Rigid Body Simulation

Steve Rotenberg
CSE291: Physics Simulation
UCSD
Spring 2019

Hat Operator

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

$$\mathbf{a} \times \mathbf{b} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \cdot \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix}$$

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

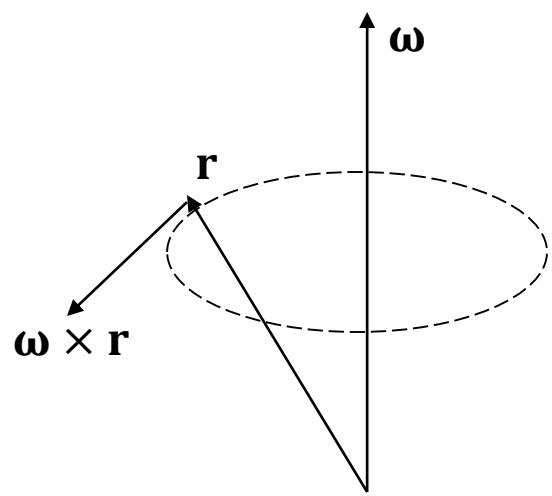
$$\hat{\mathbf{a}} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix}$$

- The *hat operator* lets us replace a non-associative cross product with an associative dot product, so is mainly used today for algebraic convenience

Angular Velocity

- Let's say we have a vector $\mathbf{r}$ that is rotating around the origin, maintaining a fixed distance
- At any instant, it has an angular velocity of $\boldsymbol{\omega}$, and its linear velocity is $\boldsymbol{\omega} \times \mathbf{r}$
- The direction of $\boldsymbol{\omega}$ is the axis of rotation and the magnitude of $\boldsymbol{\omega}$ is the rate of rotation in radians/second

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



Product Rule

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

$$\frac{d(ab)}{dt} = \frac{da}{dt}b + a\frac{db}{dt}$$

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

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

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

- Note that some of these are non-commutative, so the order of multiplication is important

Rigid Body Motion

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 can compute the total mass as the sum of the individual particle masses:

$$m = \sum m_i$$

- For a rigid body, we compute the total mass by integrating the density ρ over the volumetric domain Ω

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

Moment of Momentum

- The linear momentum $\mathbf{p}$ of a particle of mass m and velocity $\mathbf{v}$ is

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

- We define the *moment of momentum* (or *angular momentum*) of a particle at position $\mathbf{r}$ as the vector $\mathbf{L}$

$$\mathbf{L} = \mathbf{r} \times \mathbf{p}$$

- Like linear momentum, angular momentum is conserved in a closed system

Angular Momentum

- If we limit ourselves to the case where the particle $\mathbf{r}$ is undergoing a pure rotational motion (i.e., the length is constant), then we can express the particle velocity $\mathbf{v}$ as a function of the angular velocity $\boldsymbol{\omega}$

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

- This allows us to re-express $\mathbf{L}$ 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:

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

- We've introduced the *rotational inertia tensor* $\mathbf{I}$, which relates the angular momentum $\mathbf{L}$ of a single rotating particle to the angular velocity $\boldsymbol{\omega}$

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

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

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

Rotational Inertia of a Rigid Body

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

$$m\left({r_y}^2 + {r_z}^2\right)$$

- Would be replaced by:

$$\int \rho\left({r_y}^2 + {r_z}^2\right) d\Omega$$

Rigid Body Rotational Inertia

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

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

Rotational Inertia

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

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

- This is similar to how mass relates linear momentum to linear velocity with $\mathbf{p} = m\mathbf{v}$, but adds additional complexity because $\mathbf{I}$ is not only a matrix instead of a scalar, but it also changes over time as the object rotates

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 $\mathbf{f} = m\mathbf{a}$
- 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 $\mathbf{p}$ for linear momentum and vector $\mathbf{L}$ for angular momentum
- We also have a vector $\mathbf{v}$ 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

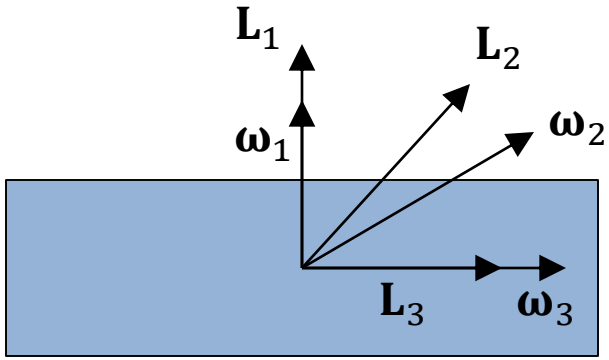
Rotational Inertia

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

- Remember eigenvalue equations of the form $\mathbf{Ax} = \lambda\mathbf{x}$ 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 **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



Diagonalization of Rotational Inertia

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

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

Principal Axes & Inertias

- If we diagonalize the $\mathbf{I}$ matrix, we get an orientation matrix $\mathbf{A}$ and a constant diagonal matrix $\mathbf{I}_0$
- The matrix $\mathbf{A}$ rotates the object from an orientation where the principal axes line up with the *x*, *y*, and *z* axes
- The three diagonal values in $\mathbf{I}_0$, (namely I_x, I_y, and 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)

Rigid Body Dynamics

- The center of mass of a rigid body behaves like a particle:
  - Position: $\mathbf{x}$
  - Velocity: $\mathbf{v} = \frac{d\mathbf{x}}{dt}$
  - Acceleration: $\mathbf{a} = \frac{d\mathbf{v}}{dt} = \frac{d^2\mathbf{x}}{dt^2}$
  - Mass: m
  - Momentum: $\mathbf{p} = m\mathbf{v}$
  - Force: $\mathbf{f} = \frac{d\mathbf{p}}{dt} = m\mathbf{a}$

Rigid Body Dynamics

- Rigid bodies have additional properties for rotation:
  - Orientation: $\mathbf{A}$
  - Angular velocity: $\boldsymbol{\omega}$
  - Angular acceleration: $\bar{\boldsymbol{\omega}} = \frac{d\boldsymbol{\omega}}{dt}$
  - Rotational inertia: $\mathbf{I} = \mathbf{A} \cdot \mathbf{I}_0 \cdot \mathbf{A}^T$
  - Angular momentum: $\mathbf{L} = \mathbf{I} \cdot \boldsymbol{\omega}$
  - Torque: $\boldsymbol{\tau} = \frac{d\mathbf{L}}{dt} = \boldsymbol{\omega} \times \mathbf{I} \cdot \boldsymbol{\omega} + \mathbf{I} \cdot \bar{\boldsymbol{\omega}}$

Applied Forces

- When we apply a force to a rigid body at some position $\mathbf{x}$, this is equivalent to applying the same force on the center of mass as well as a torque equal to $\mathbf{r} \times \mathbf{f}$, where $\mathbf{r} = \mathbf{x} - \mathbf{x}_{cm}$ and $\mathbf{x}_{cm}$ is the position of the center of mass of the rigid body
- Various forces can apply to a rigid body and they are just summed up as a single total force and torque:

$$\mathbf{f}_{total} = \sum \mathbf{f}_i$$

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

Newton-Euler Equations

- The Newton-Euler equations:

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

- Solved for acceleration:

$$\mathbf{a} = \mathbf{f}/m$$
$$\overline{\boldsymbol{\omega}} = \mathbf{I}^{-1}(\boldsymbol{\tau} - \boldsymbol{\omega} \times \mathbf{I} \cdot \boldsymbol{\omega})$$

Torque-Free Motion

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

- We can see that the acceleration will be zero if there is no force
- However, if there is no torque ($\boldsymbol{\tau} = \mathbf{0}$), there may still be a non-zero angular acceleration:

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

- This is called the *torque-free motion*, and this is responsible for the tumbling motion we see in rigid bodies

Rigid Body Constants

Rigid Body Constants

- There are certain properties that remain constant for a rigid body:
  - Shape (defined in ‘body’ coordinate frame)
  - Total mass
  - Center of mass (in body coordinates)
  - Rotational inertia tensor (in body coordinates)
- We will assume the shape is defined by a closed triangular mesh (i.e., an array of 3D points and an array of integer vertex indices (3 for each triangle))
- The other properties are called the *mass properties* of the rigid body, and include 10 unique numbers (1+3+6)

Mass Properties of a Box

- For an axis-aligned box of dimensions $a \times b \times c$, and uniform density ρ:

$$m = \rho abc$$

$$I_x = \frac{m}{12}(b^2 + c^2)$$

$$I_y = \frac{m}{12}(a^2 + c^2)$$

$$I_z = \frac{m}{12}(a^2 + b^2)$$

- The center of mass is at the origin, and the off-diagonal elements of the inertia tensor are all 0

Mass Properties of a Sphere

- For a sphere of radius r and uniform density ρ

$$m = \rho \frac{4}{3} \pi r^3$$

$$I_x = I_y = I_z = \frac{2mr^2}{5}$$

- The center of mass is at the origin, and the off-diagonal elements of the inertia tensor are all 0

Mass Properties of a Rigid Body

- For a general shape:

$$m = \int \rho d\Omega$$

$$c_x = \frac{1}{m}\int \rho r_x d\Omega$$

$$c_y = \frac{1}{m}\int \rho r_y d\Omega$$

$$c_z = \frac{1}{m}\int \rho r_z d\Omega$$

$$I_{xx} = \int \rho\left({r_y}^2 + {r_z}^2\right)d\Omega$$

$$I_{yy} = \int \rho({r_x}^2 + {r_z}^2)d\Omega$$

$$I_{zz} = \int \rho\left({r_x}^2 + {r_y}^2\right)d\Omega$$

$$I_{xy} = -\int \rho r_x r_y d\Omega$$

$$I_{xz} = -\int \rho r_x r_z d\Omega$$

$$I_{yz} = -\int \rho r_y r_z d\Omega$$

Mass Properties

- The mass properties include the total mass, the center of mass, and the rotational inertia tensor
- We can use the Mirtich-Eberley algorithm to compute all 10 of these very efficiently for a triangle mesh, in O(n) time

Model Normalization

- It is often useful to *normalize* our rigid body such that the center of mass is at the origin in body coordinates and the principal axes line up with the *x*, *y*, and *z* axes, thus making the rotational inertia tensor diagonal
- In this form, the original 10 mass properties reduce to only 4 numbers: mass m, and the principal rotational inertias I_x, I_y, and I_z
- If we initialize a model as an axis aligned box, cylinder, or sphere, we can assume it is already aligned this way
- If we load a model from a file, we can normalize it like this:
  1. Load file and store as vertices and triangles
  2. Use Mirtich-Eberley algorithm to compute the 10 mass properties
  3. Translate all mesh vertices by the negative of the center of mass, thus centering it at the origin
  4. Subtract rotational inertia of a particle of mass m at the center of mass
  5. Diagonalize the rotational inertia tensor to get $\mathbf{I}_0$ and $\mathbf{A}$ (we can use the Cyclic Jacobi algorithm…)
  6. Transform all mesh vertices by $\mathbf{A}^T$ to align principal axes to *xyz*

Rigid Body Initialization

- To initialize a rigid body, we can either:
  - Use basic shapes like axis-aligned cylinders, spheres, or boxes and use analytical formulas for the principal inertias
  - Load 3D triangle mesh geometry and compute mass properties and then normalize
- Either way, we end up with 4 principal mass properties and a mesh with vertices and triangles

Rigid Body Simulation

Rigid Body Class

```
class RigidBody {
public:
        RigidBody(float a,float b,float c);          // Axis aligned box
        RigidBody(const char *filename);             // Load geometry from file and normalize

        void ApplyForce(const glm::vec3 &f,const glm::vec3 &pos);
        void Integrate(float timestep);
private:
        // Constants
        std::vector<glm::vec3> Points;
        std::vector<uint> Triangles;

        float Mass;
        glm::vec3 PrincipalInertia;

        // Variables
        glm::vec3 Position;
        glm::vec3 Momentum;
        glm::mat3 Orientation;
        glm::vec3 AngularMomentum;

        // Temps
        glm::vec3 Force;
        glm::vec3 Torque;
};
```

Apply Force

```
void RigidBody::ApplyForce(const glm::vec3 &f,const glm::vec3 &pos)
{
        Force += f;
        Torque += glm::cross(pos-Position, f);
}
```

Velocity vs. Momentum

- For linear motion (and using forward Euler integration), we start with forces and ultimately end up with positions:

$$\mathbf{a} = \mathbf{f}/m$$
$$\mathbf{v}_{i+1} = \mathbf{v}_i + \mathbf{a}\Delta t$$
$$\mathbf{x}_{i+1} = \mathbf{x}_i + \mathbf{v}_{i+1}\Delta t$$

- We could just as well have used momentum $\mathbf{p} = m\mathbf{v}$ instead of velocity:

$$\mathbf{p}_{i+1} = \mathbf{p}_i + \mathbf{f}\Delta t$$
$$\mathbf{v} = \mathbf{p}_{i+1}/m$$
$$\mathbf{x}_{i+1} = \mathbf{x}_i + \mathbf{v}\Delta t$$

- The difference would be negligible, because of the very simple relationship between momentum $\mathbf{p}$ and velocity $\mathbf{v}$

Angular Velocity vs. Angular Momentum

- It's a little different in the angular case, due to the more complex relationship between angular velocity $\boldsymbol{\omega}$ and angular momentum $\mathbf{L}$

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

- This leads to the complex tumbling behavior of rigid bodies
- As with the linear case, we can choose to track velocity $\boldsymbol{\omega}$ over time or momentum $\mathbf{L}$
- However, in this case, it may make a difference which one we choose

Conservation of Angular Momentum

- It is nice to have fundamental properties like conservation of angular momentum preserved exactly if possible
- If we track (integrate) the angular momentum over time instead of the angular velocity, this gives us a much easier way to explicitly preserve conservation of angular momentum
- It means that if a rigid body is tumbling in space and not experiencing any forces or torques, it will maintain a constant angular momentum forever, regardless of the time step, as nothing is ever changing the angular momentum
- The same would not be the case if we tracked angular velocity over time instead of angular momentum. We would get gradual drift of angular momentum due to inaccuracies introduced in the finite time integration

Angular Velocity vs. Angular Momentum

- If we track angular velocity $\boldsymbol{\omega}$:

$$\bar{\boldsymbol{\omega}} = \mathbf{I}^{-1}(\boldsymbol{\tau} - \boldsymbol{\omega}_i \times \mathbf{I} \cdot \boldsymbol{\omega}_i)$$
$$\boldsymbol{\omega}_{i+1} = \boldsymbol{\omega}_i + \bar{\boldsymbol{\omega}}\Delta t$$
$$\mathbf{A}_{i+1} = \mathrm{Rotate}(\boldsymbol{\omega}_{i+1}\Delta t) \cdot \mathbf{A}_i$$

- Or if we track angular momentum $\mathbf{L} = \mathbf{I}\boldsymbol{\omega}$:

$$\mathbf{L}_{i+1} = \mathbf{L}_i + \boldsymbol{\tau}\Delta t$$
$$\boldsymbol{\omega} = \mathbf{I}^{-1}\mathbf{L}_{i+1}$$
$$\mathbf{A}_{i+1} = \mathrm{Rotate}(\boldsymbol{\omega}\Delta t) \cdot \mathbf{A}_i$$

Buss' Method

- "Accurate and Efficient Simulation of Rigid Body Rotations", Samuel Buss, 2001
- The paper presents several methods for updating the orientation of a rigid body based on the angular momentum
- The methods are simple to implement and significantly improve the long term rotation behavior of rigid bodies

Buss' Method

- Augmented second-order method:

$$\boldsymbol{\omega} = \mathbf{I}^{-1}\mathbf{L}_i$$
$$\dot{\boldsymbol{\omega}} = \mathbf{I}^{-1}(\boldsymbol{\tau} - \boldsymbol{\omega} \times \mathbf{L}_i)$$

$$\bar{\boldsymbol{\omega}} = \boldsymbol{\omega} + \frac{h}{2}\dot{\boldsymbol{\omega}} + \frac{h^2}{\mathbf{12}}(\dot{\boldsymbol{\omega}} \times \boldsymbol{\omega})$$

$$\mathbf{A}_{i+1} = \mathrm{Rotate}(\bar{\boldsymbol{\omega}}\Delta t) \cdot \mathbf{A}_i$$
$$\mathbf{L}_{i+1} = \mathbf{L}_i + \boldsymbol{\tau}\Delta t$$

Rotational Inertia Inverse

- 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 $(\mathbf{S} \cdot \mathbf{T})^{-1} = \mathbf{T}^{-1} \cdot \mathbf{S}^{-1}$
- Likewise $(\mathbf{STU})^{-1} = \mathbf{U}^{-1}\mathbf{T}^{-1}\mathbf{S}^{-1}$
- 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 with 3 divisions

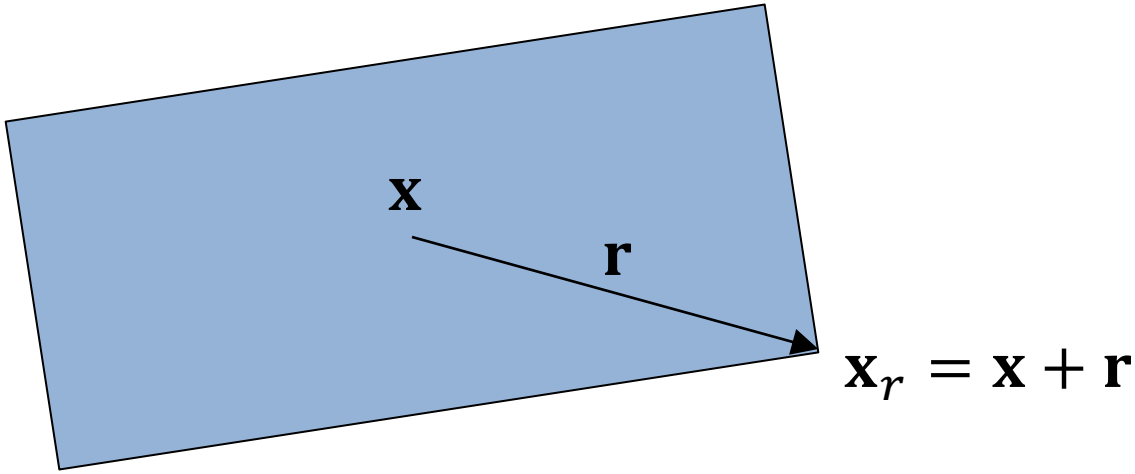
Kinematics of Offset Points

Offset Position

- Assume that all vectors are in the world coordinate system
- Let's say we have a point on a rigid body
- If $\mathbf{r}$ is the relative offset of the point from the center of mass $\mathbf{x}$, then the world space position $\mathbf{x}_r$ of the offset point is:

$$\mathbf{x}_r = \mathbf{x} + \mathbf{r}$$

Offset Position



Offset Velocity

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

$$\mathbf{x}_r = \mathbf{x} + \mathbf{r}$$

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

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

Offset Acceleration

- The offset acceleration is the derivative of the offset velocity

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

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

$$\mathbf{a}_r = \mathbf{a} + \bar{\boldsymbol{\omega}} \times \mathbf{r} + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r})$$

Kinematics of an Offset Point

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

$$\mathbf{x}_r = \mathbf{x} + \mathbf{r}$$

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

$$\mathbf{a}_r = \mathbf{a} + \bar{\boldsymbol{\omega}} \times \mathbf{r} + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r})$$

Inverse Mass Matrix

Offset Forces

- Suppose we have a particle
- If we apply a force to it, what is the resulting acceleration?

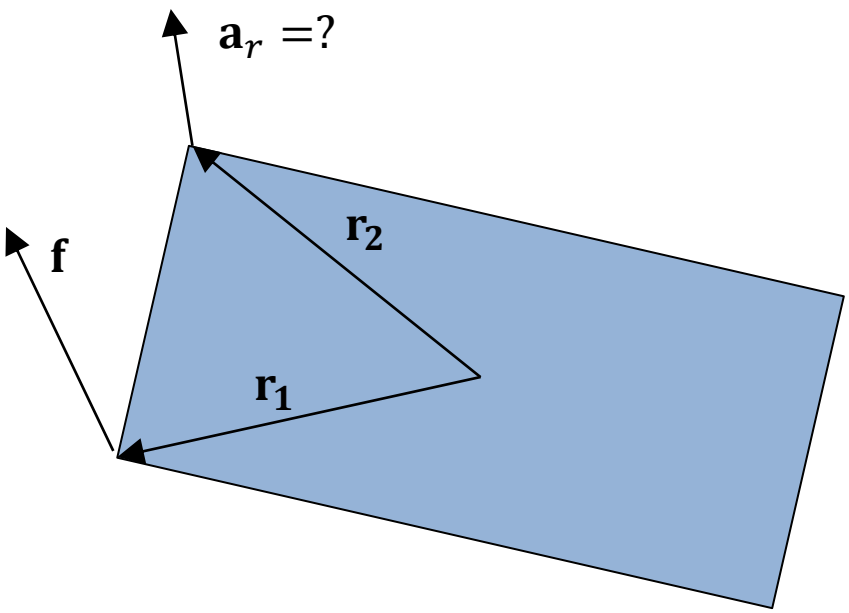
- Easy: $\mathbf{a} = \mathbf{f}/m$

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 $\mathbf{f}$ to a rigid body at offset $\mathbf{r}_1$, what is the resulting acceleration $\mathbf{a}_r$ at a (possibly) different offset $\mathbf{r}_2$?

Offset Forces

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



Offset Forces

- The applied force $\mathbf{f}$ at $\mathbf{r}_1$ results in a force and torque on the rigid body
- The force on the center of mass is just $\mathbf{f}$, so this results in an acceleration of the center of mass:

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

- The torque $\boldsymbol{\tau}$ on the rigid body is $\mathbf{r}_1 \times \mathbf{f}$, and from Euler's equation, this leads to an angular acceleration of

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

Offset Forces

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

- We're really just interested in the acceleration resulting from the applied force, so we will ignore the torque-free motion $-\boldsymbol{\omega} \times \mathbf{I} \cdot \boldsymbol{\omega}$, leaving:

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

Offset Forces

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

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

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

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

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

Offset Forces

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

- Again, we're just interested in the acceleration resulting from the applied force $\mathbf{f}$, so we can ignore the centripetal acceleration component $\boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}_2)$ leaving:

$$\mathbf{a}_r = \mathbf{a} + \bar{\boldsymbol{\omega}} \times \mathbf{r}_2$$
$$\mathbf{a}_r = \mathbf{a} - \mathbf{r}_2 \times \bar{\boldsymbol{\omega}}$$
$$\mathbf{a}_r = \mathbf{a} - \widehat{\mathbf{r}_2} \cdot \bar{\boldsymbol{\omega}}$$

Offset Forces

$$\mathbf{a}_r = \mathbf{a} - \widehat{\mathbf{r}_2} \cdot \overline{\boldsymbol{\omega}}$$

- When we substitute our previous results for $\mathbf{a}$ and $\overline{\boldsymbol{\omega}}$, we get:

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

Inverse Mass Matrix

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

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

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

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

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 $\mathbf{M}^{-1}$ the *inverse mass matrix* (and we can call $\mathbf{M}$ the *mass matrix*)
- It lets us apply a force at $\mathbf{r}_1$ and find the resulting acceleration at $\mathbf{r}_2$
- It also lets us apply an impulse at $\mathbf{r}_1$ and find the resulting change in velocity at $\mathbf{r}_2$
- 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 from going through another object
- For a constraint, we can solve the constraint force that holds an offset point still (zero acceleration)

Constraints

Holonomic Constraints

- A *holonomic constraint* is an equality constraint applied to position variables
- For example, if we connect two rigid bodies with a ball-and-socket joint, we are constraining joint offset positions of the two bodies to be equal in world space
- Various types of constraints can be used to connect rigid bodies, such as:
  - Hinge joints (1 DOF rotation)
  - Universal joints (2 DOF rotation)
  - Ball-and-socket joint (3 DOF rotation)
  - Prismatic joint (1 DOF translation)
  - Gear constraint (1 DOF translation)
- These can be used to connect multiple rigid bodies into an *articulated body*

Non-Holonomic Constraints

- *Non-holonomic constraints* are constraints that are not holonomic (duh!)
- For our purposes, this will mainly refer to *inequality* constraints on position and velocity variables (as opposed to equality constraints)
- These mainly show up in collision and contact situations, where a collision can momentarily be treated as an inequality constraint on velocity
- The closing velocity at the collision point must be less than or equal to zero, which is an inequality

Solution Methods

- Direct solver
- Linear complimentarity
- Pairwise solver
- Featherstone's algorithm
- Lagrangian dynamics

- More on collisions & constraints in the next lecture…