A FEM Based Shell Solver—Final Report

MingXuan Liu
2022012067
liu-mx22@mails.tsinghua.edu.cn

Yuchen Su
2022013267
suyc22@mails.tsinghua.edu.cn

I. Introduction

Thin shells are slender, flexible structures characterized by a high width-to-thickness ratio (greater than 100). They are ubiquitous in both natural and engineered systems, from the delicate curvature of leaves to the structural components of vehicles and architecture. Unlike flat plates or fully volumetric solids, the mechanical behavior of thin shells is dominated by their intrinsic curvature, making accurate simulation particularly challenging and computationally demanding.

In this project, we built a finite element method (FEM)-based solver for simulating the elastic deformation of thin shells, centered around the Discrete Shells framework introduced by [1]. Our solver focuses on capturing isometric deformations and exploring a range of material models — from paper-like stiffness to metallic rigidity — allowing us to model a broad spectrum of physical behaviors.

II. Related Works

A. Large timesteps in cloth simulation

The work of Baraff and Witkin (1998) introduced a foundational approach for simulating cloth that remains influential in the field. By formulating the cloth dynamics as a system of differential equations and solving them implicitly, they achieved stability even under large timesteps.

B. Discrete Shells

The Discrete Shells model by [Grinspun et al. 2003]. provides a geometrically intuitive and computationally efficient framework for simulating thin shells. It discretizes the shell surface as a triangle mesh and models the mechanical behavior using discrete analogues of bending and membrane energies. Discrete Shells excels at preserving isometry during deformation and is particularly well-suited for thin, highly flexible materials where thickness is negligible.

III. Results

A. Implementation of Large Steps in Cloth Simulation

Following the approach outlined in *Discrete Shells*, we first implemented a classical shell solver. We adopted the *Large Steps in Cloth Simulation* [2] method due to its simplicity and extensibility, as demonstrated by several other authors. The implementation was carried out using Taichi, with Taichi UI used for visualization.

At this stage, we have not yet implemented constraints. And, we employed a simplified contact model inspired by prior work (citation needed). Since we have not performed a grid search to fine-tune the simulation parameters, some visual artifacts may still be present in the results.

For the implicit solver, we used a modified Preconditioned Conjugate Gradient (PCG) method, as introduced in the original paper.

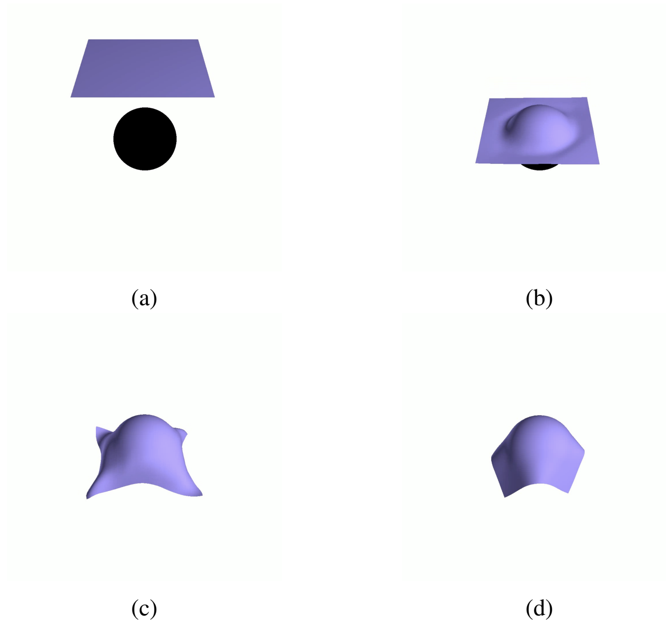


Fig. 1: A typical model of a cloth falling onto a ball

B. Bending & Membrane Energies from Discrete Shells

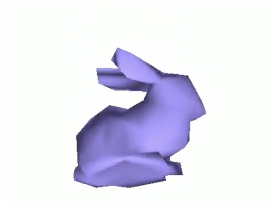
After successfully implementing our framework according to the *Large Steps in Cloth Simulation* [2], we replaced the energy model with that from *Discrete Shells* [1]. Since the calculation of stretching, shearing, and bending energy requires the computation of derivatives and Hessians, we explored several automatic differentiation (autodiff) approaches. However, it turns out that no AD can really achieve stable effects. Thus, we had no choice but to calculate these derivatives using hand-derived formulae [3].

C. Floating Point Precision

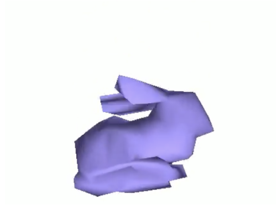
In our experiments, we realised that the precision of floating numbers could significantly influence the result of our simulation, especially when the time step is small or the stiffness coefficient is large. In order to verify our hypothesis, we designed another test with a more complex bunny model and a bigger $K_{bending} = 300.0$.

IV. Optimizations and Comparisons

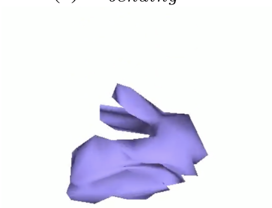
a) Implementation of IPC: Then, we attempt to adopt the Incremental Potential Contact (IPC) framework to simulate the bouncing behavior between two Stanford Bunnies — a more



(a) $K_{bending} = 4$



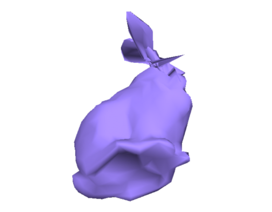
(b) $K_{bending} = 0.4$



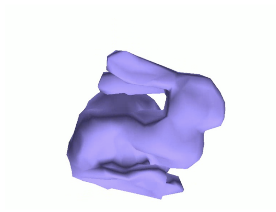
(c) $K_{bending} = 0.04$

(d) $K_{bending} = 0.004$

Fig. 2: Falling bunnies with different bending stiffnesss



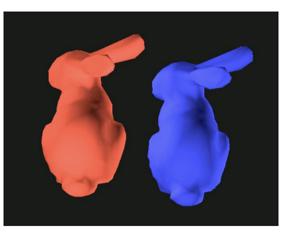
(a) $dt = 5e^{-4}$, using float



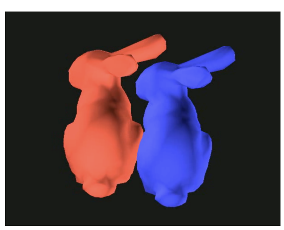
(b) $dt = 5e^{-4}$, using double

Fig. 3: Falling bunnies with different floating point precisions

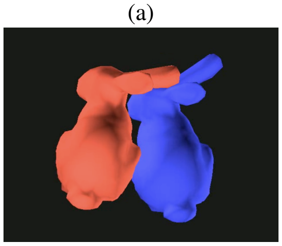
challenging and contact-dense scenario. IPC is particularly appealing for our case because it enables robust, intersection-free collision handling through a carefully designed barrier energy formulation.



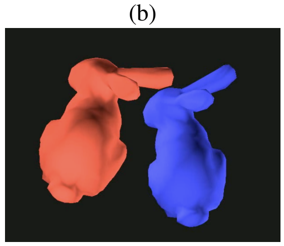
(a)



(b)



(c)



(d)

Fig. 4: Bunnies colliding with IPC energy

To achieve this, we use the following collision energy model:

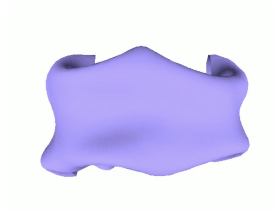
$$b(d, \hat{d}) = \begin{cases} -(d - \hat{d})^2 \ln\left(\frac{d}{\hat{d}}\right), & 0 < d < \hat{d} \\ 0, & d \geq \hat{d} \end{cases}$$

where d is the current distance between two potentially colliding primitives (e.g., vertex-face or edge-edge), and $\hat{d}$ is the activation distance threshold.

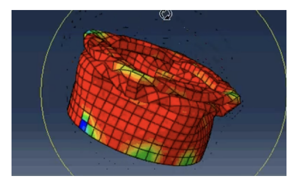
We perform a line search over both the time step Δt and the contact stiffness parameter κ, in order to find values that minimize the total potential energy while ensuring that no intersections occur during the simulation. Yet we admit non-fully successful implementation due to time constraints and being unfamiliar to taichi grammar.

b) Comparison with Abaqus: To test the accuracy of our FEM solver, we constructed a model of a lidless can, subjected it to compression in our simulator, and compared the results with those obtained from a more advanced FEM tool—Abaqus.

For convenience, we applied a force constraint in the Abaqus simulations instead of a velocity constraint. Although our simulation contains some unrealistic aspects, we observed the same cyclic indentation along the edge, which validates the key deformation behavior and supports the credibility of our solver. [1]



(a) ours



(b) Abaqus

Fig. 5: Comparison between ours & Abaqus

V. Conclusions

We developed a shell solver using Taichi based on the Discrete Shells formulation. To validate the correctness of our implementation, we conducted a series of experiments and compared the performance of different numerical methods. Our results demonstrate that using double-precision floating-point arithmetic becomes crucial as the number of vertices increases, due to accumulated numerical errors.

Future improvements include supporting a wider variety of materials and incorporating techniques from recent research to enhance both the accuracy and performance of the solver and also incorporation of IPC.

References

[1] E. Grinspun, A. N. Hirani, M. Desbrun, and P. Schröder, “Discrete shells,” in *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA ’03. Goslar, DEU: Eurographics Association, 2003, p. 62–67.

[1] The reulte of abaqus simulation is borrowed from https://www.youtube.com/watch?v=29XRT5USQxo&themeRefresh=1

[2] D. Baraff and A. Witkin, "Large steps in cloth simulation," in *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '98. New York, NY, USA: Association for Computing Machinery, 1998, p. 43–54. [Online]. Available: https://doi.org/10.1145/280814.280821

[3] R. Tamstorf and E. Grinspun, "Discrete bending forces and their jacobians," *Graphical Models*, vol. 75, no. 6, pp. 362–370, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1524070313000209