

Chapter 2

Configuration-based Optimization Approach

To tackle computational challenges in 6-DoF haptic rendering of fine manipulation, in this book we present a novel constraint-based approach: the configuration-based optimization approach. In this chapter, we introduce the basics of the configuration-based optimization approach to 6-DoF haptic rendering of rigid body in contact. First, we provide an overview of the approach in Sect. 2.1, and then we introduce each component in the computational pipeline in Sects. 2.2–2.4, through which we explain the differences between this approach and other haptic rendering approaches. In Section 2.5, we introduce experimental results that validate the approach.

2.1 Overview of the Approach

We first introduce the problem and the motivation of the approach. Next, we outline the framework of the approach with the basic components in a flowchart. We conclude this section by summarizing the main contributions of the approach.

2.1.1 Problem Formulation

We introduce the following terms to facilitate description:

- *Haptic tool*: It is the avatar in a virtual environment of joystick handle of the haptic device held by a user in the physical world; it reflects the exact location of the joystick handle as the user moves the handle and can penetrate the surface of a rigid object in the virtual world. In 3-DoF haptic rendering, this tool has been named as the haptic interface point (HIP) (Ho et al. 1999), or haptic handle (Lin and Otaduy 2008).

Electronic supplementary material The online version of this article (doi:[10.1007/978-3-662-44949-3_2](https://doi.org/10.1007/978-3-662-44949-3_2)) contains supplementary material, which is available to authorized users.

- *Graphic tool*: It is the graphic display of the haptic tool that forms a contact with an object but does not penetrate the object to simulate the physical reality of contact. This term is also called the god-object (Zilles and Salisbury 1995), virtual proxy (Ruspini et al. 1997), surface contact point (SCP) (Ho et al. 1999), or virtual tool (Ortega et al. 2007).
- *Collision response*: The process is to compute the pose of the graphic tool in contact and to simulate the contact force and torque (Lin and Otaduy 2008).

The motivation of the configuration-based optimization approach is to tackle the problem of computing the pose of the graphic tool when the haptic tool penetrates into a virtual object so that its pose cannot be directly used as the pose of the graphic tool, as the latter simulates contact with the virtual object. Intuitively, the relation between the haptic tool and the corresponding graphic tool can be illustrated by the magnet metaphor shown in Fig. 2.1, where the haptic tool inside the virtual object can be viewed as a magnet, the object's surface can be viewed as a metal container, and the graphic tool can be viewed as attracted by and following the haptic tool as close as possible but cannot enter the container (i.e., cannot penetrate through the object). If the haptic tool is outside the virtual object and in the free space, the graphic tool collocates with the haptic tool.

The configuration-based optimization approach takes as input the configuration of the haptic tool and computes the corresponding configuration of the graphic tool as the result of optimizing the configuration of the graphic tool directly rather than optimizing the acceleration or velocity (Duriez et al. 2006; Ortega et al. 2007) to avoid inaccuracies in the configuration of the graphic tool due to numerical integration.

In addition, the configuration-based optimization approach allows us to compute the tool configuration without incorporating typical mesh to mesh continuous collision detection [such as FAST or C2A (Redon 2004; Zhang et al. 2006; Tang et al. 2009)] and thus avoid the associated computational burden. For example, in (Ortega et al. 2007), the final configuration of the graphic tool is computed by continuous collision detection (CCD) instead of optimization. In each simulation

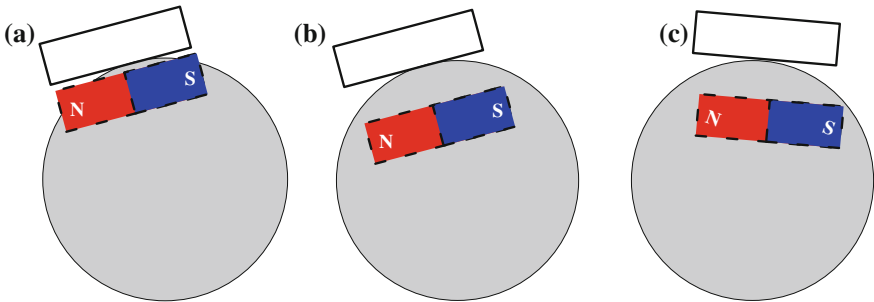


Fig. 2.1 The magnet metaphor. **a** Haptic tool entering the virtual object. **b** Graphic tool following the translating haptic tool. **c** Graphic tool following the rotating haptic tool

cycle, an acceleration-based optimization and a subsequent integration are executed before the CCD, because the result of integration cannot maintain non-penetration between the moving graphic tool and the other objects. However, the configuration-based optimization approach can provide a solution directly satisfying the non-penetration constraints without requiring a post-optimization CCD module.

For a tool and objects modeled in polygonal meshes, the configuration-based optimization approach produces high-fidelity rendering (Wang et al. 2011), but efficient operations are limited to small-region contacts and a tool of simple shape. In order to treat more complex contact scenarios efficiently, we consider the tool and objects modeled in sphere trees. Sphere trees facilitate a uniform expression of constraints for complex contacts involving both convex and concave geometric features, which simplifies configuration-based optimization. Detailed analysis about the reason for adopting sphere trees for constraint modeling will be explained in Sect. 2.2.1.

Since we focus on haptic simulation of multi-region contacts with frequent contact switches, we do not assume very large and fast movements of the haptic tool when the haptic tool intersects with an object, that is, we assume quasi-static movements. This assumption is reasonable for related applications, such as simulating dental operations, where a dentist moves the dental tool carefully and slowly in a patient's mouth.

2.1.2 Framework

The input and output of the optimization problem are:

- Given: (1) the 6D configuration of the haptic tool at the previous and current time step ($\mathbf{q}_h^{t-1}$, $\mathbf{q}_h^t$), and (2) the 6D configuration of the object (e.g., a tooth), the geometric model and the physical property (e.g., stiffness and friction coefficient) of the object.
- Compute: (1) The 6D configuration of the graphic tool at the current time step ($\mathbf{q}_g^t$) while ensuring no penetration between the graphic tool and the object, (2) the 6D feedback force and torque $\mathbf{F}'$.

We use the following configuration-based optimization model (As shown in Fig. 2.2)

$$\begin{cases} \text{Min:} & f(\mathbf{q}_g^t, \mathbf{q}_h^t) \\ \text{Subject to:} & V_O \cap V_T = \emptyset \end{cases} \quad (2.1)$$

where

$$\begin{cases} \mathbf{q}_h = [x_h, y_h, z_h, \gamma_h, \beta_h, \alpha_h]^T \\ \mathbf{q}_g = [x_g, y_g, z_g, \gamma_g, \beta_g, \alpha_g]^T \end{cases} \quad (2.2)$$

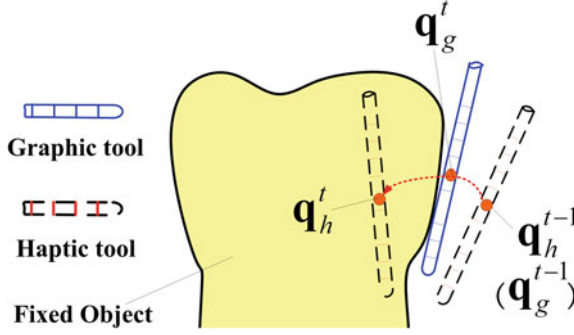


Fig. 2.2 Configuration-based optimization model. Copyright © IEEE. All rights reserved. Reprinted, with permission, from Wang et al. (2013)

In Eq. (2.1), V_O denotes the volume of the object and V_T denotes the volume of the graphic tool. The constraint is to maintain exact contact between the graphic tool and the object, i.e., there is no perceptible penetration and no separation. The key challenge in this optimization model is how to formulate the constraint in Eq. (2.1), which will be discussed in Sect. 2.2.

In Eq. (2.1), the objective function $f(\bullet)$ is a distance metric to describe the difference between configurations of the haptic tool and the graphic tool. In related literature (Zhang 2009), different kinds of distance metrics were introduced, but the time cost for computing each of those metrics is far greater than 1 ms, which does not satisfy the update rate of 1 kHz, as required by haptic rendering, called *haptic rate*.

In order to make sure that the graphic tool follows the movement of the haptic tool without unnatural movement as the haptic tool translates and rotates, we imagine that a six-dimensional spring connects the graphic tool to the haptic tool and define the following deformation energy of the spring as the objective function to minimize:

$$f(q_g^t) = \sum_{j=1}^3 \frac{1}{2} k_t (q_{gj}^t - q_{hj}^t)^2 + \sum_{j=4}^6 \frac{1}{2} k_r (q_{gj}^t - q_{hj}^t)^2 \quad (2.3)$$

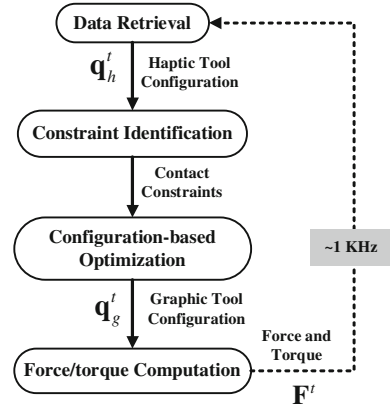
where k_t and k_r are translational and rotational stiffness, respectively. For implementation of the proposed model, these two values are chosen as $k_t = 1$ N/mm and $k_r = 1,000$ mNm/rad to fully exploit the ability of the haptic device used.

The above objective function can be further represented in the following matrix form:

$$f(\mathbf{q}_g^t, \mathbf{q}_h^t) = \frac{1}{2} (\mathbf{q}_g^t - \mathbf{q}_h^t)^T \mathbf{G} (\mathbf{q}_g^t - \mathbf{q}_h^t) \quad (2.4)$$

where $\mathbf{G}$ is a diagonal 6×6 stiffness matrix. $\mathbf{q}_h^t = (x_h^t, y_h^t, z_h^t, \gamma_h^t, \beta_h^t, \alpha_h^t)$, and $\mathbf{q}_g^t = (x_g^t, y_g^t, z_g^t, \gamma_g^t, \beta_g^t, \alpha_g^t)$.

Fig. 2.3 Flowchart of the configuration-based optimization approach to 6-DoF haptic rendering. Copyright © IEEE. All rights reserved. Reprinted, with permission, from Wang et al. (2013)



Thus, the goal of the optimization is to obtain a contact configuration of the graphic tool that minimizes the deformation energy of the spring connecting it and the haptic tool.

Different parameter values in the matrix $\mathbf{G}$ will lead to different configurations for the graphic tool. This could be implemented intentionally to simulate different effects in non-realistic simulation scenarios such as computer games.

The flowchart of the configuration-based optimization approach to 6-DoF haptic rendering is shown in Fig. 2.3.

After computing the configuration of the graphic tool, the 6D feedback force and torque can be derived using a function that reflects the change of the relative configuration between the graphic tool and the haptic tool at current time step

$$\mathbf{F}^t = F(\mathbf{q}_g^t, \mathbf{q}_h^t) \quad (2.5)$$

where detailed functions of the above model depend on the physical properties of the tool and the objects.

In order for the configuration-based optimization approach to work effectively and efficiently, the following issues need to be resolved:

1. How to represent the tool and the object with complex shapes? The geometric representation should support diverse shapes, fast collision detection of possible contacts and penetrations, and modeling versatile contact scenarios, especially when the free moving space of the tool is a non-convex space. This issue will be elaborated in Sect. 2.2.
2. How to model, construct, and identify the non-penetration contact constraints between the graphic tool and the objects? Both single-region and multi-region contacts need to be modeled. The collision detection algorithm should be efficient. This issue will be elaborated in Sect. 2.2.
3. How to solve the optimization problem robustly and efficiently? Since contacts always occur at the surface of objects, while the constraints formula is expressed

in terms of configuration variables. Efficient transformation between the Cartesian space description of contact constraints and the configuration space of the graphic tool is needed to ensure fast collision response of 1 kHz. This issue will be elaborated in Sect. 2.3.

4. How to compute the feedback force and torque and ensure stability? This issue will be elaborated in Sect. 2.4.

2.2 Object and Contact Modeling Using Sphere Trees

Object and contact modeling is the fundamental step to realize the framework shown in Fig. 2.3. How to represent or model the geometry of objects in a virtual environment directly affects the computation performance for collision detection and contact constraint identification. An object is most commonly represented as a polygonal mesh, where the boundary of the object is approximated by connected triangles. A hierarchy of simpler bounding volumes, such as rectangles or spheres, is often used to speed up collision detection, with the polygonal mesh model at the bottom of the hierarchy (such as the leaf level of a tree structure). An object can also be represented as a sphere tree, where the root sphere encloses the entire object, and its children are smaller spheres enclosing parts of the object, and so on, so that at the leaf level, the spheres are the smallest in the greatest number to provide more accurate approximation of an object. Sphere trees provide convenience in representing complex contact constraints involving multiple contact regions that polygonal meshes lack. In the following, we first describe in detail the difficulty of formulating contact constraints between objects in polygonal mesh models and then present contact constraint formulation between objects in sphere-tree models. Finally, we describe how a sphere tree can be constructed.

2.2.1 Difficulties of Using Polygonal Meshes to Model Objects

Depending on if the space that the tool can freely move around, i.e., the free space, is convex or non-convex, contact constraints have to be formulated differently.

2.2.1.1 Convex Free Space

For the interaction between a convex-shaped tool and arbitrary-shaped objects, we need to identify different pairs of contact primitives between the graphic tool and the object. If mesh representations are used for both the tool and the object, the contact type in each contact region can be described by pairs of contacting geometric primitives, i.e., vertex, edge, and triangle face on each mesh.

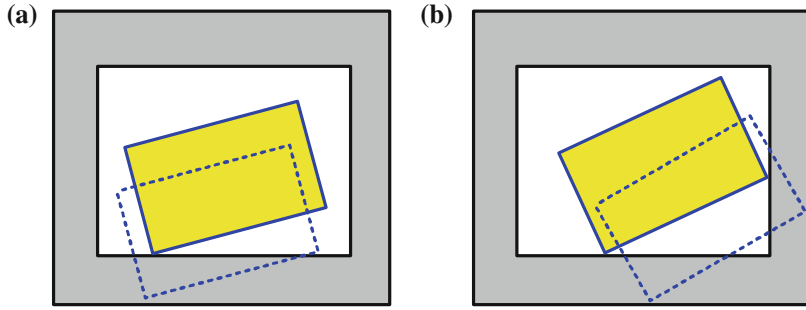


Fig. 2.4 Free space of the tool is a convex space. **a** Single-region contact. **b** Multi-region contacts

For the scenario in Fig. 2.4, a rectangular shaft in blue is moving within the interior volume of a rectangular hole, while the free space of the tool is a convex space. The dashed shaft represents the haptic tool, and the solid shaft represents the graphic tool. Denote a vertex of the tool as p_i , $i = 1, \dots, 4$, and an unilateral surface constraint of the hole as F_i . A set of simultaneous point-face non-penetration constraints can be used for Eq. (2.1) in general, whether the contact is of a single region or multiple regions.

For the vertex-face contact type (V-F type), the constraint in Eq. (2.1) can be expressed as

$$g[p] \geq 0 \quad (2.6)$$

where $g[\cdot]$ represents a unilateral constraint formed by each triangle on the surface of the object. p refers to a boundary point or a vertex on the surface of the graphic tool. For each triangle on the surface of the object, we define a *constraint plane*

$$Ax + By + Cz + D = 0 \quad (2.7)$$

The normal of the plane is set as the normal of the triangle. Thus, we have

$$g[p] = \frac{(Ax + By + Cz + D)}{\sqrt{A^2 + B^2 + C^2}} \quad (2.8)$$

which is defined as the signed distance from the point $p(x, y, z)$ to the constraint plane.

We can express the constraint inequality for multiple point-plane constraints as follows:

$$g_i[p_j] \geq 0 \quad i, j = 1, \dots, 4 \quad (2.9)$$

where g_i denotes unilateral constraints formed by the four planes (F_i) of the hole as shown in Fig. 2.5.

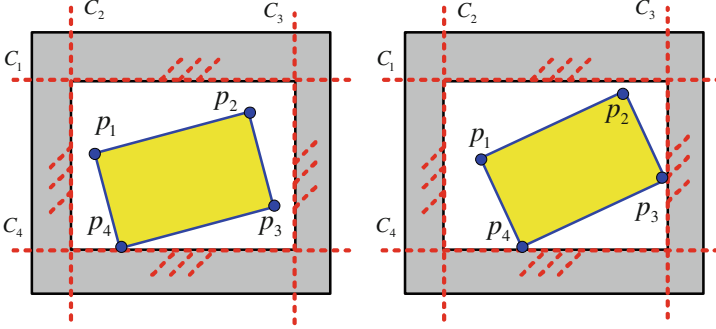


Fig. 2.5 Joint constraints functions (point-face constraints) for convex free space

These point-plane constraint inequalities consistently form an AND compound constraint, which defines the convex free space for the movement of the graphic tool.

2.2.1.2 Non-convex Free Space

For the scenarios shown in Fig. 2.6, the free space of the graphic tool is non-convex. In such a case, the related point-plane constraints do not consistently form an AND or an OR compound constraint. Figure 2.7 illustrates the constraints as forming an AND compound constraint and Fig. 2.8 illustrates the constraints as forming an OR compound constraint, but neither is correct for expressing the entire non-convex free space shown in Fig. 2.6. Thus, it is difficult to use a mesh model or a volume model to express the non-penetration constraints if the free space of the graphic tool is non-convex.

For a non-convex-shaped tool and an arbitrary object, such as shown in Fig. 2.9, the contact types are more diverse. It is even more difficult to use mesh-based object models to express the non-penetration constraints.

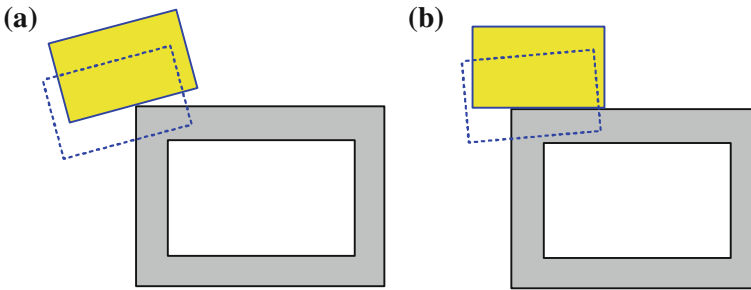


Fig. 2.6 Free space of the tool is a non-convex space. **a** Single-point contact. **b** Multi-point contacts

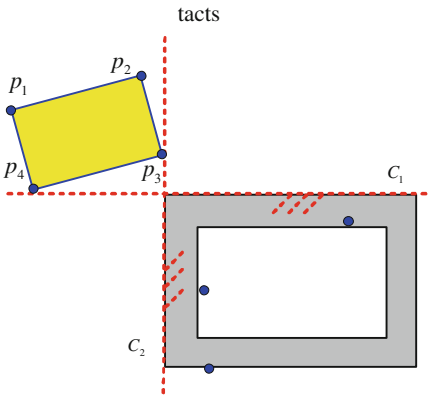


Fig. 2.7 Point-plane constraints form an AND compound constraint

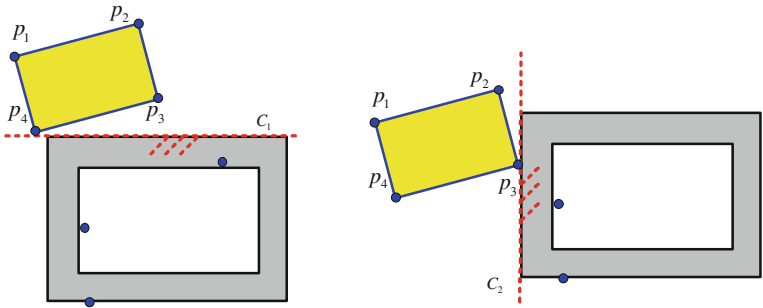


Fig. 2.8 Point-plane constraints form an OR compound constraint

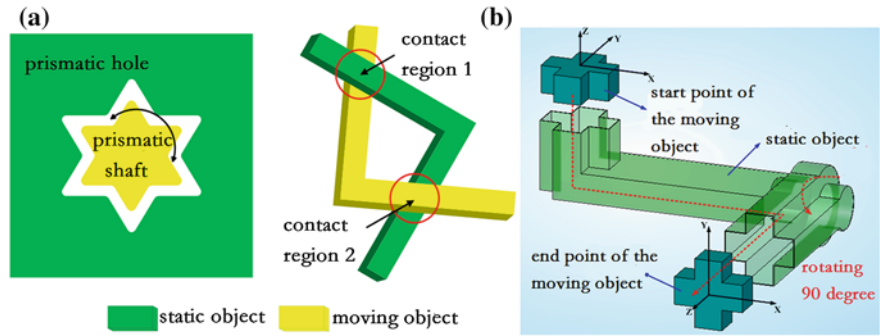


Fig. 2.9 Two interaction cases between two L-shaped objects: **a** edge-edge constraint; **b** combination constraints: edge-edge and point-face. Note the red-shaded region is the key challenge for constraint modeling

2.2.2 Contact Modeling Based on Sphere Trees

If a tool and an object are represented in sphere trees, no penetration between them means that there is no penetration between any sphere $s_T = (x_T, y_T, z_T, r_T)$ from the graphic tool's sphere tree and any sphere $s_O = (x_O, y_O, z_O, r_O)$ from the object's sphere tree, where (x, y, z, r) are the center and radius of a sphere in world coordinate system.

The non-penetration constraint can be written as follows:

$$(x_T - x_O)^2 + (y_T - y_O)^2 + (z_T - z_O)^2 \geq (r_T + r_O)^2 \quad (2.10)$$

Clearly, to model the non-penetration constraints between two objects, the sphere-tree representation provides a simple and uniform expression.

Since s_O is fixed in the virtual environment, and only s_T moves as the graphic tool moves, we can write (2.10) in a function form as follows:

$$C(x_T, y_T, z_T) \geq 0 \quad (2.11)$$

where x_T , y_T , and z_T are the functions of the graphic tool configuration $\mathbf{q}_g^t$, which are non-linear. We can establish their mapping using the following coordinate transformation.

First, we use L to represent the local coordinate system of the graphic tool, and use W to represent the world coordinate system. Then, we can construct the relationship between L and W as follows:

$$\mathbf{x}_W = \mathbf{T}_L^W \cdot \mathbf{x}_L, \quad (2.12)$$

where $\mathbf{x}_W$ and $\mathbf{x}_L$ represent the homogeneous coordinates of a point in W and L , and this point can be any sphere center of the graphic tool. And

$$\mathbf{T}_L^W = \begin{bmatrix} \mathbf{R}_L^W & \mathbf{p}_L^W \\ 0 & 1 \end{bmatrix}, \quad (2.13)$$

where

$$\mathbf{R}_L^W = \begin{bmatrix} c\alpha_g^t c\beta_g^t & c\alpha_g^t s\beta_g^t s\gamma_g^t - s\alpha_g^t c\gamma_g^t & c\alpha_g^t s\beta_g^t c\gamma_g^t + s\alpha_g^t s\gamma_g^t \\ s\alpha_g^t c\beta_g^t & ps\alpha_g^t s\beta_g^t s\gamma_g^t + c\alpha_g^t c\gamma_g^t & s\alpha_g^t s\beta_g^t c\gamma_g^t - c\alpha_g^t s\gamma_g^t \\ -s\beta_g^t & c\beta_g^t s\gamma_g^t & c\beta_g^t c\gamma_g^t \end{bmatrix} \quad (2.14)$$

and

$$\mathbf{p}_L^W = \begin{bmatrix} x_g^t & y_g^t & z_g^t \end{bmatrix}^T, \quad (2.15)$$

where $\mathbf{q}_g^t = [x_g^t, y_g^t, z_g^t, \gamma_g^t, \beta_g^t, \alpha_g^t]^T$, and the orientation angle α_g^t , β_g^t , and γ_g^t refers to the rotation angle around axis z , axis y , and axis x of the world coordinate system.

Take Eq. (2.12) into (2.11), we can construct non-linear constraint inequalities described in terms of the configuration variable of the graphic tool, i.e., the constraints in C-space can be expressed as follows:

$$C_i\left(x_T\left(q_g^t\right), y_T\left(q_g^t\right), z_T\left(q_g^t\right)\right) \geq 0 \quad i = 1, \dots, N, \quad (2.16)$$

where N refers to the number of contact sphere pairs.

To speed up the optimization, we linearize the above constraint using Taylor expansion. We will explain this linear model in Sect. 2.4.

2.2.3 Construction of Sphere Trees

Different sphere-tree models have been used in some existing penalty-based methods for efficient collision detection. Ruffaldi et al. proposed an implicit sphere tree to compute collision detection (Ruffaldi et al. 2006). They also introduced an adapted impulse-based method to solve the collision response. Weller et al. proposed a novel data structure, called inner sphere tree (IST), based on which they could easily handle distance-query and compute penetration volume (Weller and Zachmann 2009). Furthermore, they proposed a penalty-based collision response scheme which exploited penetration volumes to provide continuous force and torque.

Hubbard introduced a medial-axis-based sphere-tree model (Hubbard 1996), which is one of the best sphere-tree approaches to approximate an object with spheres. The key element of his algorithm is to find the skeleton of an object using a Voronoi diagram and create spheres from the skeleton to give a tight fit to objects. Bradshaw and O'Sullivan (Bradshaw and Sullivan 2004) have proposed several other algorithms (Merge, Burst, Expand, and Spawn) based on Hubbard's model and have extended all these algorithms in an adaptive manner. Compared to other sphere-tree models, the medial-axis sphere tree can not only approximate an object with a relatively small number of spheres, but can also achieve time-critical collision detection easily.

We use Bradshaw's sphere tree construction tool-kit (Bradshaw 2003), which contains a number of different algorithms to construct sphere trees of 3D objects. In particular, we use his combined algorithm to generate sphere trees with at most eight children per node. The combined algorithm allows the use of different sphere reduction algorithms in conjunction and chooses the one that result in the best fit. The sphere trees of a bunny and a dragon are shown in Fig. 2.10. The sphere trees of a dental probe and a tooth are shown in Fig. 2.11.

For each level in the sphere tree, we can compute the geometric error between the sphere models and the triangle-mesh representations. In Sect. 2.5.1, we use two

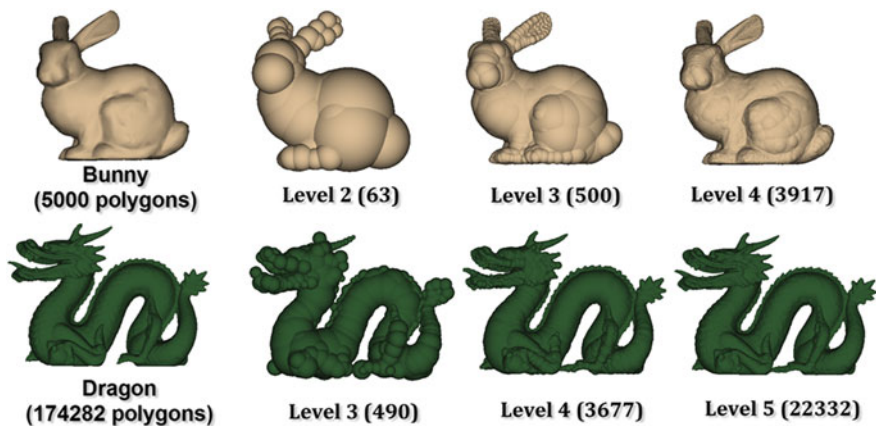


Fig. 2.10 Sphere trees generated for a bunny and a dragon. The *left* most figures are the polygonal models, and the rest are different levels of sphere trees. During implementation, level 4 (3917 spheres) is adopted to approximate a bunny and level 5 (22,332) is used for a more detailed dragon. Copyright © IEEE. All rights reserved. Reprinted, with permission, from Wang et al. (2013)

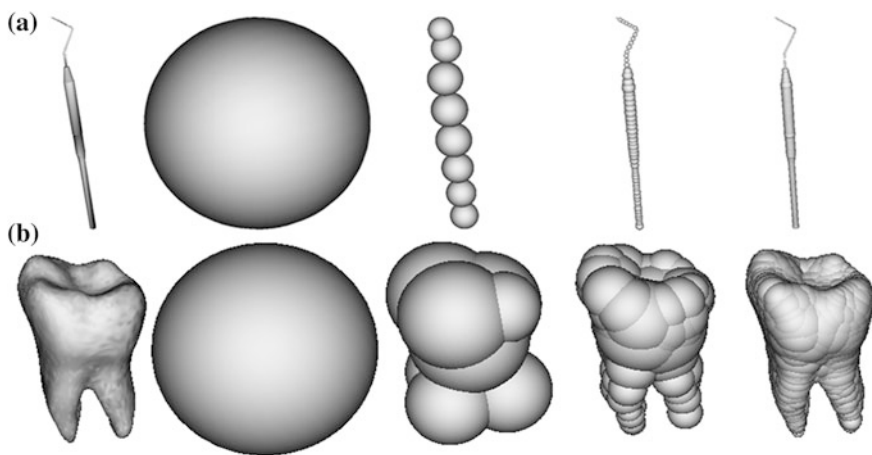


Fig. 2.11 Sphere trees generated for a dental probe and a tooth using the combined algorithm, **a** and **b** are models of the probe and tooth. The *left* most figures show the polygonal models, and the rest of the figures show the sphere trees varying from level 0 to level 3. Copyright © IEEE. All rights reserved. Reprinted, with permission, from Zhang et al. (2011)

benchmark objects to analyze the influence of the geometric error on the force-feedback fidelity.

Furthermore, physical parameters can be attached to each sphere, including stiffness, friction coefficient, and Young's Modulus.

2.3 Collision Response

We now describe how to compute the collision response when a tool interacts with objects in a virtual environment, using the configuration-based optimization approach and sphere-tree representation of the tool and objects. The computational pipeline is illustrated in following sub-sections.

2.3.1 Non-linear Transform from C-Space to Cartesian Space

In the optimization model shown in Eq. (2.1), the objective function is described in the configuration space (C-Space) of the tool, while the constraint functions are described in the Cartesian work space. There exists a non-linear mapping (in terms of trigonometric functions) between the two spaces because of the orientation components in the configuration of the graphic tool.

The coordinate transform between configuration variable $\mathbf{q}_g$ and Cartesian coordinate of the center of each sphere can be derived as Eq. (2.12).

2.3.2 Linearized Model of the Non-penetration Constraints

For fast computation, it is necessary to approximate the highly non-linear constraint for non-penetration with a linear model while keeping errors small for the solution variable (i.e., the configuration of the graphical tool). As changes of position and orientation of the haptic device are small between adjacent time steps, Taylor expansion is a suitable means for approximating the non-linear constraint.

We use a first-order Taylor expansion:

$$\begin{aligned}
 & C_i \left(x_T \left(\mathbf{q}_g^t \right), y_T \left(\mathbf{q}_g^t \right), z_T \left(\mathbf{q}_g^t \right) \right) \\
 & \approx C_i \left(x_T \left(\mathbf{q}_g^{t-1} \right), y_T \left(\mathbf{q}_g^{t-1} \right), z_T \left(\mathbf{q}_g^{t-1} \right) \right) \\
 & + \sum_{j=1}^6 \left[\frac{\partial C_i}{\partial x_T} \frac{\partial x_T}{\partial (\mathbf{q}_g^t)_j} + \frac{\partial C_i}{\partial y_T} \frac{\partial y_T}{\partial (\mathbf{q}_g^t)_j} + \frac{\partial C_i}{\partial z_T} \frac{\partial z_T}{\partial (\mathbf{q}_g^t)_j} \right] \cdot \left[(\mathbf{q}_g^t)_j - (\mathbf{q}_g^{t-1})_j \right]
 \end{aligned} \tag{2.17}$$

The error caused by Taylor expansion is smaller than $O\left((\Delta \mathbf{q}_g^t)^2\right)$. Because the change of configuration in one update cycle is relatively small, this error is imperceptible. Based on the preliminary test using a Phantom Premium device, the maximal velocity of the haptic tool in typical haptic operations is about 0.5 m/s. Because the update rate of the simulation is 1 kHz, the maximal displacement of the

haptic tool is about 0.5 mm in each cycle. Therefore, the displacement error caused by Taylor expansion is about 0.25 mm, which is far smaller than the size of the simulated objects (e.g., bunny and dragon). Note that the error from Taylor expansion is not related to the size of a sphere in the sphere-tree model. Furthermore, the truncation error will not accumulate along a sliding motion of the tool, because the configuration of the haptic tool in each simulation cycle will automatically reset the possible error of the graphic tool in the cycle.

2.3.3 Active Set Method for Solving Constrained Optimization

As described earlier, the objective function of the optimization model is quadratic, and we linearize the contact constraints via Taylor expansion. Thus, this becomes a typical quadratic programming (QP) problem. Furthermore, in the optimization model shown in Eq. (2.1), all the constraint functions are inequalities instead of equations. We need to find an efficient way to solve this problem. This problem is solved using the active set method (Nocedal and Wright 2006), which is generally considered as one of the most effective methods for small- to medium-scale QP problems, performs well in the algorithm.

The method can always find the optimal solution in a finite number of iterations, but a good start can shorten this process greatly. The variation of the graphic tool's configuration from one rendering cycle to the next is small due to the high update rates of rendering. In other words, the two optimal points (in the C-space of the graphic tool) are near each other; therefore, the optimal point in the previous loop can be a good initial point for iteration in the current loop. We choose $\mathbf{q}_g^{t-1}$ as the initial solution point $\mathbf{q}_0$. Note that the initial solution/condition should be selected without causing penetration between the graphic tool and the object.

Using the active set method, the original optimization problem in Eqs. (2.1) and (2.3) can be transformed to the following form

$$\begin{cases} \min : f(\mathbf{q}) = \frac{1}{2} \mathbf{q}^T \mathbf{M} \mathbf{q} \\ \text{s.t.} \quad J \mathbf{q} \geq -J \mathbf{q}_h^t \end{cases} \quad (2.18)$$

where J is formulated by Eq. (2.17), and

$$\mathbf{q} = (\mathbf{q}_g^t - \mathbf{q}_h^t) \quad (2.19)$$

Next, we use the active set method to find a step from one iteration to the next by solving a quadratic subproblem in which some of the constraints in (2.17) are imposed as equalities. Our algorithm continues to iterate in this manner until it reaches $\mathbf{q}_g^t$ that minimizes the quadratic objective function.

The main procedure of the active set method can be summarized as follows:

1. Find a feasible starting point;
2. Repeat until “optimal enough”;
 - (a) solve the equality problem defined by the active set (approximately);
 - (b) compute the Lagrange multipliers of the active set;
 - (c) remove a subset of the constraints with negative Lagrange multipliers;
 - (d) search for infeasible constraints;
 - (e) end repeat;

One limitation of the above active set solving method lies in the initial solution problem. Normally, we need to always maintain the initial solution in each simulation loop locating in the free space. As the initial solution is adopted as the solution of the graphic tool in the previous simulation cycle, the computation error by the truncation may produce small penetration. This issue may lead to the non-convergence of the iteration and needs to be further analyzed.

2.3.4 Contact Constraint-prediction Algorithm (CCP)

An important problem is how to find the contact constraints that the optimized configuration of the graphic tool has to satisfy, i.e., pairs of intersecting tool-object spheres. In previous work (Zhang et al. 2011), collision detection between the haptic tool and surrounding objects is carried out to obtain possible non-penetration and contact constraints. This conventional method leads to a large number of constraints when the shapes of the haptic tool and the object are complex. Furthermore, some constraints can be missed when the haptic tool is fully penetrated into objects. As shown in Fig. 2.12, if we use the pairs of intersecting spheres between the haptic tool and the object at time t to form the contact constraints for the configuration of the graphic tool at time t , the optimized (i.e., minimum energy) configuration of the graphic tool may not be penetration-free, that is, the contact constraints used in the optimization are not sufficient. To obtain sufficient number of constraints, the brute force method will be to form a non-penetration constraint from every pair of tool-object spheres, regardless if they are intersecting or not, which is inefficient.

Therefore, we introduce a *contact constraint-prediction* (CCP) method, taking advantage of the graphic tool’s contact configuration in the previous time instant $t - 1$ and the motion coherency of the graphic tool. We use a proximity detection concept to enlarge the size of the graphic tool with a predefined threshold. The CCP method first decides the maximum distance δ that the tool can move in the interval between $t - 1$ and t based on the speed of the haptic tool. Next, each sphere of the graphic tool at $t - 1$ is enlarged by this threshold δ and the intersecting pairs of spheres between the enlarged graphic tool and the object are identified. These intersecting sphere pairs are used to form the contact constraints, as shown in Fig. 2.13.

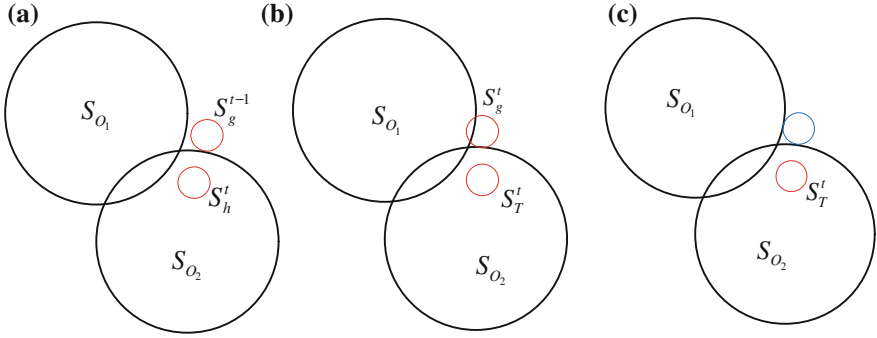


Fig. 2.12 Contact constraints from collision detection are not sufficient. **a** From collision detection, only the pair S_h^t and S_{O_2} is used to create a contact constraint. **b** S_g^t is the result of optimization with the single contact constraint from **(a)**, which intersects O_1 . **c** The desired result of optimization is shown as subject to both contact constraints from the two obstacle spheres. Copyright © IEEE. All rights reserved. Reprinted, with permission, from Wang et al. (2013)

It should be noted that a uniform δ is applied to all spheres of the graphic tool at time $t - 1$ based on the maximum distance the haptic tool can move in one haptic simulation cycle, which is corresponding to an estimated maximal velocity about 0.5 m/s.

Compared with the straightforward approach of detecting all pairs of intersecting spheres, the CCP method greatly reduces the number of constraints considered in the optimization without missing necessary constraints. We will demonstrate the benefit of the CCP method in the experiments in Sect. 2.6.

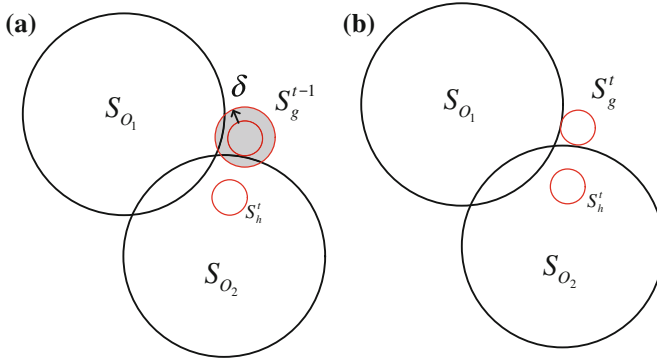


Fig. 2.13 CCP predicts all the possible contact constraints to ensure non-penetration of the graphic tool. **a** An enlarged sphere of the graphic tool at previous time step. **b** A desired position of the sphere of the graphic tool at current time step found using the CCP method. Copyright © IEEE. All rights reserved. Reprinted, with permission, from Wang et al. (2013)

2.4 Six-dimensional Force/Torque Simulation

After solving the configuration of the graphic tool, the next important step is to compute the contact force/torque exerted to the tool, and display them to the human user via the haptic device. Stability, fidelity, and fast response time are the main goals of force/torque computation and rendering models.

2.4.1 Existing Literature

Stability of haptic rendering is the most important criterion for designing a force/torque computation model. Early stability analysis in haptic rendering was focused on the problem of rendering stiff virtual walls. Several researchers reached the conclusion that high force update rates were necessary in order to achieve stable rendering (Colgate and Schenkel 1994; Brooks et al. 1990; Salcudean and Vlaar 1994). Intermediate representations (Adachi et al. 1995) were proposed to maximize the update rate of haptic rendering systems by performing a full update of the virtual environment at a low frequency (limited by computational resources and the complexity of the system) and using a simplified approximation for performing high-frequency updates of force feedback. Based on the z-width law of the haptic interface (Colgate and Brown 1994), to maintain the stability of a haptic simulation, the simulated stiffness of the virtual environment should be smaller than the maximum stiffness of the associated haptic device.

A number of techniques for 6-DoF haptic rendering follow the approach of direct rendering (Gregory et al. 2000; Kim et al. 2003; Johnson and Willemsen 2004). In direct rendering, the virtual tool follows rigidly the position of the haptic device, and contact forces are displayed directly. In this way, there is no need to simulate the dynamics of the virtual tool. However, guaranteeing stable and responsive display is a daunting task, as both the rotational impedance and the update rate of the haptic cycle may be highly varied. Small contact stiffness values result in large, visually perceptible interpenetrations, while large contact stiffness values may induce instabilities when the update rate of collision detection drops.

Colgate et al. (1995) proposed a multidimensional viscoelastic virtual coupling for stable interaction with non-linear virtual environments. If the implementation of the virtual environment is guaranteed to be discrete-time passive, the design of a stable display is reduced to appropriate tuning of the parameters of the coupling. Colgate and Schenkel (1994) pointed out that one way of obtaining a discrete-time passive virtual environment was to implicitly integrate a continuous-time passive system. Adams and Hannaford (1998) extended the concept of virtual coupling by providing a unifying framework for impedance and admittance displays.

Several techniques for 6-DoF haptic rendering combine virtual coupling with rigid body simulation of the virtual tool (McNeely et al. 1999; Chang and Colgate 1997; Berkelman 1999; Ruspini and Khatib 2000). Wan and McNeely (2003)

instead employed a quasi-static simulation of the virtual tool. As mentioned before, the transparency of haptic display through virtual coupling may degrade with a slow simulation update rate or with a large virtual tool mass.

Researchers have also explored more flexible ways of ensuring system stability or passivity. Miller et al. (2000) extended Colgate's passivity analysis techniques by relaxing the requirement of passive virtual environments but enforcing cyclo-passivity of the complete system. Hannaford et al. (2002) investigated the use of passivity observers and passivity controllers. Mahvash and Hayward (2005) derived conditions for the passivity of a virtual environment where continuous-time passive local force models were activated sequentially.

2.4.2 Force/Torque Computation for Frictionless Contacts

Once the configuration of the graphic tool for time step t is obtained through configuration-based optimization, we use the following model to compute the contact force:

$$\mathbf{F}^t = [F_x, F_y, F_z]^T = G_t[x_g^t - x_h^t, y_g^t - y_h^t, z_g^t - z_h^t]^T \quad (2.20)$$

where G_t is a diagonal 3×3 stiffness matrix with k_t in the diagonal, where k_t denotes the translational spring stiffness between the graphic tool and the haptic tool.

Note that the virtual spring between the haptic tool and the graphic tool only takes effect when the haptic tool is partially or fully immersed into a virtual object, i.e., when it moves in the *constrained space*. The haptic tool and the graphic tool simply collocate if the haptic tool is in the *free space*. Thus, this method is different from the conventional “virtual coupling” method and avoids its damping effect.

The computed torque is with respect to a fixed world coordinate system W , and thus the torque computation procedure involves several coordinate transformations in a haptic simulation time step t , as described below:

1. Obtain the homogeneous transformation matrix of the haptic tool and its Euler angles with respect to the W .
2. Obtain the Euler angles of the graphic tool with respect to the W based on the optimized $\mathbf{q}_g^t$ of Eq. (2.18).
3. Compute the relative rotation matrix from the local coordinate frame of the graphic tool to the local coordinate frame of the haptic tool, using the following equation

$${}^g_h\mathbf{R} = \mathbf{R}_L^{-1}(\mathbf{q}_h^t)\mathbf{R}_L(\mathbf{q}_g^t) \quad (2.21)$$

where the rotation matrix $\mathbf{R}_L$ is defined as

$$\mathbf{R}_L(\mathbf{q}^t) = \begin{bmatrix} c\alpha^t c\beta^t & c\alpha^t s\beta^t s\gamma^t - s\alpha^t c\gamma^t & c\alpha^t s\beta^t c\gamma^t + s\alpha^t s\gamma^t \\ s\alpha^t c\beta^t & s\alpha^t s\beta^t s\gamma^t + c\alpha^t c\gamma^t & s\alpha^t s\beta^t c\gamma^t - c\alpha^t s\gamma^t \\ -s\beta^t & c\beta^t s\gamma^t & c\beta^t c\gamma^t \end{bmatrix} \quad (2.22)$$

4. Compute the torque with respect to the local coordinate frame of the haptic tool, by the following equation

$$\mathbf{T}'_L = [T_z, T_y, T_x]_L^T = G_r \Psi(\mathbf{g}_h \mathbf{R})_{3 \times 1} \quad (2.23)$$

where G_r is a diagonal 3×3 stiffness matrix with k_r in the diagonal, where k_r denotes the rotational spring stiffness between the graphic tool and the haptic tool. $\Psi(\cdot)$ denotes a function that computes the three Euler angles $(\alpha^t, \beta^t, \gamma^t)^T$ based on a given rotation matrix as shown in Eq. (2.22). Detailed derivation of this function can be found in (Craig 1989).

5. Compute the torque with respect to W by the following equation

$$\mathbf{T}'_W = [T_z, T_y, T_x]_W^T = G_r [\mathbf{R}_L(\mathbf{q}'_h)]_{3 \times 3} \Psi^t(\mathbf{g}_h \mathbf{R}) \quad (2.24)$$

Note that in the above method, the continuity of the force/torque relies on the continuity of the graphic tool's configuration from one time step to the next. Given that the configuration of the haptic tool $\mathbf{q}'_h$ continuously changes, and that the configuration of the graphic tool is computed as the local optimum solution that maintains a minimum Euler distance in six dimensions to the configuration of the haptic tool, the configuration of the graphic tool also changes continuously. Therefore, the computed force/torque changes continuously, and the stability of the haptic rendering can be maintained. The experiments introduced in the next section also empirically validate the stability of force/torque computation.

2.4.3 Force/Torque Computation for Frictional Contacts

In order to simulate subtle feeling of multi-region frictional contacts, we introduce the following extensions to the configuration-based method for friction simulation.

Given the known configuration of the graphic tool at the previous and current time steps $t-1$ and t , and the known haptic configuration of the tool at t , we can compute the following distance vectors

$$\begin{cases} \delta \mathbf{p}'_n = \mathbf{p}_{g_eq}^t - \mathbf{p}_{h_eq}^t \\ \delta \mathbf{p}'_c = \mathbf{p}_{g_eq}^{t-1} - \mathbf{p}_{g_eq}^t \end{cases} \quad (2.25)$$

where $\delta \mathbf{p}'_n$ and $\delta \mathbf{p}'_c$ are two intermediate parameters denoting the average moving distance of several sampling points on the haptic tool, and

$$\mathbf{p}_{g_eq}^t = \frac{1}{N} \sum_{k=1}^N \mathbf{p}_{g_k}^t \quad (2.26)$$

denotes the equivalent contact point of N contact points on the graphic tool at t . $\mathbf{p}_{g_eq}^{t-1}$ denotes the corresponding point at time $t - 1$. $\mathbf{p}_{h_eq}^t$ denotes the corresponding point on the surface of the haptic tool at t .

In Eq. (2.25), we need to compute the corresponding point $\mathbf{p}_{h_eq}^t$ of $\mathbf{p}_{g_eq}^t$ by the following

$$\mathbf{p}_{h_eq}^t = {}^W_h T \cdot \left[\left({}^W_g T \right)^{-1} \cdot \mathbf{p}_{g_eq}^t \right] \quad (2.27)$$

where ${}^W_h T$ and ${}^W_g T$ denote the homogeneous transformation matrix of the haptic tool and that of the graphic tool with respect to the world frame, respectively.

Since the tool and objects are modeled by sphere trees, the contact points in Eq. (2.26) can be found from intersecting spheres between the graphic tool and the objects after $\mathbf{q}_g^t$ is obtained from optimization.

We can compute the contact force signal (combining both normal and friction forces) as follows:

$$\mathbf{F}_{3 \times 1} = k_e \left(\mathbf{p}_g^* - \mathbf{p}_h^t \right) \quad (2.28)$$

where k_e denotes the maximum stiffness of the associated haptic device, $\mathbf{p}_h^t$ denotes the 3×1 position vector in $\mathbf{q}_h^t$ (i.e., the 6D configuration vector of the haptic tool), and

$$\mathbf{p}_g^* = \mathbf{p}_g^{t-1} + (1 - k_\alpha) \left(\mathbf{p}_g^t - \mathbf{p}_g^{t-1} \right) \quad (2.29)$$

where $\mathbf{p}_g^t$ denotes the position vector of $\mathbf{q}_g^t$ (the 6D configuration vector of the graphic tool). Also

$$\begin{cases} k_\alpha = 1 & (\|\delta \mathbf{p}_c^t\| / \|\delta \mathbf{p}_n^t\|) \leq \mu_s \\ k_\alpha = \mu_d / \mu_s & (\|\delta \mathbf{p}_c^t\| / \|\delta \mathbf{p}_n^t\|) > \mu_s \end{cases} \quad (2.30)$$

where μ_s denotes the static friction coefficient, which defines a coulomb friction cone (shown in Fig. 2.14) (Duriez et al. 2006). μ_d denotes the dynamic friction coefficient.

Similarly, we can derive the contact torque (combining torques from both normal and friction forces) as follows:

$$\boldsymbol{\tau}_{3 \times 1} = k_\tau \left(\boldsymbol{\Phi}_g^* - \boldsymbol{\Phi}_h^t \right) \quad (2.31)$$

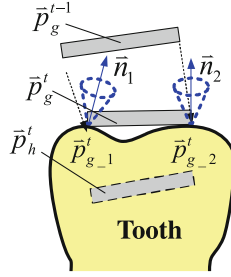


Fig. 2.14 Multi-point contacts, friction cones (in blue), and configurations involved in computing frictional contact force/torque. Copyright © IEEE. All rights reserved. Reprinted, with permission, from Wang et al. (2012a, b, c)

where k_τ denotes the maximum rotational stiffness of the associated haptic device. Φ_h^t denotes the 3×1 vector of Euler angles in $\mathbf{q}_h^t$. Φ_g^* denotes the vector of updated Euler angles after considering friction effect, which is computed by interpolating the general rotation angle as in the following steps:

1. The relative motion of the graphic tool from $t - 1$ to t can be derived and expressed by a homogeneous transformation matrix ${}^W_g T^t \cdot \left[\left({}^W_g T^{t-1} \right)^{-1} \right]$.
2. Using the general rotation matrix (Craig 1989), the equivalent general rotation axis $\bar{a}$ and the rotation angle θ_g can be obtained.
3. Let p be the contact point on the graphic tool that is closest to the rotation axis $\bar{a}$ at time t . Let the distance between p and $\bar{a}$ be r . Then define

$$\begin{cases} k_\theta = 1 & (r \cdot \theta_g / \|\delta \mathbf{p}_n^t\|) \leq \mu_s \\ k_\theta = \mu_d / \mu_s & (r \cdot \theta_g / \|\delta \mathbf{p}_n^t\|) > \mu_s \end{cases} \quad (2.32)$$

4. Next compute an interpolated rotation angle

$$\theta_g^* = (1 - k_\theta) \theta_g \quad (2.33)$$

5. Finally, compute the new Euler angles Φ_g^* corresponding to the interpolated rotation angle.

One important step to simulate static friction is to reset the configuration of the graphic tool to the configuration in the previous time step, i.e.,

$$\text{if } (k_\alpha = 1 \ \& \ k_\theta = 1), \text{ then } \mathbf{q}_g^t = \mathbf{q}_g^{t-1} \quad (2.34)$$

This means that the graphic tool is stuck on the object's surface (i.e., neither translation nor rotation can occur). It should be noted that the above reset operation may occur for several consecutive steps. The number of the consecutive reset steps

N_{reset} is a function of the time that the human operator pauses the movement of the haptic tool and the speed of the operator's movement of the haptic tool.

Compared to the iterative solutions using Gauss–Seidel algorithm (Duriez et al. 2006), the mathematical model of the method is easier to be implemented by engineering developers.

2.5 Experimental Validation

In this section, we introduce a number of experiments using benchmark virtual environments to validate the configuration-based optimization approach to 6-DoF haptic simulation.

A Phantom Premium 3.0 6-DoF is used as the haptic device to display six-dimensional forces and torques (shown in Fig. 2.15). The specifications of the computer are as follows: Intel (R) Core (TM) 2 2.20 GHz, 2 GB memory, and X1550 Series Radeon graphical card. Note that no GPU is used in the implementation of the haptic simulation method.

Three metrics are considered in performance evaluation:

- Accuracy: When a contact occurs, there is no visual artifact (i.e., no interpenetration between the tool and the object), and no haptic artifact (i.e., no force felt when there is still a distance between the tool and the object, or no artificial friction and sticking when the tool moves along the object's surface).
- Stability: There is no big jump or vibration of the graphic tool between two adjacent simulation time steps, and the feedback force/torque is smooth and stable.

Fig. 2.15 Interaction with a Phantom Premium 3.0/6-DoF device. Copyright © IEEE. All rights reserved. Reprinted, with permission, from Wang et al. (2013)



- Computation update rate: the update rate is higher than 1 kHz under all contact scenarios (i.e., either a single contact or a multi-region contact case).

2.5.1 Results of Accuracy Analysis

We evaluate accuracy in terms of accuracy of the computed force/torque signal and accuracy of the configuration of the graphic tool.

Accuracy of the computed feedback force/torque signal is defined as the following error at any time step to measure the haptic realism of the target haptic rendering method:

$$\begin{cases} \delta_F = (\mathbf{F}_{gm} - \mathbf{F}_{tm}) / \|\mathbf{F}_{gm}\| \\ \delta_\tau = (\boldsymbol{\tau}_{gm} - \boldsymbol{\tau}_{tm}) / \|\boldsymbol{\tau}_{gm}\| \end{cases} \quad (2.36)$$

where $\mathbf{F}_{gm}$ and $\mathbf{F}_{tm}$ indicates the three-dimensional force signal from a ground-truth model and a target model, respectively. $\boldsymbol{\tau}_{gm}$ and $\boldsymbol{\tau}_{tm}$ indicates the three-dimensional torque signal from the ground-truth model and the target model, respectively.

Similarly, the error of position and orientation signal of the graphic tool is defined to evaluate the visual realism of the target haptic rendering method:

$$\begin{cases} \delta_X = (\mathbf{x}_{gm} - \mathbf{x}_{tm}) / \|\mathbf{x}_{gm}\| \\ \delta_\theta = (\boldsymbol{\theta}_{gm} - \boldsymbol{\theta}_{tm}) / \|\boldsymbol{\theta}_{gm}\| \end{cases} \quad (2.37)$$

where $\mathbf{x}_{gm}$ and $\mathbf{x}_{tm}$ indicate the three-dimensional position signal of the graphic tool from the ground-truth model and the target model, respectively. $\boldsymbol{\theta}_{gm}$ and $\boldsymbol{\theta}_{tm}$ indicate the three-dimensional orientation angle of the graphic tool from the ground-truth model and the target model, respectively.

Based on those error definitions, the smaller of the error, the more accurate is the target haptic rendering algorithm.

The accuracy of simulation comes from the quadratic programming (QP) formulation. In contrast to previous methods, exact non-penetration between the graphic tool and the object can be maintained, without introducing numeric error caused by integration computations. Possible factors that can affect accuracy are the geometric error of using spheres to approximate an object's shape and the error incurred in using Taylor expansion. Our experiments quantify that those two factors have negligible effects so that the method can achieve high accuracy.

In each experiment, we use analytic functions to describe a ground-truth model, i.e., an object with analytic expression of its geometry. Specifically, an ellipsoid or a cube is used. A sphere with 5 mm radius is used as the tool. Different levels of sphere trees are utilized for the cube and the ellipsoid. The experiments are expected to show that the geometric error between a high-resolution sphere-tree model and the analytic model of the object is sufficiently small, so that the force

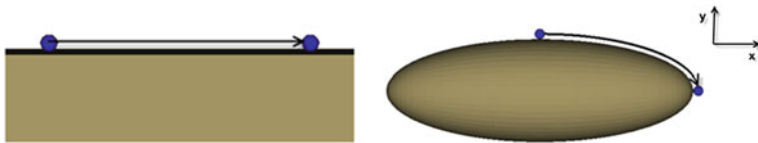


Fig. 2.16 Sliding trajectory of tool on the target object's surface is confined to the x - y plane. Copyright © IEEE. All rights reserved. Reprinted, with permission, from Wang et al. (2013)

error (computed by Eq. 2.36) between the sphere-tree model and the analytic model is also small enough, i.e., smaller than human's discriminating threshold.

Here, a user moves the haptic tool along a fixed path on the planar surface of the cube or the curved surface of the ellipsoid represented by a sphere tree (as shown in Fig. 2.16). The ideal force magnitude should be 1 N, and the ideal force direction should be along the surface normal.

In each case, we record samples of the contact force. The errors of contact force components between the ideal plane and the sphere-tree model are shown in Fig. 2.17. The horizontal axis denotes the simulation time step (in millisecond), and vertical axis denotes the error of force components computed by Eq. (2.36). Similarly, the errors of contact force components between the ideal ellipsoid and the sphere-tree model are shown in Fig. 2.18. It should be noted that the errors shown are only for force (not torque) along the x - and y -axes, because the predefined sliding trajectory is confined to the x - y plane, and there is no rotation.

From the figures, we can see clearly that the geometric error decreases as deeper sphere trees with more spheres are utilized. But the trend is not necessarily linear. Rigorous experiments or theoretical analysis is needed to determine the exact relationship between the error and the resolution of the sphere tree. For the cube model, the average force error under 4,096 spheres is at about 0.05. For the

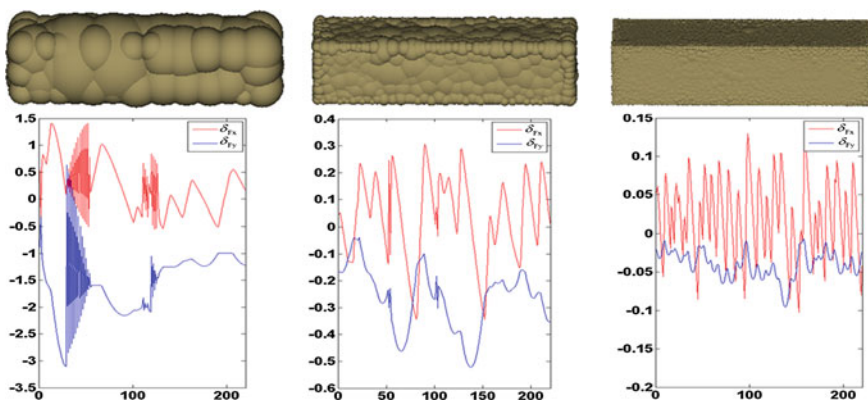


Fig. 2.17 Results of sliding along the cube (the number of spheres are 64, 512, and 4,096, respectively). Copyright © IEEE. All rights reserved. Reprinted, with permission, from Wang et al. (2013)

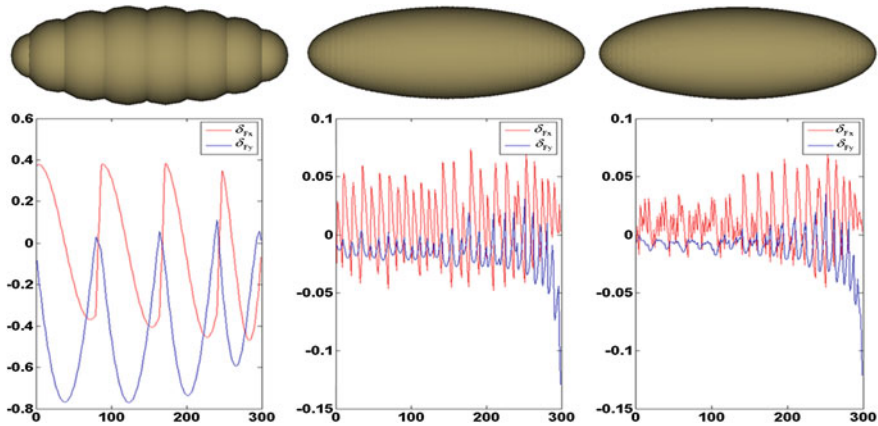


Fig. 2.18 Results of sliding along the ellipsoid (the number of spheres is 8, 64, and 512, respectively). Copyright © IEEE. All rights reserved. Reprinted, with permission, from Wang et al. (2013)

ellipsoid model, the maximum force error under 512 spheres is at about 0.05. Since human's Just Noticeable Difference (JND) for force magnitude is about 6–8 % (Jones et al. 1992; Pang et al. 1991; Tan et al. 1995), the average 5–10 % force errors in Figs. 2.20 and 2.21 suggest that the computed force/torque based on the sphere-tree models is accurate enough comparing to the computed force/torque based on the precise, analytic model of the tool, and object.

Using the sphere-tree model is more suitable for approximating curved objects than using polygonal mesh models. For a ground-truth model such as the ellipsoid, a sphere-tree model with a total of 64 spheres can achieve lower force error than a polygonal mesh with over 4,000 triangles. On the other hand, to simulate straight edges or flat surfaces, a finer sphere-tree model is required than simulating a curved object. As shown in Figs. 2.20 and 2.21, the sphere tree with a total of 4,096 spheres to simulate the cube yields a greater force error than the sphere tree with a total of 512 spheres to simulate the ellipsoid. The perceptual artifacts depend on the shape of the explored object.

As for the truncation error caused by the Taylor expansion, it is recorded in all the experiments, when the tool slides along the object's surface. The error values are always less than one micrometer. Hence, it is sufficiently small so that the corresponding visual artifact is negligible.

2.5.2 Stability and Update Rate—Exp 1: Dental Operations

In a dental operation, multi-region contacts often occur between the dental tool and oral tissues. Frequent contact switches occur when the tool moves or rotates within

the small oral cavity or even in the narrow cavity of periodontal pocket. Two kinds of dental operations are simulated to validate the stability of the haptic rendering method.

2.5.2.1 Periodontal Operation

One important periodontal operation is a periodontal pocket examination through probing. Periodontal tissue consists of gingiva, periodontal ligament, alveolar bone, and root cementum. Figure 2.19 shows the comparison between healthy periodontium and inflamed periodontium. As shown in Fig. 2.19c, periodontal pocket is pathologically deepened gingival sulcus (the shallow crevice between free gingiva and tooth surface). Therefore, periodontal pocket probing is an important operation for periodontal diagnosis. During the examination, a dentist uses a periodontal probe to check the depth value of the pocket, and thus to determine the degree of the periodontal inflammation.

This operation involves frequent changes of multiple contacts and thus is ideal for checking the stability of the haptic simulation method. As shown in Fig. 2.20, first, the operator slides the probe on the tooth and gingiva until finding the location of inflammation. Then, the tool is manipulated into the sulcus between the tooth and gingiva for inspection. The quite even curves of the force and torque demonstrate the stability of the algorithm. Also, the operator cannot feel any vibration of the haptic device. After the twentieth second, we can see clear increase of the torque

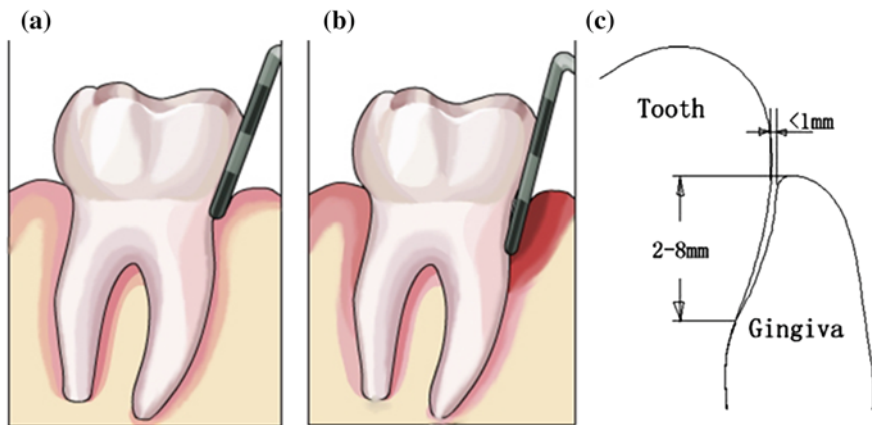


Fig. 2.19 A periodontal pocket. Copyright © IEEE. All rights reserved. **a** Healthy periodontium, **b** periodontium with inflammation, **c** illustration of periodontal pocket. Reprinted, with permission, from Wang et al. (2013)

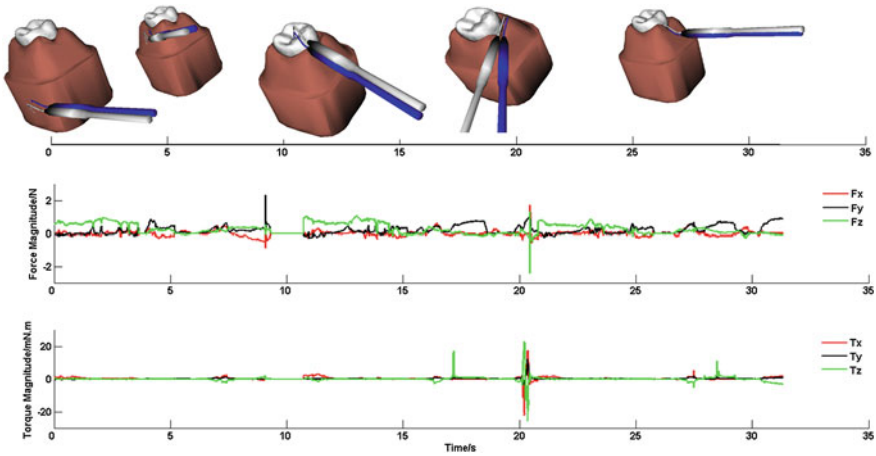


Fig. 2.20 Although the haptic tool (the *blue one*) has penetrated into the tooth/gingival, the graphic tool (the *white one*) remains just in contact. Copyright © IEEE. All rights reserved. Reprinted, with permission, from Wang et al. (2013)

signal when the operator voluntarily rotates the tool in the periodontal pocket. Non-penetration is maintained between the graphic tool (in white color) and the tissues (both tooth and gingiva). A video showing the process is available online from the book Website <http://extras.springer.com/>.

2.5.2.2 Oral Cavity Inspection

In oral cavity inspection, a dentist uses a dental probe to examine possible diseases such as angular stomatitis, glossitis, swollen or bleeding gums, and decayed teeth. For this task, a complete oral virtual environment is necessary to simulate the movement of the probe within the oral cavity, which includes all teeth, lower jaw, upper jaw, tongue, and cheek. Figure 2.21a shows the dental surgery simulator. From Table 2.1, we can see the total number of triangles and leaf-node spheres is rather large in this experiment.

This is a challenging task to simulate at the desired haptic rate (i.e., 1 kHz). We test the simulation in the following steps: (1) Examine every single tooth to see if any decay exists. (2) Examine the gingiva to see if it is inflamed (this manipulation will make the tool contact several parts in a mouth, and this is why the curves shown in Fig. 2.21 rise up after the twentieth second). The results shown in Fig. 2.21b demonstrate that the haptic rendering method is efficient and provides stable interaction. A video showing the process is available online from the book Website <http://extras.springer.com/>.

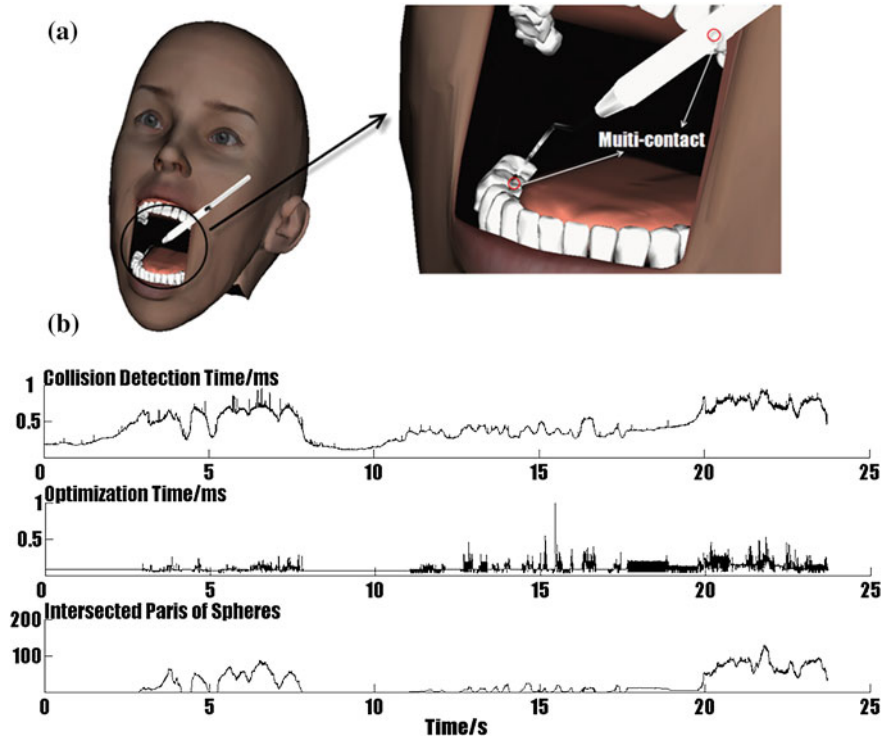


Fig. 2.21 Oral cavity inspection. Collision detection time and optimization time are related to the number of intersected spheres. The algorithm can still maintain haptic rates even though hundreds of spheres intersect. Copyright © IEEE. All rights reserved. Reprinted, with permission, from Wang et al. (2013). **a** Interaction scenario. **b** Time cost for collision detection and optimization

Table 2.1 Number of triangles and leaf-node spheres in the oral cavity inspection experiment

	Polygons	Leaf-node spheres
Probe	1,088	512
Face	3,138	512
Upper jaw	2,280	512
Lower jaw	2,280	512
Tongue	1,045	512
Teeth	$28 \times 4,000$	28×512

2.5.3 Stability and Update Rate—Exp 2: Bunny Versus Bunny

In Fig. 2.22, the brown bunny represents the graphic tool, and the white bunny represents the static object. Both bunnies are modeled using level-4 sphere trees (with 3,076 spheres in each bunny).

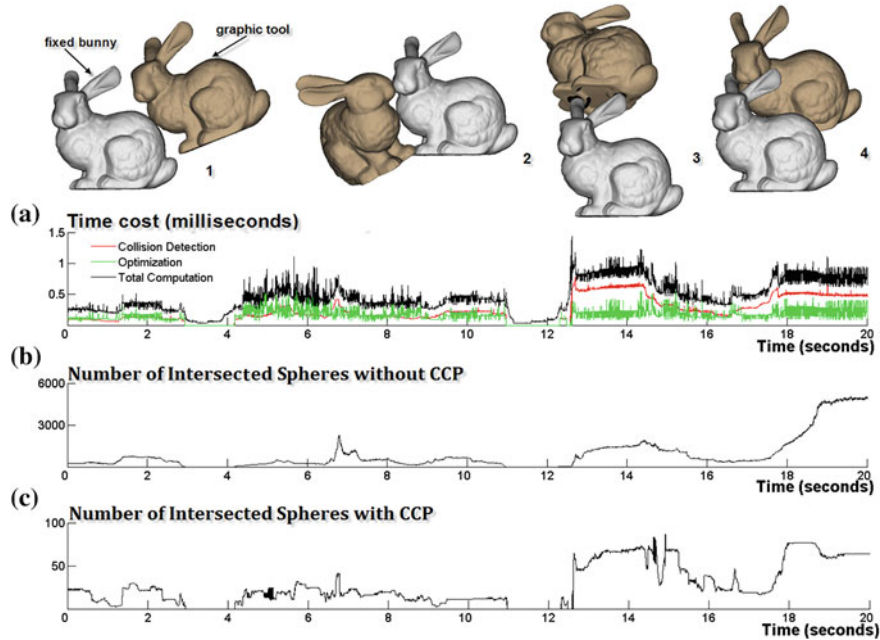


Fig. 2.22 Results of haptic interaction with the Stanford bunnies. *Top* four key states of the interaction (the *white bunny* is static). *Plot a* time cost of the main algorithms in the approach. *Plot b* the number of intersected spheres with conventional collision detection. *Plot c* the number of intersected spheres with CCP method. Copyright © IEEE. All rights reserved. Reprinted, with permission, from Wang et al. (2013)

During the simulation, several contact states are examined. In state 1, the brown bunny slides along the back of the white bunny to validate whether the method can simulate continuous contact between the tool and the object. In state 2, the brown bunny makes a multi-region contact (with three simultaneous contact regions). In state 3, the white bunny’s two ears get into the holes at the bottom of the brown bunny for fine interaction. In state 4, the brown bunny is squeezed between the two ears of the white bunny. When the operator rotates the brown bunny and maintain the contacts with the two ears, frequent contact switches occur. The interaction is very stable.

The time efficiency is one of the most significant advantages of the method compared to existing constraint-based haptic rendering methods. Figure 2.22a shows the efficiency of the method. The time cost of the collision detection fluctuates when the contact state between the tool and the object changes. The maximum time cost of the collision detection is less than 1 ms. The time cost of the optimization is always less than 0.5 ms, benefiting from the selection of the initial iteration configuration. Therefore, the overall time cost in each simulation cycle is less than 1.5 ms, which meets the strict update rate for haptic rendering. Whereas for the same example of bunny-bunny interaction in (Ortega et al. 2007), the

simulation time for the god-object is in the range of 1–15 ms. The method is more efficient because in (Ortega et al. 2007), the continuous collision detection (CCD) method relies on mesh–mesh collision check, which is more time intensive, and it is embedded in the main simulation loop.

Figure 2.22b indicates the number of intersected spheres using conventional discrete collision detection (i.e., checking intersection between the haptic tool and the virtual environment), and Fig. 2.22c indicates the number of intersected spheres using the CCP method. Comparing Fig. 2.22b with c, we can see that, CCP not only provides sufficient constraints, but also greatly reduces redundant constraints, which makes the algorithm more efficient.

2.5.4 Stability and Update Rate—Exp 3: Buddha Versus Dragon

Figure 2.23a shows a moving Buddha statue interacting with a static dragon statue. Both objects are modeled by a level-5 sphere tree consisting of more than 20,000 spheres. We compute a non-penetration configuration of the graphic tool (as shown in Fig. 2.23a) using different levels of the sphere tree (note that the Buddha and the dragon have the same level). Figure 2.23b shows a state where the Buddha and the

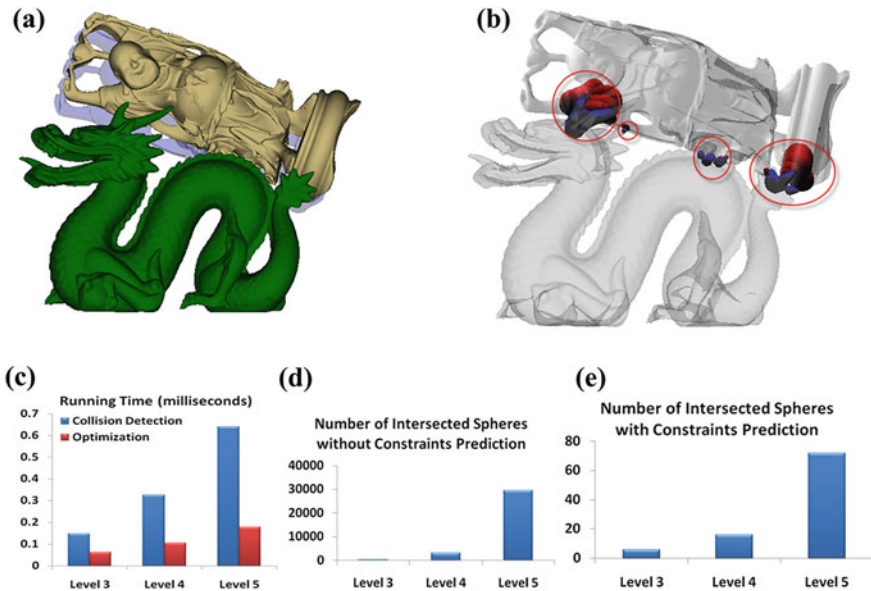


Fig. 2.23 Haptic simulation of multi-region contacts involving statues of a moving Buddha and a static dragon. Copyright © IEEE. All rights reserved. Reprinted, with permission, from Wang et al. (2013)

dragon have four simultaneous contact regions. From Fig. 2.23c, one can tell the time costs of the collision detection and optimization increase along with the increased levels. However, they are below 1 ms even for level 5 (with more than 20,000 spheres). Considering the contact distribution, the number of simultaneous contact pairs of spheres cannot be too large; otherwise the time cost for optimization will exceed 1 ms. From the experimental results, the maximum number of simultaneous pairs of intersecting spheres can be found to be about 500. As shown in Fig. 2.23d and e, the number of intersecting spheres after using the CCP algorithm is kept to a relatively low level.

The time costs for collision detection and optimization in one haptic simulation cycle are provided in Fig. 2.23. From this figure, we can see the time cost for collision detection is maintained within 1 ms even for highly detailed sphere trees (e.g., level-5 sphere trees for the dragon and Buddha). Using the CCP method, the number of intersecting sphere pairs is greatly reduced, and thus the time cost for optimization is greatly reduced.

2.5.5 Stability and Update Rate—Exp 4: Bunny Navigating

This experiment is carried out to further illustrate the performance of the method in a complex scenario with frequent contact switches.

A pipe system consists of 17 separate pipes, and a bunny is manipulated to move through it. We show four key states of the motion in Fig. 2.24, and the black curve indicates the path of the bunny's centroid. Because it is a complex, irregular virtual environment, a human operator is required to find a way out for the bunny relying on the force feedback. In this experiment, frequent contact switches occur, i.e., the number of the contact regions and the pairs of intersecting spheres in each region will change whenever small translation/rotation of the bunny occurs. The approach performs well for such a challenging task with frequent contact switches.

As shown in Fig. 2.24a, b, the force/torque magnitudes change frequently in all three directions. However, as shown in Fig. 2.24c, d, the difference between adjacent force/torque values is relatively small, i.e., the stability of the haptic manipulation is maintained. The update rate of the haptic simulation cycle is always over 1 kHz.

2.5.6 Validation of Friction Simulation

As shown in Fig. 2.25, the force signals (with or without friction effect) during two continuous movement segments are recorded. In the two segments, the operator moves the tool approximately along z -axis and x -axis, respectively. During each movement segment, multi-region contacts occur between the graphic tool and the tooth.

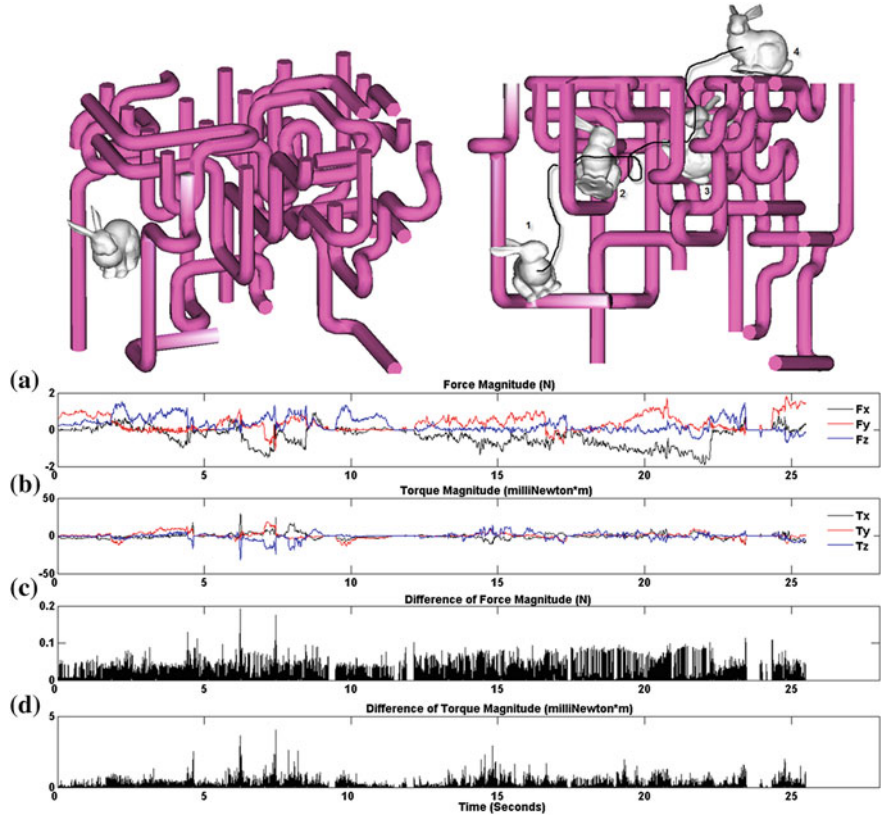


Fig. 2.24 A bunny through a pipe system to simulate frequent contact switches. State 2 indicates manipulating the bunny through U-shape pipes. From state 2 to state 3, the bunny is turned around to make its body get through first so that the bunny’s ears can get through. Copyright © IEEE. All rights reserved. Reprinted, with permission, from Wang et al. (2013)

We can see that the component of the friction force is consistent with the movement direction of the haptic tool, i.e., the friction component along z -axis is significant in the first segment, and the friction component along x -axis is significant in the second segment. The static and dynamic friction coefficients were set as 0.8 and 0.3, respectively, in the force/torque curves.

Because the movements of the tool might not be strictly aligned with the z direction in the first segment, there is also an increase in the force along x -axis. During the second segment, the movement is sliding motion on top of the tooth.

Next, progressive change of friction force is validated. A circular area is set as a calculus. Because the scaler and the calculus are both modeled by sphere trees, multi-point contacts occur during the scaling process. As shown in Fig. 2.26, we can observe the decrease of friction force along y -axis during the removal process (as shown in video, the scratching movement of the scaler is mainly along the y -axis). As calculus removal proceeds, the number of reset steps becomes smaller,

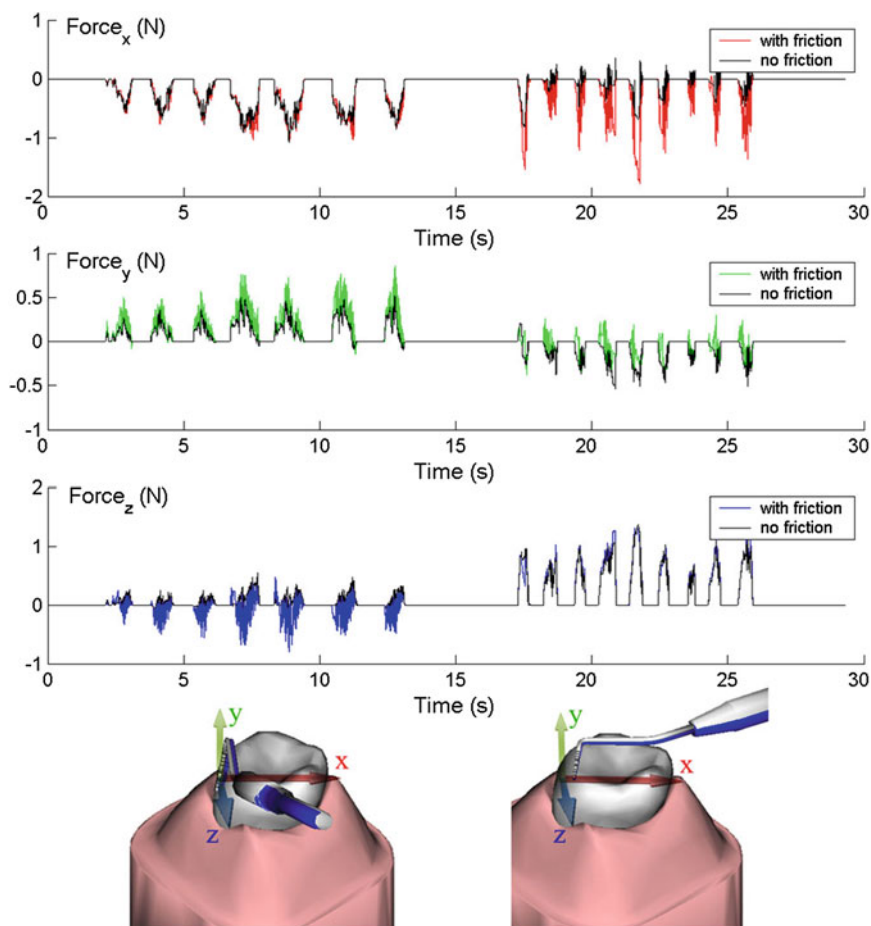


Fig. 2.25 Multi-contact force signals including both normal and friction effect during two movement segments. Copyright © IEEE. All rights reserved. Reprinted, with permission, from Wang et al. (2012a, b, c)

and the magnitude of friction force also becomes smaller. After the calculus was completely removed, no friction could be felt. A video showing the process is available online from the book Website <http://extras.springer.com/>.

The configuration-based simulation method for multi-contact friction is very efficient and can maintain 1 kHz update rate of the haptic loop without the use of special computation hardware such as GPU or multi-core. It can simulate progressive or changing friction effect conveniently and stably.

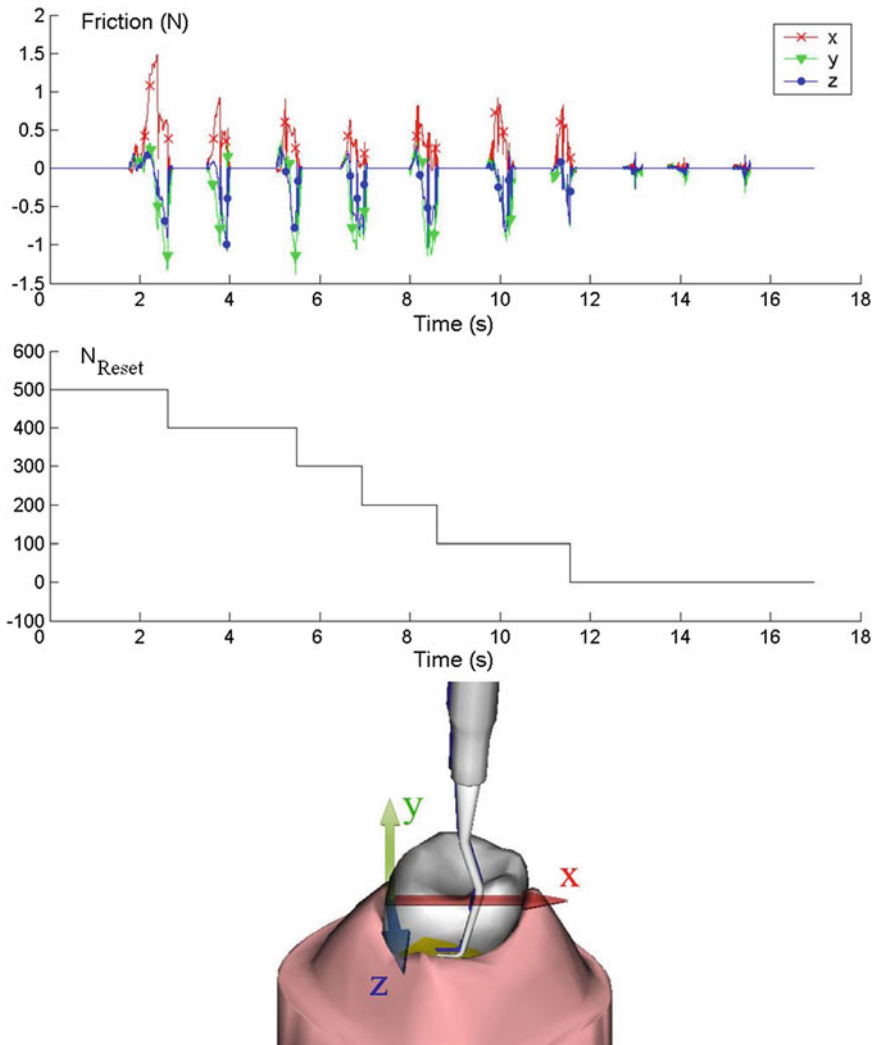


Fig. 2.26 Changing friction effects during calculus removal process by gradually change the value of the number of reset steps. Copyright © IEEE. All rights reserved. Reprinted, with permission, from Wang et al. (2012a, b, c)

2.6 Summary

In this Chapter, we have introduced an efficient constraint-based 6-DoF haptic rendering approach, i.e., a configuration-based optimization method based on sphere-tree representations of objects to compute the collision response for 6-DoF haptic rendering for fine manipulation.

This approach is a significant extension of the god-object method in three-dimensional space (Zilles and Salisbury 1995). With this approach, visual artifacts are avoided, i.e., no perceptible penetration or separation between the graphic tool and an object. The graphic tool is stopped at the surface of the object. Different from existing constraint-based haptic rendering methods, the method is efficient such that it computes collision response at the 1 kHz haptic rate. Hence, it can capture the subtle change of contact constraints in adjacent time steps and can simulate multi-region contacts stably when the tool moves within a narrow space.

The efficiency of the method is ensured by a number of techniques. Sphere trees are used to represent both the tool and objects and model non-penetration constraints between the graphic tool and arbitrary-shaped objects in a simple and uniform expression. In addition to taking advantage of quick intersection check between spheres, a contact constraint-prediction (CCP) method further ensures the correctness and computation efficiency of finding contact constraints. Taylor expansion is used to linearize the non-linear constraint inequalities to achieve and maintain high computation rate for the optimization while maintaining small truncation errors. Finally, feedback force/torque is computed without using virtual coupling.

Extensive experimental testing demonstrates that with this method, high accuracy, stability, and high update rates are achieved in some complex task environments with multi-region contacts and frequent contact switches.

For multi-region contacts involving large contact regions, the time cost of the optimization method will become a bottleneck because the number of the contact pairs can be very large. Even using CCP, the time cost will exceed 1 ms. This problem can be solved by eliminating some redundant constraints before doing optimization.

In the following chapters, this approach will be extended to simulating fine geometric features such as sharp edges, and to virtual environments consisting of both rigid and deformable objects.

References

- Adachi Y, Kumano T, Ogino K (1995) Intermediate representation for stiff virtual objects. In: Proceedings of virtual reality annual international symposium, pp 203–210
- Adams RJ, Hannaford B (1998) A two-port framework for the design of unconditionally stable haptic interfaces. In: Proceedings of IEEE/RSJ international conference on intelligent robots systems, pp 1254–1259
- Berkelman PJ (1999) Tool-based haptic interaction with dynamic physical simulations using Lorentz magnetic levitation. PhD dissertation, Robotics Institution, Carnegie Mellon University, Pittsburgh, PA
- Bradshaw G, Sullivan CO (2004) Adaptive medial-axis approximation for sphere-tree construction. *ACM Trans Graph* 23(1):1–26
- Bradshaw G (2003) Sphere-tree construction toolkit. <http://isg.cs.tcd.ie/spheretree/>, Feb 2003

- Brooks Jr FP, Ouh-Young M, Batter JJ, Kilpatrick PJ, Baskett F (eds) (1990) Project GROPE—haptic displays for scientific visualization. In: *Proceedings of computer graphics (SIGGRAPH)*, Aug 1990, vol 24, pp 177–185
- Chang B, Colgate JE (1997) Real-time impulse-based simulation of rigid body systems for haptic display. In: *Proceedings of ASME Dynamic Systems and Control Division*, pp 145–152
- Colgate JE, Brown JM (1994) Factors affecting the Z-width of a haptic display. In: *Proceedings of IEEE international conference on robotics and automation*, Los Alamitos, CA, pp 3205–3210
- Colgate JE, Schenkel GG (1994) Passivity of a class of sampled-data systems: application to haptic interfaces. In: *Proceedings of American control conference*, pp 3236–3240
- Colgate JE, Stanley MC, Brown JM (1995) Issues in the haptic display of tool use. In: *Proceedings of IEEE/RSJ international conference on intelligent robots systems*, pp 140–145
- Craig JJ (1989) *Introduction to robotics: mechanics and control*, 2nd edn. Addison-Wesley, Reading
- Duriez C, Dubois F, Kheddar A, Andriot C (2006) Realistic haptic rendering of interacting deformable objects in virtual environments. *IEEE Trans Vis Comput Graph* 12:36–47
- Gregory A, Mascarenhas A, Ehmann S, Lin MC, Manocha D (2000) 6-DOF haptic display of polygonal models. In: *Proceedings of IEEE visualization conference*, pp 139–146
- Hannaford B, Ryu J-H, Kim YS (2002) Stable control of haptics. In: McLaughlin ML, Hespanha JP, Sukhatme GS (eds) *Touch in virtual environments*, Chap 2. Prentice-Hall, Upper Saddle River, pp 47–70
- Ho C, Basdogan C, Srinivasan MA (1999) An efficient haptic rendering technique for displaying 3D polyhedral objects and their surface details in virtual environments. *Presence Teleoperators Virtual Environ* 8(5):477–491
- Hubbard PM (1996) Approximating polyhedra with spheres for time-critical collision detection. *ACM Trans Graph* 15(3):179–210
- Johnson DE, Willemsen P (2004) Accelerated haptic rendering of polygonal models through local descent. In: *Proceedings of haptics symposium*, pp 18–23
- Jones LA, Hunter IW, Irwin RJ (1992) Differential thresholds for limb movement measured using adaptive techniques. *Percept Psychophys* 52:529–535
- Kim YJ, Otaduy MA, Lin MC, Manocha D (2003) Six-degree-of-freedom haptic rendering using incremental and localized computations. *Presence* 12(3):277–295
- Lin MC, Otaduy M (2008) *Haptic rendering: foundations, algorithms, and applications*. A K Peters, Ltd, USA
- Mahvash M, Hayward V (2005) High-fidelity passive force-reflecting virtual environments. *IEEE Trans Robot* 21(1):38–46
- McNeely W, Puterbaugh K, Troy J (1999) Six degree-of-freedom haptic rendering using voxel sampling. In: *Proceedings of ACM SIGGRAPH*
- Miller BE, Colgate JE, Freeman RA (2000) Guaranteed stability of haptic systems with nonlinear virtual environments. *IEEE Trans Robot Autom* 16(6):712–719
- Nocedal J, Wright SJ (2006) *Numerical optimization*, 2nd edn. Springer, Berlin
- Ortega M, Redon S, Coquillart S (2007) A six degree-of-freedom god-object method for haptic display of rigid bodies with surface properties. *IEEE Trans Vis Comput Graph* 13(3):458–469
- Pang X, Tan HZ, Durlach NI (1991) Manual discrimination of force using active finger motion. *Percept Psychophys* 49(6):531–540
- Redon S (2004) Fast continuous collision detection and handling for desktop virtual prototyping. *Virtual Reality* 8(1):63–70
- Ruffaldi E, Morris D, Edmunds T, Barbagli F, Pai D (2006) Standardized evaluation of haptic rendering systems. In: *IEEE haptic interfaces for virtual environment and teleoperator systems, VR*
- Ruspini D, Khatib O (2000) A framework for multi-contact multi-body dynamic simulation and haptic display. In: *Proceedings of IEEE/RSJ international conference on intelligent robots systems*, pp 1322–1327
- Ruspini DC, Kolarov K, Khatib O (1997) The haptic display of complex graphical environments. In: Whitted T (ed) *Proceedings of SIGGRAPH 97, computer graphics proceedings, annual conference series*. Addison Wesley, Reading, pp 345–352

- Salcudean SE, Vlaar TD (1994) On the emulation of stiff walls and static friction with a magnetically levitated input/output device. In: Proceedings of ASME haptic interfaces for virtual environment and teleoperator systems, pp 303–310
- Tan HZ, Durlach NI, Beauregard GL, Srinivasan MA (1995) Manual discrimination of compliance using active pinch grasp: the roles of force and work cues. *Percept Psychophys* 57:495–510
- Tang M, Kim YJ, Manocha D (2009) C2A: controlled conservative advancement for continuous collision detection of polygonal models. *ICRA*, pp 849–854
- Wan M, McNeely WA (2003) Quasi-static approximation for 6 degrees-of-freedom haptic rendering. In: Proceedings of IEEE visualization conference, pp 257–262
- Wang D, Zhang X, Zhang Y, Xiao J (2011) Configuration-based optimization for six degree-of-freedom haptic rendering for fine manipulation. In: Proceedings of IEEE International Conference on Robotics and Automation (ICRA), pp 906–912
- Wang D, Zhang Y, Hou J, Wang Y, Lü P, Chen Y, Zhao H (2012a) iDental: a haptic-based dental simulator and its preliminary evaluation. *IEEE Trans Haptics* 5(4):332–343
- Wang D, Liu S, Zhang X, Zhang Y, Xiao J (2012b) Six-degree-of-freedom haptic simulation of organ deformation in dental operations. In: IEEE international conference on robotics and automation, ICRA 2012, St. Paul, 14–18 May 2012, pp 1050–1056
- Wang D, Liu S, Xiao J, Hou J, Zhang Y (2012c) Six degree-of-freedom haptic simulation of pathological changes in periodontal operations. In: 2012 IEEE/RSJ international conference on intelligent robots and systems (IROS2012), Vilamoura, Algarve, 7–12 Oct 2012
- Wang D, Zhang X, Zhang Y, Xiao J (2013) Configuration-based optimization for six degree-of-freedom haptic rendering for fine manipulation. *IEEE Trans Haptics* 6(2):167–180
- Weller R, Zachmann G (2009) A unified approach for physically-based simulations and haptic rendering. In: Proceedings of the ACM SIGGRAPH symposium on video games. ACM, New York, NY, USA, pp 151–159
- Zhang L (2009) Efficient motion planning using generalized penetration depth computation. PhD thesis, Computer Science Department, University of North Carolina at Chapel Hill
- Zhang X, Lee M, Kim YJ (2006) Interactive continuous collision detection for non-convex polyhedra. *Vis Comput* 22(9):749–760
- Zhang X, Wang D, Zhang Y, Xiao J (2011) Configuration-based optimization for six degree-of-freedom haptic rendering using sphere-trees. In: Proceedings of the IEEE International Conference on Intelligent Robot and Systems (IROS), pp 2602–2607
- Zilles CB, Salisbury JK (1995) A Constraint-based god-object method for haptic display. In: Proceedings of IEEE/RSJ international conference on intelligent robots and systems, Aug 1995

Haptic Rendering for Simulation of Fine Manipulation

Wang, D.; Xiao, J.; Zhang, Y.

2014, XII, 162 p. 127 illus., 108 illus. in color.,

Hardcover

ISBN: 978-3-662-44948-6