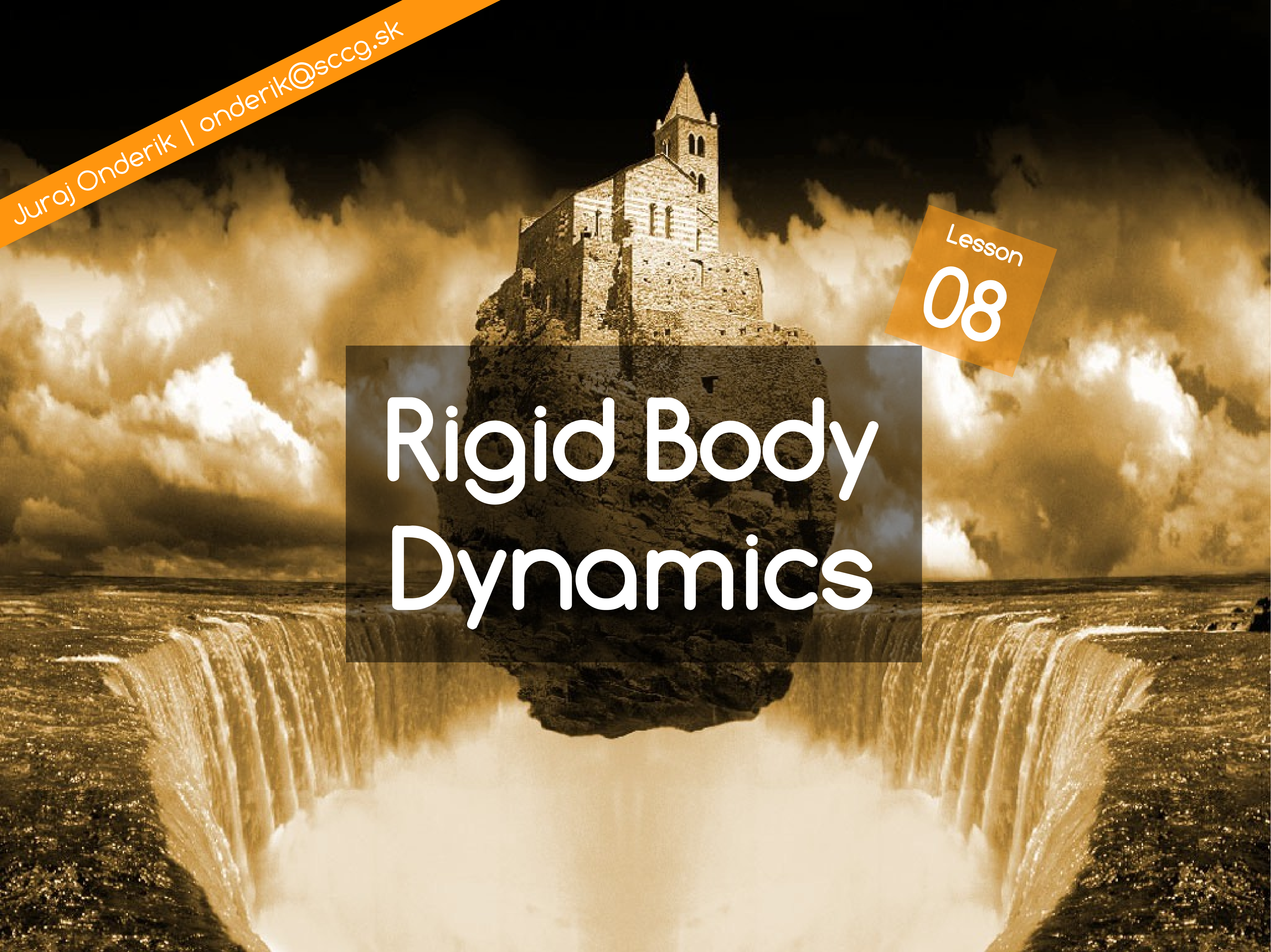


Juraj Onderik | onderik@sccg.sk

Lesson 08

Rigid Body Dynamics

Lesson 08 Outline

- Problem definition and motivations
- Dynamics of rigid bodies
- The equation of unconstrained motion (ODE)
- User and time control
- Demos / tools / libs

Rigid Body Concepts

Concept of Rigid Bodies

- Assumption of Rigidity: The shape of rigid body never undergoes any deformation during simulation
- Motion concept: Due to rigidity overall motion of body is a composition of

- 1) Linear motion of the center of mass (CoM)
- 2) Angular motion - rotation of body shape around center of mass

Position and Orientation

- Position is represented as vector $\mathbf{c} = (x, y, z)$
- Orientation can by represented using:
- 1) Euler Angles: $\mathbf{q} = (\varphi, \theta, \psi)$
  - This is the minimal 6 (3+3) DOF representation of body.
  - Problems of gimbal lock (non-uniqueness)
- 2) Rotation Matrices: $\mathbf{R} = (R_{i,j}) \in R^{3x3}$
  - Overdetermined representation. Must by orthogonalized.
- 3) Unit Quaternions: $\mathbf{q} = (x, y, z, w)$
  - 7 (3+4) DOF representation solved by simple normalization. Very suitable for angular velocity integration

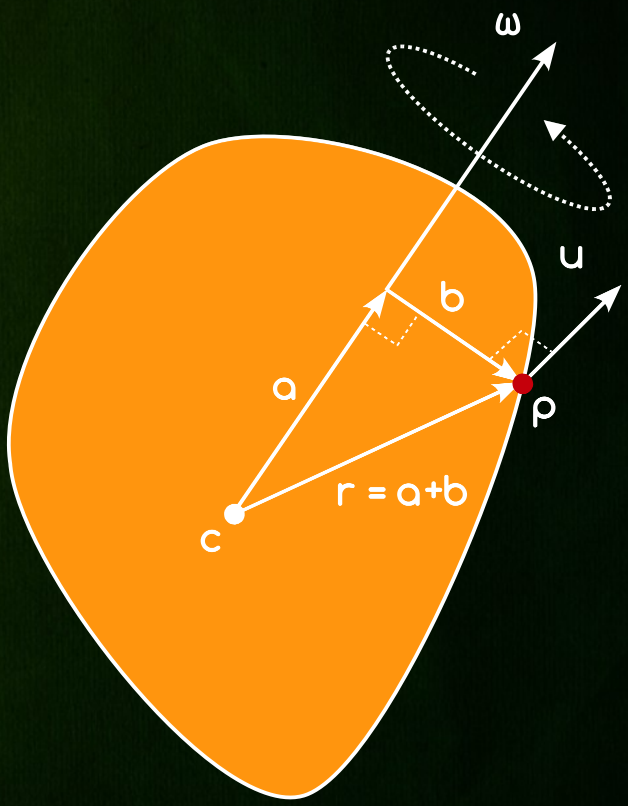
Linear and Angular Velocity

- Linear velocity $v(t)$ is simply the time derivative of position
  - Formally: $v(t) = c'(t) = dc(t)/dt$

- Angular velocity $\omega(t)$ is a vector parallel to rotational axis with the length equal to spin velocity
  - Spin velocity = total radians body spin around rotational axis per second.
  - Formally: $q'(t) = 0.5\, Q\, \omega(t)$ (see later for details)

Linear and Angular Velocity

- Assume some body point $p = c + r$
  - Local displacement $r = a + b$ can be decomposed into axis parallel "a" and axis perpendicular "b"
- Current velocity u of point p is
  - Perpendicular to rotation axis
  - Proportional to length of angular velocity $|\omega|$ and distance from rotation axis $|b|$
  - Formally $|u| = |\omega||b| \rightarrow u = \omega \times b$
- Since $\omega \times a = 0$
- $u = \omega \times b = \omega \times a + \omega \times b = \omega \times r \ (= r')$



Linear and Angular Velocity

- Cross product matrix $a^\times$ for vector $a = (a_x, a_y, a_z)$ is
  - antisymmetric 3x3 matrix

$$\mathbf{a} \times \mathbf{b} = \mathbf{a}^\times \mathbf{b} = \begin{pmatrix} 0 & -\mathbf{a}_z & +\mathbf{a}_y \\ +\mathbf{a}_z & 0 & -\mathbf{a}_x \\ -\mathbf{a}_y & +\mathbf{a}_x & 0 \end{pmatrix} \begin{pmatrix} \mathbf{b}_x \\ \mathbf{b}_y \\ \mathbf{b}_z \end{pmatrix}$$

- Rotation matrix R is a orthonormal 3x3 matrix

$$\mathbf{R} = \begin{pmatrix} \mathbf{R}_x & \mathbf{R}_y & \mathbf{R}_z \end{pmatrix} = \begin{pmatrix} \mathbf{R}_{xx} & \mathbf{R}_{xy} & \mathbf{R}_{xz} \\ \mathbf{R}_{yx} & \mathbf{R}_{yy} & \mathbf{R}_{yz} \\ \mathbf{R}_{zx} & \mathbf{R}_{zy} & \mathbf{R}_{zz} \end{pmatrix}$$

Linear and Angular Velocity

- Time derivative of rotation matrix R with respect to angular velocity ω is (assuming r' = ω x r = $\omega^\times$ r)

$$\dot{\mathbf{R}} = \begin{pmatrix} \dot{\mathbf{R}}_x & \dot{\mathbf{R}}_y & \dot{\mathbf{R}}_z \end{pmatrix} = \begin{pmatrix} \omega^\times \mathbf{R}_x & \omega^\times \mathbf{R}_y & \omega^\times \mathbf{R}_z \end{pmatrix} = \omega^\times \begin{pmatrix} \mathbf{R}_x & \mathbf{R}_y & \mathbf{R}_z \end{pmatrix} = \omega^\times \mathbf{R}$$

- Time derivative of orientation quaternion q=(x,y,z,w) is

$$\dot{\mathbf{q}} = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{w} \end{pmatrix} = \frac{1}{2} \begin{pmatrix} +w & -z & +y \\ +z & +w & -x \\ -y & +x & +w \\ -x & -y & -z \end{pmatrix} \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} = \frac{1}{2} \mathbf{Q} \boldsymbol{\omega}$$

  - Q is 4x3 “quaternion matrix”

Center of Mass

- Consider rigid body as a collection of particles with their positions ρ_i and masses m_i
- Center of mass "c" is a weighted average of all particles

$$\mathbf{c} = \frac{\sum m_i \mathbf{p}_i}{\sum m_i} = \frac{\sum m_i \mathbf{p}_i}{M}$$

  - where $M = \Sigma m_i$ is total mass of body
- Relative position r_i of i-th particle satisfies $\rho_i = c + r_i$
- Current i-th particle position is $\rho_i = c + R\, r_{0i}$
  - R is current rotation matrix of body
  - r_{0i} is initial local-space position of i-th particle

Linear and Angular Momentum

- Assuming each particle has its own mass m_i and velocity $u_i = \omega \times r_i + v$, we define its linear momentum "P_i" and i-th angular momentum "L_i" as
  - $P_i = m_i u_i$
  - $L_i = r_i \times P_i = m_i r_i \times u_i$
- Summing up Pi and Li over all particles we get total linear momentum "P" and angular momentum "L"
- $P = \Sigma P_i = \Sigma m_i u_i = \Sigma m_i (\omega \times r_i + v) = \ldots = M v$
- $L = \Sigma L_i = \Sigma m_i r_i \times u_i = \Sigma m_i r_i \times (\omega \times r_i + v) = \ldots = J \omega$
  - where matrix J is the current inertia tensor

Mass and Inertia Tensor

- Total mass M and inertial tensor J are defined as

$$M = \sum m_i$$

$$\mathbf{J} = -\sum m_i \mathbf{r}_i^{\times} \mathbf{r}_i^{\times} = \sum m_i \begin{pmatrix} \mathbf{r}_{iy}^2 + \mathbf{r}_{iz}^2 & -\mathbf{r}_{ix}\mathbf{r}_{iy} & -\mathbf{r}_{ix}\mathbf{r}_{iz} \\ -\mathbf{r}_{iy}\mathbf{r}_{ix} & \mathbf{r}_{ix}^2 + \mathbf{r}_{iz}^2 & -\mathbf{r}_{iy}\mathbf{r}_{iz} \\ -\mathbf{r}_{iz}\mathbf{r}_{ix} & -\mathbf{r}_{iz}\mathbf{r}_{iy} & \mathbf{r}_{ix}^2 + \mathbf{r}_{iy}^2 \end{pmatrix}$$

  - Unlike scalar mass M, inertia tensor J is time dependent
- Initial inertia is $J_0 = -\Sigma\, m_i\, r_{0i}^{\times}\, r_{0i}^{\times}$
  - Bodies never deform, thus current inertia can be expressed in terms of initial inertia J_0 and current rotation matrix R
- $J = RJ_0R^T$ and $J^{-1} = RJ^{-1}_0R^T$

Mass and Inertia Tensor

- J_1 = Inertia tensor of sphere with radius r and mass m
- J_2 = Inertia tensor of solid box with mass m and width w, height h and depth d

$$\mathbf{J_1} = \begin{pmatrix} \frac{2mr^2}{5} & 0 & 0 \\ 0 & \frac{2mr^2}{5} & 0 \\ 0 & 0 & \frac{2mr^2}{5} \end{pmatrix} \qquad \mathbf{J_2} = \begin{pmatrix} \frac{m}{12}(h^2+d^2) & 0 & 0 \\ 0 & \frac{m}{12}(w^2+d^2) & 0 \\ 0 & 0 & \frac{m}{12}(w^2+h^2) \end{pmatrix}$$

Mass and Inertia Tensor

- Translated inertia tensor by offset r is
- $J = J_0 + m(r^T r\, \mathbf{1} - r r^T)$
  - where 1 is 3x3 identity matrix and r is a column vector, ie. transposed $r^T = (r_x, r_y, r_z)$ is row vector, thus
  - $r^T r$ (inner or dot product) is scalar
  - $r r^T$ (outer product) is a 3x3 matrix
- Given body with n solid parts with mass m_i, center of mass c_i and inertia tensor J_{0i}, total body
  - Mass $m = \Sigma\, m_i$
  - Inertia $J = \Sigma\, J_i = \Sigma\, (J_{0i} + m_i(c_i^T c_i\, \mathbf{1} - c_i c_i^T))$
  - Center of mass $c = (\Sigma\, m_i c_i) \,/\, (\Sigma\, m_i)$

Linear and Angular Acceleration

- The time derivative of inertia J (and J^{-1}) is
- $J' = (RJ_0R^T)' = R'J_0R^T + RJ_0R'^T = \ldots = \omega^\times J - J\,\omega^\times$
- $J'^{-1} = (RJ^{-1}{}_0R^T)' = R'J^{-1}{}_0R^T + RJ^{-1}{}_0R'^T = \ldots = \omega^\times J^{-1} - J^{-1}\,\omega^\times$
- Linear acceleration "a" is defined as
- $a = v' = (M^{-1}P)' = M^{-1}P' = M^{-1}f$
  - Where f is force - time derivative of linear momentum P
- Angular acceleration "α" is defined as
- $\alpha = \omega' = (J^{-1}L)' = J'^{-1}L + J^{-1}L' = \ldots = 0 - J^{-1}\omega^\times J\omega + J^{-1}\tau$
  - Where τ is torque - time derivative of angular momentum L

Force and Torque

- Force fi and torque τi of i-th particle are
- $f_i = m_i a_i$ (i-th force)
- $\tau_i = r_i \times f_i = m_i r_i \times a_i$ (i-th torque)
- Summing up over all particles we get the famous Newton-Euler equations for total force and torque
- $f = \Sigma f_i = \Sigma m_i a_i = \ldots = M v' = P'$
- $\tau = \Sigma \tau_i = \Sigma m_i r_i \times a_i = \ldots = J\omega + \omega^{\times} J\omega = \ldots = L'$

Summary of Rigid Body Concepts

- We can summarize main physical properties (quantities) of rigid bodies as either
  - Kinematical (pure geometrical, mass "independent")
  - Dynamical (physical, mass "dependent")

	Kinematical Properties		Dynamical Properties	
lin	Position	$\mathbf{c}(t) \in \mathbf{R}^{3x1}$	Mass	$M \in \mathbf{R}^{1x1}$
ang	Orientation	$\mathbf{q}(t) \in \mathbf{R}^{4x1}$	Inertia Tensor	$\mathbf{J}(t) \in \mathbf{R}^{3x3}$
lin	Linear velocity	$\mathbf{v}(t) \in \mathbf{R}^{3x1}$	Linear Momentum	$\mathbf{P}(t) \in \mathbf{R}^{3x1}$
ang	Angular velocity	$\boldsymbol{\omega}(t) \in \mathbf{R}^{3x1}$	Angular Momentum	$\mathbf{L}(t) \in \mathbf{R}^{3x1}$
lin	Linear acceleration	$\mathbf{a}(t) \in \mathbf{R}^{3x1}$	Force	$\mathbf{f}(t) \in \mathbf{R}^{3x1}$
ang	Angular acceleration	$\boldsymbol{\alpha}(t) \in \mathbf{R}^{3x1}$	Torque	$\boldsymbol{\tau}(t) \in \mathbf{R}^{3x1}$

Rigid Body Equation of Motion

- The rigid body equation of unconstrained motion can be summarized as the following ODE

$$\frac{d}{dt}\mathbf{x}(t) = \frac{d}{dt}\begin{pmatrix}\mathbf{c}(t)\\ \mathbf{q}(t)\\ \mathbf{P}(t)\\ \mathbf{L}(t)\end{pmatrix} = \begin{pmatrix}\mathbf{v}(t)\\ \frac{1}{2}\mathbf{Q}(t)\boldsymbol{\omega}(t)\\ \mathbf{f}(t)\\ \boldsymbol{\tau}(t)\end{pmatrix}$$

- Where auxiliary variables are

$$\mathbf{Q}(t) = \begin{pmatrix} +\mathbf{q}_w(t) & -\mathbf{q}_z(t) & +\mathbf{q}_y(t)\\ +\mathbf{q}_z(t) & +\mathbf{q}_w(t) & -\mathbf{q}_x(t)\\ -\mathbf{q}_y(t) & +\mathbf{q}_x(t) & +\mathbf{q}_w(t)\\ -\mathbf{q}_x(t) & -\mathbf{q}_y(t) & -\mathbf{q}_z(t)\end{pmatrix}$$

$$\begin{aligned}\mathbf{v}(t) &= M^{-1}\mathbf{P}(t)\\ \boldsymbol{\omega}(t) &= \mathbf{J}^{-1}(t)\mathbf{L}(t)\\ \mathbf{J}^{-1}(t) &= \mathbf{R}(t)\mathbf{J}_0^{-1}\mathbf{R}^{\mathrm{T}}(t)\end{aligned}$$

User and Time control

User and Time control

- According to the time control of the simulation, we can split the overall simulation process into three nested layers
  - The Presentation Layer
  - The Collision Layer
  - The Simulation Layer.

Time control: Presentation Layer

- From users point-of-view the overall simulation must be present (rendered) in a sequence of animation frames
- The size of the frame is obviously application dependent:
- In time-critical and interactive applications (VR) it is usually fixed and defined by the user/device (min. 25 frames per seconds)
- In large, complex offline simulations it can vary upon the computational expenses

Time control: Collision Layer

- Within each frame the motion solver perform some sub-steps to advance the motion correctly.
- Due to collision and constraint resolution discontinuities arise in the motion
- Depending on the time of collision detection (resolution) the number (size) of "collision steps" can be fixed or adaptive
- When handling multiple penetrating objects in one step fixed time stepping is usually suitable
- If only one collision is resolved at once adaptive time stepping technique should be used

Backtracking Approach

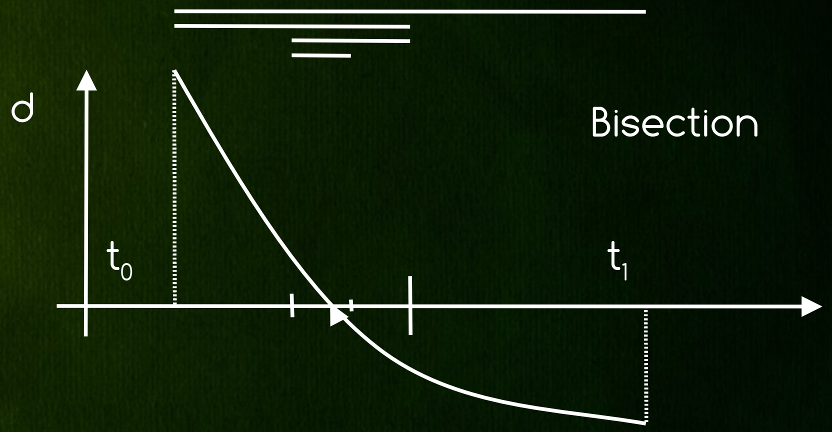
- We want to advance the simulation form t_0 to t_1
- Use bisection to find the first collision occurrence
  - First check for collisions at t_1 , next at mid time $t_m = 0.5(t_0+t_1)$
  - If there is some collision proceed similar back in (t_0,t_m)
  - Otherwise proceed in second half interval (t_m, t_1)
  - Proceed similar until desired number of iterations
- if we know the time derivative of the separation distance the search can be even faster
- It is simple, robust, can have slow convergence and tunneling problem (some collisions are missed)

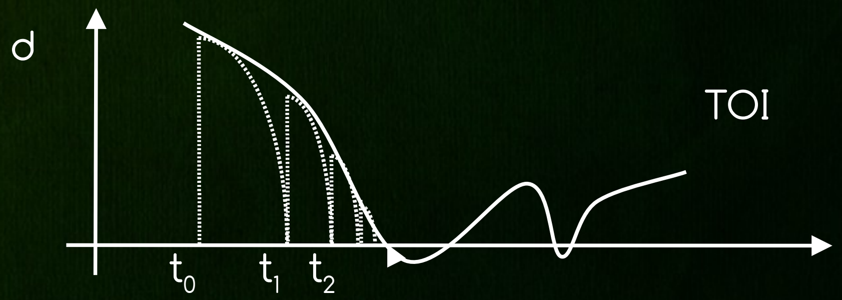
One-Side Approach

- The One-Side Approach is a more conservative technique. We always advance the simulation forward in time.
  - This is possible, since between collisions objects move along ballistic trajectories and we can estimate the lower bound of their Time of Impact (TOI)
- Given upper bounds on angular and linear velocities we can estimate maximal translation of any surface point (on both estimated bodies) w.r.t. some direction axis d
- Find earliest time when bodies may penetrate. If no collision occurs, we advance bodies

User and Time control

- During both methods full collision detection is performed on estimated times
- Alternative solution is to use continuous collision detection





Time control: Simulation Layer

- Within each "collision" step the motion solver must integrate the motion equation
- Numerical ODE solver usually requires several integration steps to achieve desired accuracy and stability
- Again we can choose a fixed or adaptive time stepping scheme


The
End