3/25 Lecture

Rigid Body Dynamics

Recall that $x(\vec{t}) \rightarrow$position, $v(\vec{t}) \rightarrow$velocity, and $m \rightarrow$ mass.

Linear momentum(P(t) = m$v(t)$
We know that $x'(t) = v(t)$ and

$F = ma = mv'(t) = \frac{d}{d(t)}P(t) = F$.
These two equations describe linear motion.

Other information we know:
$\omega(t) \rightarrow$ angular velocity of body (3x1 vector)
q(t)$\rightarrow$ orientation of body (unit quaternion, 4x1 vector)
I(t)$\rightarrow$ inertia tensor (3x3 matrix). Our inertia tensor has a form for world space and for object space.

In fact, $I_{world}(t) = R(t)I_{object}R(t)^T$ where R(t) is our rotation matrix.

Angular momentum L(t) = I(t)$\omega(t) \rightarrow \frac{d}{d(t)}L(t) = L'(t) = \tau$ (**Applied torque**).

Knowing all of these equations is well and good, however we need a way to be able to compute or approximate them. To do so, we need a system to discretize time. Recall that we accomplished this using our forward and backward Euler equations.

We came up with:
P'(t) = F and $\frac{P^{n+1}-P^n}{\Delta t} = F \implies P^{n+1} = P^n + F\Delta t$. (Forward Euler)
$v^{n+1} = \frac{P^{n+1}}{m}$ and $x' = v \rightarrow \frac{x^{n+1}-x^n}{\Delta t} = v^{n+1} \implies x^{n+1} = x^n + \Delta t v^{n+1}$. (Backward Euler)

Finally, we also have for our angular momentum $L'(t) = \tau$ the following

(Forward Euler) $\frac{L^{n+1}-L^n}{\Delta t} = \tau \implies L^{n+1} = L^n + \Delta t\tau$

Recall that L = I_w and $w^{n+1} = I^{-1}L^{n+1}$ where I^{-1} is the world inertia and L^{n+1} is the space tensor.
Note: $I^{-1} = R(t)I_{obj}^{-1}R(t)^T$, and I_{obj}^{-1} never changes.

Lastly, how do we compute q^{n+1}(4x1 matrix)?
We know that $q'(t) = 1/2\omega(t)q(t)$ is equivalent to $R'(t) = \omega(t)^*R(t)$ where $\omega(t)^*$ is the conjugate of $\omega(t)$.

First, we define $\vec{\omega} =$

$$\begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix}$$

Then, $\omega^* =$

$$\begin{bmatrix} 0 & -w_3 & w_2 \\ w_3 & 0 & -w_1 \\ -w_3 & w_1 & 0 \end{bmatrix}$$

Notice that ω^* is skew symmetric.

Aside: when we write $\vec{a}x\vec{b} = \vec{c}$, it is somewhat more intuitive in a linear algebra perspective to think of this as $[\vec{a}x][\vec{b}] = \vec{c} \iff \omega^* R(t) = R'(t)$.

And so we have, $R'(t) = \omega^* R(t) \implies \frac{R^{n+1} - R^n}{\Delta t} = \omega^{*n+1} R^n \implies R^{n+1} = R^n + \Delta t \omega^{*n+1} R^n$.

Additionally, $R^{n+1} \iff q^{n+1}$ (unit quaternion). However, note that because q^{n+1} is a unit quaternion, then this means that $R^T R = I$ and R^n is orthonormal.

Hence, before we can move from q^{n+1} to R^{n+1}, we have to orthonormalize R^{n+1} first.

Rigid Body Collision

For collision simulation, we have already existing engines at our disposal: Bullet Physics (pybullet.org) and Open Dynamics Engine (ode.org).

There are two types of collisions we care about.

Resting contact/persistent collision: collision which occurs between an object and its surface (think: gravity pushing down on you and with equal force pushing you back up).
Separating (non-persistent) collision: collisions between objects in motion.