The Affine Particle-In-Cell Method

Chenfanfu Jiang et al.
SIGGRAPH 2015

Presented by Ji-su Ban
2019. 06. 27

Computer Graphics @ Korea University
Abstract

• This APIC method retains the stability of the original PIC, without suffer from the dissipation of the original PIC, also without the noise and instability of FLIP as well.

• The primary observation is that removing the dissipation in the original PIC results from a loss of information when transferring between grid and particle representations.
Introduction (1/2)

- Where is the beginning of the problem?
  - The hybrid Lagrangian/Eulerian method such as FLIP is used to solve some problem easily.
  - But transferring between particle and grid creates error.
Introduction (2/2)

• What is the focus of the paper?
  ▪ How that error can be minimized with minimal effort.
  ▪ **How the transfer between grid and particle is done.**
  ▪ In this paper, it controls noise by keeping the pure filter property of PIC but minimizing information loss by enriching each particle with a 3x3 matrix (locally affine).

• What are the advantages of this Affine Particle-In-Cell method?
  ▪ Reduce dissipation effectively.
  ▪ Preserve angular momentum.
  ▪ Prevent instabilities.
In fluid simulation,

<table>
<thead>
<tr>
<th>Eulerian</th>
<th>Lagrangian</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pressure and viscosity updates</td>
<td>advection</td>
</tr>
</tbody>
</table>

“The particle-in-cell method for numerical solution of problems in fluid dynamics.”
- [Harlow and Francis H(Univ. California) / Meth Comp Phys 1964]
  - Particle-In-Cell
  - The first and simplest method of that type.
  - To transfer all information of the velocities that computed in grid over particles by interpolating directly.

<table>
<thead>
<tr>
<th>Advantage</th>
<th>Disadvantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Effective and simple to implement</td>
<td>Suffer from dissipation</td>
</tr>
</tbody>
</table>

Result from frequent particle/grid transfers
Previous work

FLIP

- “FLIP: a method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions.”
  - [Brackbill and Ruppel (Los Alamos National Lab) / Computational Physics 1986]
  - The paper addressed the problem of dissipation in PIC.
  - To transfer increments of velocities and displacements from grid to particles.

<table>
<thead>
<tr>
<th>Advantage</th>
<th>Disadvantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reduce the dissipation by a small offset (increments)</td>
<td>Suffer from a loss of information</td>
</tr>
</tbody>
</table>

Result from more particles than grid nodes. Some particle modes are not seen by the grid and get no physical response.
Previous work

PIC/FLIP (Blending)

• “Animating sand as a Fluid”
  ▪ [Zhu and Bridson (Univ. British Colombia) / SIGGRAPH 2005]
    • Blend between pure PIC and FLIP to stabilize the simulator.
    • $u_{\text{new}}^{\text{particle}} = \alpha \cdot \text{interpolate}(u_{\text{new}}^{\text{grid}}, x_p) + (1 - \alpha)[u_{\text{old}}^{\text{particle}} + \text{interpolate}(\Delta u_{\text{grid}})]$

<table>
<thead>
<tr>
<th>Advantage</th>
<th>Disadvantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>More stable and reduce dissipation</td>
<td>Manual tuning of the blend weights on case-by-case.</td>
</tr>
</tbody>
</table>
Previous work

Particle with angular momentum

• “Smoothed dissipative particle dynamics with angular momentum conservation”
  ▪ [Kathrin Muller(Theoretical Soft Matter and Biophysics) et al. / Journal of Computational Physics 2015]
  • Smoothed dissipative particle dynamics + angular momentum (SDPD+a)
  • Developed an SPH approach that augments particle with a sample of angular momentum.
  • Do not use a grid.
Method Outline

- The only difference between the method
  - How the transfer between grid and particles is done.
- RPIC (Rigid Particle-In-Cell): consider each particle with the angular momentum lost in the grid to particle transfer.
- APIC (Affine Particle-In-Cell): For non-rigid motion like shearing, particles are endowed with a full affine representation of the local grid data.
- Lagrangian coupling technique without APIC.
Notation

- **Subscripts** $p, q$ (particle) / $i, j$ (grid) / $a$ (MAC grid axis direction)
- **Superscripts** $n$ (the current time step), $n+1$ (the next time step)
- **Lowercase bold for vectors, uppercase bold for matrices.**
- **Exception of angular momentum**, $L_p^n$ denote a **vector**.
- **Locations** are $p$ (particle), $n$ (grid node), $f$ (MAC grid face center), or $g$ (global; does not live at any location in space).
- **Quantities are of type** $s$ (scalar), $v$ (vector), $m$ (matrix) or $t$ (rank $- 3$ tensor).

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Location</th>
<th>Type</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m_p^n$</td>
<td>$p$</td>
<td>$s$</td>
<td>mass</td>
</tr>
<tr>
<td>$x_p^n$</td>
<td>$p$</td>
<td>$v$</td>
<td>position</td>
</tr>
<tr>
<td>$v_p^n$</td>
<td>$p$</td>
<td>$v$</td>
<td>velocity</td>
</tr>
<tr>
<td>$F_p$</td>
<td>$p$</td>
<td>$m$</td>
<td>deformation gradient</td>
</tr>
<tr>
<td>$f_p$</td>
<td>$p$</td>
<td>$v$</td>
<td>force</td>
</tr>
<tr>
<td>$\omega_p^n$</td>
<td>$p$</td>
<td>$v$</td>
<td>angular velocity</td>
</tr>
<tr>
<td>$L_p^n$</td>
<td>$p$</td>
<td>$v$</td>
<td>angular momentum</td>
</tr>
<tr>
<td>$K_p^n$</td>
<td>$p$</td>
<td>$m$</td>
<td>inertia tensor</td>
</tr>
<tr>
<td>$B_p^n$</td>
<td>$p$</td>
<td>$m$</td>
<td>affine state</td>
</tr>
<tr>
<td>$C_p^n$</td>
<td>$p$</td>
<td>$m$</td>
<td>velocity derivatives</td>
</tr>
<tr>
<td>$c_{pa}$</td>
<td>$p$</td>
<td>$v$</td>
<td>velocity derivatives (MAC only)</td>
</tr>
<tr>
<td>$D_p^n$</td>
<td>$p$</td>
<td>$m$</td>
<td>inertia-like tensor</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Location</th>
<th>Type</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_{ip}^n$</td>
<td>$n+p$</td>
<td>$s$</td>
<td>weights</td>
</tr>
<tr>
<td>$\nabla w_{ip}^n$</td>
<td>$n+p$</td>
<td>$v$</td>
<td>weight gradients</td>
</tr>
<tr>
<td>$m_{ai}$</td>
<td>$f$</td>
<td>$s$</td>
<td>mass</td>
</tr>
<tr>
<td>$x_{ai}$</td>
<td>$f$</td>
<td>$v$</td>
<td>position</td>
</tr>
<tr>
<td>$v_{ai}$</td>
<td>$f$</td>
<td>$s$</td>
<td>velocity</td>
</tr>
<tr>
<td>$v_{at+1}$</td>
<td>$f$</td>
<td>$s$</td>
<td>intermediate velocity</td>
</tr>
<tr>
<td>$w_{at+1}$</td>
<td>$f$</td>
<td>$p$</td>
<td>weights</td>
</tr>
<tr>
<td>$\nabla w_{at+1}$</td>
<td>$f$</td>
<td>$v$</td>
<td>weight gradients</td>
</tr>
<tr>
<td>$\Phi$</td>
<td>$g$</td>
<td>$s$</td>
<td>total potential energy</td>
</tr>
<tr>
<td>$L_{P,n}$</td>
<td>$g$</td>
<td>$v$</td>
<td>total particle angular momentum</td>
</tr>
<tr>
<td>$L_{G,n}$</td>
<td>$g$</td>
<td>$v$</td>
<td>total grid angular momentum</td>
</tr>
<tr>
<td>$N(x)$</td>
<td>$g$</td>
<td>$s$</td>
<td>interpolation kernel</td>
</tr>
<tr>
<td>$\Delta x$</td>
<td>$g$</td>
<td>$s$</td>
<td>grid spacing</td>
</tr>
<tr>
<td>$\epsilon_a$</td>
<td>$g$</td>
<td>$v$</td>
<td>axis vector</td>
</tr>
<tr>
<td>$v^*$</td>
<td>$g$</td>
<td>$m$</td>
<td>cross product matrix of $v$</td>
</tr>
<tr>
<td>$\epsilon$</td>
<td>$g$</td>
<td>$t$</td>
<td>permutation tensor</td>
</tr>
<tr>
<td>$I$</td>
<td>$g$</td>
<td>$m$</td>
<td>identity matrix</td>
</tr>
</tbody>
</table>
Problem of PIC method

- The loss of angular momentum manifests as rotational motion damping

Cause

- When the transfer from particle to grid conserves angular momentum, the transfer from grid back to particle does not.

\[ m_i^n = \sum_p w_{ip}^n m_p, \quad m_i^n \mathbf{v}_i^n = \sum_p w_{ip}^n m_p \mathbf{v}_p^n, \quad (1) \]

\[ w_{ip}^n = N(\mathbf{x}_p^n - \mathbf{x}_i) \]

\[ \mathbf{v}_i^n \rightarrow \tilde{\mathbf{v}}_i^{n+1} \]

\[ \mathbf{v}_p^{n+1} = \sum_i w_{ip}^n \tilde{\mathbf{v}}_i^{n+1}. \quad (2) \]

\[ \mathbf{L}_{tot}^{P,n} = \sum_p \mathbf{x}_p^n \times m_p \mathbf{v}_p^n \quad \mathbf{L}_{tot}^{G,n} = \sum_i \mathbf{x}_i \times m_i^n \mathbf{v}_i^n. \quad (3) \]
• In effort to **reduce the information loss** when transferring, modify the original PIC transfer to **facilitate conservation of angular momentum in grid -> particle transfer**.

• One particle is incapable of representing angular momentum.
• we can additionally store a local angular momentum $L_p^n$ on each particle.
• We can prevent the loss of angular momentum when transferring from grid to particle.

\[
L_p^{n+1} = \sum_i w_{ip}^n (x_i - x_p^n) \times m_p \dot{v}_i^{n+1}. \\
L_{tot}^{P,n} = \sum_p (x_p^n \times m_p v_p^n) + L_p^n.
\]
Particle-grid transfers

Rigid Particle-In-Cell (RPIC) (2/5)

- The transfer from the grid to particle are given by

\[
\begin{align*}
\mathbf{v}_p^{n+1} &= \sum_i w_{ip}^n \mathbf{v}_i^{n+1} \\
L_p^{n+1} &= \sum_i w_{ip}^n (\mathbf{x}_i^n - \mathbf{x}_p^n) \times m_p \mathbf{\tilde{v}}_i^{n+1}.
\end{align*}
\]

- Process
  1. Advection particle
  2. Transfer velocities from particles to grid
  3. Compute NS equation and pressure, get new velocities in grid
  4. Transfer velocities + angular momentum from grid to particle
  5. Advection particle
  6. Transfer velocities from particles to grid with angular momentum
Particle-grid transfers
Rigid Particle-In-Cell (RPIC) (3/5)

- The transfer from the grid to particle are given by

\[
\begin{align*}
\mathbf{v}_p^{n+1} &= \sum_i w_{ip}^{n} \tilde{\mathbf{v}}_i^{n+1} \\
L_p^{n+1} &= \sum_i w_{ip}^{n} (\mathbf{x}_i^n - \mathbf{x}_p^n) \times m_p \tilde{\mathbf{v}}_i^{n+1}.
\end{align*}
\]

- The transfer from particles to the grid are given by

\[
\begin{align*}
m_i^n &= \sum_p w_{ip}^n m_p \\
K_p^n &= \sum_j w_{jp}^n m_p (\mathbf{x}_j^n - \mathbf{x}_p^n)^* (\mathbf{x}_j^n - \mathbf{x}_p^n)^T \\
m_i^n \mathbf{v}_i^n &= \sum_p w_{ip}^n m_p (\mathbf{v}_p^n + ((K_p^n)^{-1} L_p^n) \times (\mathbf{x}_i^n - \mathbf{x}_p^n)) \\
\omega_p^n &= (K_p^n)^{-1} L_p^n \\
\mathbf{v}_p^n + \omega_p^n \times (\mathbf{x}_i^n - \mathbf{x}_p^n) \\
\end{align*}
\]

cf. PIC

\[
m_i^n \mathbf{v}_i^n = \sum_p w_{ip}^n m_p \mathbf{v}_p^n
\]
Particle-grid transfers

Rigid Particle-In-Cell (RPIC) (4/5)

- Conservation of angular momentum
  - Particle to grid

\[
L_{\text{tot}}^{G,n} = \sum_i x_i^n \times m_i^n v_i^n
\]

\[
= \sum_i x_i^n \times \left( \sum_p w_{ip}^n m_p (v_p^n + ((K_p^n)^{-1} L_p^n) \times (x_i^n - x_p^n)) \right)
\]

\[
= \sum_{i,p} x_i^n \times w_{ip}^n m_p v_p^n + \sum_{i,p} x_i^n \times w_{ip}^n m_p ((K_p^n)^{-1} L_p^n) \times (x_i^n - x_p^n))
\]

\[
= \sum_p \left( \sum_i w_{ip}^n x_i^n \right) \times m_p v_p^n + \sum_{i,p} x_i^n \times w_{ip}^n m_p (x_i^n - x_p^n)^T (K_p^n)^{-1} L_p^n
\]

\[
= \sum_p x_p^n \times m_p v_p^n + \sum_{i,p} (x_i^n - x_p^n) \times w_{ip}^n m_p (x_i^n - x_p^n)^T (K_p^n)^{-1} L_p^n + \sum_{i,p} x_p^n \times w_{ip}^n m_p (x_i^n - x_p^n)^T (K_p^n)^{-1} L_p^n
\]

\[
= \sum_p x_p^n \times m_p v_p^n + \sum_p \left( \sum_i m_p w_{ip}^n (x_i^n - x_p^n)^T (x_i^n - x_p^n)^T \right) (K_p^n)^{-1} L_p^n + \sum_p x_p^n \times m_p \left( \sum_i w_{ip}^n (x_i^n - x_p^n) \right)^T (K_p^n)^{-1} L_p^n
\]

\[
= \sum_p x_p^n \times m_p v_p^n + \sum_p K_p^n (K_p^n)^{-1} L_p^n
\]

\[
= \sum_p (x_p^n \times m_p v_p^n + L_p^n)
\]

\[
= L_{\text{tot}}^{P,n}
\]
Particle-grid transfers
Rigid Particle-In-Cell (RPIC) (5/5)

- Conservation of angular momentum
  - Grid to particle

\[
L_{tot}^{P,n+1} = \sum_p (x_p^{n+1} \times m_p v_p^{n+1} + L_p^{n+1})
\]

\[
= \sum_p \left( x_p^{n+1} \times m_p \left( \sum_i w_{ip}^{n} \tilde{v}_i^{n+1} \right) + \left( \sum_i w_{ip}^{n} (x_i^n - x_p^n) \times m_p \tilde{v}_i^{n+1} \right) \right)
\]

\[
= \sum_{i,p} \left( x_p^{n+1} \times m_p w_{ip}^{n} \tilde{v}_i^{n+1} + w_{ip}^{n} (x_i^n - x_p^n) \times m_p \tilde{v}_i^{n+1} \right)
\]

\[
= \sum_{i,p} \left( x_p^{n+1} - x_p^n \right) \times m_p w_{ip}^{n} \tilde{v}_i^{n+1} + \sum_{i,p} w_{ip}^{n} x_i^n \times m_p \tilde{v}_i^{n+1}
\]

\[
= \Delta t \sum_p v_p^{n+1} \times m_p \sum_i w_{ip}^{n} \tilde{v}_i^{n+1} + \sum_{i,p} w_{ip}^{n} (x_i^{n+1} - \Delta t \tilde{v}_i^{n+1}) \times m_p \tilde{v}_i^{n+1}
\]

\[
= \Delta t \sum_{i,p} v_p^{n+1} \times m_p v_i^{n+1} + \sum_{i,p} w_{ip}^{n} x_i^{n+1} \times m_p \tilde{v}_i^{n+1}
\]

\[
= \sum_i \left( \sum_p w_{ip}^{n} m_p \right) x_i^{n+1} \times \tilde{v}_i^{n+1}
\]

\[
= \sum_i x_i^{n+1} \times m_i v_i^{n+1}
\]

\[
= L_{tot}^{G,n+1}
\]
Particle-grid transfers
Affine Particle-In-Cell (APIC) (1/4)

• When non-rigid motion such as shearing, it still damps out.
• Extend the idea of enriching our velocity representation to handle shearing mode by idealizing the velocity as locally affine on each particle.
• Shearing mode is motivated analogously to the piecewise rigid case.
• When transferring from particle to grid, defining matrix $C_p^n$ from vector $L_p^n$ is not possible, which complicates the process of the transfer from grid to particles.
  • When transferring from particle to grid,
    \[ v_p^n + C_p^n(x_i - x_p^n) \]
• So, rather than trying to conserve angular momentum in the transfer from grid to particle, it seeks to preserve affine velocity fields across both transfers.
Particle-grid transfers
Affine Particle-In-Cell (APIC) (2/4)

• Affine motions

A motion is said to be affine if the one-parameter family of deformations (4) is such that for any two body points A and B and for any t

$$\phi(\vec{p}_B, t) = \phi(\vec{p}_A, t) + \mathbf{F}(t)(\vec{p}_B - \vec{p}_A),$$

(52)

where \(\mathbf{F}(t)\) is a tensor such that \(\det \mathbf{F}(t) > 0\). Hence an affine motion is defined by the motion of any body point, say A, and by the values of the deformation gradient \(\mathbf{F}(t)\).

The velocity field in an affine motion is given by

$$\dot{\phi}(\vec{p}_B, t) = \dot{\phi}(\vec{p}_A, t) + \dot{\mathbf{F}}(t)(\vec{p}_B - \vec{p}_A).$$

(53)

Replacing (52) with

$$\vec{p}_B(t) = \vec{p}_A(t) + \mathbf{F}(t)(\vec{p}_B - \vec{p}_A),$$

(54)

we get

$$\dot{\vec{p}}_B(t) = \dot{\vec{p}}_A(t) + \dot{\mathbf{F}}(t)(\vec{p}_B(t_0) - \vec{p}_A(t_0)).$$

(55)

Since from (54)

$$\vec{p}_B(t_0) - \vec{p}_A(t_0) = \mathbf{F}^{-1}(t_0)(\vec{p}_B(t) - \vec{p}_A(t)),$$

(56)

then

$$\dot{\vec{p}}_B(t) = \dot{\vec{p}}_A(t) + \dot{\mathbf{F}}(t)\mathbf{F}^{-1}(t_0)(\vec{p}_B(t) - \vec{p}_A(t)).$$

(57)

Setting

$$\mathbf{L}(t) := \dot{\mathbf{F}}(t)\mathbf{F}^{-1}(t),$$

(58)

the expression (57) can be written

$$\dot{\vec{p}}_B(t) = \dot{\vec{p}}_A(t) + \mathbf{L}(t)(\vec{p}_B(t) - \vec{p}_A(t)).$$

(59)
If the velocities before the transfer represent an affine velocity field, \( \tilde{v}_i^{n+1} = v + Cx_i^n \), then after the process, this velocity field is exactly reproduced \( \tilde{v}_i^{n+1} = \tilde{v}_i^{n+1} \).

\[
\begin{align*}
v_p^{n+1} &= \sum_i w_{ip}^n \tilde{v}_i^{n+1} \\
&= \sum_i w_{ip}^n (v + Cx_i^n) \\
&= \sum_i w_{ip}^n v + \sum_i w_{ip}^n Cx_i^n \\
&= v \sum_i w_{ip}^n + C \sum_i w_{ip}^n x_i^n \\
&= v + Cx_p^n
\end{align*}
\]

\[
\begin{align*}
B_p^{n+1} &= \sum_i w_{ip}^n \tilde{v}_i^{n+1} (x_i^n - x_p^n)^T \\
&= \sum_i w_{ip}^n (v + Cx_i^n) (x_i^n - x_p^n)^T \\
&= \sum_i w_{ip}^n v(x_i^n - x_p^n)^T + \sum_i w_{ip}^n Cx_i^n (x_i^n - x_p^n)^T \\
&= v \left( \sum_i w_{ip}^n (x_i^n - x_p^n)^T \right) + C \sum_i w_{ip}^n (x_i^n - x_p^n)(x_i^n - x_p^n)^T + \sum_i w_{ip}^n Cx_p^n (x_i^n - x_p^n)^T \\
&= CD_p^n + Cx_p^n \left( \sum_i w_{ip}^n (x_i^n - x_p^n) \right)^T \\
&= CD_p^n
\end{align*}
\]
Particle-grid transfers

Affine Particle-In-Cell (APIC) (4/4)

- The transfer from particles to the grid are given by

\[
m^n_i = \sum_p w^n_{ip} m_p \\
D^n_p = \sum_i w^n_{ip} (x^n_i - x^n_p)(x^n_i - x^n_p)^T = \sum_i w^n_{ip} x^n_i (x^n_i)^T - x^n_p (x^n_p)^T \\
m^n_i \mathbf{v}^n_i = \sum_p w^n_{ip} m_p (\mathbf{v}^n_p + B^n_p (D^n_p)^{-1} (x^n_i - x^n_p))
\]

- With the transfer to particle given by

\[
\mathbf{v}^{n+1}_p = \sum_i w^n_{ip} \mathbf{\tilde{v}}^{n+1}_i \\
B^{n+1}_p = \sum_i w^n_{ip} \mathbf{\tilde{v}}^{n+1}_i (x^n_i - x^n_p)^T.
\]

- The preservation of affine velocity fields also conserves angular momentum.
Lagrangian forces

- Use Lagrangian forces as Eulerian forces
  - Compute $f_i$ from $f_p$.
  - Find a means of computing $\delta f_i$ given $\delta u_i$

$$f_p = -\frac{\partial \Phi}{\partial x_p} \quad \delta f_p = \sum_q \frac{\partial f_p}{\partial x_q} \delta u_q \quad x_p = \sum_i w_{ip} x_i$$

$$f_i = \sum_p w_{ip} f_p \quad \delta f_i = \sum_{p,q,j} w_{ip} \frac{\partial f_p}{\partial x_q} w_{jq} \delta u_j. \quad (15)$$

- Since these forces are applied to the grid, both the MPM and Lagrangian approaches can be employed in the same simulation.
- This provides an effective means of coupling MPM with mesh-based approaches.
- This gives the precise surface tracking of Lagrangian techniques coupled with the automatic collision handling of Eulerian grids.
Results

A Comparison of results
Results

Free-surface flow

Figure 5. Comparison of the method on a fountain simulation.

APIC and FLIP are the least dissipative. PIC is the most damped. FLIP surface displays a noisy edge.

Figure 8. APIC resolves the complex free-surface dynamics of a rushing river on a rocky terrain.
Results

Granular materials (1/2)
Results
Granular materials (2/2)

APIC

FLIP

PIC

FLIP .95
Results

Coupling MPM and Lagrangian

Figure 11. APIC coupled simulation

Figure 12. Comparison using mesh-based cubes

Figure 12. FLIP produce noise during the collision
Results
Lava

Figure 6. APIC resolve a periodic flow. FLIP goes unstable, with hotter interior particle exploding outwards.

Figure 16. APIC performance with lava flow
Results

Energy

- Figure 13.
  - APIC and RPIC are effective at resolving rotation.
  - FLIP is more consistent about retaining energy, since it is immune to dissipation.

- Figure 14.
  - APIC has less energy loss than PIC.
Discussion and limitations

• APIC
  ▪ **Eliminate** nearly all of the *artificial dissipation* of pure PIC.
  ▪ Affect nothing about the ringing instability.
  ▪ The need to store an extra matrix per particle and perform a few extra operations during the transfers from grid to particles.
  ▪ But be negligible as runtime costs are dominated by others.

• FLIP
  ▪ **Avoids dissipation** at the cost of *losing stability*.
  ▪ **Preserve energy better** in many cases.
  ▪ Be slowest due to its instability and consequent larger velocities.

• PIC
  ▪ Be fastest, since it damps out motion (**dissipation**) and has the smallest velocities.
Appendix A. Momentum

- **Linear momentum**
  - $\mathbf{p} = m\mathbf{v}$

- **Angular momentum**
  - $\mathbf{L} = \mathbf{r} \times \mathbf{p} = \mathbf{r} \times m\mathbf{v}$
  - $\mathbf{L} = I\omega$

- **Linear velocity**
  - $\mathbf{v} = r\omega$

- **Inertia moment and Inertia tensor**
  - $I = mr^2$

관성 텐서는 기존 관성 모멘트가 회전축에 따라 달라지는 것을 보완하고, 평면 상의 강체 회전 뿐만 아니라 3차원 상의 회전을 기술하기 위한 관성 모멘트에 대응하는 물리량이다.
Appendix B. Cross-product matrix

- The skew symmetric matrix
  - Also called anti symmetric: $S^T = -S$
  - $S^T + S = 0$
  - Always singular $\det(S) = 0$
  - In 3D,

$$\begin{pmatrix}
0 & -z & y \\
z & 0 & -x \\
-y & x & 0
\end{pmatrix} , \mathbf{v} = [x, y, z]$$

- An alternative way to express the vector cross product
  - $a \times b = S(a)b$