# Physically Based Modeling
**CS 15-863 Notes      Spring 1997**
**Particle Collision and Contact**

## 1 Collisions with Springs

Suppose we wanted to implement a particle simulator with a “floor”: a solid horizontal plane which particles could bounce off of, or lie on. Assuming all the particles started out above the floor, they’d all have to stay there for the rest of the simulation. How could we go about implementing this?

Clearly, there are two problems. We need to detect when particles are near the floor, and when they are, we need to do something to stop them from passing through the floor. One obvious thing to do is to impose a vertical spring force of $-ky(t)$ whenever the $y$-coordinate $y(t)$ of the particle is less than zero. (We’ll suppose the floor is simply the line $y = 0$.) If we make $k$ large enough, than we can prevent the particle from moving too far beneath the floor. The problem is, that by making $k$ large, we set ourselves up for stiff ODE’s, as mentioned in earlier notes. Also, no matter what value of $k$ you pick, I can always choose a heavy enough particle (or a particle moving downwards with a high enough velocity) so that the upwards $-ky(t)$ spring-force isn’t strong enough to stop the particle from moving substantially below the line $y = 0$.

Of course, we could modify the spring force to completely stop the particle from moving below the floor, no matter what we throw at it. Suppose the spring force is $\frac{1}{ky(t)^2}e^{\frac{1}{ky(t)}}$. Then as $y(t)$ decreases towards zero, the spring force becomes strong quickly; in fact, it is impossible for the particle to ever attain $y(t) = 0$ given this spring force. Note that in this case we have “action at a distance”: the spring force acts even before $y(t)$ becomes negative. By making $k$ large though, we can make the force fall off for positive values of $y(t)$ quickly. Of course, this is a pretty stiff force function; $-ky(t)$ is a walk in the park in comparison...

## 2 Impulses

For both the linear springs and the exponential springs, as we increase $k$ we cause the particle’s velocity to change more quickly. If we view our floor as an impenetrable obstacle, we should just take this process to the limit, and let the particle undergo an instantaneous change in velocity.

As an example, suppose we have a particle of mass $m$ and velocity $v_0$ (a vector). Lets apply a force $F$ to the particle for a period of time $\Delta t$. We’ll assume that $F$ points directly opposite to $v_0$, and that no other forces act on the particle. The particle’s velocity $v_1$ time $\Delta t$ later is

$$v_1 = \int_0^{\Delta t} F/m\, dt + v_0 = \frac{F\Delta t}{m} + v_0$$

Suppose we want the force $F$ to completely cancel the initial velocity $v_0$ after time $\Delta t$. Setting

$v_1$ to zero and solving, we get

$$F = \frac{-mv_0}{\Delta t}$$

(again under the assumption that $F$ is directly opposite $v_0$). Clearly, to stop the particle instantaneously (that is, with $\Delta t = 0$) requires an infinite force. We can stop the particle arbitrarily quickly, by making $F$ suitably strong, but we can't stop the particle instantaneously with a force $F$.

Instead, we'll define a new quantity called an *impulse*. Lets imagine applying a force $F$ over shorter and shorter time intervals $\Delta t$. As $\Delta t$ shrinks, $F$ grows. Suppose that $F$ grows without bound as $\Delta t$ goes to zero, such that $\lim_{\Delta t \to 0} F = \infty$ in such a way that

$$\lim_{\Delta t \to 0} F \Delta t$$

converges to some finite value $J$. The quantity $J$ defined in such a way is called an *impulse*. Like force, impulse is a vector quantity. Impulse has the dimensions of force times time, or equivalently, mass times velocity. What is impulse good for?

Imagine that $\lim_{\Delta t \to 0} F \Delta t = J = -mv_0$; that is, no matter what $\Delta t$ we choose, we pick $F$ so that $F \Delta t = -mv_0$. Then as $\Delta t$ goes to zero, we have

$$v_1 = \lim_{\Delta t \to 0} \int_0^{\Delta t} F/m \, dt + v_0 = \lim_{\Delta t \to 0} \frac{F \Delta t}{m} + v_0 = J/m + v_0 = 0.$$

In the limit then, we have made $v_1$ zero instantaneously. The idea of an impulse is that when applied to a particle, it produces an instantaneous change of velocity. Unlike force, an impulse is applied only for a zero-length period of time.

If we define the momentum $P$ of a particle with mass $m$ and velocity $v$ as $P = mv$, then applying an impulse $J$ (of arbitrary direction) changes the momentum to $P + J$, and the velocity from $P/m$ to $(P + J)/m$. Note that any other finite forces (e.g. gravity, wind) are ignored when an impulse is applied, because the effects of these finite forces on the velocity go to zero as $\Delta t$ goes to zero. By defining impulses, we can abstractly consider the idea of an instantaneous change of velocity, by ignoring the continuous change of velocity that occurs over an arbitrarily small time interval $\Delta t$.

# 3 Collisions with Planes

Lets apply the concept of an impulse to let our particles bounce off an arbitrary plane. First, we need to worry about detecting collisions between a particle and a plane. Consider a particle with position $x(t)$ in space, and a plane with a unit normal $\hat{n}$. The normal $\hat{n}$ is chosen so that $\hat{n}$ points towards the legal side of the plane. (For the case of the floor, $\hat{n}$ points straight up.) Let $q_0$ be any point on the plane. Then the plane is defined as the set of points $p$ such that

$$\hat{n} \cdot (p - q_0) = 0.$$

Our particle is on the legal side of the plane as long as $\hat{n} \cdot (x(t) - q_0) > 0$. When $\hat{n} \cdot (x(t) - q_0) = 0$, the particle has collided with the plane (figure 1). Determining exactly when the particle has collided is in general a hard problem. Our approach will be as follows: whenever we find the particle is on the illegal side of the plane, we'll displace it in the $\hat{n}$ direction so its exactly on the plane. Then we'll apply an impulse to it to kick it away from the wall. The correct way to do collision detection is explained in section 3.3.

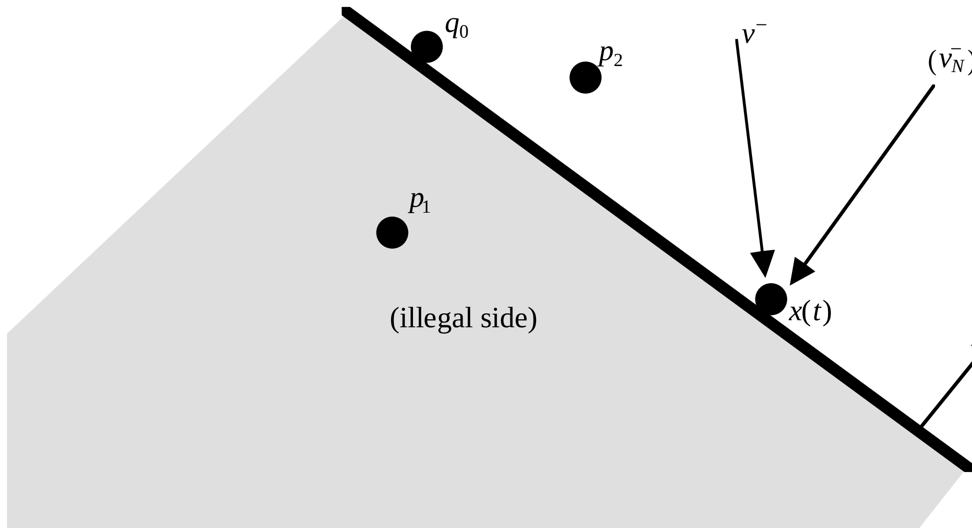


Figure 1: The vector $p_1 - q_0$ is opposite $\hat{n}$, so $(p_1 - q_0) \cdot \hat{n} < 0$ and $p_1$ is on the illegal side of the plane. Similarly, $(p_2 - q_0) \cdot \hat{n} > 0$.

## 3.1 Collision Response

Our particle is in contact with the plane. What impulse $J$ should we apply to the particle to kick it off the wall? First, some notation. Lets let the initial velocity, before application of $J$, be $v^-$. Lets let the velocity after application of $J$ be $v^+$. Define the *normal speed* $v_N^-$ before the collision (figure 1) by

$$v_N^- = \hat{n} \cdot v^-$$

and the *normal speed* $v_N^-$ after the collision by

$$v_N^+ = \hat{n} \cdot v^+.$$

Note that $v^-$ and $v^+$ are vectors while $v_N^-$ and $v_N^+$ are scalars.

To determine $J$, we use an empirical law for collisions. The law states that a collision can be described in terms of a *coefficient of restitution* denoted by $\epsilon$, where $0 \leq \epsilon \leq 1$. The law relates the initial normal speed to the final normal speed by

$$v_N^+ = -\epsilon v_N^-. \tag{1}$$

If $\epsilon = 1$, then the normal speed is completely reversed, and we have in effect a "superball." If $\epsilon = 0$, the collision is completely "dead," in that the particle won't bounce away from the wall.

We can use this law to determine $J$ as follows. Assume that there is no friction between the particle and the wall. Then the direction of $J$ is parallel to $\hat{n}$, and should be in the same direction.

That is, we can write $J = j\hat{n}$ for some scalar $j$. Our intuition is that $j$ should be a positive number. To compute $j$, we express $v_N^+$ in terms of $j$ and use equation (1). We have

$$v^+ = v^- + \frac{J}{m} = v^- + \frac{j\hat{n}}{m}$$

where $m$ is the particle's mass. Then

$$\begin{aligned} v_N^+ &= \hat{n} \cdot v^+ \\ &= \hat{n} \cdot v^- + j\hat{n} \cdot \frac{\hat{n}}{m} \\ &= v_N^- + \frac{j}{m} \end{aligned}$$

since $\hat{n} \cdot \hat{n} = 1$. Using equation (1), we get

$$v_N^+ = v_N^- + \frac{j}{m} = -\epsilon v_N^-$$

which we can solve to obtain

$$j = -(1+\epsilon) m v_N^-.$$

Note that since we presumably started with $v_N^-$ being negative (since the particle had velocity partly opposite $\hat{n}$) we get $j > 0$ as expected.

If we want to add a frictional component, we can use a (simplified) version of Coulomb's friction law. For any impulse of strength $j$ in the $\hat{n}$ direction, there will also be an impulse tangent to the plane of strength $\mu j$, where $\mu$ is the coefficient of friction. Reasonable values for $\mu$ would be in the range zero to perhaps $\frac{1}{3}$. The direction of this tangential impulse is directly opposite the component of $v^-$ in the plane. The component of $v^-$ tangent to the plane is

$$v^- - v_N^- \hat{n}.$$

Thus, to add friction, let the total impulse be

$$j\hat{n} - \mu j \frac{v^- - v_N^- \hat{n}}{|v^- - v_N^- \hat{n}|}.$$

(Don't try this if $v^-$ has no tangential component.)

## 3.2 Particle/Plane Contact

Suppose that we set $\epsilon$ to zero, so that after the collision, $v_N^+$ was zero. Then the particle would be resting on the plane. In this case, we will have to apply a force to stop the particle from being pushed through the plane.

Suppose that the net force acting on the particle is $F_{ext}$. The force $F_{ext}$ would include gravity forces, drag from velocity, and any inter-particle forces such as springs. We need to stop the particle from accelerating opposite $\hat{n}$. To do this, we add a contact force $F_c$ that acts on the particle. Again, lets start with the frictionless case, where $F_c$ acts in the $\hat{n}$ direction. So we can write $F_c = f_c\hat{n}$ for some scalar $f_c$.

To constrain the particle's acceleration, we can write

$$\hat{n} \cdot \ddot{x}(t) = 0 \tag{2}$$

which prevents the particle from accelerating in the $\hat{n}$ direction. The total force on the particle is $F_{ext} + F_c$, so

$$\ddot{x}(t) = \frac{F_{ext} + F_c}{m} = \frac{F_{ext} + f_c\hat{n}}{m}.$$

Substituting into equation (2) yields

$$\hat{n} \cdot \frac{F_{ext} + f_c\hat{n}}{m} = \frac{\hat{n} \cdot F_{ext} + f_c}{m} = 0$$

so that

$$f_c = -\hat{n} \cdot F_{ext}. \tag{3}$$

Now, if $F_{ext}$ is opposite $\hat{n}$, then $f_c$ is positive. That is, we require an outwards force to prevent the particle from being pushed opposite $\hat{n}$. However, if $F_{ext}$ was along $\hat{n}$, lifting the particle off the plane in the positive $\hat{n}$ direction, $f_c$ would have to be negative to stop the particle from leaving the surface. If you want your planes to be sticky, you solve for $f_c$ using equation (3), and use whatever answer you get. But if you want the particle to release from the surface, you will want to set $f_c$ to zero if equation (3) yields a negative value of $f_c$.

Finally, if you want to add friction, its about the same as before (using a simplified model of friction that is). If the particle is sliding along the plane, you add in a tangential force that is directly opposite to the velocity, with a magnitude of $\mu f_c$. (However, if the particle has no sliding velocity, things are a bit trickier. In this case, what you want to do is to compute a tangential force such that the component of $\ddot{x}(t)$ in the plane is zero. The strength of that force should not exceed $\mu f_c$; if it does, you should chop the magnitude down to be $\mu f_c$. Clear?)

## 3.3 Collision Detection

In general, computing exactly when a particle collides with a plane is difficult. Here we'll explain the right way to implement collision detection, though the displacement method will do for this course.

Again, imagine a particle dropping towards the floor. Suppose we consider the particle at times $t_0$, $t_0 + \Delta t$, $t_0 + 2\Delta t$ etc. and suppose the time of collision, $t_c$, at which the particle actually strikes the floor, lies between $t_0$ and $t_0 + \Delta t$. Ideally, we'd like to run our simulator up to time $t_c$, change the velocity of the particle (to make it bounce off the floor), and then restart the simulator. If you're using an ODE method other than Euler's method, this is essential because the ODE solver doesn't realize that the motion equations are discontinuous at $t_c$. (Yes, discontinuous—the particle's position is continuous over time, but its velocity is not at $t_c$, and the velocity is a variable of the ODE.)

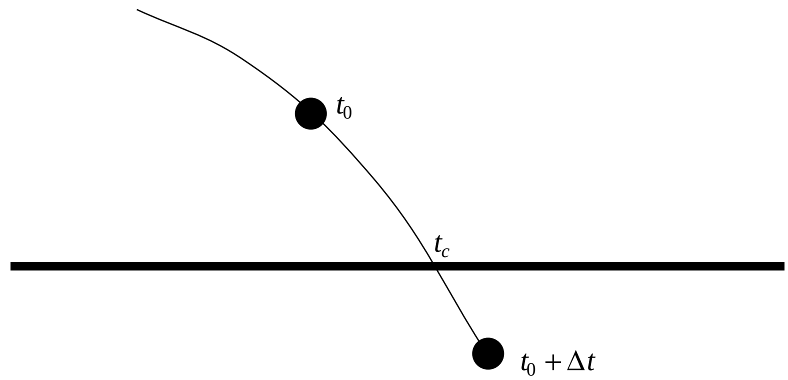


Figure 2: At time $t_0 + \Delta t$, the particle is found to lie below the floor. Thus, the actual time of collision $t_c$ lies between the time of the last known legal position, $t_0$, and $t_0 + \Delta t$.

So in terms of ODE solution, we view this as solving up to time $t_c$, and then restarting at time $t_c$ with a new initial velocity.

The big problem of course is finding $t_c$. At time $t_0$, we find that the particle lies above the floor, but at the next time step, $t_0 + \Delta t$, we find the particle is beneath the floor, which means that inter-penetration has occurred.

If we're going to stop and restart the simulator at time $t_c$, we'll need to compute $t_c$. All we know so far is that $t_c$ lies between $t_0$ and $t_0 + \Delta t$. In general, solving for $t_c$ exactly is difficult, so we solve for $t_c$ numerically, to within a certain tolerance. A simple way of determining $t_c$ is to use a numerical method called *bisection*. If at time $t_0 + \Delta t$ we detect inter-penetration, we inform the ODE solver that we wish to restart back at time $t_0$, and simulate forward to time $t_0 + \Delta t/2$. If the simulator reaches $t_0 + \Delta t/2$ without encountering inter-penetration, we know the collision time $t_c$ lies between $t_0 + \Delta t/2$ and $t_0 + \Delta t$. Otherwise, $t_c$ is less than $t_0 + \Delta t/2$, and we try to simulate from $t_0$ to $t_0 + \Delta t/4$. Eventually, the time of collision $t_c$ is computed to within some suitable numerical tolerance. The accuracy with which $t_c$ is found depends on the collision detection routines. The collision detection routines have some parameter $\epsilon$. We decide that our computation of $t_c$ is "good enough" when the particle inter-penetrates the floor by no more than $\epsilon$, and is less than $\epsilon$ above the floor. At this point we declare that the particle is in contact with the floor (figure 3).

How to actually implement all of this depends on how you interact with your ODE routines. One might use exception handling code to signal the ODE of various events (collisions, interpenetration), or pass some sort of messages to the ODE solver.

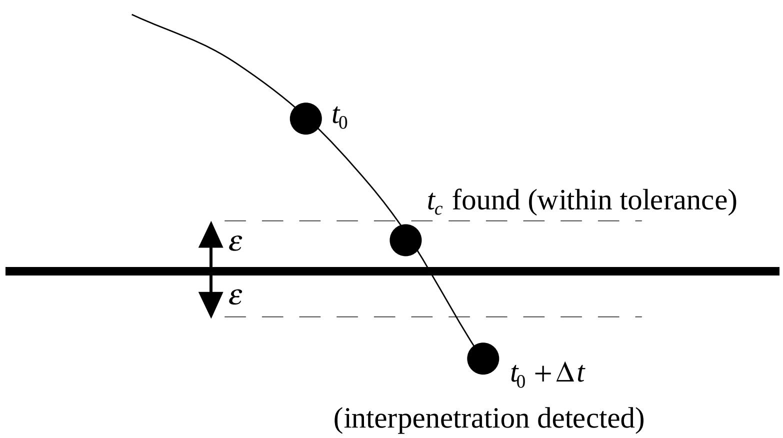


Figure 3: When the particle is found to be within some tolerance $\epsilon$ of contacting the floor, then $t_c$ is considered to have been computed to within sufficient accuracy.