

Haptic Rendering of Thin, Deformable Objects with Spatially Varying Stiffness

Priyadarshini Kumari and Subhasis Chaudhuri

Vision and Image Processing Laboratory, Department of Electrical Engineering,
Indian Institute of Technology Bombay, Powai, Mumbai-400076
{priyadarshini,sc}@ee.iitb.ac.in

Abstract. In the real world, we often come across soft objects having spatially varying stiffness, such as human palm or a wart on the skin. In this paper, we propose a novel approach to render thin, deformable objects having spatially varying stiffness (inhomogeneous material). We use the classical Kirchhoff thin plate theory to compute the deformation. In general, the physics-based rendering of an arbitrary 3D surface is complex and time-consuming. Therefore, we approximate the 3D surface locally by a 2D plane using an area-preserving mapping technique - Gall-Peters mapping. Once the deformation is computed by solving a fourth-order partial differential equation, we project the points back onto the original object for proper haptic rendering. The method was validated through user experiments and was found to be realistic.

Keywords: Haptic rendering · deformable thin surface · spatially varying stiffness · Kirchhoff's thin plate theory · Gall-Peters projection

1 Introduction

Haptic perception becomes more realistic when we incorporate the physical and material properties of an object. For the kinesthetic rendering of rigid objects, it is sufficient to calculate only the force as they do not undergo any deformation under the applied force. However, the rendering of deformable objects is more complex since we need to calculate the force as well as the deformation. If the stiffness of the object varies over the surface, there is further complications as the rendered force is a function of both local deformation and local variation in stiffness. The continuity of deformation and a smooth movement of the proxy on the deforming surface are some of the key concerns while rendering a deformable object having a variable stiffness.

In the last two decades, haptic rendering has found several applications on interactions with the virtual environment, such as virtual museum[17], dental surgery simulation[10], and force modeling for needle insertion[14]. However, these interactions cannot handle skin or bowel simulator, which requires the object to be both deformable and have variable stiffness. The proposed method tries to achieve this special type of rendering.

We propose a method to render a point cloud data representing thin deformable objects having spatially varying stiffness such as skin and elastic sheets. First, the point cloud in the neighborhood of the proxy is locally fit into a hemispherical surface and using the parameter of the fitted sphere these points are then projected on a plane using Gall-Peters projection[7]. This being an area-preserving mapping, preserves the stiffness variation on the projected 2D plane. Subsequently, we use the Kirchhoff’s thin plate theory[18] to calculate the deformation due to an applied force at the proxy and for a given stiffness pattern. The solution is obtained by solving a fourth-order non-homogeneous Partial Differential Equation (PDE). Once the deformation map is estimated, it is back projected on the original 3D surface for proper rendering.

Most of the work done in the field of deformable object rendering can be classified into two categories: geometry-based deformation and physics-based deformation. Geometry based models are fast but do not necessarily focus on the physics involved in the deformation. On the other hand, physics-based rendering model is computationally expensive as the simulation of physical and material properties of the object is complex. In geometry-based modeling most of the methods use Gaussian[12] or a polynomial function to move the surrounding vertices around Haptic Interaction Point (HIP). Another widely practiced method is spline-based in which control points are assigned on the object, and these control points are manipulated to effect a smooth deformation [5], [11].

A FEM based method, such as [3] provides the most accurate rendering. However, this is computationally very demanding for haptic applications, requiring specialized hardware. It has been shown in [9] that by sacrificing the accuracy marginally, the computations can be speeded up by a factor of 10 by using a mass-spring model. However, the choice of dampers is very critical to ascertain the desired dynamic behavior, and usually, they are chosen heuristically. The purpose of the proposed method is to avoid these heuristics by solving a proper PDE over a 2D plate.

In smoothed-particle-hydrodynamics (SPH) approach[4], authors have introduced a technique that renders the object in different possible states (fluid, elastic, and rigid) in the same scene. However, in this technique, the stiffness of the entire object is varied as opposed to spatially varying the stiffness. Moreover in [4], a volume model of the object is used, while we use a 3D surface model which is comparatively much faster than the volume model.

The key contribution of our work is the development of an alternate, stable, physics-based haptic rendering method for an inhomogeneous, elastic deformable object.

2 Proposed Method

The input for our method is the point cloud representation of the 3D object. Since there is a one-to-one relationship between the point cloud and mesh data[6] (albeit the rendering process is a bit different), the algorithm can be extended to other forms of data representation. The point cloud is given by 4-tuple point

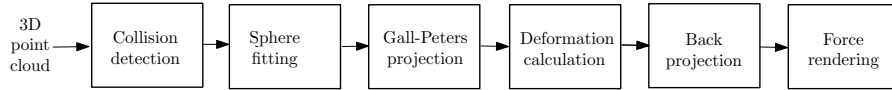


Fig. 1: Various steps involved in haptic rendering of a deformable object with variable stiffness.

$P_i = \{(x_i, y_i, z_i), d_i\}$, where (x_i, y_i, z_i) is the location of the i^{th} point and d_i is the corresponding stiffness. The overall algorithm can be broken into several stages as shown in Fig. 1. The various stages of algorithm are described in the following subsections.

2.1 Collision Detection

With point cloud data, we do not have surface normal defined at a point. In order to provide the appropriate force feedback to the user, we need the normal information at the collision point. We follow a proxy-based technique proposed in [16] for computation of normals. In this method, the normal is computed from the radial overshoot of each point inside the spherical proxy. Once the collision occurs, HIP penetrates the object, and the proxy is constrained to lie on the surface by using a dynamic function as defined in [16]. A collision is detected when the condition $(\mathbf{v}_n \cdot \mathbf{v}_h) < 0$ is satisfied, where \mathbf{v}_n is the computed surface normal at the point of collision and \mathbf{v}_h is the vector joining the proxy to HIP as shown in Fig. 2. Once the collision point is detected, we select the point cloud

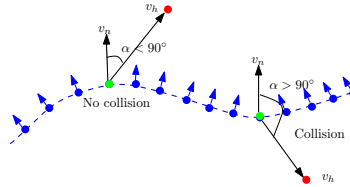


Fig. 2: Illustration of collision detection. The proxy and the HIP are shown in green and red colors, respectively. The point cloud is shown in blue. Blue arrows show the computed outward surface normals.

in the neighborhood of the point of collision for fitting a sphere locally.

2.2 Spherical Facet Modeling

We use the algebraic distance-based method proposed in [15] to fit the surface points in the neighborhood of the proxy. Let $f(x, y, z)$ be an equation of sphere in the algebraic form

$$\begin{aligned}
f(x, y, z) &= u_0(x^2 + y^2 + z^2) + u_1x + u_2y + u_3z + u_4 = 0 \\
\text{subject to } & u_1^2 + u_2^2 + u_3^2 - 4u_0u_4 = 1
\end{aligned} \tag{1}$$

where $\mathbf{u} = [u_0 \ u_1 \ u_2 \ u_3 \ u_4]^T$ represents the coefficients describing the sphere. The above mentioned constraint can be written in the matrix form as $\mathbf{u}^T \mathbf{C} \mathbf{u} = 1$ with \mathbf{C} as given in (2). So any point (x_i, y_i, z_i) lying on the surface of sphere will have an algebraic distance equal to zero. We minimize the sum of squared algebraic distances to estimate the best possible spherical fit \mathbf{u} . Although several possible constraints have been suggested in [2],[1], we use the constraint $\mathbf{u}^T \mathbf{C} \mathbf{u} = 1$, as Pratt [15] has shown that such a constraint has only one point of singularity of being a zero radius sphere. The resulting cost function can be expressed by introducing a Lagrangian multiplier λ as:

$$\underset{\mathbf{u}}{\text{minimize}} \left(\|\mathbf{D}\mathbf{u}\|^2 - \lambda(\mathbf{u}^T \mathbf{C} \mathbf{u} - 1) \right) \tag{2}$$

$$\mathbf{D} = \begin{bmatrix} x_1^2 + y_1^2 + z_1^2 & x_1 & y_1 & z_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^2 + y_n^2 + z_n^2 & x_n & y_n & z_n & 1 \end{bmatrix}, \mathbf{C} = \begin{bmatrix} 0 & 0 & 0 & 0 & -2 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ -2 & 0 & 0 & 0 & 0 \end{bmatrix}$$

where \mathbf{D} is an $n \times 5$ matrix derived from n points from the point cloud data, selected in the neighborhood of the collision point.

The solution is given by solving the generalized eigenvalue problem $\mathbf{D}^T \mathbf{D} \mathbf{u} = \lambda \mathbf{C} \mathbf{u}$, which can be solved using QZ algorithm developed by Moler and Stewart [13]. The center and the radius of the sphere can be calculated from the components of \mathbf{u} as given in [8]

$$\mathbf{c} = -\frac{1}{2u_4} [u_1 \ u_2 \ u_3], \quad r = \sqrt{c^T c - \frac{u_0}{u_4}}. \tag{3}$$

Once we get the center \mathbf{c} and the radius r of the sphere, all the points are locally projected on to a 2D plane as discussed in the next subsection.

All points, however, need not lie on the surface of the best fit sphere, as shown in Fig. 3a. In order to approximate the object surface by an area-preserving planar patch, we need to project all points on the surface of the fitted sphere. We use a projection technique in which we join all the points (which are not lying on the surface of the sphere) from the center. The joining line intersects the sphere at two points. The intersecting point nearest to the object point is chosen to be the projection of the corresponding point on the sphere. Figure 3b shows the best fit sphere to a set of object points near the proxy.

As it was mentioned earlier that deformation due to an arbitrary variation in material properties is easier to compute for a thin sheet, the fitted sphere is mapped onto a 2D plane. For an equi-curvature surface (like a sphere), the amount of energy required for deformation is proportional to the surface area

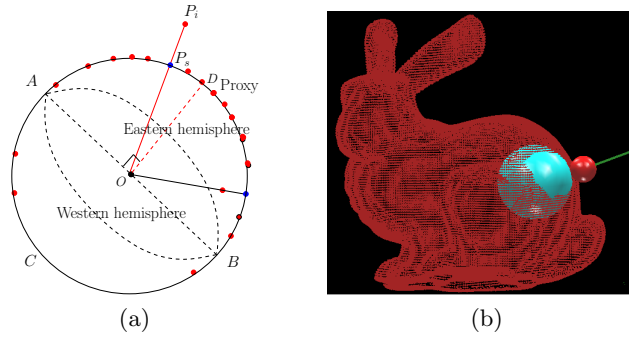


Fig. 3: (a) Illustration of projection of points on the surface of the best fit sphere and employing the constraint of hemisphere fitting. The object points P_i are shown in red, and the projected points P_s are shown in blue. (b) Illustration of the best fit sphere to the set of object points near the proxy. The proxy is shown with a red ball, and the green line points to the direction of normal at the collision point. (Data Courtesy: <http://graphics.stanford.edu/data/3Dscanrep/>)

when the stiffness is constant. Hence we require a projection operator $g : \mathbb{R}^3 \rightarrow \mathbb{R}^2$, which is area-preserving so that deformation computation is a function of stiffness alone. Gall-Peters projection [7] used in cartography is one such operator that preserves the area. Hence we use Gall-Peters projection in a way so that the proxy position lies at the center of the unfolded globe. While fitting the sphere, if there are points on the western hemisphere, these points are not visible at the proxy, and hence they are rejected, and only a hemispherical facet is considered. This is illustrated in Fig. 3a, where D is the proxy on the surface, AB corresponds to the plane joining the north pole (A) and the south pole (B) and the point D corresponds to $(0, 0)$ latitude and longitude of the sphere. Points lying on the western hemisphere (shown as the arc ACB) are discarded.

2.3 Area Preserving 3D to 2D Projection

Since Gall-Peters projection describes a technique to project the globe onto a plane, we have used it for a sphere to plane projection. The longitude and latitude of each point on the sphere is calculated from polar coordinates of the points. Considering the z -axis to be perpendicular to the computer screen, each point on the sphere is projected onto the x - z plane using (4).

$$x = \frac{r\alpha}{\sqrt{2}} \quad z = r\sqrt{2}\sin(\beta), \quad (4)$$

where α and β are longitude and latitude of the point in radians respectively and r is radius of the sphere. While using the Gall-Peters projection some issues emerge, when points lie either near the pole region or near the 180° longitude.

1. A point lying on the north/south pole gets stretched to a line on the plane after projection.
2. Points on 180° longitude get mapped on both the extreme of the 2D plane.

We avert such problem by shifting the coordinate axis of sphere so that our proxy (point D in Fig. 3a) falls in the region of zero latitude and zero longitude. After shifting the axes, we discard the points on the other half of the sphere opposite to the proxy (arc ACB in Fig. 3a). In this way, all neighborhood points around the proxy fall near the center of the plane, which would be very useful while employing an infinite horizon boundary condition in section 2.5. Once the points are projected on a plane, we compute the deformation of each point using Kirchhoff thin plate theory [18]. As mentioned earlier, computation of deformation on the planar sheet is easier than directly computing on the object surface. Therefore we compute the deformation of each point on 2D plane and subsequently project the points back on the original 3D object.

2.4 Deformation Computation on a Planar Sheet

In Kirchhoff theory of plate [18], the thickness of the plate is assumed to be very small as compared to the planar dimensions. In addition, some more assumptions are made to reduce the three dimensional problem to two dimensions. These assumptions are summarized as follow

1. The line normal to the neutral axis remain straight after deformation.
2. The normal stress in the direction of thickness is neglected.
3. Thickness of the plate does not change during bending.

Let s , w and t be the displacements of a point along x, y and z directions, respectively, and $|\mathbf{F}|$ is a distributed load on the planar sheet. Since the planar sheet is on the x-z plane, we can write the variation of s and t across thickness in terms of displacement w as $s = -y \frac{\partial w}{\partial x}$, $t = -y \frac{\partial w}{\partial z}$.

Taking all the assumptions into account, the deformation of a point is governed by the following PDE as explained in [18].

$$\frac{\partial^4 w}{\partial x^4} + 2 \frac{\partial^4 w}{\partial x^2 \partial z^2} + \frac{\partial^4 w}{\partial z^4} = \frac{|\mathbf{F}|}{D}, \quad \text{or} \quad \nabla^4 w = \frac{|\mathbf{F}|}{D}, \quad (5)$$

where ∇^4 is the biharmonic operator and D is flexural rigidity of the object. Interestingly, this equation was first obtained by Lagrange in 1811. The flexural rigidity is a material property of the object and is defined as

$$D = \frac{Eh^3}{12(1 - \nu^2)}, \quad (6)$$

where E , h , ν are the modulus of elasticity, thickness of the plate and Poisson's ratio (usually $0 < \nu < 0.3$), respectively. Equation (5) is valid only if material property D of the object is constant. In this case, computation of deformation

also becomes easier as the governing differential equation (5) is a standard biharmonic equation. However in real world, there are numerous objects which have spatially varying material properties when E and ν are functions of x and z . Equation (5) cannot be applied to compute deformation for such objects. Hence we move to the next stage where stiffness of the object is allowed to vary over the surface.

In order to compute the deformation in variable stiffness object we use the extended Kirchhoff thin plate theory[18]. The deformation of each point in such cases is governed by the following equation

$$\begin{aligned}
 & D\nabla^4 w + 2\frac{\partial D}{\partial x}\frac{\partial}{\partial x}(\nabla^2 w) + 2\frac{\partial D}{\partial z}\frac{\partial}{\partial z}(\nabla^2 w) + \nabla^2 D(\nabla^2 w) \\
 & - (1 - \nu) \left(\frac{\partial^2 D}{\partial x^2} \frac{\partial^2 w}{\partial z^2} - 2\frac{\partial^2 D}{\partial x \partial z} \frac{\partial^2 w}{\partial x \partial z} + \frac{\partial^2 D}{\partial z^2} \frac{\partial^2 w}{\partial x^2} \right) = |\mathbf{F}|.
 \end{aligned} \tag{7}$$

The detail derivation of (5) and (7) is explained in [18]. Unfortunately finding an analytical solution of this form of PDE is extremely difficult. Hence we solve the equation in discrete domain by using Jacobi iterative method for a given boundary condition. As the Gall-Peters projection uses non-linear mapping function (4), points are not projected evenly on a plane. Therefore we first sample the projected points on the uniform grid and subsequently we discretize all the variables and parameters of (7) by using central difference. The value of D at a grid node is taken as the value of D of the closest projected point from the grid node. Our initial condition on deformation is $w(i, j) = 0$. The final expression for iterative updating can be written in the form of a function $L()$:

$$w^{n+1}(i, j) = L(|F|, \mathbf{D}(i, j), w_N^n(i, j)) \tag{8}$$

where the superscript n denotes the n^{th} iteration and $w_N^n(i, j)$ refers to the various lattice entries in the neighborhood of the central lattice (i, j) , while $\mathbf{D}(i, j)$ is the given flexural rigidity map and its all derivatives (upto second order) at the lattice (i, j) . In each iteration, we update the w matrix and compare it with w matrix of previous iteration. We stop the iteration when the Frobenius norm of difference between current and previous w matrices becomes very small. The final w matrix gives the deformation at the grid point. However, one needs the deformation value at the location where the 3D point was originally projected using (4). The deformations at projected points are computed using bi-linear interpolation.

2.5 Boundary Conditions for PDE

As we mentioned earlier, in order to obtain the solution of the PDE given in (7), we need suitable boundary conditions. For simplicity, we assume edges of the projected plane to be parallel to the coordinate axis X and Z .

We need two boundary conditions at each edge. In our work, we assume all four edges of projected plate to be fixed (i.e deformation at infinite horizon is

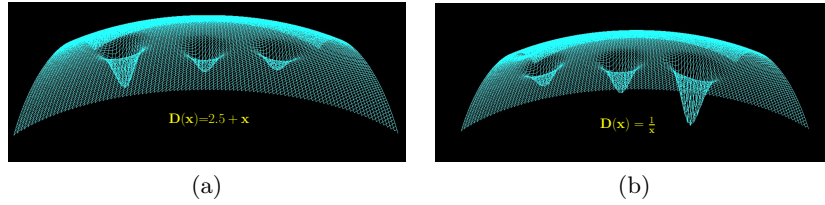


Fig. 4: Computed deformations on the object surface with different flexural rigidity functions under the same amount of applied force: (a) Linear, (b) Hyperbolic (x in cm). Deformation at various locations are superimposed together for ease of visualization.

zero). Hence the deflection and the first order derivative become zero at all four edges. After getting the deformation locally on the projected plane, we project back the points to compute the deformation on the original object for proper rendering. Points on the object are displaced in the direction of the force \mathbf{F} using the governing equation (9)

$$(9)$$

where \hat{P}_i is the final position of the i^{th} point on the deformed surface and P_i corresponds to the position before deformation with w_i being the deformation at the point.

3 Force Rendering

In order to haptically render the object, we first detect the collision of HIP with the object as discussed in section 2.1 and compute the force if a collision is detected. Once the collision is detected, the force needs to be fed back by the haptic device to provide the sensation of touch. Two important factors for force computation are its magnitude and its direction. The magnitude of force should be proportional to the penetration depth of HIP from the surface and its direction should be in the direction of the normal at the point of collision. Hence the reaction force is calculated using the expression, $F = -\frac{EA}{h}(|X_h - X_p|)$. Here X_h is the HIP position, X_p is the proxy position, A is the area of the plate and E is the modulus of elasticity at that location.

4 Results

The proposed method was implemented in Visual Studio 2010 in a Windows 7 platform with a Core(TM)2 Quad CPU Q8400 processor @ 2.66 GHz clock speed with 8 GB RAM. We use OpenGL 2.0 for graphic rendering and HAPI library for haptic rendering. We experimented with 3D point cloud model using Falcon haptic device from NOVINT. In order to make the rendering faster we segregate

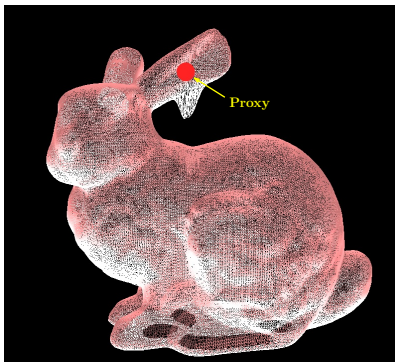


Fig. 5: Deformation shown on the bunny model.

the tasks of deformation computation, proxy updation and force computation through three different threads while programming. We observed that the deformation thread, as expected, runs relatively slowly due to iterative solution for equation (7). The average time required to run the deformation thread once is around 160 ms. We have set $v = 0.2$ in the study.

We initially use a simulated elliptical object consisting of 40000 points. The 3D grid used to sample the point cloud is of size $200 \times 200 \times 200$ with inter-node space of 0.025cm. Figure 4 shows the deformations at several points on the object surface with different flexural rigidity functions. In Fig. 4a, flexural rigidity is varied gradually along the horizontal direction while in Fig. 4b it is varied rapidly. We applied the same amount of force on the object surface and observed that deformation decreases as flexural rigidity increases as shown in Fig. 4a. It may be noted that the deformation is restricted to the neighborhood only due to the choice of boundary conditions. To explain this further, we experiment on a real object model and show the deformation at the ear lobe of a bunny in Fig. 5. One may expect the ear lobe to be bent due to the applied force like a cantilever. Instead one observes a dip within the ear lobe, exact shape of which would depend on the size of the chosen neighborhood.

We also verify the performance of the proposed algorithm in terms of user experience while interacting with various object models. We set up an experiment in which users were asked to interact with an object model (say Fig. 5) and rate the experience on a scale of 1 – 5, with 1 being very poor to 5 being excellent. Most of the users (80%) rated the experience as very good (rating 4) remaining 20% gave the rating of 3. When asked if they are able to perceive the change in stiffness, 90% users responded positively.

5 Conclusion

We have proposed a numerical method to solve the haptic rendering problem of elastic deformable objects with spatially varying stiffness. It is a physics-based

rendering technique and gives a realistic feeling of touching the object. We have not observed any divergence issue in the computation of the deformation map or the proxy movement over the deforming surface. In order to mathematically handle any object, an area-preserving map projection technique is used to transform an arbitrary 3D surface to a planar one following which Kirchhoff's thin plate deformation propagation model involving a fourth-order PDE is used to compute deformation in case of spatially varying stiffness. We also tested with some subjects and observed that the haptic-rendering technique using the proposed algorithm greatly augmented the user's experience. Our future work will involve the choice of alternate boundary conditions and extension to dealing with volumetric data.

References

1. Albano, A.: Representation of digitized contours in terms of conic arcs and straight-line segments. *Computer Graphics and Image Processing* 3(1), 23–33 (1974)
2. Bookstein, F.L.: Fitting conic sections to scattered data. *Computer Graphics and Image Processing* 9(1), 56–71 (1979)
3. Chan, S.C., Blevins, N.H., Salisbury, K.: Deformable haptic rendering for volumetric medical image data. In: *World Haptics Conference (WHC)*, 2013. pp. 73–78. IEEE (2013)
4. Cirio, G., Marchal, M., Otaduy, M., Lécuyer, A., et al.: Six-dof haptic interaction with fluids, solids, and their transitions. In: *World Haptics Conference (WHC)*, 2013. pp. 157–162. IEEE (2013)
5. Coquillart, S.: Extended free-form deformation: a sculpturing tool for 3D geometric modeling, vol. 24. ACM (1990)
6. Fabio, R., et al.: From point cloud to surface: the modeling and visualization problem. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences* 34(5), W10 (2003)
7. Gall, J.: Use of cylindrical projections for geographical astronomical, and scientific purposes. *The Scottish Geographical Magazine* 1(4), 119–123 (1885)
8. Guennebaud, G., Gross, M.: Algebraic point set surfaces. In: *ACM Transactions on Graphics (TOG)*. vol. 26, p. 23 (2007)
9. Hammer, P.E., Sacks, M.S., Pedro, J., Howe, R.D.: Mass-spring vs. finite element models of anisotropic heart valves: Speed and accuracy. In: *ASME 2010 Summer Bioengineering Conference*. pp. 643–644. American Society of Mechanical Engineers (2010)
10. Heiland, M., Petersik, A., Pflesser, B., Tiede, U., Schmelzle, R., Höhne, K.H., Handels, H.: Realistic haptic interaction for computer simulation of dental surgery. In: *International Congress Series*. vol. 1268, pp. 1226–1229. Elsevier (2004)
11. Hsu, W.M., Hughes, J.F., Kaufman, H.: Direct manipulation of free-form deformations. In: *ACM Siggraph Computer Graphics*. vol. 26, pp. 177–184. ACM (1992)
12. Luciano, C.J., Banerjee, P.P., Rizzi, S.H.: Gpu-based elastic-object deformation for enhancement of existing haptic applications. In: *IEEE International Conference on Automation Science and Engineering*. pp. 146–151. IEEE (2007)
13. Moler, C.B., Stewart, G.W.: An algorithm for generalized matrix eigenvalue problems. *SIAM Journal on Numerical Analysis* 10(2), 241–256 (1973)
14. Okamura, A.M., Simone, C., Leary, M.: Force modeling for needle insertion into soft tissue. *IEEE Transactions on Biomedical Engineering* 51(10), 1707–1716 (2004)

15. Pratt, V.: Direct least-squares fitting of algebraic surfaces. *ACM SIGGRAPH computer graphics* 21(4), 145–152 (1987)
16. Sreeni, K., Chaudhuri, S.: Haptic rendering of dense 3d point cloud data. In: *Haptics Symposium (HAPTICS)*. pp. 333–339. IEEE (2012)
17. Sreeni, K., Priyadarshini, K., Praseedha, A., Chaudhuri, S.: Haptic rendering of cultural heritage objects at different scales. In: *Haptics: Perception, Devices, Mobility, and Communication*, pp. 505–516. Springer (2012)
18. Ventsel, E., Krauthammer, T.: *Thin plates and shells: theory: analysis, and applications*. CRC press (2001)