Theme 3 Milestone 2: Rigid Body Simulation

COMS W4167 Physically Based Computer Animation
Due: Wednesday, Now 2nd, 2016, 9:59:59pm

Academic Honesty Policy

You are permitted and encouraged to discuss your work with other students. You may work out equations in writing on paper or a whiteboard. You are encouraged to use Piazza to converse with other students, the TA, and the instructor.

HOWEVER, you may NOT share source code or hardcopies of source code. Refrain from activities or the sharing of materials that could cause your source code to APPEAR TO BE similar to another student's enrolled in this or previous years. We will be monitoring source code for individuality. Cheating will be dealt with severely. Cheaters will be punished. Source code should be yours and yours only. Do not cheat.

1 Introduction

In Theme III Milestone I we derived the equations of motion for a rigid body and explored the simulation of rigid body dynamics in the absence of collisions. In Theme II we explored various methods for resolving collisions, and encountered many shortcomings with these methods (recall the problems associated with the rigidification failsafe). In this milestone we will bring together ideas from these previous milestones and derive a method for rigid body contact. In particular, the response methods in this milestone can gracefully handle breaking contact in the face of multiple co-dependent collisions.

The methods in this milestone depend on more advanced numeric techniques than previous assignments; we will make use of both a linear complimentariry problem (LCP) solver, as well as a convex quadratic-program solver.

2 Program Submission

Please submit your solution using our submission script on the CLIC machines at /home/cs4167/scripts/grade.py as before. Detailed instructions are available on Piazza.

3 New XML Features

This milestone introduces the following new simulation features:

1. The *rigidbodycollisionhandling* feature specifies the detection and response methods to use:

   ```
   <rigidbodycollisionhandling detection="all-pairs" response="lcp"/>
   ```

   The *detection* attribute specifies which collision detection method to use; the only valid value for this assignment is *all-pairs*. The *response* method specifies which response method to use; the only valid values for this assignment are *lcp* and *velocity-projection*.

2. We have added a *fixed* attribute to the *rigidbody* feature:

   ```
   <rigidbody p="40" p="41" p="42" p="43" vx="0" vy="0" omega="0" r="0.5" fixed="1"/>
   ```

   If *fixed* is set to 1, forces and contact impulses will not effect the body.

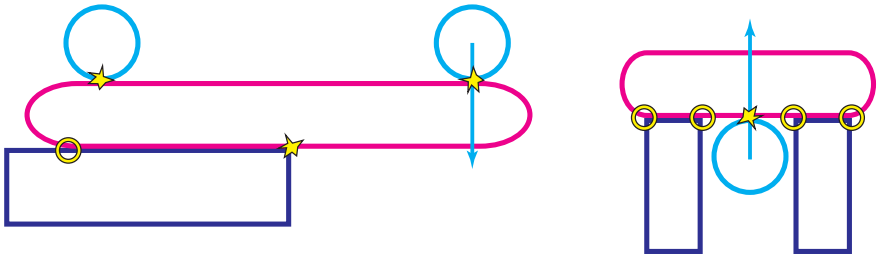

Figure 1: Breaking contacts in a rigid body simulation. Active contacts are denoted by small stars, breaking contacts by small circles. Naively resolving collisions in an iterative fashion can produce an incorrect solution.

4 The Linear Complimentarity Problem

4.1 Problem Formulation

Consider a system of rigid bodies in contact at k points. In order to prevent interpenetration, we would like to compute a set of k impulses applied at these contact points that yield separating relative velocities. That is, for each contact i, we seek impulses that yield $\dot{g}_i = \hat{\mathbf{n}}_i \cdot (\dot{\mathbf{r}}_{ip}^+ - \dot{\mathbf{r}}_{iq}^+) \geq 0$. We denote the relative velocity at the i^{th} contact point by $\dot{g}_i$, the contact normal at the i^{th} contact by $\hat{\mathbf{n}}_i$, and the post-impulse world-space velocities of rigid bodies p and q at the i^{th} contact by $\dot{\mathbf{r}}_{ip}^+$ and $\dot{\mathbf{r}}_{iq}^+$ (a $-$ superscript denotes pre-impulse). Denote the change in velocity due to the action of *all* impulses at the i^{th} contact point on each body by $\delta\dot{\mathbf{r}}_{ip}$ and $\delta\dot{\mathbf{r}}_{iq}$. With this notation, the i^{th} constraint becomes $\dot{g}_i = \hat{\mathbf{n}}_i \cdot (\dot{\mathbf{r}}_{ip}^- - \dot{\mathbf{r}}_{iq}^-) + \hat{\mathbf{n}}_i \cdot (\delta\dot{\mathbf{r}}_{ip} - \delta\dot{\mathbf{r}}_{iq}) \geq 0$ where $\dot{\mathbf{r}}_{ip}^-$ and $\dot{\mathbf{r}}_{iq}^-$ are the known pre-impulse velocities. Let us examine the change in velocities in more detail.

Denote the i^{th} impulse by $\lambda_i \hat{\mathbf{n}}_i$. Expanding $\delta\dot{\mathbf{r}}_{ip}$ in terms of the effect of all impulses on the center of mass velocity and the effect of all impulse-induced torques on the angular velocity, we find that $\delta\dot{\mathbf{r}}_{ip} = \delta\mathbf{v}_p + \delta\boldsymbol{\omega}_p \times \mathbf{r}'_{ip}$ (recall that prime denotes world-space positions relative to the center of mass). To compute the change in velocity, we simply sum the effect of all impulses on body p (let s_{jp} indicate whether impulse j acts on body p, and if so capture the correct sign): $\delta\mathbf{v}_p = M_p^{-1} \sum_{j=0}^{k-1} s_{jp}\lambda_j \hat{\mathbf{n}}_j$. To compute the impulse-induced torque, we sum impulses crossed with their point of action relative to the center of mass of body p: $\delta\boldsymbol{\omega}_p = I_p^{-1} \sum_{j=0}^{k-1} \mathbf{r}'_{jp} \times s_{jp}\lambda_j \hat{\mathbf{n}}_j$. Making these substitutions, we have: $\delta\dot{\mathbf{r}}_{ip} = M_p^{-1} \sum_{j=0}^{k-1} s_{jp}\lambda_j \hat{\mathbf{n}}_j + I_p^{-1} \sum_{j=0}^{k-1} \mathbf{r}'_{jp} \times s_{jp}\lambda_j \hat{\mathbf{n}}_j \times \mathbf{r}'_{ip}$.

Armed with the knowledge of how all impulses change the velocity at the points of contact, we can expand the constraint as:

$$\begin{aligned}
\dot{g}_i &= \hat{\mathbf{n}}_i \cdot (\dot{\mathbf{r}}_{ip}^- - \dot{\mathbf{r}}_{iq}^-) + \hat{\mathbf{n}}_i \cdot (\delta\dot{\mathbf{r}}_{ip} - \delta\dot{\mathbf{r}}_{iq}) \\
&= \sum_{j=0}^{k-1} \lambda_j \left(s_{jp} \hat{\mathbf{n}}_i^T M_p^{-1} \hat{\mathbf{n}}_j \right) + \sum_{j=0}^{k-1} \lambda_j \left(s_{jp} \hat{\mathbf{n}}_i^T I_p^{-1} \mathbf{r}'_{jp} \times \hat{\mathbf{n}}_j \times \mathbf{r}'_{ip} \right) \\
&- \sum_{j=0}^{k-1} \lambda_j \left(s_{jq} \hat{\mathbf{n}}_i^T M_q^{-1} \hat{\mathbf{n}}_j \right) - \sum_{j=0}^{k-1} \lambda_j \left(s_{jq} \hat{\mathbf{n}}_i^T I_q^{-1} \mathbf{r}'_{jq} \times \hat{\mathbf{n}}_j \times \mathbf{r}'_{iq} \right) \\
&+ \hat{\mathbf{n}}_i \cdot (\dot{\mathbf{r}}_{ip}^- - \dot{\mathbf{r}}_{iq}^-)
\end{aligned}$$

Observe that this constraint is linear in all impulse magnitudes λ_j. We can therefore express the contact constraints as a linear function of the impulse magnitudes: $\mathbf{A}\boldsymbol{\lambda} + \mathbf{b}$. The i^{th} entry of this vector gives the post-impulse relative velocity at the i^{th} contact. The non-penetration constraints are thus:

$$\boxed{\mathbf{A}\boldsymbol{\lambda} + \mathbf{b} \geq 0} \tag{1}$$

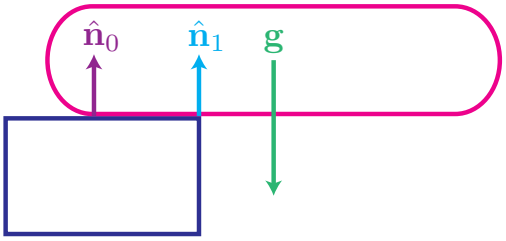

Figure 2: Block resting on table with center of mass over the edge.

Our work is not yet complete, however. As currently formulated, nothing prevents constraint impulses from pulling. Unless we are modeling an adhesive surface, this behavior is undesirable, so we impose the additional constraint that contact forces can only push:

$$\boxed{\boldsymbol{\lambda} \geq 0} \tag{2}$$

If we were to stop here, we would have a *linear programming* problem that we could solve using any number of algorithms. We need to introduce one final constraint before our work is complete, however. We want to enforce the fact that conditions that aren't currently violated should cause no impulse (i.e. $\lambda = 0$). If the motion of two objects is breaking away, or they are not in contact, our formulation should not apply an impulse. This often happens when resolving one constraint eliminates the need for another constraint to act (ref. Figure 1).

Otherwise, if we ignore the fact that no impulses should be applied along breaking contacts, we could potentially apply a response that is too strong and add energy to the system. Thus, we introduce the condition that:

$$\boxed{(\mathbf{A}\boldsymbol{\lambda} + \mathbf{b})_i \lambda_i = 0 \qquad \forall i} \tag{3}$$

What does this condition say? If $\lambda_i > 0$, then $(\mathbf{A}\boldsymbol{\lambda} + \mathbf{b})_i = 0$. That is, if an impulse is acting at some point, the bodies must remain in contact at that point. If $(\mathbf{A}\boldsymbol{\lambda} + \mathbf{b})_i > 0$, then $\lambda_i = 0$. That is, if bodies are separating, no impulse can act. This celebrated result is known as the *Signorini-Fichera Condition*. Together, these three conditions on contact constraints can be solved as a linear complimentarity problem.

4.2 Example System

Before proceeding, we will first examine an example problem and its solution.

4.3 Block On Table Edge With Two Contacts

Consider a block sitting on a fixed table with 2 contact points, and with the block's center of mass positioned beyond the edge of the table. See Figure 2. Labeling the rigid body A, we can write the constraints on the relative velocity as:

$$\dot{g}_0 = \hat{\mathbf{n}}_0 \cdot \dot{\mathbf{r}}^+_{0A} = \hat{\mathbf{n}}_0 \cdot \dot{\mathbf{r}}^-_{0A} + \hat{\mathbf{n}}_0 \cdot \delta\dot{\mathbf{r}}_{0A} \geq 0$$
$$\dot{g}_1 = \hat{\mathbf{n}}_1 \cdot \dot{\mathbf{r}}^+_{1A} = \hat{\mathbf{n}}_1 \cdot \dot{\mathbf{r}}^-_{1A} + \hat{\mathbf{n}}_1 \cdot \delta\dot{\mathbf{r}}_{1A} \geq 0$$

Expanding $\delta\dot{\mathbf{r}}_{0A}$ and $\delta\dot{\mathbf{r}}_{1A}$ we find

$$\delta\dot{\mathbf{r}}_{0A} = \delta\mathbf{v}_A + \delta\boldsymbol{\omega}_A \times \mathbf{r}'_{0A}$$
$$\delta\dot{\mathbf{r}}_{1A} = \delta\mathbf{v}_A + \delta\boldsymbol{\omega}_A \times \mathbf{r}'_{1A}$$

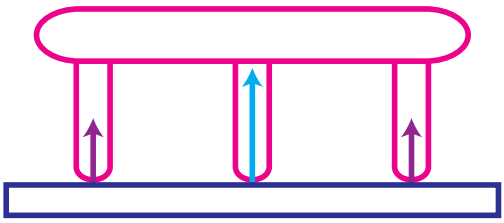

Figure 3: A table under gravity in two dimensions with three contacts. Observe that multiple sets of impulses give the same solution - we could apply only the large impulse to the center leg, or the two smaller impulses to the outer legs.

where primes denote the positions of the contacts relative to the body's center of mass. $\delta \mathbf{v}_A$ is the change in the center of mass' velocity due to both impulses, and $\delta \boldsymbol{\omega}_A$ is the change in angular velocity about the center of mass due to both impulses. Expanding $\delta \mathbf{v}_A$ and $\delta \boldsymbol{\omega}_A$ we find:

$$\begin{aligned}\delta \mathbf{v}_A &= \lambda_0 \hat{\mathbf{n}}_0 / M_A + \lambda_1 \hat{\mathbf{n}}_1 / M_A \\ \delta \boldsymbol{\omega}_A &= \mathbf{r}'_{0A} \times \lambda_0 \hat{\mathbf{n}}_0 / I_A + \mathbf{r}'_{1A} \times \lambda_1 \hat{\mathbf{n}}_1 / I_A\end{aligned}$$

Substituting these expressions into the constraints, we find

$$\begin{aligned}\dot{g}_0 &= \hat{\mathbf{n}}_0 \cdot \dot{\mathbf{r}}^-_{0A} + \lambda_0 \left(\hat{\mathbf{n}}_0 \cdot \hat{\mathbf{n}}_0 / M_A + \hat{\mathbf{n}}_0 \cdot \mathbf{r}'_{0A} \times \hat{\mathbf{n}}_0 \times \mathbf{r}'_{0A} / I_A \right) + \lambda_1 \left(\hat{\mathbf{n}}_0 \cdot \hat{\mathbf{n}}_1 / M_A + \hat{\mathbf{n}}_0 \cdot \mathbf{r}'_{1A} \times \hat{\mathbf{n}}_1 \times \mathbf{r}'_{0A} / I_A \right) \geq 0 \\ \dot{g}_1 &= \hat{\mathbf{n}}_1 \cdot \dot{\mathbf{r}}^-_{1A} + \lambda_0 \left(\hat{\mathbf{n}}_1 \cdot \hat{\mathbf{n}}_0 / M_A + \hat{\mathbf{n}}_1 \cdot \mathbf{r}'_{0A} \times \hat{\mathbf{n}}_0 \times \mathbf{r}'_{1A} / I_A \right) + \lambda_1 \left(\hat{\mathbf{n}}_1 \cdot \hat{\mathbf{n}}_1 / M_A + \hat{\mathbf{n}}_1 \cdot \mathbf{r}'_{1A} \times \hat{\mathbf{n}}_1 \times \mathbf{r}'_{1A} / I_A \right) \geq 0\end{aligned}$$

or in matrix form

$$\begin{pmatrix} \hat{\mathbf{n}}_0 \cdot \hat{\mathbf{n}}_0 / M_A + \hat{\mathbf{n}}_0 \cdot \mathbf{r}'_{0A} \times \hat{\mathbf{n}}_0 \times \mathbf{r}'_{0A} / I_A & \hat{\mathbf{n}}_0 \cdot \hat{\mathbf{n}}_1 / M_A + \hat{\mathbf{n}}_0 \cdot \mathbf{r}'_{1A} \times \hat{\mathbf{n}}_1 \times \mathbf{r}'_{0A} / I_A \\ \hat{\mathbf{n}}_1 \cdot \hat{\mathbf{n}}_0 / M_A + \hat{\mathbf{n}}_1 \cdot \mathbf{r}'_{0A} \times \hat{\mathbf{n}}_0 \times \mathbf{r}'_{1A} / I_A & \hat{\mathbf{n}}_1 \cdot \hat{\mathbf{n}}_1 / M_A + \hat{\mathbf{n}}_1 \cdot \mathbf{r}'_{1A} \times \hat{\mathbf{n}}_1 \times \mathbf{r}'_{1A} / I_A \end{pmatrix} \begin{pmatrix} \lambda_0 \\ \lambda_1 \end{pmatrix} + \begin{pmatrix} \hat{\mathbf{n}}_0 \cdot \dot{\mathbf{r}}^-_{0A} \\ \hat{\mathbf{n}}_1 \cdot \dot{\mathbf{r}}^-_{1A} \end{pmatrix} \geq \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Let us substitute in specific values and examine the solution. Let $\hat{\mathbf{n}}_0 = (0,1)^T$, $\hat{\mathbf{n}}_1 = (0,1)^T$, $M_A = 4$, $I_A = 20$, $r'_{0A} = (-2,-1,0)$, $r'_{1A} = (-1,-1,0)$, $\dot{\mathbf{r}}^-_{0A} = (0,-2,0)$, and $\dot{\mathbf{r}}^-_{1A} = (0,-2,0)$. This gives the system:

$$\begin{pmatrix} 13/20 & 9/20 \\ 9/20 & 7/20 \end{pmatrix} \begin{pmatrix} \lambda_0 \\ \lambda_1 \end{pmatrix} + \begin{pmatrix} -2 \\ -2 \end{pmatrix} \geq \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

which is solved by $\lambda_0 = 0$ and $\lambda_1 \approx 5.7143$ (as a sanity check, verify that these values satisfy the three constraints in the LCP). What does this mean? $\lambda_0 = 0$ implies that the leftmost impulse is not active and the contact is breaking. $\lambda_1 \approx 5.7143$ implies that the second point remains in contact with the table, and that an impulse is acting to prevent interpenetration. The net effect is that the block will pivot about the end of the table, as expected.

5 Redundant Constraints

While our formulation is enough to obtain a correct solution, it does not pin down a unique solution. For example, Figure 3 illustrates a table with three legs resting on the ground. There are a (uncountably infinite) number of solutions to this contact problem as we have phrased it. We could apply a single impulse of magnitude I to the center leg and no impulses to the side legs, we could apply two impulses of magnitude $I/2$ to the side legs, or we could apply any combination of (pushing) impulses with total magnitude I that induce no torque.

What does this redundancy mean in linear algebra terminology? For the three legged table, the matrix $\mathbf{A}$ is *singular*. That is, $\mathbf{A}\boldsymbol{\lambda}+\mathbf{b} = 0$ has multiple solutions. Many of the standard LCP algorithms, unfortunately,

will fail when presented with a singular matrix. While one can construct a number of interesting problems that do not have this singular matrix, if we desire a truly robust method we will have to work a little harder. It is important to stress that this is a numerical issue. Conceptually (numerics aside), even though $\boldsymbol{\lambda}$ is not uniquely defined, the final velocities are uniquely defined. Luckily, this problem can be solved by a number of numerical methods, some of which are more robust to this redundancy than the LCP. We will repose this problem as a minimization, but first let us formalize some of the concepts that we explored above, using a notation that applies equally to the LCP and minimization viewpoints.

6 More General Notation

The notation employed thus far has been traditionally used in the graphics literature. While the derivations so far have been straightforward in that they follow immediately from writing down the physics of rigid bodies and 'turning the crank', they can be a tad unwieldy.

Before rephrasing the contact problem to address the singular matrix $\mathbf{A}$, let us take a moment to construct a more general notation than the one employed above. Rewriting the notation will have a few advantages. First, the more general notation will make our results easier to carry over to other settings (e.g. simulation of deformable bodies). Second, the more general notation will make higher-level algebraic manipulations easier, and makes it easier to see higher-level properties of the system, such as the symmetry of $\mathbf{A}$.

6.1 Notation

Take each rigid body's *reduced* or *generalized* coordinate representation (center of mass position, orientation), and concatenate them into a single column vector $\mathbf{q}$. Similarly, concatenate the rate of change of the reduced coordinates into a single column vector $\dot{\mathbf{q}}$. As an example, for our $2D$ representation or rigid bodies, $\mathbf{q} \in \mathbb{R}^{3N}$ and $\dot{\mathbf{q}} \in \mathbb{R}^{3N}$, where N is the number of rigid bodies. For 3 rigid bodies these vectors look like:

$$\mathbf{q} = \begin{pmatrix} X_x^0 & X_y^0 & \theta^0 & X_x^1 & X_y^1 & \theta^1 & X_x^2 & X_y^2 & \theta^2 \end{pmatrix}$$
$$\dot{\mathbf{q}} = \begin{pmatrix} V_x^0 & V_y^0 & \omega^0 & V_x^1 & V_y^1 & \omega^1 & V_x^2 & V_y^2 & \omega^2 \end{pmatrix}$$

When applying forces and impulses to rigid bodies, we need some way to compute world-space positions and velocities from reduced positions and velocities. Let $\mathbf{r}_i(\mathbf{q})$ be a function that computes the world space position of the i^{th} point from the reduced representation. For example, if the seventh particle is attached to the first rigid body, in our representation this function takes the form (tilde denotes body space):

$$\mathbf{r}_7(\mathbf{q}) = R(\theta^1)\tilde{\mathbf{r}}_7 + \mathbf{X}^1$$

To compute the world-space velocity of the i^{th} particle, we simply employ the chain rule. That is, $\dot{\mathbf{r}}_i(\mathbf{q}) = \nabla \mathbf{r}_i \dot{\mathbf{q}}$. Continuing with the above example, $\nabla \mathbf{r}_7 \in \mathbb{R}^{2\times 9}$ is given by:

$$\nabla \mathbf{r}_7 = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & -\sin\theta^1\tilde{r}_{7x} - \cos\theta^1\tilde{r}_{7y} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \cos\theta^1\tilde{r}_{7x} - \sin\theta^1\tilde{r}_{7y} & 0 & 0 & 0 \end{pmatrix}$$

Convince yourself that the multiplication $\nabla \mathbf{r}_i \dot{\mathbf{q}}$ gives the same velocity we derived in Theme III Milestone I.

Consider now a set of contacts C. For some contact $k \in C$, the contact occurs between two points $\mathbf{r}_i \in \mathbb{R}^2$ and $\mathbf{r}_j \in \mathbb{R}^2$. Let the normal for this contact, $\hat{\mathbf{n}}_k \in \mathbb{R}^2$, point from i to j. We can thus express the relative velocity at this contact as $\hat{\mathbf{n}}_k^T(\dot{\mathbf{r}}_i - \dot{\mathbf{r}}_j) = \hat{\mathbf{n}}_k^T(\nabla \mathbf{r}_i - \nabla \mathbf{r}_j)\dot{\mathbf{q}} \equiv \hat{\mathbf{n}}_k^T \Gamma_k \dot{\mathbf{q}}$. For our simulations, $\Gamma_k \in \mathbb{R}^{2\times 3N}$, and contains 2 non-zero $\mathbb{R}^{2\times 3}$ blocks. Continuing our example of three rigid bodies, if the 3^{rd} contact is between the 7^{th} point (on body 1) and the 9^{th} point (on body 2), Γ_3 is given by:

$$\Gamma_3 = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & -\sin\theta^1\tilde{r}_{7x} - \cos\theta^1\tilde{r}_{7y} & -1 & 0 & \sin\theta^2\tilde{r}_{9x} + \cos\theta^2\tilde{r}_{9y} \\ 0 & 0 & 0 & 0 & 1 & \cos\theta^1\tilde{r}_{7x} - \sin\theta^1\tilde{r}_{7y} & 0 & -1 & -\cos\theta^2\tilde{r}_{9x} + \sin\theta^2\tilde{r}_{9y} \end{pmatrix}$$

If we want to apply an equal and opposite impulse $\mathbf{y}$ to points i and j at contact k, the resulting impulse on the reduced coordinates is given by $\Gamma_k^T \mathbf{y}$. Similarly, we can map the contact normals to their reduced coordinate counterparts by $\boldsymbol{\eta}_k = \Gamma_k^T \hat{\mathbf{n}}_k \in \mathbb{R}^{3N}$. Finally, concatenating these normals into one matrix, we have $\mathbf{N} = \begin{pmatrix} \boldsymbol{\eta}_0 & \boldsymbol{\eta}_1 & \dots & \boldsymbol{\eta}_{|C|-1} \end{pmatrix} \in \mathbb{R}^{3N\times|C|}$.

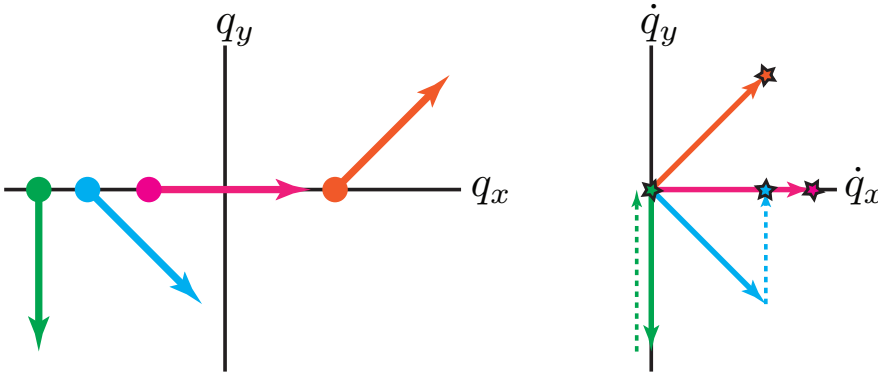

Figure 4: On the left, unit-mass particles moving in configuration space constrained to lie above the horizontal. On the right, the same particles in velocity space and the constraint impulses. Post-impulse velocities are denoted by stars. Notice that the post-impulse velocities are given by the closest point in the admissible region.

6.2 A Revisited

Let us revisit the construction of the system $\mathbf{A}\boldsymbol{\lambda} + \mathbf{b}$ for the LCP. The post-impulse relative velocity at all contact points in world-space can be expressed in terms of the reduced coordinates as $\mathbf{N}^T\dot{\mathbf{q}}^+$. Mapping the impulse magnitudes in world-space to the reduced coordinate space gives $\mathbf{N}\boldsymbol{\lambda}$, and the corresponding changes in velocities are expressed as $\mathbf{M}^{-1}\mathbf{N}\boldsymbol{\lambda}$. Thus, we find that $\mathbf{N}^T\dot{\mathbf{q}}^+ = \mathbf{N}^T(\dot{\mathbf{q}}^- + \mathbf{M}^{-1}\mathbf{N}\boldsymbol{\lambda}) = \mathbf{N}^T\mathbf{M}^{-1}\mathbf{N}\boldsymbol{\lambda} + \mathbf{N}^T\dot{\mathbf{q}}^- \geq 0$. That is, $\mathbf{A} = \mathbf{N}^T\mathbf{M}^{-1}\mathbf{N}$ and $\mathbf{b} = \mathbf{N}^T\dot{\mathbf{q}}^-$. The symmetry of $\mathbf{A}$ now follows immediately from the symmetry of $\mathbf{M}$ (take the transpose of $\mathbf{N}^T\mathbf{M}^{-1}\mathbf{N}$ and see what you get).

7 Velocity-Projection for Inelastic Contact

When solving the LCP, we encountered problems with a singular matrix that arose due to redundant contact directions. However there is a way to rephrase our problem as a minimization problem. More specifically, a convex quadratic-program, an optimization problem for which algorithms exist that are robust in the face of redundant constraints (see [1]).

How do we pose our problem as a minimization? We want our contact to be resolved in a way that changes velocity the least. This will preserve the physical properties of the system as much as possible while ensuring no penetration occurs. We do this by minimizing the kinetic energy of the velocity difference. Think about it like this: we can't conserve energy, but we can try to change it as little as necessary.

If our initial velocity is $\mathbf{v}^-$ and our resulting velocity is $\mathbf{v}^+$, let us define the difference as $\delta\mathbf{v}$ (ref. Figure 5). Minimizing the kinetic energy of the difference then is: $\underset{\delta\mathbf{v}}{argmin}\ \frac{1}{2}\delta\mathbf{v}^T M\delta\mathbf{v}$.

The constraint that we have is non-penetration. This means $0 \leq \mathbf{v}^{+T}n = \mathbf{v}^{-T}n + \delta\mathbf{v}^T n = c(\delta\mathbf{v})$.

The Lagrangian function for this problem is $L(\delta\mathbf{v}, \lambda) = \frac{1}{2}\delta\mathbf{v}^T M\delta\mathbf{v} - \lambda(\mathbf{v}^{-T}n + \delta\mathbf{v}^T n)$.

Optimization theory then tells us that the necessary conditions for a minimum are the so called KKT (Karush-Kuhn-Tucker) conditions (ref. [2, p. 321]). They are:

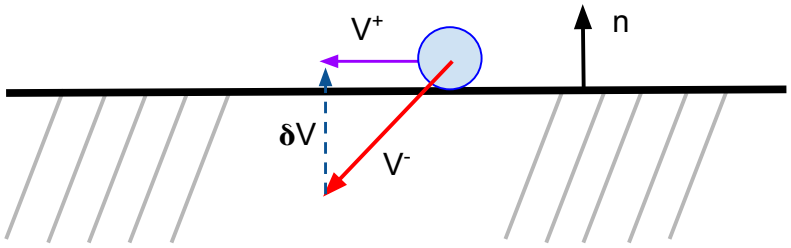

Figure 5: A sketch of our minimization problem.

$$\begin{aligned} \nabla_{\delta \mathbf{v}} L &= 0 \\ c(\delta \mathbf{v}) &\geq 0 \\ \lambda &\geq 0 \\ \lambda c(\delta \mathbf{v}) &= 0 \end{aligned} \tag{4}$$

Substituting our definitions in, this means $M\delta\mathbf{v} - \lambda n = 0$, so:

$$\delta \mathbf{v} = M^{-1} \lambda n \tag{5}$$

$$\mathbf{v}^{-T} n + \delta \mathbf{v}^T n \geq 0 \tag{6}$$

$$\lambda \geq 0 \tag{7}$$

$$\lambda(\mathbf{v}^{-T} n + \delta \mathbf{v}^T n) = 0 \tag{8}$$

7.1 Connection to LCP

In fact we see that this gives the same result as the LCP formulation. (5) is equivalent to the LCP definition of the impulses, (6) is equivalent to (1) (the non-penetration condition), (7) is equivalent to (2) (non-adhesion) and (8) is equivalent to (3) (the *Signorini-Fichera Condition*).

As we would expect, the KKT conditions guarantee that impulses are only active for velocities that are bringing particles into the inadmissible region. Further, notice that impulses only act in the positive constraint gradient direction, that is they are only pushing. Finally, observe that all post-impulse relative velocities are separating or zero. Thus, all of the content of the LCP formulation is reflected in the velocity-projection solution.

7.2 General formulation of the problem

As a general formulation, the post-collision velocity is given by the minimizer of $\delta T = \frac{1}{2}(\mathbf{v} - \dot{\mathbf{q}}^-)^T \mathbf{M}(\mathbf{v} - \dot{\mathbf{q}}^-)$ subject to $\mathbf{N}^T\mathbf{v} \geq 0$. Expanding this multiplication, we find that $\delta T = \frac{1}{2}\mathbf{v}^T\mathbf{M}\mathbf{v} - \frac{1}{2}\mathbf{v}^T\mathbf{M}\dot{\mathbf{q}}^- - \frac{1}{2}(\dot{\mathbf{q}}^-)^T\mathbf{M}\mathbf{v} - \frac{1}{2}(\dot{\mathbf{q}}^-)^T\mathbf{M}\dot{\mathbf{q}}^- = \frac{1}{2}\mathbf{v}^T\mathbf{M}\mathbf{v} - \mathbf{v}^T\mathbf{M}\dot{\mathbf{q}}^- - \frac{1}{2}(\dot{\mathbf{q}}^-)^T\mathbf{M}\dot{\mathbf{q}}^-$ where we have used the symmetry of $\mathbf{M}$ to combine the

cross terms. We can drop the constant term $-\frac{1}{2}(\dot{\mathbf{q}}^-)^T\mathbf{M}\dot{\mathbf{q}}^-$, as constant offsets will not change the minimum of the function. Thus, the post-collision velocities are given by:

$$\boxed{\dot{\mathbf{q}}^+ = \underset{\mathbf{v}}{argmin}\left(\frac{1}{2}\mathbf{v}^T\mathbf{M}\mathbf{v} - \mathbf{v}^T\mathbf{M}\dot{\mathbf{q}}^- : \mathbf{N}^T\mathbf{v} \geq 0\right)}$$

8 Required Features

8.1 Rigid-Body Rigid-Body All-Pairs Detection

To detect collisions, please complete the *detectCollisions* method of *RigidBodyAllPairsCollisionDetector.cpp*. As input this method takes a vector of all rigid bodies in the simulation, and as output constructs a set of template type *RigidBodyCollision* containing all collisions in the system. For each pair of rigid bodies, detect any vertex-edge collisions. A vertex and an edge are considered colliding if the distance between the two is less than *(keep this inequality strict!)* the sum of the radii of each body. You should be able to adapt this code from Theme II. When inserting a *RigidBodyCollision* into the collision set, you will have to compute:

1. The index of the first rigid body of the collision (i in the constructor)
2. The index of the second rigid body of the collision (j in the constructor)
3. The vector from the center of mass of the first body to the point of contact (*r0* in the constructor)
4. The vector from the center of mass of the second body to the point of contact (*r1* in the constructor)
5. The unit-length contact normal (*nhat* in the constructor)

8.2 LCP

Please complete the *resolveCollisions* method of *RigidBodyLCPCollisionResolver.cpp*. As input this method takes both a vector of all rigid bodies and a set of all collisions between rigid bodies, and as output modifies the velocities and angular velocities of the rigid bodies so as to prevent penetration.

You will have to compute the matrix $\mathbf{A}$ and the vector $\mathbf{b}$ described in Sections 4 and 6.2. We have provided a function *lcputils::solveLCPwithODE* that takes as inputs $\mathbf{A}$ and $\mathbf{b}$ and returns a vector $\boldsymbol{\lambda}$ of impulse magnitudes (please see the code for an example). You will then have to apply the resulting impulses to the rigid bodies.

We have provided the expected values for $\mathbf{A}$ and $\mathbf{b}$ in the scene files *collisionslcp/test01.xml* and *collisionslcp/test03.xml*. This code uses the implementation of Dantzig's method from the open source rigid body engine ODE.

Note on fixed rigid bodies: Impulses have no effect on fixed rigid bodies. Therefore, to handle fixed rigid bodies in the response, when you are constructing $\mathbf{A}$ ignore any terms corresponding to an impulses' effect on a fixed body.

8.3 Velocity-Projection

Please complete the *resolveCollisions* method of *RigidBodyVelocityProjectionCollisionResolver*. As input this method takes both a vector of all rigid bodies and a set of all collisions between rigid bodies, and as output modifies the velocities and angular velocities of the rigid bodies so as to prevent penetration.

You will have to compute the mass matrix $\mathbf{M}$, the product $-\mathbf{M}\dot{\mathbf{q}}^-$, and the generalized constraint matrix $\mathbf{N}$ described in Sections 6 and 7. We have provided a function *solve_quadprog* that takes these quantities as inputs and returns the solution to the QP. Please see the code for an example.

We have provided the expected values for $\mathbf{M}$, $-\mathbf{M}\dot{\mathbf{q}}^-$, and $\mathbf{N}$ in the scene files *collisionsvelocityprojection/test01.xml* and *collisionsvelocityprojection/test05.xml*. This code uses the open-source QuadProg++ implementation of the Goldfarb-Idnani algorithm.

Note on fixed rigid bodies: The easiest way to handle fixed bodies in the solve is to simply not expose those bodies' degrees of freedom to the QP solver. That is, shrink $\mathbf{M}$, $\mathbf{M}\dot{\mathbf{q}}^-$, and $\mathbf{N}$ to not include fixed degrees of freedom, and do not add fixed bodies' contributions to $\mathbf{N}$.

9 Extra Credits: Elastic Response

The LCP method as we formulated above attempts to find a valid (non-pulling, normal) impulse such that the post-response velocity does not violate the collision constraint. The Velocity Projection method above attempts to project the velocity back to the admissible domain too, removing any constraint-violating component but nothing more. As a result, both responses are fully inelastic by construction, which is apparent from the $\mathbf{N}^T\dot{\mathbf{q}}^+ \geq 0$ constraint common to both methods. An elastic response can be obtained by changing this constraint from post-response velocity cannot be approaching to requiring that the post-response velocity is at least as much departing as the pre-response velocity was approaching, i.e. $\mathbf{N}^T\dot{\mathbf{q}}^+ \geq -\mathbf{N}^T\dot{\mathbf{q}}^-$. We refer the readers to recent work by [3] for more details.

However, as soon as we move into the elastic regime, the multiple simultaneous contact problem becomes much more complicated, as pointed out in [3]. It is straightforward to list a small set of fundamental properties that a desired collision response method should possess, such as symmetry preservation, energy conservation and not creating artificial sticking between objects, but none of the existing methods prior to [3] was able to achieve all of these desiderata. Fortunately, with a critical observation on the property of LCP, it is possible to design a method satisfying all the desiderata, through relatively minor changes to what we have built in the previous sections. Simply speaking, LCP/Velocity Projection should be applied prudently and repeated when necessary.

For this section, please carefully read [3], and

1. implement fully elastic response in both LCP and Velocity Projection methods. The formulation in [3] is general for any coefficient of restitution between 0 and 1, but for this milestone we are only interested in the fully elastic case.

2. implement the Generalized Reflection algorithm (algorithm 2 in [3]) on top of both LCP and Velocity Projection methods. The resulting collision response algorithm should have all the desired properties.

All of the implementation in this section should be done in source files *RigidBodyGRLCPCollisionResolver.cpp/.h* and *RigidBodyGRVelocityProjectionCollisionResolver.cpp/.h*, separate from the inelastic LCP and Velocity Projection algorithms in the previous sections. Two new collision response methods, gr-lcp and gr-velocity-projection, have been added to the response property of the *rigidbodycollisionhandling* XML tag.

You can find the tests in the t3m2 extra credit folder to test the various properties of your collision response implementation. You will be awarded 50% extra credit for a correct Generalized Reflection implementation that passes all these tests. Most of these tests are taken from [3], therefore, in addition to the oracles visual output, you also have the papers video as a qualitative reference to compare your animation to:

`http://www.cs.columbia.edu/cg/rosi/rosi.mov`

Note on line 7 of Algorithm 2 in [3]: Since our LCP solver and Quadratic Programming solver, which GR uses as building blocks, have limited numerical precision, a small epsilon should be used for this test. Specifically, please only proceed into the true branch if the pre-response normal velocity (left hand side of the inequality on line 7) is smaller than 10^{14}.

10 Creative Scene

As part of your final submission for this milestone, please include a scene of your design that best shows off your program. Based on the quality of your scene, you will have the opportunity to earn up to 15% extra credit. Your scene will be judged by a secret of committee of top scientists using the highly refined criteria of:

1. How well the scene shows off this milestones magic ingredients (a la Iron Chef).
2. Aesthetic considerations. The more beautiful, the better.
3. Originality.

Top examples will be posted on Piazza, and possibly demoed for the class. To submit this scene, place the XML file in the Creative directory of your submission. Please name your scene file youruni tXmX.xml where youruni is your uni. Note that if you do not follow this requirement your scene may not be picked up by the grading script and may not receive any credit. We also ask that you include a movie of your creative scene in the Creative directory so that it can be posted on Piazza, should it be chosen by the judge committee. You can find instructions on how to generate movies from simulations on Piazza.

11 FAQ

Theme 3 Milestone 2 FAQ:

Q: What is the Matrix M referred to in the writeup?
A: This is the diagonal 'Mass' matrix. In our case the matrix M is a diagonal matrix with repeating sets of three entries per body: M, M, I (so the third entry per body is the moment of inertia).

Q: What are the vectors r and $\dot{r}$?
A: $r = R(\theta)\tilde{r} + \text{CoM}$ (like milestone 1, so no third dimension)
$\dot{r}$ is the corresponding $(\nabla r)\dot{q}$ (which has a third dimension)

Q: Should we include fixed bodies in our mass matrix M?
A: No. The mass matrix should be square with length $3(N - K)$, with N being the number of bodies, and K being the number of fixed bodies.

Q: Whats the best way to debug penetration for Velocity Projection?
A: Velocity projection method is a constrained optimization. There are naturally two kinds of bugs: ones that violate the constraints, and ones that fails the optimization. The first kind of bugs can be spotted by observing that your solution doesn't satisfy the constraint inequalities. The second kind of bugs can be spotted by checking the direction of the gradient of the objective function (in our case the norm of velocity change).

If you are seeing penetration it is a violation of the constraint "normal velocity ¿= 0". You can try to find a minimal example that reproduces this violation and verify your implementation manually.

Notes:

- Your Mass matrix should not have negative entries in it.
- Make sure your impulses only get applied once. People have had issues in the past were they applied it multiple times on the same collision point.
- This assignment has proven very tricky in the past, however everything is in the writeup. Many issues from previous years have been resolved by reading the writeup more closely.
- Be very very careful about what calculations are done in body space coordinates and which are world space.
- A key to this milestone is being consistent with a choice of sign in collision detection normal direction. All will work out in the math if you stay consistent.

- A zero rotation is an identity matrix, not all zeros.
- The outgoing velocity of two colliding objects is dependent on their individual masses. Conserve Momentum.
- It is extremely useful for understanding and debugging purposes to work out the dimensions of your matrix algebra by hand and confirm the code matches your results.
- Be sure to note whether the variables passed into a function you are using are using body-coordinates or world-space as a reference frame. This is often a cause of error.
- Having a velocity of zero is not the same as being a fixed body.
- Lambda only tells you the strength of the impulse that should be applied at that contact point, so you should ensure that it's being applied in the correct direction for each body.

References

[1] D Goldfarb and A Idnani. A numerically stable dual method for solving strictly convex quadratic programs. *Mathematical Programming*, (27), 1983.

[2] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, 2006.

[3] Breannan Smith, Danny M Kaufman, Etienne Vouga, Rasmus Tamstorf, and Eitan Grinspun. Reflections on simultaneous impact. *ACM Transactions on Graphics (TOG)*, 31(4):106, 2012.