Vol.2016-CG-165 No.24
Vol.2016-DCC-14 No.24
Vol.2016-CVIM-204 No.24
2016/11/10

Enhancing Subspaces for Elastic Deformation with Collisions

Duosheng Yu[1,a)] Takashi Kanai[1,b)]

Abstract: In general, subspace methods for elastic deformations can greatly increase the simulation speed and have good global supports. However, when novel external collisions are encountered, the expressivity of subspace is not enough, and then obvious artifacts will appear. In this paper, we present an efficient data-driven approach to adaptively enhance the expressivity of subspace for elastic deformations with novel collisions. We firstly capture several time-series shapes from an object by performing full-space simulation. We then construct a database of subspace including displacements and potential derivatives among shapes of all time-series. In run-time simulation, we choose several proper bases from such a database and use them as run-time local basis. As a result, we show that our approach can achieve faster and more accurate computation of elastic deformation when novel collisions happen.

1. Introduction

The simulation of deformable objects has become more and more important for the realisticity of films, computer games, virtual reality and related fields. Among the methods of deformable object simulation, Finite Element Method (FEM) is widely used for simulating physically-correct deformable object. However, since the full-space FEM simulation is required to calculate internal forces and force differentials of all 3D volumetric elements, it is too expensive to use FEM to simulate high resolution deformable objects in real-time.

To address this problem, the subspace method, also known as model reduction [21], uses pre-computed basis vectors to project the original high-dimensional system into the low-dimensional space that those bases span. Then the simulation of low-dimensional system only depends on the number of basis vectors that is rather smaller than the original DOFs. Therefore, such a simulation can achieve a great acceleration.

Subspace methods typically have good global support, however, can also cause unrealistic artifacts when the expressivity of subspace is not enough, such as novel external collisions are encountered (see Fig. 1). Hahn et al. [9] presented a method that precomputes global bases as usual and in run-time simulation a solution of Boussinesq analysis is used to construct a set of bases to represent local deformation. However, this analytic solution is only suitable for small deformation. Teng et al. [22] combined both full-space and subspace, and a full-space simulation is used for collided area which losts the merit of high efficiency.

In this paper, we present a data-driven approach to solve these two limitations. Using the fact that adding basis vectors can increase the span of subspace, we firstly capture some feature basis vectors by doing full-space collided simulation and save them as subspace database. In run-time simulation, we select proper basis vectors from database to enhance the expressivity of subspace for collided deformation.

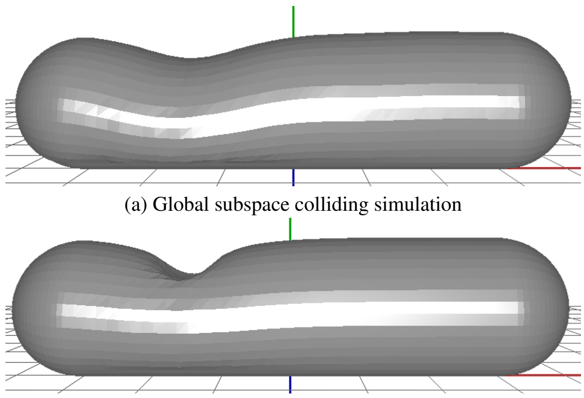
(a) Global subspace colliding simulation
(b) Full-space colliding simulation

Fig. 1 Comparison of traditional subspace method and full-space colliding simulation. *We simulate a falling rigid cylinder to collide a capsule. The global subspace is clearly lack of local information than full-space simulation.*

Our work can be categorized as a novel extension of the idea in Hahn et al's approach [8] to deformable object, but the detail of our method is definitely different. Hahn et al's approach constructs a subspace database for cloth simulation that is able to well reproduce wrinkles and folds. Unlike Hahn et al's approach which creates database by performing full-space simulation with certain character animations, our training stage is using a rigid object to collide simulated object in a pre-defined region. Hahn et al's approach also selects basis vectors by aligning basis vectors to the gradient of current time-step configuration. On the other hand, we select a whole subspace according to the collided position.

[1] The University of Tokyo, 3-8-1 Komaba, Meguro-ku, Tokyo 153-8902, Japan.
[a)] duoshengyu@graco.c.u-tokyo.ac.jp
[b)] kanait@acm.org

The rest of this paper is organized as follows. We will briefly introduce recent developments in deformable object simulation in Section 2. In Section 3 we simply explain subspace method for deformable object simulation. Next, the detail of our data-driven method is described in Section 4. We then show our results compared with other methods in Section 5. We finally conclude our work and describe future work in Section 6.

2. Related Work

Since Terzopoulos et al. [23] introduced the theory of elasticity in computer graphics, deformable object simulation has achieved a lot of developments. There are several methods to simulate a deformable object such as mass-spring system, FEM, shape matching, a recently-developed position based dynamics, etc. There are great surveys [18] and [6] which provide a good cover of deformable object simulation.

We are particularly interested in FEM method, which is one of most poplar methods in deformable object simulation [21]. Objects are discretized into continuously connected volumes, then continuum mechanics equation which can provide physically correct deformation is solved. The simplest FEM method uses linear material for small deformations, however, it typically causes artifacts under large deformations. A linear co-rotational FEM by Müller et al. [17] extracts rotations of local elements to warp stiffness. McAdams et al. [16] and Barbič [2] further compute an exact co-rotational stiffness matrix which makes simulation more stable. Other nonlinear materials could deal with large deformation, such as St. Venant-Kirchhoff, Neo-Hookean, Mooney-Rivlin materials [7]. Li et al. [15] and Xu et al. [28] do not restrict to these standard materials, and they proposed material design which can adjust the properties of the material for specific application.

Subspace methods were firstly introduced to deformable object simulation by Pentland and Willams [19], but were applied only for linear materials. James et al. [11] used precomputed modal analysis to efficiently simulate dynamic deformation of muscles in character animation. Hauser et al. [10] showed that subspace modal framework can be easily coupled with external constraints, such as manipulation, collision, etc. Barbič and James [3] presented modal derivatives for fast subspace integration of reduced nonlinear St. Venant-Kirchhoff material. A fast evaluation of internal forces based on subspace by optimizing cubatures [1] for different types of materials could reduce the cost of evaluating subspace forces from $O(r^4)$ to $O(r^2)$, where r is the dimension of subspace. Subspace methods have proved its power over a lot of graphics applications, such as shape interpolation [26], skinning character dynamics [27], animation editing [4, 15] and fluid simulation [24].

A variety of works have been proposed to address the expressivity limitation of subspace. Barbič et al. [5] and Kim et al. [13] used multi-domain technologies. The basis vectors usually have global support, so the object is partitioned into several connected domains and subspace simulation is performed for each sub-domain to localize the influence of basis vectors. Kim et al. [12] proposed error estimation for subspace simulation, detecting when subspace is capable of performing the next time-step, and switching to full-space simulation when the expressivity of subspace is not enough. Harmon et al. [9] used analytic solution to calculate local bases. Hahn et al. [8] created database to enrich run-time subspace. Teng et al. [22] combines subspace and full-space at the same time. Our method complements the limitation of their works.

3. Background

3.1 Equation of simulation

We will first explain the base of FEM deformable object simulation. In computer graphics, we often use 3D volumetric elements like tetrahedrons or hexahedrons to approximate the object we want to simulate. Given a tetrahedral mesh with N vertices, the deformation $u \in \mathbb{R}^{3N}$ is the displacement of vertices away from the rest configuration X of mesh in world coordinates. The equation that governs the motion of a deformable object can be written as:

$$M\ddot{u} + D\dot{u} + f_{int}(u) = f_{ext}, \tag{1}$$

where $M \in \mathbb{R}^{3N\times3N}$ is the mass matrix of object, $D \in \mathbb{R}^{3N\times3N}$ is the damping matrix, f_{int} , $f_{ext} \in \mathbb{R}^{3N}$ is an internal force and an external force of this object respectively. Dots denote time derivatives.

This equation can be simplified to yield a quasistatic equation:

$$Ku = f_{ext} - f_{int}, \tag{2}$$

where $K \in \mathbb{R}^{3N\times3N}$ is the tangent stiffness matrix with respect to current configuration. The quasistatic equation always makes the simulated result reach to rest configuration over time. For simplicity we use this equation in our result.

3.2 Subspace method

In full-space simulation, Equation (1) and (2) have to evaluate strain and stress of all volumetric elements and solve a large linear equation, which is too expensive for interactive applications. To address this problem, subspace method is used to accelerate the simulation of dynamical systems described by differential equations. The idea underlying subspace method is that using a subspace basis matrix $U \in \mathbb{R}^{3N\times r}$ the original high dimensional system is projected to a low custom dimensional system, where r ($r << 3N$) is a dimension of this subspace. The ordinary differential equation in Equation (1) and (2) can be projected by pre-multiplying U^T from the left and U from the right:

$$\begin{aligned}\bar{M} = U^T MU, \bar{D} = U^T DU, \bar{K} = U^T KU,\\ \bar{f}_{int} = U^T f_{int}, \bar{f}_{ext} = U^T f_{ext},\end{aligned} \tag{3}$$

The original $3N$ high dimensional equation then becomes r low dimensional equation:

$$\bar{M}\ddot{q} + \bar{D}\ddot{q} + \bar{f}_{int}(q) = \bar{f}_{ext}, \tag{4}$$

$$\bar{K}u = \bar{f}_{ext} - \bar{f}_{int}, \tag{5}$$

where q is the reduced coordinates. The affine space of U can be represented as a linear combination of basis vectors attached to the reference state of simulated object $\{Uq + X|q \in \mathbb{R}^r\}$. Here we apply the idea of [9], any subset of precomputed basis vectors can be chosen for simulation. We can think that our runtime subspace is selected from a larger, unknown bases $r' >> r$. Theoretically, there are an infinite number of such bases, and we gather those bases which we are interested to database. During run-time simulation, we can augment our run-time subspace with the vectors from database which are most likely to capture the deformation of the object:

$$U = [G \quad L] \in \mathbb{R}^{3N\times(r+s)}, \tag{6}$$

where G is the precomputed r global subspace basis matrix to support global simulation. L is the s local subspace basis matrix, which is selected from our database for augmenting the affine space of our run-time subspace. So the projection of subspace is:

$$\bar{K} = U^T KU = \begin{bmatrix} G^T KG & G^T KL \\ L^T KG & L^T KL \end{bmatrix}, \tag{7}$$

$$\bar{f}_{int} = U^T f_{int} = \begin{bmatrix} G^T f_{int} \\ L^T f_{int} \end{bmatrix}. \tag{8}$$

The other variants would be projected by the same way. A global subspace basis matrix G can be computed by a variety of methods, such as modal analysis [20], modal derivatives [3], modal extension [25], etc. The subspaces of those methods all have good global support. Here we simply introduce the modal analysis method which solves the generalized eigen problem,

$$Kv = \lambda Mv, \tag{9}$$

for the smallest r eigenvalues. We then collect the corresponding eigenvectors to construct subspace basis matrix. We use this method to construct our global subspace G. The construction of local subspace L will be introduced in Section 4.

4. Proposed Method

4.1 Overview

Fig. 2 shows the overview of our system. Here we briefly describe each stage of our system.

The input of our system is a tetrahedra mesh of a deformable object we want to simulate and a rigid object that creates collisions. During pre-process, we firstly define a region where we predict that collision will happen in run-time and compute our global subspace using modal analysis in Equation (9). We then perform full-space simulations using the rigid object collides simulated object along the normal direction of this region. We capture several keyframes from those motions and then construct our database using the method described in Section 4.2. In run-time simulation, collision detection is performed for each frame. If collision does not happen, we simulate an object with global subspace. We would choose a proper local subspace and add it to a global subspace when collision happens in a predicted region.

4.2 Motion-based database

The first question is what should be stored in our database. Here we extend the concept of pose-space deformation [14] to motion space. A set of poses which could be obtained directly by skinning or inverse kinematics, and several pose controls from artists such as volume preservation or elimination of candy wrapper are given. Pose space deformation then solves an interpolation problem to generate desired animations of a character.

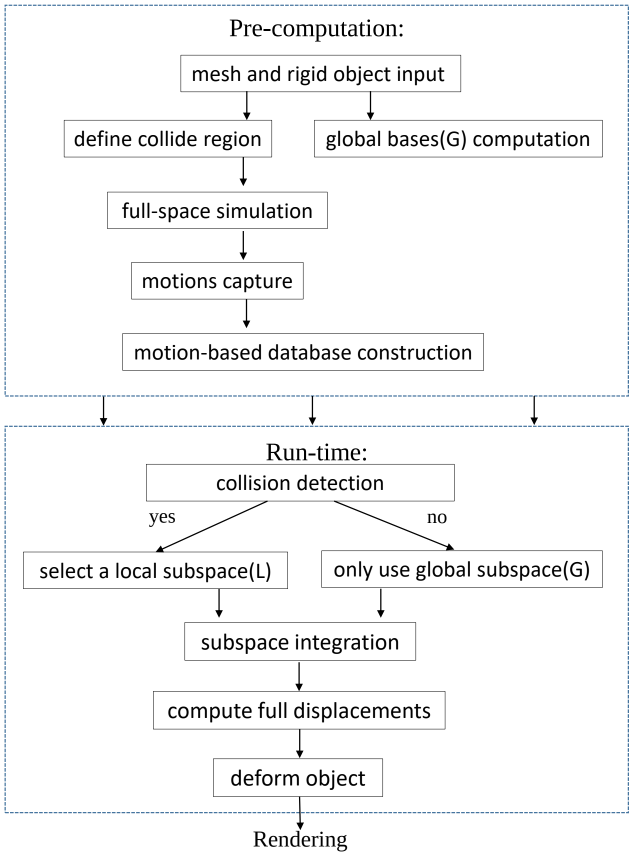

Fig. 2 Overview of our system.

In our case, the deformation of collision is defined as continuous dynamics. Since specifically sparse poses only span the space among themselves, and the expressivity of such a subspace is not enough for our collided deformation, we choose to use motion as the ingredient of our database.

Training stage

We perform full-space colliding simulation for the equally-spaced region where we predict that collision happens. In this case, storing the deformation of every frames would result in a massive amount of data that is not practical. We would prefer a small amount of data that well capture the deformation. Our solution here is to capture a keyframe in each k-th frame from a motion. A keyframe is a configuration of an object at certain time. The size and the quality of database can be controlled by adjusting k. More keyframes would result larger size but more accurate database.

Basis computation

Assume we have captured m keyframes from a motion. A set of keyframes and local subspace of a motion i is defined as $X_i = \{x_1, x_2, ..., x_m\}$ and L_i respectively, where $x_i \in \mathbb{R}^{3N}$ is the position of vertices in i-th keyframe. The number in subscripts indicates a chronological order of this motion.

We firstly calculate the affine space which linearly spans this motion using the displacement between keyframes as described in [8, 22]. These displacements represent the deformation of this motion.

$$x_k - x_{k-1} \quad \text{subject to} \quad k \in \{2, 3, ..., m\}. \tag{10}$$

Data: Sets of keyframes $\{X_1, X_2, ...X_i\}$ of training motions
Result: Database $\{L_1, L_2, ..., L_i\}$
for $i = 1, 2, ...$ **do**
    $L_i \leftarrow \{\}$;
    for $k = 2, 3, ..., m$ **do**
        $L_i \leftarrow L_i \cup \{x_k - x_{k-1}\}$;
    end
    for $j, k = 1, 2, 3, ..., m; j \neq k$ **do**
        Solve $K(x_i)D_j = F_j(x_i)$;
        $L_i \leftarrow L_i \cup \{D_j\}$;
    end
    do PCA for L_i (optional);
end

Algorithm 1: Database construction algorithm.

We add all resulting $m - 1$ vectors to a local subspace L_i.

We then compute the derivatives of all these keyframe configurations where the corresponding derivative with respect to a keyframe configuration is a vector in $\mathbb{R}^{3N}$, as desribed in [26]. All derivative vectors of a keyframe configuration linearly span a $m - 1$ subspace that best approximates other keyframe configurations. We can think that this affine space is the tangent space at keyframe configuration x_i.

To get more accurate simulated result, we need a subspace that contains the tangent spaces of the m keyframe states. The derivatives are calculated by solving the following equation,

$$K(x_i)D_j(x_i) = F_j(x_i), \qquad (11)$$

where $K(x_i)$ is the tangent stiffness matrix in configuration x_i, $F_j(x_i)$ is the internal force deforming x_i into x_j, and $D_j(x_i)$ is the resulting derivatives. We do this for all states, so the resulting tangent subspace has $m \times m - m$ vectors. We finally add all these vectors to L_i. Totally, L_i will contain $m^2 - 1$ vectors.

The construction of the database is listed in Algorithm 1. After we collect the difference vectors and the derivative vectors of a motion to L_i. A truncated PCA over each local subspace L_i to orthogonalize this local basis is optionally used, the size of database can be further reduced.

4.3 Basis selection

Once a motion-based subspace database is constructed, the remaining question is how to select a proper subspace in run-time simulation. We want this subspace to be low-dimensional and to well capture the collided deformation.

One solution is to compare the current state with the vectors in database using some evaluate metrics such as L2 norm, and to gather a constant number of vectors that best approximate the current state to construct our run-time subspace. However in our experiment, we found that this solution is not practical. Since our database contains a large amount of vectors, updating a subspace every frame would take too much time which does not meet our motivation. Also, updating a subspace with a regular interval frames would result in discontinuous subspaces which make the simulation become unstable. Another solution is to create a fixed subspace from all training data using data analysis technologies such as PCA. However, this method losts the detail of local deformation.

For these reasons, our current solution chooses to select a whole subspace L_i according to the position where collision happens in run-time simulation. In the training stage, in addtion to computing local subspaces for full-space motions, we also store the position where a rigid object collides a simulation object. In run-time simulation, when collision happens, we compare the position where collision happens with the saved positions. We then choose the closest position and the corresponding L_i for our run-time subspace.

Table 1 **L^2-error of displacement.** *"der" is the subspace of our method, and "dis" is subspace which only contain displacements.*

frame no.	20th	30th	40th	50th
der	0.0115	0.0231	0.0327	0.0197
dis	0.1452	0.2447	0.3707	0.4660

5. Results and Discussion

Implementation

In our result, we use the co-rotational material introduced in [21]. We perform our simulation over the tetrahedral mesh of a capsule shape with 5,312 vertices and 18,604 tetrahedra. We define that the top region on a mesh would be collided in run-time simulation. We firstly perform full-space simulation using a rigid cylinder which collides such a region to generate 51 motions (each motion has 75 frames) for our method. Using this training data, we capture 5 keyframes in each 17 frames interval for a motion, then calculate our motion-based local subspace, resulting in 24 basis vectors for a motion.

Validation

We firstly demonstrate that the accuracy of our motion-based subspace is better than the previous method. We use a rigid cylinder to collide with a capsule for both full-space and different subspaces that are created by using the same motion that collides at the same place as training data. The simulation results are shown in Figure 3.

In Table 1, we also list the relative L^2-errors for the displacement between the rest configuration and a certain configuration in a frame. This L^2-error is computed as,

$$\varepsilon = \|D^i_{full} - D^i_{sub}\|_2, \qquad (12)$$

where D^i_{full} is the displacement of full-space simulation and D^i_{sub} is the displacement of subspace simulation at i-th frame respectively.

Next, to validate our subspace selection strategy, we implement a user interface that provides an intuitive controller for changing the position that a rigid cylinder would collide. Whenever the user clicks in the region where is pre-defined, the rigid cylinder will fall from the clicked position. We use this interface randomly to choose four positions and capture the deformation. Figure 4 shows our result. It is obvious that our method can well capture local deformation.

Evaluation of Computational Time

We list the computational times for pre-computation in table 2 and run-time simulation in table 3. At run-time, we simulate 500 frames in each method, then measure its average time of all frames. The times for computing internal forces and stiffness matrix of both full-space and subspace are the same. In full-space simluation, we used a conjugate gradient method to solve a large

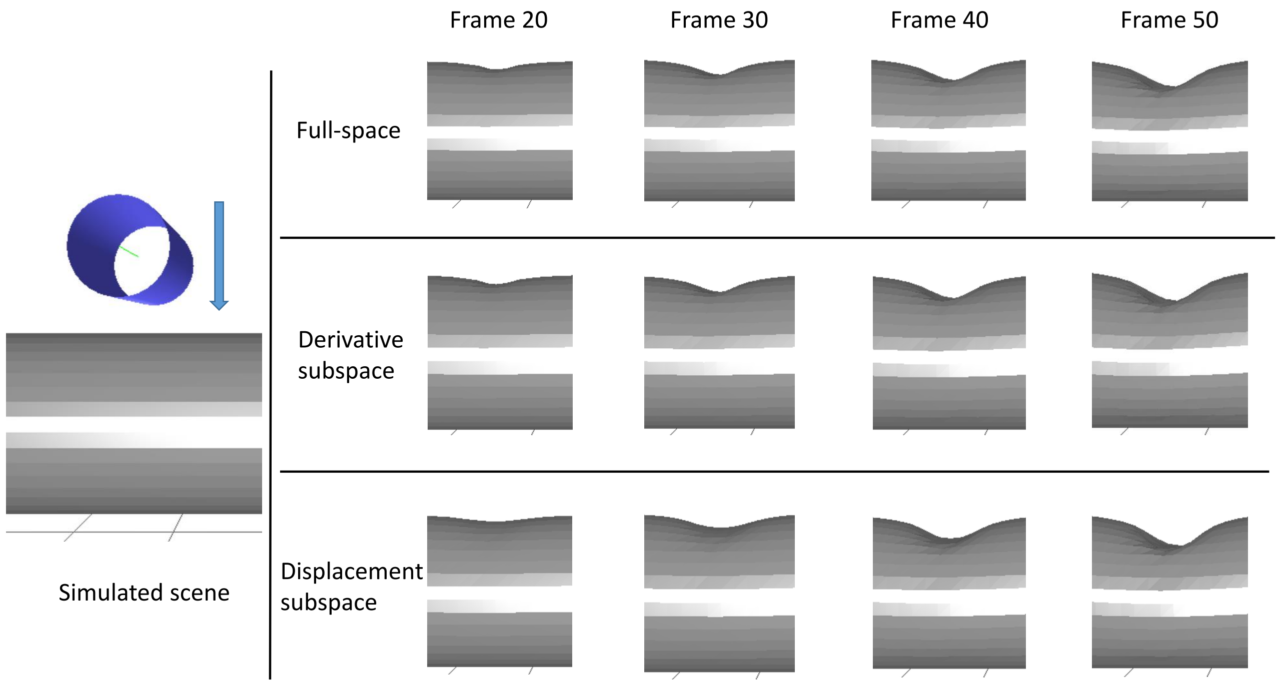

Fig. 3 Comparison of our subspace with previous subspace and full-space simulations. *Left: Our simulated scene which a rigid cylinder falls and collides with a deformable capsule. Right: Our simulation result. The top row is the result of full-space simulation, the middle row is the result of our method, and the bottom row is the result of previous subspace method, respectively.*

Table 2 Computational times for pre-computation.

	training	modal analysis	database construction
Time used	1534.69s	1.13s	441.49s

Table 3 Simulation statistics. *From left to right: the dimension of global subspace (r) , the dimension of local subspace (s), the computational times per frame under full-space (f-s), global subspace only (g-s) and our subspace using database (d-s), the speedup of our method with respect to full-space simulation (sp).*

r	s	f-s	g-s	d-s	sp
30	24	0.389s	0.138s	0.165s	2.36×

sparse linear equation, and to solve a small dense linear equation in subspace we used LU-decomposition. Our database method is slightly slower than the global subspace simulation because we add a local subspace L to a global subspace G which makes the dimension of our subspace larger. Nevertheless, it is obvious that our method can achieve the speedup of subspace and compensate the expressivity of subspace.

6. Conclusion and Future Work

In this paper, we propose a scheme that achieves accurate local deformation of collided deformable simulation using a carefully-constructed subspace database to enhance the expressivity of existed subspace. Our database is motion-based which can capture more detail than pose-based. The result shows that our method can achieve the deformation that well approximates full-space simulation. With the position-based selection, we can keep the simulation in a very low-dimensional subspace. This makes the scheme well-suited for real-time applications that involve collisions of deformable objects such as video games, surgery simulation, and so on.

There are several aspects that we should consider as our future work.

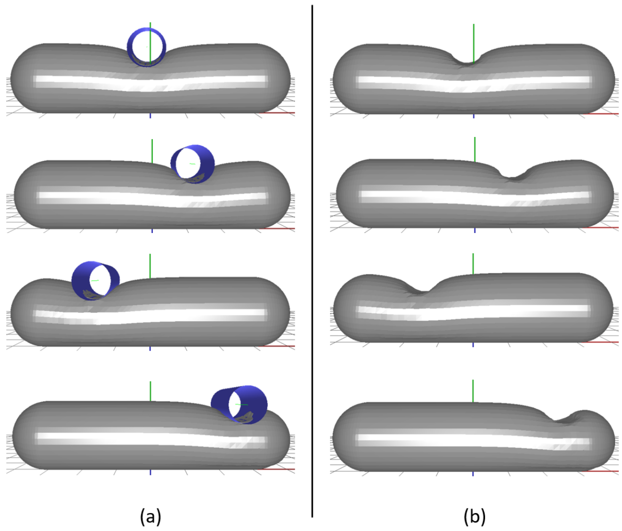

Fig. 4 Validation of our select strategy. *(a) is the result with rigid cylinder rendered, (b) is the result without rigid cylinder rendered. They were captured in same time-step.*

6.1 Training stage

We currently define a region for the colliding simulation which is relatively small with respect to whole surface of our simulation object, and we only fell a rigid object from a single direction. Then, if the collision happened outside of the region where our database is covered, our method produces deformations very close to traditional subspace simulation. Our final goal is to construct a system that the user can drag the rigid object to collide deformable object from arbitrary position and direction.

For such an extension, we are thinking about partitioning the entire surface into several sub-regions so that the area of each sub-region is no more than a certain threshold. For example, using k-means clustering with $k = \lceil S_t/(CS_{ave}) \rceil$, where S_t is a total area of surface, S_{ave} is the average area of surface, and C is a user-defined constant. Another choice is to use Poisson disk sampling, which samples the vertices on the surface subject to their distances no more than user-defined values. After partitioning or sampling, we can set up a hemisphere with the center of a sub-region or at a sampled vertex, then perform full-space collide simulation from the vertices of a icosahedron in this hemisphere to the center to create our training data.

6.2 Subspace selection

The above aspect which is considered for future improvement is our data selection strategy. Currently we just choose one subspace L_i which is a trained motion closest to where novel collision happens. This subspace can be seen as an approximate span of novel collision subspace. We think about selecting several subspaces near the collided position and interpolate them according to the distance between their trained motions and their collided positions. We expect this strategy can provide more accurate subspace.

6.3 Other aspects

Computing reduced internal forces and reduced stiffness matrix by projecting their full-space version will take too much computational resources, because every volumetric elements have to be evaluated. Cubature by An et al. [1] can significantly increase the performance of subspace simulation by using a small amount of key elements to rapidly evaluate reduced internal force and reduced stiffness.

However, the original cubature is trained with a pre-computed global subspace. In contrast, our subspace is updated in run-time simulation, so the original cubature will cause error forces and stiffness. A new cubature scheme should be researched.

The current co-rotational material we used cannot deal with extreme deformation of volumetric elements which make the deformation unstable. This problem may be eliminated by applying with indefiniteness correction [16].

References

[1] An, S. S., Kim, T. and James, D. L.: Optimizing Cubature for Efficient Integration of Subspace Deformations, *ACM Trans. Graph.*, Vol. 27, No. 5, pp. 165:1–165:10 (online), DOI: 10.1145/1409060.1409118 (2008).

[2] Barbič, J.: Exact Corotational Linear FEM Stiffness Matrix, Technical report, University of Southern California (2012).

[3] Barbič, J. and James, D. L.: Real-Time Subspace Integration for St. Venant-Kirchhoff Deformable Models, *ACM Trans. Graph.*, Vol. 24, No. 3, pp. 982–990 (online), DOI: 10.1145/1073204.1073300 (2005).

[4] Barbič, J., Sin, F. and Grinspun, E.: Interactive Editing of Deformable Simulations, *ACM Trans. Graph.*, Vol. 31, No. 4, pp. 70:1–70:8 (online), DOI: 10.1145/2185520.2185566 (2012).

[5] Barbič, J. and Zhao, Y.: Real-time Large-deformation Substructuring, *ACM Trans. Graph.*, Vol. 30, No. 4, pp. 91:1–91:8 (online), DOI: 10.1145/2010324.1964986 (2011).

[6] Bender, J., Müller, M., Otaduy, M. A., Teschner, M. and Macklin, M.: A Survey on Position-Based Simulation Methods in Computer Graphics, *Computer Graphics Forum*, Vol. 33, No. 6, pp. 228–251 (online), DOI: 10.1111/cgf.12346 (2014).

[7] Bonet, J. and Wood, R. D.: *Nonlinear Continuum Mechanics for Finite Element Analysis*, Cambridge University Press (1997).

[8] Hahn, F., Thomaszewski, B., Coros, S., Sumner, R. W., Cole, F., Meyer, M., DeRose, T. and Gross, M.: Subspace Clothing Simulation Using Adaptive Bases, *ACM Trans. Graph.*, Vol. 33, No. 4, pp. 105:1–105:9 (online), DOI: 10.1145/2601097.2601160 (2014).

[9] Harmon, D. and Zorin, D.: Subspace Integration with Local Deformations, *ACM Trans. Graph.*, Vol. 32, No. 4, pp. 107:1–107:10 (online), DOI: 10.1145/2461912.2461922 (2013).

[10] Hauser, K. K., Shen, C. and O'Brien, J. F.: Interactive Deformation Using Modal Analysis with Constraints, *Proc. Graphics Interface 2003*, Canadian Human-Computer Commnication Society, pp. 247–256 (2003).

[11] James, D. L. and Pai, D. K.: DyRT: Dynamic Response Textures for Real Time Deformation Simulation with Graphics Hardware, *ACM Trans. Graph.*, Vol. 21, No. 3, pp. 582–585 (online), DOI: 10.1145/566654.566621 (2002).

[12] Kim, T. and James, D. L.: Skipping Steps in Deformable Simulation with Online Model Reduction, *ACM Trans. Graph.*, Vol. 28, No. 5, pp. 123:1–123:9 (online), DOI: 10.1145/1618452.1618469 (2009).

[13] Kim, T. and James, D. L.: Physics-Based Character Skinning Using Multidomain Subspace Deformations, *IEEE Transactions on Visualization and Computer Graphics*, Vol. 18, No. 8, pp. 1228–1240 (online), DOI: 10.1109/TVCG.2012.78 (2012).

[14] Lewis, J. P., Cordner, M. and Fong, N.: Pose Space Deformation: A Unified Approach to Shape Interpolation and Skeleton-driven Deformation, *Proc. SIGGRAPH '00*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, pp. 165–172 (online), DOI: 10.1145/344779.344862 (2000).

[15] Li, S., Huang, J., de Goes, F., Jin, X., Bao, H. and Desbrun, M.: Space-time Editing of Elastic Motion Through Material Optimization and Reduction, *ACM Trans. Graph.*, Vol. 33, No. 4, pp. 108:1–108:10 (online), DOI: 10.1145/2601097.2601217 (2014).

[16] McAdams, A., Zhu, Y., Selle, A., Empey, M., Tamstorf, R., Teran, J. and Sifakis, E.: Efficient Elasticity for Character Skinning with Contact and Collisions, *ACM Trans. Graph.*, Vol. 30, No. 4, pp. 37:1–37:12 (online), DOI: 10.1145/2010324.1964932 (2011).

[17] Müller, M. and Gross, M.: Interactive Virtual Materials, *Proc. Graphics Interface 2004*, Canadian Human-Computer Communications Society, pp. 239–246 (2004).

[18] Nealen, A., Müller, M., Keiser, R., Boxerman, E. and Carlson, M.: Physically Based Deformable Models in Computer Graphics, *Computer Graphics Forum*, Vol. 25, No. 4, pp. 809–836 (online), DOI: 10.1111/j.1467-8659.2006.01000.x (2006).

[19] Pentland, A. and Williams, J.: Good Vibrations: Modal Dynamics for Graphics and Animation, *SIGGRAPH Comput. Graph.*, Vol. 23, No. 3, pp. 207–214 (online), DOI: 10.1145/74334.74355 (1989).

[20] Shabana, A. A.: *Theory of vibration: Volume II: discrete and continuous systems*, Springer Science & Business Media (2012).

[21] Sifakis, E. and Barbic, J.: FEM Simulation of 3D Deformable Solids: A Practitioner's Guide to Theory, Discretization and Model Reduction, *ACM SIGGRAPH 2012 Courses*, New York, NY, USA, ACM, pp. 20:1–20:50 (online), DOI: 10.1145/2343483.2343501 (2012).

[22] Teng, Y., Meyer, M., DeRose, T. and Kim, T.: Subspace Condensation: Full Space Adaptivity for Subspace Deformations, *ACM Trans. Graph.*, Vol. 34, No. 4, pp. 76:1–76:9 (online), DOI: 10.1145/2766904 (2015).

[23] Terzopoulos, D., Platt, J., Barr, A. and Fleischer, K.: Elastically Deformable Models, *SIGGRAPH Comput. Graph.*, Vol. 21, No. 4, pp. 205–214 (online), DOI: 10.1145/37402.37427 (1987).

[24] Treuille, A., Lewis, A. and Popović, Z.: Model Reduction for Real-time Fluids, *ACM Trans. Graph.*, Vol. 25, No. 3, pp. 826–834 (online), DOI: 10.1145/1141911.1141962 (2006).

[25] von Tycowicz, C., Schulz, C., Seidel, H.-P. and Hildebrandt, K.: An Efficient Construction of Reduced Deformable Objects, *ACM Trans. Graph.*, Vol. 32, No. 6, pp. 213:1–213:10 (online), DOI: 10.1145/2508363.2508392 (2013).

[26] von Tycowicz, C., Schulz, C., Seidel, H.-P. and Hildebrandt, K.: Real-Time Nonlinear Shape Interpolation, *ACM Trans. Graph.*, Vol. 34, No. 3, pp. 34:1–34:10 (online), DOI: 10.1145/2729972 (2015).

[27] Xu, H. and Barbič, J.: Pose-space Subspace Dynamics, *ACM Trans. Graph.*, Vol. 35, No. 4, pp. 35:1–35:14 (online), DOI: 10.1145/2897824.2925916 (2016).

[28] Xu, H., Sin, F., Zhu, Y. and Barbič, J.: Nonlinear Material Design Using Principal Stretches, *ACM Trans. Graph.*, Vol. 34, No. 4, pp. 75:1–75:11 (online), DOI: 10.1145/2766917 (2015).

