

Fundamental models

- 1- Particles
- 2- Rigid bodies**
- 3- Continuum models

Rigid body description

- Solid defined within a domain $\Omega \subset \mathbb{R}^3$
- With a density of mass $\rho(p_i)$ at each point $p_i \in \Omega$

- Total mass of the solid m

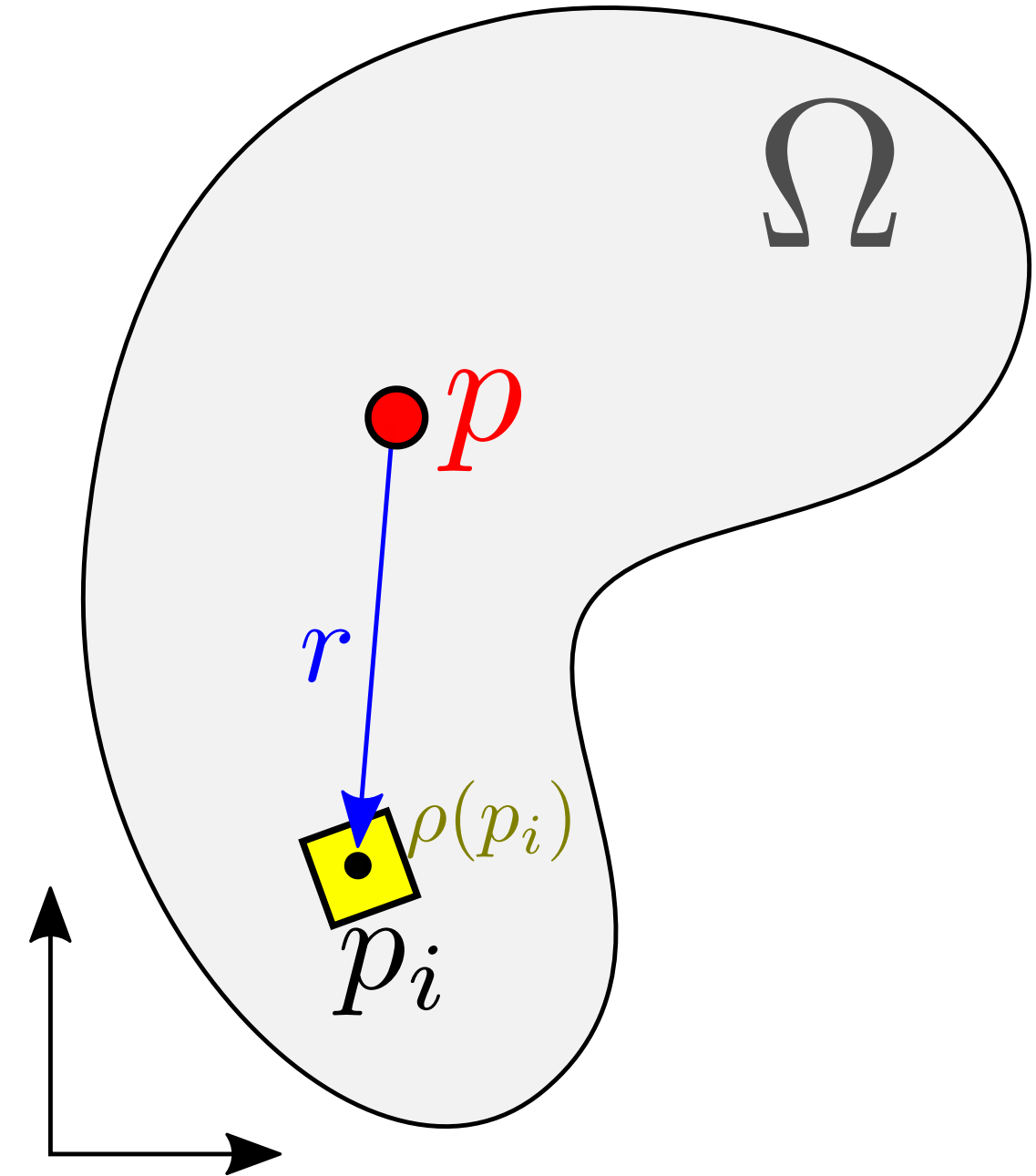
$$m = \int_{p_i \in \Omega} \rho(p_i) d\Omega$$

- Position of the center of mass (com) p

$$p = \frac{1}{m} \int_{p_i \in \Omega} \rho(p_i) p_i d\Omega$$

- Relative position of a position p_i with respect to com

$$r = p_i - p$$



Position and speed of a point on the rigid body

The center of mass has, at time t ,

- a position $p(t)$
- a velocity $p'(t) = v(t)$

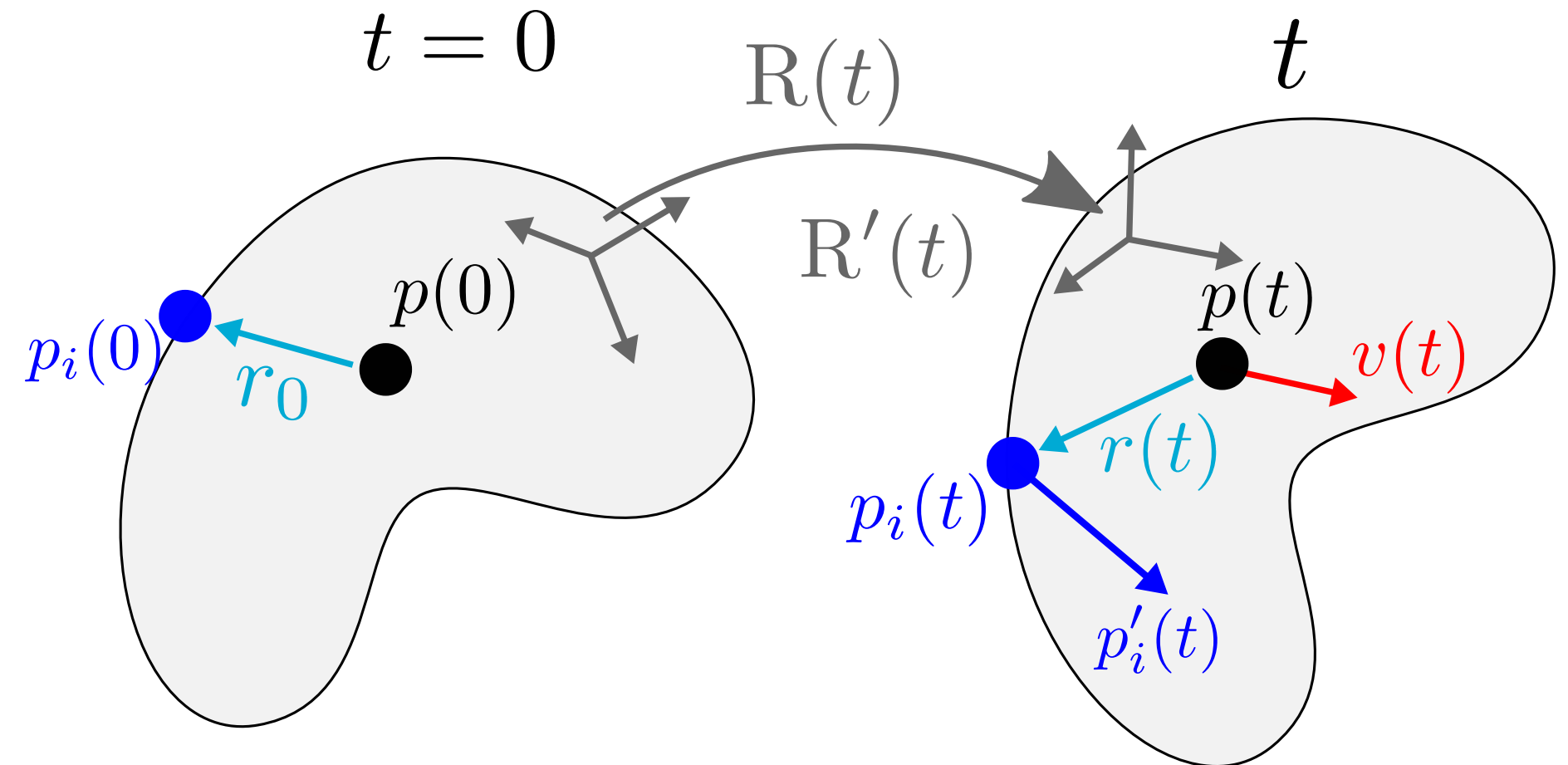
The body has an orientation $R(t)$

A point of the rigid body has

- a position $p_i(t) = p(t) + R(t) r_0$

with $r_0 = p_i(0) - p(0)$

- a speed $p'_i(t) = v(t) + R'(t) r_0$



Angular speed

Speed of p_i : $p'_i(t) = v(t) + R'(t) r_0$

Introduce angular speed $\omega \in \mathbb{R}^3$ such that

$$p'_i(t) = v(t) + \omega(t) \times r(t)$$

$\omega \simeq$ vector expressing the instantaneous rotation of $r(t)$

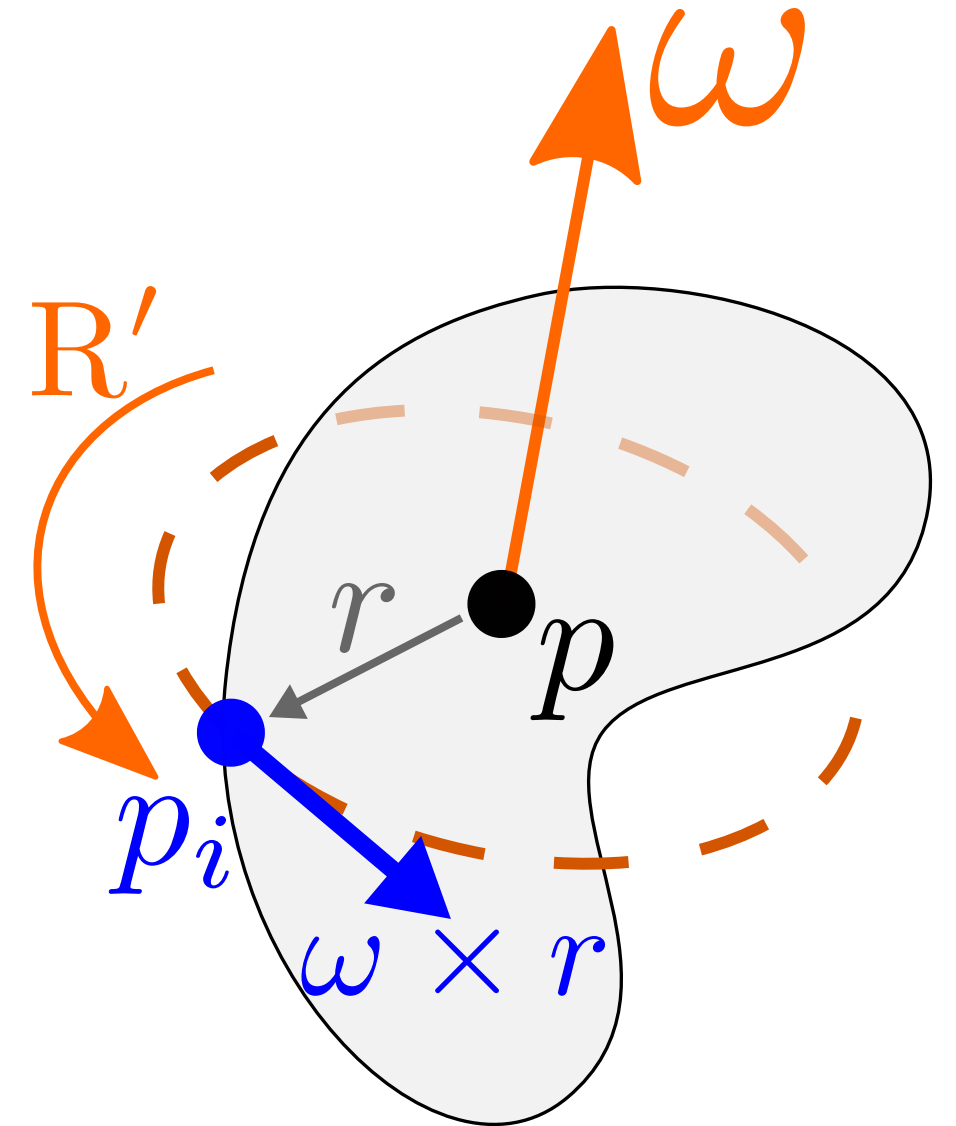
By identification $R'(t) r_0 = \omega(t) \times r(t)$

$$\Rightarrow R'(t) r_0 = \omega(t) \times (R(t) r_0)$$

Matrix expression of $\omega = (\omega_x, \omega_y, \omega_z)$

$$\hat{\omega} = \begin{pmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{pmatrix}$$

$$\Rightarrow R'(t) = \hat{\omega}(t) R(t)$$



Rigid body kinematics

Similarly to particles

- Position of the com: $p(t)$
- Velocity of the com: $v(t) = p'(t)$

Linear Momentum: $P(t) = m v(t)$ $\left(= \int_{\Omega} \rho v_i(t) d\Omega \right)$

Specific to rigid body

- Orientation of the body: $R(t) \in \mathbb{R}^{3 \times 3}$
- Angular velocity of the body: $\omega(t)$ such that $\hat{\omega} = R'(t) R^T(t)$

Angular Momentum: $L(t) = I(t) \omega(t)$

with $I(t)$: Inertia tensor

$$I(t) = R(t) I_0 R^T(t)$$

$$I_0 = \int_{r \in \Omega} \rho(r) (r^T r I_d - r r^T) d\Omega$$

Mass: resistance to change of speed (of the com)

Inertia: resistance to change of angular speed

Inertia tensor

Defining inertia tensor formulation from angular momentum definition

Angular momentum expressed with respect to an arbitrary point p_0 : $r(p_i) = p_i - p_0$

$$L = \int_{\Omega} r \times (\rho r') d\Omega = \int_{\Omega} \rho r \times (p' + \omega \times r) d\Omega \quad (\text{first part sum to } 0)$$

$$\Rightarrow L = \int_{\Omega} \rho r \times \omega \times r d\Omega = \int_{\Omega} \rho r \times (-r \times \omega) d\Omega$$

$$\Rightarrow L = \underbrace{\left(\int_{\Omega} \rho \hat{r} \hat{r}^T d\Omega \right)}_I \omega = I \omega \quad \text{with } \hat{r} = \begin{pmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{pmatrix}$$

Changing the reference frame

$$\Rightarrow L = \left(\int_{\Omega} \rho (R\hat{r}_0) (R\hat{r}_0)^T d\Omega \right) \omega = R \underbrace{\left(\int_{\Omega} \rho(r) \hat{r}_0 \hat{r}_0^T d\Omega \right)}_{I_0} R^T \omega = R I_0 R^T \omega$$

Inertia tensor properties

$$I = \int_{r \in \Omega} \rho(r) \begin{pmatrix} r_y^2 + r_z^2 & -r_x r_y & -r_x r_z \\ -r_x r_y & r_x^2 + r_z^2 & -r_y r_z \\ -r_x r_z & -r_y r_z & r_x^2 + r_y^2 \end{pmatrix} d\Omega = \int_{r \in \Omega} \rho(r) (r^T r \text{ Id} - r r^T) d\Omega$$

- I is usually expressed at the center of mass p
- I depends on the body orientation. Given a rotation R : $I = R I_0 R^T$
 $\Rightarrow$ compute once I_0 in a rest position, then update it using R
- There exist a frame in which I is diagonal (principle axes of inertia).
Corresponds to eigenvectors of matrix I .

Dynamics: Forces and torques on a solid

- Given a local force $f(p_i)$ acting on a position $p_i \in \Omega$
- Contribute to 2 global components applied on the body:

- Total **net force** F applied on the shape

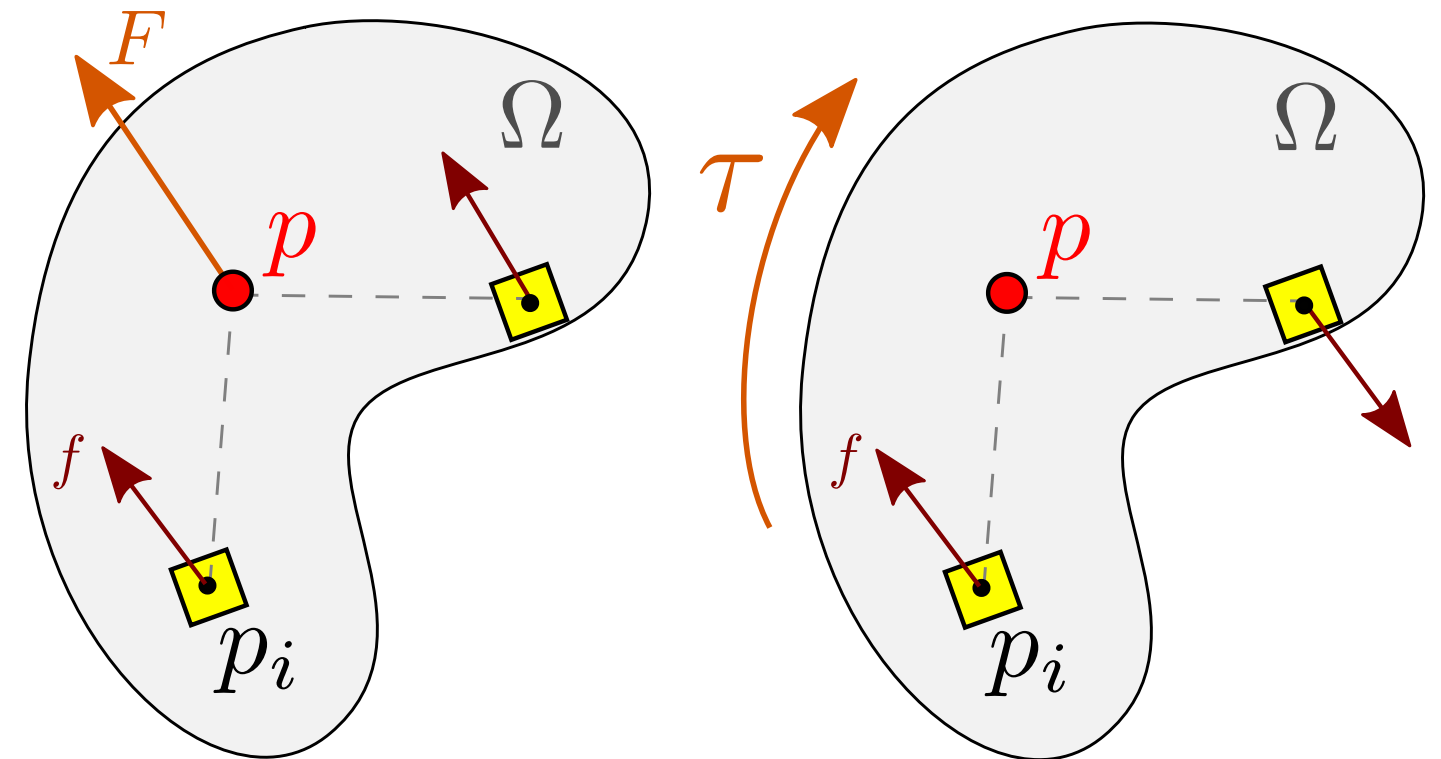
Induce change of linear momentum (a displacement of COM)

$$F = \int_{p_i \in \Omega} f(p_i) d\Omega$$

- **Torque** τ applied on the body

Induce change of angular momentum (spin of the solid)

$$\tau = \int_{p_i \in \Omega} (p_i - p) \times f(p_i) d\Omega$$



Dynamics

Similarly to particles

Force F is related to the change of linear momentum $F(t) = P'(t) = (m v)'(t)$

Specific to rigid bodies

Torque τ is related to the change of angular momentum $\tau(t) = L'(t) = (I \omega)'(t)$

Equation of Motion

Fundamental principle of dynamics for rigid body

$$\frac{d}{dt} \begin{pmatrix} p \\ P \\ R \\ L \end{pmatrix} = \begin{pmatrix} v \\ F \\ \hat{\omega} R \\ \tau \end{pmatrix}$$

Rigid solid dynamics in practice

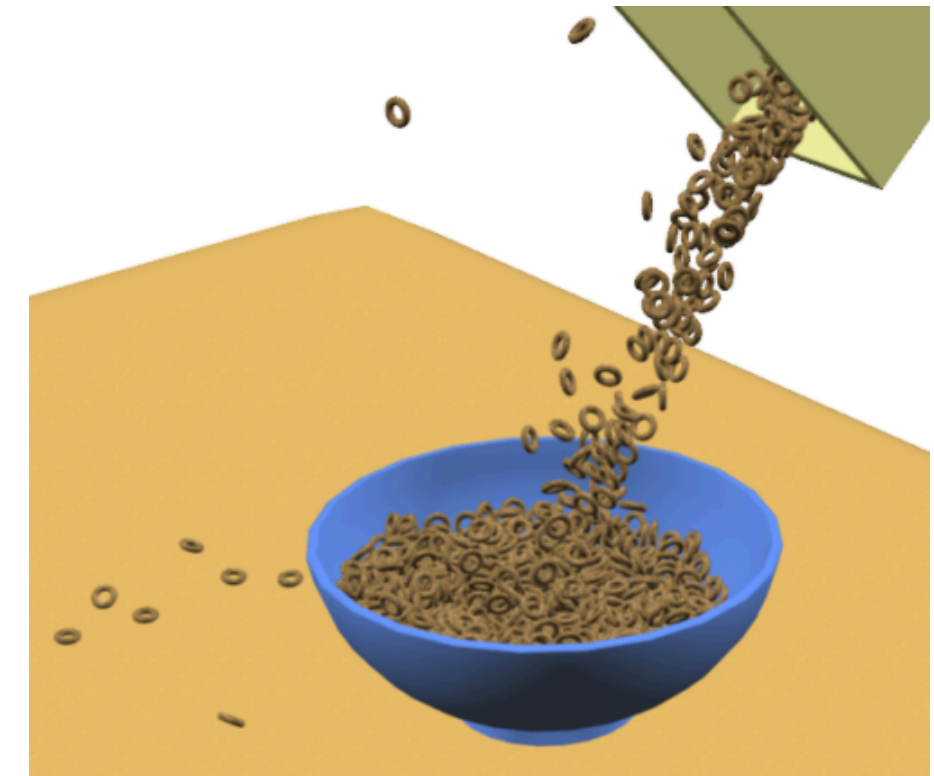
1. Initial condition

- $p(t_0), v(t_0), R(t_0), \omega(t_0)$ given as initial condition
- Precompute $I_0 = I(t_0)$
- Compute $L(t_0) = I_0 \omega(t_0)$

2. Temporal Evolution

Iterate over time t_k

- Compute total force $F(t_k)$ and torque $\tau(t_k)$
- Compute $I(t_k) = R(t_k) I_0 R^T(t_k)$
- Compute $\omega(t_k) = I(t_k)^{-1} L(t_k)$
- Numerical integration updating state vector
 $\rightarrow (p(t_{k+1}), P(t_{k+1}), R(t_{k+1}), L(t_{k+1}))$
- *Handle collision*



$$\begin{pmatrix} p'(t) \\ P'(t) \\ R'(t) \\ L'(t) \end{pmatrix} = \begin{pmatrix} v(t) \\ F(t) \\ \hat{\omega}(t) R(t) \\ \tau(t) \end{pmatrix}$$

$$I(t) = R(t) I_0 R^T(t)$$

$$L(t) = I(t) \omega(t)$$

$$P(t) = m v(t)$$

Collisions in rigid bodies

Collisions at position p change both linear and angular velocity

Use of impulse (sudden change of velocity) J .

$$J = \int_{t_0}^{t_0+\Delta t} F(t) dt$$

Impulse split into

- Force impulse $\Delta P = m \Delta v = J$

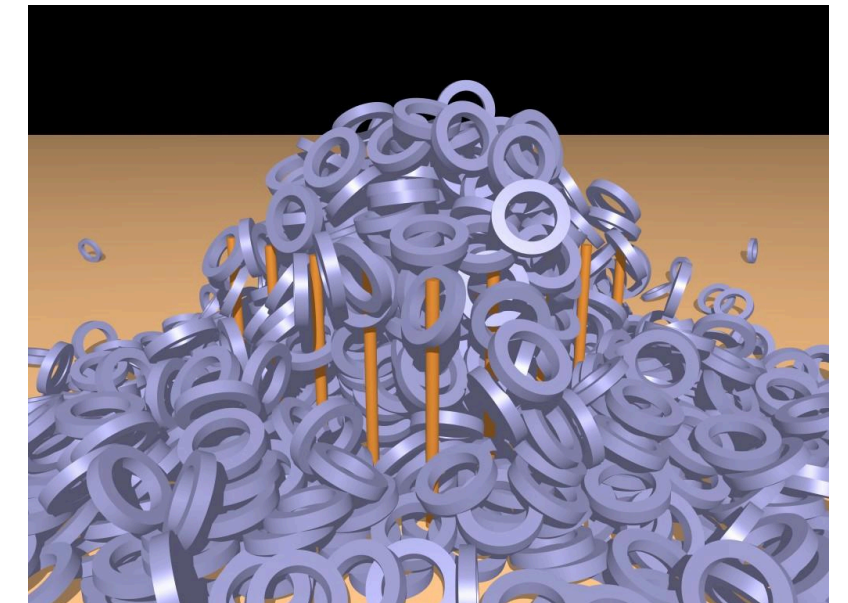
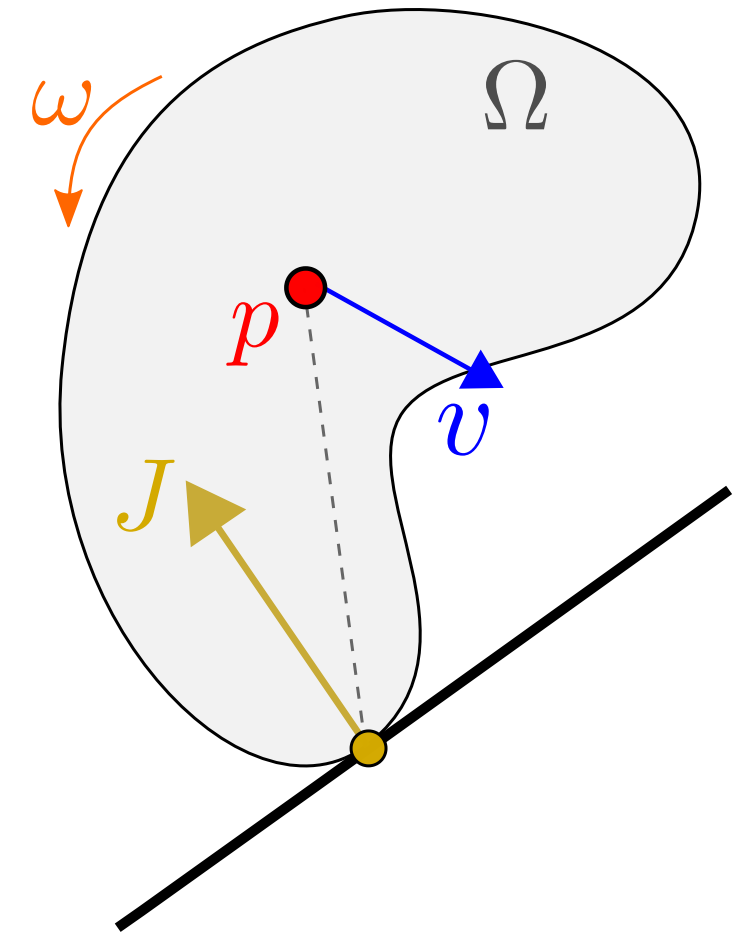
- Torque impulse $\Delta L = I \Delta \omega = (p - p_{com}) \times J$

Elastic collision between two solids at position p_i :

$J = j n$, n : normal of the separating planes

$$j = (v_1(p_i) - v_2(p_i)) \cdot n / K$$

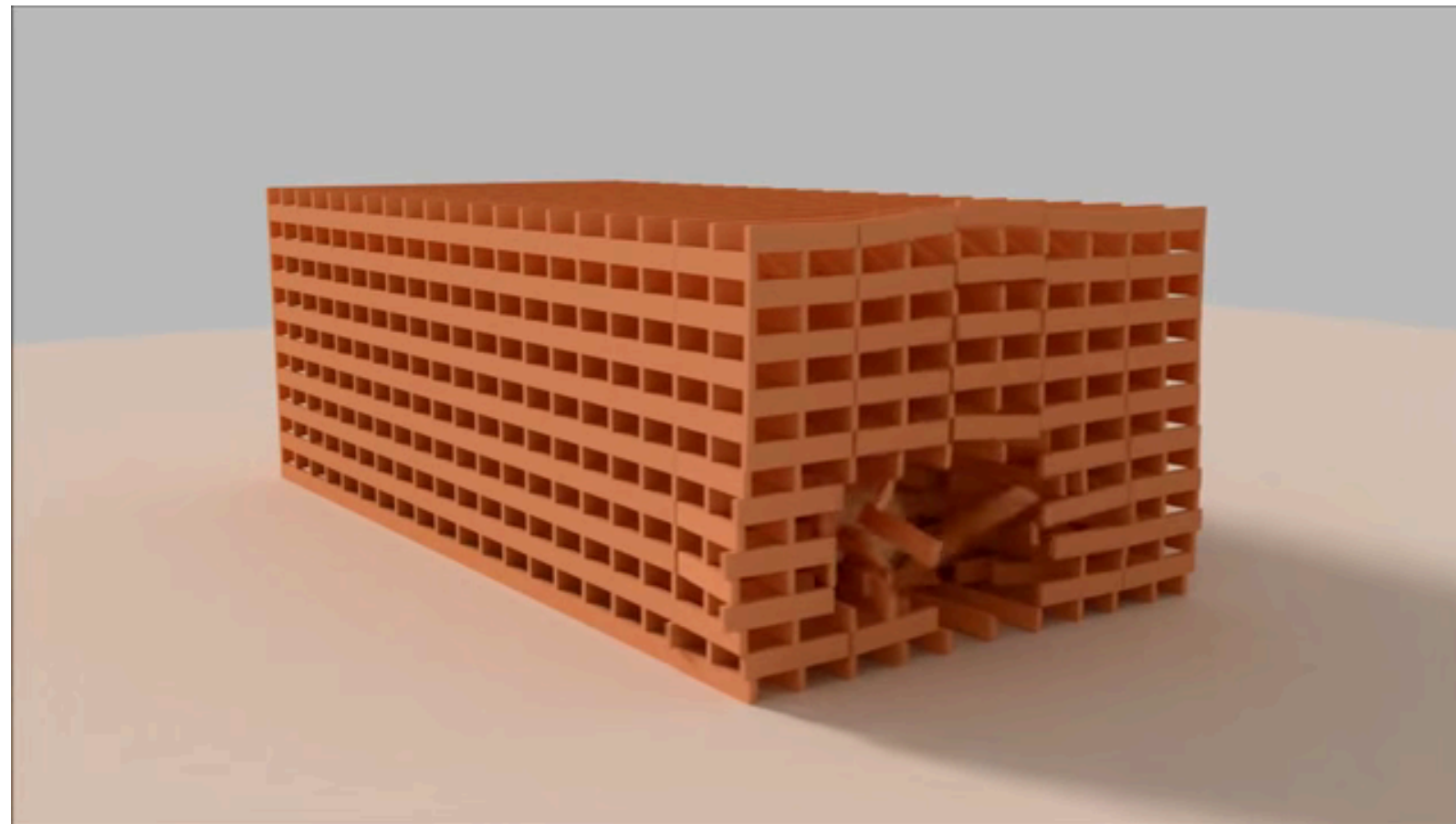
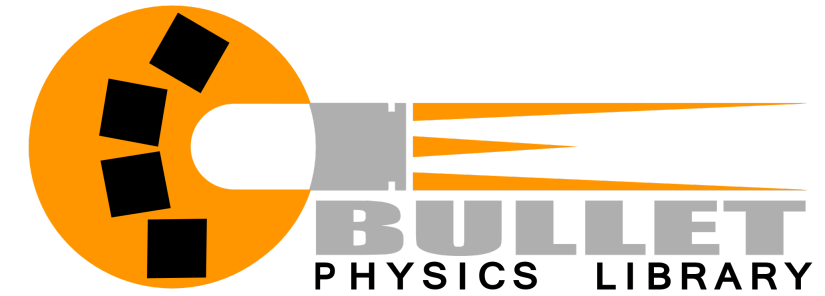
$$K = 1/m_1 + 1/m_2 + n \cdot (I_1^{-1}(r_1 \times n) \times r_1 + I_2^{-1}(r_2 \times n) \times r_2)$$



More details [\[D. Barff. Physically Based Modeling. SIGGRAPH Course Notes 1999\]](#)

Rigid bodies usage

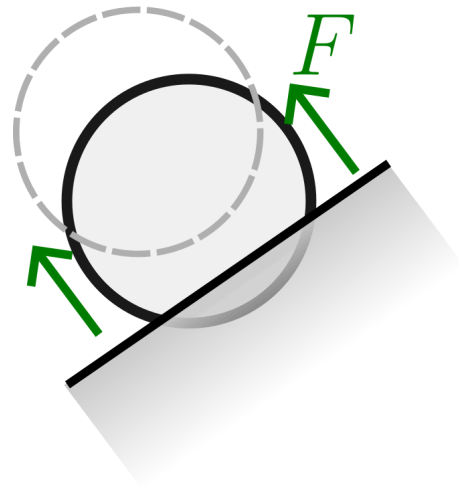
- Standard usage for rigid bodies motions
 - Limited to non-deformable shapes
- Common in VFX (explosions), and *simulation games* (cars, airplanes, etc).
- Standard library: [Bullet physics](#) (ex. used in Blender).



Position Based Dynamics (PBD)

Constraint handling

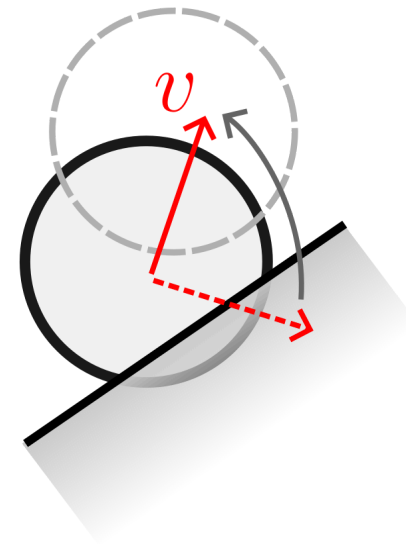
Force based



$$v_i^{k+1} = v_i^k + F_i/m_i \Delta t$$

- (+) Physical interpretation
- (+) Arbitrary constraint
- (-) Constraint stiffness: Divergence

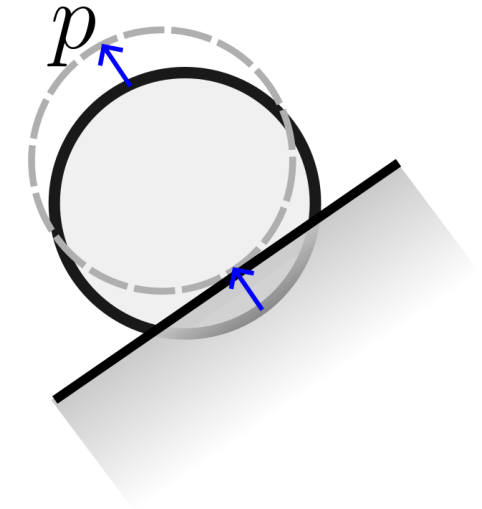
Impulse based



$$v_i^{k+1} = v_i^k + J/m$$

- (+) Stability
- (-) Collision only
- (-) Doesn't correct existing collisions

Position based



$$p_i^{k+1} \rightarrow \mathcal{F}(p^k)$$

- (+) Stability
- (+) Arbitrary constraint
- (-) Physical interpretation

PBD: General Idea

Handle positional constraints by moving vertex positions directly

Velocity computed as change of position

(+) Very simple, unconditionally stable

Formalized in 2005

[Matthias Muler, Bruno Heidelberger, Marcus Hennix, John Racliff. *Position based Dynamics*. PRIPHYS 2005.]

PBD simulation step

1) Integrate *standard forces* to predict position

$$p_{\text{prev}} = p$$

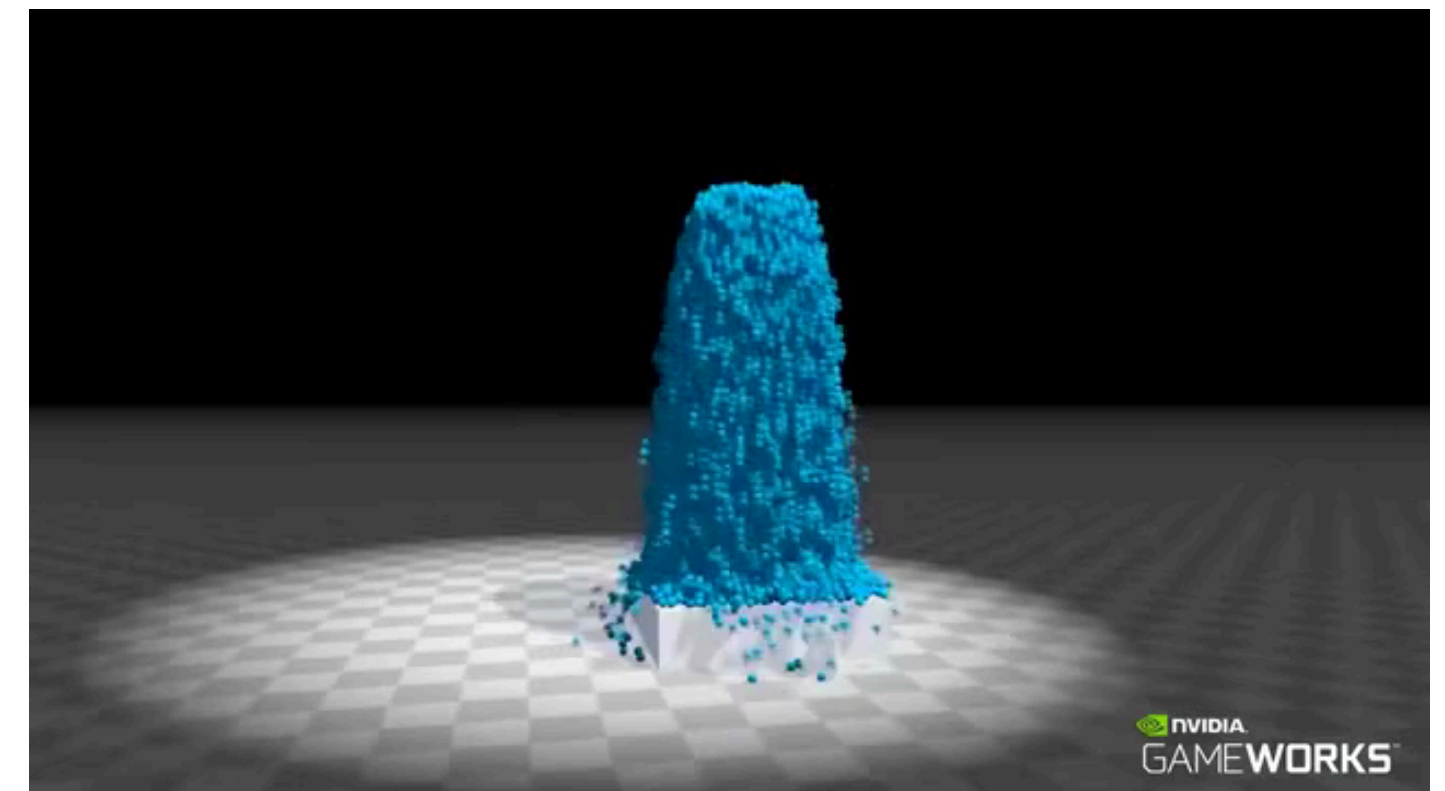
$$v = v + \Delta t F / m$$

$$p = p + \Delta t v$$

2) Project p on constraints

3) Update velocity

$$v = (p - p_{\text{prev}}) / \Delta t$$



PBD: Generic Non Linear Constraints

Handling arbitrary non constraint $C(p) > 0$ over $p = (p_1, \dots, p_N)$

Solve iteratively $C(p + dp) \simeq C(p) + \nabla C \cdot dp > 0$ using Lagrange multiplier

$$dp = \lambda M^{-1} (\nabla C)^T, M = \text{diag}(m_1, \dots, m_N): \text{masses}$$

$$\Delta p_i = -\lambda / m_i \nabla_{p_i} C(p)^T$$

$$\Rightarrow \lambda = \frac{C(p)}{\sum_i 1/m_i \|\nabla_{p_i} C(p)\|^2}$$

Note: Converges to infinite stiffness

Proposition of eXtended PBD (XPBD)

$$\Rightarrow \lambda = \frac{C(p)}{\sum_i 1/m_i \|\nabla_{p_i} C(p)\|^2 + \frac{\alpha}{(\Delta t)^2}}$$

$\alpha = 1/K$: compliance (inverse stiffness)

[Miles Macklin, Matthias Muller, Nuttapong Chentanez. XPBD: Position-Based Simulation of Compliant Constrained Dynamics. MIG 2016]

[Jan Bender, Matthias Muller, Miles Macklin. A survey on Position Based Dynamics. EUROGRAPHICS STAR 2017]

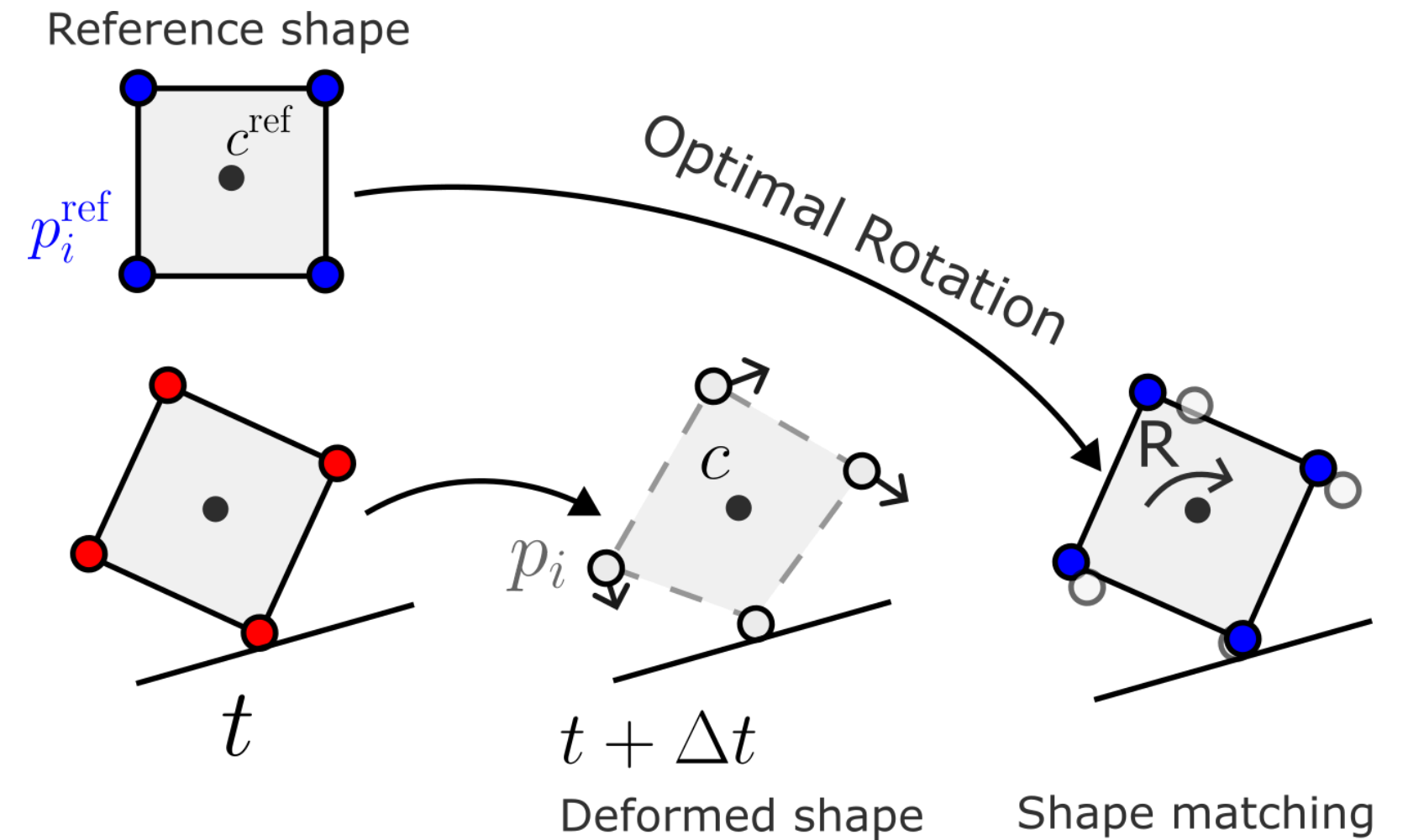
Note: Formulation may vary a bit depending on the paper.

Rigid bodies dynamic: Shape Matching

General Idea:

- 1) Consider a reference shape.
- 2) At time t , deform vertices individually
 $t \rightarrow t + \Delta t$
Solve for the collision constraints on each vertex
- 3) Shape Matching:
Compute optimal rotation R
between reference shape and current one.
Apply R on the reference shape (+ translate it).

- (+) Guarantee to preserve reference shape appearance
(-) May not preserve individual constraints



Shape Matching Formulation

Reference Shape:

Vertex position p_i^{ref}

Center of mass c^{ref}

Local vertex vector $r_i^{\text{ref}} = p_i^{\text{ref}} - c^{\text{ref}}$

Deformed shape at $t + \Delta t$

Vertex position p_i

Center of mass c

Local vertex vector $r_i = p_i - c$

Averaged transformation $T = \sum_i r_i (r_i^{\text{ref}})^T$

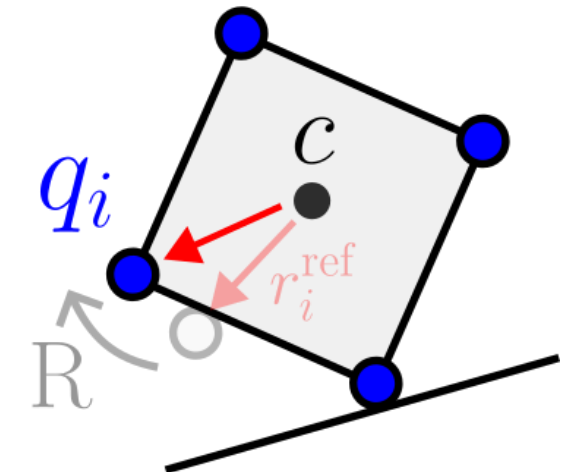
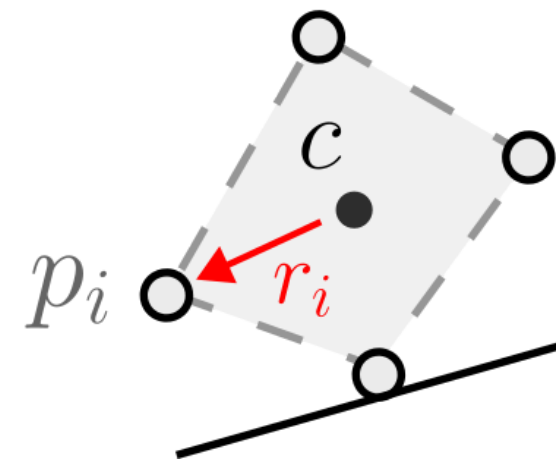
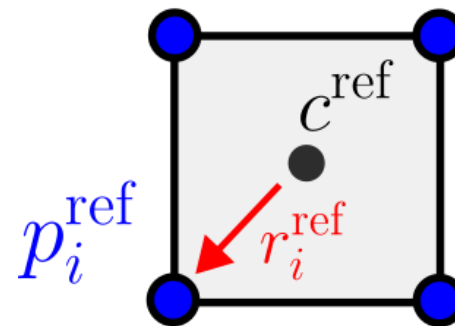
T minimizes $\sum_i \|T r_i^{\text{ref}} - r_i\|^2$

Extract optimal rotation R

via polar decomposition from T

New position $q_i = R r_i^{\text{ref}} + c$

Reference shape



Reminder Polar Decomposition: $R = WV^T$, for $svd(T) = W \Sigma V^T$

Polar Decomposition

Goal: Decompose a 3×3 matrix M into

- rotation part R
- scaling/shearing D

ex. Usefull for interpolation of transformations.

- Interpolate rotation with quaternion (ex. SLERP / LERP)
- Scaling/shearing using componentwise interpolation (ex. linear).

⇒ Polar decomposition: $M = R D$

- R : Rotation matrix
- D : Positive semi-definite matrix

[K. Shoemake and T. Duff, Matrix Animation and Polar Decomposition. Graphics Interface 92.]

- Polar decomposition is obtained from SVD

$$\text{SVD}(M) = W \Sigma V^T \text{ with } R = W V^T, \quad D = V \Sigma V^T$$

- Or numerically, R can be computed using the following iterative scheme

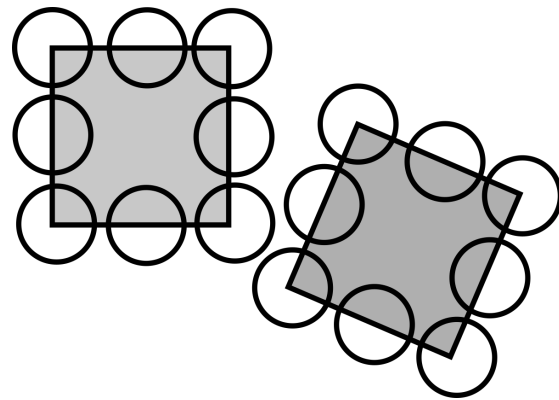
$$R_0 = M, \quad R_{i+1} = 0.5 (R_i + (R_i^{-1})^T)$$



Shape Matching Usage

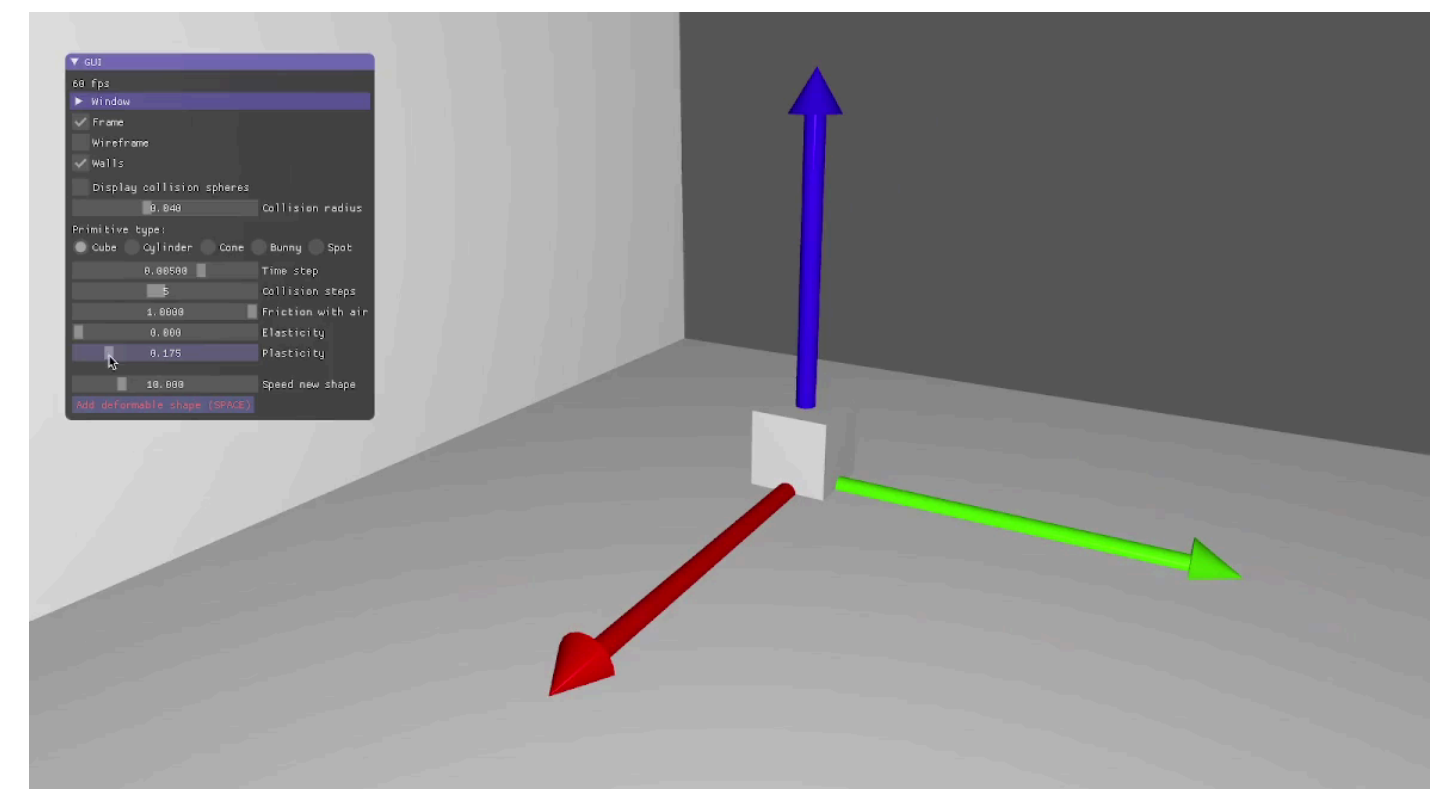
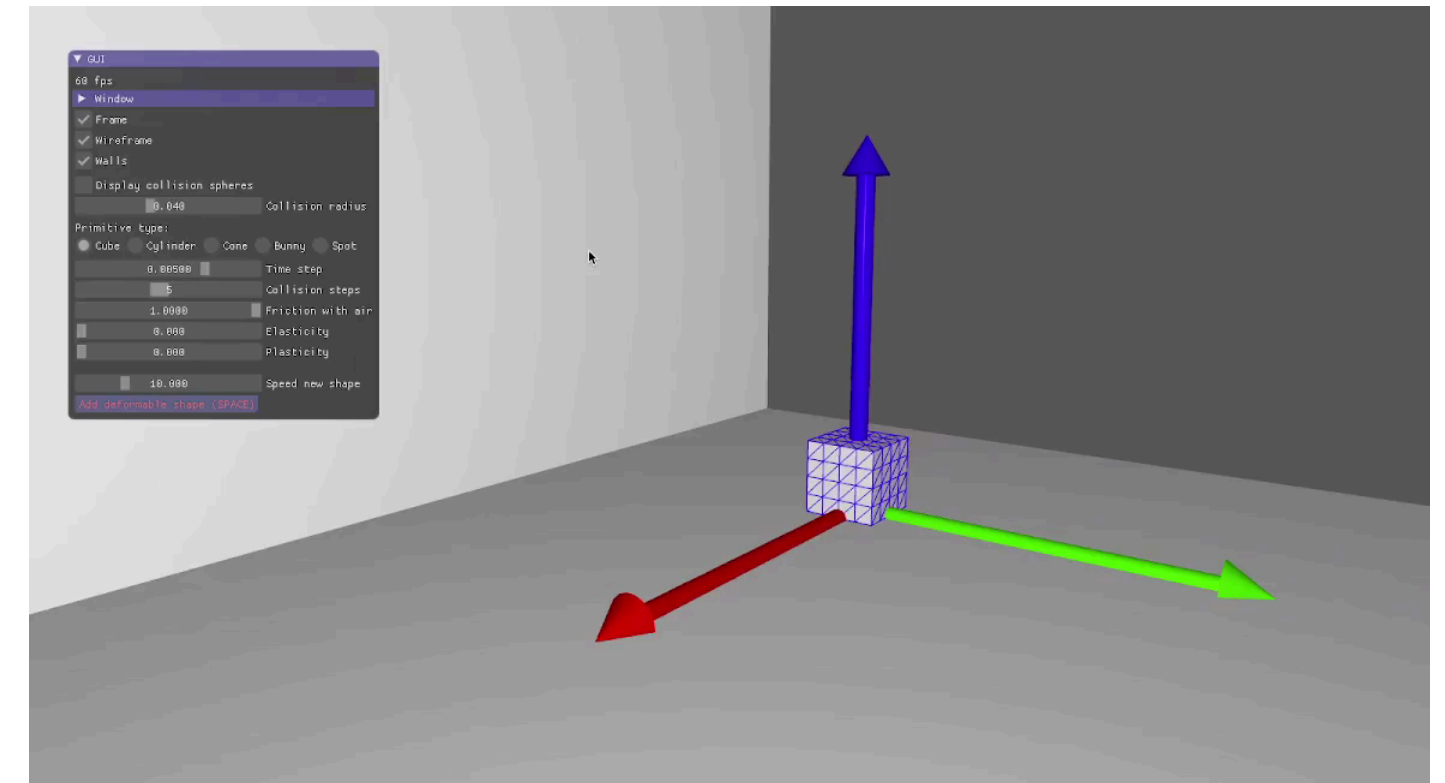
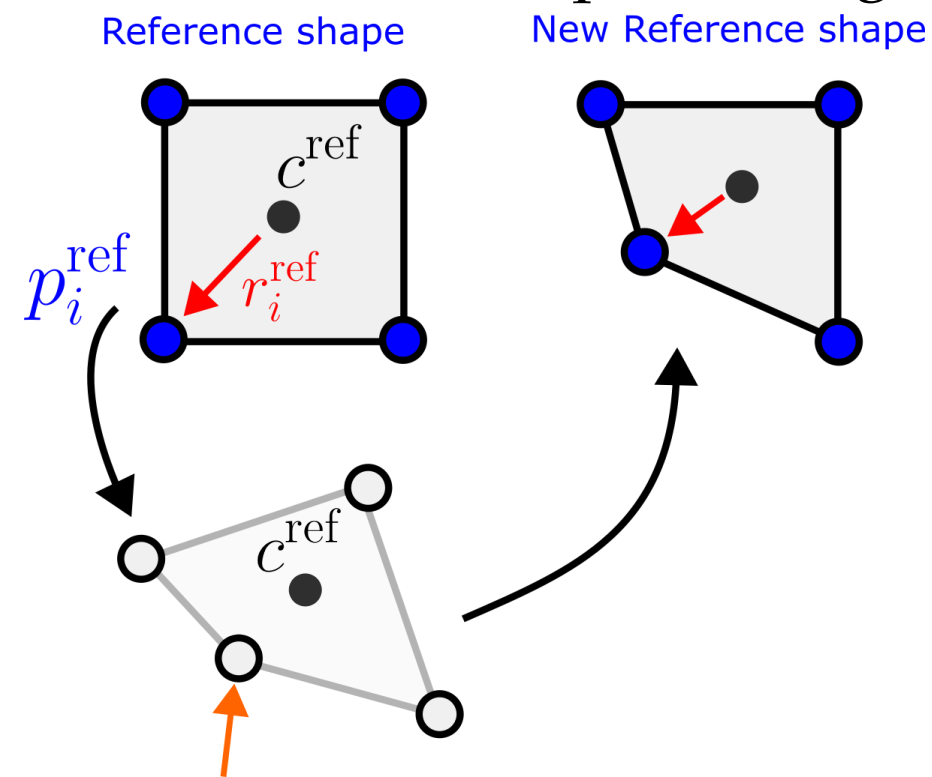
Rigid bodies

Collision b/w bodies using colliders around vertices



Plastic deformation

Update the reference shape for large deformation



[Muller et al. Meshless Deformations Based on Shape Matching. SIGGRAPH 2005]

[Macklin et al. Unified Particle Physics for Real-Time Applications. SIGGRAPH 2014]