Ellipsoid Collision as a Linear Complementarity Problem

Izak Oltman, Jaewon Hur

September 4, 2017

Contents

1 Introduction

This report finalizes a project under a Research Experience for Undergraduates (REU) program at the University of Michigan, Ann Arbor. We worked under Professor Eduardo Corona, tasked to build an algorithm aimed to avoid rigid body penetration in physics simulations. There are many applications for such an

application. One example arises when trying to design a truck to drive over sand. To accurately model this scenario, you would need to write code to keep track of each particle of sand making sure no particle intersects, and what forces are being exerted where. Another example would be modeling particles suspended in a fluid, such as bacterium or iron particles. In this case not only must each particle be tracked, but any movement of a particle will affect all other bodies through the fluid medium. Such an algorithm to handle high number of particles would need to be robust and accurate. This problem has been tackled many different ways, one of which is to pose it as a Linear Complementarity Problem. Our algorithm, which works for rigid ellipsoids, will determine at each discrete time step what contact forces are required to ensure no two bodies penetrate each other. We kept the mobility matrix as user provided so that this algorithm could ideally be used as a module in a fluid simulation. After this, we added friction to the model, still as a linear complementarity problem.

2 Outline of Algorithm

Here is a rough outline of the algorithm for each time step

1. We are given body information (position and orientation) along with the velocity information in the next time step without contact forces
2. From given information, construct a matrix (A) and a vector (b) to model the linear complementarity problem (LCP) $0 \leq A\lambda + b \perp \lambda \geq 0$
3. Use a LCP (in our case the Newton Minimum Map Method) solver to get λ, which contains the components of the contact impulses and torques required to avoid intersection
4. Return a vector containing the contact impulses and torques required to avoid intersection

3 Preliminary Notations

3.1 Coordinates

The center of mass of each ellipsoid is stored as vector $r \in \mathbb{R}^3$, which is in cartesian coordinates. The orientation of each body is stored interchangeably as a normalized quaternion $\theta \in \mathbb{R}^4$ or by rotations about each axes $\bar{\theta} = (\theta_x\ \theta_y\ \theta_z)^t$.

3.2 Quaternions

The formulation of quaternions is taken from [3]. Each ellipsoid quaternion represents rotation required to return the ellipsoid to it's principle orientation (ie on its principal axes). This rotation is characterized by a rotation of ϕ radians about a vector α. Then $\theta = (cos(\phi/2),\ sin(\phi/2)\alpha^t)^t$. A quaternion $\theta = (s, p)$ with $p \in \mathbb{R}^3$ can be tranformed into rotation matrix R_θ via

$$R_\theta = 2(pp^t + s\mathscr{C}_p + (s^2 - 1/2)\mathbb{I}_3) \tag{1}$$

Where $\mathscr{C}_p v = p \times v$ for any arbitrary 3-vector v and $\mathbb{I}_3$ is a 3 by 3 identity matrix. To advance a quaternion given a angular velocity $\omega \in \mathbb{R}^3$, we have

$$\theta^{n+1} = \theta^n + \psi\omega\Delta t \tag{2}$$

With

$$\psi = \frac{1}{2}\begin{pmatrix} -p^t \\ s\mathbb{I}_3 - \mathscr{C}_p \end{pmatrix} \tag{3}$$

It is worth noting that quaternions are not used in the formulation of this LCP to determine contact forces, they are only used when advancing the bodies.

3.3 Ellipsoids

The boundary of an ellipsoid centered at the origin along its principal axes with axes $a_1, a_2, a_3 \in \mathbb{R}_{>0}$ is defined as

$$\{x \in \mathbb{R}^3 : \frac{x_1^2}{a_1^2} + \frac{x_2^2}{a_2^2} + \frac{x_3^2}{a_3^2} - 1 = 0\} \tag{4}$$

Which can be written generally as:

$$\{x \in \mathbb{R}^3 : x^t \Lambda x - 1 = 0\} \tag{5}$$

With

$$\Lambda = \begin{pmatrix} 1/a_1^2 & & \\ & 1/a_2^2 & \\ & & 1/a_3^2 \end{pmatrix} \tag{6}$$

Now suppose this ellipsoid is oriented by the quaternion θ and centered at r, then letting $R_\theta \in \mathbb{R}^{3\times 3}$ be the corresponding rotation matrix, then the ellipsoid boundary is defined as

$$\{x \in \mathbb{R}^3 : (x-r)^t R_\theta \Lambda R_\theta^t (x-r) - 1 = 0\} \tag{7}$$

$$= \{x \in \mathbb{R}^3 : (x-r)^t A (x-r) - 1 = 0\} \tag{8}$$

4 The LCP

The purpose of this algorithm is to, at every time step, determine what contact forces are required to avoid intersections of all ellipsoid pairs. For simplicity suppose the system only contains two ellipsoids. This can be posed as the nonlinear complementarity problem (NCP):

$$0 \leq \Phi \perp \lambda \geq 0 \tag{9}$$

Where Φ is the distance between two bodies, λ is the contact force, and $a \perp b$ means that $a \cdot b = 0$. This nonlinear complementarity problem can be approximated as the linear complementarity problem (LCP):

$$0 \leq \Phi^{n+1} \perp \lambda \geq 0 \tag{10}$$

Where the $n+1$ superscript represents the distance between two bodies in the next time step. By linearizing the distance function Φ, we can get

$$\Phi^{n+1} \cong \Phi^n + dt(\frac{\partial \Phi}{\partial q_i})^t u_i^{n+1} + dt(\frac{\partial \Phi}{\partial q_j})^t u_j^{n+1} \tag{11}$$

The subscripts i and j are the indices of the two bodies that are in danger of intersecting. $(\frac{\partial \Phi}{\partial q})$ is a vector containing the partial derivatives of the distance function with respect to changes in the bodies six orientations $(dr_x, dr_y, dr_z, d\theta_x, d\theta_y, d\theta_z)$. u is the velocity of the body, again a 6 dimensional vector containg both translational and angular velocity information and equals $(\dot{r}^t, \omega^t)^t$. dt is the step size for the time discretization. The velocity needs to be linearized as well as

$$u^{n+1} \cong u^n + \mathscr{M} F \tag{12}$$

Where $\mathscr{M}$ represents the mobility matrix of a body, which inputs impulse forces and torques, and outputs translational and angular velocities. In the case of just rigid body collisions it is the diagonal matrix.

$$\begin{pmatrix} m^{-1} & & & \\ & m^{-1} & & \\ & & m^{-1} & \\ & & & I^{-1} \end{pmatrix} \tag{13}$$

with m as the mass and I the moment of inertia. $F \in \mathbb{R}^6$ represents the force and torque impulses applied to the body, which can be broken up into external and contact impulses:

$$F = F_{external} + F_{contact} \tag{14}$$

$$F_{contact} = \mathscr{N}\ \lambda \tag{15}$$

Where $\mathscr{N} \in \mathbb{R}^6$ is the direction of the contact force and torque. It will later be shown that $\mathscr{N} = (\frac{\partial \Phi}{\partial q})$. Now letting the velocity of the body in the next time step be denoted $V = u^n + \mathscr{M} F_{external}$, our LCP then becomes, after dividing by dt:

$$0 \leq ((\frac{\partial \Phi}{\partial q_i})^t \mathscr{M}_i \mathscr{N}_i + (\frac{\partial \Phi}{\partial q_j})^t \mathscr{M}_j \mathscr{N}_j)\lambda + (\frac{\Phi^n}{dt} + (\frac{\partial \Phi}{\partial q_i})^t V_i + (\frac{\partial \Phi}{\partial q_j})^t V_j) \perp \lambda \geq 0 \tag{16}$$

This then can be modified to model an arbitrary number of bodies with an arbitrary number of collisions. The new formulation will be

$$0 \leq (\partial \Phi)^t\ \mathscr{M}\ \mathscr{N}\ \lambda + (1/dt)(\Phi^n + \varepsilon) + (\partial \Phi)^t\ V \perp \lambda \geq 0 \tag{17}$$

Given m collisions, with n bodies, and suppose collision i refers to bodies indexed by a and b, then we have $\partial \Phi \in \mathbb{R}^{6n \times m}$, with

$$\partial \Phi = \begin{array}{c} \\ \scriptstyle ... \\ \scriptstyle body_a \\ \scriptstyle ... \\ \scriptstyle body_b \\ \scriptstyle ... \end{array} \begin{array}{c} \begin{array}{ccc} \scriptstyle ... & \scriptstyle collision_i & \scriptstyle ... \end{array} \\ \begin{bmatrix} ... & & ... \\ ... & (\frac{\partial \Phi}{\partial q_a}) & ... \\ ... & & ... \\ ... & (\frac{\partial \Phi}{\partial q_b}) & ... \\ ... & & ... \end{bmatrix} \end{array} \tag{18}$$

$\mathscr{M} \in \mathbb{R}^{6n \times 6n}$ is the mobility matrix. In the case of just rigid body motion, $\mathscr{M} = diag(\mathscr{M}_n\ ...\ \mathscr{M}_n)$
$\mathscr{N} \in \mathbb{R}^{6n \times m}$ is the direction of the contact forces, and will be proven to just be $(\partial \Phi)$.
$\lambda \in \mathbb{R}^m$ are all the contact forces, where the i^{th} entry is λ_i.
$\Phi^n \in \mathbb{R}^m$ are the distance of body pairs, where the i^{th} entry is Φ_i^n
$\varepsilon \in \mathbb{R}^m$ is a fictitious body barrier (tolerance) around each ellipsoid, which must be positive to model collisions.
$V \in \mathbb{R}^{6n}$ is the velocity information of all bodies in the next time step without any contact force, where the $6 \cdot a^{th}$ entry is V_a.

5 Distance functions and their Jacobians

With the setup for the LCP done, we now must determine the distance function Φ between ellipsoid - ellipsoid pairs, and ellipsoid - boundary pairs, along with computing their Jacobians with respect to changes in orientation.

5.1 Ellipsoid - Ellipsoid Distances

We used the ball method proposed in [1]. This is an iterative function that essentially approximates the ellipsoids by inscribed spheres. This method is ideal for ellipsoid collisions as it determines the distance and the two closest points on each ellipsoid which are used as the points of contact and as a guess for the iteration in the next time step.

5.2 Ellipsoid - Plane Distance

Consider an ellipsoid E and a plane P. Define an ellipsoid by the associated quaternion, three axes lengths, and center coordinates. Define a plane by a point on the plane and the vector normal.
We shift via converting the quaternion to a rotation matrix, and rotating and shifting both the plane and ellipsoid so we may rewrite E into the general form f,

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$$

and a plane with unit vector (n_1, n_2, n_3). Provided that the ellipsoid and the plane do not intersect, we may find the points closest to each other via the Lagrange Multiplier method.

$$\begin{aligned}
\nabla f &= \left(\frac{2x}{a^2}, \frac{2y}{b^2}, \frac{2z}{c^2}\right) \\
&= \lambda(n_1, n_2, n_3) \\
\implies &\begin{cases} x &= \frac{a^2\lambda n_1}{2}, \\ y &= \frac{b^2\lambda n_2}{2}, \\ z &= \frac{c^2\lambda n_3}{2} \end{cases} \\
\implies 1 &= \left(\frac{a^2\lambda n_1}{2}\right)^2 \cdot \frac{1}{a^2} + \left(\frac{b^2\lambda n_2}{2}\right)^2 \cdot \frac{1}{b^2} + \left(\frac{c^2\lambda n_3}{2}\right)^2 \cdot \frac{1}{c^2} \\
\implies \lambda &= \pm\sqrt{\frac{4}{a^2n_1^2 + b^2n_2^2 + c^2n_3^2}}
\end{aligned}$$

and we may plug this back into our original system of equations to find the appropriate x, y, z coordinates to find our desired point $\ell = (\ell_1, \ell_2, \ell_3)$ on the ellipsoid which is closest to the plane. To continue we extend the line from the ellipsoid point to the plane.

Let $p = (p_1, p_2, p_3)$ represent the point on the plane that is now shifted. From the unit normal vector, we know that our plane takes the equation

$$\begin{aligned}
n_1x + n_2y + n_3z + d &= 0 \\
\implies d &= -(n_1p_1 + n_2p_2 + n_3p_3)
\end{aligned}$$

and once we find d, we have our new plane equation. Now, consider the normal line along the point we found earlier on our ellipsoid defined in parametric form by

$$\begin{cases} x &= \ell_1 + n_1t \\ y &= \ell_2 + n_2t \\ z &= \ell_3 + n_3t \end{cases}$$

and we substitute this into our recently obtained equation

$$n_1x + n_2y + n_3z + d = 0$$

and solve for t, to obtain

$$t = \frac{-n_1\ell_1 - n_2\ell_2 - n_3\ell_3 - d}{n_1^2 + n_2^2 + n_3^2}$$

and then we may plug t back into the system of equations for the line to obtain the point on the plane closest to the ellipsoid.

5.3 Ellipsoid - Sphere Distances

Next we want to determine the closest distance of an ellipsoid in a sphere to the boundary of the sphere. Through translations and rotations, this can be simplified to the case where the ellipsoid is centered at the origin, and the axes are aligned along its principle axes. Our ellipsoid boundary is then defined as

$$\{x \in \mathbb{R}^3 \mid x^t\Lambda x - 1 = 0\} \tag{19}$$

$$\Lambda = \begin{pmatrix} 1/a_1^2 & & \\ & 1/a_2^2 & \\ & & 1/a_3^2 \end{pmatrix} \tag{20}$$

Now instead of finding the closest distance from the ellipsoid to the boundary, it is equivalent to find the furthest distance on the ellipsoid from the center $c \in \mathbb{R}^3$. This minimization problem can be posed as a Lagrange multiplier:

$$x - c + \lambda\Lambda x = 0 \tag{21}$$

$$x^t\Lambda x - 1 = 0 \tag{22}$$

This then gives us

$$x_i = \frac{c_i a_i^2}{a_i^2 + \lambda} \text{ for } i = 1, 2, 3 \tag{23}$$

Using this we can write λ as

$$f(\lambda) = \sum_{i=1}^{3} c_i a_i^2 \prod_{j\neq 1} (a_j^2 + \lambda)^2 - \prod_{j=1}^{3} (a_j^2 + \lambda)^2 = 0 \tag{24}$$

The roots of $f(\lambda)$ will provide 6 Lagrange Multipliers. Then all that is needed is to use equation 23 to find the point corresponding to each real λ, and then select the optimal one. However, the next proposition will prove that this is not needed.

Proposition 1. *Given roots $\{\lambda_1, ..., \lambda_6\}$ of $f(\lambda)$. The λ_i that corresponds to the maximum distance on the ellipsoid from a point will be such that $\lambda_i \in \mathbb{R}$ and $\lambda_i \leq \lambda_j$ for all $j = 1, 2, ..., 6$*

Proof. First recognize that the positive λ's represent local minimums for the Lagrange multiplier problem and should be ignored. A non-rigorous argument is that Λx is the outward normal of the ellipsoid, so if $\lambda > 0$, then $x - c$ and Λx are pointing anti-parallel, which means that λ corresponds to a point on the ellipsoid closest to the center of the sphere.

Let x_i be the point given by Lagrange Multiplier $\lambda_i < 0$. Then we have

$$||x_i - c|| = \lambda_i ||\Lambda x_i|| = -\lambda_i \sqrt{\sum_{j=1}^{3} (\frac{x_{i_j}}{a_j^2})^2} \tag{25}$$

$$= -\lambda_i \sqrt{\sum_{j=1}^{3} \frac{c_j^2}{(a_j^2 + \lambda_i)^2}} \tag{26}$$

Next observe that $|x_{i_j}| \leq a_j$, and so

$$|x_i| = |\frac{c_j a_j^2}{a_j^2 + \lambda_i}| \leq a_j \tag{27}$$

Rearranging, we can get $|a_j^2 + \lambda_i| \geq |c_i| a_i \geq 0$ and so if $\lambda_i \leq \lambda_k$ we have:

$$||x_i - c|| = -\lambda_i \sqrt{\sum_{j=1}^{3} \frac{c_j^2}{(a_j^2 + \lambda_i)^2}} \geq -\lambda_k \sqrt{\sum_{j=1}^{3} \frac{c_j^2}{(a_j^2 + \lambda_k)^2}} = ||x_k - c|| \tag{28}$$

□

There are subtletes that can occur in very infrequent cases when there is no unique point on the ellipsoid furthest from the sphere's center. There can be two points, a circle of points, or a sphere of points that solve the Lagrange multiplier problem. In these cases, like when a ellipsoid perfectly hits the bottom of the sphere, there are two points of contact which are equivilent to a point of contact at the center of mass incurring no torque. These cases, while infrequent, are not too difficult to fix as is shown in the appendix.

5.4 Jacobians of the distance functions

For the LCP, we must, for each pair of bodies, not only find the distance between them, Φ, but also determine how this distance changes as each body is shifted and rotated about each axes. That is to say, we must determine $\frac{\partial \Phi}{\partial r_i}$ and $\frac{\partial \Phi}{\partial \theta_i}$, where r_i is the i^{th} axes, and θ_i is a rotation about the i^{th} axes. Suppose we have two ellipsoids characterized by solutions to the two equations $(x - r_1)^t A_1 (x - r_1) - 1 = 0$ and $(y - r_2)^t A_2 (y - r_2) - 1 = 0$. Where $A_1, A_2 \in \mathbb{R}^{3\times 3}$. Now suppose we have determined the closest points on the two ellipsoids, k_1, k_2. Then we have solved the Lagrange multiplier problem:

$$k_1 - k_2 + \lambda A_1 (k_1 - r_1) = 0 \tag{29}$$

$$k_2 - k_1 + \mu A_2 (k_2 - r_2) = 0 \tag{30}$$

$$(k_1 - r_1)^t A_1 (k_1 - r_1) - 1 = 0 \tag{31}$$

$$(k_2 - r_2)^t A_2 (k_2 - r_2) - 1 = 0 \tag{32}$$

Through substitution of the Lagrange multipliers λ and μ, we can reduce this to the following:

$$k_1 - k_2 + (k_1 - r_1)^t (k_2 - k_1) A_1 (k_1 - r_1) = 0 \tag{33}$$

$$k_2 - k_1 + (k_2 - r_2)^t (k_1 - k_2) A_2 (k_2 - r_2) = 0 \tag{34}$$

Now the question is given shifts and rotations (changes to r and A respectively), how do k_1 and k_2 change? Let each variable be equal to itself plus an increment, ie let $k_1 = k_1 + \Delta k_1$. Then expanding the above equation, getting rid of second order approximations, we can get $A(\Delta k_1 \ \Delta k_2)^t = b$, where

$$A = \begin{pmatrix} \mathbb{I}_3 + A_1 c_1 (h^t - c_1^t) + c_1^t h A_1 & -\mathbb{I}_3 + A_1 c_1 c_1^t \\ -\mathbb{I}_3 + A_2 c_2 c_2^t & \mathbb{I}_3 + A_2 c_2 (-h^t - c_2^t) - c_2^t h A_2 \end{pmatrix} \tag{35}$$

$$b = \begin{pmatrix} A_1(c_1^t h \mathbb{I}_3 + c_1 h^t)\Delta r_1 - \Delta A_1 c_1 c_1^t h \\ -A_2(c_2^t h \mathbb{I}_3 + c_2 h^t)\Delta r_2 + \Delta A_2 c_2 c_2^t h \end{pmatrix} \tag{36}$$

Where $\mathbb{I}_n$ is an n by n unit matrix, $c_1 = k_1 - r_1$, $c_2 = k_2 - r_2$, and $h = k_2 - k_1$, $\hat{h} = \frac{h}{||h||}$, and Δr_i is the displacement of the center of the i^{th} ellipsoid's center. ΔA_i (first order approximation of the ellipsoid's rotation) is derived as:

$$\Delta A_i = A_{i,new} - A_i \tag{37}$$

$$\Delta A_i = R_{new} \Lambda R_{new}^t - R \Lambda R^t \tag{38}$$

$$R_{new} = R_\theta R \cong (\mathbb{I}_3 + \mathscr{C}_\theta)R = R + \mathscr{C}_\theta R \tag{39}$$

$$\Delta A_i = A_i \mathscr{C}_\theta^t + \mathscr{C}_\theta A_i \tag{40}$$

Where $\mathscr{C}_\theta \in \mathbb{R}^{3\times 3}$ is such that $\mathscr{C}_\theta \cdot v = \theta \times v$. By solving $A(\Delta k_1 \ \Delta k_2)^t = b$, we can get the change in distance through the approximation

$$\Delta\Phi = \|k_1 + \Delta k_1 - k_2 - \Delta k_2\| \cong \frac{(k_1 - k_2)^t(\Delta k_1 - \Delta k_2)}{\|k_1 - k_2\|} = \hat{h}^t(\Delta k_1 - \Delta k_2) \tag{41}$$

Letting $M_1, M_2 \in \mathbb{R}^{6\times 3}$ be such that $A^{-1} = (M_1 \ M_2)^t$, we have:

$$\Delta\Phi = \hat{h}^t(M_1 - M_2)b \tag{42}$$

After preforming extensive numerical tests, it has been found that this method is equivalent to:

$$\Delta\Phi = (\frac{\partial\Phi}{\partial q_1})^t \Delta q_1 + (\frac{\partial\Phi}{\partial q_2})^t \Delta q_2 \tag{43}$$

with,

$$\frac{\partial\Phi}{\partial q_1} = \begin{pmatrix} -\hat{h} \\ \hat{h} \times c_1 \end{pmatrix}, \frac{\partial\Phi}{\partial q_2} = \begin{pmatrix} \hat{h} \\ -\hat{h} \times c_2 \end{pmatrix} \tag{44}$$

An analytic proof of this fact, which would involve symbolically inverting A, has not been found, but would probably involve some linear algebra tricks. It may be possible to determine the error in this approximation by keeping track of the second and higher order terms that were neglected. It is important to note that errors in calculating the jacobian of the distance function are what lead to penetrating bodies. An error in the jacobian of the distance function could lead to a significant error in the guessed distance between bodies in the next time step. If the guessed distance is larger than the actual, then the next time step could end up with intersecting bodies.
Now that the method of computing the distance Jacobians is established, it is worthwhile to state the following:
Proposition 2. $\frac{\partial\Phi}{\partial q_i} = \mathscr{N}_i$

Proof. Again letting $-h$ be the unit vector from the point of contact on the second ellipsoid (k_2) to the point of contact on the first ellipsoid (k_1). If r is the center of the first ellipsoid, then let $c_1 = k_1 - r_1$. And so the component of the contact force that applies translational force to the center of mass will be $-\hat{h}$, while the component that applies torque will be $\hat{h} \times c_1$. And so we have

$$\mathscr{N}_1 = \begin{pmatrix} -\hat{h} \\ \hat{h} \times c_1 \end{pmatrix} = \frac{\partial\Phi}{\partial q_1} \tag{45}$$

□

6 Adding Friction

Friction can be modeled by approximating the Columb friction cone as a polyhedral with $2l$ edges [2]. This gives us the following equations:

$$0 \leq \Phi^{n+1} \perp \lambda \geq 0 \tag{46}$$

$$0 \leq \gamma e + D^t u_t^{n+1} \perp \beta \geq 0 \tag{47}$$

$$0 \leq \mu\lambda - e^t\beta \perp \gamma \geq 0 \tag{48}$$

Φ and λ are as before. $\gamma \in \mathbb{R}$ is to be solved. $e = (1, 1, ..., 1)^t \in \mathbb{R}^{2l}$. $D \in \mathbb{R}^{3\times 2l}$ is a collection of evenly spaced out unit vectors lying on the plane tangent to the bodies points of contact. If the point of contact is at the origin, and the tangent plane of contact is the x-y plane, and we are approximating the friction cone with 4 sides, then an acceptable D would be

$$\begin{pmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{49}$$

u_t is the relative translational velocity of one body with respect to the other. $\beta \in \mathbb{R}^{2l}$ is the coefficients of the friction force, making the actual friction force $F_{friction} = D\beta$. $\mu \in \mathbb{R}$ is the coefficient of friction.
To formulate this as a matrix for the case of only two ellipsoids, the same approach is followed, except now we have

$$F_{contact} = (\partial_i \Phi)^t \lambda + N \ D \ \beta \tag{50}$$

With

$$N = \begin{pmatrix} \mathbb{I}_3 \\ 0_3 \end{pmatrix} \tag{51}$$

$$\partial_i \Phi = (\frac{\partial \Phi}{\partial q_i}) \tag{52}$$

By multiplying by N, we are ensuring that the friction force only imposes translational force, and no torques, as that would be involve implementing rolling friction.
Furthermore, if bodies index by a and b are in contact, we have

$$u_t^{n+1} = N^t(u_a^{n+1} - u_b^{n+1}) \tag{53}$$

Again multiplying by N^t because we only care about translational velocity and not rotatational. We can then write this LCP in the form $0 \leq Ax + b \perp x \geq 0$, with

$$A = \begin{pmatrix} (\partial_1\Phi)^t \mathscr{M}_1 (\partial_1\Phi) + (\partial_2\Phi)^t \mathscr{M}_2 (\partial_2\Phi) & ((\partial_1\Phi)^t \mathscr{M}_1 - (\partial_2\Phi)^t \mathscr{M}_2) ND & 0 \\ D^t N^t (\mathscr{M}_1 (\partial_1\Phi) - \mathscr{M}_2 (\partial_2\Phi)) & D^t N^t (\mathscr{M}_1 + \mathscr{M}_2) ND & e \\ \mu & -e^t & 0 \end{pmatrix} \tag{54}$$

$$x = \begin{pmatrix} \lambda \\ \beta \\ \gamma \end{pmatrix}, \ b = \begin{pmatrix} (1/dt)(\Phi^n - \varepsilon) + (\partial_1\Phi)^t V_1 + (\partial_2\Phi)^t V_2 \\ D^t N^t V^n \\ 0 \end{pmatrix} \tag{55}$$

Again, we can generalize this for an arbitrary number of collisions by solving $0 \leq Ax+b \perp x \geq 0$, with:

$$A = \begin{pmatrix} (\partial\Phi \ \mathscr{D})^t \mathscr{M} (\partial\Phi \ \mathscr{D}) & A_1 \\ A_2 & 0 \end{pmatrix} \tag{56}$$

If we are approximating the cone with $2l$ edges, have n bodies, and have m collisions. Given a collision indexed by i involving bodies a and b, then
$\mathscr{D} \in \mathbb{R}^{2lm \times 6n}$ where:

$$\mathscr{D} = \begin{array}{c} \\ \scriptstyle ... \\ \scriptstyle body_a \\ \scriptstyle ... \\ \scriptstyle body_b \\ \scriptstyle ... \end{array} \begin{array}{c} \begin{array}{ccc} \scriptstyle ... & \scriptstyle collision_i & \scriptstyle ... \end{array} \\ \begin{bmatrix} ... & & ... \\ ... & N \cdot D_i & ... \\ ... & & ... \\ ... & -N \cdot D_i & ... \\ ... & & ... \end{bmatrix} \end{array} \tag{57}$$

$(\partial\Phi \ \mathscr{D}) \in \mathbb{R}^{6n \times (m+2lm)}$ is the concatenatination of the previously defined matrices $\partial\Phi$ and $\mathscr{D}$

$$A_1 = \begin{array}{c} \\ \scriptstyle ... \\ \scriptstyle collision_i \\ \scriptstyle ... \end{array} \begin{array}{c} \begin{array}{ccc} \scriptstyle ... & \scriptstyle collision_i & \scriptstyle ... \end{array} \\ \begin{bmatrix} ... & & ... \\ ... & e & ... \\ ... & & ... \end{bmatrix} \end{array} \tag{58}$$

$$A_2 = \begin{array}{c} \\ \scriptstyle ... \\ \scriptstyle collision_i \\ \scriptstyle ... \\ \scriptstyle n+collision_i \\ \scriptstyle ... \end{array} \begin{array}{c} \begin{array}{ccccc} \scriptstyle ... & \scriptstyle collision_i & \scriptstyle ... & \scriptstyle n+collision_i & \scriptstyle ... \end{array} \\ \begin{bmatrix} & & & & \\ ... & \mu_i & ... & & \\ & & & & \\ ... & & & -e^t & \\ & & & & ... \end{bmatrix} \end{array} \tag{59}$$

$x = (\lambda_1 \ ... \ \lambda_n \ \beta_1^t \ ... \ \beta_n^t \ ... \ \gamma_1 \ ... \ \gamma_n)^t$ and

$$b = \begin{pmatrix} (1/dt)(\Phi^n + \varepsilon) + (\partial\Phi)^t V \\ \mathscr{D}^t V \\ 0 \end{pmatrix} \tag{60}$$

6.1 Friction Problems

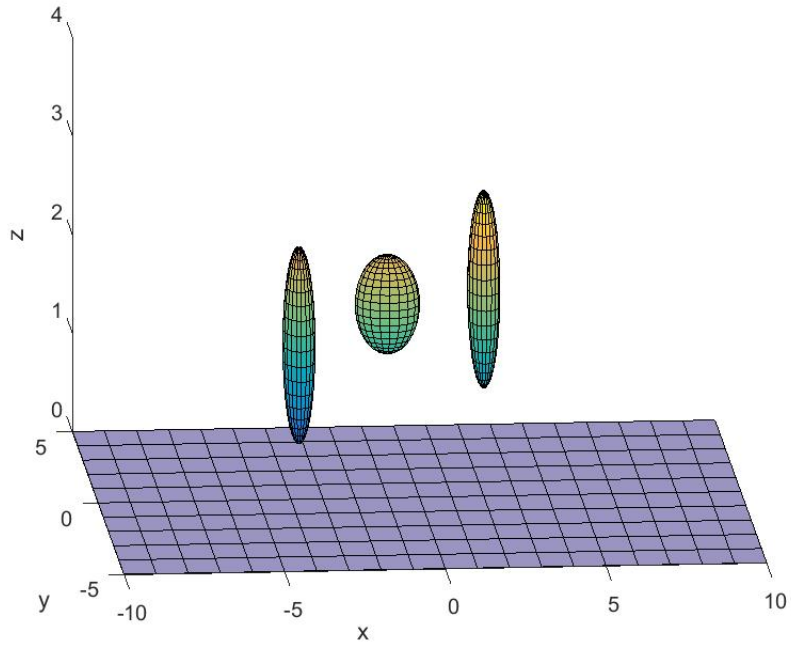
There was a bug that was never fully fixed. The simplest case of this bug was when throwing a sphere tangentially to a plane with gravity and friction. In certain cases the sphere would fall through the plane. Upon inspection this was due to an insufficient contact force from the boundary plane, which was due to the LCP solver not solving the LCP. The Newton Minimal Map solver was failing to converge on a solution and would return an insufficient answer. There are at least three reasons for this, either the LCP was incorrectly formulated leading to an unsolvable LCP, the LCP was formulated incorrectly in the code, or the the Newton Minimal Map solver has a bug. We used a different LCP solver which used a pivoting algorithm which worked perfectly for all friction simulations. The problem is that the pivoting algorithm scales cubically with the number of collision pairs while the Newton Minimal Map scales quadratically.

7 Results

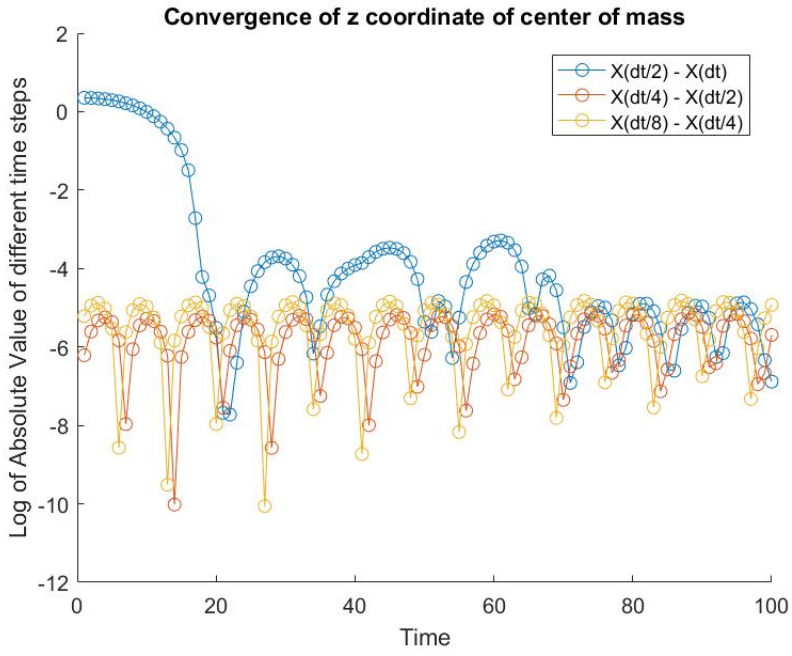
7.1 Self Convergence

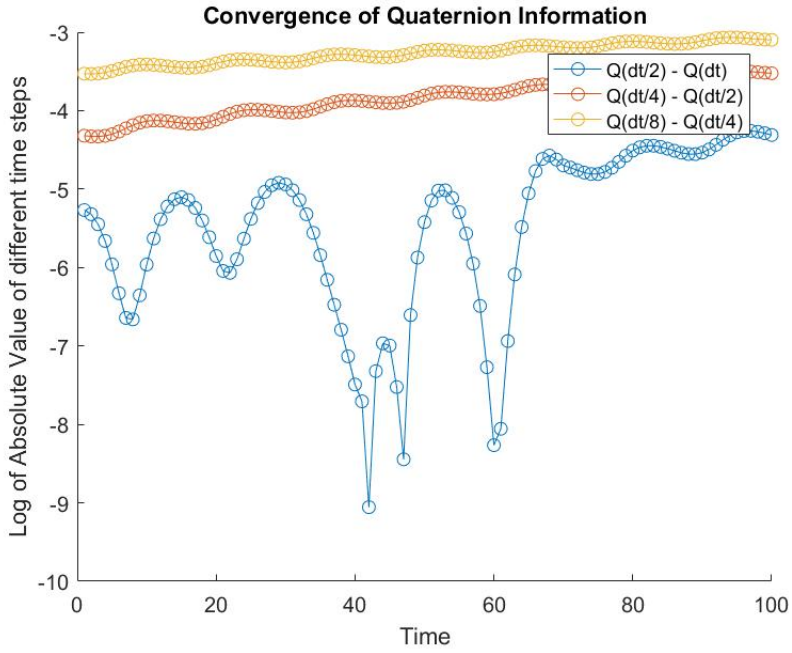
First we determine how our simulations converge with respect to shrinking time step size, dt. To do this we ran the same simulation for step sizes dt, $dt/2$, $dt/4$, and $dt/8$. Then we looked at different values of an arbitrary body: position, orientation, translational and angular velocity. We then plotted the difference between succesive simulations. For the simple simulations, such as dropping a sphere on a plane, the

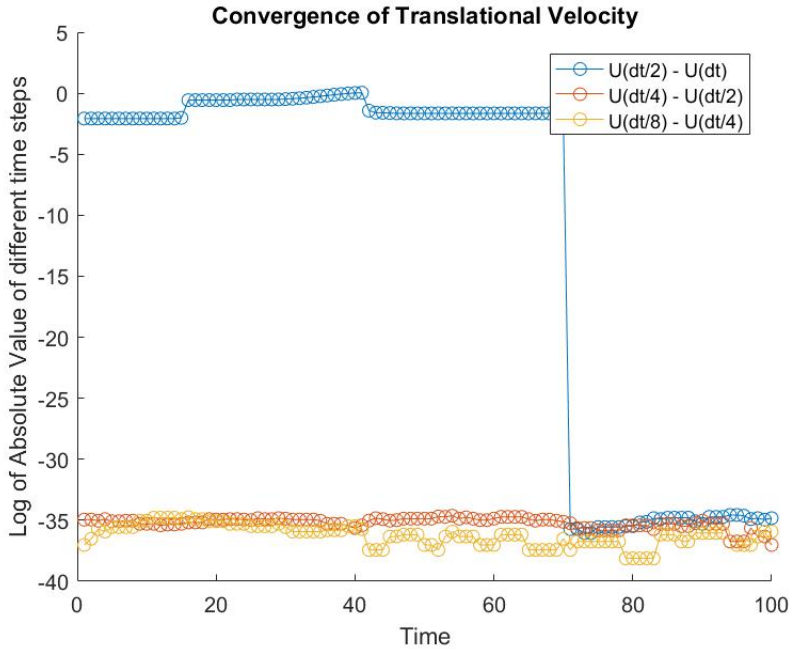
converegence was non-existent as the numerical method could determine the exact solution with a large time step. The first test is taking three different ellipsoids with a plane boundary under gravity with low initial velocities. Here is a snapshot of an early frame of the simulation.

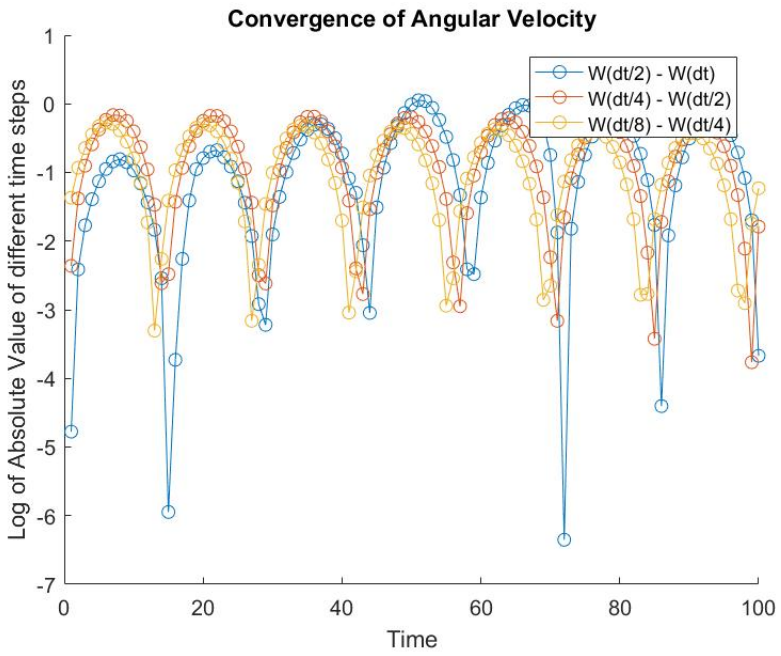


We then got the following results:

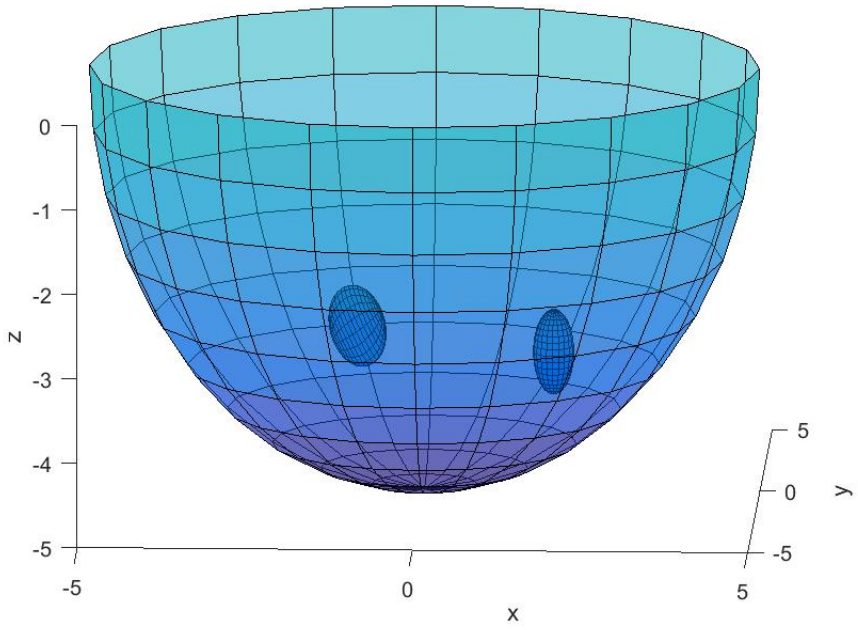



Convergence of Quaternion Information
Q(dt/2) - Q(dt)
Q(dt/4) - Q(dt/2)
Q(dt/8) - Q(dt/4)
Log of Absolute Value of different time steps
Time
-3
-4
-5
-6
-7
-8
-9
-10
0
20
40
60
80
100

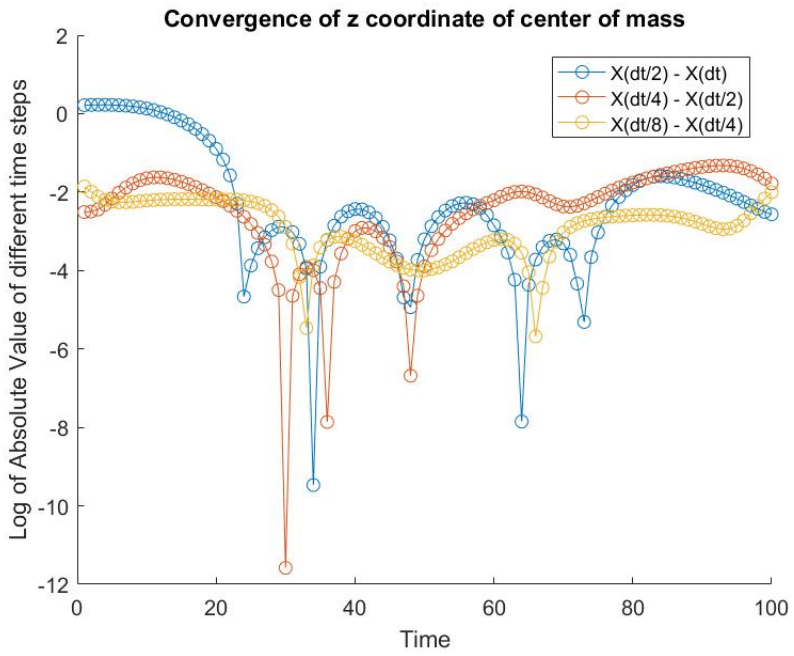

Convergence of Translational Velocity
U(dt/2) - U(dt)
U(dt/4) - U(dt/2)
U(dt/8) - U(dt/4)
Log of Absolute Value of different time steps
Time
5
0
-5
-10
-15
-20
-25
-30
-35
-40
0
20
40
60
80
100

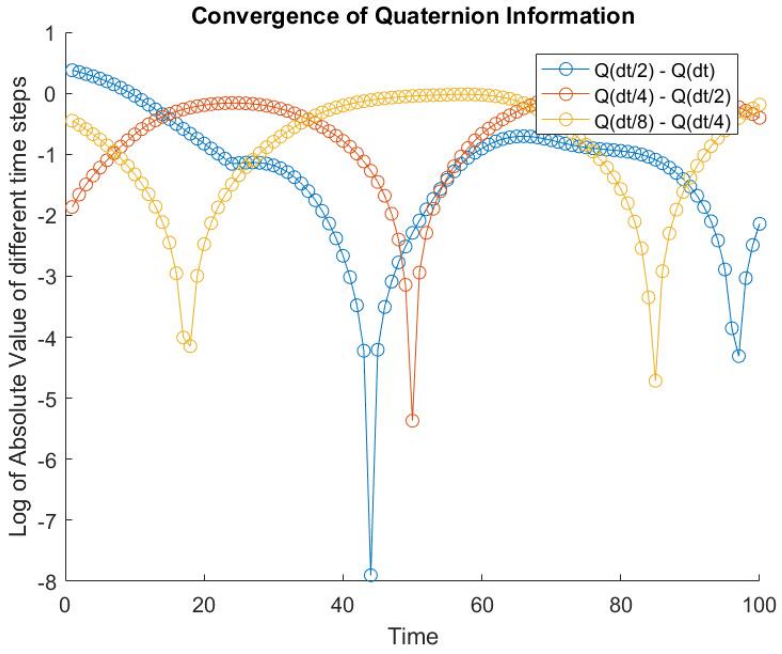


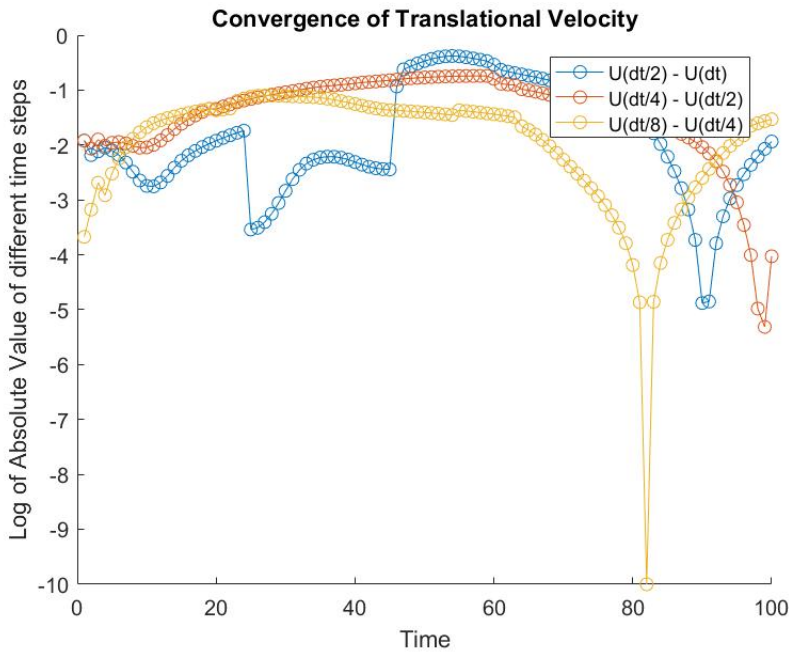
The next test is dropping two ellipsoids within a spherical boundary with gravity. Here is a snapshot of an early frame:

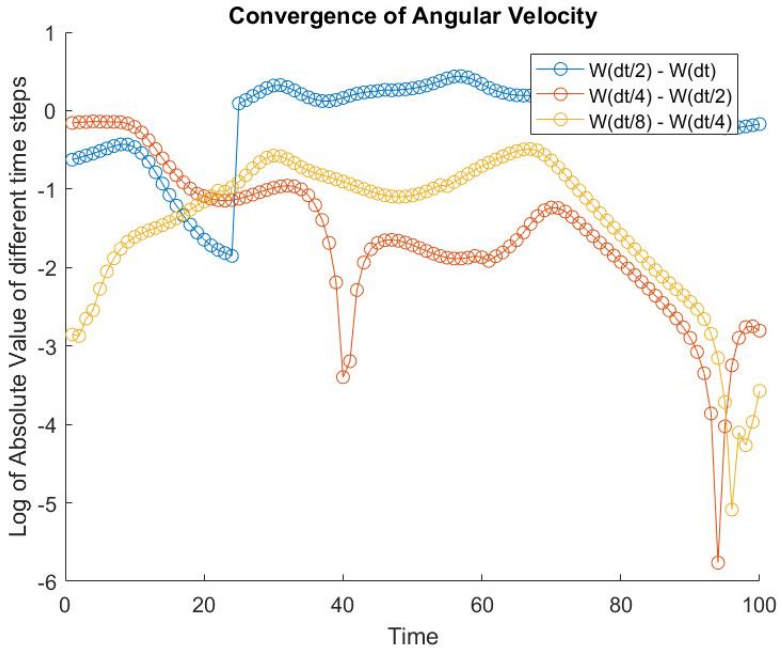


Along with tests:

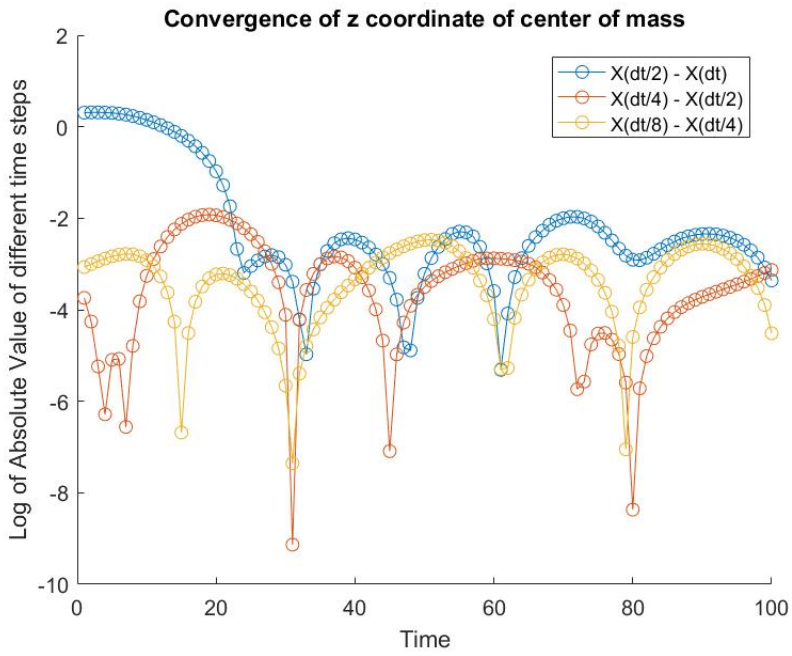

Convergence of z coordinate of center of mass
X(dt/2) - X(dt)
X(dt/4) - X(dt/2)
X(dt/8) - X(dt/4)
Log of Absolute Value of different time steps
Time
2
0
-2
-4
-6
-8
-10
-12
0
20
40
60
80
100

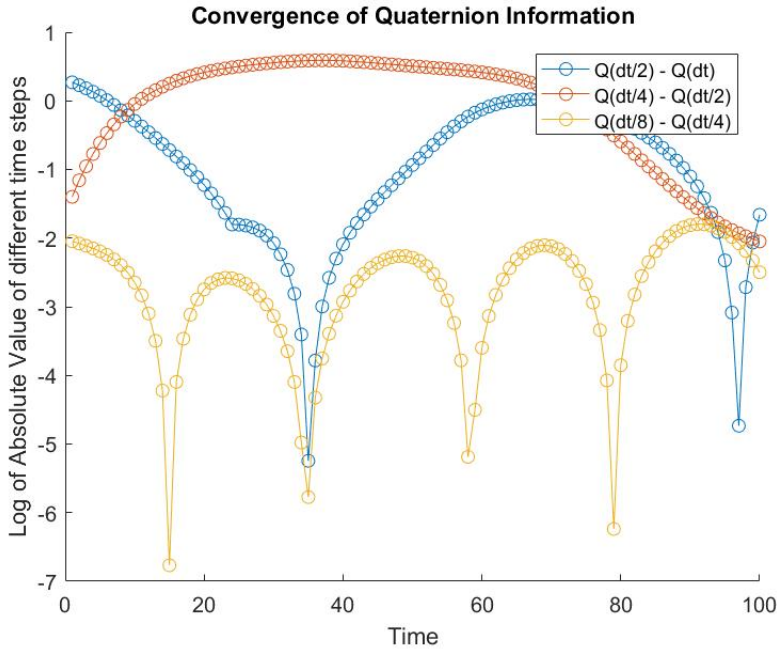

Convergence of Quaternion Information
Q(dt/2) - Q(dt)
Q(dt/4) - Q(dt/2)
Q(dt/8) - Q(dt/4)
Log of Absolute Value of different time steps
Time
1
0
-1
-2
-3
-4
-5
-6
-7
-8
0
20
40
60
80
100

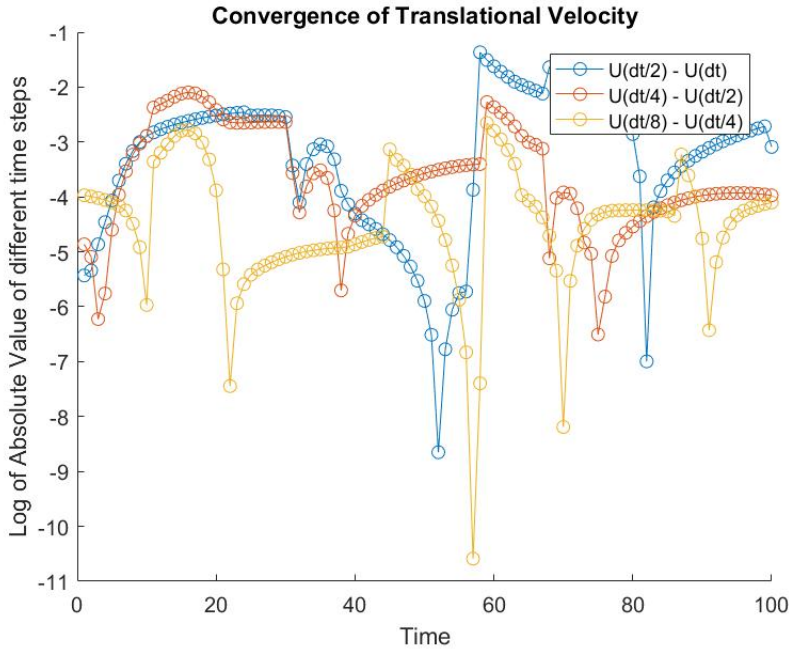


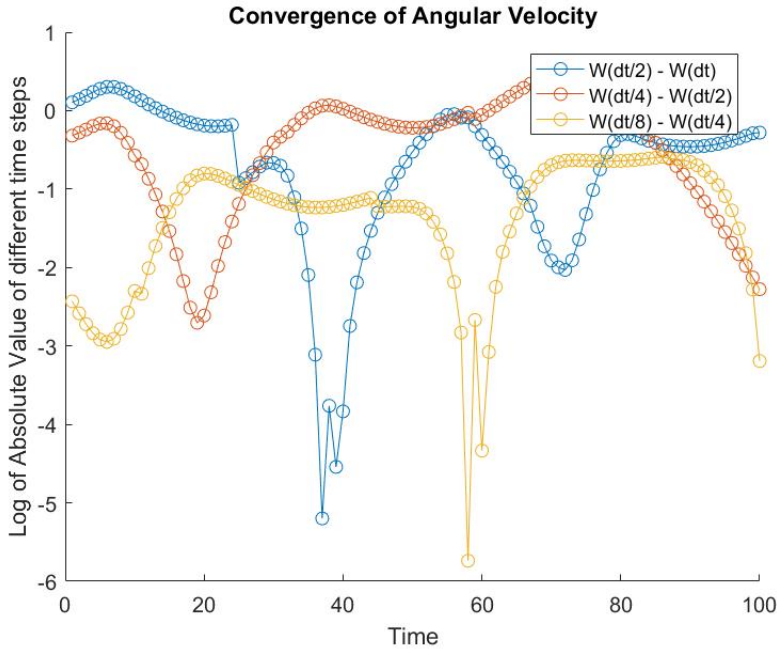


Then the same simulation is run with sliding friction using an arbitrary coefficent of friction .4, and approximating the friction cone with 4 edges. We then get the following results.


Convergence of z coordinate of center of mass
X(dt/2) - X(dt)
X(dt/4) - X(dt/2)
X(dt/8) - X(dt/4)
Log of Absolute Value of different time steps
Time
2
0
-2
-4
-6
-8
-10
0
20
40
60
80
100


Convergence of Quaternion Information
Q(dt/2) - Q(dt)
Q(dt/4) - Q(dt/2)
Q(dt/8) - Q(dt/4)
Log of Absolute Value of different time steps
Time
1
0
-1
-2
-3
-4
-5
-6
-7
0
20
40
60
80
100





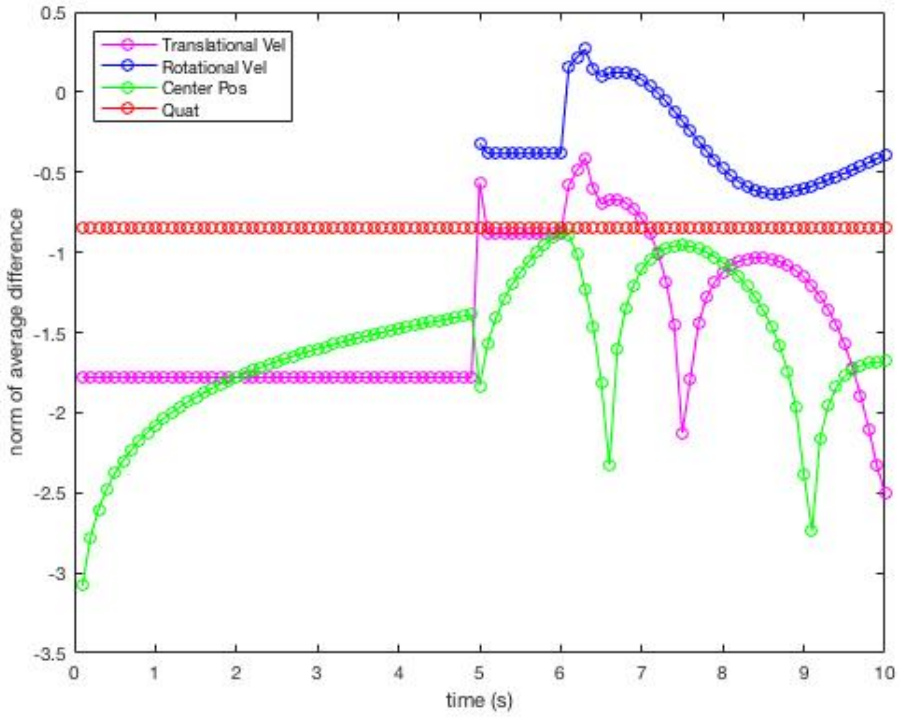
Convergence, while present, is hard to see. By numerically approximating our diffrential equations of motion with a backwards Euler scheme, we would expect linear convergence. However, every contact is a discontinuity, and as the number of contacts increase, convergence is harder to see.

7.2 Comparing Values of dt

We ran tests where we compared two simulations with the same initial position and velocity conditions, but with $dt = 0.1$ and $dt/2$ respectively, and plotted the average norm of the difference between the objects in

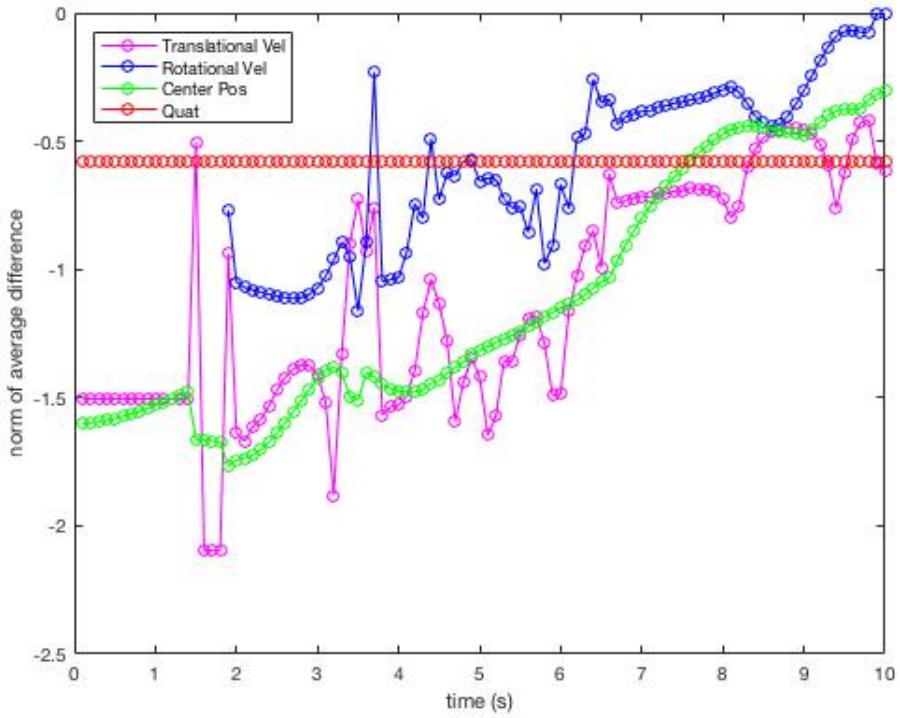
the two simulations at each time step. We shall call an object $O_i = (v_i, p_i)$ where v_i and p_i represent the velocity and position at a given time step, and where $i = 1$ if it corresponds to the simulation using $dt = 0.1$ or $i = 2$ if it corresponds to the simulation using $dt/2$. We found the following results:

7.2.1 Ellipsoid Falling onto Plane



This is the example where an ellipsoid falls onto a plane. The quaternion, translational velocity and the center position converge fairly well, as we can see, but the rotational velocity seems to have trouble at points of contact.

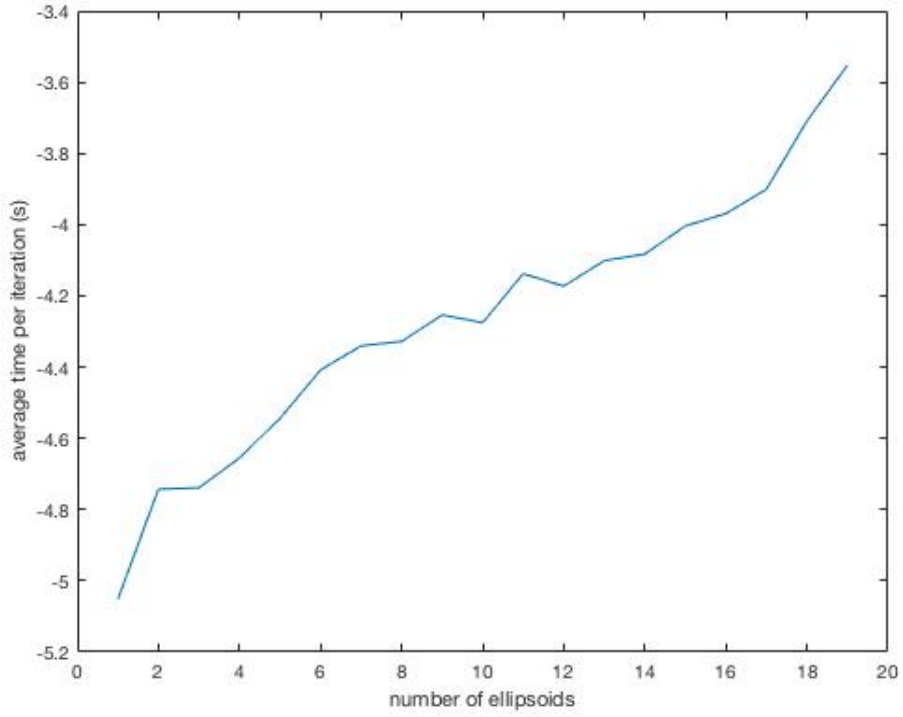
7.2.2 Ellipsoids Arranged in a Cube



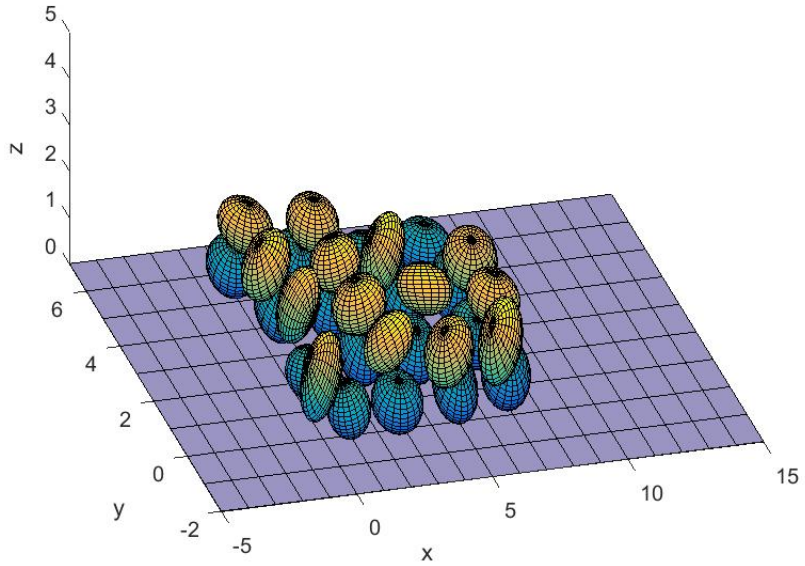
This is the example where ellipsoids arranged in a cube configuration fall onto a plane. The quaternion appears to be very consistent in error. The translational velocity's error is fairly low, with errors occurring at contacts. The rotational velocity and center positions have trouble converging, however.

7.3 Scaling

We ran a test where we arranged ellipsoids on a plane boundary (floor) along a line and measured the average amount of time spent per iteration. Friction was turned on. The amount of time was measured via the computer's internal clock, a 2015 Macbook Air on the lowest specs. The graph we found is as follows:

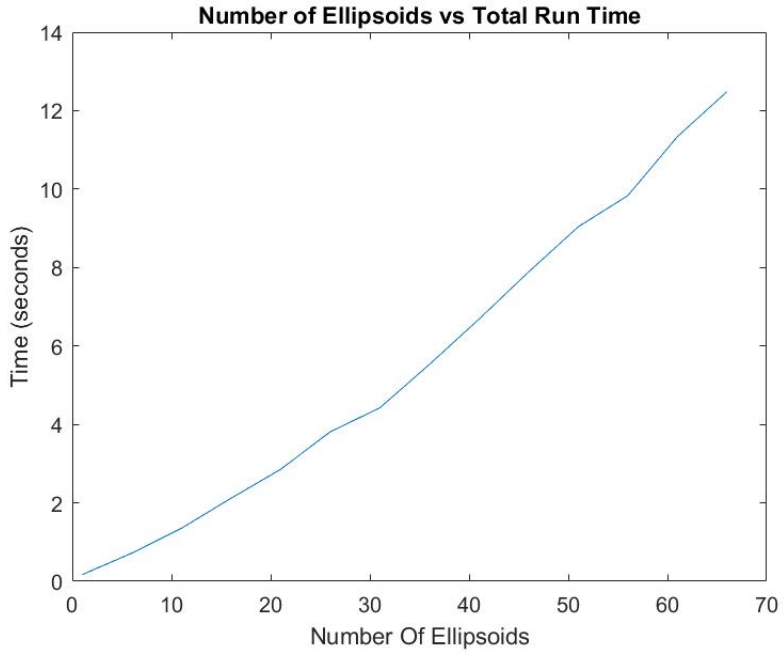


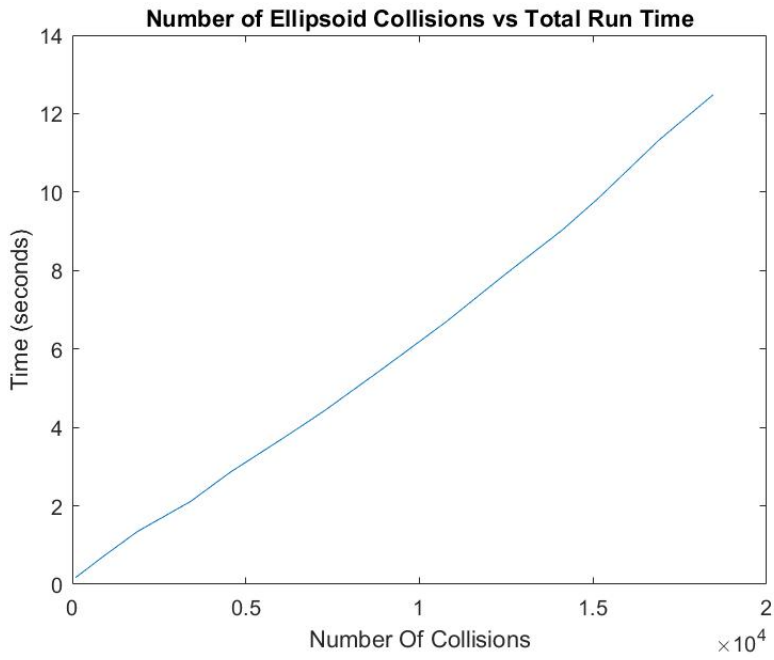
Here is a similar test showing more information with iterations of the minimal map newton solver for the LCP and GMRES function within. Here is a snapshot of 30 ellipsoids falling onto a plane boundary in a cubic lattice with random initial velocities.



And some results:

Number_Of_Ellipsoids	Total_Collisions	Total_Run_Time	Total_Iterations_MinMapNewton	Total_Iterations_GMRES_Inner	Total_Iterations_GMRES_Outer
1	92	0.1696	176	84	84
6	905	0.7182	215	115	590
11	1888	1.3516	246	146	923
16	3414	2.1160	282	182	1779
21	4545	2.8538	294	194	2292
26	6222	3.8113	279	183	2150
31	7256	4.4212	318	223	2741
36	8989	5.5292	322	232	3074
41	10754	6.6831	334	276	2919
46	12448	7.8874	350	341	2882
51	14119	9.0373	357	366	2625
56	15139	9.8330	359	346	2817
61	16913	11.3371	355	362	2663
66	18463	12.4849	371	396	2625





We observe a roughly linear relationship between the log (base 10) of the average time per iteration and the number of ellipsoids.

8 Conclusion

We have successfully built an algorithm to handle rigid body collisions of ellipsoids with sliding friction. Our algorithm is fast as it is posed as a linear complementarity problem as opposed to a nonlinear scheme. However the trade off is inaccuracies and danger of intersecting bodies if the step size is too large. Intersections cannot yet be predicted, but they can be avoided by running the simulation again with either a smaller time step and/or a larger body tolerance. Further work into the analysis of the jacobian approximation could be used to determine the precise step size to ensure non-penetration. The convergence results suggest linear convergence of important body information with respect to step size. More tests and tweaks need to be run to ensure linear convergence. Lastly, the issue with the LCP solver failing with sliding friction has yet to be resolved.

9 Appendix

9.1 Non Unique Point to Sphere

When determining the closest distance from the boundary of an ellipsoid to a sphere that encloses it, there are issues with non-uniqueness. In these cases all that must be returned is the closest distance from the ellipsoid to the sphere. Using the same notation as before, given a lagrange multiplier λ, we have

$$x_i = \frac{c_i a_i^2}{a_i^2 + \lambda} \tag{61}$$

This will have problems if $\lambda = -a_i^2$. Let $R \in \mathbb{N}^3$ be the indicies of all axes where this problem arises, ie $i \in R$ if and only if $\lambda = -a_i^2$.
If $R = (1, 1, 1)$, then the ellipsoid is a sphere centered at the origin so the distance is trivial (and probably not in danger of contact).
If R has one zero, without loss of generality (by reordering), we can assume $R = (1, 1, 0)$. Then we have x_3 by equation 61. We have a circle of infinite choices for x_1, x_2, so we can just let

$$x_1 = 0,\ x_2 = a_1\sqrt{1 - x_3^2/a_3^2} \tag{62}$$

And from this calculate the distance.
Finally for $R = (1, 0, 0)$ we have x_2, x_3, so we can just let

$$x_1 = a_1\sqrt{1 - x_2^2/a_2^2 - x_3^2/a_3^2} \tag{63}$$

And then calculate the distance.

References

[1] Lin, Anhua, and Shih-Ping Han. *On the Distance between Two Ellipsoids.* SIAM Journal on Optimization 13.1 (2002): 298-308. Web.

[2] Mihai Anitescu and F.A. Potra. *Formulating dynamic multi-rigid-body contact problems with friction as solvable Linear Complementarity Problems* Nonlinear Dynamics , 14 , 231–247, 1997. DOI: 10.1023/A:1008292328909

[3] Steven Delong *Brownian dynamics of confined rigid bodies* The Journal of Chemical Physics 143, 144107 (2015)