Simulation of deformation and rebound of an inflated elastic ball

Midterm progress report

Siyuan Chen, Xiangzhou Kong, Long Chen

1 Background

The elasticity and pressure of an inflated ball have an impact on the sports games or other aspects. For example, the difficulty to catch a football passed by a quarterback; the distance a soccer ball can fly when kicked; the exact amount of force should be applied when a point guard is trying to make a bounce pass, etc.

Studying the model of ball bouncing can help us to predict and simulate how a elastic spherical body deforms and bounces through computing and simulation. A theoretical model for spherical ball bouncing has been established by Stronge [6], with the assumption that ball shells are thin and the internal gas pressure is high enough such that the reaction due to shell bending is insignificant in comparison with the gas pressure.

On the other hand, the FEM analysis of Bao [1], neglecting the internal gas pressure, only considered the bending of the shell when the ball gets stiffer.

2 Potential impact

With discrete simulation methods, it is possible for us to take both gas pressure and shell bending into consideration. Moreover, factors like friction with the ground, other reaction forces from the ground, damping and aerodynamic drag can be included in our simulation.

We utilized discrete elastic rod model for 2D analysis and simulation in phase 1 of this project. 3D discrete shell model will be utilized in phase 2 of this project. While the detailed methodology will be explained in the following section, this project including the algorithm can be a prototype of analysis and simulation of objects in reality with complex geometry. Also, the simulation of the mechanical behavior of flexible structures including elastic objects such as basketballs, tires or inelastic objects such as aluminum cans or hats can be applied to animation and computer-generated imagery in movies [4].

3 Methodology

In current stage, we are using a 2-D circular structure to emulate a 3-D spherical shell. Discrete elastic rods [3] methods are used for our simulation.

3.1 Time marching scheme

Backward Euler method is used to march $\underline{x}$ starting from the initial position.

3.1.1 Discrete rod formulation

From Newton's second law, $F_i = m_i \ddot{q}_i$, we develop:

$$f_i = m_i \frac{q_i(t_{k+1}) - q_i(t_k)}{dt^2} - m_i \frac{\dot{q}_i(t_k)}{dt} + \frac{\partial}{\partial q_i}(E_i^b + E_i^s) - F_i^e$$

where, f_i, the reaction force, should be 0 for each degree of freedoms that is not constrained.

In this formulation,

- E_i^b is the bending energy, defined as $\frac{1}{2}EI(\phi_i - \phi_{i0})^2/dl$. This expression is derived in class;
- E_i^s is the stretching energy, defined as $\frac{1}{2}EA\epsilon_i^2/dl$. This expression is derived in class;
- F_i^e is the external force, which will be discussed in later sections.

3.1.2 Newton-Raphson iteration

We use Newton-Raphson iteration to solve each time step:

$$\underline{q} := \underline{q} - \underline{\underline{J}}/\underline{f}$$

where, J, the Jacobian, is calculated by $J_{ij} = \frac{\partial f_i}{\partial q_j}$. The Jacobian is composed by the Hessian matrices of elastic energies in each section of the structure, and the Jacobian of different external forces that may depends on $\underline{x}$.

The Jacobian is pre-computed with computer algebra system, leaving scalar parameters to be specified at run-time.

3.2 Surface contact

A predictor-corrector method is used to handle ground surface contact.

Assume a ground surface at $y = 0$, When doing time-marching, on each time step:

1. Compute $\underline{q}(t)$ as before;
2. Check if there exists any y-direction DOF i such that $q_i < 0$. If there is any, set it as a temporarily constrained DOF with $y = 0$, and recompute current time step;
3. Check if there exists any temporarily constrained DOF, such that the normal force between the surface and the node is negative (i.e. $f_i < 0$). Remove such temporary constraint, and recompute current time step.

3.3 Other forces

External force $F_i^e = F_i^p + F_i^g + F_i^d + F_i^f$

- F_i^p is discretized uniform force along rods, it is used to simulate inflation pressure effect of a 3-D ball;
- F_i^g a constant gravity force $F_i^g = m_i g$;
- F_i^d is damping force that may depends on q_i and $\dot{q}_i$;
- F_i^f is frictional force that may depends on q_i and $\dot{q}_i$.

3.3.1 Damping

Damping force can be used to remove unphysical oscillations from the structure.

The most simple local damping force formation would be proportional to local velocity, i.e. $F_i = -cu_i$. However, as our structure may be translating in a high bulk-velocity, such damping force is not suitable.

Using a damping force proportional to velocity is a good idea to remove unphysical oscillations. To get rid of the effect of high-velocity bulk motion, we simply subtract the bulk velocity, which is represented by the velocity of the center of mass, i.e. $F_i^s = -c(u_i - u_{cm})$.

Clearly, when the circular structure is translating with is no shape changes, no parts of the structure will receive any damping force; damping forces are only applied when deformation actually happens. Since our structure is axisymmetric, the velocity relative to the center of mass would have the same effect for different nodes.

This damping force is found effective to reduce high-frequency oscillations in our results, without slowing down the bulk motion.

3.3.2 Friction

Ideally, we expect frictional forces to follow $F_i^f = \mu f_i$, i.e. the discretized frictional force at each node is proportional to the normal force between the node and the ground surface.

To reduce numerical instability and guarantee convergence, we use the following approximations:

- When calculating $F^f(t_{k+1})$, the velocity in last time step $\dot{q}(t_k)$ is used, instead of current velocity $\dot{q}(t_{k+1})$;
- Frictional force is reduced when velocity is very low

3.4 Adaptive time stepping

Adaptive time stepping is a effective way to achieve stability with relatively low computational cost [5].

Our adaptive scheme limits the momentum change per time step on each node. The momentum change is effectively calculated by the reaction force f multiplied by time step size dt. Once the actual fdt goes above our threshold, we decrease the step size and recompute; when actual fdt is below the threshold, we increase the step size.

We examine the ratio of actual fdt can our threshold, to determine how much do we need to increase/decrease the step size. Instead of exactly following the ratio, we tend to be more conservative when increasing the step size, so that we can reduce the number of retries due to over-aggressive step sizes.

4 Tools and logistics

Currently, we write the DER code in Python, and the library NumPy is used for carrying out linear algebra. We may seek performance improvement by rewriting the code in C++ instead.

The library SymPy is used for carrying out symbolic differentiation.

We developed a visualization console for displaying our results, which is based on Three.js.

5 Results

The objectives in our project proposal is largely finished.

We are able to simulate a circular structure go under deformation with our code. Pressure and damping forces are taken into account and produced physically sensible results. Surface contact with friction is taken into consideration realistically. With surface friction, the rotation while bouncing is well simulated.

Our assumption that a higher pressure will allow a ball to bounce higher is verified by the simulation.

6 Challenges and ongoing work

6.1 Self-intersection prevention

Under strong impact force and large deformation, our self-intersection is found to occur. Currently, we did not prevent the structure from self-intersection.

Baraff [2] described a solution of inserting a damped spring force to push nodes apart. It is also mentioned that a coherency-based bounding box approach can be used to detect such self-intersection efficiently.

6.2 3-D sphere with discrete shell

After finishing the 2-D scenario, we are going to apply discrete shell methods to simulate a real 3-D spherical shell in the following month.

There are a few challenges:

- Generating meshes for ball that are suitable for discrete shell methods;
- Reducing computational time when the number of nodes are huge.

6.3 Fluid-structure interaction

Fluid-structure interaction will be relevant in two different ways:

- Internal inflation pressure: for a ball, when the internal volume is changed due to shell deformation, the internal air pressure will no longer be constant. However, the air compression is not necessarily isentropic, and the relationship between volume and pressure will be complicated and path-dependent.
- External drag force: when a ball is flying in the air, aerodynamic drag force is not alway negligible. Considering the ball might be spinning, this can be relatively complicated.

References

[1] Bao, R., and Yu, T. Collision and rebound of ping pong balls on a rigid target. *Materials & Design 87* (2015), 278 – 286.

[2] Baraff, D., and Witkin, A. Large steps in cloth simulation. In *SIGGRAPH 98 Conference Proceedings* (1998), pp. 43–54.

[3] Bergou, M., Wardetzky, M., Robinson, S., Audoly, B., and Grinspun, E. Discrete elastic rods. In *ACM SIGGRAPH 2008 Papers* (2008), SIGGRAPH '08, pp. 63:1–63:12.

[4] Grinspun, E. *A Discrete Model of Thin Shells*. Birkhäuser Basel, 2008, pp. 325–337.

[5] Söderlind, G., and Wang, L. Adaptive time-stepping and computational stability. *Journal of Computational and Applied Mathematics 185*, 2 (2006), 225–243.

[6] Stronge, B., and Ashcroft, A. Large deflections during bounce of inflated balls. In *The Engineering of Sport 6* (2006), Springer New York, pp. 109–114.