A Review of The Affine Particle-In-Cell Method

Presenter: Pin-Hua Ho

Introduction

- What’s Affine Paticle-In-Cell (APIC)?

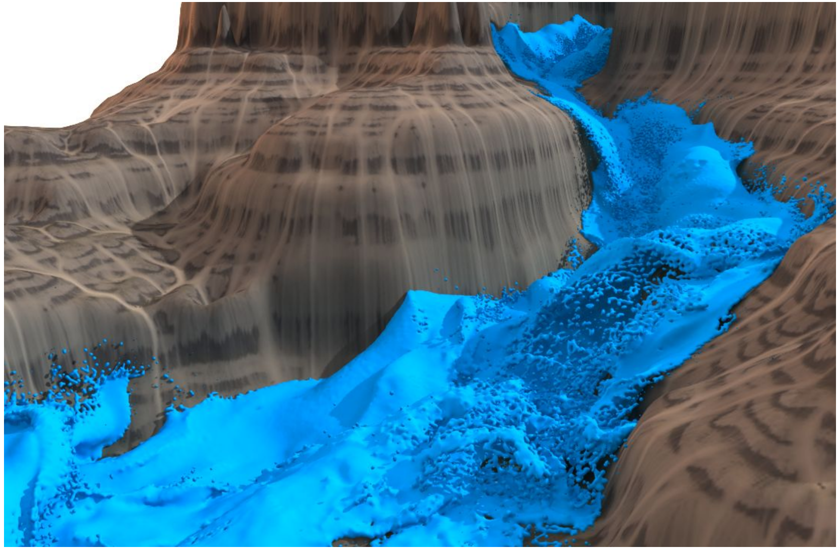
Introduction

- What’s Affine Paticle-In-Cell (APIC)?
- APIC is a **transfer scheme** for **hybrid particle/grid** simulation

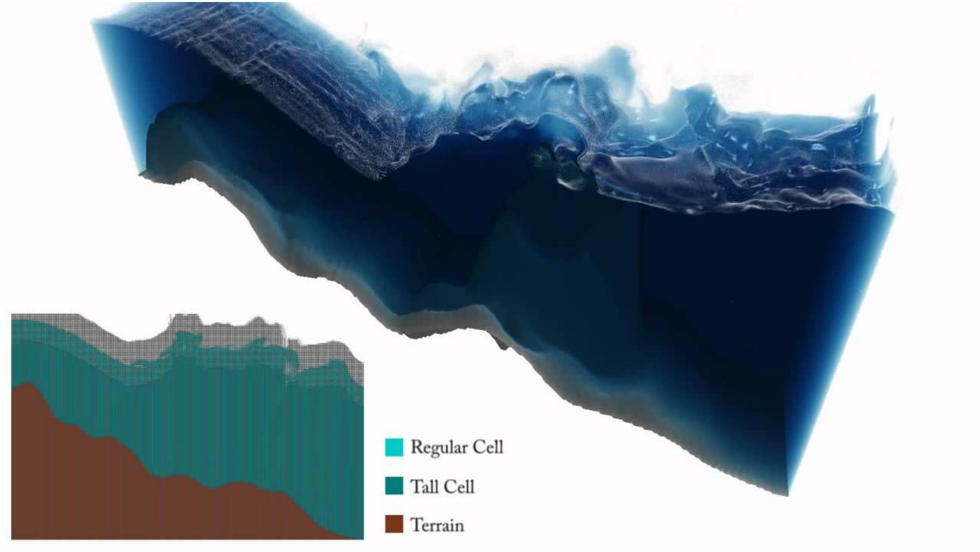
Introduction

- What’s Affine Paticle-In-Cell (APIC)?
- APIC is a **transfer scheme** for **hybrid particle/grid** simulation
- transfer scheme? hybrid particle/grid method?

Fluid Simulation: Lagrangian vs. Eulerian



Lagrangian particles



Eulerian grid

image: RWTH Computer Animation Group

image: Chentanez et al., Real-Time Eulerian Water Simulation Using a Restricted Tall Cell Grid, Siggraph 2011

Fluid Simulation: Lagrangian

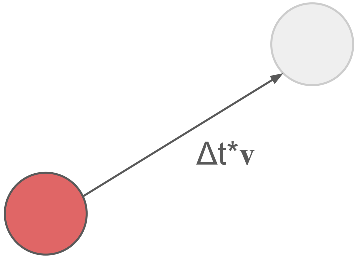
- Describe the fluid as a set of **particles** moving through space

Fluid Simulation: Lagrangian

- Describe the fluid as a set of **particles** moving through space
- Good at handling **advection**
  - Advection refers to the transport of a fluid material by bulk motions

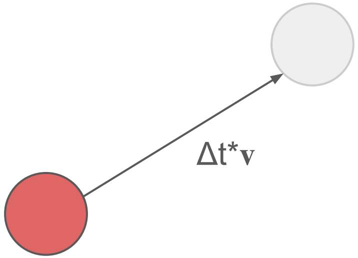
Fluid Simulation: Lagrangian

- Describe the fluid as a set of **particles** moving through space
- Good at handling **advection**
  - Advection refers to the transport of a fluid material by bulk motions
- The fluid material is the particles themselves
- Kinematic motion → advection
  - Mass conservation is trival



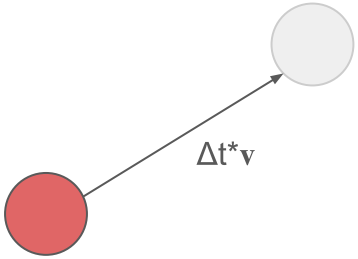
Fluid Simulation: Lagrangian

- Describe the fluid as a set of **particles** moving through space
- Good at handling **advection**
    - Advection refers to the transport of a fluid material by bulk motions
- The fluid material is the particles themselves
- Kinematic motion → advection
    - Mass conservation is trival
- Bad at solving dynamics



Fluid Simulation: Lagrangian

- Describe the fluid as a set of **particles** moving through space
- Good at handling **advection**
  - Advection refers to the transport of a fluid material by bulk motions
- The fluid material is the particles themselves
- Kinematic motion → advection
  - Mass conservation is trival
- Bad at solving dynamics
- Unstructured particle cloud
  - Hard to calculate spatial derivatives



Fluid Simulation: Eulerian

- Describe the fluid by a fixed spatial grid

Fluid Simulation: Eulerian

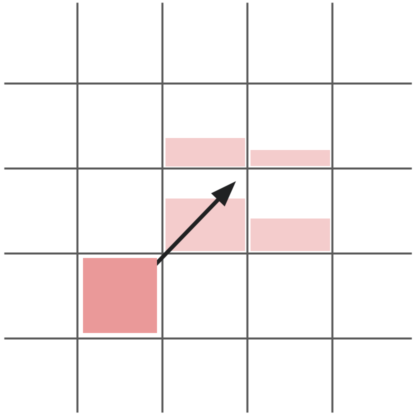
- Describe the fluid by a fixed spatial grid
- Good at solving dynamics
  - Spatially structured grid cells
  - Great for calculating spatial derivatives

Fluid Simulation: Eulerian

- Describe the fluid by a fixed spatial grid
- Good at solving dynamics
  - Spatially structured grid cells
  - Great for calculating spatial derivatives
- Bad at handling advection

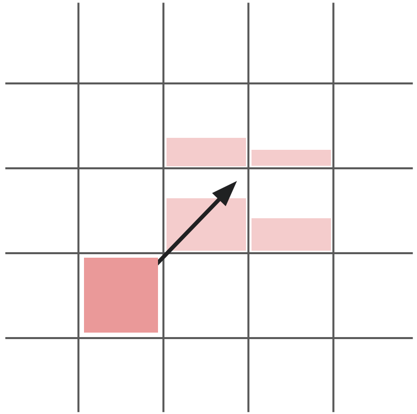
Fluid Simulation: Eulerian

- Describe the fluid by a fixed spatial grid
- Good at solving dynamics
  - Spatially structured grid cells
  - Great for calculating spatial derivatives
- Bad at handling advection
  - Fluid transport is handled by updating grid values



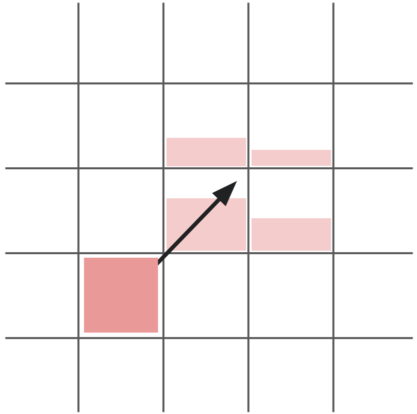
Fluid Simulation: Eulerian

- Describe the fluid by a fixed spatial grid
- Good at solving dynamics
  - Spatially structured grid cells
  - Great for calculating spatial derivatives
- Bad at handling advection
  - Fluid transport is handled by updating grid values
  - Inevitably, interpolations



Fluid Simulation: Eulerian

- Describe the fluid by a fixed spatial grid
- Good at solving dynamics
  - Spatially structured grid cells
  - Great for calculating spatial derivatives
- Bad at handling advection
  - Fluid transport is handled by updating grid values
  - Inevitably, interpolations → numerical erros
  - Mass conservation is not trival



Fluid Simulation: Lagrangian “and” Eulerian

- How about we combine them and get both advantages?

Fluid Simulation: Lagrangian “and” Eulerian

- How about we combine them and get both advantages?
- This is how the **hybrid particle/grid** method came from

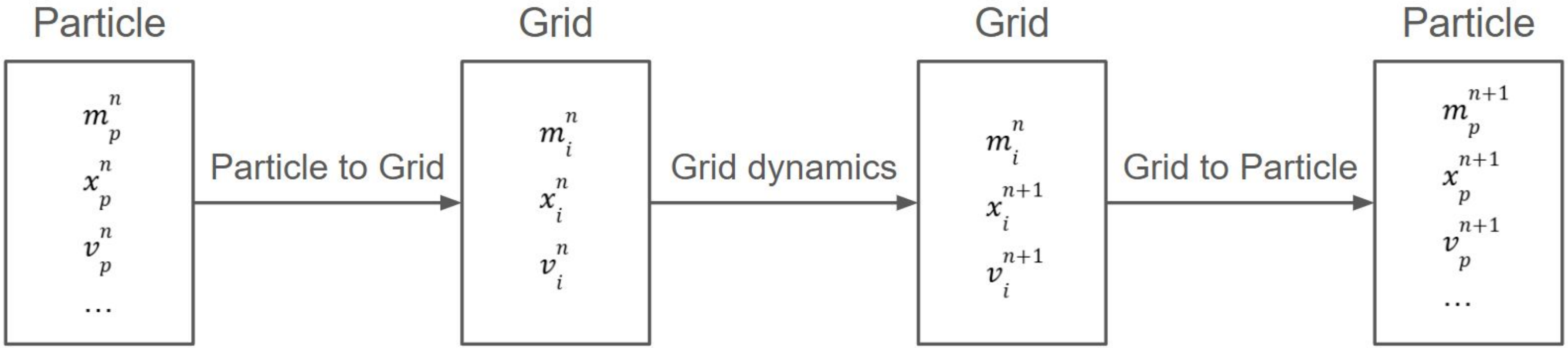
Fluid Simulation: Lagrangian “and” Eulerian

- How about we combine them and get both advantages?
- This is how the **hybrid particle/grid** method came from
- Use Lagrangian particles to perform advection

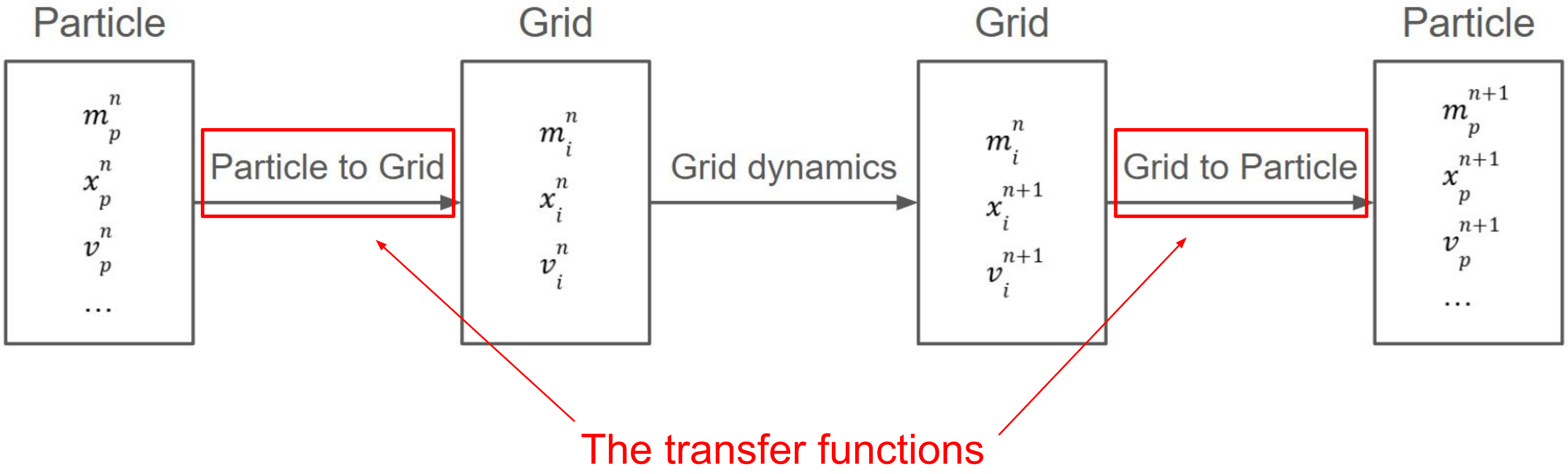
Fluid Simulation: Lagrangian “and” Eulerian

- How about we combine them and get both advantages?
- This is how the **hybrid particle/grid** method came from
- Use Lagrangian particles to perform advection
- And Eulerian grid to solve dynamics

Fluid Simulation: The Hybrid Method



Fluid Simulation: The Hybrid Method



The Classics: PIC and FLIP

- There are two classic hybrid methods in fluid simulation

The Classics: PIC and FLIP

- There are two classic hybrid methods in fluid simulation
- The Particle-In-Cell (PIC) method

The Classics: PIC and FLIP

- There are two classic hybrid methods in fluid simulation
- The Particle-In-Cell (PIC) method
- The Fluid Implicit Particle (FLIP) method

The Classics: PIC

- PIC was the first hybrid method proposed by Harlow in 1964
- Effective and simple to implement
- Quickly became prominent

The Classics: PIC

- PIC was the first hybrid method proposed by Harlow in 1964
- Effective and simple to implement
- Quickly became prominent
- However, it has a major drawback

The Classics: PIC

- PIC was the first hybrid method proposed by Harlow in 1964
- Effective and simple to implement
- Quickly became prominent
- However, it has a major drawback
    - The dissipation

The Classics: PIC (cont'd)

- The dissipation
- Total energy loss over time

The Classics: PIC (cont’d)

- The dissipation
- Total energy loss over time
- Mismatch of degree of freedom between particles and the grid

The Classics: PIC (cont'd)

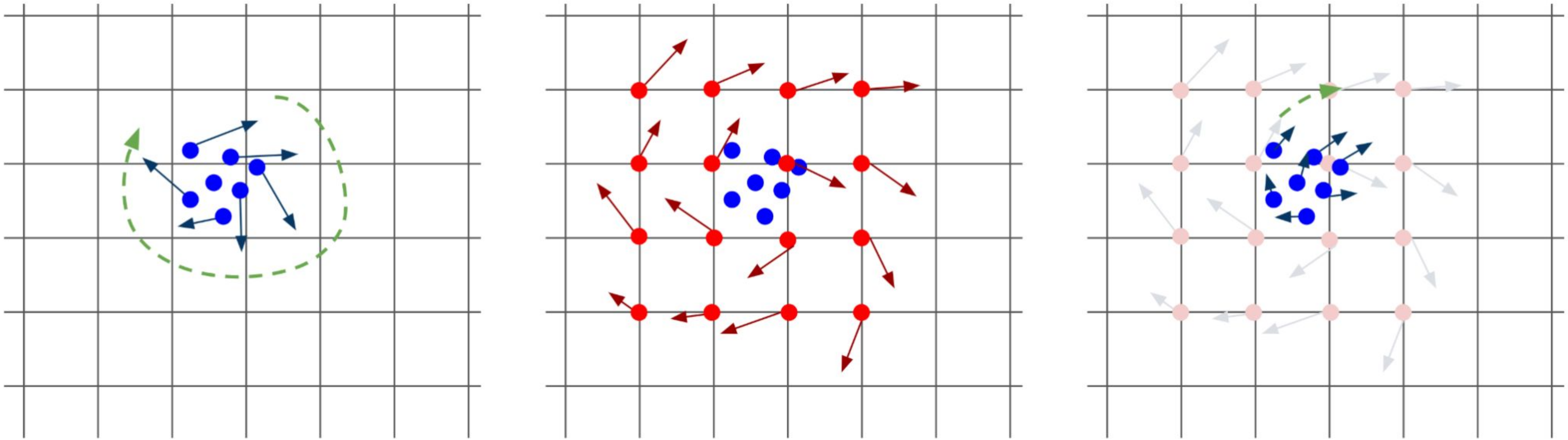
- The dissipation
- Total energy loss over time
- Mismatch of degree of freedom between particles and the grid
- Ususally, # of particles >> # of grid nodes

The Classics: PIC (cont’d)

- The dissipation
- Total energy loss over time
- Mismatch of degree of freedom between particles and the grid
- Ususally, # of particles >> # of grid nodes
- Particles can carry more information then the grid could

The Classics: PIC (cont'd)

- When transfer between particles and the grid



(a) Before P2G

(b) Rasterized onto grid

(c) After G2P

The Classics: FLIP

- FLIP was proposed by Brackbill in 1988
- Aimed to solve the dissipation in PIC

The Classics: FLIP

- FLIP was proposed by Brackbill in 1988
- Aimed to solve the dissipation in PIC
- Instead of interpolating and overriding the particle value at each time step

The Classics: FLIP

- FLIP was proposed by Brackbill in 1988
- Aimed to solve the dissipation in PIC
- Instead of interpolating and overriding the particle value at each time step
- Grid only calculate the increments then apply them to particles

The Classics: FLIP (cont'd)

- During the G2P transfers:

$$\mathbf{v}_p^{n+1} \leftarrow \sum_i w_{ip} \mathbf{v}_i^{n+1}.$$

PIC

$$\mathbf{v}_p^{n+1} \leftarrow \mathbf{v}_p^{n} + \sum_i w_{ip} \mathbf{\Delta v}_i^{n+1}.$$

FLIP

The Classics: FLIP (cont’d)

- FLIP's solution was effective
- Massively improved conserving total energy

The Classics: FLIP (cont’d)

- FLIP's solution was effective
- Massively improved conserving total energy
- However, it comes with a cost

The Classics: FLIP (cont’d)

- The mismatch of DoG between particles and the grid persists
- Some sub-grid fluid motions or **modes** couldn’t be seen by the grid

The Classics: FLIP (cont’d)

- The mismatch of DoG between particles and the grid persists
- Some sub-grid fluid motions or **modes** couldn’t be seen by the grid
- Thus getting no proper physical response

The Classics: FLIP (cont’d)

- The mismatch of DoG between particles and the grid persists
- Some sub-grid fluid motions or **modes** couldn’t be seen by the grid
- Thus getting no proper physical response
- Results in so-called “ringing instability”

The Classics: FLIP (cont’d)

- The mismatch of DoG between particles and the grid persists
- Some sub-grid fluid motions or **modes** couldn’t be seen by the grid
- Thus getting no proper physical response
- Results in so-called “ringing instability”
- The errors gets amplified overtime

$$\mathbf{v}_p^{n+1} \leftarrow \boxed{\mathbf{v}_p^n} + \sum_i w_{ip} \boldsymbol{\Delta} \mathbf{v}_i^{n+1}.$$

Using old values → errors persists!

Where Should We Start

- Key observation:

Where Should We Start

- Key observation:
- During G2P transfers
- Multiple grid node values are reduced to a single particle value

Where Should We Start

- Key observation:
- During G2P transfers
- Multiple grid node values are reduced to a single particle value
- How about making a single particle to carry more information?

The First Try: Rigid Particle-In-Cell (RPIC)

- Angular momentum loss was the primary observation from PIC
- Explicitly compensate for that angular momentum loss

The First Try: Rigid Particle-In-Cell (RPIC)

- Angular momentum loss was the primary observation from PIC
- Explicitly compensate for that angular momentum loss
- A particle could influence multiple nearby grid nodes

The First Try: Rigid Particle-In-Cell (RPIC)

- Angular momentum loss was the primary observation from PIC
- Explicitly compensate for that angular momentum loss
- A particle could influence multiple nearby grid nodes
- inherently creating a “soft boundary”

The First Try: Rigid Particle-In-Cell (RPIC)

- Angular momentum loss was the primary observation from PIC
- Explicitly compensate for that angular momentum loss
- A particle could influence multiple nearby grid nodes
- inherently creating a “soft boundary”
- Treating the particle as a rigid body with volume

The First Try: Rigid Particle-In-Cell (RPIC) (cont’d)

- Let each particle remember its own angular momentum

The First Try: Rigid Particle-In-Cell (RPIC) (cont'd)

- Let each particle remember its own angular momentum
- Then?

$$
\begin{array}{lcl}
m_p^n & \longrightarrow & m_i^n \\
x_p^n & \longrightarrow & x_i^n \\
v_p^n & \longrightarrow & v_i^n \\
L_p^n & &
\end{array}
$$

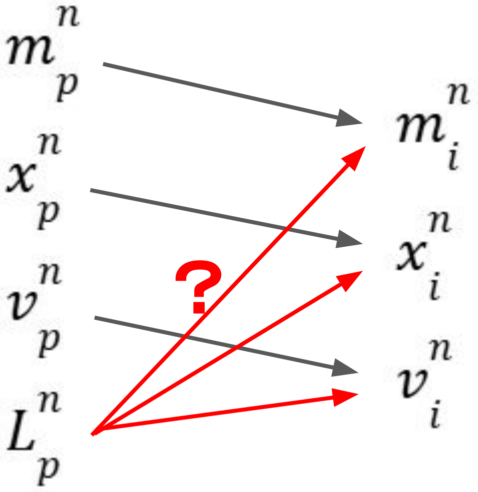
P2G Transfer

$$
\begin{array}{lcl}
m_i^n & \longrightarrow & m_p^{n+1} \\
x_i^{n+1} & \longrightarrow & x_p^{n+1} \\
v_i^{n+1} & \longrightarrow & v_p^{n+1} \\
& & L_p^{n+1}
\end{array}
$$

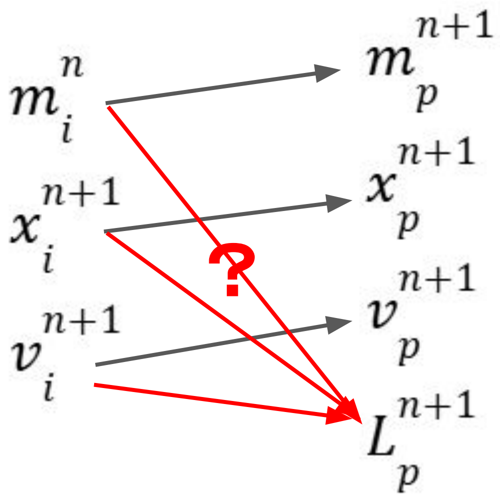
G2P Transfer

The First Try: Rigid Particle-In-Cell (RPIC) (cont'd)

- Let each particle remember its own angular momentum
- Then?



P2G Transfer



G2P Transfer

The First Try: Rigid Particle-In-Cell (RPIC) (cont’d)

- Have to design a new set of transfer functions

The First Try: Rigid Particle-In-Cell (RPIC) (cont’d)

- Have to design a new set of transfer functions
- The rigid body rotation

The First Try: Rigid Particle-In-Cell (RPIC) (cont'd)

- Have to design a new set of transfer functions
- The rigid body rotation

$$\mathbf{L}_p^{n+1} = \sum_i w_{ip}^n (\mathbf{x}_i - \mathbf{x}_p^n) \times m_p \mathbf{v}_i^{n+1}.$$

G2P Transfer

The First Try: Rigid Particle-In-Cell (RPIC) (cont'd)

- Have to design a new set of transfer functions
- The rigid body rotation

$$\mathbf{L}_p^{n+1} = \sum_i w_{ip}^n (\mathbf{x}_i - \mathbf{x}_p^n) \times m_p \mathbf{v}_i^{n+1}.$$

G2P Transfer

- Like a discrete integration over the piece-wise rigid body

The First Try: Rigid Particle-In-Cell (RPIC) (cont'd)

- For P2G transfer

$$m_i^n v_i^n = \sum_p w_{ip}^n m_p(\mathbf{v}_p^n + ((\mathbf{K}_p^n)^{-1}\mathbf{L}_p^n) \times (\mathbf{x}_i - \mathbf{x}_p^n).$$

- Matrix K is the inertia tensor
- The term $((\mathbf{K}_p^n)^{-1}\mathbf{L}_p^n)$ is the angular velocity of the particle
- The cross product gives the tangent velocity at grid location x_i

The First Try: Rigid Particle-In-Cell (RPIC) (cont’d)

- RPIC effectively conserves angular momentum

The First Try: Rigid Particle-In-Cell (RPIC) (cont’d)

- RPIC effectively conserves angular momentum
- However, rotation is only one of the many modes
- Thus dissipation persists

The First Try: Rigid Particle-In-Cell (RPIC) (cont’d)

- RPIC effectively conserves angular momentum
- However, rotation is only one of the many modes
- Thus dissipation persists
- Need a more powerful description for the local transform

The Affine Particle-In-Cell (APIC)

- If a single vector $\mathbf{L}_p^n$ is not enough
- How about a 3x3 matrix?

The Affine Particle-In-Cell (APIC)

- If a single vector $\mathbf{L}_p^n$ is not enough
- How about a 3x3 matrix?
- An affine transform matrix

The Affine Particle-In-Cell (APIC)

- If a single vector $\mathbf{L}_p^n$ is not enough
- How about a 3x3 matrix?
- An affine transform matrix
- That can describe the full, locally affine velocity field

The Affine Particle-In-Cell (APIC) (cont’d)

- Again, have to design new transfer functions
- First consider how to update it

The Affine Particle-In-Cell (APIC) (cont'd)

- Again, have to design new transfer functions
- First consider how to update it
- It's impossible to uniquely define 9 components from the 3x1 vector $\mathbf{L}_p^n$

The Affine Particle-In-Cell (APIC) (cont'd)

- Again, have to design new transfer functions
- First consider how to update it
- It's impossible to uniquely define 9 components from the 3x1 vector $\mathbf{L}_p^n$
- Divide the affine velocity matrix into 2 matrices
- Like p = mv?

The Affine Particle-In-Cell (APIC) (cont'd)

- The matrix division: $\mathbf{C}_p^n = \mathbf{B}_p^n (\mathbf{D}_p^n)^{-1}$.
- $\mathbf{C}_p^n$ is the affine velocity fields

The Affine Particle-In-Cell (APIC) (cont'd)

- The matrix division: $\mathbf{C}_p^n = \mathbf{B}_p^n (\mathbf{D}_p^n)^{-1}$.
- $\mathbf{C}_p^n$ is the affine velocity fields
- $\mathbf{D}_p^n$ is a inertia tensor-like matrix, which can be computed:

$$\mathbf{D}_p^n = \sum_i w_{ip}^n (\mathbf{x}_i - \mathbf{x}_p^n)(\mathbf{x}_i - \mathbf{x}_p^n)^T.$$

- D can be computed at the beginning of each time step

The Affine Particle-In-Cell (APIC) (cont'd)

- The matrix division: $\mathbf{C}_p^n = \mathbf{B}_p^n(\mathbf{D}_p^n)^{-1}$.
- $\mathbf{C}_p^n$ is the affine velocity fields
- $\mathbf{D}_p^n$ is a inertia tensor-like matrix, which can be computed:

$$\mathbf{D}_p^n = \sum_i w_{ip}^n (\mathbf{x}_i - \mathbf{x}_p^n)(\mathbf{x}_i - \mathbf{x}_p^n)^T.$$

- D can be computed at the beginning of each time step
- Then by definition, $\mathbf{B}_p^n$ is a momentum-like matrix

The Affine Particle-In-Cell (APIC) (cont'd)

- In G2P transfer, the term $\mathbf{B}_p^n$ is updated by:

$$\mathbf{B}_p^{n+1} = \sum_i w_{ip}^n \mathbf{v}_i^{n+1} (\mathbf{x}_i - \mathbf{x}_p^n)^T.$$

The Affine Particle-In-Cell (APIC) (cont'd)

- In G2P transfer, the term $\mathbf{B}_p^n$ is updated by:

$$\mathbf{B}_p^{n+1} = \sum_i w_{ip}^n \mathbf{v}_i^{n+1} (\mathbf{x}_i - \mathbf{x}_p^n)^T.$$

- Then during the P2G transfer, anagolous to the RPIC:

$$m_i^n \mathbf{v}_i^n = \sum_p w_{ip}^n m_p (\mathbf{v}_p^n + \mathbf{B}_p^n (\mathbf{D}_p^n)^{-1} (\mathbf{x}_i - \mathbf{x}_p^n)).$$

The Affine Particle-In-Cell (APIC): The Proof

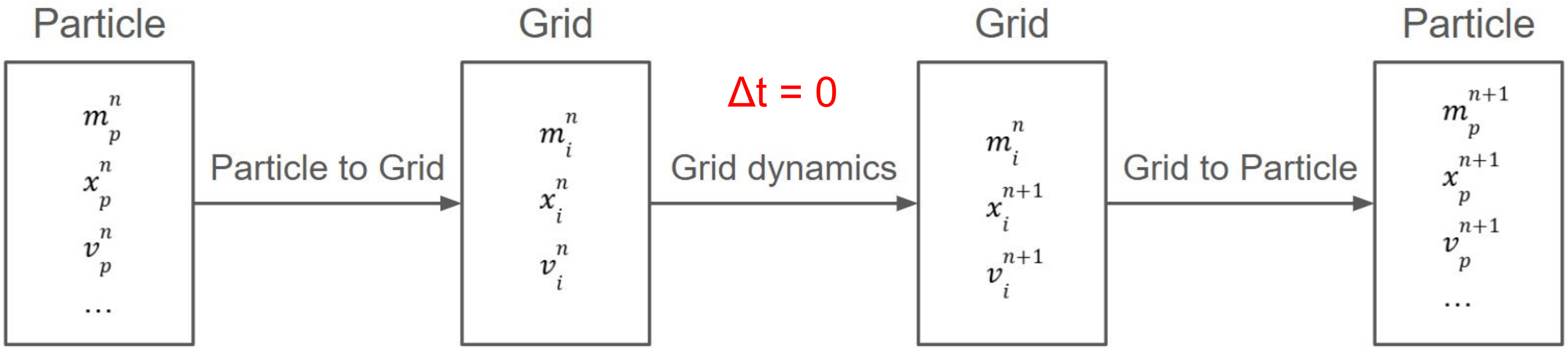
- How do we know the newly designed transfer function is correct?

The Affine Particle-In-Cell (APIC): The Proof

- How do we know the newly designed transfer function is correct?
- The authors proved conservation properties in the supplemental document

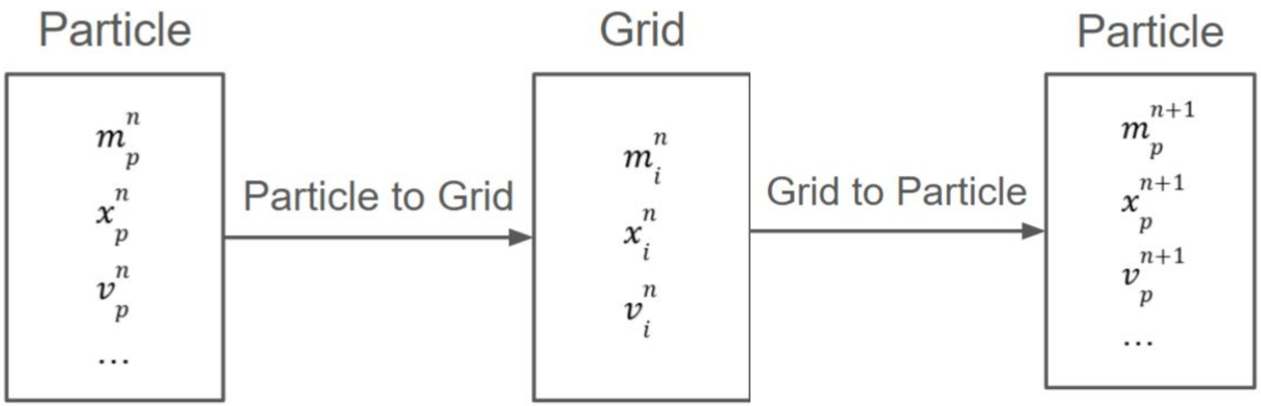
The Affine Particle-In-Cell (APIC): The Proof

- How do we know the newly designed transfer function is correct?
- The authors proved conservation properties in the supplemental document
- Concept: Set Δt = 0



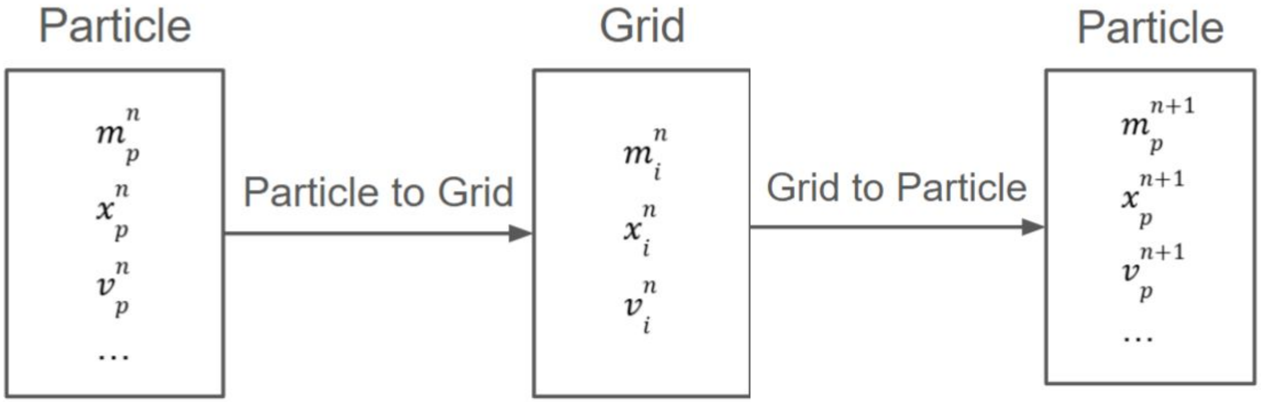
The Affine Particle-In-Cell (APIC): The Proof

- How do we know the newly designed transfer function is correct?
- The authors proved conservation properties in the supplemental document
- Concept: Set Δt = 0



The Affine Particle-In-Cell (APIC): The Proof

- How do we know the newly designed transfer function is correct?
- The authors proved conservation properties in the supplemental document
- Concept: Set $\Delta t = 0$
- Prove that during P2G and G2P transfers
  - Linear momentum conserves
  - Angular momentum conserves



Results

- APIC successfully improved dissipation based on PIC
- Didn’t rely on the unsafe direct data path in FLIP
  - Won’t have instability issue as FLIP does

Results

- APIC successfully improved dissipation based on PIC
- Didn’t rely on the unsafe direct data path in FLIP
  - Won’t have instability issue as FLIP does
- Compared with pure PIC, FLIP and the linear blend of PIC/FLIP
- The linear blend of PIC/FLIP is a practical use to mitigate both disadvantages

Results

image: Jiang et al. The Affine Particle-In-Cell Method

Discussion

- APIC didn’t solve nor worsen ringing instability
- In terms of energy conservation, FLIP still does a better job
- The computational and memory overheads are dominated by other factor

Conclusion

- Traditional PIC suffers from dissipation
- FLIP fixed dissipation but introduced uncontrollable noise and instability
- The linear blend of PIC and FLIP mitigated but mixed both dissipation and noise
    - Moreover, finetuning the blending weight case by case adds another complexity for simulation

Conclusion (cont'd)

- APIC, based on PIC's framework, solved most of the dissipation
  - and being stable at the same time
  - provides a stable yet detailed, and controllable simulation
- APIC is versatile as a particle/grid transfer scheme
  - Can be use for other simulation frameworks as well, such as material point method (MPM)

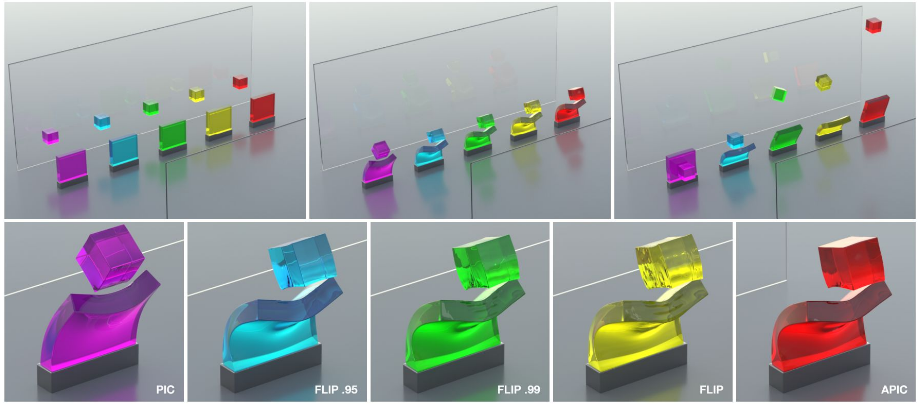


image: Jiang et al. The Affine Particle-In-Cell Method

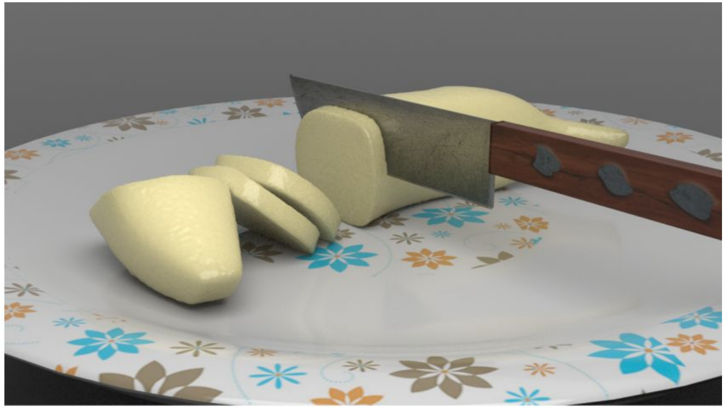


image: Hu et al. A Moving Least Squares Material Point Method (MLS-MPM) with Displacement Discontinuity and Two-Way Rigid Body Coupling

Thanks for listening!