Joshua Kao
Math 164
Professor Yong
11 May 2008

Final Project Report

My goal was to simulate the motion of a football, particularly when it bounces off the ground. To do so, I first calculated the equations of motion for torque free motion. To find these, we begin with Euler's equations for torque free motion

$$I_x\dot{\omega}_x + (I_z - I_y)\omega_y\omega_z = 0,$$
$$I_y\dot{\omega}_y + (I_x - I_z)\omega_x\omega_z = 0,$$
$$I_z\dot{\omega}_z + (I_y - I_x)\omega_x\omega_y = 0,$$

where the I are the moments of inertia about the corresponding axes and the ω are the angular velocities in the rotating frame. First, position the football so that the angular momentum vector of the ball is aligned with the z-axis and the center of the ball is on the origin. We then solve Euler's equations using the cylindrical symmetry of the football

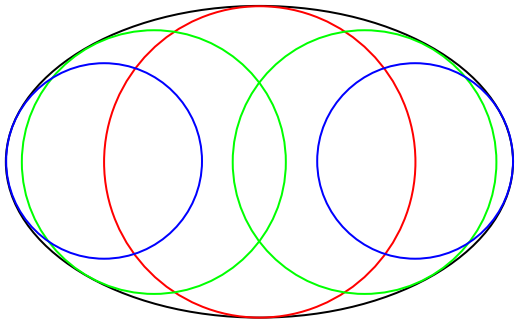
$$I_x = I_y \equiv I_{xy}$$

which greatly reduces the third equation above. Then we change frames to using the Euler angles φ, θ, and ψ using the following relations

$$L_x = I_{xy}\omega_x = L\sin\theta\sin\psi, \quad \omega_x = \dot{\phi}\sin\theta\sin\psi + \dot{\theta}\cos\psi,$$
$$L_y = I_{xy}\omega_y = L\sin\theta\cos\psi, \quad \omega_y = \dot{\phi}\sin\theta\cos\psi - \dot{\theta}\sin\psi,$$
$$L_z = I_z\omega_z = L\cos\theta. \quad \omega_z = \dot{\phi}\cos\theta + \dot{\psi}.$$

The Euler equations rotate the angular momentum vector relative to a fixed coordinate system. After solving these equations we end up with three equations for the time derivative of the Euler angles. This procedure to solve the equations can be found in the Mathematica notebook. Also refer to the code to see which Euler angle system is being used. Now $\varphi(t)$, $\theta(t)$, and $\psi(t)$ are solved so we can rotate the ball over time in a torque free environment. All of the parameters and equations were drawn from Peter J. Brancazio's paper *Rigid-boyd dynamics of a football*. In the MATLAB code I have also included a way to rotate the angular momentum vector to some initial state so that the precession can occur in different orientations.

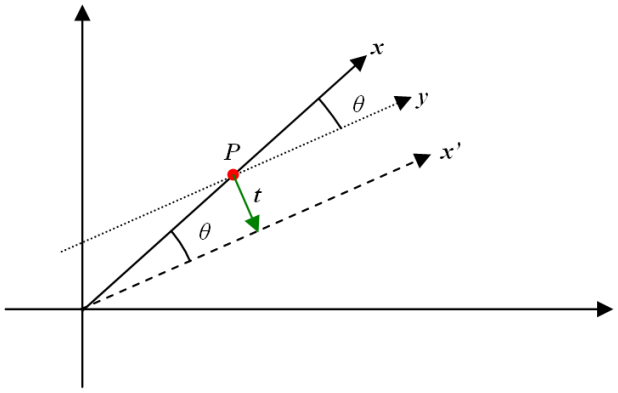
The next step is to try and describe the behavior of collisions. First, we need a way to detect the collisions. The most general solution is probably to find out when a surface describing the ball intersects with a plane representing the ground. One could conceivable come up with an equation for an ellipsoid and then at every time step, find the minimum z value of the ellipsoid and see if it is at or close to zero. I found that this was too complicated given the time-constraints and could also be computationally expensive. So I approximated an ellipsoid using four intersecting spheres seen here. They are much easier to work with both conceptually and computationally.



Now that we have established how to detect collisions we now need to be able to describe the motion of the ball when it collides. After extensive research, I found several problems in using the Euler angles to solve for the motion of the ball. Although Euler's equations with torque can be solved just fine numerically, when turning that into motion using Euler angles there is the problem of gimbal lock, which is to say, there are singularities in the equation when transitioning between complete rotations, such as from 0° to 360°. One solution to this is to switch between different Euler angle representations when gimbal lock occurs, which I considered too difficult to implement given the time constraints. The second solution is to use quaternations to describe the rotations. However, I am unfamiliar with quaternations and there was also a lack of time to learn the math.

Thus, the motion described in the script is completely arbitrary. However, it does have inklings of some of the physics I wished to describe. The first is when the ball collides, there is an impulse given by the normal force of the ground on the ball. We split that normal force into two components, one directed toward the ball's center of gravity and the other perpendicular to that axis. The force toward the center of gravity moves the ball like a point mass, while the force that is perpendicular imparts a torque. We then decide a coefficient of restitution that describes how much energy is lost thermally.

The second aspect of the collision physics is rolling of the ball. This is not in the current implementation of the script. In order to implement rolling, I need to fall back on the sphere segment representation of the ellipsoid. This is because I still do not have an expression describing the surface. Also, rolling spheres are easy to model and the script would only need to determine which sphere was currently in contact with the ground. The only concern is the issue of calculating the moment of inertia to describe the rolling motion. Either I could condense the rest of the ball onto several point masses on the surface of the sphere or use a parallel axis theorem of some sort to approximate the moment of inertia of the ball assuming the spherical rolling was not too different from ellipsoidal rolling. Another issue with the rolling motion is the matter of maintaining the Euler angle representation while rotating the ellipsoid about an origin that was not the center of gravity. The only solution I have determined thus far is to still rotate about the center of gravity but also add a compensating translation, which is mathematically equivalent. As seen in the diagram below, rotating $\boldsymbol{x}$ about the origin to make $\boldsymbol{x'}$ is the same as rotating $\boldsymbol{x}$ about $\boldsymbol{P}$ to make $\boldsymbol{y}$ and then adding a translation $\boldsymbol{t}$.



In conclusion: 3D is hard.

Sources:

Brancazio, Peter J. "Rigid-body dynamics of a football". Am. J. Phys. Vol 55 No 5. May 1987.

"Euler Angles – from Wolfram Mathworld".
http://mathworld.wolfram.com/EulerAngles.html. Date Accessed: May 8, 2008.