Uma Implementação do Método FLIP para Simulação 2D de Fluidos

Aluno: Gabriel Carvalho Sanches Rocha
Orientador: Paulo Aristarco Pagliosa

Universidade Federal do Mato Grosso do Sul

Março de 2021

Roteiro

Introdução

- Animação baseada em física

Introdução

- Animação baseada em física
- Simulação de fluidos

Introdução

- O que são fluidos?

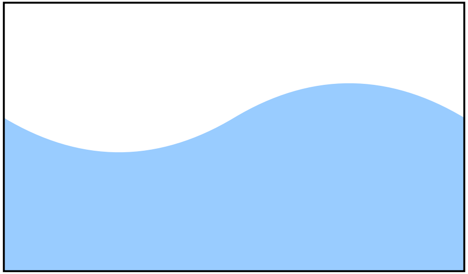
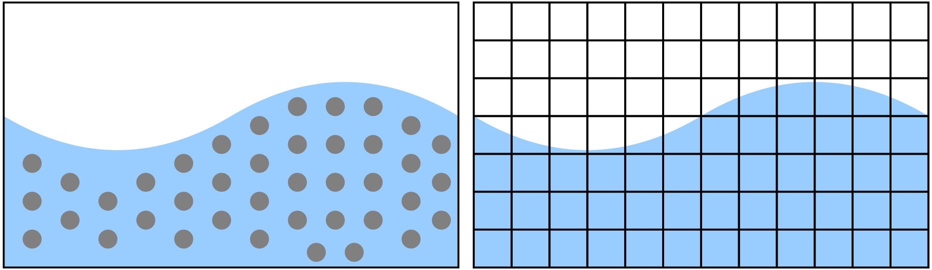


Figura: Líquido em um recipiente.

Introdução

- O que são fluidos?
- Como simular fluidos?



(a) Líquido discretizado com partículas. (b) Líquido discretizado com grade regular.

Figura: Abordagens Lagrangiana e Euleriana.

Introdução

- O que são fluidos?
- Como simular fluidos?
- Abordagens híbridas
    - *Particle-in-cell*
    - *Fluid-implicit-particle*

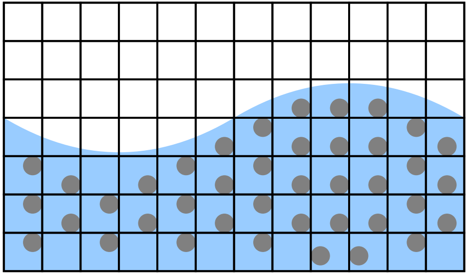


Figura: Abordagem híbrida usando partículas e grade.

Introdução

- Objetivos
  1. Estudar os métodos híbridos PIC e FLIP
  2. Implementar os métodos PIC e FLIP para simular fluidos em duas dimensões
- Contribuições
  1. Fornecer referência para outras implementações
  2. Extensão com uso de grades adaptativas balanceadas

Fundamentos

- Equações de Navier-Stokes

$$\frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} + \frac{\nabla p}{\rho} = \vec{g} + \mu \nabla^2 \vec{u} \tag{1}$$

$$\nabla \cdot \vec{u} = 0 \tag{2}$$

Fundamentos

$$\frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} + \frac{\nabla p}{\rho} = \vec{g} + \mu \nabla^2 \vec{u} \qquad \text{(1 revisitada)}$$

Gradiente para funções escalares

$$\nabla f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right) \qquad (3)$$

Gradiente para funções vetoriais

$$\nabla \vec{f} = \begin{bmatrix} \frac{\partial f_x}{\partial x} & \frac{\partial f_x}{\partial y} \\ \\ \frac{\partial f_y}{\partial x} & \frac{\partial f_y}{\partial y} \end{bmatrix} \qquad (4)$$

Fundamentos

$$\frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} + \frac{\nabla p}{\rho} = \vec{g} + \mu \nabla^2 \vec{u} \qquad \text{(1 revisitada)}$$

Gradiente para funções escalares

$$\nabla f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right) \tag{3}$$

Gradiente para funções vetoriais

$$\nabla \vec{f} = \begin{bmatrix} \frac{\partial f_x}{\partial x} & \frac{\partial f_x}{\partial y} \\ \frac{\partial f_y}{\partial x} & \frac{\partial f_y}{\partial y} \end{bmatrix} \tag{4}$$

Fundamentos

$$\frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} + \frac{\nabla p}{\rho} = \vec{g} + \mu \nabla^2 \vec{u} \qquad \text{(1 revisitada)}$$

Laplaciano para funções escalares

$$\nabla^2 f = \nabla \cdot \nabla f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \qquad (5)$$

Laplaciano para funções vetoriais

$$\nabla^2 \vec{f} = (\nabla^2 f_1, \nabla^2 f_2) \qquad (6)$$

Fundamentos

$$\nabla \cdot \vec{u} = 0 \qquad \text{(2 revisitada)}$$

Divergente

$$\nabla \cdot \vec{f} = \frac{\partial f_x}{\partial x} + \frac{\partial f_y}{\partial y} \qquad (7)$$

Fundamentos

- *Pipeline* dos métodos PIC e FLIP

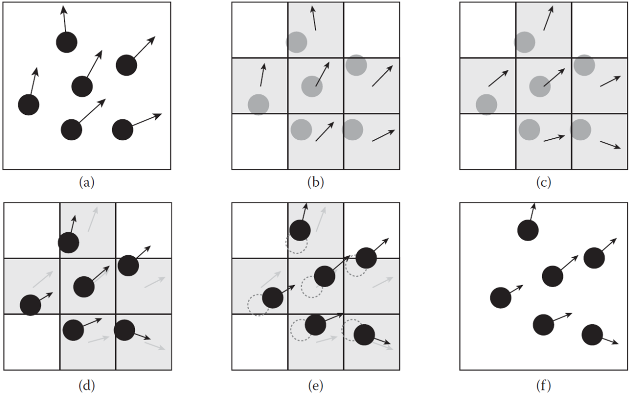


Figura: Imagem retirada de *Fluid Engine Development*, de Doyub Kim.

Fundamentos

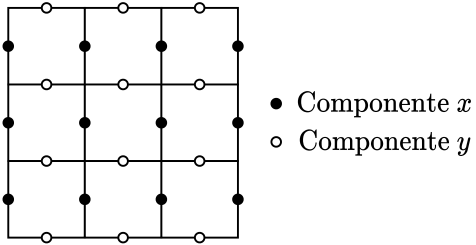


Figura: A velocidade do fluido é armazenada em uma grade regular com pontos nos centros da face.

Fundamentos

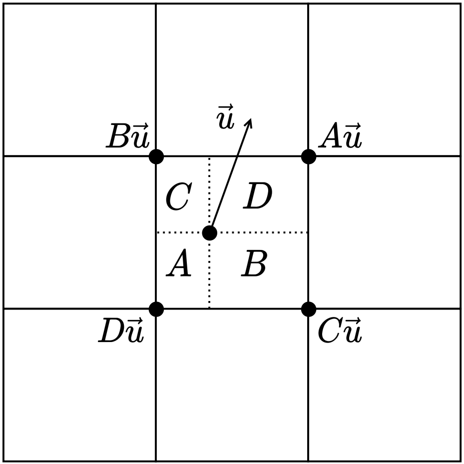


Figura: Transferência de partícula para grade.

- Porque usar o FLIP?

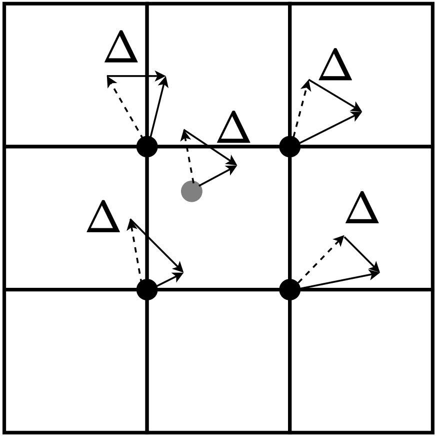


Figura: Transferência da grade para as partículas usada no método FLIP.

Fundamentos

- Como calcular operadores diferenciais com discretização em grade?

Fundamentos

- Diferenças Finitas

$$\frac{\partial f}{\partial x} \approx \frac{f^{i+1,j} - f^{i-1,j}}{2\Delta x} \tag{8}$$

$$\nabla f \approx \left(\frac{f^{i+1,j} - f^{i-1,j}}{2\Delta x}, \frac{f^{i,j+1} - f^{i,j-1}}{2\Delta y} \right) \tag{9}$$

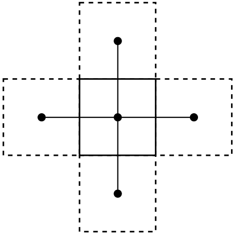


Figura: *Stencil* em uma grade regular.

Implementação

- C++ 17
- `CG`
- Módulo `Sparse` de `Eigen`
- Interface gráfica `ImGui`

Implementação

- Como solucionar as equações de Navier-Stokes?

Implementação

- Método de Euler implícito para resolver $\mu\nabla^2\vec{u}$
- Resolver sistemas $A \cdot x = b$ para x e y

$$\begin{bmatrix} c+1 & -c & 0 & \cdots & 0 \\ -c & 2c+1 & -c & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & -c & c+1 \end{bmatrix} \cdot f^{n+1} = f^n, \tag{10}$$

onde

$$c = \frac{\Delta t \mu}{\Delta h^2}. \tag{11}$$

Implementação

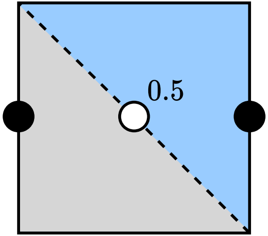


Figura: Método fracionário para interface sólido-fluido.

Implementação

Condições de Contorno

- Solução única para as equações de Navier-Stokes

Implementação

Condições de Contorno

- Solução única para as equações de Navier-Stokes
- Definem o fluxo em interfaces sólidas

Implementação

Condições de Contorno

- Solução única para as equações de Navier-Stokes
- Definem o fluxo em interfaces sólidas

Condição de Contorno de Newmann

$$\vec{u} \cdot \vec{n} = 0 \tag{12}$$

Implementação

Condições de Contorno

- Solução única para as equações de Navier-Stokes
- Definem o fluxo em interfaces sólidas

Condição de Contorno de Newmann

$$\vec{u} \cdot \vec{n} = 0 \tag{12}$$

Condição de Contorno de Dirichlet

$$\vec{u} = \max\left(1 - \lambda \frac{\max(-\vec{u} \cdot \vec{n}, 0)}{|\vec{u}_p|}, 0\right) \vec{u}_p \tag{13}$$

Implementação

- *Signed distance field*

Implementação

- *Signed distance field*
- Busca por partículas vizinhas

Implementação

- *Signed distance field*
- Busca por partículas vizinhas

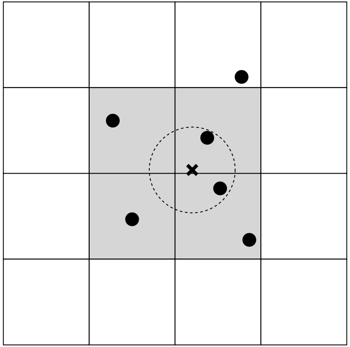


Figura: Busca de partículas vizinhas ao ponto **x** em um dado raio de busca.

Implementação

- Métricas
  - 40 classes
  - + 5000 linhas de código

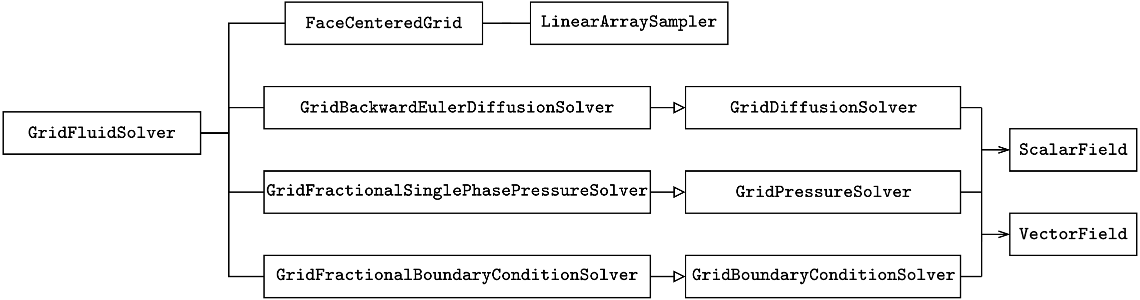


Figura: Classe base do solucionador PIC

Implementação

- Métricas
  - 40 classes
  - $+$ 5000 linhas de código

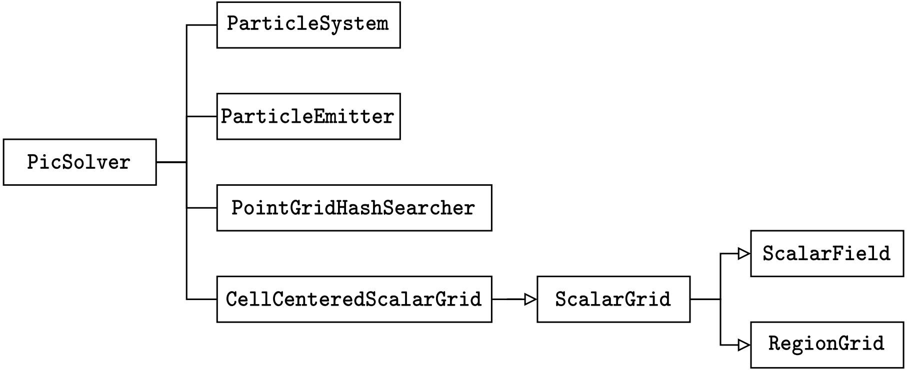


Figura: Classe base do solucionador FLIP

Resultados

Cena	n	t	F	μ	ρ
DB	10500	78	480	0	1
DB	10500	91	480	0.05	1
DDB	22051	99	480	0	1
WD	25073	126	480	0	1

Tabela: Números e parâmetros das simulações.

Obrigado!