이화여자대학교 대학원
2019학년도
석사학위 청구논문

Efficient Penetration Depth Computation for Deforming Tetrahedra using Object Norm

컴 퓨 터 공 학 과
Jisu Kim
2020

Efficient Penetration Depth Computation for Deforming Tetrahedra using Object Norm

이 논문을 석사학위 논문으로 제출함

2019 년 12 월

이화여자대학교 대학원

컴 퓨 터 공 학 과 Jisu Kim

Jisu Kim 의 석사학위 논문을 인준함

지도교수 김 영 준

심사위원 이 상 호

민 동 보

김 영 준

이화여자대학교 대학원

Table of Contents

List of Figures

List of Tables

Abstract

Soft object simulation has been an important research area in both computer graphics and robotics fields. In computer graphics, it is used to make a realistic and visually plausible animation of soft objects, such as tissues or cloths, and it is applied in computer animation, in interactive video games, or in virtual reality. In the robotics field, soft object manipulation, robot-assisted surgery, and bio-inspired soft robots are example of areas that require accurate deformable simulation.

Finite element method (FEM) is the most popular method employed to simulate the deformation of soft objects. It discretizes these objects into a network of finite elements (FEs), such as tetrahedra, and it calculates the deformation of each element based on the relationships between strain and internal stress imposed by external loads on the object. To apply accurate responsive forces, one must calculate the proper penetration metric for the deforming elements.

A penetration metric between intersecting objects is used in various fields to generate a robust movement of these objects in penalty-based simulation or to calculate contact forces as a haptic feedback in haptic rendering. However, a penetration metric for deformable objects are relatively less studied, whereas that for rigid objects are extensively investigated.

In this dissertation, we propose a novel penetration metric, called deformable penetration depth (PD_d), to define the measure of inter-penetration between two linearly deforming tetrahedra by using an object norm. First, we show that a distance metric for a tetrahedron deforming between two configurations can be found in closed form based on object norm. Then, we show that the PD_d between an intersecting pair of static and deforming tetrahedra can be determined by solving a quadratic programming (QP) problem of the distance metric with nonpenetration constraints over all possible separating directions. We also show that the PD_d between an intersecting pair of two deforming tetrahedra can be more efficiently determined by

solving a similar QP problem under some assumption on separating directions. We have implemented our algorithm on a standard PC platform using an off-the-shelf QP optimizer, and we experimentally show that both the static/deformable and deformable/deformable tetrahedra cases can be solvable in tens of milliseconds. To show the tightness of our distance metric, we have compared the metric results of PD_d against those of rigid penetration depth, and our results are tighter than those for the rigid case by a factor of three.

I. Introduction

A. Motivation

Physics-based simulation of soft objects has been widely investigated for many years in the fields of computer graphics and robotics. To generate a realistic deforming motion of soft objects, researchers in the field of computer graphics have suggested numerous deformable models [1], [2], and these models have been applied in various fields, such as in cloth [3] and hair [4] simulations, in computer animations, and in interactive video games. Surgical simulation [5] is another important application that requires an accurate simulation of deformable objects, particularly human body tissues. Since soft object deformation usually comes with dense contacts, including self-collisions, collision detection and response are also important considerations in order to achieve a physically plausible simulation..

In the robotics community, the handling of soft objects is increasingly becoming popular and important. For instance, soft object manipulation is recently gaining wide attention in order to achieve robust manipulation of real-world demand [6]. Besides grasping soft objects [7], other applications of deformable object manipulation include string insertion [8] and cloth folding [9]. Robotic surgery [10] is another huge research area that deals with tissue deformation. Robots not only can manipulate soft objects but can also be deformed by themselves. In bio-inspired robotics, compliant actuation is a must in order to mimic biological entities, resulting in the development of soft robots [11] (Figure 2). In all these areas, as well as in computer graphics fields, accurate simulation of soft object deformation with intensive contact is crucial to ensure manipulation robustness and to reduce the manufacturing cost of a robot.

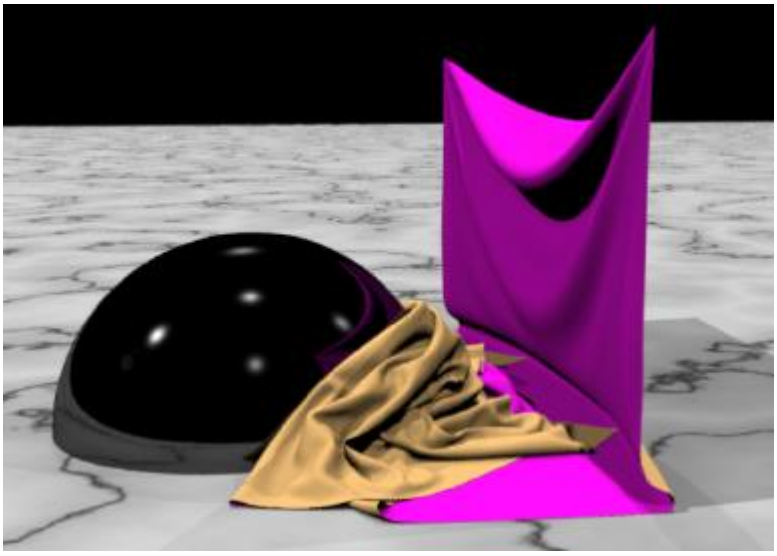


Figure 1. Cloth simulation [3]

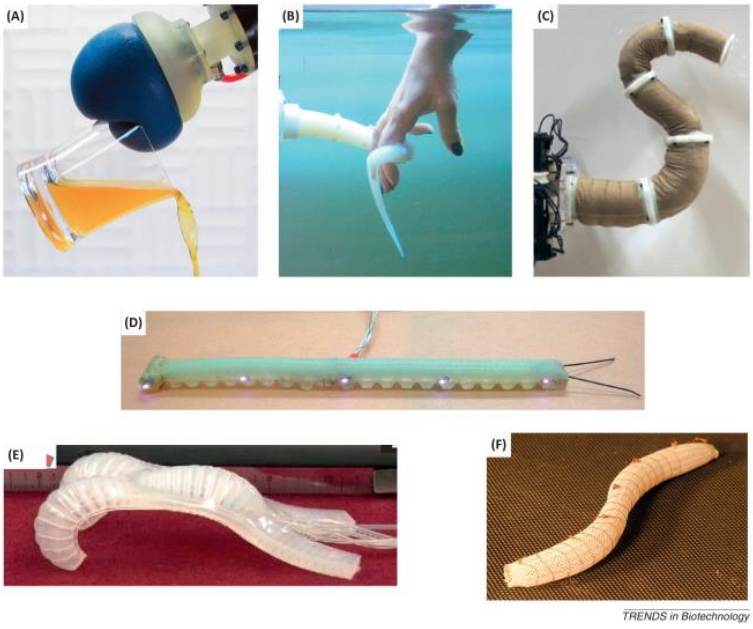


Figure 2. Examples of soft robots [11]

The finite element method (FEM) is a general method used to simulate deformable motion in soft objects with diverse material and structural properties, and it has been extensively studied for many decades in the area of computer-aided engineering, structural dynamics, computer animation, and soft robotics [12], [13]. Typically, FEM models soft objects with a network of many finite elements (FEs), such as tetrahedra. The deformation of these soft objects is calculated by solving a large system of equations based on constitutive laws, defining the relationship between strain and internal stress imposed by external loads on the object. Furthermore, for accurate simulation of contact dynamics using FEM, it is important to define a proper penetration measure between FEs so that proper responsive forces are applied in order to simulate their contact behavior.

Penetration metric is a measure of inter-penetration between objects and is often defined as the minimum distance that can separate intersected objects. Penetration metric is important in various fields. For instance, it is used to calculate the contact force between colliding objects in physics-based simulation [14]. In haptic rendering (Figure 4), responsive forces for haptic feedback is calculated from the penetration metric [15]. It is also used in retraction-based motion planning to find the contact configurations in a narrow passage [16].

The penetration metric for rigid objects has been extensively studied. The most popular metric is penetration depth (PD), which is the minimal translation required to resolve the penetration between rigid objects [17], [18]. In calculating the PD, exact method and approximation methods are suggested for both convex and non-convex models [19]. Moreover, translational PD is extended to a generalized PD [20], which uses rigid motion instead of translation to measure the distance between two configurations of objects. Unfortunately, there has been no rigorous formulation in the literature to define a penetration measure or metric for deformable FEs, particularly tetrahedra.

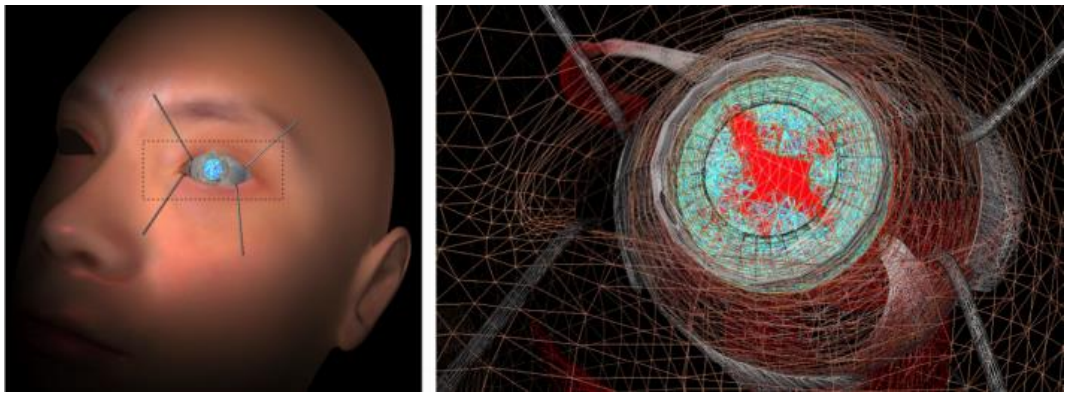


Figure 3. Cataract surgery simulation with FEM meshes [5]

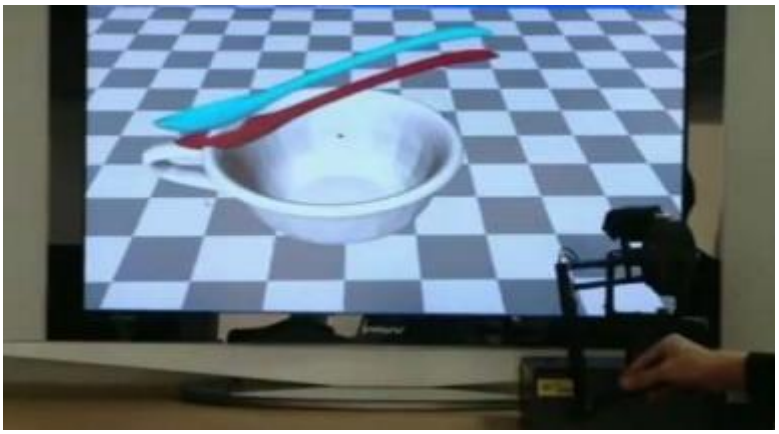


Figure 4. Haptic rendering [15]

B. Research Objectives

Our goal is to propose a novel penetration metric, called deformable penetration depth (PD_d), to define a measure of inter-penetration between two linearly deforming tetrahedra using object norm [21]. In this dissertation, we define PD_d as the minimal deformation that resolves the penetration between two linearly deforming tetrahedra. To calculate the PD_d, we need to address the following questions:

- How do we measure the amount of deformation for a tetrahedron?
- How do we set up non-penetration constraints between two tetrahedra?
- How do we calculate minimum deformation that satisfies such non-penetration constraints?

We provide a geometric (or kinematic) solution to this problem after decoupling it from the underlying dynamics. To make this problem simple and tractable while keeping the generality to a reasonable degree, we make the following assumptions about the deformation of a tetrahedron:

- **The tetrahedron is linearly deforming**: The displacement of an arbitrary point inside the tetrahedron can be calculated as a linear combination of the vertices' displacement, which is often used in conventional FEM simulation [12].
- **Separating direction is obtained from the rest configuration**: During deformation, the vertices used to calculate separating direction should always be on the plane parallel to that of the rest configuration.

Considering the assumptions above, we approach our problem by starting with a simple case wherein one tetrahedron is static and the other tetrahedron is deforming and then by generalizing the problem to the case involving two deforming tetrahedra. We also propose a

strategy on how to accelerate the entire computation.

C. Main Contributions

The main contribution of this dissertation are as follows:

- To define PD_d, we generalize the concept of object norm to a deformable case and show that a distance metric for a tetrahedron deforming between two configurations can be found in closed form.

- Case 1: We show that the PD_d between an intersected pair consisting of one static and one deforming tetrahedra can be determined by solving a quadratic programming (QP) problem over all possible separating directions in terms of the distance metric with non-penetration constraints.

- Case 2: We show that the PD_d between an intersected pair of deforming tetrahedra can be determined by solving a similar QP problem.

- Case 3: We approximate case 2 through a rigid PD calculation based on the separating axis theorem, and we demonstrate that this computation can be accelerated by an order of magnitude compared with case 2 while keeping the approximation error to less than 5%.

- We compare the metric results of our PD_d against those of rigid PD; our results are three times tighter than the rigid case.

D. Organization

The rest of this dissertation is organized as follows. In the next section, we briefly present relevant works on PD computation. In Sec. III, we present a closed-form solution of our penetration metric and subsequently apply it to obtain the PD_{d} for static versus deformable (Sec. IV) and deformable versus deformable tetrahedral cases (Sec. V). In Sec. VI, we demonstrate that these solutions can be obtained more rapidly. In Sec. VII, we present various experimental results of our PD algorithms. We present our conclusions in Sec. VIII.

II. Related Works

A. Penetration Metric for Rigid Objects

Various penetration metrics for rigid objects have been suggested in the literature [19]. Translational PD is the most well-known metric and is defined by a minimal distance that separates two intersecting objects [17], [18]. They proposed an algorithm to compute exact PD using the Minkowski sum. However, the time complexity in computing the entire Minkowski sum is quite high; $O(n^2)$ for convex models and $O(n^6)$ for non-convex models, where n is the number of polygons in the models. For convex models, approximation methods have been proposed to reduce the computation time [22], [23]. For non-convex models, the use of exact PD [24] and approximations [14], [25], [26], [27] and real-time algorithms [28] is suggested. Moreover, Kim et al. [29] presented a hybrid algorithm that combines local optimization with machine learning.

Since the penetration metric is commonly used to compute the responsive forces in contact dynamics, the continuity of the metric in terms of direction and magnitude must be considered; the Phong projection [30] and the dynamic Minkowski sums [31] are used to compute continuous PD (Figure 5). Both the positive and negative distances (i.e., PD) and the minimal distance value over a continuous time interval can be computed [32]. The works of Weller and Zachmann [33], Allard et al. [34], Wang et al. [35] use penetration volume as a continuous penetration metric. Also, Nirel and Lischinski [36] present a volume-based global collision resolution method. The generalized penetration depth (PD_g) (Figure 6) is defined as a minimal rigid transformation through which interpenetrating objects must undergo to resolve the penetration [20]. Pointwise PD is defined as the distance between the deepest intersecting points of two objects [37].

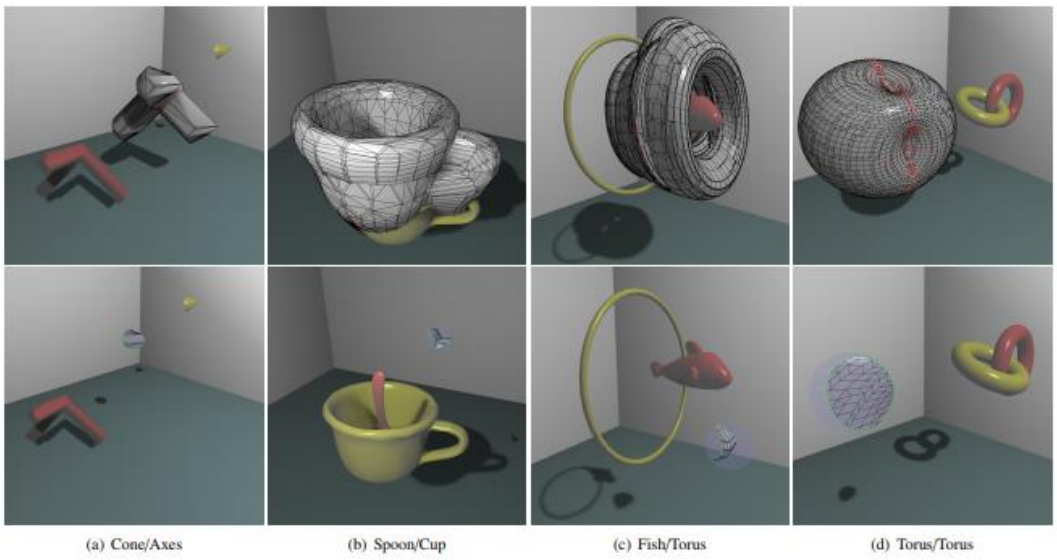


Figure 5. Full Minkowski sum (top) and local Minkowski sum (bottom) [31]

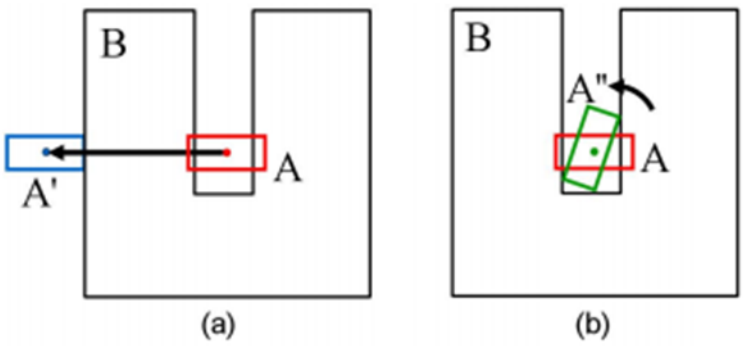


Figure 6. Generalized penetration depth [20]

B. Penetration Metric for Deformable Objects

The computation of penetration metric (or PD) for deformable objects is relatively less studied compared with that for rigid objects. In particular, no attempt has been made to rigorously define such a metric, and thus our work tackles this problem.

Distance field-based representations are often used to compute penalty forces for deformable objects [38]. Distance fields contain the minimal distance of each point in the embedding space to the object surface. As the object deforms, distance fields should be updated, and the updating can be accelerated using graphics processing units (GPUs) [39], [40]. Figure 7 shows the deformed distance field inside an object before and after deformation. Heidelberger et al. [41] has improved the consistency of PD by using a propagation scheme. There are a few FEM-based methods that calculate penetration metrics. Hirota et al. [42] suggested the use of material depth, an approximation of distance field, which is the distance between the interior and object–boundary points. The energy-based method [43] also uses a signed distance in material space to compute the PD.

Layered depth images (LDIs) [44] are a data structure representing multiple layers of geometry rendered from a fixed viewpoint (Figure 8). Heidelberger et al. [45], [46] suggested a method to estimate penetration volume using pixel depths and normals from LDIs. The methods reported by Faure et al. [47] uses a surface rasterization method based on LDIs [34], [48] to compute the repulsive forces (Figure 9). However, most of these existing works are heuristically defined and more importantly, they do not guarantee full separation of intersecting objects. In other words, all these metrics provide only the lower bound of PD and not the upper bound, unlike our method.

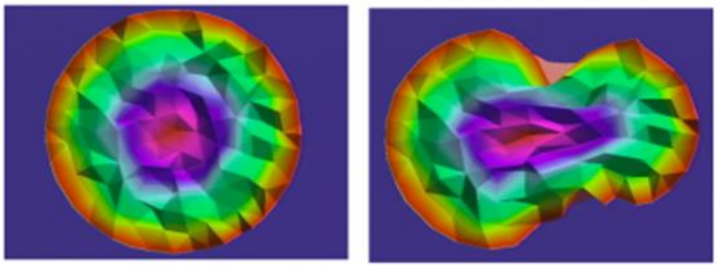


Figure 7. Deformed distance fields [38]

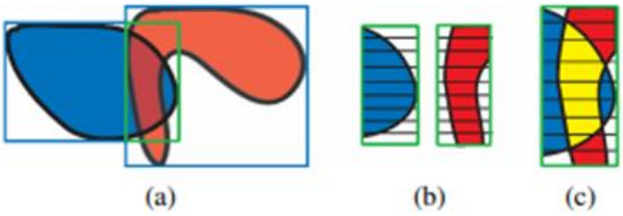


Figure 8. Layered depth images [48]

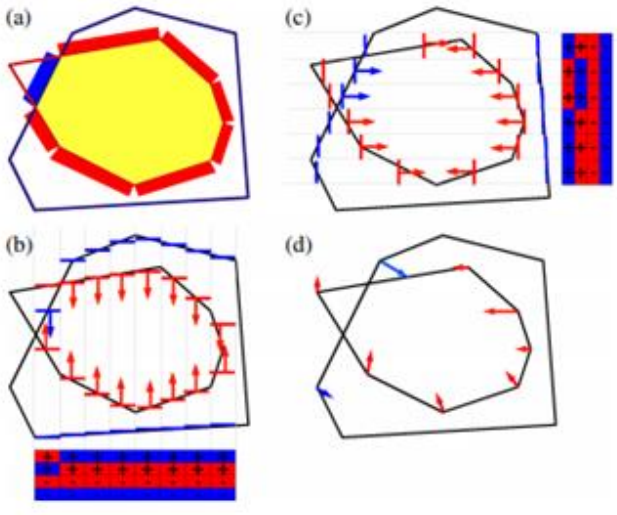


Figure 9. Repulsive force computation using layered depth images [48]

III. Metric Formulation

In this section, we present the $\mathrm{PD_d}$ by defining the closed-form of distance metric for deformable objects using object norm and then demonstrate that $\mathrm{PD_d}$ is the generalization of the translational PD for rigid objects.

A. Closed-Form of Distance Metric for Deformable Objects

Given a tetrahedron $\mathcal{T} \in \mathbb{R}^3$ linearly deforming between the rest $\mathbf{q}_0 \in \mathbb{R}^{12}$ and the deformed configurations $\mathbf{q}_1 \in \mathbb{R}^{12}$, an arbitrary point $\mathbf{r} \in \mathcal{T}(\mathbf{q}_0)$ and its counterpart $\mathbf{p} \in \mathcal{T}(\mathbf{q}_1)$ can be represented by a barycentric coordinate $\mathbf{b} = (b_1, b_2, b_3)$, which is constant during deformation [13] (also illustrated in Figure 10):

$$\mathbf{r} = \mathbf{r}_0 + (\mathbf{r}_1 - \mathbf{r}_0)b_1 + (\mathbf{r}_2 - \mathbf{r}_0)b_2 + (\mathbf{r}_3 - \mathbf{r}_0)b_3$$
$$\mathbf{p} = \mathbf{p}_0 + (\mathbf{p}_1 - \mathbf{p}_0)b_1 + (\mathbf{p}_2 - \mathbf{p}_0)b_2 + (\mathbf{p}_3 - \mathbf{p}_0)b_3.$$

Then, the distance metric $\sigma(\mathbf{q}_0, \mathbf{q}_1)$ using object norm [21], [49] can be formulated as follows:

$$\begin{aligned}\sigma(\mathbf{q}_0, \mathbf{q}_1) &= \frac{1}{V}\int \|\mathbf{p} - \mathbf{r}\|^2 dV \\ &= 6\int \|\mathbf{D}\mathbf{b} + \mathbf{d}_0\|^2 d\mathbf{b},\end{aligned} \tag{1}$$

where $V = \frac{1}{6}$ is the volume of $\mathcal{T}$ in terms of the barycentric coordinate, $\mathbf{d}_i = (\mathbf{p}_i - \mathbf{r}_i)$, and $\mathbf{D} \in \mathbb{R}^{3\times3} = [\mathbf{d_1} - \mathbf{d_0} | \mathbf{d_2} - \mathbf{d_0} \mid \mathbf{d_3} - \mathbf{d_0}]$. Equation 1 induces a new tetrahedron $\mathcal{T}_d$, called *displacement tetrahedron* (Figure 11), which has four vertices $\mathbf{d}_i = (\mathbf{p}_i - \mathbf{r}_i), i = 0, \cdots, 3$, and

the object norm is the sum of the squared distances from $\mathbf{d_0}$ for $\forall \mathbf{d} \in \mathcal{T}_d$.

Let $\mathbf{d}_i = (x_i, y_i, z_i)^T$. Then, the closed-form solution of the proposed metric can be calculated as follows:

$$\sigma(\mathbf{q}_0, \mathbf{q}_1) = \frac{1}{10} \sum_{\forall i \geq j \in \{0,\ldots,3\}} \mathbf{d}_i^T \mathbf{d}_j \tag{2}$$

Let $\mathbf{I}_d$ be the inertia tensor of the displacement tetrahedron $\mathcal{T}_d$. Then, the moments of inertia of $\mathbf{I}_d, \mathbf{I}_{xx}, \mathbf{I}_{yy}, \mathbf{I}_{zz}$, with constant mass density μ_d are as follows:

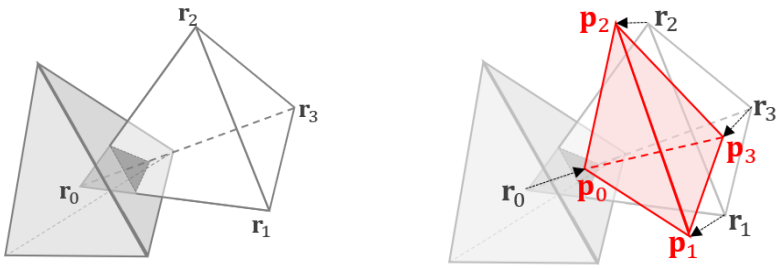
$$\mathbf{I}_{xx} = \frac{1}{60} \mu_d |det(\mathbf{D})| \sum_{\forall i \geq j} \mathbf{d}_i^T [\mathbf{0}\ \mathbf{e_2}\ \mathbf{e_3}] \mathbf{d}_j$$

$$\mathbf{I}_{yy} = \frac{1}{60} \mu_d |det(\mathbf{D})| \sum_{\forall i \geq j} \mathbf{d}_i^T [\mathbf{e_1}\ \mathbf{0}\ \mathbf{e_3}] \mathbf{d}_j$$

$$\mathbf{I}_{zz} = \frac{1}{60} \mu_d |det(\mathbf{D})| \sum_{\forall i \geq j} \mathbf{d}_i^T [\mathbf{e_1}\ \mathbf{e_2}\ \mathbf{0}] \mathbf{d}_j,$$

where $\mathbf{e}_i$ is the standard basis for $\mathbb{R}^3$. Since $|det(\mathbf{D})| = 6V_d = 1$, where V_d is the volume of $\mathcal{T}_d$ (which is always $\frac{1}{6}$), then

$$\sigma(\mathbf{q_0}, \mathbf{q_1}) = \frac{3}{\mu_d} \left(\mathbf{I}_{xx} + \mathbf{I}_{yy} + \mathbf{I}_{zz}\right) = \frac{3}{\mu_d} \mathrm{tr}(\mathbf{I}_d) \tag{3}$$



(a) Rest configuration $\mathbf{q}_0$ (b) Deformed configuration $\mathbf{q}_1$

Figure 10. Linear deformation of a tetrahedron that resolves inter-penetration

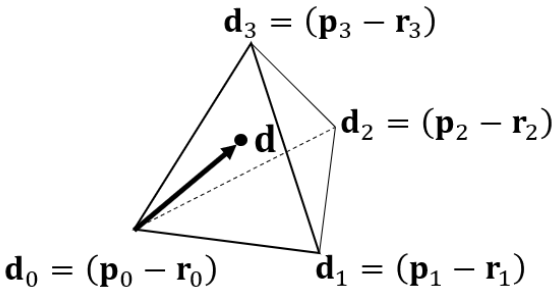


Figure 11. Displacement tetrahedron

B. Definition of $\mathrm{PD_d}$

We use Equation 2 or 3 to define the $\mathrm{PD_d}$ as follows: For an intersecting pair of linearly deforming tetrahedra, $\mathcal{T}_1, \mathcal{T}_2$, with corresponding initial configurations $\mathbf{q}_0^{\mathcal{T}_1}, \mathbf{q}_0^{\mathcal{T}_2}$, their $\mathrm{PD_d}$ is

$$\mathrm{PD_d}\left(\mathcal{T}_1\left(\mathbf{q}_0^{\mathcal{T}_1}\right), \mathcal{T}_2\left(\mathbf{q}_0^{\mathcal{T}_2}\right)\right) = \min_{\left(\mathbf{q}_1^{\mathcal{T}_1}, \mathbf{q}_1^{\mathcal{T}_2}\right) \in \mathcal{C}} \sqrt{\sigma\left(\mathbf{q}_0^{\mathcal{T}_1}, \mathbf{q}_1^{\mathcal{T}_1}\right) + \sigma\left(\mathbf{q}_0^{\mathcal{T}_2}, \mathbf{q}_1^{\mathcal{T}_2}\right)} \tag{4}$$

where $\mathcal{C} \subset \mathbb{R}^{24}$is the contact (configuration) space imposed by $\mathcal{T}_1, \mathcal{T}_2$. In Sec. IV, we present our $\mathrm{PD_d}$ computation algorithm when $\mathcal{T}_1$ is static; that is, $\mathbf{q}_0^{\mathcal{T}_1} = \mathbf{q}_1^{\mathcal{T}_1}$, and this restriction is relaxed in Sec. V by allowing both $\mathcal{T}_1, \mathcal{T}_2$ to be deformable.

C. Generalization of Rigid PD

Note that the proposed $\mathrm{PD_d}$ is a generalization of the classical rigid (or translational) PD [17], [18]. Since the object norm in Equation 3 can be interpreted as an average displacement of all the points inside a deformable object during deformation when the object is purely translated by $\mathbf{d}$, the object norm will be equivalent to the squared norm of the translation vector $\mathbf{d}$ from Equation 2, that is,

$$\sigma = \frac{1}{10}\sum \mathbf{d}_i^{\mathrm{T}}\mathbf{d}_i = \mathbf{d}_i^{\mathrm{T}}\mathbf{d}_i = \|\mathbf{d}\|^2 \tag{5}$$

Thus, according to Equation 4, $\mathrm{PD_d} = \mathrm{PD}$ when $\mathbf{q} \in \mathrm{SE}(3)$.

IV. Static versus Deforming Tetrahedra

In this section, we solve the optimization problem of Equation 4 when $\mathcal{T}_1$ is static but $\mathcal{T}_2$ is still deformable. Figure 12 shows the possible contact cases after penetration resolution, as follows: face–vertex (FV), vertex–face (VF), or edge–edge (EE) contacts, where $\{\mathbf{s}_i\} \subset \mathcal{T}_1$ is a static tetrahedron and $\{\mathbf{p}_i\} \subset \mathcal{T}_2$ is a deforming tetrahedron. Since the contact space $\mathcal{C}$ in Equation 4 can be realized by three cases, the normal vector of the contact plane is used as a direction to separate $\mathcal{T}_1, \mathcal{T}_2$, which can also minimize Equation 4. According to the separating axis theorem (SAT) [50], the total number of possible separating directions that we need to consider is 44 given that there are 8 VF (or FV) and 36 EE contact pairs between the two tetrahedra.

In general, when both $\mathcal{T}_1, \mathcal{T}_2$ are deforming, even for linear deformation, the separating directions can be non-linear, making Equation 4 difficult to solve. In our work, to make the problem tractable, we assume that the separating directions are always obtained from the rest configurations $\mathbf{q}_0^{\mathcal{T}_1}, \mathbf{q}_0^{\mathcal{T}_2}$ and thus are constant. Note that this is a reasonable assumption unless objects deform severely, which is reasonable for most practical robotic applications. A vertex in $\mathcal{T}_1$ or $\mathcal{T}_2$ is considered constrained if they are involved in calculating the separating direction (i.e., the contact normal); otherwise, consider it free. For example, if a face normal of $\mathcal{T}_1$ (e.g., $\mathbf{n}_{\mathrm{FV}}$ in Figure 12(a)) is selected as a separating direction, the vertices incident to the face (e.g., $\mathbf{s}_0, \mathbf{s}_1, \mathbf{s}_2,$ in Figure 12(a)) are constrained. After optimization, free vertices may or may not be on the contact plane, but constrained vertices are always on the plane.

A. Separating Direction

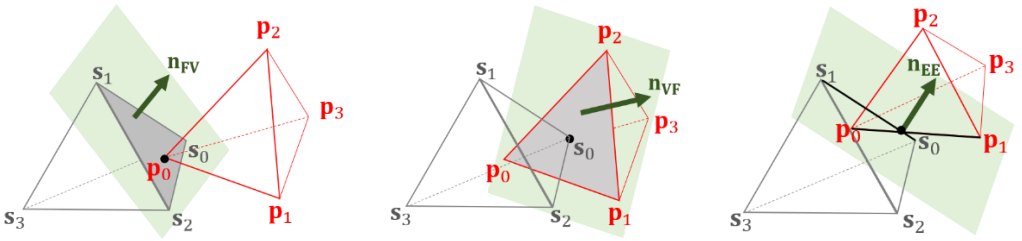
Let $\{\mathbf{s}_i\} \subset \mathcal{T}_1$, $\{\mathbf{r}_i\} \subset \mathcal{T}_2\left(\mathbf{q}_0^{\mathcal{T}_2}\right), \{\mathbf{p}_i\} \subset \mathcal{T}_2\left(\mathbf{q}_1^{\mathcal{T}_2}\right), i = 0, \cdots, 3$ be the four vertices of $\mathcal{T}_1, \mathcal{T}_2\left(\mathbf{q}_0^{\mathcal{T}_2}\right)$, $\mathcal{T}_2\left(\mathbf{q}_1^{\mathcal{T}_2}\right)$, respectively. We first calculate the constant normal vector of a separating plane using the rest configurations. Let $\mathbf{n}$ be such a normal vector for each contact state. Then the following is the result of normal vector calculation:

$$\begin{aligned} \mathbf{n}_{\mathrm{FV}} &= (\mathbf{s}_1 - \mathbf{s}_0) \times (\mathbf{s}_2 - \mathbf{s}_0) \\ \mathbf{n}_{\mathrm{VF}} &= (\mathbf{r}_1 - \mathbf{r}_0) \times (\mathbf{r}_2 - \mathbf{r}_0) \\ \mathbf{n}_{\mathrm{EE}} &= (\mathbf{r}_1 - \mathbf{r}_0) \times (\mathbf{s}_1 - \mathbf{s}_0) \end{aligned} \tag{6}$$

Furthermore, to make the separation direction consistent, the direction is decided based on the following rules:

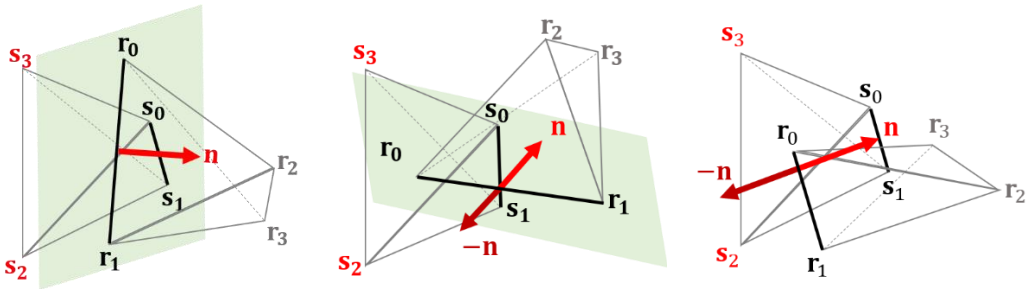
1) $\mathbf{n}_{\mathrm{FV}}$ should be outward from $\mathcal{T}_1$.
2) $\mathbf{n}_{\mathrm{VF}}$ should be inward to $\mathcal{T}_2$.
3) $\mathbf{n}_{\mathrm{EE}}$ should point away from the non-contacting vertices of $\mathcal{T}_1$ (e.g., $\mathbf{s_2}, \mathbf{s_3}$ in Figure 12(c)).

Note that the third case for $\mathbf{n}_{\mathrm{EE}}$ works only when the non-contacting vertices of $\mathcal{T}_1$ lie in the same half-space (Figure 13(a)). If this is not the case (Figure 13(b)), both directions are tested to minimize Equation 4. There also exist degenerate cases for $\mathbf{n}_{\mathrm{EE}}$ when two contacting EE pairs are parallel to each other. In this case, the normal is calculated from the shortest distance vector between the two EE pairs (Figure 13(c)). When the two EE pairs are co-linear, the separating direction can be any of the vectors perpendicular to the edge. Thus, we can use the normal directions of the faces incident to the edges as a candidate separating direction, which is redundant as it can be covered by the VF or FV cases.



(a) FV contact (b) VF contact (c) EE contact

Figure 12. Three contact cases after penetration resolution.



(a) Normal (b) Undecidable (c) Parallel

Figure 13. Various separating directions $\mathbf{n}$ for an EE contact

B. Non-penetration Constraints

Equation 4 is essentially a constrained optimization problem wherein the constraints are non-penetration constraints. Due to the constant assumption on the separating directions employed in Sec. IV-A, we can now linearize the non-penetration constraints and thus set up a solvable QP problem afterwards. Specifically, we write the non-penetration constraints for each contact case as follows:

1) FV case (Figure 12(a)): since all the vertices s_i of the static tetrahedron $\mathcal{T}_1$ lie in the same half space, we simply impose non-penetration constraints for the vertices $\mathbf{p}_i$ of the deforming tetrahedron $\mathcal{T}_2$ as below, where $\boldsymbol{s}_0$ is the vertex defining the FV contact:

$$\mathbf{n}_{\text{FV}} \cdot (\mathbf{p}_i - \mathbf{s}_0) \geq 0 \tag{7}$$

2) VF case (Figure 12(b)): with constrained vertices $(\mathbf{p}_k, k = 0, \cdots, 2)$ of $\mathcal{T}_2$ and free vertices $\mathbf{s}_j, j = 0, \cdots, 3$ of $\mathcal{T}_1$, the free vertex $\mathbf{p}_3$ and the constrained vertex $\mathbf{p}_0$ on the plane, we formulate seven non-penetration constraints, as follows:

$$\begin{aligned} \mathbf{n}_{\text{VF}} \cdot (\mathbf{p}_k - \mathbf{p}_0) &= 0, \\ \mathbf{n}_{\text{VF}} \cdot (\mathbf{s}_j - \mathbf{p}_0) &\leq 0, \\ \mathbf{n}_{\text{VF}} \cdot (\mathbf{p}_3 - \mathbf{p}_0) &\geq 0, \end{aligned} \tag{8}$$

where $k \in \{0, \ldots, 2\}, j \in \{0, \ldots, 3\}$.

3) EE case (Figure 12(c)): with constrained vertices $(\mathbf{s}_0, \mathbf{s}_1, \mathbf{p}_0, \mathbf{p}_1)$, the free vertices $(\mathbf{s}_2, \mathbf{s}_3)$ should lie in the same half space, and free vertices $(\mathbf{p}_2, \mathbf{p}_3)$ should lie in the other half space. Without loss of generality, assuming that $\mathbf{p}_0$ is on the separating plane, we can formulate seven

constraints.

$$\begin{aligned} \mathbf{n}_{\mathrm{EE}} \cdot (\mathbf{s}_k - \mathbf{p}_0) &= 0 \\ \mathbf{n}_{\mathrm{EE}} \cdot (\mathbf{p}_1 - \mathbf{p}_0) &= 0 \\ \mathbf{n}_{\mathrm{EE}} \cdot (\mathbf{s}_j - \mathbf{p}_0) &\leq 0 \\ \mathbf{n}_{\mathrm{EE}} \cdot (\mathbf{p}_j - \mathbf{p}_0) &\geq 0 \end{aligned} \tag{9}$$

where $k \in \{0,1\}, j \in \{2,3\}$.

C. Constrained Optimization

Considering that we have now set up both an objective function (Equation 3) and non-penetration constraints for each contact case (Equations 7–9), we now solve the constrained optimization in Equation 4. Given that the objective function is quadratic and the constraints are linear, our problem is a QP problem. This problem is solvable given that the objective function is semi-positive definite and that the constraint space is convex. We use an off-the-shelf QP solver, such as Gurobi Optimizer [51], to efficiently solve this problem.

V. Deforming versus Deforming Tetrahedra

In this section, we extend the optimization problem of Equation 4 to the case where both $\mathcal{T}_1$ and $\mathcal{T}_2$ are deforming.

A. Non-penetration Constraints

The non-penetration constraints used in this section require a new variable, $\mathbf{p}_s \in \mathbb{R}^3$, which is a point on the separating plane that decides on the position of the plane. For the static and deforming tetrahedron case, the position of a separating plane is automatically decided once we have determined the separating direction since there will be at least one vertex in the static tetrahedron involved in contact. But now, not only the configurations of both tetrahedra but also the position of the separating plane should be optimized at the same time because there is no static vertex to determine the position of it .

When $\mathbf{p}_s$ is introduced, it becomes easier to write the constraints since we do not need to consider which vertex to select as a point on the plane. As illustrated in Figure 14, the constrained vertices (red) should lie on the separating plane and the free vertices (black) should lie on either side of the plane while $\mathbf{p}_s$ constrains the location of the separating plane. The only difference between each contact case is the separating direction and the set of constrained vertices, namely, $\mathcal{C}_s$ for $\mathcal{T}_1$ and $\mathcal{C}_p$ for $\mathcal{T}_2$; for instance, $\mathcal{C}_s = \{\mathbf{s}_0, \mathbf{s}_1, \mathbf{s}_2\}, \mathcal{C}_p = \emptyset$ in Figure 14(a) and $\mathcal{C}_s = \{\mathbf{s}_0, \mathbf{s}_1\}, \mathcal{C}_p = \{\mathbf{p}_0, \mathbf{p}_1\}$ in Figure 14(b). Once they are decided, we can write the non-penetration constraints in a more general form than those presented in Sec. IV. When a separating direction $\mathbf{n}$ for each contact case is chosen based on the corresponding constrained vertices $\mathbf{s}_c \in \mathcal{T}_1$, $\mathbf{p}_c \in \mathcal{T}_2$ and on the free vertices $\mathbf{s}_f \in \mathcal{T}_1$, $\mathbf{p}_f \in \mathcal{T}_2$ of each deforming

tetrahedron $\mathcal{T}_1$, $\mathcal{T}_2$, the eight non-penetration constraints can be formulated as follows:

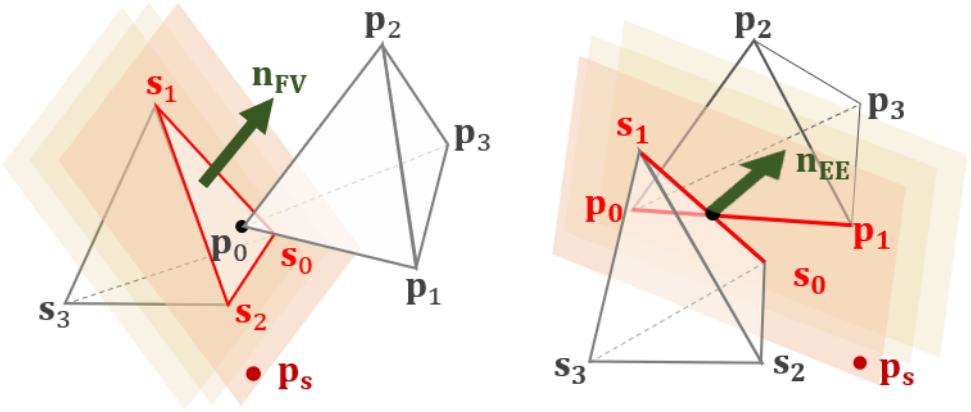
$$\begin{aligned}
\mathbf{n} \cdot (\mathbf{s}_c - \mathbf{p}_s) &= 0, \forall \mathbf{s}_c \in \mathcal{C}_s, \\
\mathbf{n} \cdot (\mathbf{p}_c - \mathbf{p}_s) &= 0, \forall \mathbf{p}_c \in \mathcal{C}_p, \\
\mathbf{n} \cdot (\mathbf{s}_f - \mathbf{p}_s) &\leq 0, \forall \mathbf{s}_f \in \mathcal{F}_s, \\
\mathbf{n} \cdot (\mathbf{p}_f - \mathbf{p}_s) &\geq 0, \forall \mathbf{p}_f \in \mathcal{F}_p,
\end{aligned} \tag{10}$$

where $\mathcal{F}_s$, $\mathcal{F}_p$ are the set of free vertices for $\mathcal{T}_1$, $\mathcal{T}_2$, respectively.

The non-penetration constraints for the static/deforming tetrahedron in Sec. IV can be viewed as a special case of this form, where $\mathbf{s}_\mathrm{c}$ and $\mathbf{s}_f$ are fixed and $\mathbf{p}_s$ is chosen among the constrained vertices. For example, in FV case, since $\mathcal{C}_p = \emptyset$, the second constraint in Equation 10 can be ignored. Moreover, as $\mathbf{p}_s$ is chosen from $\mathcal{C}_s$, the first and third constraints in Equation 10 are automatically satisfied, and the rest of the four constraints remain just like in Equation 7. The VF case (Equation 8) and EE case (Equation 9) have seven constraints because the chosen constrained vertex removes one of the equality constraints; for example, since $\mathbf{p}_s = \mathbf{p}_0 \in \mathcal{C}_p$, $\mathbf{n} \cdot (\mathbf{p}_0 - \mathbf{p}_0) = 0$.

B. Constrained Optimization

Similar to that in the previous section, PD_d can be calculated by solving the QP problem of Equation 4. Note that the non-penetration constraints in Equation 10 are still linear since the separating directions are constant.



(a) FV or VF case (b) EE case

Figure 14. Constrained vertices (red) and free vertices (black) defined by $\mathbf{p_s}$

VI. Acceleration Technique

In order to compute the PD_d, we need to solve a QP problem with constraints, each of which is associated with a separation direction; the entire solution typically takes tens of milliseconds to be completed with the use of an off-the-shelf QP solver. However, if we can pre-select a set of candidate directions that possibly includes the optimal separating direction, we can significantly reduce the overall computation time. Under the hypothesis that the deformation of soft objects is not severe, we assume that the penetration (equivalently separating) direction for deformable objects is close to the penetration direction when the objects behave rigidly. Moreover, in our experiment, we find that, on average, 52.33% of the results obtained by rigid PD coincides with the optimal direction after running full optimization on PD_d.

To leverage this observation and to accelerate PD_d computation, we calculate the rigid PD based on SAT [50] and feed it to the optimization problem in Equation 4; this approach is equivalent to using a single contact constraint in Equation 4.

A. Rigid PD Calculation using SAT

Rigid PD can be determined through various methods, such as using the GJK algorithm or Minkowski sums [17], [18]. However, we choose to use a SAT-based algorithm since our PD_d algorithm can be considered as a general case of PD and can be implemented similar to that in [52]. In this section, we show that the PD between two intersected tetrahedra can be calculated using the SAT. Specifically, we prove that the PD is equal to the smallest overlapping length projected over all possible separating axes in the SAT.

Definition 1. The projected length $L_{\mathbf{n}}(\mathcal{T})$ of a given simplicial complex $\mathcal{T}$ along an axial direction n is defined as follows:

$$L_{\mathbf{n}}(\mathcal{T}) = \max\left(\{\left|\left(\mathbf{x}_i - \mathbf{x}_j\right) \cdot \mathbf{n}\right| \left|\forall \mathbf{x}_i, \mathbf{x}_j \in \mathcal{T}\right\}\right) \tag{11}$$

where $\mathbf{x}_i, \mathbf{x}_j$ are the vertices in $\mathcal{T}$. Then, the pair of vertices $\mathbf{x}_i, \mathbf{x}_j$ that realizes the projected length $L_{\mathbf{n}}(\mathcal{T})$ is called the supporting vertices for $\mathbf{n}$.

Theorem 1. The PD of two intersected tetrahedra $\mathcal{T}_1, \mathcal{T}_2$ can be calculated as follows:

$$\mathrm{PD}(\mathcal{T}_1, \mathcal{T}_2) \;=\; \min(\{L_{\mathbf{n}}(\mathcal{T}_1) + L_{\mathbf{n}}(\mathcal{T}_2) - L_{\mathbf{n}}(\mathcal{T}_1 \cup \mathcal{T}_2) | \; \forall \mathbf{n} \in N\}) \tag{12}$$

where $N = \{\mathbf{n}_{\mathrm{FV}}, \mathbf{n}_{\mathrm{VF}}, \mathbf{n}_{\mathrm{EE}}\}$ is a set of possible separating directions between $\mathcal{T}_1,\ \mathcal{T}_2$.

Proof. According to the SAT [50], two convex objects $\mathcal{T}_1, \mathcal{T}_2$ do not overlap if there exists a separating axis $\mathbf{n}$ that prevents the axial projection of $\mathcal{T}_1, \mathcal{T}_2$ onto $\mathbf{n}$ from overlapping.

The SAT can be rewritten using Equation 11 as follows:

$$\exists \mathbf{n} \in N,\ L_{\mathbf{n}}(\mathcal{T}_1 \cup \mathcal{T}_2) \geq L_{\mathbf{n}}(\mathcal{T}_1) + L_{\mathbf{n}}(\mathcal{T}_2). \tag{13}$$

Thus, if two objects are interpenetrated, $L_{\mathbf{n}}(\mathcal{T}_1) + \mathrm{L}_{\mathbf{n}}(\mathcal{T}_2) - L_{\mathbf{n}}(\mathcal{T}_1 \cup \mathcal{T}_2) > 0$ for $\forall \mathbf{n}$. Let ε be the result of evaluating Equation 12 and $\mathbf{m}$ be the corresponding separating direction. Thus,

$$\mathbf{m} = \underset{\mathbf{n}}{\operatorname{argmin}}\left\{\left(L_{\mathbf{n}}(\mathcal{T}_1) + L_{\mathbf{n}}(\mathcal{T}_2) - L_{\mathbf{n}}(\mathcal{T}_1 \cup \mathcal{T}_2)\right) \middle| \forall \mathbf{n} \in N\right\} \tag{14}$$

Then, we can prove Theorem 1 by showing that

1. $\varepsilon\mathbf{m}$ separates $\mathcal{T}_1$ and $\mathcal{T}_2$ by translation; and
2. ε is the smallest magnitude among such translations.

Let $\mathcal{T}_1'$, $\mathcal{T}_2'$ be the tetrahedra of $\mathcal{T}_1, \mathcal{T}_2$ translated by $\varepsilon\mathbf{m}$. Since these tetrahedra are not rotated, the projected length of each tetrahedron on the axis $\mathbf{m}$ is the same as before translation:

$$\begin{aligned} L_{\mathbf{m}}(\mathcal{T}_1) &= L_{\mathbf{m}}(\mathcal{T}_1') \\ L_{\mathbf{m}}(\mathcal{T}_2) &= L_{\mathbf{m}}(\mathcal{T}_2') \end{aligned} \tag{15}$$

Since the direction of the translation is the same as the projection axis, the length of the translation vector is $|\varepsilon\mathbf{m} \cdot \mathbf{m}| = \varepsilon$. Then, the entire projected length of the translated tetrahedra is as follows:

$$\begin{aligned} L_{\mathbf{m}}(\mathcal{T}_1' \cup \mathcal{T}_2') &= L_{\mathbf{m}}(\mathcal{T}_1 \cup \mathcal{T}_2) + \varepsilon \\ &= L_{\mathbf{m}}(\mathcal{T}_1 \cup \mathcal{T}_2) + L_{\mathbf{m}}(\mathcal{T}_1) + L_{\mathbf{m}}(\mathcal{T}_2) - L_{\mathbf{m}}(\mathcal{T}_1 \cup \mathcal{T}_2) \\ &= L_{\mathbf{m}}(\mathcal{T}_1) + L_{\mathbf{m}}(\mathcal{T}_2) \\ &= L_{\mathbf{m}}(\mathcal{T}_1') + L_{\mathbf{m}}(\mathcal{T}_2'), \end{aligned} \tag{16}$$

implying that the two tetrahedra translated by $\varepsilon\mathbf{m}$ do not overlap because of the SAT. Figure 15 shows an example of two tetrahedra projected on the axis $\mathbf{m}$ before and after translation $\varepsilon\mathbf{m}$, where $\mathbf{m}$ is the face normal of $\mathcal{T}_1$. The projected length of each tetrahedron does not change during the translation.

Let $\tilde{\varepsilon}\tilde{\mathbf{m}}$ be an arbitrary translation that separates $\mathcal{T}_1, \mathcal{T}_2$, and let $\widetilde{\mathcal{T}_1}$, $\widetilde{\mathcal{T}_2}$ be the translated copies of the tetrahedra. Since $\widetilde{\mathcal{T}_1}$, $\widetilde{\mathcal{T}_2}$ are separated, according to the SAT, there exists a separating axis $\tilde{\mathbf{n}}$ that satisfies $L_{\tilde{\mathbf{n}}}\left(\widetilde{\mathcal{T}_1} \cup \widetilde{\mathcal{T}_2}\right) \geq L_{\tilde{\mathbf{n}}}\left(\widetilde{\mathcal{T}_1}\right) + L_{\tilde{\mathbf{n}}}\left(\widetilde{\mathcal{T}_2}\right)$. For the given set $\mathcal{T}_1 \cup \mathcal{T}_2$ and the direction $\tilde{\mathbf{n}}$, let $\mathbf{x}_1$ and $\mathbf{x}_2$ be the two supporting vertices used to calculate the projection $L_{\tilde{\mathbf{n}}}(\mathcal{T}_1 \cup \mathcal{T}_2) = |(\mathbf{x}_2 - \mathbf{x}_1) \cdot \tilde{\mathbf{n}}|$.

Now, suppose that these two vertices can support the tetrahedra even after being translated by $\tilde{\varepsilon}\tilde{\mathbf{m}}$ (Figure 16(a)), and let $\tilde{\mathbf{x}}_1$ and $\tilde{\mathbf{x}}_2$ be the corresponding vertices after translation. Then the displacement between the two vertices after translation can be calculated as follows: $\tilde{\mathbf{x}}_2 - \tilde{\mathbf{x}}_1 = \mathbf{x}_2 - \mathbf{x}_1 + \varepsilon\tilde{\mathbf{m}}$.

The projected length is $L_{\tilde{\mathbf{n}}}\left(\widetilde{\mathcal{T}_1} \cup \widetilde{\mathcal{T}_2}\right) = |(\tilde{\mathbf{x}}_2 - \tilde{\mathbf{x}}_1) \cdot \tilde{\mathbf{n}}| = |(\mathbf{x}_2 - \mathbf{x_1} + \varepsilon\tilde{\mathbf{m}}) \cdot \tilde{\mathbf{n}}|$. Then,

$$\begin{aligned} L_{\tilde{\mathbf{n}}}(\mathcal{T}_1 \cup \mathcal{T}_2) + |\tilde{\varepsilon}\tilde{\mathbf{m}} \cdot \tilde{\mathbf{n}}| &= |(\mathbf{x}_2 - \mathbf{x}_1) \cdot \tilde{\mathbf{n}}| + |\varepsilon\tilde{\mathbf{m}} \cdot \tilde{\mathbf{n}}| \\ &\geq |(\mathbf{x}_2 - \mathbf{x}_1) \cdot \tilde{\mathbf{n}} + \varepsilon\tilde{\mathbf{m}} \cdot \tilde{\mathbf{n}}| = L_{\tilde{n}}\left(\widetilde{\mathcal{T}_1} \cup \widetilde{\mathcal{T}_2}\right) \\ &\geq L_{\tilde{\mathbf{n}}}\left(\widetilde{\mathcal{T}_1}\right) + L_{\tilde{\mathbf{n}}}\left(\widetilde{\mathcal{T}_2}\right) = L_{\tilde{\mathbf{n}}}(\mathcal{T}_1) + L_{\tilde{\mathbf{n}}}(\mathcal{T}_2) \end{aligned} \tag{17}$$

Otherwise, the supporting vertices are changed after translation (Figure 16(b)). Let $\tilde{\mathbf{x}}_1'$, $\tilde{\mathbf{x}}_2'$ be the supporting vertices after translation and $\mathbf{x}_1'$, $\mathbf{x}_2'$ be the corresponding vertices before translation. Then, $L_{\tilde{\mathbf{n}}}\left(\widetilde{\mathcal{T}_1} \cup \widetilde{\mathcal{T}_2}\right) = |(\tilde{\mathbf{x}}_2' - \tilde{\mathbf{x}}_1') \cdot \tilde{\mathbf{n}}| = |(\mathbf{x}_2' - \mathbf{x}_1' + \varepsilon\tilde{\mathbf{m}}) \cdot \tilde{\mathbf{n}}|$. Since $\mathbf{x}_1'$ and $\mathbf{x}_2'$ are not the supporting vertices before translation, according to Equation 11, the projected length of $\mathbf{x}_1'$ and $\mathbf{x}_2'$ must be smaller than that of supporting vertices $\mathbf{x}_1$, $\mathbf{x}_2$: i.e., $L_{\tilde{\mathbf{n}}}(\mathcal{T}_1 \cup \mathcal{T}_2) = |(\mathbf{x}_2 - \mathbf{x}_1) \cdot \tilde{\mathbf{n}}| \geq |(\mathbf{x}_2' - \mathbf{x}_1') \cdot \tilde{\mathbf{n}}|$. Thus,

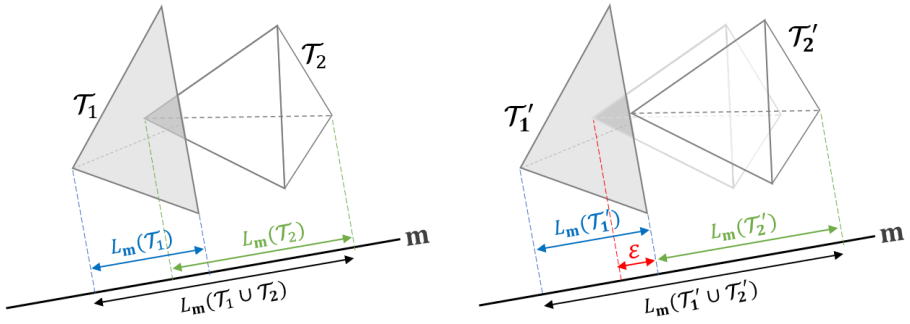
$$
\begin{aligned}
L_{\tilde{\mathbf{n}}}(\mathcal{T}_1 \cup \mathcal{T}_2) + |\tilde{\varepsilon}\tilde{\mathbf{m}} \cdot \tilde{\mathbf{n}}| &= |(\mathbf{x}_2 - \mathbf{x}_1) \cdot \tilde{\mathbf{n}}| + |\tilde{\varepsilon}\tilde{\mathbf{m}} \cdot \tilde{\mathbf{n}}| \\
&\geq |(\mathbf{x}_2' - \mathbf{x}_1') \cdot \tilde{\mathbf{n}}| + |\varepsilon\tilde{\mathbf{m}} \cdot \tilde{\mathbf{n}}| \\
&\geq |(\mathbf{x}_2' - \mathbf{x}_1') \cdot \tilde{\mathbf{n}} + \varepsilon\tilde{\mathbf{m}} \cdot \tilde{\mathbf{n}}| = L_{\tilde{\mathbf{n}}}(\tilde{\mathcal{T}}_1 \cup \tilde{\mathcal{T}}_2) \\
&\geq L_{\tilde{\mathbf{n}}}(\tilde{\mathcal{T}}_1) + L_{\tilde{\mathbf{n}}}(\tilde{\mathcal{T}}_2) = L_{\tilde{\mathbf{n}}}(\mathcal{T}_1) + L_{\tilde{\mathbf{n}}}(\mathcal{T}_2)
\end{aligned} \tag{18}
$$

Figure 16 shows an example of an arbitrary translation $\tilde{\varepsilon}\tilde{\mathbf{m}}$ that separates two tetrahedra and their projection on the separating axis $\tilde{\mathbf{n}}$. Figure 16(a) and Figure 16(b) show two cases of projection results according to the changes in supporting vertices before and after translation. In either case, we can see that $|\tilde{\varepsilon}\tilde{\mathbf{m}} \cdot \tilde{\mathbf{n}}| \geq L_{\tilde{\mathbf{n}}}(\tilde{\mathcal{T}}_1) + L_{\tilde{\mathbf{n}}}(\tilde{\mathcal{T}}_2) - L_{\tilde{\mathbf{n}}}(\mathcal{T}_1 \cup \mathcal{T}_2)$.

Since $|\tilde{\mathbf{m}}| = |\tilde{\mathbf{n}}| = 1$,

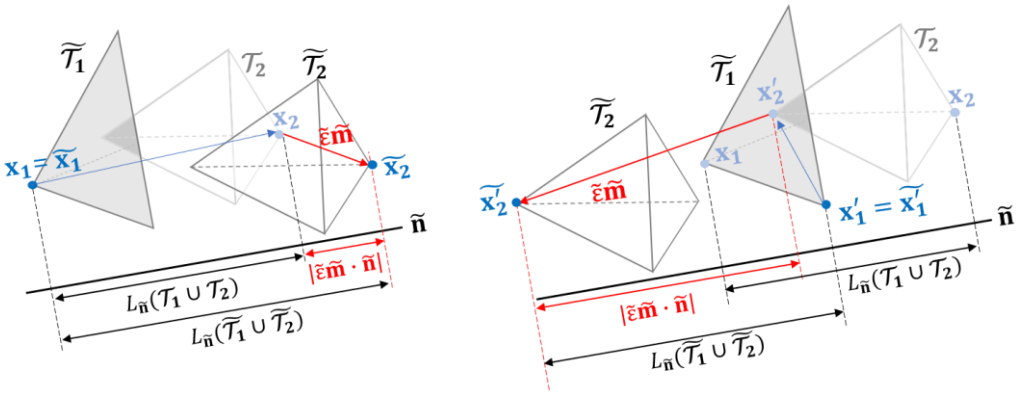
$$
\begin{aligned}
\tilde{\varepsilon} &\geq \tilde{\varepsilon}|\tilde{\mathbf{m}} \cdot \tilde{\mathbf{n}}| \\
&\geq L_{\tilde{\mathbf{n}}}(\tilde{\mathcal{T}}_1) + L_{\tilde{\mathbf{n}}}(\tilde{\mathcal{T}}_2) - L_{\tilde{\mathbf{n}}}(\mathcal{T}_1 \cup \mathcal{T}_2) \\
&\geq \varepsilon
\end{aligned} \tag{19}
$$

Therefore, ε is the minimum translational distance that separates $\mathcal{T}_1$, $\mathcal{T}_2$. ■



(a) Interpenetrated state (b) Translated state

Figure 15. Two tetrahedra projected on the axis $\mathbf{m}$ before and after translation



(a) Same supporting vertices (b) Changed supporting vertices

Figure 16. Projection results according to the change in supporting vertices

B. Acceleration using Rigid PD

Rigid PD can now be computed by calculating the smallest overlapping length of tetrahedra projected over all possible separating axes in SAT. This SAT-based approach is fast and simple to implement because it only needs projection and comparison functions. We can use the corresponding direction and resulting contact features to formulate non-penetration constraints of the QP problem. We have compared the results with that of full optimization over all possible separating direction, and find that the approximation error is below 5%. Interestingly, instead of taking the "minimum direction" in Equation 12, even if we take only the first six directions with ascending order of minimum distances, more than 90% of PD_d direction in Sec. V can be still found. This observation opens up a new possibility of reducing the possible error in this approximation while spending a little more time on search.

VII. Results

In this section, we show the implementation results of our PD algorithms for static/deformable case, deformable/deformable case, and deformable/deformable case with acceleration. In our experiments, the target tetrahedra are randomly generated, which are bounded by a cube with a side length of 10. We have tested 10^4 intersecting pairs of tetrahedra with different volume sizes ranging from 0.1 to 130, and we have computed their $\mathrm{PD_d}$'s. We have implemented our algorithms using C++ on a Windows 10 PC with an AMD Rizen7 1700x 3.6GHz CPU, and 32GB memory. We have used the Gurobi Optimizer to solve the QP problems. Figure 17 shows the overall structures of the implementations.

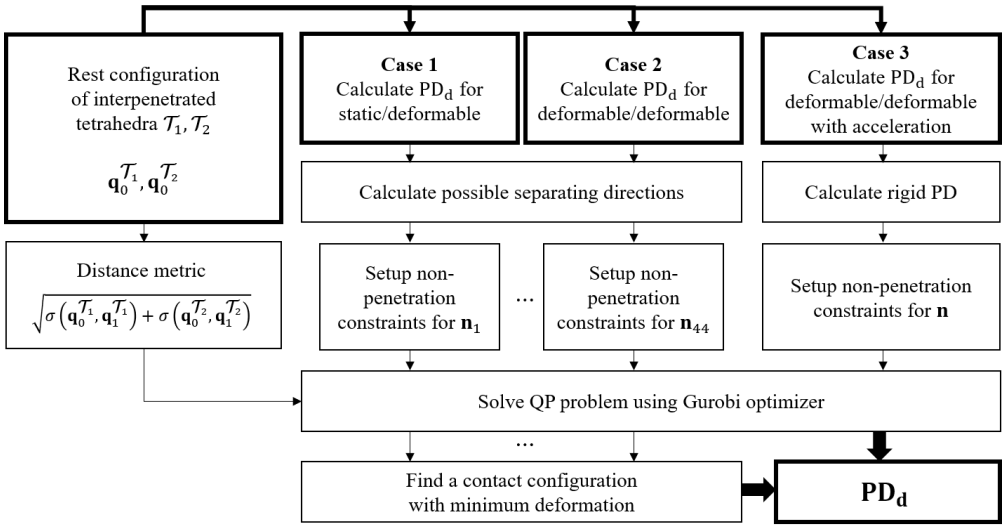


Figure 17. Implementation overview

We also implemented GUI (Figure 18) using OpenGL so that users can easily check out the rest configurations, optimized configurations, and every deformed result of all possible

separating directions in 3D scene. Simple animation is also implemented to show the deformation of two tetrahedra from the rest configuration to the optimized, penetration-resolved configurations.

A. Performance

As illustrated in Figure 19, every result using PD_d resolves penetration. Figure 19(a) shows the initial penetrated state of two tetrahedra. In Figure 19(b)–(d), the solid colored tetrahedra are the results, as follows: PD_d = 0.5692 and 0.3469, PD = 1.496, respectively. The average performance results of each case are shown in Table. I. The running time for the static/deformable case is slightly faster than the deformable/deformable case since a fewer number of variables are required for optimization. The accelerated deformable/deformable case can be calculated in 1.07 ms on the average. Figure 20 and Figure 21 are the example of all possible deformed configurations over 44 separating directions for static/deformable and deformable/deformable cases, respectively. Two intersected tetrahedra in light color show the rest configurations and tetrahedra in pink and purple show the resulting deformed state using PD_d; the contact configuration with smallest deformation among all possible results.

Table 1. Performance Statistics

Criterion	**STAT/DEF**	**DEF/DEF**	**DEF/DEF(ACCEL.)**
Performance	32.26 ms	46.29 ms	1.07 ms
$\mathbf{PD_d/PD}$ (Standard Deviation)	43.59% (13.17%)	27.94% (3.15%)	29.39% (3.47%)

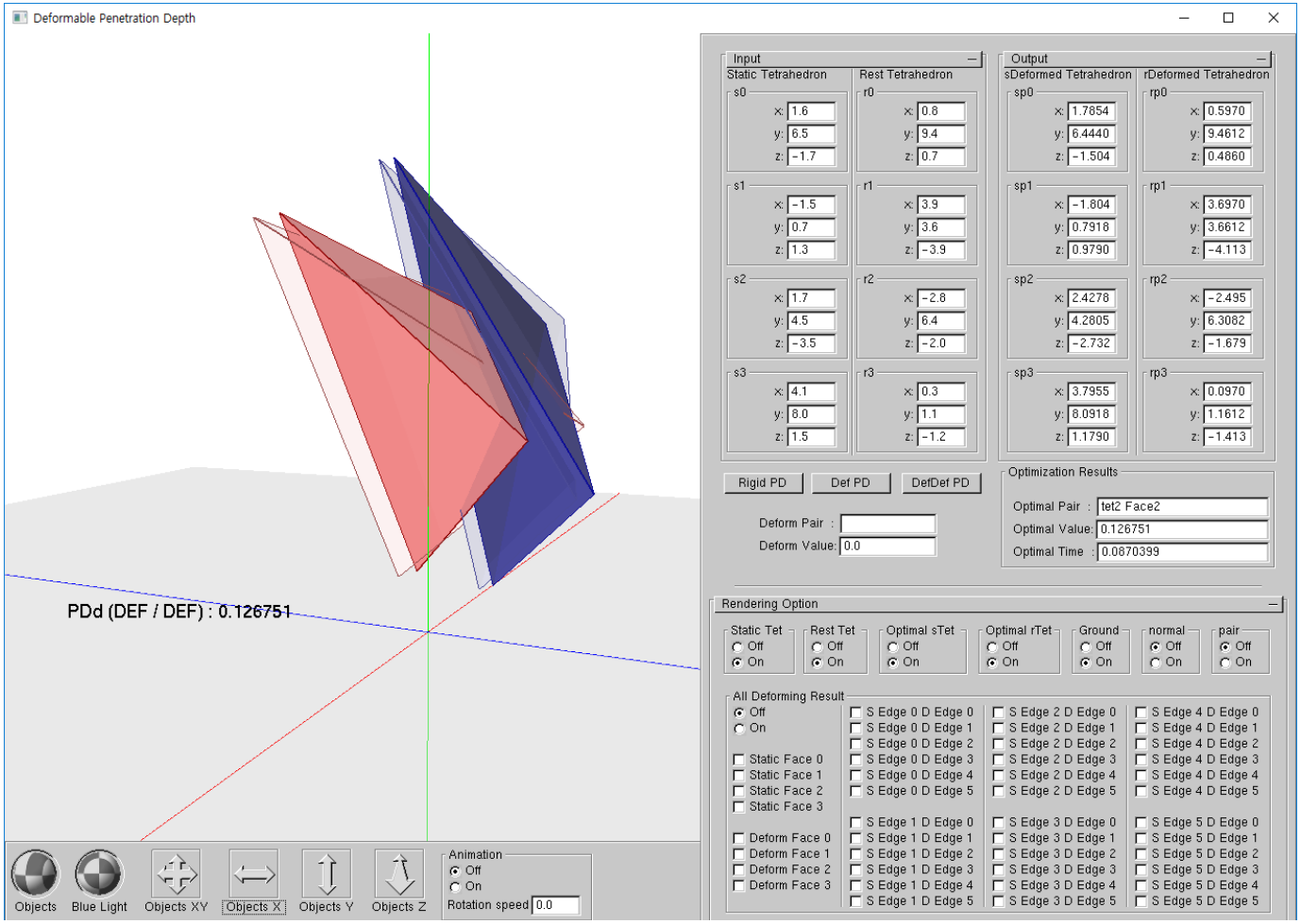
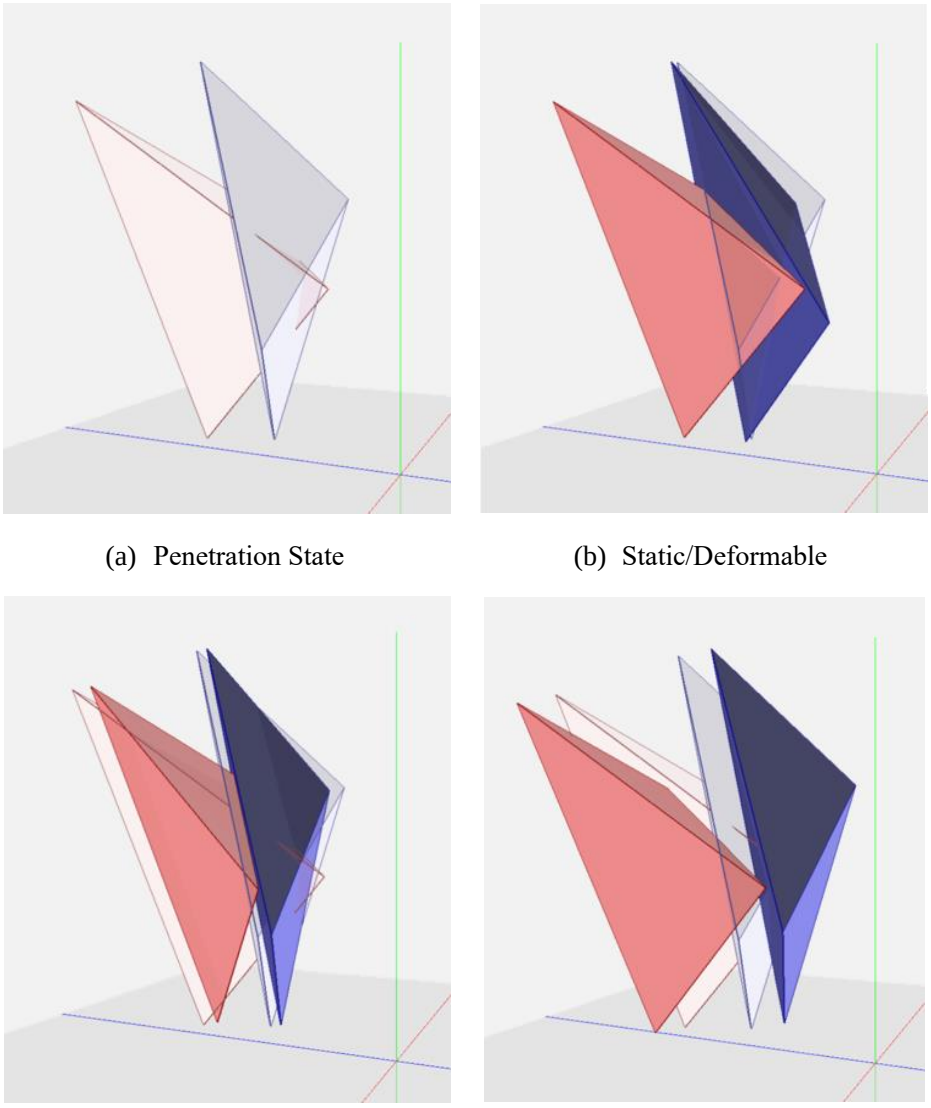


Figure 18. GUI implementation



(a) Penetration State (b) Static/Deformable

(c) Deformable/Deformable (d) Rigid PD

Figure 19. Implementation results for PD_d and rigid PD.

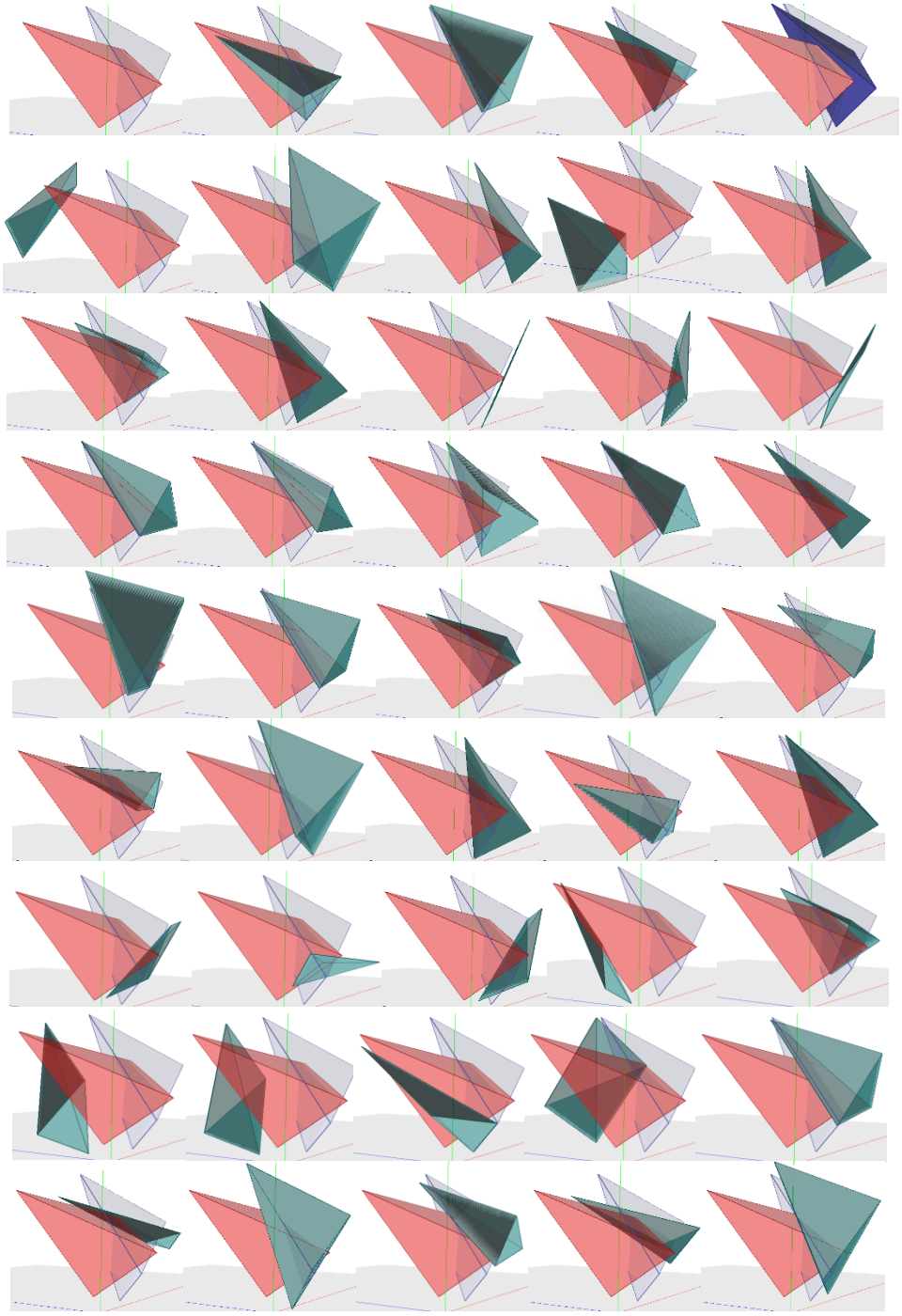


Figure 20. All possible deformed configurations for the static/deformable case

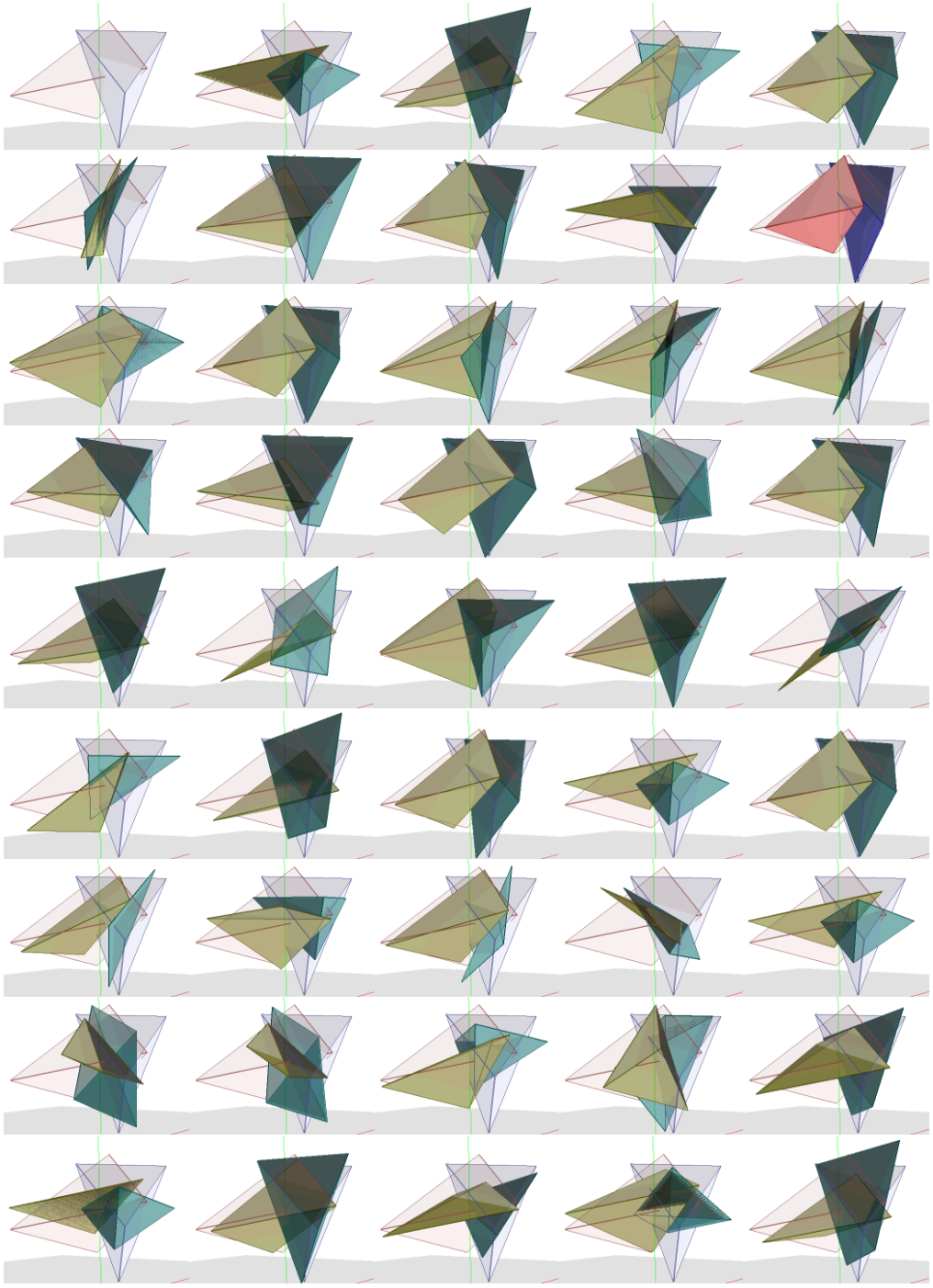


Figure 21. All possible deformed configurations for the deformable/deformable case

B. Discussion

To the best of our knowledge, there are no available PD algorithms for deformable models that guarantee full separation (i.e., penetration metric with a tight upper bound). All the known PD algorithms for deformable models provide only a lower bound, which does not guarantee a full penetration resolution. Thus, it is unfair to compare our algorithms against the existing algorithms for deformable models; instead, we compare our algorithms with rigid PD, which can be considered as an upper bound for $\mathrm{PD_d}$.

In this case, to show the tightness of the metric upper bounds, we calculate the relative magnitude of $\mathrm{PD_d}$ with respect to rigid PD. Figure 22 shows the relative magnitude of $\mathrm{PD_d}$ over rigid PD in penetration resolution tests of 10^4 randomly intersecting tetrahedron pairs. The average magnitude is 27.94 with a standard deviation of 3.15%. This result demonstrates that $\mathrm{PD_d}$ provides a much tighter deformation metric than the rigid PD.

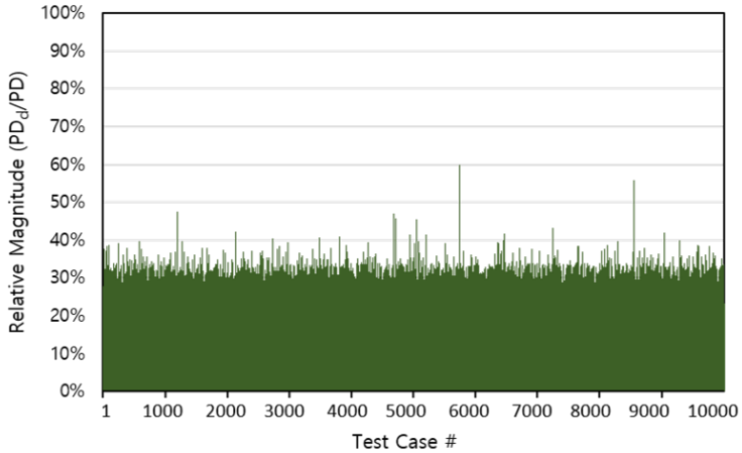


Figure 22. Relative magnitude of $\mathrm{PD_d}$ over rigid PD

VIII. Conclusion

We have formulated a new penetration metric based on object norm for a pair of intersecting tetrahedra undergoing linear deformation. The new metric, called PD_d, optimizes an average displacement of all the points inside the tetrahedra undergoing linear deformation to separate these tetrahedra. PD_d can be computed by solving a QP problem based on the distance metric with non-penetration constraints. We have implemented three cases of computing PD_d, namely, rigid versus deformable case and deformable versus deformable case with and without acceleration. Our experimental results show that we can compute PD_d in a fraction of milliseconds for intersecting, deformable tetrahedra.

There are a few limitations to our algorithm. To derive a tractable optimization problem for PD_d, we have assumed that the separation direction can be obtained from the rest pose of deforming tetrahedra. Even though the proposed method is straightforwardly extendable to a set of tetrahedra by applying our metric to each element in the set, it would be interesting to pursue a technique that accelerates this computation to make it more useful for FEM-type simulation. One plausible direction would be to combine the iterative contact space projection technique [28] with our deformable metric. Our metric does not guarantee volume preservation during deformation, which is another interesting topic for a future work.

Bibliography

[1] S. F. F. S. Gibson and B. Mirtich, “A survey of deformable modeling in computer graphics,” *Merl - a Mitsubishi Electric Research Laboratory*, pp. 1–31, 1997.

[2] A. Nealen, M. Müller, R. Keiser, E. Boxerman, and M. Carlson, “Physically based deformable models in computer graphics,” in *Computer Graphics Forum*, 2006, vol. 25, no. 4, pp. 809–836.

[3] R. Bridson, R. Fedkiw, and J. Anderson, “Robust treatment of collisions, contact and friction for cloth animation,” *ACM SIGGRAPH 2005 Courses, SIGGRAPH 2005*, pp. 594–603, 2005.

[4] K. Ward, F. Bertails, T. Y. Kim, S. R. Marschner, M. P. Cani, and M. C. Lin, “A survey on hair modeling: Styling, simulation, and rendering,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 2, pp. 213–233, 2007.

[5] O. Comas, Z. A. Taylor, J. Allard, S. Ourselin, S. Cotin, and J. Passenger, “Efficient nonlinear FEM for soft tissue modelling and its GPU implementation within the open source framework SOFA,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 5104 LNCS, pp. 28–39, 2008.

[6] D. Henrich and H. Wörn, *Robot manipulation of deformable objects*. Springer Science & Business Media, 2012.

[7] S. Hirai, T. Tsuboi, and T. Wada, “Robust grasping manipulation of deformable objects,” *Proceedings of the IEEE International Symposium on Assembly and Task Planning*, pp. 411–416, 2001.

[8] W. Wang, D. Berenson, and D. Balkcom, “An online method for tight-tolerance insertion tasks for string and rope,” *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2015-June, no. June, pp. 2488–2495, 2015.

[9] S. Miller, J. Van Den Berg, M. Fritz, T. Darrell, K. Goldberg, and P. Abbeel, “A geometric approach to robotic laundry folding,” *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 249–267, 2012.

[10] R. H. Taylor, A. Menciassi, G. Fichtinger, P. Fiorini, and P. Dario, “Medical robotics and computer-integrated surgery,” in *Handbook of Robotics*, Springer, 2016, pp. 1657–1684.

[11] S. Kim, C. Laschi, and B. Trimmer, “Soft robotics: a bioinspired evolution in robotics,” *Trends in Biotechnology*, vol. 31, no. 5, pp. 287–294, 2013.

[12] S. S. Rao, *The finite element method in engineering*. Butterworth-heinemann, 2017.

[13] E. Sifakis and J. Barbic, “FEM simulation of 3D deformable solids: a practitioner’s guide to theory, discretization and model reduction,” in *ACM SIGGRAPH 2012 Courses*, 2012, p. 20.

[14] Y. J. Kim, M. A. Otaduy, M. C. Lin, and D. Manocha, “Fast penetration depth computation for physically-based animation,” in *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2002, pp. 23–31.

[15] Y. Li, M. Tang, S. Zhang, and Y. J. Kim, “Six-degree-of-freedom haptic rendering using translational and generalized penetration depth computation,” *2013 World Haptics Conference, WHC 2013*, pp. 289–294, 2013.

[16] J. Pan, L. Zhang, and D. Manocha, “Retraction-based RRT planner for articulated models,” in *2010 IEEE International Conference on Robotics and Automation*, 2010, pp. 2529–2536.

[17] S. Cameron and R. Culley, “Determining the minimum translational distance between two convex polyhedra,” in *Proceedings. 1986 IEEE International Conference on Robotics and Automation*, 1986, vol. 3, pp. 591–596.

[18] D. Dobkin, J. Hershberger, D. Kirkpatrick, and S. Suri, “Computing the intersection-depth of polyhedra,” *Algorithmica*, vol. 9, no. 6, pp. 518–533, 1993.

[19] M. C. Lin, D. Manocha, and Y. J. Kim, “Collision and Proximity Queries,” in *Handbook of Discrete and Computational Geometry*, CRC Press, 2017.

[20] L. Zhang, Y. J. Kim, G. Varadhan, and D. Manocha, “Generalized penetration depth computation,” *Computer-Aided Design*, vol. 39, no. 8, pp. 625–638, 2007.

[21] K. Kazerounian and J. Rastegar, “Object norms: A class of coordinate and metric independent norms for displacement,” in *Flexible Mechanism, Dynamics and*

Analysis: ASME Design Technical Conference, 1992, vol. 47, no. 1–2, pp. 271–275.

[22] G. Van Den Bergen, "Proximity queries and penetration depth computation on 3d game objects," in *Game Developers Conference*, 2001, vol. 170.

[23] Y. J. Kim, M. C. Lin, and D. Manocha, "Incremental penetration depth estimation between convex polytopes using dual-space expansion," *IEEE Transactions on Visualization and Computer Graphics*, vol. 10, no. 2, pp. 152–163, 2004.

[24] P. Hachenberger, "Exact Minkowksi sums of polyhedra and exact and efficient decomposition of polyhedra into convex pieces," *Algorithmica*, vol. 55, no. 2, pp. 329–345, 2009.

[25] S. Redon and M. C. Lin, "A fast method for local penetration depth computation," *Journal of Graphics Tools*, vol. 11, no. 2, pp. 37–50, 2006.

[26] J.-M. Lien, "Covering Minkowski sum boundary using points with applications," *Computer Aided Geometric Design*, vol. 25, no. 8, pp. 652–666, 2008.

[27] J.-M. Lien, "A simple method for computing Minkowski sum boundary in 3D using collision detection," in *Algorithmic Foundation of Robotics VIII*, vol. 57, G. Chirikjian, H. Choset, M. Morales, and T. Murphey, Eds. Springer Berlin / Heidelberg, 2009, pp. 401–415.

[28] Y. Lee, M. Tang, Y. J. Kim, M. Lee, and C. Je, "PolyDepth: Real-time penetration depth computation using iterative contact-space projection," *ACM Transactions on Graphics*, vol. 31, no. 1, pp. 1–14, 2012.

[29] Y. Kim, D. Manocha, and Y. J. Kim, "Hybrid penetration depth computation using local projection and machine learning," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 4804–4809.

[30] Y. Lee and Y. J. Kim, "PhongPD: Gradient-continuous penetration metric for polygonal models using Phong projection," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 57–62.

[31] Y. Lee, E. Behar, J.-M. Lien, and Y. J. Kim, "Continuous penetration depth computation for rigid models using dynamic Minkowski sums," *Computer-Aided Design*, vol. 78, pp. 14–25, 2016.

[32] Y. Lee, A. Kheddar, and Y. J. Kim, “Continuous signed distance computation for polygonal robots in 3D,” in *2019 IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[33] R. Weller and G. Zachmann, “Inner sphere trees for proximity and penetration queries,” in *Proceedings of Robotics: Science and Systems*, 2009.

[34] J. Allard, F. Faure, H. Courtecuisse, F. Falipou, C. Duriez, and P. G. Kry, “Volume contact constraints at arbitrary resolution,” *ACM Transactions on Graphics (TOG)*, vol. 29, no. 4, p. 82, 2010.

[35] B. Wang, F. Faure, and D. K. Pai, “Adaptive image-based intersection volume,” *ACM Transactions on Graphics (TOG)*, vol. 31, no. 4, p. 97, 2012.

[36] D. Nirel and D. Lischinski, “Fast penetration volume for rigid bodies,” in *Computer Graphics Forum*, 2018, vol. 37, no. 2, pp. 239–250.

[37] M. Tang, M. Lee, and Y. J. Kim, “Interactive Hausdorff distance computation for general polygonal models,” in *ACM Transactions on Graphics (TOG)*, 2009, vol. 28, no. 3, p. 74.

[38] S. Fisher and M. C. Lin, “Fast penetration depth estimation for elastic bodies using deformed distance fields,” in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180)*, 2001, vol. 1, pp. 330–336.

[39] K. Hoff, A. Zaferakis, M. Lin, and D. Manocha, “Fast 3d geometric proximity queries between rigid and deformable models using graphics hardware acceleration,” *UNC-CH Technical Report TR02-004*, 2002.

[40] A. Sud, N. Govindaraju, R. Gayle, I. Kabul, and D. Manocha, “Fast proximity computation among deformable models using discrete Voronoi diagrams,” in *ACM SIGGRAPH*, 2006, pp. 1144–1153.

[41] B. Heidelberger, M. Teschner, R. Keiser, M. Müller, and M. H. Gross, “Consistent penetration depth estimation for deformable collision response,” in *Vision, Modeling and Visualization 2004 : Proceedings, November 16-18, 2004, Stanford, USA*, 2004, pp. 339–346.

[42] G. Hirota, S. Fisher, A. State, C. Lee, and H. Fuchs, “An implicit finite element method for elastic solids in contact,” in *Proceedings Computer Animation 2001. Fourteenth Conference on Computer Animation (Cat. No. 01TH8596)*, 2001, pp. 136–254.

[43] J. Teran, E. Sifakis, G. Irving, and R. Fedkiw, “Robust quasistatic finite elements and flesh simulation,” in *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2005, pp. 181–190.

[44] J. Shade, S. Gortler, L. He, and R. Szeliski, “Layered depth images,” in *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, 1998, pp. 231–242.

[45] B. Heidelberger, M. Teschner, and M. H. Gross, “Real-time volumetric intersections of deforming objects.,” in *Vision, Modeling, and Visualization*, 2003, vol. 3, pp. 461–468.

[46] B. Heidelberger, M. Teschner, and M. Gross, “Detection of collisions and self-collisions using image-space techniques,” *Journal of WSCG*, vol. 12, no. 3, pp. 145–152, 2004.

[47] F. Faure *et al.*, “Sofa: A multi-model framework for interactive physical simulation,” in *Soft Tissue Biomechanical Modeling for Computer Assisted Surgery*, Springer, 2012, pp. 283–321.

[48] F. Faure, J. Allard, F. Falipou, and S. Barbier, “Image-based collision detection and response between arbitrary volumetric objects,” in *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2008, pp. 155–162.

[49] L. Zhang, Y. Kim, and D. Manocha, “A fast and practical algorithm for generalized penetration depth computation,” in *Proceedings of Robotics: Science and Systems*, 2007.

[50] S. Gottschalk, M. C. Lin, and D. Manocha, “OBBTree: A hierarchical structure for rapid interference detection,” in *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, 1996, pp. 171–180.

[51] L. L. C. Gurobi Optimization, “Gurobi optimizer reference manual.” 2018.

[52] Y. Lee and Y. J. Kim, “Simple and parallel proximity algorithms for general polygonal models,” *Computer Animation and Virtual Worlds*, vol. 21, no. 3–4, pp. 365–374, 2010.

국문 초록

김지수

컴퓨터공학과

이화여자대학교 대학원

변형체 시뮬레이션은 근육이나 천과 같은 다양한 변형체들의 사실적인 묘사를 위해 컴퓨터 그래픽스 분야에서 꾸준히 연구되어 왔으며, 최근 로보틱스 분야에서도 소프트 로봇, 로봇보조수술 등의 발전으로 그 필요성이 증가하고 있다. 변형체 시뮬레이션에는 유한요소해석법이 널리 이용되고 있는데 이는 물체를 사면체 같은 작은 요소의 집합으로 정의하고 각 요소에 가해지는 외력에 의한 변형을 계산하는 방법이다. 이때 적절한 접촉 반력을 적용하기 위해서는 변형체의 침투깊이 계산이 필수적이다. 중첩된 물체 간의 침투깊이는 페널티 기반 물리 시뮬레이션에서 물체의 안정적인 움직임을 생성하거나, 햅틱 렌더링에서 접촉 반력을 계산하여 햅틱 피드백을 주는 등 다양한 분야에 이용된다. 하지만, 강체 간의 침투깊이는 다양하게 연구된 바 있으나 변형체에 대한 침투깊이를 엄밀하게 다룬 연구는 상대적으로 매우 적은 편이다.

따라서 본 논문에서는 중첩된 두 변형가능한 사면체 간의 상호 침투량을 측정할 수 있는 변형체 침투깊이(deformable penetration depth, PD_d)를 두 선형적 변형사면체를 분리하는 최소한의 변형으로 정의하고, 이 문제를 기하학적으로 계산하는 방법을 제안한다.

먼저 사면체의 변형 전후 형상(configuration)이 주어졌을 때 그 변형량을 측정할 수 있는 거리 메트릭을 제안한다. 거리 메트릭은 오브젝트 놈(object norm)을 이용하여 정의하며 이는 물체 내부의 모든 점의 평균 변형량으로 해석할 수 있다. 본 논문에서는 거리메트릭을 닫힌 형태의 수식으로 나타낼 수 있음을 보였다.

두 사면체를 분리하되 변형을 최소화하기 위해서는 분리 후 두 사면체가 접촉상태가 되어야 한다. 특정 접촉상태를 만족하는 최소한의 변형은 비침투제약조건을 만족

하면서 거리메트릭을 최소화하는 이차계획법(quadratic programming) 문제를 풀어서 구할 수 있다. 즉 변형체 침투깊이 계산은 두 사면체를 가장 적게 변형해서 만들 수 있는 접촉상태를 찾는 문제로 재정의할 수 있으므로 모든 가능한 접촉상태를 구하고, 각각의 최적화문제를 풀면 변형체 침투깊이를 얻을 수 있다.

본 논문에서는 분리축 정리(separating axis theorem)를 이용해 중첩된 두 사면체의 모든 가능한 접촉상태를 구하고 각각의 접촉상태에 따른 비침투제약조건을 설정해 이차계획법문제를 풀어냄으로써 변형체 침투깊이를 계산하는 방법을 제안하였다. 먼저 비교적 간단한 경우인 고정사면체/변형사면체를 분리하는 방법을 보인 후, 변형중에 분리방향이 변하지 않는다는 가정을 통해 두 변형사면체를 분리하는 문제로 확장할 수 있음을 보였다. 또한, 최적의 접촉상태를 미리 알고 있다면 계산량을 줄일 수 있으므로 강체의 침투깊이를 이용하여 두 사면체의 최종 접촉상태를 미리 구함으로써 성능을 향상하는 방법을 제안하였다.

제안하는 방법을 실험한 결과 고정사면체/변형사면체, 변형사면체/변형사면체 두 경우 변형체 침투깊이를 수십 밀리 초안에 계산할 수 있다는 것을 보였으며, 변형사면체/변형사면체를 강체침투깊이로 전처리한 경우에 5% 이하의 오차로 약 1 밀리초에 계산이 가능함을 보였다. 또한 변형체 침투깊이가 기존 강체 침투깊이에 비해 평균적으로 약 세 배 작은 값을 제공한다는 것을 확인하였다.