A Correction To Brian Mirtich's Thesis: "Impulse-based Dynamic Simulation of Rigid Body Systems"

Rahil Baber

15th February 2011

rahilbaber@hotmail.com

Contents

1 Introduction

Firstly let me say that if you're interested in rigid body simulation Brian Mirtich's thesis, entitled "Impulse-based Dynamic Simulation of Rigid Body Systems", is a great resource and well worth reading. A copy can be found at:

http://www.kuffner.org/james/software/dynamics/mirtich/index.html

I was particularly interested in how to simulate collisions with friction which is covered in Chapter 3. Unfortunately at the end of Section 3.4.2 there is a flaw in his argument which means his method of finding the impulse has to be changed slightly. This document is intended to highlight that and to provide a reworking of the algorithm. This document does not cover resolving contact forces between objects at rest (of which I'm still unsure of the best way to deal with), rather we deal only with objects colliding with some non-negligible velocity.

If you only care that the simulation looks right, then the algorithm given in the thesis is more than adequate (as nobody has reported any unrealistic behaviour since its publication in 1996). If however you are concerned about keeping to the laws of physics (even though there is no such thing as a totally rigid body) then later in this document I'll outline a corrected version of the algorithm, which as an added bonus should be easier to implement.

1.1 The Error

This section and Section 1.2 are perhaps in the wrong place. It would make more sense to place them at the end of this document, after I've gone over all the variable definitions and maths behind the algorithm. However, I want to provide a strong motivation early on for why this document is necessary and why you should continue reading it. The content is mainly aimed at those who are familiar with Chapter 3 of the thesis. Those who are not should still be able to follow the argument, though will have to take what I say on faith. Feel free to skip ahead to Section 2.

Let me begin by describing where I think the flaw occurs in Mirtich's argument. Basically his algorithm involves integrating various formulas with respect to p_z (a component of impulse). Unfortunately working out the limits of integration is problematic using p_z, so instead he does a change of variable to u_z (a component of the separation velocity). However, for this change of variable to be valid he needs du_z/dp_z to be positive. Annoyingly this isn't always true, there exist pathological examples where the larger the normal force you apply to separate the objects the more they accelerate towards each other. He argues that although du_z/dp_z may not initially be positive it will eventually become positive and stay positive for the rest of the calculation. So far I'm happy to believe everything he has said. He then goes on to say that if du_z/dp_z is initially negative then integrate using p_z until du_z/dp_z becomes positive then proceed as normal using u_z instead. This is where I think a problem lies, essentially he is assuming that *once* it is positive it stays positive. This is subtly different to the fact that *eventually* it is positive and stays positive. To illustrate my point see Figure 1.

Maybe he *meant* to say once du_z/dp_z is positive it stays positive, but unfortunately I can construct examples in which it starts positive and turns negative. This means we can't change to integrating with respect to u_z. Another consequence is that colliding objects may undertake a compression phase, then a decompression phase, followed by another compression phase, and then a final decompression phase

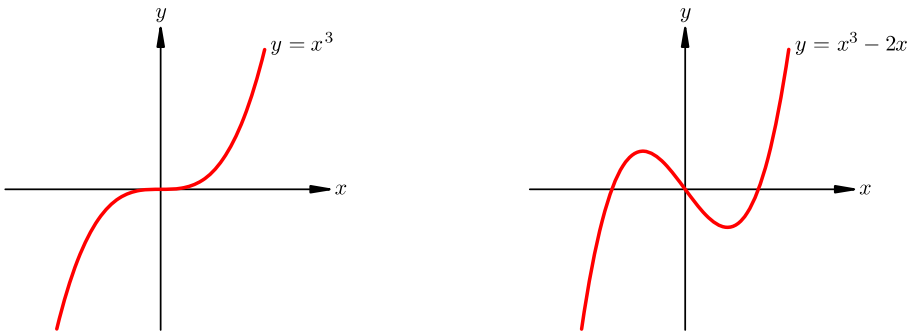

Figure 1: As x increases both x^3 and $x^3 - 2x$ eventually become positive and stay positive, but only x^3 has the property that once it is positive it stays positive.

before the objects separate. The algorithm outlined in the thesis only supposes one compression phase and decompression phase occurs, which is a problem that needs addressing.

I should point out that none of my claims have been verified by anyone else. I did contact James Kuffner who hosts the thesis, about a possible mistake, he forwarded my e-mail to Brian Mirtich. Brian Mirtich did reply saying that he would have a look at it, but he would have to re-read his thesis before he could answer any of my questions. So there maybe nothing wrong with the thesis and it's just my poor understanding of what was actually written (in which case I'd appreciate it if you'd let me know). In any case let me continue, arrogantly assuming I'm correct.

1.2 An Example

For those who think I'm wrong I'll construct an example which I believe verifies my claims. Although it is a pathological example finding such examples isn't hard. With a bit of effort a more reasonable example could probably be found. I will provide values to setup a collision, so that others (if so inclined) can simulate it and check that du_z/dp_z can turn from being positive to negative, and that two pairs of compression decompression phases can occur. If you don't know how to simulate the collision I suggest you skip ahead to the next section, where I start defining the variables and conventions I will use. From there you can either delve into the maths we will use to simulate the collision or just skip to the sections where I describe the algorithm (Sections 4 and 5).

I'll ignore the units of measurements as they are unimportant as long as we are consistent. Let the coefficient of friction μ be 0.5, and the coefficient of restitution e be 0.9. The two objects collide such that the normal force acts in the z direction, and the components of their separation velocity u_x, u_y, u_z are initially $630, -780, -0.22$ respectively. Because $u_z < 0$ we know they are colliding and an impulse is needed to cause them to move away from each other. Let one of the colliding objects have infinite mass so that the inverse of its inertia tensor is the zero matrix. Consequently it will make no contribution to the collision matrix K (a matrix used to help us compute impulses). If you are uneasy with infinite mass objects, you can instead give it a sufficiently large but finite mass such that

its contribution to K is negligible. The other object has a mass of 1, and an inverse inertia tensor of

$$\begin{pmatrix} 9 & 6 & -6 \\ 6 & 6 & -2 \\ -6 & -2 & 9 \end{pmatrix}.$$

Note that its eigenvalues are approximately 17.8, 5.6, and 0.5 implying it's a positive definite matrix and hence a valid inverse inertia tensor. The collision occurs at a position of $(1, 1, 1)$ relative to the centre of mass of the object of mass 1. (Where the collision occurs on the other object is unimportant). All this information allows to compute the collision matrix,

$$K = \begin{pmatrix} 20 & -23 & 4 \\ -23 & 31 & -7 \\ 4 & -7 & 4 \end{pmatrix}.$$

It has eigenvalues which are approximately 50.5, 3.5, and 1.0, proving that it is positive definite (a good sanity check). Using K we can plot u_z versus p_z see Figure 2. u_z starts at -0.22, and du_z/dp_z is positive, eventually u_z becomes positive and reaches a local maximum at $p_z = 22.5$. Then we see du_z/dp_z become negative as claimed. When u_z is negative the collision is in a compression phase and when it is positive it is in a decompression phase. The graph has roots at approximately 14.6, 29.8, and 56.0 hence there are two compression phases and two decompression phases, proving my other claim.

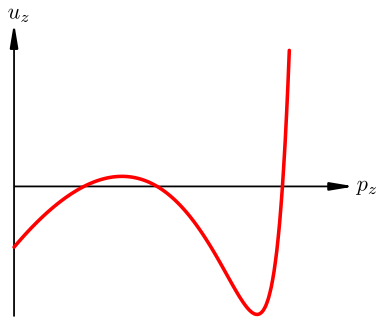

Figure 2: A graph of u_z versus p_z.

2 Variables And Conventions

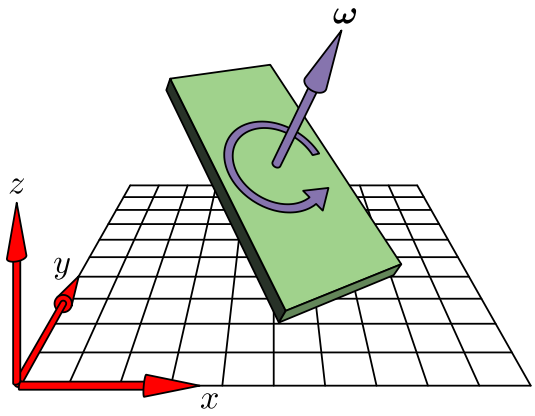

Figure 3: The red arrows indicate the directions in which x, y, z increase in our right handed coordinate system. The purple arrows give an example of an angular velocity $\boldsymbol{\omega}$ and the direction it causes the green box to rotate.

Keeping track of the motion of the objects requires a large number of variables. This section's purpose is to collate all the relevant variable definitions into one place, making it hopefully easier to lookup what the variables in formulas mean. However, first let us go over some conventions and notation.

It is *essential* we are clear about our sign conventions and adhere to them strictly to ensure we don't introduce sign errors. We will use a right handed coordinate system, with axis labelled x, y, z, see Figure 3. An angular velocity $\boldsymbol{\omega}$ is a 3-dimensional vector. It indicates the number of radians per second by its magnitude $|\boldsymbol{\omega}|$, and the axis of rotation via its direction $\boldsymbol{\omega}/|\boldsymbol{\omega}|$. Our convention for the direction of rotation will be that an object moves anticlockwise when looking at it in the opposite direction to the axis of rotation, see Figure 3. We will always take an object's centre of mass (which I'll abbreviate to CM) as the point it is rotating about. I find this choice of pivot separates out the linear and angular motion more cleanly.

A lot of the variables we will be manipulating will be 3-dimensional vectors, such as the linear and angular velocities. A natural way to transform and manipulate vectors is via matrix multiplications. Taking the cross product of two vectors is a common operation that we will wish to perform. Let $\mathbf{a}$ and $\mathbf{b}$ be any two 3-dimensional vectors. We can write their cross product as follows

$$\begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix} \times \begin{pmatrix} b_x \\ b_y \\ b_z \end{pmatrix} = \begin{pmatrix} a_y b_z - a_z b_y \\ a_z b_x - a_x b_z \\ a_x b_y - a_y b_x \end{pmatrix} = \begin{pmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{pmatrix} \begin{pmatrix} b_x \\ b_y \\ b_z \end{pmatrix}.$$

Let $\tilde{\mathbf{a}}$ denote the matrix that is formed from $\mathbf{a}$, and has the same effect as the operation "$\mathbf{a}\times$" i.e.

$$\tilde{\mathbf{a}} = \begin{pmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{pmatrix}.$$

This notation will be useful later on. Throughout this document, I've attempted to highlight (with coloured boxes) equations which will be implemented, in the final algorithm. Hopefully this will make things easier, when it comes to looking up the derivation of parts of the implementation.

The collisions we are simulating will involve two rigid bodies, which we'll call object A and object B. We will assume that when they collide the objects intersect at a single point and it is at this point impulses will be applied to separate them. We will not cover what to do when the intersection consists of more than one point, e.g. when a "face-face" collision occurs. Our algorithm will take as input the point of contact and the direction the normal force acts. The impulse object A is subjected to we'll call $\mathbf{p}$. By Newton's 3rd law, forces are equal and opposite, so the impulse applied to object B will be $-\mathbf{p}$. The algorithm's goal will be to update the velocities of the objects by calculating $\mathbf{p}$. Since impulses happen instantaneously the position and orientation of the objects will remain unchanged after the collision, and so will the masses and inertia tensors.

Let us now define some variables, some of which will be used by our algorithm, and some of which are just used in deriving important formulas.

$\mathbf{p} =$ The impulse applied to object A from B. Its components are represented by p_x, p_y, p_z.
$\mathbf{f} =$ A unit vector indicating the direction the normal force (see Section 3.1) acts on object A from B.
$e =$ The coefficient of restitution between objects A and B.
$\mu =$ The coefficient of friction between objects A and B.

$m_A =$ The mass of object A.
$I_A =$ The inertia tensor of A in its current orientation, about its CM.
$\mathbf{r}_A =$ The displacement from the CM of A to the point of contact with B.
$\mathbf{v}_{0A} =$ The initial velocity of the CM of A, before applying any impulses.
$\boldsymbol{\omega}_{0A} =$ The initial angular velocity of the CM of A, before applying any impulses.
$\mathbf{v}_A =$ The velocity of the CM of A, after applying the impulse $\mathbf{p}$.
$\boldsymbol{\omega}_A =$ The angular velocity about the CM of A, after applying the impulse $\mathbf{p}$.

$m_B =$ The mass of object B.
$I_B =$ The inertia tensor of B in its current orientation, about its CM.
$\mathbf{r}_B =$ The displacement from the CM of B to the point of contact with A.
$\mathbf{v}_{0B} =$ The initial velocity of the CM of B, before applying any impulses.
$\boldsymbol{\omega}_{0B} =$ The initial angular velocity of the CM of B, before applying any impulses.
$\mathbf{v}_B =$ The velocity of the CM of B, after applying the impulse $-\mathbf{p}$.
$\boldsymbol{\omega}_B =$ The angular velocity about the CM of B, after applying the impulse $-\mathbf{p}$.

$\mathbf{u}_{0A} =$ The initial velocity of the point of contact on A, before an impulse is applied.
$\mathbf{u}_A =$ The velocity of the point of contact on A, after impulse $\mathbf{p}$ is applied.
$\mathbf{u}_{0B} =$ The initial velocity of the point of contact on B, before an impulse is applied.
$\mathbf{u}_B =$ The velocity of the point of contact on B, after impulse $-\mathbf{p}$ is applied.
$\mathbf{u}_0 = \mathbf{u}_{0A} - \mathbf{u}_{0B}$, the separation velocity of the objects, before any impulses have been applied.
$\mathbf{u} = \mathbf{u}_A - \mathbf{u}_B$, the separation velocity of the objects, after the impulse has been applied. Its components are represented by u_x, u_y, u_z.

t = Time.

t_{start} = The time at which the collision starts.

t_{end} = The time at which the collision ends.

t_s = The time at which the integration starts.

t_e = The time at which the integration ends.

δ = The step size during the numerical integration (a fixed small positive value).

$\mathbb{I}$ = The 3 by 3 identity matrix. (Not to be confused with I which indicates an inertia tensor.)

R = A rotation matrix.

K = The collision matrix. Used to calculate $\mathbf{u}$ from $\mathbf{p}$, see Section 3.2.
K_{ij} represents the entry in the ith row and jth column of the matrix K.

K^{-1} = The inverse of the collision matrix K. Used to calculate $\mathbf{p}$ from $\mathbf{u}$.
$(K^{-1})_{ij}$ represents the entry in the ith row and jth column of the matrix K^{-1}.

W = Work done by the normal forces during the collision.

W_c = Work done by the normal forces during compression.

W_d = Work done by the normal forces during decompression.

3 The Maths

This section is mostly just a subset of Chapter 3 of Brian Mirtich's thesis. There are a few differences though, such as a new termination condition for integration, and some proofs have been simplified.

In writing the algorithm I had to go through all the equations, deciding which ones were no longer valid, and which needed to be implemented. Since I was going through the calculations in detail anyway, I thought why not write it up as I go along. So this section is basically an exercise to remind myself in my own words what I think is going on, and where the equations come from. In Section 4 I will collate all the formulas I derive, so feel free to skip ahead, but if you're like me you might find this section interesting.

3.1 Rotating The Space

When two objects collide it is useful to split the force acting between them into two components: a normal force (that pushes the objects apart), and a tangential force (i.e. friction opposing the motion of the objects). The normal force is the force perpendicular to the contact surface. The direction of the normal force can usually be determined by examining the geometries of the objects at the point of contact. For example if a corner of object A makes contact with a face of object B, then the normal vector of the face should be the direction the normal force acts. There are, however, degenerate cases which can make things difficult, for example when the contact point lies on a corner of object A and on a corner of object B. Dealing with such things is not the aim of this document. We will assume that the direction of the normal force has been determined somehow and is given as an input to the collision algorithm.

The formulas we will be working with will be significantly simpler if the normal force acts in the direction $(0,0,1)^T$. If the normal force's direction is not $(0,0,1)^T$ we can rotate the space and objects to get an equivalent problem in which the force's direction is $(0,0,1)^T$. We can then resolve the collision using the simplified formulas and rotate the space back to its original orientation. Applying the rotations is relatively cheap in terms of CPU usage. We will see later that part of the collision resolution algorithm will involve integrating numerically, which is considerably more time consuming.

If the normal force acts in the direction given by the unit vector $\mathbf{f}$, then we can form the appropriate rotation transformation as follows. The axis of rotation is given by

$$\mathbf{n} = \frac{\mathbf{f} \times (0,0,1)^T}{|\mathbf{f} \times (0,0,1)^T|},$$

and the angle of rotation θ can be calculated by looking at

$$\begin{aligned} \mathbf{f} \cdot (0,0,1)^T &= \cos\theta, \\ |\mathbf{f} \times (0,0,1)^T| &= \sin\theta. \end{aligned}$$

From $\mathbf{n}$ and θ we can construct the rotation matrix

$$R = (1 - \cos\theta)\mathbf{n}\mathbf{n}^T + \cos\theta\mathbb{I} + \sin\theta\tilde{\mathbf{n}}$$

see Appendix A. (Care must be taken when $\mathbf{f}$ is very close or equal to $\pm(0,0,1)^T$ as division by zero errors may occur.)

We will assume from now on that the normal force that acts on object A from B does so in the direction $(0,0,1)^T$.

3.2 The Collision Matrix K

In this section we will derive the collision matrix K which is instrumental in resolving the collision. The collision matrix encapsulates the key properties of both objects. It is formed from object A and B's inertia tensors I_A, I_B, their masses m_A, m_B, and $\mathbf{r}_A, \mathbf{r}_B$ the relative positions of the point of contact from their centre of mass.

Before we go any further let's state some of the assumptions we'll be making about the collision. We are assuming the collision will happen over a negligible time period i.e. $t_{end} - t_{start} \approx 0$. Hence we are using impulses rather than forces, and consequently we can ignore external forces such as gravity. During the collision we will assume that the position, mass, and orientation of the objects remain constant (i.e. $\mathbf{r}_A, \mathbf{r}_B, m_A, m_B, I_A, I_B$ are all constant). You may be unhappy with these assumptions, but without them it seems very difficult to form equations we can use to make simulations. (If you do manage to remove/better justify these assumptions let me know).

Given the initial linear and angular velocities of objects A and B ($\mathbf{v}_{0A}, \mathbf{v}_{0B}, \boldsymbol{\omega}_{0A}, \boldsymbol{\omega}_{0B}$) our aim is to calculate the linear and angular velocities after the collision ($\mathbf{v}_A, \mathbf{v}_B, \boldsymbol{\omega}_A, \boldsymbol{\omega}_B$). By looking at the linear and angular momentum of the objects we get

$$\begin{aligned} \mathbf{p} &= m_A\mathbf{v}_A - m_A\mathbf{v}_{0A}, & \mathbf{r}_A \times \mathbf{p} &= I_A\boldsymbol{\omega}_A - I_A\boldsymbol{\omega}_{0A}, \\ -\mathbf{p} &= m_B\mathbf{v}_B - m_B\mathbf{v}_{0B}, & -\mathbf{r}_A \times \mathbf{p} &= I_B\boldsymbol{\omega}_B - I_B\boldsymbol{\omega}_{0B}, \end{aligned}$$

where $\mathbf{p}$ is the impulse applied during the collision. Hence if we know $\mathbf{p}$ we can work out the linear and angular velocities of the objects after the collision,

$$\begin{aligned} \mathbf{v}_A &= \mathbf{v}_{0A} + \frac{1}{m_A}\mathbf{p}, & \boldsymbol{\omega}_A &= \boldsymbol{\omega}_{0A} + I_A^{-1}(\mathbf{r}_A \times \mathbf{p}), \\ \mathbf{v}_B &= \mathbf{v}_{0B} - \frac{1}{m_B}\mathbf{p}, & \boldsymbol{\omega}_B &= \boldsymbol{\omega}_{0B} - I_B^{-1}(\mathbf{r}_B \times \mathbf{p}). \end{aligned} \tag{1}$$

To calculate $\mathbf{p}$ we need to look at the separation velocity of the objects at the point of contact, both immediately before and immediately after the collision. The velocities at the points of contact before and after the collision are

$$\begin{aligned} \mathbf{u}_{0A} &= \mathbf{v}_{0A} + \boldsymbol{\omega}_{0A} \times \mathbf{r}_A, & \mathbf{u}_A &= \mathbf{v}_A + \boldsymbol{\omega}_A \times \mathbf{r}_A, \\ \mathbf{u}_{0B} &= \mathbf{v}_{0B} + \boldsymbol{\omega}_{0B} \times \mathbf{r}_B, & \mathbf{u}_B &= \mathbf{v}_B + \boldsymbol{\omega}_B \times \mathbf{r}_B. \end{aligned}$$

Using these equations we can calculate the separation velocity after the collision

$$\begin{aligned} \mathbf{u} &= \mathbf{u}_A - \mathbf{u}_B \\ &= \mathbf{v}_A + \boldsymbol{\omega}_A \times \mathbf{r}_A - \mathbf{v}_B - \boldsymbol{\omega}_B \times \mathbf{r}_B \end{aligned}$$

and similarly the initial separation velocity $\mathbf{u}_0 = \mathbf{u}_{0A} - \mathbf{u}_{0B}$ is given by

$$\mathbf{u}_0 = \mathbf{v}_{0A} + \boldsymbol{\omega}_{0A} \times \mathbf{r}_A - \mathbf{v}_{0B} - \boldsymbol{\omega}_{0B} \times \mathbf{r}_B.$$

Hence the change in separation velocity is

$$\begin{aligned}\mathbf{u} - \mathbf{u}_0 &= (\mathbf{v}_A + \boldsymbol{\omega}_A \times \mathbf{r}_A - \mathbf{v}_B - \boldsymbol{\omega}_B \times \mathbf{r}_B) - (\mathbf{v}_{0A} + \boldsymbol{\omega}_{0A} \times \mathbf{r}_A - \mathbf{v}_{0B} - \boldsymbol{\omega}_{0B} \times \mathbf{r}_B) \\ &= (\mathbf{v}_A - \mathbf{v}_{0A}) - (\mathbf{v}_B - \mathbf{v}_{0B}) + (\boldsymbol{\omega}_A - \boldsymbol{\omega}_{0A}) \times \mathbf{r}_A - (\boldsymbol{\omega}_B - \boldsymbol{\omega}_{0B}) \times \mathbf{r}_B.\end{aligned}$$

Applying the equations in (1) gives

$$\begin{aligned}\mathbf{u} - \mathbf{u}_0 &= \frac{1}{m_A}\mathbf{p} + \frac{1}{m_B}\mathbf{p} + (I_A^{-1}(\mathbf{r}_A \times \mathbf{p})) \times \mathbf{r}_A + (I_B^{-1}(\mathbf{r}_B \times \mathbf{p})) \times \mathbf{r}_B \\ &= \frac{1}{m_A}\mathbf{p} + \frac{1}{m_B}\mathbf{p} - \mathbf{r}_A \times (I_A^{-1}(\mathbf{r}_A \times \mathbf{p})) - \mathbf{r}_B \times (I_B^{-1}(\mathbf{r}_B \times \mathbf{p})) \\ &= \frac{1}{m_A}\mathbf{p} + \frac{1}{m_B}\mathbf{p} - \tilde{\mathbf{r}}_A I_A^{-1}\tilde{\mathbf{r}}_A\mathbf{p} - \tilde{\mathbf{r}}_B I_B^{-1}\tilde{\mathbf{r}}_B\mathbf{p} \\ &= \left(\frac{1}{m_A}\mathbb{I} + \frac{1}{m_B}\mathbb{I} - \tilde{\mathbf{r}}_A I_A^{-1}\tilde{\mathbf{r}}_A - \tilde{\mathbf{r}}_B I_B^{-1}\tilde{\mathbf{r}}_B\right)\mathbf{p}\end{aligned}$$

Note that the right hand side of the equation is simply a 3 by 3 matrix times $\mathbf{p}$. This is *the collision matrix* which we denote by K to simplify notation. So now we can write

$$\mathbf{u} - \mathbf{u}_0 = K\mathbf{p}, \tag{2}$$

where

$$K = \frac{1}{m_A}\mathbb{I} + \frac{1}{m_B}\mathbb{I} - \tilde{\mathbf{r}}_A I_A^{-1}\tilde{\mathbf{r}}_A - \tilde{\mathbf{r}}_B I_B^{-1}\tilde{\mathbf{r}}_B.$$

Note that K is a constant during the collision due to the assumptions we made that $\mathbf{r}_A, \mathbf{r}_B, m_A, m_B, I_A$, and I_B are all constant. In the next section we will describe how to calculate $\mathbf{u}$ (using the fact that K is constant), and from this we can calculate $\mathbf{p}$ using

$$\mathbf{p} = K^{-1}(\mathbf{u} - \mathbf{u}_0).$$

Once we have $\mathbf{p}$, the equations in (1) give us the new velocities the objects will move with.

There is an issue of whether K^{-1} exists, it may be that K is singular or ill-defined because either I_A or I_B doesn't have an inverse. We can show this is not the case by showing that K, I_A, and I_B are all *positive definite* matrices (see Appendix B for a definition) and that such matrices always have an inverse which themselves are positive definite. Since these facts are relatively important, and will be used later, I'll expand the argument out a bit in Appendix B.

3.3 Calculating The Separation Velocity $\mathbf{u}$

Rather than think of $\mathbf{p}$ as the overall impulse that gets applied during the collision, we can think of it as a function $\mathbf{p}(t)$ which indicates the overall impulse that has been applied by time t. Similarly we can regard $\mathbf{u}$ as evolving over time. We know $\mathbf{p}(t_{start}) = \mathbf{0}$ and $\mathbf{u}(t_{start}) = \mathbf{u}_0$. Our aim is to work out $\mathbf{p}(t_{end})$ and $\mathbf{u}(t_{end})$.

If we differentiate impulses with respect to time we get forces. Due to the way we have orientated the objects (see Section 3.1) the normal force is just dp_z/dt (where p_z is the z component of $\mathbf{p}(t)$, I'll start dropping the "(t)" part for convenience). If the relative tangential velocity of the objects is non-zero then the objects are said to be *sliding*. Friction will oppose that motion, and according to Coulomb's law of friction the magnitude of the force should be $\mu dp_z/dt$, where μ is the coefficient of friction between the two bodies. We can deduce the direction of motion from u_x, u_y (components of $\mathbf{u}$). Hence the tangential forces are given by

$$\frac{dp_x}{dt} = -\mu \frac{dp_z}{dt} \frac{u_x}{\sqrt{u_x^2 + u_y^2}}, \qquad \frac{dp_y}{dt} = -\mu \frac{dp_z}{dt} \frac{u_y}{\sqrt{u_x^2 + u_y^2}}.$$

We will assume that during the collision the normal force is always positive, $dp_z/dt > 0$ (as it is constantly trying to push the objects apart). This means p_z monotonically increases with time, therefore we can use it instead of t to parameterize the progress of the collision. Just to be clear, now we are thinking of $\mathbf{p}$, and $\mathbf{u}$ as functions of p_z, i.e. $\mathbf{p}(p_z), \mathbf{u}(p_z)$. Using $d\mathbf{p}/dt$ we can calculate $d\mathbf{p}/dp_z$

$$\frac{d\mathbf{p}}{dp_z} = \frac{d\mathbf{p}}{dt}\frac{dt}{dp_z} = \begin{pmatrix} -\mu \frac{dp_z}{dt} \frac{u_x}{\sqrt{u_x^2+u_y^2}} \\ -\mu \frac{dp_z}{dt} \frac{u_y}{\sqrt{u_x^2+u_y^2}} \\ \frac{dp_z}{dt} \end{pmatrix} \frac{dt}{dp_z} = \begin{pmatrix} -\mu u_x / \sqrt{u_x^2 + u_y^2} \\ -\mu u_y / \sqrt{u_x^2 + u_y^2} \\ 1 \end{pmatrix}.$$

Since both K and $\mathbf{u}_0$ are constants during the collision we can use $\mathbf{u} - \mathbf{u}_0 = K\mathbf{p}$ (see equation (2)) to get

$$\frac{d\mathbf{u}}{dp_z} = K\frac{d\mathbf{p}}{dp_z}.$$

Substituting $d\mathbf{p}/dp_z$ means we have

$$\begin{pmatrix} du_x/dp_z \\ du_y/dp_z \\ du_z/dp_z \end{pmatrix} = K \begin{pmatrix} -\mu u_x / \sqrt{u_x^2 + u_y^2} \\ -\mu u_y / \sqrt{u_x^2 + u_y^2} \\ 1 \end{pmatrix}. \tag{3}$$

We can use this formula to numerically integrate $\mathbf{u}$ (with initial conditions $\mathbf{u} = \mathbf{u}_0$). However, there are two major problems: when do we stop numerically integrating $\mathbf{u}$, and what happens when $u_x = u_y = 0$? Providing we can solve these two problems we can calculate the value of $\mathbf{u}$ at the end of the collision. The value can then be used to update the velocities of the objects (see Section 3.2).

We will deal first with the question of when to stop integrating.

3.4 The Termination Condition

In Section 3.3 we derived a way to calculate $\mathbf{u}$ by numerically integrating a set of differential equations. However, it is unclear at what point we stop integrating. In the frictionless case ($\mu = 0$) a common approach is to use Newton's law of restitution which says that when

$$u_z = -eu_{0z}$$

then we should stop integrating, where e is the coefficient of restitution between the two bodies, and u_{0z} is the z component of $\mathbf{u}_0$. Unfortunately when we start looking at collisions with friction ($\mu > 0$)

this termination rule can cause the total energy of the system to increase after a collision which does not make sense. To remedy this, we use Stronge's hypothesis instead, which says

$$W_d = -e^2 W_c$$

where W_c, and W_d is the work done by the normal force during compression and decompression respectively. This ensures that energy is lost by forces acting in the z direction, and since the tangential frictional forces always oppose motion, we can guarantee that the total energy will decrease. Note that when there is no friction Stronge's hypothesis gives the same results as Newton's law of restitution.

So far what I've done is identical to that covered in Mirtich's thesis. However he was only assuming there was one compression phase and decompression phase. I believe there could occasionally be two, as such I interpret W_c as the work done in all the compression phases, and W_d as the sum of the work done in all the decompression phases. This is perhaps not what Stronge intended but that's what I'll use (if you know of a better condition let me know).

So how can we calculate W_c and W_d? We can tell whether the collision is in a compression or decompression phase by looking at u_z. If u_z is positive the objects are "moving away" from each other and so they must be decompressing (technically we are assuming that the collision is instantaneous so the positions remain fixed). If u_z is negative the objects are in a compression phase. Once we've determined whether the collision is in a compression or decompression phase we need to determine the amount of work being done. The rate of change of work with respect to time is power, and power is force times velocity. If we call W the work done by the normal force dp_z/dt, we get

$$\frac{dW}{dt} = \frac{dp_z}{dt} u_z.$$

We are numerically integrating with respect to p_z, not t. Fortunately changing parameters is simple

$$\frac{dW}{dp_z} = u_z. \tag{4}$$

As we numerically integrate $\mathbf{u}$ we can use the above formula to update W_c and W_d depending on the sign of u_z. When $W_d = -e^2 W_c$ is satisfied we stop integrating, and use the current value of $\mathbf{u}$ to calculate $\mathbf{p}$ and update the velocities of the objects. Fortunately, as we'll discuss in Section 3.11, the condition $W_d = -e^2 W_c$ is guaranteed to be satisfied eventually.

3.5 Dealing With $u_x = u_y = 0$

Now lets move our attention back to the situation of numerically integrating $\mathbf{u}$ when u_x, u_y become 0. The formula (3) given in Section 3.3 is no longer sufficient, as $\sqrt{u_x^2 + u_y^2} = 0$ and causes a division by 0 to occur. In fact when we were deriving (3) we explicitly made the assumption that the relative tangential velocity of the objects is non-zero.

Coulomb's law of friction tells us that the magnitude of the frictional force is at most $\mu dp_z/dt$. When the tangential velocity is non-zero we get the full amount of frictional force. When the tangential velocity is zero (i.e. $u_x = u_y = 0$) we have to decide whether the frictional force is sufficient to keep the velocity at 0. If it is we get *stable sticking*. If the tangential acceleration is too great for friction to

oppose we get *unstable sticking*, and the tangential velocity will become non-zero. Determining which of these two cases we lie in is our first task.

Stable sticking occurs when the frictional force is large enough to stop the tangential forces. This condition is encapsulated by

$$\sqrt{\left(\frac{dp_x}{dt}\right)^2 + \left(\frac{dp_y}{dt}\right)^2} \leq \mu \frac{dp_z}{dt}.$$

Squaring both sides and multiplying by $(dt/dp_z)^2$ gives a condition using p_z instead of t

$$\left(\frac{dp_x}{dp_z}\right)^2 + \left(\frac{dp_y}{dp_z}\right)^2 \leq \mu^2.$$

If stable sticking does occur then we know that $du_x/dp_z = du_y/dp_z = 0$ (both u_x and u_y should remain as zero). Also we know $d\mathbf{u}/dp_z = K d\mathbf{p}/dp_z$ (see Section 3.3), therefore

$$\frac{d\mathbf{p}}{dp_z} = K^{-1} \frac{d\mathbf{u}}{dp_z}$$

which implies

$$\begin{pmatrix} dp_x/dp_z \\ dp_y/dp_z \\ 1 \end{pmatrix} = K^{-1} \begin{pmatrix} 0 \\ 0 \\ du_z/dp_z \end{pmatrix} = \frac{du_z}{dp_z} \begin{pmatrix} (K^{-1})_{13} \\ (K^{-1})_{23} \\ (K^{-1})_{33} \end{pmatrix}$$

where $(K^{-1})_{ij}$ refers to the entry in the ith row and jth column of K^{-1}. (I've avoided using the notation K^{-1}_{ij} in case it is confused with $1/K_{ij}$.) Since $1 = (K^{-1})_{33} du_z/dp_z$, we have

$$\frac{du_z}{dp_z} = \frac{1}{(K^{-1})_{33}} \tag{5}$$

and consequently

$$\frac{dp_x}{dp_z} = \frac{(K^{-1})_{13}}{(K^{-1})_{33}}, \qquad \frac{dp_y}{dp_z} = \frac{(K^{-1})_{23}}{(K^{-1})_{33}}.$$

Substituting into our stable sticking condition and rearranging, gives the condition our algorithm will use

$$((K^{-1})_{13})^2 + ((K^{-1})_{23})^2 \leq \mu^2 ((K^{-1})_{33})^2.$$

You may be a little concerned that $du_z/dp_z = 1/(K^{-1})_{33}$ might be negative (which implies u_z may never become positive, and we will remain perpetually in a compression phase). Worse yet, what if $(K^{-1})_{33} = 0$? However, those of you who read Appendix B will know that K^{-1} is a positive definite matrix, and hence $(K^{-1})_{33} = (0,0,1)K^{-1}(0,0,1)^T > 0$.

3.6 Stable Sticking

If we are in the stable sticking case, we know for the rest of the collision u_x, u_y will remain at zero. Equations (4) and (5)

$$\frac{dW}{dp_z} = u_z, \qquad \frac{du_z}{dp_z} = \frac{1}{(K^{-1})_{33}}$$

tell us how W and u_z progress. These equations are sufficiently simple that we can derive a closed form solution. Since $du_z/dp_z > 0$ (see Section 3.5) we can change variables to get

$$\frac{dW}{du_z} = (K^{-1})_{33} u_z.$$

This trivially integrates to

$$W = \frac{(K^{-1})_{33}}{2} u_z^2 + c$$

where c is the constant of integration. If we call t_s, t_e the time we start and end the integration we get

$$W(t_e) - W(t_s) = \frac{(K^{-1})_{33}}{2} (u_z(t_e)^2 - u_z(t_s)^2). \tag{6}$$

When we hit $u_x = u_y = 0$ and stable sticking occurs we could either be in a compression phase or a decompression phase. If we are in a compression phase, we use the (6) to work out how much more work is done until $u_z = 0$ and we enter the decompression phase

$$W(t_e) - W(t_s) = -\frac{(K^{-1})_{33}}{2} u_z(t_s)^2.$$

Once we are in a decompression phase (either from when we first hit $u_x = u_y = 0$ or from going through a compression phase first) we know how much work we need to do in order for $W_d = -e^2 W_c$ to hold and the collision to end. We can use (6) to work out $u_z(t_{end})$

$$u_z(t_{end}) = \sqrt{\frac{2}{(K^{-1})_{33}} (W(t_e) - W(t_s)) + u_z(t_s)^2}.$$

At this point our algorithm will be done, we have $\mathbf{u}(t_{end})$, so we can calculate $\mathbf{p}(t_{end})$ and update the velocities of the objects.

3.7 Unstable Sticking

If $u_x = u_y = 0$ and $((K^{-1})_{13})^2 + ((K^{-1})_{23})^2 > \mu^2((K^{-1})_{33})^2$ holds then we are in the unstable sticking case. Although u_x, u_y are both zero, in the next "time step" at least one of them will be non-zero. However, what their new values will be is not obvious.

To gain some insight we can use (3) (see Section 3.3) to look at some trajectories of (u_x, u_y), see Figure 4 (we ignore the u_z component in order to get a 2-dimensional diagram which is easier to draw). The values of μ and K used were $\mu = 0.7$ and

$$K = \begin{pmatrix} 20 & 0 & 1 \\ 0 & 4 & 6 \\ 1 & 6 & 10 \end{pmatrix} \qquad \text{which implies} \qquad K^{-1} = \frac{1}{76} \begin{pmatrix} 4 & 6 & -4 \\ 6 & 199 & -120 \\ -4 & -120 & 80 \end{pmatrix}$$

(it is easy to check unstable sticking occurs). The Figure shows 16 trajectories, with start points

$$(u_x, u_y) = (r\cos(2\pi n/16), r\sin(2\pi n/16))$$

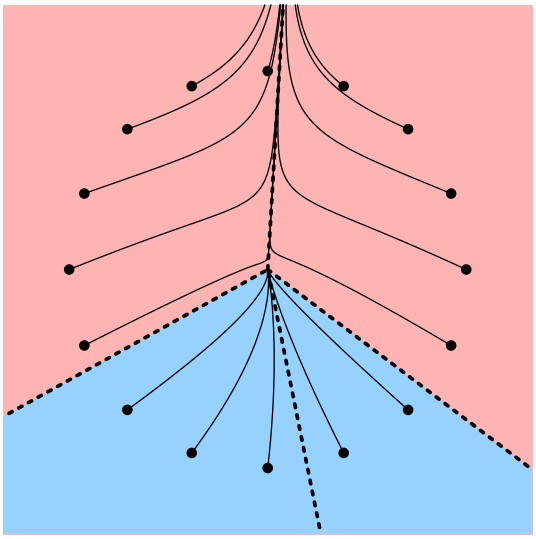

Figure 4: The trajectories of (u_x, u_y) (solid black lines). The dotted lines indicate rays of constant sliding. The initial value of (u_x, u_y) is indicated by the black circles. The origin is at the intersection of the dotted lines. The blue region indicates initial conditions which results in trajectories moving to the origin. Trajectories in the red region converge to the ray that is at approximately 87 degrees.

where n is an integer taking values between 0 and 15, and r is some fixed value. The start values are indicated by the black circles, and the black solid lines indicate how the values evolve over time (or p_z). The value of u_z does not play a part in determining the trajectory of (u_x, u_y). The value of r is also irrelevant in determining the shape of the curves. The trajectory for $r = 2$ looks the same as for $r = 1$ except it is scaled by a factor of 2.

The black dotted lines indicate initial positions that result in trajectories which appear as straight lines moving towards or away from the origin (i.e. on these lines $(du_x/dp_z, du_y/dp_z) = (\lambda u_x, \lambda u_y)$ for some λ). The rate at which (u_x, u_y) moves to or from the origin is a constant on these lines, as such we will call them *rays of constant sliding*. There are three types of rays: converging, stationary, and diverging. If (u_x, u_y) lies on a converging ray then it moves towards the origin. On a diverging ray (u_x, u_y) moves away from the origin, and (u_x, u_y) remains fixed on a stationary ray. In Figure 4 the rays of constant sliding occur at approximately $87, 209, 281$, and 323 degrees (anticlockwise from the positive u_x axis). The 87 degree ray is a diverging ray, the others are converging rays.

On examination of Figure 4 we see that the 5 trajectories at the bottom end up at the origin and the others converge towards the diverging ray at 87 degrees. A more detailed analysis would reveal that any trajectory starting in the blue region moves to the origin and any starting in the red region converges to the diverging ray. The rays at 209 and 323 degrees split the plane into the red and blue region.

Since Figure 4 looks the same (just smaller) when r is very small, we can deduce that trajectories very close to the origin either end up at the origin or move away from the origin following very closely to the diverging ray. Hence we will assume that when a trajectory gets to the origin (and we are in the unstable sticking case) that it leaves along the diverging ray. (This is also the only path which would not cause trajectories to cross.)

Resolving unstable sticking in this way is fine for the K, and μ which produced Figure 4, but in

general can we always find a diverging ray, and if there are two or more which one do we choose to leave along? It will turn out that in the unstable sticking case there is always a diverging ray and it is unique. We will prove this later in Section 3.10, but for now we'll just assume it is true.

3.8 Finding The Diverging Ray

If $u_x = u_y = 0$ and unstable sticking occurs then (u_x, u_y) must leave the origin. As discussed in the previous section it will leave along the diverging ray (which always exists and is unique). Hence the first thing we must determine is the direction of the diverging ray.

On a ray of constant sliding $(du_x/dp_z, du_y/dp_z)$ should be parallel to (u_x, u_y). We can find the rays of constant sliding by looking at (u_x, u_y) on the unit circle. Let $u_x = \cos\theta$ and $u_y = \sin\theta$, hence by (3) (see Section 3.3) we have

$$\frac{du_x}{dp_z} = -\mu K_{11}\cos\theta - \mu K_{12}\sin\theta + K_{13},$$
$$\frac{du_y}{dp_z} = -\mu K_{21}\cos\theta - \mu K_{22}\sin\theta + K_{23}.$$

A good way to test if vectors are parallel is to check if their cross product is $\mathbf{0}$. Hence we require

$$\begin{pmatrix}0\\0\\0\end{pmatrix} = \begin{pmatrix}u_x\\u_y\\0\end{pmatrix} \times \begin{pmatrix}du_x/dp_z\\du_y/dp_z\\0\end{pmatrix} = \begin{pmatrix}0\\0\\u_x(du_y/dp_z) - u_y(du_x/dp_z)\end{pmatrix},$$

or equivalently

$$\begin{aligned}0 &= u_x\frac{du_y}{dp_z} - u_y\frac{du_x}{dp_z}\\ &= \cos\theta(-\mu K_{21}\cos\theta - \mu K_{22}\sin\theta + K_{23}) - \sin\theta(-\mu K_{11}\cos\theta - \mu K_{12}\sin\theta + K_{13}).\end{aligned}$$

Recall that K is symmetric so $K_{12} = K_{21}$, consequently our condition becomes

$$0 = -\mu K_{12}(\cos^2\theta - \sin^2\theta) + \mu(K_{11} - K_{22})\sin\theta\cos\theta + K_{23}\cos\theta - K_{13}\sin\theta. \tag{7}$$

The roots of this equation give the angles of the rays of constant sliding.

We can reduce (7) to a quartic equation by using the trigonometry substitution of $p = \tan(\theta/2)$. There exist closed form solutions to finding the roots of a quartic equation, making them very quick to calculate. We will not discuss how to solve quartic equations in this document, but the information is readily available online.

We have to pay special attention to when $\theta = \pi$, as p becomes undefined. π is a root only when $0 = -\mu K_{12} - K_{23}$. Trigonometry identities tell us

$$\cos(\theta/2) = \frac{1}{\sqrt{1+p^2}}, \qquad \sin(\theta/2) = \frac{p}{\sqrt{1+p^2}},$$

hence

$$\cos\theta = \cos^2(\theta/2) - \sin^2(\theta/2) = \frac{1-p^2}{1+p^2}, \qquad \sin\theta = 2\sin(\theta/2)\cos(\theta/2) = \frac{2p}{1+p^2}.$$

Equation (7) now becomes

$$0 = -\mu K_{12}\frac{(1-p^2)^2 - 4p^2}{(1+p^2)^2} + \mu(K_{11} - K_{22})\frac{2p(1-p^2)}{(1+p^2)^2} + K_{23}\frac{1-p^2}{1+p^2} - K_{13}\frac{2p}{1+p^2}$$

which simplifies to

$$0 = -\mu K_{12}(1 - 6p^2 + p^4) + \mu(K_{11} - K_{22})(2p - 2p^3) + K_{23}(1 - p^4) - K_{13}(2p + 2p^3).$$

To summarize

the rays of constant sliding have angles which are the solutions to

$$a_4 p^4 + a_3 p^3 + a_2 p^2 + a_1 p + a_0 = 0$$

where

$$\begin{aligned} a_0 &= \mu K_{12} - K_{23} \\ a_1 &= 2K_{13} + 2\mu K_{22} - 2\mu K_{11} \\ a_2 &= -6\mu K_{12} \\ a_3 &= 2K_{13} + 2\mu K_{11} - 2\mu K_{22} \\ a_4 &= \mu K_{12} + K_{23} \\ p &= \tan(\theta/2). \end{aligned}$$

If $a_4 = 0$ then $\theta = \pi$ is also a solution.

So now we know how to calculate the rays of constant sliding, but we still need to work out which one is the diverging ray. We took the cross product of $(u_x, u_y, 0)^T$ and $(du_x/dp_z, du_y/dp_z, 0)^T$ to work out whether they were parallel, we'll take the dot product to see if they point in the same direction. If the dot product is positive then the ray is diverging. Rewriting this condition in terms of θ gives

$$-\mu K_{11}\cos^2\theta - \mu K_{22}\sin^2\theta - 2\mu K_{12}\sin\theta\cos\theta + K_{13}\cos\theta + K_{23}\sin\theta > 0.$$

3.9 Calculating u On The Diverging Ray

In the previous section we outlined how to find the diverging ray when unstable sticking occurs. Let ψ be the angle of the diverging ray. We can resolve unstable sticking by updating (u_x, u_y) from $(0, 0)$ to $(\varepsilon\cos\psi, \varepsilon\sin\psi)$, for some suitably small $\varepsilon > 0$, and then proceed numerically integrating according to (3). However, because we are on a ray of constant sliding we can find a closed form solution for the rest of the integration which will save time.

Before we continue we need to show that $du_z/dp_z > 0$ on the diverging ray (otherwise we could end

up trapped in a compression phase forever). Consider

$$\begin{aligned}\frac{du_z}{dp_z} &= (0,0,1)\left(\frac{d\mathbf{u}}{dp_z}\right)\\ &= (-\mu\cos\psi, -\mu\sin\psi, 1)\left(\frac{d\mathbf{u}}{dp_z}\right) + \mu(\cos\psi,\sin\psi,0)\left(\frac{d\mathbf{u}}{dp_z}\right)\\ &= (-\mu\cos\psi, -\mu\sin\psi, 1)K\begin{pmatrix}-\mu\cos\psi\\-\mu\sin\psi\\1\end{pmatrix} + \mu(\cos\psi,\sin\psi,0)\left(\frac{d\mathbf{u}}{dp_z}\right).\end{aligned}$$

The term on the left is positive because K is positive definite (see Appendix B). The term on the right is non-negative because the dot product, ignoring the z components, of $\mathbf{u}$ and $d\mathbf{u}/dp_z$ is positive. (This was the condition we used to determine whether a ray was diverging in Section 3.8.) Hence we have $du_z/dp_z > 0$ on a diverging ray.

On a diverging ray $\mathbf{u}$ progresses according to

$$\frac{d\mathbf{u}}{dp_z} = K\begin{pmatrix}-\mu\cos\psi\\-\mu\sin\psi\\1\end{pmatrix},$$

see equation (3). The right hand side is a constant, for convenience let us call it $\mathbf{k}$. We've just proved that $k_z = du_z/dp_z > 0$, and consequently we have

$$\frac{du_x}{du_z} = \frac{du_x}{dp_z}\frac{dp_z}{du_z} = \frac{k_x}{k_z}, \qquad \frac{du_y}{du_z} = \frac{du_y}{dp_z}\frac{dp_z}{du_z} = \frac{k_y}{k_z}, \qquad \frac{dW}{du_z} = \frac{dW}{dp_z}\frac{dp_z}{du_z} = \frac{u_z}{k_z}.$$

We can integrate to get u_x, u_y, and W in terms of u_z,

$$\begin{aligned}u_x(t_e) &= \frac{k_x}{k_z}(u_z(t_e) - u_z(t_s)) + u_x(t_s),\\ u_y(t_e) &= \frac{k_y}{k_z}(u_z(t_e) - u_z(t_s)) + u_y(t_s),\end{aligned}$$

$$W(t_e) - W(t_s) = \frac{1}{2k_z}(u_z(t_e)^2 - u_z(t_s)^2), \tag{8}$$

where t_s, and t_e are the times we start and end the integration respectively.

We know the value of u_z as it leaves the origin and if we are given the value of u_z at the end of the collision we will have the means to calculate the final values of u_x and u_y. Just as in the stable sticking case there are two situations to consider, whether we leave the origin in a compression phase or a decompression phase. If we are in a compression phase we can use (8) to calculate how much work will be done until $u_z = 0$,

$$W(t_e) - W(t_s) = -\frac{1}{2k_z}u_z(t_s)^2.$$

Once we are in a decompression phase, we know how much work has to be done in order to satisfy $W_d = -e^2 W_c$, and we can use (8) to calculate $u_z(t_{end})$,

$$u_z(t_{end}) = \sqrt{2k_z(W(t_e) - W(t_s)) + u_z(t_s)^2}.$$

We now have all the formulas we need for our algorithm, however there are a few loose ends to tie up. We need to show that in the unstable sticking case there always exists a unique diverging ray. We also need to show that the algorithm will always terminate.

3.10 Proving There Exists A Unique Diverging Ray

In this section we will show that there exists a diverging ray and that it is unique when unstable sticking occurs, i.e. when

$$((K^{-1})_{13})^2 + ((K^{-1})_{23})^2 > \mu^2((K^{-1})_{33})^2. \tag{9}$$

A really nice proof involving ellipses is given in Mirtich's thesis. However, it requires a few diagrams to properly explain, which I'm too lazy to draw, so instead I've opted to give a more equation based argument.

A diverging ray at an angle of θ satisfies

$$\begin{pmatrix} \lambda\cos\theta \\ \lambda\sin\theta \\ \alpha \end{pmatrix} = K \begin{pmatrix} -\mu\cos\theta \\ -\mu\sin\theta \\ 1 \end{pmatrix}$$

for some $\lambda > 0$ and α. Equivalently this condition can be written as

$$\lambda \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} = -\mu M \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} + \begin{pmatrix} K_{13} \\ K_{23} \end{pmatrix} \qquad \text{where} \qquad M = \begin{pmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{pmatrix},$$

or

$$\begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} = N_\lambda^{-1} \begin{pmatrix} K_{13} \\ K_{23} \end{pmatrix} \qquad \text{where} \qquad N_\lambda = \lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \mu M.$$

Consequently it is enough to show there exits a unique $\lambda > 0$ such that $N_\lambda^{-1}(K_{13}, K_{23})^T$ is a unit vector, when unstable sticking occurs.

Before we continue I need to point out some observations (you may need to read Appendix B first). Note that M is a positive definite matrix, the proof is as follows. By the definition of a positive definite matrix we need to show $(x, y)M(x, y)^T$ is positive for all $(x, y) \neq (0, 0)$, but $(x, y)M(x, y)^T = (x, y, 0)K(x, y, 0)^T$ which is positive because K is positive definite, and hence we are done. Since M is positive definite so is N_λ. Positive definite matrices always have inverses hence both M^{-1} and N_λ^{-1} exist, specifically

$$N_\lambda^{-1} = \frac{1}{(\lambda + \mu K_{11})(\lambda + \mu K_{22}) - \mu^2 K_{12}K_{21}} \begin{pmatrix} \lambda + \mu K_{22} & -\mu K_{12} \\ -\mu K_{21} & \lambda + \mu K_{11} \end{pmatrix}. \tag{10}$$

By considering the identity $KK^{-1} = \mathbb{I}$ we can deduce

$$K_{11}(K^{-1})_{13} + K_{12}(K^{-1})_{23} + K_{13}(K^{-1})_{33} = 0$$
$$K_{21}(K^{-1})_{13} + K_{22}(K^{-1})_{23} + K_{23}(K^{-1})_{33} = 0$$

and therefore

$$M \begin{pmatrix} (K^{-1})_{13} \\ (K^{-1})_{23} \end{pmatrix} + (K^{-1})_{33} \begin{pmatrix} K_{13} \\ K_{23} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

which means

$$\begin{pmatrix} K_{13} \\ K_{23} \end{pmatrix} = -\frac{1}{(K^{-1})_{33}} M \begin{pmatrix} (K^{-1})_{13} \\ (K^{-1})_{23} \end{pmatrix}. \tag{11}$$

We can now deal with the special case of when $\mu = 0$.

When $\mu = 0$,

$$N_\lambda^{-1} \begin{pmatrix} K_{13} \\ K_{23} \end{pmatrix} = \frac{1}{\lambda} \begin{pmatrix} K_{13} \\ K_{23} \end{pmatrix}$$

Clearly there exists a unique $\lambda > 0$ for which $(K_{13}, K_{23})^T/\lambda$ is a unit vector, provided $(K_{13}, K_{23})^T \neq (0,0)^T$. If $K_{13} = K_{23} = 0$ then by (11) we have $(K^{-1})_{13} = (K^{-1})_{23} = 0$, which cannot occur because (9) tells us that

$$((K^{-1})_{13})^2 + ((K^{-1})_{23})^2 > 0.$$

For the remainder of this section we can assume $\mu > 0$.

The magnitude of $N_\lambda^{-1}(K_{13}, K_{23})^T$ can be thought of as a function parameterized by λ. This function is continuous for $\lambda \geq 0$, see (10). Hence by showing that the magnitude is less than 1 for large λ and greater than 1 for $\lambda = 0$ by the intermediate value theorem we know there exits a value of λ for which the magnitude is precisely 1.

As λ tends to infinity it is easy to see from (10) that $N_\lambda^{-1}(K_{13}, K_{23})^T$ converges to $(0,0)^T$. Consequently there will exist a λ for which the magnitude of $N_\lambda^{-1}(K_{13}, K_{23})^T$ is less than 1.

For $\lambda = 0$ we have $N_\lambda = \mu M$ hence

$$N_\lambda^{-1} \begin{pmatrix} K_{13} \\ K_{23} \end{pmatrix} = \frac{1}{\mu} M^{-1} \begin{pmatrix} K_{13} \\ K_{23} \end{pmatrix}.$$

Substituting (11) gives

$$N_\lambda^{-1} \begin{pmatrix} K_{13} \\ K_{23} \end{pmatrix} = -\frac{1}{\mu(K^{-1})_{33}} \begin{pmatrix} (K^{-1})_{13} \\ (K^{-1})_{23} \end{pmatrix},$$

which has a magnitude greater than 1, because (9) tells us that

$$\left(\frac{(K^{-1})_{13}}{\mu(K^{-1})_{33}}\right)^2 + \left(\frac{(K^{-1})_{23}}{\mu(K^{-1})_{33}}\right)^2 > 1.$$

We've shown there exists a $\lambda > 0$ which produces a diverging ray, all that remains is to prove that the ray is unique. Suppose it is not unique, i.e. there exists $\lambda, \theta, \lambda', \theta'$ such that $\lambda, \lambda' > 0, \theta \neq \theta'$ and

$$\left(\lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \mu M\right) \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} = \left(\lambda' \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \mu M\right) \begin{pmatrix} \cos\theta' \\ \sin\theta' \end{pmatrix} = \begin{pmatrix} K_{13} \\ K_{23} \end{pmatrix}.$$

Therefore

$$\mu M \begin{pmatrix} \cos\theta - \cos\theta' \\ \sin\theta - \sin\theta' \end{pmatrix} = \lambda' \begin{pmatrix} \cos\theta' \\ \sin\theta' \end{pmatrix} - \lambda \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix},$$

hence

$$\begin{aligned} \begin{pmatrix} \cos\theta - \cos\theta' \\ \sin\theta - \sin\theta' \end{pmatrix}^T \mu M \begin{pmatrix} \cos\theta - \cos\theta' \\ \sin\theta - \sin\theta' \end{pmatrix} &= \begin{pmatrix} \cos\theta - \cos\theta' \\ \sin\theta - \sin\theta' \end{pmatrix}^T \left(\lambda' \begin{pmatrix} \cos\theta' \\ \sin\theta' \end{pmatrix} - \lambda \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix}\right) \\ &= (\lambda + \lambda')(\cos\theta\cos\theta' + \sin\theta\sin\theta' - 1) \\ &= -\frac{1}{2}(\lambda + \lambda') \begin{pmatrix} \cos\theta - \cos\theta' \\ \sin\theta - \sin\theta' \end{pmatrix}^T \begin{pmatrix} \cos\theta - \cos\theta' \\ \sin\theta - \sin\theta' \end{pmatrix}. \end{aligned}$$

This is a contradiction as the right hand side is positive (as M is positive definite, $\mu > 0$, $(\cos\theta - \cos\theta', \sin\theta - \sin\theta') \neq (0,0)$) and the left hand side is clearly negative. This completes the proof that in the unstable sticking case there exists a unique diverging ray.

As an afterthought it occurred to me that the condition $|N_\lambda^{-1}(K_{13}, K_{23})^T| = 1$ can be written as a quartic equation in λ. This quartic equation can be used as an alternative way to find the diverging ray to that given in Section 3.8.

3.11 Proving The Algorithm Terminates

Showing the algorithm terminates isn't too hard. All we need to do is show that eventually du_z/dp_z becomes and remains larger than some fixed positive constant. This ensures that u_z will eventually become positive and that the termination condition $W_d = -e^2 W_c$ will be satisfied.

First let's consider what happens if the trajectory of $\mathbf{u}$ starts on a ray of constant sliding (note that du_z/dp_z is a constant on a ray of constant sliding). If we start on a diverging ray, we've shown $du_z/dp_z > 0$ (see Section 3.9). If we start on a converging ray we'll eventually end up at the origin at which point we check for stable or unstable sticking. If unstable sticking occurs we'd leave along the diverging ray which is a case we've already covered. If stable sticking occurs we've shown $du_z/dp_z = 1/(K^{-1})_{33} > 0$ (see Section 3.6). If we're on a stationary ray then $du_x/dp_z = du_y/dp_z = 0$, in which case we can show $du_z/dp_z = 1/(K^{-1})_{33} > 0$ (the argument is the same as that given for stable sticking). Hence if we start on a ray of constant sliding the algorithm will terminate.

If the trajectory doesn't start on a ray of constant sliding then one of two things can happen. The first is that the trajectory reaches the origin, at which point stable or unstable sticking occurs, which we've shown causes the algorithm to terminate. The second is that the trajectory converges to a ray of constant sliding. (We will prove later that there always exists at least one ray of constant sliding.) To prove we have convergence to a ray we look at the trajectory in terms of polar coordinates (r, θ) rather than (u_x, u_y). We can easily show that

$$\frac{d\theta}{dp_z} = \frac{1}{r}\left(-\sin\theta\frac{du_x}{dp_z} + \cos\theta\frac{du_y}{dp_z}\right), \qquad \frac{dr}{dp_z} = \cos\theta\frac{du_x}{dp_z} + \sin\theta\frac{du_y}{dp_z}.$$

Note that $d\theta/dp_z$ is 0 on a ray of constant sliding. It is not too hard to show that θ will converge monotonically to a root of $d\theta/dp_z$ (as $rd\theta/dp_z$ and dr/dp_z are bounded continuous functions of θ). However, the $1/r$ factor does require some special attention (we have to show it doesn't cause θ to converge to something which is not a ray of constant sliding). I've decided to omit the technical details, but for those interested see Appendix C.

On a converging ray $dr/dp_z < 0$. If θ converges to a converging ray then (by the continuity of dr/dp_z and the monotonicity of the convergence of θ) eventually dr/dp_z will become and remain smaller than some fixed negative value. This means that the trajectory must end up at the origin, which will cause the algorithm to terminate. On a stationary and diverging ray we've shown $du_z/dp_z > 0$. Hence if θ is converging to a stationary or diverging ray (by the continuity of du_z/dp_z) eventually du_z/dp_z will become and remain larger than some fixed positive constant, causing the algorithm to terminate.

All that is left is to show that there always exists a ray of constant sliding. The rays occur at the roots

of

$$-\mu K_{12}(\cos^2\theta - \sin^2\theta) + \mu(K_{11} - K_{22})\sin\theta\cos\theta + K_{23}\cos\theta - K_{13}\sin\theta, \tag{12}$$

see equation (7) in Section 3.8. Using the fact that $a\cos x + b\sin x$ can always be written in the form $\alpha\sin(x+\beta)$ we can rewrite (12) as

$$c_1\sin(2\theta + c_2) + c_3\sin(\theta + c_4), \tag{13}$$

where c_1, c_2, c_3, c_4 are constants which can be determined from (12). Clearly (13) has a root if $c_1 = 0$ or $c_3 = 0$. The only interesting case to consider is if $c_1 \neq 0$ and $c_3 \neq 0$. We will show there exists a root by showing there exists a value of θ for which (13) is positive and a value for which it is negative. Hence by the intermediate value theorem a root must exist.

We can assume $c_3 > 0$ (the argument is virtually identical for $c_3 < 0$). Note that $c_3\sin(\theta + c_4)$ is positive when θ lies in the region $(-c_4, \pi - c_4)$. Furthermore $c_1\sin(2\theta + c_2)$ goes through an entire oscillation in that region and hence is at some point positive. Therefore there exists a value of $\theta \in (-c_4, \pi - c_4)$ for which (13) is positive. Similarly we can show there is a value of $\theta \in (\pi - c_4, 2\pi - c_4)$ for which (13) is negative. Consequently by the intermediate value theorem there exists a root.

4 The Algorithm

The algorithm is longer and scarier looking than I intended but it is not all that complicated, do not be put off by first impressions. In Section 5 there is a simplified version for 2D collisions which you may prefer.

The algorithm will involve numerically integrating a set of differential equations. We will use Euler's method to do the integration (other methods could be used but I've chosen Euler's method as it is simple and robust). As part of the method we need to choose a step size $\delta > 0$. The smaller the step size the more accurate the algorithm will be, but the longer it will take.

The algorithm takes as input

$$\delta, e, \mu, \mathbf{f}, m_A, I_A, \mathbf{r}_A, \mathbf{v}_{0A}, \boldsymbol{\omega}_{0A}, m_B, I_B, \mathbf{r}_B, \mathbf{v}_{0B}, \boldsymbol{\omega}_{0B},$$

and returns as output

$$\mathbf{v}_A, \boldsymbol{\omega}_A, \mathbf{v}_B, \boldsymbol{\omega}_B.$$

The description of the algorithm is given below.

1. We need to rotate the space so that the normal force acts in the direction $(0,0,1)^T$.

   Calculate the rotation matrix

   $$R = (1 - \cos\theta)\mathbf{n}\mathbf{n}^T + \cos\theta\mathbb{I} + \sin\theta\tilde{\mathbf{n}}$$

   where

   $$\mathbf{n} = \frac{\mathbf{f} \times (0,0,1)^T}{|\mathbf{f} \times (0,0,1)^T|}, \quad \cos\theta = \mathbf{f} \cdot (0,0,1)^T, \quad \sin\theta = |\mathbf{f} \times (0,0,1)^T|,$$

   and $\tilde{\mathbf{n}}$ is formed from the components of $\mathbf{n}$,

   $$\tilde{\mathbf{n}} = \begin{pmatrix} 0 & -n_z & n_y \\ n_z & 0 & -n_x \\ -n_y & n_x & 0 \end{pmatrix}.$$

   (If $\mathbf{f}$ is close or equal to $\pm(0,0,1)^T$ the above formulas may cause a division by zero to occur, this is a special case which must be dealt with separately.) Once we have calculated R we can rotate the problem by replacing

   $$I_A, \mathbf{r}_A, \mathbf{v}_{0A}, \boldsymbol{\omega}_{0A}, I_B, \mathbf{r}_B, \mathbf{v}_{0B}, \boldsymbol{\omega}_{0B}$$

   with

   $$RI_AR^{-1}, R\mathbf{r}_A, R\mathbf{v}_{0A}, R\boldsymbol{\omega}_{0A}, RI_BR^{-1}, R\mathbf{r}_B, R\mathbf{v}_{0B}, R\boldsymbol{\omega}_{0B}$$

   respectively. Note that taking the transpose of R is a quick way to calculate its inverse ($R^T = R^{-1}$).

2. Calculate the initial separation velocity

   $$\mathbf{u}_0 = \mathbf{v}_{0A} + \boldsymbol{\omega}_{0A} \times \mathbf{r}_A - \mathbf{v}_{0B} - \boldsymbol{\omega}_{0B} \times \mathbf{r}_B.$$

   (It might be a good idea at this point to check that the z component of $\mathbf{u}_0$ is negative, if it isn't then the objects aren't colliding and the algorithm should terminate.)

Calculate the collision matrix

$$K = \frac{1}{m_A}\mathbb{I} + \frac{1}{m_B}\mathbb{I} - \tilde{\mathbf{r}}_A I_A^{-1}\tilde{\mathbf{r}}_A - \tilde{\mathbf{r}}_B I_B^{-1}\tilde{\mathbf{r}}_B,$$

where $\tilde{\mathbf{r}}_A$ and $\tilde{\mathbf{r}}_B$ are formed from the components of $\mathbf{r}_A$ and $\mathbf{r}_B$ respectively (see Step 1, or Section 2).

Set

$$\mathbf{u} = \mathbf{u}_0, \quad W_c = 0, \quad W_d = 0.$$

3. *The integration loop.*

   (a) If $u_x = u_y = 0$ (or approximately 0 for robustness) sticking has occurred, exit the loop and go to Step 4.

   (b) Sticking has not occurred.
   Check whether we are in a compression phase or a decompression phase. If $u_z < 0$ increment W_c by $u_z\delta$, otherwise increment W_d by $u_z\delta$.
   Update $\mathbf{u}$ to

$$\begin{pmatrix} u_x \\ u_y \\ u_z \end{pmatrix} + \delta K \begin{pmatrix} -\mu u_x/\sqrt{u_x^2+u_y^2} \\ -\mu u_y/\sqrt{u_x^2+u_y^2} \\ 1 \end{pmatrix}.$$

   (c) If $W_d \geq -e^2 W_c$ the integration has ended, go to Step 7, otherwise go to Step 3a to continue the integration.

4. Sticking has occurred, check if it is stable or unstable.
   If $((K^{-1})_{13})^2 + ((K^{-1})_{23})^2 \leq \mu^2((K^{-1})_{33})^2$ holds then go to Step 5 else go to Step 6.

5. *Stable sticking has occurred.*

   (a) If $u_z < 0$ go to Step 5b else go to Step 5c.

   (b) We are in a compression phase.
   Increment W_c by $-(K^{-1})_{33}u_z^2/2$, set u_z to 0, then go to Step 5c.

   (c) We are in a decompression phase.
   Set the value of u_z to

$$\sqrt{\frac{2}{(K^{-1})_{33}}(-e^2 W_c - W_d) + u_z^2}.$$

   (You may also want to remove any rounding errors in u_x and u_y by setting them to 0.)
   Go to Step 7.

6. *Unstable sticking has occurred.*

   (a) We need to find the unique diverging ray of constant sliding.
   The rays of constant sliding have angles which are the solutions to

$$a_4 p^4 + a_3 p^3 + a_2 p^2 + a_1 p + a_0 = 0$$

where

$$\begin{aligned} a_0 &= \mu K_{12} - K_{23}, \\ a_1 &= 2K_{13} + 2\mu K_{22} - 2\mu K_{11}, \\ a_2 &= -6\mu K_{12}, \\ a_3 &= 2K_{13} + 2\mu K_{11} - 2\mu K_{22}, \\ a_4 &= \mu K_{12} + K_{23}, \\ p &= \tan(\theta/2). \end{aligned}$$

If $a_4 = 0$ then $\theta = \pi$ is also a solution. There exists closed form solutions to quartic equations (which can be found online). Once we have determined the rays of constant sliding we need to determine which one is diverging (there will be precisely one). The diverging ray is the one which satisfies

$$-\mu K_{11}\cos^2\theta - \mu K_{22}\sin^2\theta - 2\mu K_{12}\sin\theta\cos\theta + K_{13}\cos\theta + K_{23}\sin\theta > 0.$$

(b) Let ψ be the angle of the diverging ray (as determined by the previous step). Set k_x, k_y, k_z according to

$$\begin{pmatrix} k_x \\ k_y \\ k_z \end{pmatrix} = K \begin{pmatrix} -\mu\cos\psi \\ -\mu\sin\psi \\ 1 \end{pmatrix}.$$

Store a copy of the value of u_z in u_{old}.

If $u_z < 0$ go to Step 6c else go to Step 6d.

(c) We are in a compression phase.

Increment W_c by $-u_z^2/(2k_z)$, set u_z to 0, then go to Step 6d.

(d) We are in a decompression phase.

Set the value of u_z to

$$\sqrt{2k_z(-e^2W_c - W_d) + u_z^2}.$$

Go to Step 6e.

(e) Update u_x to

$$\frac{k_x}{k_z}(u_z - u_{old}) + u_x,$$

and update u_y to

$$\frac{k_y}{k_z}(u_z - u_{old}) + u_y.$$

We have now calculated the final value of $\mathbf{u}$, go to Step 7.

7. We know the value of $\mathbf{u}$ at the end of the collision. We can use it to calculate the impulse,

$$\mathbf{p} = K^{-1}(\mathbf{u} - \mathbf{u}_0).$$

From this we can calculate the new velocities of the objects

$$\begin{aligned} \mathbf{v}_A &= \mathbf{v}_{0A} + \frac{1}{m_A}\mathbf{p}, & \boldsymbol{\omega}_A &= \boldsymbol{\omega}_{0A} + I_A^{-1}(\mathbf{r}_A \times \mathbf{p}), \\ \mathbf{v}_B &= \mathbf{v}_{0B} - \frac{1}{m_B}\mathbf{p}, & \boldsymbol{\omega}_B &= \boldsymbol{\omega}_{0B} - I_B^{-1}(\mathbf{r}_B \times \mathbf{p}). \end{aligned}$$

8. The final step is to rotate the space back to its original orientation. We replace

$$\mathbf{v}_A, \boldsymbol{\omega}_A, \mathbf{v}_B, \boldsymbol{\omega}_B$$

with

$$R^{-1}\mathbf{v}_A, R^{-1}\boldsymbol{\omega}_A, R^{-1}\mathbf{v}_B, R^{-1}\boldsymbol{\omega}_B$$

respectively. The algorithm terminates outputting $\mathbf{v}_A, \boldsymbol{\omega}_A, \mathbf{v}_B, \boldsymbol{\omega}_B$.

Often collisions will happen with essentially immovable objects, such as the ground. We can simulate this by giving the immovable object's mass and inertia tensor very large but finite values. Another solution is to derive the equations and algorithm under the assumption that one object is immovable and its motion remains unaffected by the collision. The equations and the algorithm are virtually identical so I will omit the details but the necessary modifications to the algorithm are as follows. If object A is immovable simply replace occurrences of $1/m_A$ and I_A^{-1} with 0 and the zero matrix (a matrix whose entries are all zero) respectively. Similarly if object B is immovable replace $1/m_B$ and I_B^{-1} with 0 and the zero matrix.

5 2D Collisions

It occurred to me as an afterthought that if we are only interested in objects living in a 2-dimensional plane (i.e. the objects are planar, there is no z-component to their velocities, and all rotations occur about the z-axis) then the collision resolution algorithm becomes significantly simpler. This is because all the trajectories of $\mathbf{u}$ would lie on rays of constant sliding, hence we can write down closed form solutions to the integration avoiding the costly numerical integration loop. Also finding the diverging ray becomes trivial, meaning we don't have to implement a quartic equation solver.

I suspect a fair number of people would be interested in the simplified 2D version of the algorithm, so I've provided it. I won't go over the maths as 2D collisions are just a special subcase of 3D collisions.

5.1 Variables And Conventions

Before we delve into the algorithm I should highlight the fact that I'm lazy and have re-used the variable names from the 3D algorithm. Unfortunately the dimensionality of some of the variables have changed which may cause some confusion (the meanings of the variables, however, remain unchanged).

The variables

$$e, \mu, m_A, m_B, W_c, W_d$$

remain as scalars. We are assuming the collision takes place in the xy-plane so there is no longer a need for the z-component in vectors. The vectors

$$\mathbf{p}, \mathbf{u}, \mathbf{u}_0, \mathbf{r}_A, \mathbf{r}_B, \mathbf{v}_{0A}, \mathbf{v}_{0B}, \mathbf{v}_A, \mathbf{v}_B, \mathbf{f},$$

are now all 2-dimensional. The rotation matrix R and the collision matrix K (as well as their inverses) are 2 by 2 matrices (rather than 3 by 3). In place of 3 by 3 matrices representing inertia tensors we instead use the moment of inertia in the 2D case. Hence I_A and I_B become scalars. The angular velocities

$$\omega_{0A}, \omega_{0B}, \omega_A, \omega_B,$$

also become scalars. The convention we will use for angular velocities is that a positive value indicates anticlockwise motion. Just to be crystal clear, if object A has its centre of mass at the origin and is experiencing a constant angular velocity of ω_A then the point $(1, 0)$ on A rotates to the point $(\cos(\omega_A t), \sin(\omega_A t))$ at time t.

5.2 The Algorithm

As in Section 4 the algorithm looks long and horrible. However, it is really not that bad. It should be easier to implement than its 3D counterpart and take significantly less time to resolve collisions.

The algorithm takes as input

$$e, \mu, \mathbf{f}, m_A, I_A, \mathbf{r}_A, \mathbf{v}_{0A}, \omega_{0A}, m_B, I_B, \mathbf{r}_B, \mathbf{v}_{0B}, \omega_{0B},$$

and returns as output

$$\mathbf{v}_A, \omega_A, \mathbf{v}_B, \omega_B.$$

The description of the algorithm is given below.

1. We need to rotate the space so that the normal force acts in the direction $(0,1)^T$.
   Calculate the rotation matrix
   $$R = \begin{pmatrix} f_y & -f_x \\ f_x & f_y \end{pmatrix},$$
   where f_x, f_y are the components of the unit vector $\mathbf{f}$.
   Once we have calculated R we can rotate the problem by replacing
   $$\mathbf{r}_A, \mathbf{v}_{0A}, \mathbf{r}_B, \mathbf{v}_{0B}$$
   with
   $$R\mathbf{r}_A, R\mathbf{v}_{0A}, R\mathbf{r}_B, R\mathbf{v}_{0B}$$
   respectively.

2. Calculate the initial separation velocity
   $$\mathbf{u}_0 = \mathbf{v}_{0A} + \omega_{0A} \begin{pmatrix} -r_{Ay} \\ r_{Ax} \end{pmatrix} - \mathbf{v}_{0B} - \omega_{0B} \begin{pmatrix} -r_{By} \\ r_{Bx} \end{pmatrix},$$
   where $r_{Ax}, r_{Ay}, r_{Bx}, r_{By}$ are the components of $\mathbf{r}_A$ and $\mathbf{r}_B$. (It might be a good idea at this point to check that the y component of $\mathbf{u}_0$ is negative, if it isn't then the objects aren't colliding and the algorithm should terminate.)
   Calculate the collision matrix
   $$K = \left(\frac{1}{m_A} + \frac{1}{m_B}\right)\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \frac{1}{I_A}\begin{pmatrix} r_{Ay}^2 & -r_{Ax}r_{Ay} \\ -r_{Ax}r_{Ay} & r_{Ax}^2 \end{pmatrix} + \frac{1}{I_B}\begin{pmatrix} r_{By}^2 & -r_{Bx}r_{By} \\ -r_{Bx}r_{By} & r_{Bx}^2 \end{pmatrix}.$$
   Set
   $$\mathbf{u} = \mathbf{u}_0, \quad W_c = 0, \quad W_d = 0.$$

3. Check the type of ray we are on.
   (We will store the value of du_x/dp_y and du_y/dp_y in k_x and k_y respectively.)
   (a) If $u_x = 0$ sticking has occurred, go to Step 5.
   (b) If $u_x > 0$ set
   $$\begin{pmatrix} k_x \\ k_y \end{pmatrix} = K \begin{pmatrix} -\mu \\ 1 \end{pmatrix},$$
   and check the value of k_x.
   If $k_x \geq 0$ we are on a diverging or stationary ray go to Step 8.
   If $k_x < 0$ we are on a converging ray go to Step 4.
   (c) If $u_x < 0$ set
   $$\begin{pmatrix} k_x \\ k_y \end{pmatrix} = K \begin{pmatrix} \mu \\ 1 \end{pmatrix},$$
   and check the value of k_x.
   If $k_x > 0$ we are on a converging ray go to Step 4.
   If $k_x \leq 0$ we are on a diverging or stationary ray go to Step 8.

4. *We are on a converging ray.*

   (a) Set
   $$u_{origin} = u_y - \frac{k_y}{k_x} u_x.$$
   (u_{origin} is the value u_y would reach if we integrated $\mathbf{u}$ until u_x became 0.)

   (b) If $u_{origin} \leq 0$
   then we'll reach the origin (i.e. $u_x = 0$) before we enter the decompression phase, set
   $$W_c = -\frac{u_x u_y}{k_x} + \frac{k_y u_x^2}{2k_x^2},$$
   then set
   $$u_x = 0, \qquad u_y = u_{origin},$$
   and go to Step 5.

   (c) If $0 < u_{origin} < -eu_y$
   then we'll reach the origin before the decompression phase finishes, set
   $$W_c = -\frac{u_y^2}{2k_y}, \qquad W_d = \frac{u_{origin}^2}{2k_y},$$
   then set
   $$u_x = 0, \qquad u_y = u_{origin},$$
   and go to Step 5.

   (d) If $-eu_y \leq u_{origin}$
   then the decompression phase finishes before we reach the origin, update u_x to
   $$u_x - (1+e)\frac{k_x}{k_y} u_y,$$
   then update u_y to $-eu_y$, and go to Step 9.

5. Sticking has occurred, check if it is stable or unstable.
   If $|K_{12}| \leq \mu K_{11}$ holds then go to Step 6 else go to Step 7.

6. *Stable sticking has occurred.*
   Set
   $$\begin{pmatrix} k_x \\ k_y \end{pmatrix} = \begin{pmatrix} 0 \\ 1/(K^{-1})_{22} \end{pmatrix}$$
   then go to Step 8.

7. *Unstable sticking has occurred.* We need to find the unique diverging ray.
   Set
   $$\begin{pmatrix} k_x \\ k_y \end{pmatrix} = K \begin{pmatrix} -\mu \\ 1 \end{pmatrix}.$$
   If $k_x > 0$ then go to Step 8, otherwise set
   $$\begin{pmatrix} k_x \\ k_y \end{pmatrix} = K \begin{pmatrix} \mu \\ 1 \end{pmatrix}.$$
   then go to Step 8.

8. *We are on a diverging/stationary ray (or stable sticking has occurred).*

   (a) Store a copy of the value of u_y in u_{old}.
   If $u_y < 0$ go to Step 8b else go to Step 8c.

   (b) We are in a compression phase.
   Increment W_c by $-u_y^2/(2k_y)$, set u_y to 0, then go to Step 8c.

   (c) We are in a decompression phase.
   Set the value of u_y to
   $$\sqrt{2k_y(-e^2W_c - W_d) + u_y^2}.$$
   Update u_x to
   $$\frac{k_x}{k_y}(u_y - u_{old}) + u_x.$$
   We have now calculated the final value of $\mathbf{u}$, go to Step 9.

9. We know the value of $\mathbf{u}$ at the end of the collision. We can use it to calculate the impulse,
   $$\mathbf{p} = K^{-1}(\mathbf{u} - \mathbf{u}_0).$$
   From this we can calculate the new velocities of the objects
   $$\mathbf{v}_A = \mathbf{v}_{0A} + \frac{1}{m_A}\mathbf{p}, \qquad \omega_A = \omega_{0A} + \frac{1}{I_A}\begin{pmatrix} -r_{Ay} \\ r_{Ax} \end{pmatrix} \cdot \mathbf{p},$$
   $$\mathbf{v}_B = \mathbf{v}_{0B} - \frac{1}{m_B}\mathbf{p}, \qquad \omega_B = \omega_{0B} - \frac{1}{I_B}\begin{pmatrix} -r_{By} \\ r_{Bx} \end{pmatrix} \cdot \mathbf{p}.$$

10. The final step is to rotate the space back to its original orientation.
    We replace $\mathbf{v}_A, \mathbf{v}_B$ with $R^{-1}\mathbf{v}_A, R^{-1}\mathbf{v}_B$ respectively. Note that taking the transpose of R is a quick way to calculate its inverse ($R^T = R^{-1}$). The algorithm terminates outputting $\mathbf{v}_A, \omega_A, \mathbf{v}_B, \omega_B$.

The algorithm can be modified to handle immovable objects just as we did in Section 4. We simply replace instances of $1/m_A$ and $1/I_A$ with 0 if object A is immovable, or $1/m_B$ and $1/I_B$ with 0 if object B is.

6 Final Remarks

Let me re-iterate that the majority of the arguments, equations, and algorithms in this document come from Brian Mirtich's thesis. I urge you to check it out for more details (as well as references to the work which led to this algorithm).

I haven't been paid to write this document (I'm currently unemployed). I've spent my free time writing it because I think Mirtich's algorithm is sufficiently interesting and useful that the error I found deserves to be corrected. I make no claims to the validity or accuracy of the equations / arguments / algorithms I've outlined, so don't blame / sue me if you base some critical program on them and it ends up failing and or crashing causing damage.

I've given out my e-mail on the title page of this document, so get in touch if you found this document useful or you have some complaint. (I can't guarantee I'll reply to your e-mail or that it won't end up accidently in my junk mail folder, also don't contact me via instant messenger as I may block you.)

I'm happy to receive e-mails about the following:

- You've found an error / mistake.
- You agree / disagree that there is a mistake in Mirtich's thesis.
- You don't understand the arguments presented and want some help. Please don't ask me to explain basic things like vectors, matrices, positive definite matrices, etc. ask on the various online forums first, and if you're still stuck then get in touch.
- You have some suggestions to improve the mathematical argument or the document in general.
- You have some real-life data which shows the algorithm is good / bad at simulating collisions.
- You're implementing one of the algorithms (I'm particularly interested in e-mails on this topic). It doesn't matter whether your implementing them for some serious large commercial project or just personally for fun.
- You are using part or all of the document to teach others.
- You refer to this document in some way (e.g. an html link to this document, or a reference in an academic paper).
- You've found the document interesting / useful and just want to let me know.

Do get in touch as I'm interested in what people think of this document and how it gets used.

Appendices

A Constructing The Rotation Matrix

Given a unit vector $\mathbf{n}$ which gives the axis you want to rotate about, and an angle θ, we can construct a rotation matrix R. To work out what the formula for R should be, we will write down a formula in terms of $\mathbf{n}$ and θ for $R\mathbf{x}$, where $\mathbf{x}$ is a generic vector.

First we'll carefully choose three orthogonal axes represented by the vectors $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$. Let $\mathbf{e}_1 = \mathbf{n}$, this is a natural choice and will help make rotating about $\mathbf{n}$ easier. The component of $\mathbf{x}$ in this direction is $(\mathbf{n} \cdot \mathbf{x})\mathbf{n}$. Let $\mathbf{e}_2 = \mathbf{x} - (\mathbf{n} \cdot \mathbf{x})\mathbf{n}$, this is the part of $\mathbf{x}$ perpendicular to $\mathbf{n}$. The remaining axis $\mathbf{e}_3$ is determined by the other two since we want the axes to be orthogonal. Hence $\mathbf{e}_3 = \mathbf{e}_1 \times \mathbf{e}_2 = \mathbf{n} \times (\mathbf{x} - (\mathbf{n} \cdot \mathbf{x})\mathbf{n}) = \mathbf{n} \times \mathbf{x}$. The magnitude of $\mathbf{e}_1$ is 1 as it is just the unit vector $\mathbf{n}$. The magnitude of $\mathbf{e}_2$ is the same as $\mathbf{e}_3$ (due to the fact that all three axis are orthogonal and $\mathbf{e}_3 = \mathbf{e}_1 \times \mathbf{e}_2$, hence $|\mathbf{e}_3| = |\mathbf{e}_1||\mathbf{e}_2| = |\mathbf{e}_2|$). This will be useful in simplifying the calculation.

We can write $\mathbf{x}$ in terms of the orthogonal vectors, as $\mathbf{x} = (\mathbf{n} \cdot \mathbf{x})\mathbf{e}_1 + 1\mathbf{e}_2 + 0\mathbf{e}_3$. Rotating about $\mathbf{e}_1$ is now very simple, we get

$$R\mathbf{x} = (\mathbf{n} \cdot \mathbf{x})\mathbf{e}_1 + \cos\theta\mathbf{e}_2 + \sin\theta\mathbf{e}_3.$$

Substituting the values of $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$, tells us

$$\begin{aligned} R\mathbf{x} &= (\mathbf{n} \cdot \mathbf{x})\mathbf{n} + \cos\theta(\mathbf{x} - (\mathbf{n} \cdot \mathbf{x})\mathbf{n}) + \sin\theta\mathbf{n} \times \mathbf{x} \\ &= \mathbf{n}(\mathbf{n}^T\mathbf{x}) + \cos\theta(\mathbf{x} - \mathbf{n}(\mathbf{n}^T\mathbf{x})) + \sin\theta\tilde{\mathbf{n}}\mathbf{x} \\ &= (\mathbf{n}\mathbf{n}^T + \cos\theta(\mathbb{I} - \mathbf{n}\mathbf{n}^T) + \sin\theta\tilde{\mathbf{n}})\mathbf{x}. \end{aligned}$$

Since $\mathbf{x}$ was a generic vector, we are free to remove it from both sides of the equation, to get the desired result that

$$R = (1 - \cos\theta)\mathbf{n}\mathbf{n}^T + \cos\theta\mathbb{I} + \sin\theta\tilde{\mathbf{n}}.$$

B Showing K Is Positive Definite

Firstly, let's define the term *positive definite*. A symmetric n by n matrix M is *positive definite* if for all non-zero vectors $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{x}^T M\mathbf{x} > 0$. If instead the weaker property $\mathbf{x}^T M\mathbf{x} \geq 0$ held true then we would call M *positive semidefinite*.

Before we prove K is positive definite we need to outline some properties of positive definite matrices. If M_1 is positive definite and M_2 is positive (semi)definite then $M_1 + M_2$ is positive definite, because $\mathbf{x}^T(M_1 + M_2)\mathbf{x} = \mathbf{x}^T M_1\mathbf{x} + \mathbf{x}^T M_2\mathbf{x} > 0$. Another useful fact is that every symmetric matrix S can be written in the form $S = R^T DR$, where R is an *orthogonal matrix* (a matrix which is equivalent to a rotation transformation) and D is a *diagonal matrix* (a matrix whose only non-zero entries are D_{ii} for $i = 1, \ldots, n$). Furthermore the entries on the diagonal of D are the eigenvalues of S. I won't attempt to prove these facts, but they should be covered in any undergraduate text on linear algebra.

Next let us show that positive definite matrices always have inverses which themselves are positive definite. A positive definite matrix is symmetric so can be written as $R^T DR$ for an appropriate choice of orthogonal matrix R and diagonal matrix D. Hence $\mathbf{x}^T R^T DR\mathbf{x} = (R\mathbf{x})^T D(R\mathbf{x}) > 0$ holds for $\mathbf{x} \neq \mathbf{0}$. Since R is orthogonal it is invertible ($R^{-1} = R^T$) therefore $R\mathbf{x}$ can be any non-zero vector, which implies D must be positive definite. It is not hard to see that a diagonal matrix is positive definite if and only if it has positive values on its diagonal. We can construct the inverse of a diagonal matrix D by taking the reciprocal of its diagonal entries. Using these facts we can construct $R^T D^{-1} R$ the inverse of the positive definite matrix $R^T DR$. The inverse has eigenvalues which are the reciprocal of the original matrix, hence all its eigenvalues are still positive, it is symmetric, and so is also positive definite.

We can show that K is positive definite, by showing it is a sum of positive (semi)definite matrices. Clearly $\mathbb{I}/m_A$ is positive definite as is $\mathbb{I}/m_B$. We need only show $-\tilde{\mathbf{r}}_A I_A^{-1}\tilde{\mathbf{r}}_A$ is positive semidefinite, the argument for $-\tilde{\mathbf{r}}_B I_B^{-1}\tilde{\mathbf{r}}_B$ will be identical. Note that $\tilde{\mathbf{r}}_A^T = -\tilde{\mathbf{r}}_A$, hence we are equivalently trying to show $\tilde{\mathbf{r}}_A^T I_A^{-1}\tilde{\mathbf{r}}_A$ is positive semidefinite. Since inertia tensors are symmetric with strictly positive eigenvalues we know they are positive definite and hence have inverses which are also positive definite. Let us write I_A^{-1} as $R^T DR$ for some appropriate choice of orthogonal and diagonal matrices. $\mathbf{x}^T\tilde{\mathbf{r}}_A^T I_A^{-1}\tilde{\mathbf{r}}_A\mathbf{x} = (R\tilde{\mathbf{r}}_A\mathbf{x})^T D(R\tilde{\mathbf{r}}_A\mathbf{x})$. Since $R\tilde{\mathbf{r}}_A\mathbf{x}$ is simply a vector (which may be zero) and D has positive diagonal entries, we can safely say that $\tilde{\mathbf{r}}_A^T I_A^{-1}\tilde{\mathbf{r}}_A$ is positive semidefinite, and therefore K is positive definite and has an inverse which is also positive definite.

C Showing Trajectories Converge To Rays

In Section 3.11 we described the motion of the trajectory of $\mathbf{u}$ in terms of polar coordinates. Specifically we showed that $d\theta/dp_z$ and dr/dp_z, which we'll denote by θ' and r' (to simplify notation), could be written as

$$\theta' = \frac{f(\theta)}{r}, \qquad r' = g(\theta),$$

where $f(\theta)$, $g(\theta)$ are continuous bounded periodic functions with period 2π. The rays of constant sliding occur at values of θ for which $f(\theta) = 0$. Our task is to show that for trajectories not starting on a ray of constant sliding that θ converges monotonically to a root of $f(\theta)$. We can assume that as p_z increases the trajectory does not hit the origin (i.e. $r \neq 0$) otherwise the algorithm would terminate. We can also assume there is at least one root of $f(\theta)$.

Let (r_0, θ_0) be the start point of the trajectory in polar coordinates. We do not start on a ray of constant sliding so $f(\theta_0) \neq 0$. Without loss of generality we can assume $f(\theta_0) > 0$. Consequently $\theta' > 0$ and initially θ increases. Let θ_{root} be the first root of f after θ_0 (θ_{root} always exists because f is periodic with at least one root). Hence as p_z increases θ continually increases towards θ_{root} (θ cannot decrease until it passes θ_{root}). To show θ doesn't converge before it reaches θ_{root}, we will prove that θ is able to reach any $\theta_{end} \in (\theta_0, \theta_{root})$ after a finite increase in p_z.

Let r'_{max} be the largest value of r' (recall that r' is bounded so r'_{max} is finite). If $r'_{max} \leq 0$ then set $r'_{max} = 1$. Let f_{min} be the smallest value of f in the region $[\theta_0, \theta_{end}]$ (note that $f_{min} > 0$). Hence as p_z increases the following holds

$$\theta' > \frac{f_{min}}{r_0 + r'_{max} p_z}.$$

Consequently we can prove that θ reaches θ_{end} before p_z becomes

$$P = (\exp((\theta_{end} - \theta_0) r'_{max}/f_{min}) - 1)\frac{r_0}{r'_{max}}$$

because

$$\int_0^P \frac{f_{min}}{r_0 + r'_{max} p_z} dp_z = \left[\frac{f_{min}}{r'_{max}} \ln(r_0 + r'_{max} p_z)\right]_0^P = \theta_{end} - \theta_0.$$

Although not strictly necessary for the arguments given in Section 3.11 we can also show that θ never reaches θ_{root} after a finite change in p_z. If it did, then let $r_{min} > 0$ be the smallest value of r the trajectory reached. Due to the nature of f we can show that there exists a finite constant $c > 0$ and $\theta_c \in (\theta_0, \theta_{root})$ such that when $\theta \in (\theta_c, \theta_{root})$, θ' satisfies

$$\theta' < \frac{(\theta_{root} - \theta)c}{r_{min}}.$$

Consequently the value of p_z when θ reaches $\theta_{root} - \varepsilon$ (for some small $\varepsilon > 0$) must be more than

$$\int_{\theta_c}^{\theta_{root}-\varepsilon} \frac{r_{min}}{(\theta_{root} - \theta)c} d\theta = \frac{r_{min}}{c} \ln((\theta_{root} - \theta_c)/\varepsilon).$$

Hence the closer we get to θ_{root} the larger the change in p_z, in fact it tends to infinity. Therefore θ never reaches θ_{root} but converges monotonically to it.