Rigid Body Dynamics 2

CSE169: Computer Animation

Instructor: Steve Rotenberg

UCSD, Spring 2016

Cross Product & Hat Operator

$$\mathbf{a} \times \mathbf{b} = [a_y b_z - a_z b_y \quad a_z b_x - a_x b_z \quad a_x b_y - a_y b_z]$$

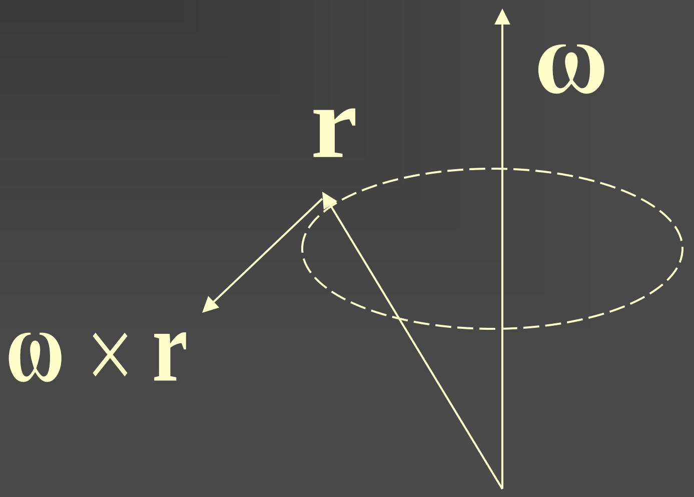
$$\mathbf{a} \times \mathbf{b} = \hat{\mathbf{a}} \cdot \mathbf{b}$$

$$\hat{\mathbf{a}} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix}$$

Derivative of a Rotating Vector

- Let's say that vector **r** is rotating around the origin, maintaining a fixed distance
- At any instant, it has an angular velocity of **ω**

$$\frac{d\mathbf{r}}{dt} = \boldsymbol{\omega} \times \mathbf{r}$$



Product Rule

- The product rule of differential calculus can be extended to vector and matrix products as well

$$\frac{d(\mathbf{a}\cdot\mathbf{b})}{dt}=\frac{d\mathbf{a}}{dt}\cdot\mathbf{b}+\mathbf{a}\cdot\frac{d\mathbf{b}}{dt}$$

$$\frac{d(\mathbf{a}\times\mathbf{b})}{dt}=\frac{d\mathbf{a}}{dt}\times\mathbf{b}+\mathbf{a}\times\frac{d\mathbf{b}}{dt}$$

$$\frac{d(\mathbf{A}\cdot\mathbf{B})}{dt}=\frac{d\mathbf{A}}{dt}\cdot\mathbf{B}+\mathbf{A}\cdot\frac{d\mathbf{B}}{dt}$$

Rigid Bodies

- We treat a rigid body as a system of particles, where the distance between any two particles is fixed
- We will assume that internal forces are generated to hold the relative positions fixed. These internal forces are all balanced out with Newton’s third law, so that they all cancel out and have no effect on the total momentum or angular momentum
- The rigid body can actually have an infinite number of particles, spread out over a finite volume
- Instead of mass being concentrated at discrete points, we will consider the density as being variable over the volume

Rigid Body Mass

- With a system of particles, we defined the total mass as:

$$m = \sum_{i=1}^{n} m_i$$

- For a rigid body, we will define it as the integral of the density ρ over some volumetric domain Ω

$$m = \int_{\Omega} \rho \, d\Omega$$

Angular Momentum

- The linear momentum of a particle is $\mathbf{p} = m\mathbf{v}$
- We define the moment of momentum (or angular momentum) of a particle at some offset $\mathbf{r}$ as the vector $\mathbf{L} = \mathbf{r} \times \mathbf{p}$
- Like linear momentum, angular momentum is conserved in a mechanical system
- If the particle is constrained only to rotate so that the direction of $\mathbf{r}$ is changing but the length is not, we can re-express its velocity as a function of angular velocity $\boldsymbol{\omega}$:

$$\mathbf{v} = \boldsymbol{\omega} \times \mathbf{r}$$

- This allows us to re-express **L** as a function of $\boldsymbol{\omega}$:

$$\mathbf{L} = \mathbf{r} \times \mathbf{p} = \mathbf{r} \times (m\mathbf{v}) = m\mathbf{r} \times \mathbf{v} = m\mathbf{r} \times (\boldsymbol{\omega} \times \mathbf{r})$$
$$\mathbf{L} = -m\mathbf{r} \times (\mathbf{r} \times \boldsymbol{\omega})$$
$$\mathbf{L} = -m\hat{\mathbf{r}} \cdot \hat{\mathbf{r}} \cdot \boldsymbol{\omega}$$

Rotational Inertia

$$\mathbf{L} = -m\hat{\mathbf{r}} \cdot \hat{\mathbf{r}} \cdot \boldsymbol{\omega}$$

- We can re-write this as:

$$\mathbf{L} = \mathbf{I} \cdot \boldsymbol{\omega} \quad \textit{where} \quad \mathbf{I} = -m\hat{\mathbf{r}} \cdot \hat{\mathbf{r}}$$

- We've introduced the rotational inertia matrix **I**, which relates the angular momentum of a rotating particle to its angular velocity

Rotational Inertia of a Particle

$$\mathbf{I} = -m\hat{\mathbf{r}} \cdot \hat{\mathbf{r}}$$

$$\mathbf{I} = -m\begin{bmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{bmatrix}$$

$$\mathbf{I} = -m\begin{bmatrix} -r_y^2 - r_z^2 & r_x r_y & r_x r_z \\ r_x r_y & -r_x^2 - r_z^2 & r_y r_z \\ r_x r_z & r_y r_z & -r_x^2 - r_y^2 \end{bmatrix}$$

Rotational Inertia of a Particle

$$\mathbf{I} = \begin{bmatrix} m\left(r_y^2 + r_z^2\right) & -mr_xr_y & -mr_xr_z \\ -mr_xr_y & m\left(r_x^2 + r_z^2\right) & -mr_yr_z \\ -mr_xr_z & -mr_yr_z & m\left(r_x^2 + r_y^2\right) \end{bmatrix}$$

$$\mathbf{L} = \mathbf{I} \cdot \boldsymbol{\omega}$$

Rotational Inertia of a Rigid Body

- For a rigid body, we replace the single mass and position of the particle with an integration over all of the points of the rigid body times the density at that point

Rigid Body Rotational Inertia

$$\mathbf{I} = \begin{bmatrix} \int \rho\left(r_y^2 + r_z^2\right)d\Omega & -\int \rho r_x r_y d\Omega & -\int \rho r_x r_z d\Omega \\ -\int \rho r_x r_y d\Omega & \int \rho\left(r_x^2 + r_z^2\right)d\Omega & -\int \rho r_y r_z d\Omega \\ -\int \rho r_x r_z d\Omega & -\int \rho r_y r_z d\Omega & \int \rho\left(r_x^2 + r_y^2\right)d\Omega \end{bmatrix}$$

$$\mathbf{I} = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{xy} & I_{yy} & I_{yz} \\ I_{xz} & I_{yz} & I_{zz} \end{bmatrix}$$

Rotational Inertia

- The rotational inertia matrix $\mathbf{I}$ is a 3x3 symmetric matrix that is essentially the rotational equivalent of mass
- It relates the angular momentum of a system to its angular velocity by the equation

$$\mathbf{L} = \mathbf{I} \cdot \boldsymbol{\omega}$$

- This is similar to how mass relates linear momentum to linear velocity, but rotation adds additional complexity

$$\mathbf{p} = m\mathbf{v}$$

Rotational Inertia

- The center of mass of a rigid body behaves like a particle- it has position, velocity, momentum, etc., and it responds to forces through **f**=*m***a**
- Rigid bodies also add properties of rotation. These behave in a similar fashion to the translational properties, but the main difference is in the velocity-momentum relationships:

$$\mathbf{p} = m\mathbf{v} \quad vs. \quad \mathbf{L} = \mathbf{I}\boldsymbol{\omega}$$

- We have a vector **p** for linear momentum and vector **L** for angular momentum
- We also have a vector **v** for linear velocity and vector $\boldsymbol{\omega}$ for angular velocity
- In the linear case, the velocity and momentum are related by a single scalar *m*, but in the angular case, they are related by a matrix $\mathbf{I}$
- This means that linear velocity and linear momentum always line up, but angular velocity and angular momentum don't
- Also, as $\mathbf{I}$ itself changes as the object rotates, the relationship between $\boldsymbol{\omega}$ and **L** changes
- This means that a constant angular momentum may result in a non-constant angular velocity, thus resulting in the tumbling motion of rigid bodies

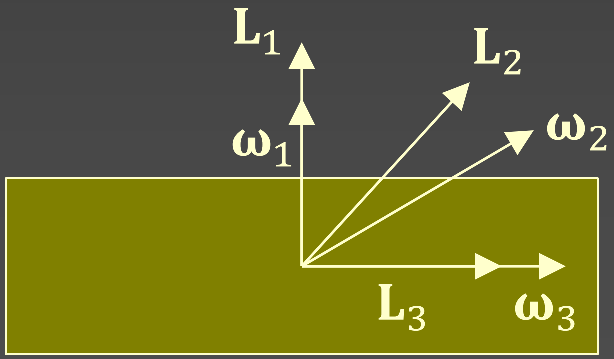
Rotational Inertia

$$\mathbf{L} = \mathbf{I}\boldsymbol{\omega}$$

- Remember eigenvalue equations of the form **Ax**=*b***x** where given a matrix **A**, we want to know if there are any vectors **x** that when transformed by **A** result in a scaled version of the x (i.e., are there vectors who's direction doesn't change after being transformed?)
- A symmetric 3x3 matrix (like $\mathbf{I}$) has 3 real eigenvalues and 3 orthonormal eigenvectors
- If the angular momentum **L** lines up with one of the eigenvectors of $\mathbf{I}$, then $\boldsymbol{\omega}$ will line up with **L** and the angular velocity will be constant
- Otherwise, the angular velocity will be non-constant and we will get tumbling motion
- We call these eigenvectors the *principal axes* of the rigid body and they are constant relative to the geometry of the rigid body
- Usually, we want to align these to the x, y, and z axes when we initialize the rigid body. That way, we can represent the rotational inertia as 3 constants (which happen to be the 3 eigenvalues of $\mathbf{I}$)

Principal Axes

- We see three example angular momentum vectors $\mathbf{L}$ and their corresponding angular velocities $\boldsymbol{\omega}$, all based on the same rotational inertial matrix $\mathbf{I}$
- We can see that $\mathbf{L}_1$ and $\mathbf{L}_3$ must be aligned with the principal axes, as they result in angular velocities in the same direction as the angular momentum



Principal Axes & Inertias

- If we diagonalize the **I** matrix, we get an orientation matrix **A** and a constant diagonal matrix $\mathbf{I}_o$
- The matrix **A** rotates the object from an orientation where the principal axes line up with the *x*, *y*, and *z* axes
- The three values in $\mathbf{I}_o$, (namely $\mathbf{I}_x$, $\mathbf{I}_y$, and $\mathbf{I}_z$) are the principal inertias. They represent the resistance to torque around the corresponding principal axis (in a similar way that mass represents the resistance to force)

Diagonalization of Rotational Inertial

$$\mathbf{I} = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{xy} & I_{yy} & I_{yz} \\ I_{xz} & I_{yz} & I_{zz} \end{bmatrix}$$

$$\mathbf{I} = \mathbf{A} \cdot \mathbf{I}_0 \cdot \mathbf{A}^T \qquad where \qquad \mathbf{I}_0 = \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix}$$

Particle Dynamics

$\mathbf{x}$	Position
$\mathbf{v} = \dfrac{d\mathbf{x}}{dt}$	Velocity
$\mathbf{a} = \dfrac{d\mathbf{v}}{dt} = \dfrac{d^2\mathbf{x}}{dt^2}$	Acceleration
m	Mass
$\mathbf{p} = m\mathbf{v}$	Momentum
$\mathbf{f} = \dfrac{d\mathbf{p}}{dt} = m\mathbf{a}$	Force

Rigid Body Dynamics

$\mathbf{A}$	Orientation (3x3 matrix)
$\boldsymbol{\omega}$	Angular Velocity (vector)
$\bar{\boldsymbol{\omega}} = \frac{d\boldsymbol{\omega}}{dt}$	Angular Acceleration (vector)
$\mathbf{I} = \mathbf{A} \cdot \mathbf{I}_0 \cdot \mathbf{A}^T$	Rotational Inertia (3x3 matrix)
$\mathbf{L} = \mathbf{I}\boldsymbol{\omega}$	Momentum (vector)
$\boldsymbol{\tau} = \frac{d\mathbf{L}}{dt} = \mathbf{r} \times \mathbf{f} = \boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega} + \mathbf{I}\bar{\boldsymbol{\omega}}$	Torque (vector)

Newton-Euler Equations

$$\mathbf{f} = m\mathbf{a}$$

$$\boldsymbol{\tau} = \boldsymbol{\omega} \times \mathbf{I} \cdot \boldsymbol{\omega} + \mathbf{I} \cdot \overline{\boldsymbol{\omega}}$$

Torque-Free Motion

- We can solve the Newton-Euler equations for the acceleration terms:

$$\mathbf{a} = \frac{1}{m}\mathbf{f}$$

$$\bar{\boldsymbol{\omega}} = \mathbf{I}^{-1} \cdot (\boldsymbol{\tau} - \boldsymbol{\omega} \times \mathbf{I} \cdot \boldsymbol{\omega})$$

- We see that acceleration **a** will be 0 if there is no force **f**
- However, if there is no torque **τ**, there may still be some angular acceleration:

$$\bar{\boldsymbol{\omega}} = -\mathbf{I}^{-1} \cdot (\boldsymbol{\omega} \times \mathbf{I} \cdot \boldsymbol{\omega})$$

- We call this *torque-free motion* and this is responsible for tumbling motion we see in rigid bodies

Rigid Body Simulation

- Each frame, we can apply several forces to the rigid body, that sum up to one total force and one total torque

$$\mathbf{f} = \sum \mathbf{f}_i \qquad \boldsymbol{\tau} = \sum \mathbf{r}_i \times \mathbf{f}_i$$

- We can then integrate the force and torque over the time step to get the new linear and angular momenta

$$\mathbf{p}' = \mathbf{p} + \mathbf{f}\Delta t \qquad \mathbf{L}' = \mathbf{L} + \boldsymbol{\tau}\Delta t$$

- We can then compute the linear and angular velocities from those:

$$\mathbf{v} = \frac{1}{m}\mathbf{p}' \qquad \boldsymbol{\omega} = \mathbf{I}^{-1}\mathbf{L}'$$

- We can now integrate the new position and orientation:

$$\mathbf{x}' = \mathbf{x} + \mathbf{v}\Delta t \qquad \mathbf{A}' = \mathbf{A} \cdot Rotate(\boldsymbol{\omega}\Delta t)$$

Rigid Body Simulation

- At some point, we need to compute $\mathbf{I}^{-1}$ where $\mathbf{I} = \mathbf{A} \cdot \mathbf{I}_0 \cdot \mathbf{A}^T$

- Note the identity $(\boldsymbol{S} \cdot \mathbf{T})^{-1} = \mathbf{T}^{-1} \cdot \boldsymbol{S}^{-1}$
- Likewise $(\boldsymbol{S}\mathbf{TU})^{-1} = \mathbf{U}^{-1}\mathbf{T}^{-1}\boldsymbol{S}^{-1}$
- Also, as $\mathbf{A}$ is orthonormal, $\mathbf{A}^{-1} = \mathbf{A}^T$

- Therefore $\mathbf{I}^{-\mathbf{1}} = (\mathbf{A} \cdot \mathbf{I}_0 \cdot \mathbf{A}^T)^{-1} = \mathbf{A} \cdot \mathbf{I}_0{}^{-1} \cdot \mathbf{A}^T$
- As $\mathbf{I}_0$ is diagonal, $\mathbf{I}_0{}^{-1}$ is easy to pre-compute

Rigid Body Simulation

```
RigidBody {
    void Update(float time);
    void ApplyForce(Vector3 &f,Vector3 &pos);
private:
    // constants
    float Mass;
    Vector3 RotInertia;    // Ix, Iy, & Iz from diagonal inertia

    // variables
    Matrix34 Mtx;          // contains position & orientation
    Vector3 Momentum,AngMomentum;

    // accumulators
    Vector3 Force,Torque;
};
```

Rigid Body Simulation

```
RigidBody::ApplyForce(Vector3 &f,Vector3 &pos) {
    Force += f;
    Torque += (pos-Mtx.d) x f
}
```

Rigid Body Simulation

```
RigidBody::Update(float time) {
    // Update position
    Momentum += Force * time;
    Mtx.d += (Momentum/Mass) * time;        // Mtx.d = position

    // Update orientation
    AngMomentum += Torque * time;
    Matrix33 I = Mtx·I_0·Mtx^T// A·I_0·A^T
    Vector3 ω = I^-1·L
    float angle = |ω| * time; // magnitude of ω
    Vector3 axis = ω;
    axis.Normalize();
    Mtx.RotateUnitAxis(axis,angle);

    // Zero out force & torque
    Force=Torque=Vector3(0,0,0);
}
```

Rigid Body Set-Up

- To define a rigid body from a physics point of view, we need only 4 constants: its mass *m*, and its principal rotational inertias I_x, I_y, and I_z
- For collision detection and rendering, we will also want some type of geometry- and we can calculate the inertia properties from this
- We expect that the geometry for the rigid body is positioned such that the center of mass lies at the origin and that the principal axes line up with x, y, and z
- One way to do this is to use simple shapes like spheres and boxes. We can use simple formulas to calculate *m*, I_x, I_y, and I_z from the dimensions and density (see last lecture for some of these)
- Alternately, we can use a triangle mesh as input and calculate the inertia properties from that

Mirtich-Eberly Algorithm

- In 1996, Brian Mirtich published an algorithm for analytically calculating the inertia properties of a polygonal mesh and in 2002, David Eberly streamlined the algorithm specifically for triangle meshes
- The resulting algorithm loops through each triangle, makes several relatively simple calculations per triangle, and ultimately ends up with exact values for the total volume, center of mass, and all 6 rotational inertia integrals
- We could conceivably input any mesh, calculate the properties, and then re-center the mesh to move the center of mass to the origin, and then diagonalize the rotational inertia matrix and re-rotate the mesh by the resulting rotation to align the principal axes with x, y, and z

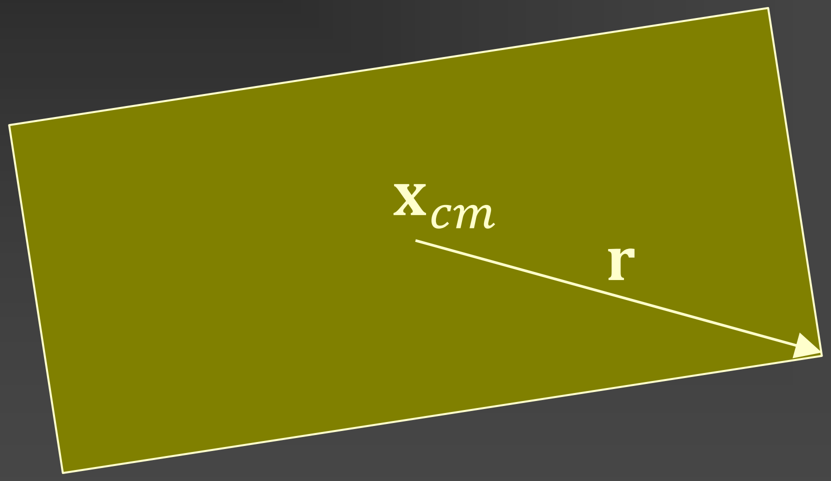
Kinematics of Offset Points

Offset Position

- Let’s say we have a point on a rigid body
- If **r** is the world space offset of the point relative to the center of mass of the rigid body, then the position **x** of the point in world space is:

$$\mathbf{x} = \mathbf{x}_{cm} + \mathbf{r}$$

Offset Position



$$\mathbf{x} = \mathbf{x}_{cm} + \mathbf{r}$$

Offset Velocity

- The velocity of the offset point is just the derivative of its position

$$\mathbf{x} = \mathbf{x}_{cm} + \mathbf{r}$$

$$\mathbf{v} = \frac{d\mathbf{x}}{dt} = \frac{d\mathbf{x}_{cm}}{dt} + \frac{d\mathbf{r}}{dt}$$

$$\mathbf{v} = \mathbf{v}_{cm} + \boldsymbol{\omega} \times \mathbf{r}$$

Offset Acceleration

- The offset acceleration is the derivative of the offset velocity

$$\mathbf{v} = \mathbf{v}_{cm} + \boldsymbol{\omega} \times \mathbf{r}$$

$$\mathbf{a} = \frac{d\mathbf{v}}{dt} = \frac{d\mathbf{v}_{cm}}{dt} + \frac{d\boldsymbol{\omega}}{dt} \times \mathbf{r} + \boldsymbol{\omega} \times \frac{d\mathbf{r}}{dt}$$

$$\mathbf{a} = \mathbf{a}_{cm} + \overline{\boldsymbol{\omega}} \times \mathbf{r} + \boldsymbol{\omega} \times \left(\boldsymbol{\omega} \times \mathbf{r}\right)$$

Kinematics of an Offset Point

- The kinematic equations for a fixed point on a rigid body are:

$$\mathbf{x} = \mathbf{x}_{cm} + \mathbf{r}$$

$$\mathbf{v} = \mathbf{v}_{cm} + \boldsymbol{\omega} \times \mathbf{r}$$

$$\mathbf{a} = \mathbf{a}_{cm} + \overline{\boldsymbol{\omega}} \times \mathbf{r} + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r})$$

Inverse Mass Matrix

Offset Forces

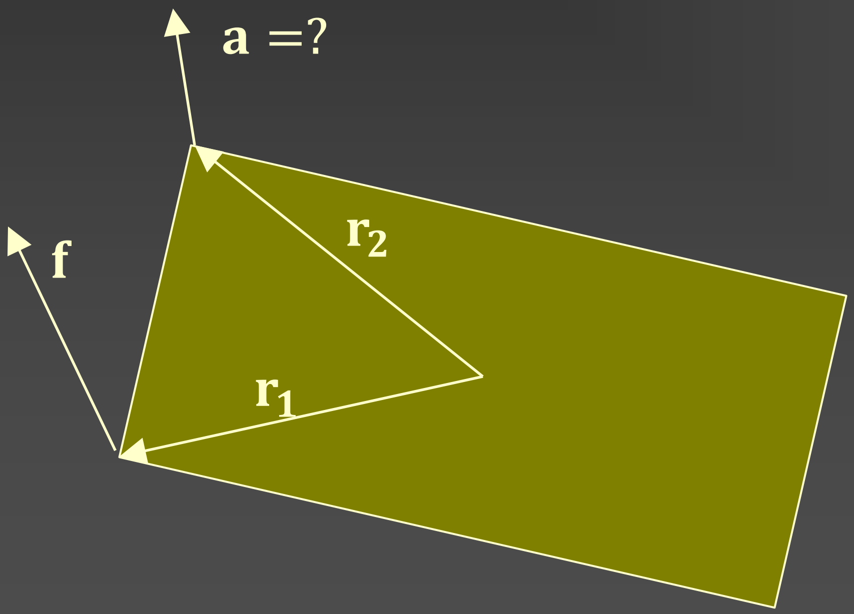
- Suppose we have a particle
- If we apply a force to it, what is the resulting acceleration?
- Easy: $\mathbf{a} = \frac{1}{m}\mathbf{f}$

Offset Forces

- With rigid bodies, the same holds true for the acceleration of the center of mass
- However, what if we're interested in the acceleration of some offset point?
- If we apply a force **f** to a rigid body at offset $\mathbf{r}_1$, what is the resulting acceleration at a (possibly) different offset $\mathbf{r}_2$?

Offset Forces

- If we apply a force **f** to a rigid body at offset $\mathbf{r}_1$, what is the resulting acceleration at a offset $\mathbf{r}_2$?



Offset Forces

- The applied force **f** at $\mathbf{r}_1$ results in a force and torque on the rigid body.
- The force on the center of mass is just **f**, so this results in an acceleration of the center of mass by $\frac{1}{m}\mathbf{f}$
- The torque $\boldsymbol{\tau}$ on the rigid body is $\mathbf{r}_1 \times \mathbf{f}$, leading to an angular acceleration of:

$$\bar{\boldsymbol{\omega}} = \mathbf{I}^{-1} \cdot (\boldsymbol{\tau} - \boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega})$$

Offset Forces

- $\bar{\boldsymbol{\omega}} = \mathbf{I}^{-1} \cdot (\boldsymbol{\tau} - \boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega})$
- We're really just interested in the resulting acceleration from the applied force, so we will ignore the acceleration from the torque-free motion $\boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega}$
- Since $\boldsymbol{\tau} = \mathbf{r}_1 \times \mathbf{f} = \widehat{\mathbf{r}}_1 \cdot \mathbf{f}$, we get

$$\bar{\boldsymbol{\omega}} = \mathbf{I}^{-1} \cdot \widehat{\mathbf{r}}_1 \cdot \mathbf{f}$$

Offset Forces

- So, when we apply a force **f** at **r**$_1$, we get the resulting rigid body accelerations:

$$\mathbf{a}_{cm} = \frac{1}{m}\mathbf{f}$$

$$\bar{\boldsymbol{\omega}} = \mathbf{I}^{-1} \cdot \widehat{\mathbf{r}_1} \cdot \mathbf{f}$$

- But we're interested in the acceleration at offset **r**$_2$, so we need to use:

$$\mathbf{a} = \mathbf{a}_{cm} + \bar{\boldsymbol{\omega}} \times \mathbf{r}_2 + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}_2)$$

Offset Forces

- $$\mathbf{a} = \mathbf{a}_{cm} + \bar{\boldsymbol{\omega}} \times \mathbf{r}_2 + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}_2)$$
- Again, we're just interested in the acceleration resulting from **f**, so we can ignore the centripetal acceleration component $\boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}_2)$

$$\mathbf{a} = \mathbf{a}_{cm} + \bar{\boldsymbol{\omega}} \times \mathbf{r}_2$$

$$\mathbf{a} = \mathbf{a}_{cm} - \mathbf{r}_2 \times \bar{\boldsymbol{\omega}}$$

$$\mathbf{a} = \mathbf{a}_{cm} - \widehat{\mathbf{r}}_2 \cdot \bar{\boldsymbol{\omega}}$$

$$\mathbf{a} = \frac{1}{m}\mathbf{f} - \widehat{\mathbf{r}}_2 \cdot \mathbf{I}^{-1} \cdot \widehat{\mathbf{r}}_1 \cdot \mathbf{f}$$

Inverse Mass Matrix

$$\mathbf{a} = \frac{1}{m}\mathbf{f} - \hat{\mathbf{r}}_2 \cdot \mathbf{I}^{-1} \cdot \hat{\mathbf{r}}_1 \cdot \mathbf{f}$$

$$\mathbf{a} = \left(\begin{bmatrix} 1/m & 0 & 0 \\ 0 & 1/m & 0 \\ 0 & 0 & 1/m \end{bmatrix} - \hat{\mathbf{r}}_2 \cdot \mathbf{I}^{-1} \cdot \hat{\mathbf{r}}_1 \right) \cdot \mathbf{f}$$

$$\mathbf{a} = \mathbf{M}^{-1} \cdot \mathbf{f}$$

$$\mathbf{M}^{-1} = \begin{bmatrix} 1/m & 0 & 0 \\ 0 & 1/m & 0 \\ 0 & 0 & 1/m \end{bmatrix} - \hat{\mathbf{r}}_2 \cdot \mathbf{I}^{-1} \cdot \hat{\mathbf{r}}_1$$

Inverse Mass Matrix

$$\mathbf{M}^{-1} = \begin{bmatrix} 1/m & 0 & 0 \\ 0 & 1/m & 0 \\ 0 & 0 & 1/m \end{bmatrix} - \widehat{\mathbf{r}_2} \cdot \mathbf{I}^{-1} \cdot \widehat{\mathbf{r}_1}$$

- We call $\mathbf{M}^{-1}$ an ‘inverse mass matrix’, (and we can call **M** the mass matrix)
- It lets us apply a force at $\mathbf{r}_1$ and find the resulting acceleration at $\mathbf{r}_2$ in a f=ma format
- It also lets us apply an impulse at $\mathbf{r}_1$ and find the resulting change in velocity
- Note: $\mathbf{r}_1$ can equal $\mathbf{r}_2$, allowing us to find the resulting acceleration at the same offset where we apply the force

Inverse Mass Matrix

- Why do we care?
- Well, this lets us do all kinds of useful things such as collisions and constraints
- For a collision, for example, we can use it to solve what impulse will prevent the velocity of a colliding point to go through another object
- For a constraint, we can solve the constraint force the holds an offset point still (zero acceleration)

Collisions & Constraints

Collisions

- Lets say we have a rigid body that hits the ground at offset point **r**
- We will assume the collision has zero elasticity and high enough friction to zero out any tangential velocity
- The velocity at the collision point is

$$\mathbf{v} = \mathbf{v}_{cm} + \boldsymbol{\omega} \times \mathbf{r}$$

- The collision will result in an impulse that causes the offset velocity **v** to go to zero immediately after the collision
- Therefore, we want to solve for impulse **j**:

$$-(\mathbf{v}_{cm} + \boldsymbol{\omega} \times \ \mathbf{r}\ ) = \mathbf{M}^{-1}\mathbf{j}$$

Collisions

- There’s a lot more to say about collisions
- For starters, that example was of one rigid body colliding with an infinite mass and we’ll need to have rigid bodies colliding with other rigid bodies
- Also, we want to handle non-zero elasticity, and a realistic friction model
- Then comes issues of multiple simultaneous collisions
- Then static contact situations…
- Then rolling and sliding…
- Then stacking…

Constraints

- It is common to add constraints to a rigid body system that allows us to create articulated figures with various joint types
- For example, we could constrain an offset position of one rigid body to match a different offset position on another rigid body, thus creating a ball-and-socket type joint
- The constraint would apply some force to the first rigid body and an equal an opposite force to the second. The constraint force would be whatever force is necessary to keep the two points from separating
- We can use the inverse mass matrix technique to formulate an equation that lets us solve for the unknown constraint force
- If multiple objects are constrained together in a large group, then we must simultaneously solve for all of these constraint forces in one big system of equations

Constraints & Collisions

- It turns out that constraints and collisions can be formulated in very similar ways although there are some important differences (namely, constraints form equality equations and collisions form inequality equations)
- They can all be included in one big system and then solved by a special inequality solver called an LCP solver (LCP stands for linear complementarity problem) or by other techniques
- This lets us combine rigid body motion, constraints, and collisions into one single process
- The details are outside the scope of this class, but I’ve included a pdf on the web page that gives an overview of the state of the art in rigid body simulation