with previous constraint-based proxies, the way in which the proxy moves in translation and rotation is defined by the relative stiffnesses of the translational and rotational spring constants in the virtual linkage.

Experiment 2 shows that the transition between proxy modes was effective at preventing numerical instabilities. It can be seen from the results that until approximately 4.2 s, the constraint-based proxy is being used and the proxy follows the edge of the hole with no penetration. After this time, the hole completely encloses the haptic probe and there are no valid constraint-based solutions. Subsequently, the method switches to using the penalty-based simulation. The proxy-object separation plot shows that, after switching mode, the proxy penetrates the hole by up to 0.6 mm. This is to be expected as the probe's diameter is larger than the hole and this penetration avoids erratic unstable movement that would occur for the constraint-based method in this case. During the time in which the probe is clamped, the number of contacts increases to over 500, due to the very high resolution triangular mesh representing the inside of the hole. Though the proximity query does rise to almost 1 ms, the proxy simulation time remains below 0.1 ms. As the block expands again in the second half of the experiment, the probe is released and the method switches back to using the constraint-based solver and there are no longer any penetrations.

In the final experiment, the results show the robustness and stability of the penalty based method. The proxy-object separation plot illus-

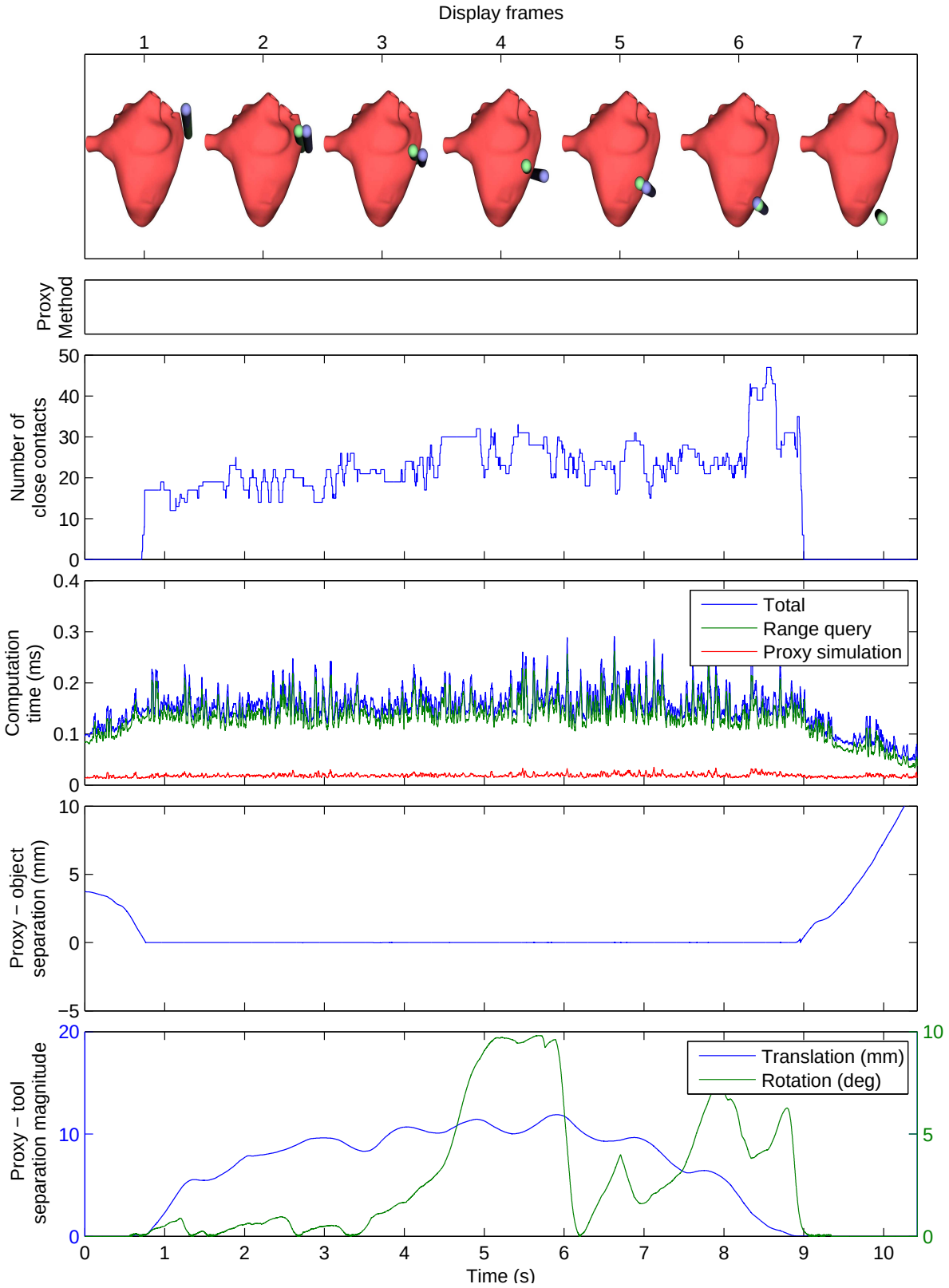


Figure 5: *Plot and visualisation of experiment 1. Visualisation shows a user interacting with the beating heart and the plots show the synchronised metric values. The proxy is shown as the blue cylinder and the actual tool is shown as the green cylinder. The proxy method bar is white for constraint-based and black for penalty-based.*

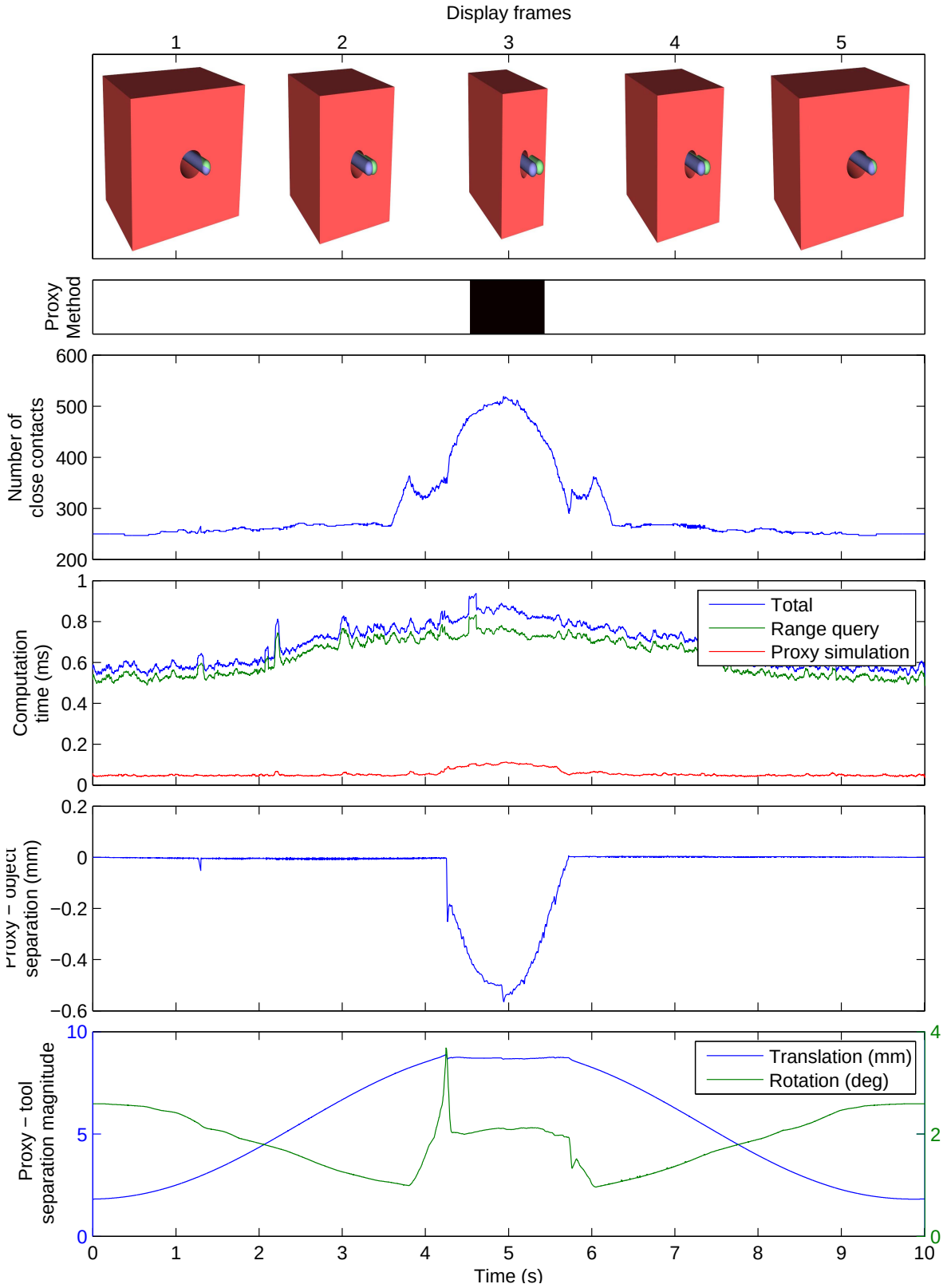


Figure 6: *Plot and visualisation of experiment 2. Visualisation shows a user interacting with the deforming hole and the plots show the synchronised metric values. The proxy is shown as the blue cylinder and the actual tool is shown as the green cylinder. The proxy method bar is white for constraint-based and black for penalty-based.*

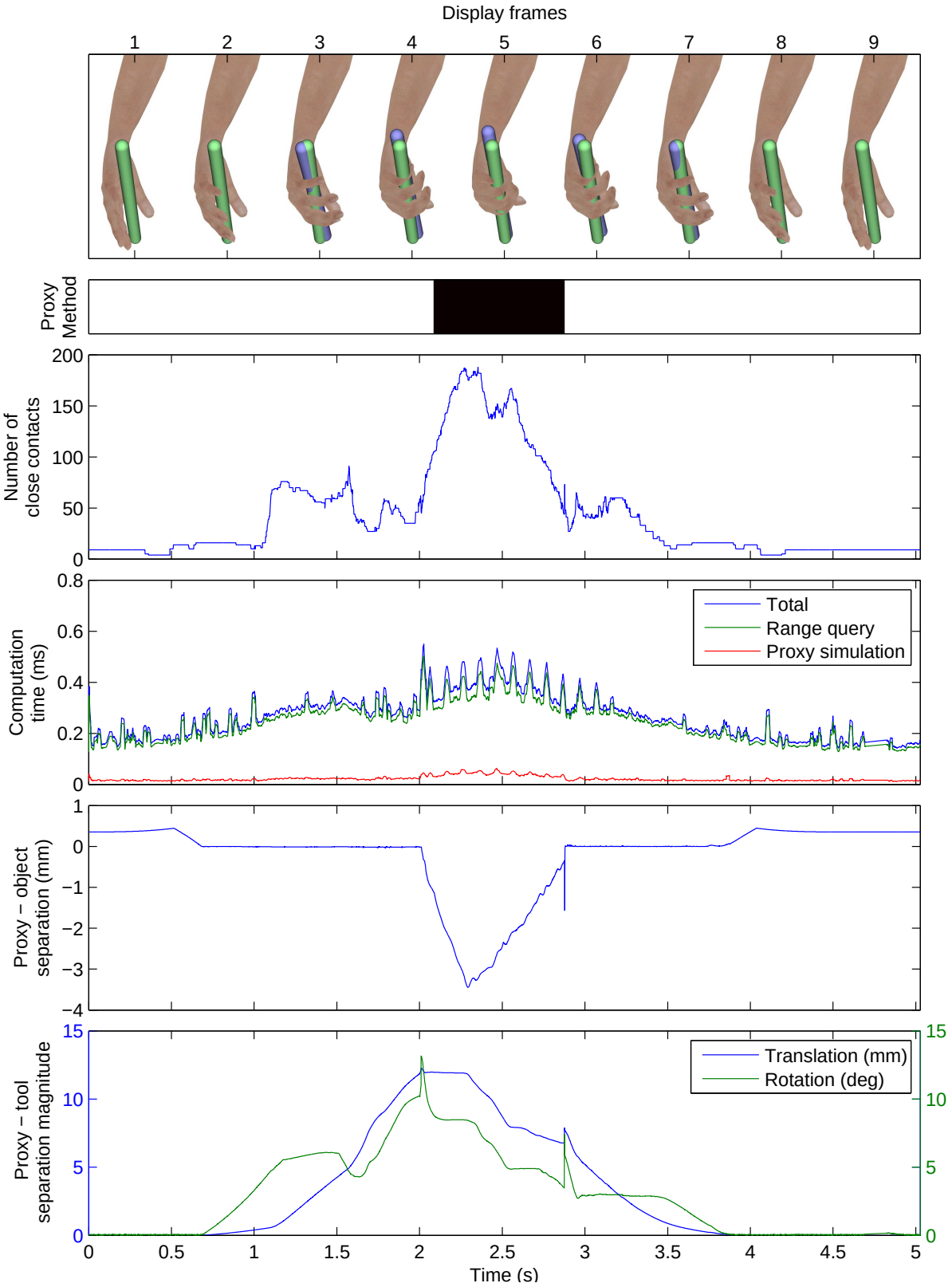


Figure 7: *Plot and visualisation of experiment 3. Visualisation shows a user interacting with the simulated gripping hand and the plots show the synchronised metric values. The proxy is shown as the blue cylinder and the actual tool is shown as the green cylinder. The proxy method bar is white for constraint-based and black for penalty-based.*

trates the gripping process. At the start, the proxy is separate from the hand and perfectly aligned with the tool. At 0.5 s, the gripping hand starts to touch the proxy and pulls it along. This causes the proxy-object separation to become zero and the proxy-tool separation to increase. At 2 s, the gripping hand fully encloses the probe and there are no stable numerical solutions to the constraint-based simulation. As a result, the method switches to the penalty-based method and the proxy starts to penetrate the object to relive the clamping. As the hand releases, the proxy first switches back to being constraint-based at 2.8 s and then is completely released from the hand at 4 s.

One clearly noticeable feature of the results for experiment 2 and 3 is that there are sharp peaks in the separation plots when the proxy method swaps between constraint and penalty. In the implementation described here, this is minimised by the very nature of the methods both being dynamic based and having a simulated mass and inertia for the proxy. i.e. though one expects an instantaneous change in the effective force applied to the proxy, this would lead to an instantaneous change in acceleration, but a more gradual change in position. From these plots it is clear that there is still some degree of disturbance during the switching, and more effective methods for overcoming this will be investigated in the future.

5 Conclusions

This paper presents an proof of functionality for a novel hybrid proxy concept that has several benefits for use in haptically rendering dynamic objects.

The dynamic proxy formulation described here allows for the computation of a target haptic pose which follows that of the haptic device, while not penetrating the object geometries in normal motion. The method adapts the conventional Gauss' principle method so that it can operate with deforming geometries by optimising the motion based on a set of 'close contacts' between the tool and object rather than actual contacts. The wider range of contacts considered by the method allows it to operate without the requirement for continuous collision detection and also allows it to recover intersecting geometries which can occur as the result of object deformation.

As continuous collision detection is computationally demanding, particularly with a deforming mesh, the removal of this requirement for the proxy could have a significant implication for the resolution of the deforming geometries which can be rendered. The control rate required for a smooth and stable haptic interaction is approximately 1 kHz. The computational demands of the proxy dynamic simulation therefore limits the size, and hence resolution, of the geometries that can be rendered.

By allowing the proxy dynamics to switch between the rigid and accurate constraint-based dynamics and the flexible and robust penalty-based dynamics, this method achieves good performance during normal exploratory interactions while avoiding numerical instabilities when the proxy becomes enclosed by the objects being rendered.

5.1 Limitations

The main limitation of this method is the assumed linearity of the proximity Jacobian. This assumption is negligible when the controller frequency is sufficiently high relative to the movement velocity; however, if the maximum velocity of the object's deformation, or the movement of the user, is extremely high or even discontinuous, then this approximation could break down. The effect of this is that the points that are almost in contact will have their motion dominated by non-linear rotational components which cannot be considered at present. However, in the three experiments performed, even with comparatively fast surface movements, the motion of the proxy was smooth and avoided all penetration.

5.2 Future Work

This paper has demonstrated the feasibility of controlling a proxy in a hybrid way and using discrete range queries rather than continuous collision detection. Future work will involve further numerical analysis of the stability of the constraint-based methods so that the switching logic can be refined and more thorough user trials will be undetaken to assess the quality of the haptic feedback. An interesting route to pursue would be considering components of the proxy's motion in different (orthogonal) directions independently of one another. This would allow for constraint-based methods in the un-gripped dimensions while the penalty-based methods could be used in dimensions that are, simultaneously. This is expected to further improve the fidelity of the haptic feedback.

Acknowledgements

This research was supported by EU Grant FP7-ICT-2009-6-270460.

References

ALLARD, J., COTIN, S., FAURE, F., BENSOUSSAN, P.-J., POYER, F., DURIEZ, C., DELINGETTE, H., AND GRISONI, L. 2007. SOFA - an open source framework for medical simulation. In *Medicine Meets Virtual Reality*.

BARAFF, D. 1989. Analytical methods for dynamic simulation of non-penetrating rigid bodies. In *Proc. Annual Conf. Comp. Graphic. & Interactive. Tech. (SIGGRAPH)*, 223–232.

BARBIC, J., AND JAMES, D. 2008. Six-dof haptic rendering of contact between geometrically complex reduced deformable models. *IEEE Trans. Haptics 1*, 1 (Jan), 39–52.

BOWYER, S. A., AND RODRIGUEZ Y BAENA, F. 2014. Deformation invariant bounding spheres for dynamic active constraints in surgery. *Proc. Inst. of Mech. Eng., Part H: J. Eng. in Med. 228*, 4 (Apr.), 350–361.

BOWYER, S. A., AND RODRIGUEZ Y BAENA, F. 2015. Dissipative control for physical human-robot interaction. *IEEE Trans. Robot. 31*, 6 (Dec.), 1281–1293.

BOYD, S., AND VANDENBERGHE, L. 2004. *Convex Optimization*. Cambridge University Press.

BRUYNINCKX, H. 2001. Open robot control software: the OROCOS project. In *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 3, 2523–2528.

CONSTANTINESCU, D., SALCUDEAN, S., AND CROFT, E. 2004. Haptic rendering of rigid body collisions. In *International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 2–8.

CONSTANTINESCU, D., SALCUDEAN, S. E., AND CROFT, E. A. 2005. Haptic rendering of rigid contacts using impulsive and penalty forces. *IEEE Trans. Robot. 21*, 3 (June), 309–323.

ERICSON, C. 2004. *Real-Time Collision Detection*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

HOFF, K., ZAFERAKIS, A., LIN, M., AND MANOCHA, D. 2002. Fast 3d geometric proximity queries between rigid and deformable models using graphics hardware acceleration. Tech.

Rep. TR02-004, Department of Computer Science, University of North Carolina.

JE, C., TANG, M., LEE, Y., LEE, M., AND KIM, Y. J. 2012. PolyDepth: Real-time penetration depth computation using iterative contact-space projection. *ACM Trans. on Graph. 31*, 1, 5.

KUHN, H. W. 1956. Solvability and consistency for linear equations and inequalities. *The American Mathematical Monthly 63*, 4 (Apr.), 217–232.

KWOK, K.-W., VITIELLO, V., AND YANG, G.-Z. 2010. Control of an articulated snake robot under dynamic active constraints. In *Proc. Int. Conf. Med. Img. Comp. & Comp. Asst. Int.*, vol. 6363 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 229–236.

LE, B. H., AND DENG, Z. 2014. Robust and accurate skeletal rigging from mesh sequences. *ACM Trans. Graph. 33*, 4 (July), 84:1–84:10.

MCNEELY, W. A., PUTERBAUGH, K. D., AND TROY, J. J. 1999. Six degree-of-freedom haptic rendering using voxel samping. In *Proc. Annual Conf. Comp. Graphic. & Interactive. Tech. (SIGGRAPH)*.

MIRTICH, B., AND CANNY, J. 1995. Impulse-based simulation of rigid bodies. In *Symposium on Interactive 3D graphics*, 181–217.

ORTEGA, M., REDON, S., AND COQUILLART, S. 2007. A six degree-of-freedom god-object method for haptic display of rigid bodies with surface properties. *IEEE Trans. Vis. Comput. Graphics 13*, 3 (May), 458–469.

OTADUY, M., AND LIN, M. 2006. A modular haptic rendering algorithm for stable and transparent 6-dof manipulation. *IEEE Trans. Robot. 22*, 4 (Aug.), 751–762.

REDON, S., KHEDDAR, A., AND COQUILLART, S. 2002. Gauss' least constraints principle and rigid body simulations. In *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 1, 517–522.

RUSPINI, D. C., KOLAROV, K., AND KHATIB, O. 1997. The haptic display of complex graphical environments. In *24th Annual Conf. Comp. Graph. & Inter. Tech.*, ACM Press/Addison-Wesley Publishing Co., 345–352.

RYDÉN, F., AND CHIZECK, H. 2013. A method for constraint-based six degree-of-freedom haptic interaction with streaming point clouds. In *Proc. IEEE Int. Conf. Robot. Autom.*, 2353–2359.

SALISBURY, K., AND TARR, C. 1997. Haptic rendering of surfaces defined by implicit functions. In *Annual Symposium for Haptic Interfaces for Virtual Environment and Teleoperator Systems*, ASME, 61–67.

WANG, D., ZHANG, X., ZHANG, Y., AND XIAO, J. 2011. Configuration-based optimization for six degree-of-freedom haptic rendering for fine manipulation. In *Proc. IEEE Int. Conf. Robot. Autom.*, 906–912.

WILHELMSEN, D. R. 1976. A nearest point algorithm for convex polyhedral cones and applications to positive linear approximations. *Mathematics of Computation 30*, 133 (Jan.), 48–57.

ZILLES, C. B., AND SALISBURY, J. K. 1995. A constraint-based god-object method for haptic display. In *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, vol. 3, 146–151.