Simulation in Computer Graphics
Rigid Bodies

Matthias Teschner
Motivation

www.hairyharry.de
Outline

- Representation
  - Position and orientation
- Kinematics
  - Linear and angular velocity
- Dynamics
  - Inertia tensor
  - Linear and angular momentum
  - Force and torque
  - Simulation loop
- Miscellaneous
Elastic Solid vs. Rigid Body

Elastic Solid
- Represented with $n$ particles
- Relative movements
- Mass, position, velocity, force considered per particle
- No notion of orientation

Rigid Body
- Represented with $n$ particles
- No relative movements
- Interaction implicitly modeled
- One position / orientation
- One velocity / angular velocity
- Mass distribution
Particle Representation of a Rigid Body

**Initial state**

- Reference position: $\bar{x}_{CM} = (0, 0, 0)^T$

- Position of a particle relative to the reference position: $\bar{r}_i = \bar{x}_i - \bar{x}_{CM}$

- Particle position in global coordinates: $\bar{x}_i = \bar{x}_{CM} + r_i + \text{Rot}(\bar{r}_i)$

**Body frame coordinates**

**World frame coordinates**

Rigid body consists of $n$ particles.
**Reference Position – Center of Mass**

\[ \mathbf{x}_{CM} = \frac{\sum_i m_i \mathbf{x}_i}{\sum_i m_i} = \frac{\sum_i m_i \mathbf{x}_i}{M} \]

Center of mass

\[ M \mathbf{x}_{CM} = \sum_i m_i \mathbf{x}_i \]

Same point under translation and rotation

\[ \frac{1}{M} \sum_i m_i T \mathbf{x}_i = T \frac{1}{M} \sum_i m_i \mathbf{x}_i = T \mathbf{x}_{CM} \]
Center of Mass - Property

The linear acceleration of the center of mass can be computed from the mass of the rigid body and from the sum of all forces acting at arbitrary rigid body positions. The positions of the applied forces do not influence the linear acceleration.

\[
f_i = m_i \frac{d^2}{dt^2} \mathbf{x}_i \quad \text{Force at position } \mathbf{x}_i
\]

\[
F = \sum_i f_i = \sum_i m_i \frac{d^2}{dt^2} \mathbf{x}_i = \frac{d^2}{dt^2} \sum_i m_i \mathbf{x}_i \quad \text{Overall force}
\]

\[
= \frac{d^2}{dt^2} M \mathbf{x}_{\text{CM}} = M \frac{d^2}{dt^2} \mathbf{x}_{\text{CM}}
\]

\[
F = \sum_i f_i = M \frac{d^2}{dt^2} \mathbf{x}_{\text{CM}}
\]
Orientation in 3D

- Rotation matrix
  - 3x3 elements to represent 3 degrees of freedom

\[
\text{Rot}(\mathbf{v}) = A\mathbf{v} = \begin{pmatrix} a_1 & a_2 & a_3 \end{pmatrix} \cdot \begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix} = v_x a_1 + v_y a_2 + v_z a_3
\]
Orientation in 3D

- $A$ has to be orthonormal
  
  $$AA^T = \begin{pmatrix} a_1 & a_2 & a_3 \end{pmatrix} \cdot \begin{pmatrix} a_1^T \\ a_2^T \\ a_3^T \end{pmatrix} = I$$
  
  $\text{Det}(A) = 1$

- Magnitude of eigenvalues is 1, so $A$ preserves the length of $r$

- Position of a body point $x_i = x_{CM} + r_i = \bar{x}_{CM} + t + A\bar{r}_i$
  
  translation + rotation
Outline

- Representation
  - Position and orientation
- Kinematics
  - Linear and angular velocity
- Dynamics
  - Inertia tensor
  - Linear and angular momentum
  - Force and torque
  - Simulation loop
- Miscellaneous
Body in Motion

- Time-dependent position
  \[ x_i(t) = \bar{x}_{CM} + t(t) + A(t)\bar{r}_i = x_{CM}(t) + A(t)\bar{r}_i \]

- Velocity
  \[ \frac{d}{dt} x_i(t) = \frac{d}{dt} x_{CM}(t) + \frac{d}{dt} A(t)\bar{r}_i + A(t) \frac{d}{dt} \bar{r}_i = \frac{d}{dt} x_{CM}(t) + \frac{d}{dt} A(t)\bar{r}_i \]
  linear velocity + angular velocity
Angular Velocity in 3D

- Angular velocity $\omega$ is a 3D vector
  - In direction of axis of rotation
  - $|\omega|$ is the magnitude of the angular velocity [rad/s]

\[
\left\| \frac{d}{dt} \mathbf{x} \right\| = |\omega| \cdot r = |\omega| \cdot |\mathbf{x}| \cdot \sin(\phi)
\]

\[
\frac{d}{dt} \mathbf{x} = \omega \times \mathbf{x}
\]

\[
\omega = \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} \rightarrow \tilde{\omega} = \begin{pmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{pmatrix}
\]

\[
\frac{d}{dt} \mathbf{x} = \tilde{\omega} \mathbf{x}
\]
Rigid Body Kinematics

– What is the relation between \( \tilde{\omega} \) and \( \frac{d}{dt} A \)?

– Angular velocity rotates all axis (columns of \( A \))

\[
\frac{d}{dt} A = \begin{pmatrix} \frac{d}{dt} a_1 \\ \frac{d}{dt} a_2 \\ \frac{d}{dt} a_3 \end{pmatrix} = \begin{pmatrix} \tilde{\omega} \cdot a_1 \\ \tilde{\omega} \cdot a_2 \\ \tilde{\omega} \cdot a_3 \end{pmatrix} = \tilde{\omega} \cdot A
\]

– Position of a point \( x_i(t) = x_{CM}(t) + A(t) \cdot \tilde{r}_i \)

– Velocity of a point

\[
\frac{d}{dt} x_i(t) = \frac{d}{dt} x_{CM}(t) + \frac{d}{dt} A(t) \cdot \tilde{r}_i + A(t) \cdot \frac{d}{dt} \tilde{r}_i
\]

\[
\frac{d}{dt} x_i(t) = v(t) + \tilde{\omega}(t) \cdot A(t) \cdot \tilde{r}_i
\]

\[
\frac{d}{dt} x_i(t) = v(t) + \tilde{\omega}(t) \cdot (x_i(t) - x_{CM}(t))
\]
State Vector

\[
\begin{pmatrix}
\mathbf{x}(t) \\
\mathbf{v}(t)
\end{pmatrix}
\quad
\begin{pmatrix}
\mathbf{x}(t) \\
A(t) \\
\mathbf{v}(t) \\
\omega(t)
\end{pmatrix}
\]

Particle

Rigid body
Outline

– Representation
  – Position and orientation
– Kinematics
  – Linear and angular velocity
– Dynamics
  – Inertia tensor
  – Linear and angular momentum
  – Force and torque
  – Simulation loop
– Miscellaneous
Rigid Body Dynamics

- Forces change
  - Linear velocity
  - Angular velocity
- Linear velocity change

\[ F = \sum_i f_i = \sum_i m_i \frac{d^2}{dt^2} x_i = \frac{d^2}{dt^2} \sum_i m_i x_i \]

\[ = \frac{d^2}{dt^2} M \dot{x}_{CM} = M \frac{d^2}{dt^2} \dot{x}_{CM} \]

\[ \frac{d^2}{dt^2} \dot{x}_{CM} = \frac{F}{M} = \sum_i \frac{f_i}{M} \]

- Like a particle, but ...
Angular Momentum

– The angular momentum of a particle w.r.t. the center of mass is

\[ L_i = \mathbf{r}_i \times m_i \mathbf{v}_i = \mathbf{r}_i \times m_i (\mathbf{\omega} \times \mathbf{r}_i) \]

– The total angular momentum of the body is

\[
L = \sum_i L_i = \sum_i \mathbf{r}_i \times m_i (\mathbf{\omega} \times \mathbf{r}_i) \\
= \sum_i -m_i \mathbf{\tilde{r}}_i \mathbf{\tilde{r}}_i \mathbf{\omega} = \left( \sum_i -m_i \mathbf{\tilde{r}}_i \mathbf{\tilde{r}}_i \right) \cdot \mathbf{\omega} \\
= I\mathbf{\omega}
\]
Inertia Tensor

- The total angular momentum is \( \mathbf{L} = \mathbf{I} \omega \) with \( \mathbf{I} \) being a 3x3 matrix (the inertia tensor of the body).
- \( \mathbf{I} \) depends on the rotated configuration \( \mathbf{I} = \sum_i -m_i \mathbf{\tilde{r}}_i \mathbf{\tilde{r}}_i \).
- The inertia tensor for the original body can be pre-computed, e.g. \( \tilde{\mathbf{I}} = \sum_i -m_i \mathbf{\tilde{r}}_i \mathbf{\tilde{r}}_i \).
- Similarity transform relates time-dependent \( \mathbf{I} \) and pre-computed \( \tilde{\mathbf{I}} \): \( \mathbf{I} = \mathbf{A} \tilde{\mathbf{I}} \mathbf{A}^T \).

\[
\mathbf{L} = \mathbf{I} \omega = \mathbf{A} \tilde{\mathbf{I}} \mathbf{A}^T \mathbf{A} \tilde{\omega} = \mathbf{A} \tilde{\mathbf{I}} \tilde{\omega}
\]
**Torque**

- The torque of a particle w.r.t. the center of mass is
  \[ \tau_i = \mathbf{r}_i \times \mathbf{f}_i \]
- The total torque of the body is
  \[ \mathbf{\tau} = \sum_i \tau_i = \sum_i \mathbf{r}_i \times \mathbf{f}_i \]
Newton's Second Law (Angular)

- Angular momentum \( \mathbf{L} = \sum_i \mathbf{r}_i \times m_i \mathbf{v}_i = \mathbf{I} \omega \)
- Torque \( \mathbf{\tau} = \sum_i \mathbf{r}_i \times \mathbf{f}_i \)
- The angular version of Newton's Second law reads \( \frac{d}{dt} \mathbf{L} = \mathbf{\tau} \)
- Tells us, how the forces \( \mathbf{f}_i \)
  change the angular velocity \( \omega \)
  \( \mathbf{\tau} = \sum_i \mathbf{r}_i \times \mathbf{f}_i \)
  \( \mathbf{L} = \mathbf{L} + \Delta t \cdot \mathbf{\tau} \)
  \( \omega = \mathbf{I}^{-1} \mathbf{L} \)
Linear vs. Angular Quantities

Linear momentum
\( p = M v \)

Angular momentum
\( L = I \omega \)

Linear velocity
\( v = M^{-1} p \)

Angular velocity
\( \omega = I^{-1} L \)

Time-derivative of the linear momentum
\( \frac{d}{dt} p = F \)

Time-derivative of the angular momentum
\( \frac{d}{dt} L = \tau \)
Dynamics

\[
\frac{d}{dt} \begin{pmatrix}
    x(t) \\
    A(t) \\
    p(t) \\
    L(t)
\end{pmatrix} = \begin{pmatrix}
    v(t) \\
    \tilde{\omega} \cdot A(t) \\
    F(t) \\
    \tau(t)
\end{pmatrix}
\]
Simulation Step

Pre-computation
\[ M \leftarrow \sum_i m_i \]
\[ \bar{x}_{CM} \leftarrow \frac{1}{M} \sum_i x_i m_i \]
\[ \bar{r}_i \leftarrow \bar{x}_i - \bar{x}_{CM} \]
\[ \bar{I}^{-1} \leftarrow \left( \sum_i -m_i \bar{r}_i \bar{r}_i \right)^{-1} \]

Initialization
\[ \bar{x}_{CM}, \bar{v}_{CM}, A, L \]
\[ \bar{I}^{-1} \leftarrow A \bar{I}^{-1} A^T \]
\[ \omega \leftarrow \bar{I}^{-1} L \]

Simulation step
\[ \tau \leftarrow \sum_i r_i \times f_i \]
\[ F \leftarrow \sum_i f_i \]
\[ \bar{x}_{CM} \leftarrow \bar{x}_{CM} + \Delta t \cdot \bar{v}_{CM} \]
\[ \bar{v}_{CM} \leftarrow \bar{v}_{CM} + \Delta t \cdot F / M \]
\[ A \leftarrow A + \Delta t \cdot \bar{\omega} A \]
\[ L \leftarrow L + \Delta t \cdot \tau \]
\[ \bar{I}^{-1} \leftarrow A \bar{I}^{-1} A^T \]
\[ \omega \leftarrow \bar{I}^{-1} L \]
\[ r_i \leftarrow A \cdot \bar{r}_i \]
\[ x_i \leftarrow x_{CM} + r_i \]
\[ \bar{v}_i \leftarrow \bar{v}_{CM} + \omega \times r_i \]
Reorthonormalization of the Orientation

- Orientation matrix is updated with $A \leftarrow A + \Delta t \cdot \tilde{\omega}A$
- Errors accumulate
- $A$ is not orthonormal anymore
- Gram-Schmidt orthonormalization

$$b_1 = a_1/|a_1|$$
$$b_2 = a_2 - (b_1 \cdot a_2)b_1$$
$$b_2 = b_2/|b_2|$$
$$b_3 = a_3 - (b_1 \cdot a_3)b_1 - (b_2 \cdot a_3)b_2$$
$$b_3 = b_3/|b_3|$$
Force vs. Torque Puzzle

- Is force being considered twice?
  - Linear and angular acceleration ($F = \sum_i f_i$, $\tau = \sum_i r_i \times f_i$)

![Diagram showing force and torque over time]

10 s later

$E = \frac{1}{2} M v^T v$

10 s later

longer path!

$E = \frac{1}{2} M v^T v + \frac{1}{2} \omega^T I \omega$
Outline

- Representation
  - Position and orientation
- Kinematics
  - Linear and angular velocity
- Dynamics
  - Inertia tensor
  - Linear and angular momentum
  - Force and torque
  - Simulation loop
- Miscellaneous
Alternative Orientation Representations

- Quaternions
  \[ q = (w, x, y, z) \]
- 4 components represent 3 DoF
  \[ q = \left( \cos \left( \frac{\phi}{2} \right), \sin \left( \frac{\phi}{2} \right) \cdot (a_x, a_y, a_z) \right) \]
- Often used in rigid body computations for rotations
  \[ \text{Rot}(v) = q \circ (0, v_x, v_y, v_z) \circ q^{-1} \]
Quaternions

- Quaternions are an extension of complex numbers

\[ q = (w, v) = w + v = (w, x, y, z) \]
\[ = w + x \cdot i + y \cdot j + z \cdot k \]
\[ i^2 = j^2 = k^2 = -1 \]
\[ i \cdot j = k, \quad j \cdot k = i, \quad k \cdot i = j \]
\[ j \cdot i = -k, \quad k \cdot j = -i, \quad i \cdot k = -j \]
**Basic Operations**

- Addition
  \[ q_1 + q_2 = (w_1 + w_2) + (v_1 + v_2) \]

- Dot product
  \[ q_1 \cdot q_2 = w_1 w_2 + v_1 \cdot v_2 \]

- Conjugate
  \[ \overline{q} = w + \overline{v} = w - v \]

- Magnitude (module)
  \[ |q|^2 = q \cdot \overline{q} = w^2 + v \cdot v \]
  \[ = w^2 + v_x^2 + v_y^2 + v_z^2 \]
  \[ q^{-1} = \frac{\overline{q}}{|q|^2} \]
Basic Operations

- Multiplication

\[ q_1 \circ q_2 = (w_1 + v_1)(w_2 + v_2) \]
\[ = (w_1w_2 - v_1v_2) + (w_1v_2 + w_2v_1 + v_1 \times v_2) \]
\[ = (w_1w_2 - x_1x_2 - y_1y_2 - z_1z_2) \]
\[ + (w_1x_2 + w_2x_1 + y_2z_1 - y_1z_2)i \]
\[ + (w_1y_2 + w_2y_1 + z_2x_1 - z_1x_2)j \]
\[ + (w_1z_2 + w_2z_1 + x_2y_1 - x_1y_2)k \]
Unit Quaternions

- Quaternions with $|q| = 1$ can be used to represent orientations.
- Corresponding orientation matrix
  \[ A = 2 \begin{pmatrix}
  0.5 - y^2 - z^2 & xy + wz & xz - wy \\
  xy - wz & 0.5 - x^2 - z^2 & yz + wx \\
  xz + wy & yz - wx & 0.5 - x^2 - y^2
\end{pmatrix} \]
- Rotation of a point $Ap = q \circ (0, p^T) \circ \bar{q}$
Application in the Simulation Loop

- Initialization of orientation \( \mathbf{q} \) and angular velocity \( \mathbf{\omega} \) in the body frame
- Update of the angular velocity: \( \frac{d}{dt} \mathbf{\omega} = \mathbf{I}^{-1}(\mathbf{\tau} - \mathbf{\omega} \times (\mathbf{I}\mathbf{\omega})) \)
- Update of the orientation: \( \frac{d}{dt} \mathbf{q} = 0.5 \mathbf{q} \circ (0, \mathbf{\omega}^T) \)
- Updated orientation has to be normalized
- Torque needs to be transformed from world to body frame, angular velocity from body to world frame
Contact Handling

- Detect collisions (see collision detection slides)
- Avoid penetrations
  - Change time step or
  - Push body back
- Compute collision response
  - Colliding contact ("easy")
  - Resting contact (hard)
Colliding Contact

- Force driven
  - Penetration causes force
  - Late, slow, easy to compute
- Impulse driven
  - Manipulation of velocities instead of accelerations
  - Coefficient of restitution $e$ ($e=0$ plastic, $e=1$ elastic)

$$\Delta v_{CM} = \frac{J}{M}$$
$$\Delta L = (x_{impact} - x_{CM}) \times J$$
$$J = jn$$
$$j = \frac{-(1+e)v_{rel}}{\frac{1}{M_a} + \frac{1}{M_b} + \left[(I_a^{-1}(r_a \times n)) \times r_a + (I_b^{-1}(r_b \times n)) \times r_b\right] \cdot n}$$
Resting Contact

- Find all collisions with small relative velocities
- Solve for all contact forces simultaneously such that for each contact force
  - The force prevents interpenetration
  - The force is repulsive only
  - The force is zero if the bodies separate
- Linear system / linear complementary problem (LCP)
Summary

– Representation
  – Position and orientation
– Kinematics
  – Linear and angular velocity
– Dynamics
  – Inertia tensor
  – Linear and angular momentum
  – Force and torque
  – Simulation loop
– Miscellaneous
References

Acknowledgement

– This slide set is courtesy of Matthias Mueller-Fischer, NVIDIA Corp. It has been modified.