コンピュータグラフィクス論

– アニメーション(3) –

2021年5月27日

高山 健志

休講日の変更

6月3日(木) → 6月17日(木)

流体アニメーション

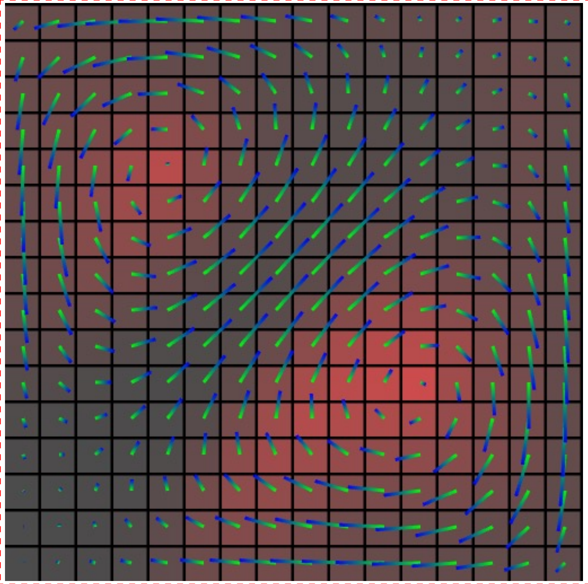
https://www.youtube.com/watch?v=KoEbwZq2ErU

https://www.youtube.com/watch?v=6WZZARzpckw

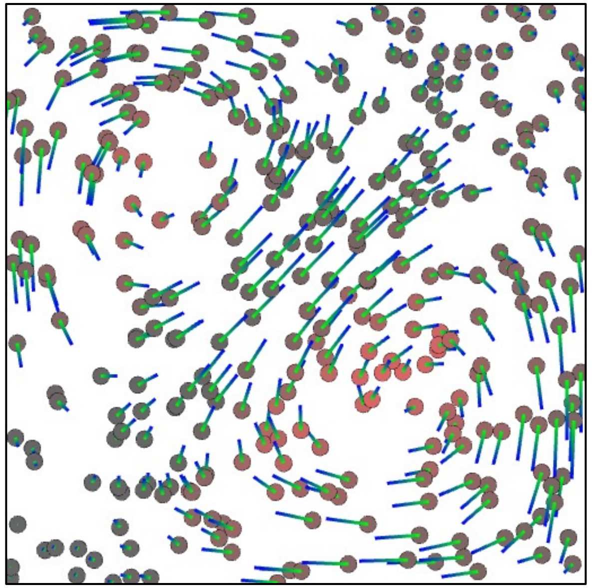
https://www.youtube.com/watch?v=WFwi0qLV8hQ

Position Based Fluids [Macklin SIGGRAPH13]
Detail-Preserving Paint Modeling for 3D Brushes [Chu NPAR10]

二つの異なるアプローチ



Eulerian
- 格子上のセルに速度とその他情報を保存
  - e.g. 煙の密度、温度
- 場の勾配等を計算しやすい ➔ 流体計算の王道
- オフライン用途に適する
  - リアルタイムにも使える



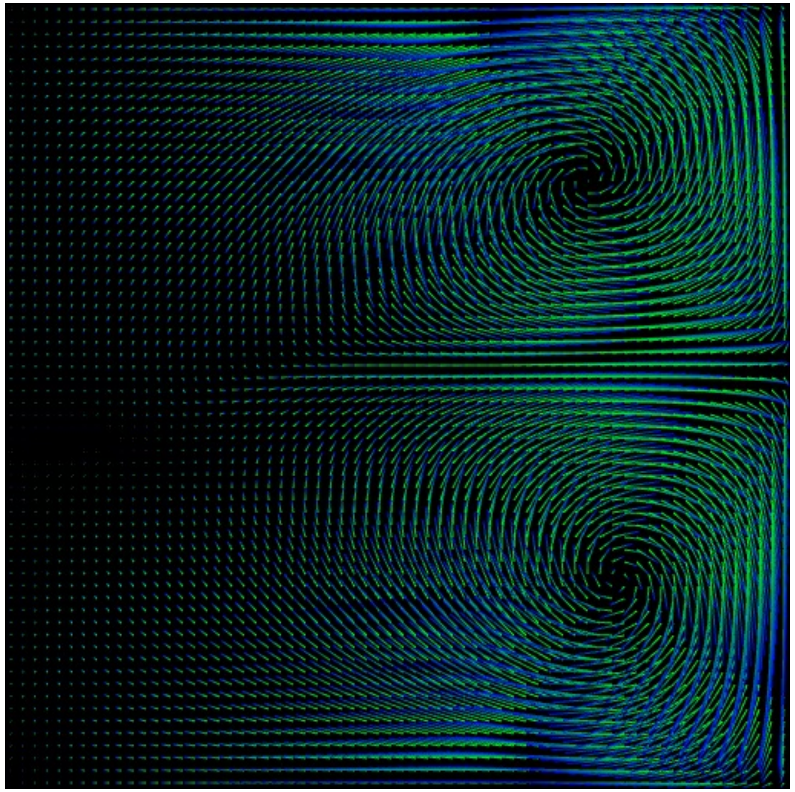
Lagrangian
- パーティクルに情報を持たせ、速度に従って動かす
- 場の勾配等の計算に工夫が必要 ➔ ハック (?)
- リアルタイム用途に適する

Stable Fluids [Stam, SIGGRAPH 99]

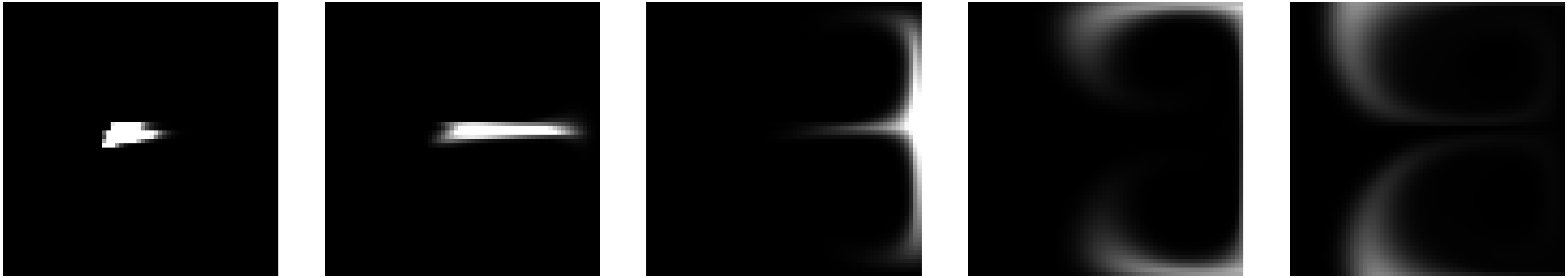
- 時間幅によらず無条件に安定 ➔ ゲーム向き
- ゲーム開発者向けの易しい解説記事
  - Real-Time Fluid Dynamics for Games (GDC 2003)
    - https://www.dgp.toronto.edu/public_user/stam/reality/Research/pdf/GDC03.pdf
  - 簡潔なサンプルコード (500行未満)
    - http://www.dgp.toronto.edu/people/stam/reality/Research/zip/CDROM_GDC03.zip
- 目標：これを理解できるようにする

定常的な速度場に沿った物理量の移流

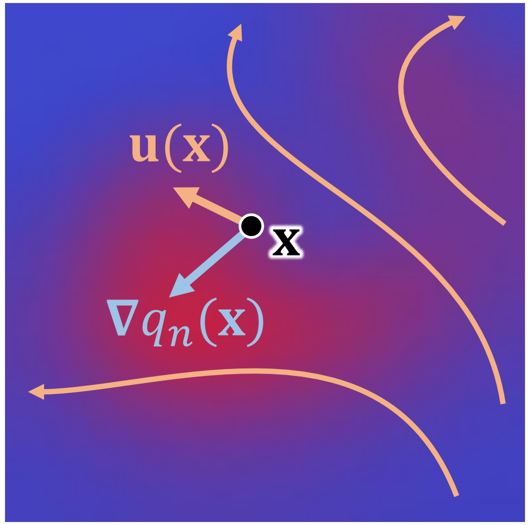
- 物理量：温度、煙の密度、etc
- 方法：
  - 陽的な方法 ➔ 不安定
  - Semi-Lagrangian 法 ➔ 安定



速度場



陽的な方法 [Foster 96]



- Given: 2D 領域上の (定常) 速度場 $\mathbf{u}: \mathbb{R}^2 \mapsto \mathbb{R}^2$

- ある物理量 q の、時刻 t における分布 $q_n: \mathbb{R}^2 \mapsto \mathbb{R}$ から、時刻 $t+h$ における分布 q_{n+1} を陽的に求める：

$$q_{n+1}(\mathbf{x}) = q_n(\mathbf{x}) - h\,\mathbf{u}(\mathbf{x}) \cdot \nabla q_n(\mathbf{x})$$

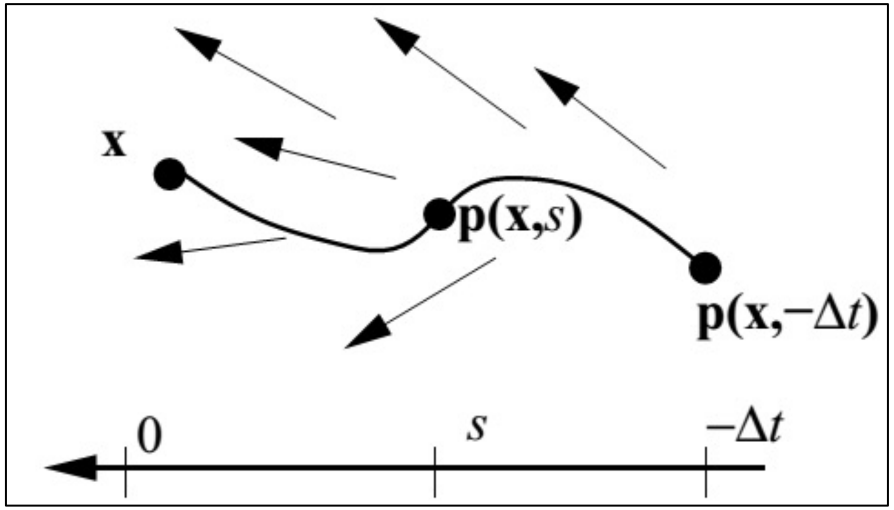
符号に注意！

- 変化量が時間幅 h に比例 ➔ h を大きくしすぎると発散する ☹

Realistic animation of liquids [Foster GMIP96]

Semi-Lagrangian 法 [Stam 99]

- 時刻 $t+h$ において位置 $\mathbf{x}$ に流れ着くパーティクルを仮に考える
- そのパーティクルの時刻 t における位置 $\tilde{\mathbf{x}}$ を求め、そこでの値を使う：



$$\tilde{\mathbf{x}} = \mathrm{trace}(\mathbf{u}, \mathbf{x}, -h)$$
$$q_{n+1}(\mathbf{x}) = q_n(\tilde{\mathbf{x}})$$

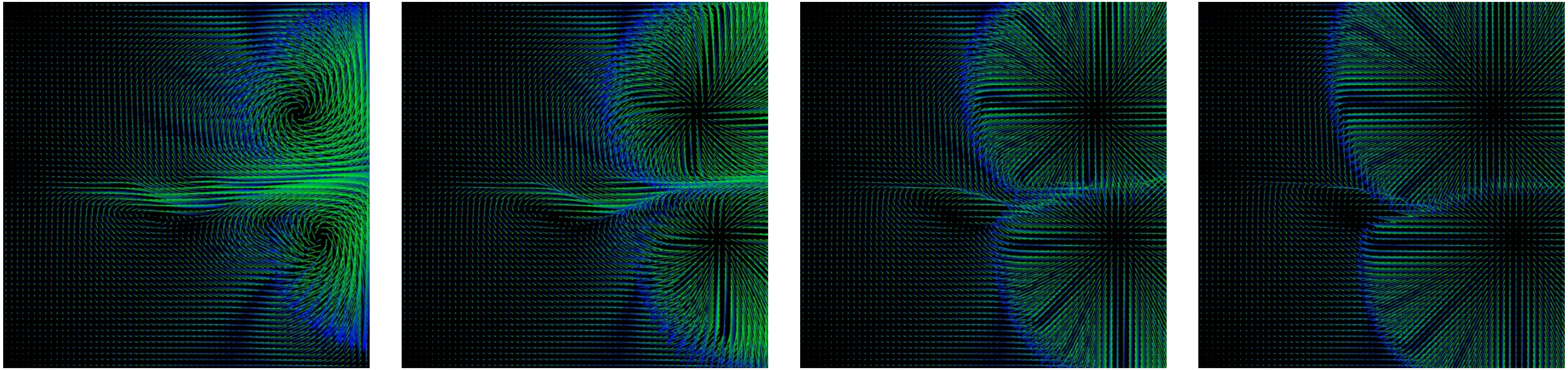
  - パーティクル自体のデータは不要
  - trace の方法：線形予測、Runge-Kutta, etc

- q_n をリサンプリングして q_{n+1} を求めるので、時間幅によらず安定！

速度場の動的な変化

- 他の物理量と同様、semi-Lagrangian 法で速度場そのものを移流
- が、そのままでは全然流体っぽくならない！

もっと渦を巻くべき！



リアルさのための条件：非圧縮性

$$\nabla \cdot \mathbf{u}(\mathbf{x}) = 0 \qquad \forall \mathbf{x}$$

- 至る所で発散がゼロ (divergence-free)
  - 各セルについて、隣接するセルとの間の流出量・流入量の合計がゼロ
  - 質量保存の法則とも言える

- 移流後のベクトル場 $\mathbf{w}$ は、一般に非圧縮性条件を満たさない！
  そこで、

$$\nabla \cdot (\mathbf{w} - \nabla q) = 0$$

Helmholtz 分解

となるスカラー場 q を求め、条件を満たす速度場 $\mathbf{u} = \mathbf{w} - \nabla q$ を得る

ポアソン (Poisson) 方程式

$$\boldsymbol{\nabla} \cdot (\mathbf{w} - \boldsymbol{\nabla} q) = 0$$
$$\Leftrightarrow$$
$$\Delta\, q = \boldsymbol{\nabla} \cdot \mathbf{w}$$

$\Delta = \boldsymbol{\nabla} \cdot \boldsymbol{\nabla}$

- q は以下のエネルギーを最小化 ➔ projection と呼ばれる

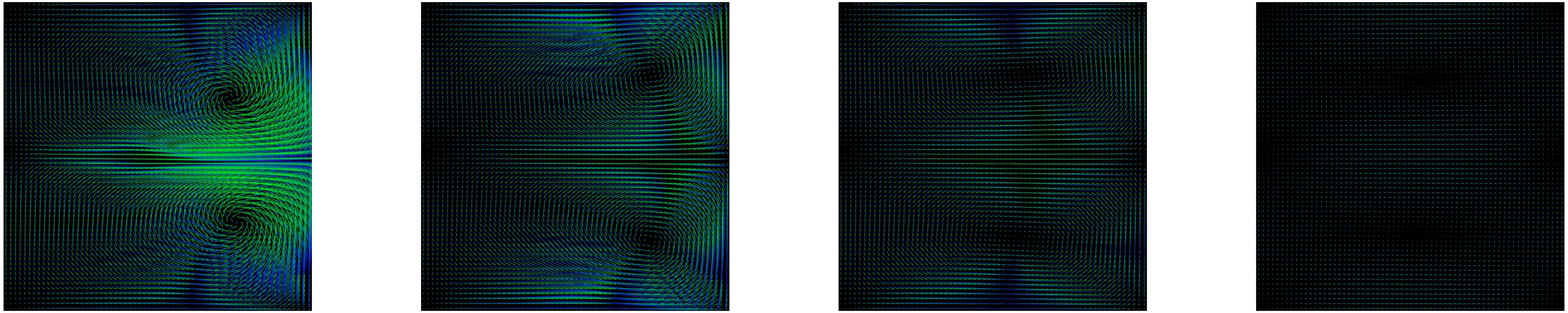
$$E(q) = \int_{\Omega} \|\mathbf{w} - \boldsymbol{\nabla} q\|^2$$

- 大規模疎行列で表される方程式
- よくある解法：
  - Gauss-Seidel ➔ 実装が簡単、遅い (サンプルコードで採用)
  - (Preconditioned) Conjugate Gradient ➔ 速い
  - Multigrid ➔ かなり速い、実装が大変 (?)

拡散

- 分布がより滑らかになろうとする効果

- 速度場に対して適用すると、粘性を表現できる



拡散方程式

$$\frac{\partial q}{\partial t} = \nu \, \Delta q$$

ν : 係数

- 陽的な解法

$$q_{n+1}(\mathbf{x}) = q_n(\mathbf{x}) + h \, \nu \, \Delta q_n(\mathbf{x})$$

  - 変化量が時間幅 h に比例 ➔ 不安定

- 陰的な解法

$$q_n(\mathbf{x}) = q_{n+1}(\mathbf{x}) - h \, \nu \, \Delta q_{n+1}(\mathbf{x})$$

  - 時間幅 h によらず安定
  - 大規模疎行列で表される方程式 (Poisson 方程式と同様)

非圧縮 Navier-Stokes 方程式

$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u}\cdot\nabla)\mathbf{u} - \frac{1}{\rho}\nabla p + \nu\Delta\mathbf{u} + \mathbf{f} \qquad \text{s.t.} \quad \nabla\cdot\mathbf{u} = 0$$

移流　圧力　粘性　外力

- 混乱しやすい点：

  $(\mathbf{u}\cdot\nabla) = \left(u_{\mathrm{x}}\frac{\partial}{\partial x} + u_{\mathrm{y}}\frac{\partial}{\partial y}\right)$ は微分演算子であり、発散 $\nabla\cdot\mathbf{u}$ とは意味が違う！

- x 成分：

$$\frac{\partial u_{\mathrm{x}}}{\partial t} = -\left(u_{\mathrm{x}}\frac{\partial u_{\mathrm{x}}}{\partial x} + u_{\mathrm{y}}\frac{\partial u_{\mathrm{x}}}{\partial y}\right) - \frac{1}{\rho}\frac{\partial p}{\partial x} + \nu\left(\frac{\partial^2 u_{\mathrm{x}}}{\partial x^2} + \frac{\partial^2 u_{\mathrm{x}}}{\partial y^2}\right) + f_{\mathrm{x}}$$

- Projection で求めたスカラー場 $q(\mathbf{x}) = p(\mathbf{x})/\rho$ は、圧力に相当
  - 圧力の高い所から低い所へ向かって加速度が発生

シミュレーションの流れ

- 速度場の更新 (vel_step)
  - 外力の加算
  - 拡散
  - Project
  - 移流
  - Project

速度場 $\mathbf{u}(\mathbf{x})$ の方程式

$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u} \cdot \nabla)\mathbf{u} - \frac{1}{\rho}\nabla p + \nu \Delta \mathbf{u} + \mathbf{f}$$

移流と拡散を行う前に project しておくこと！

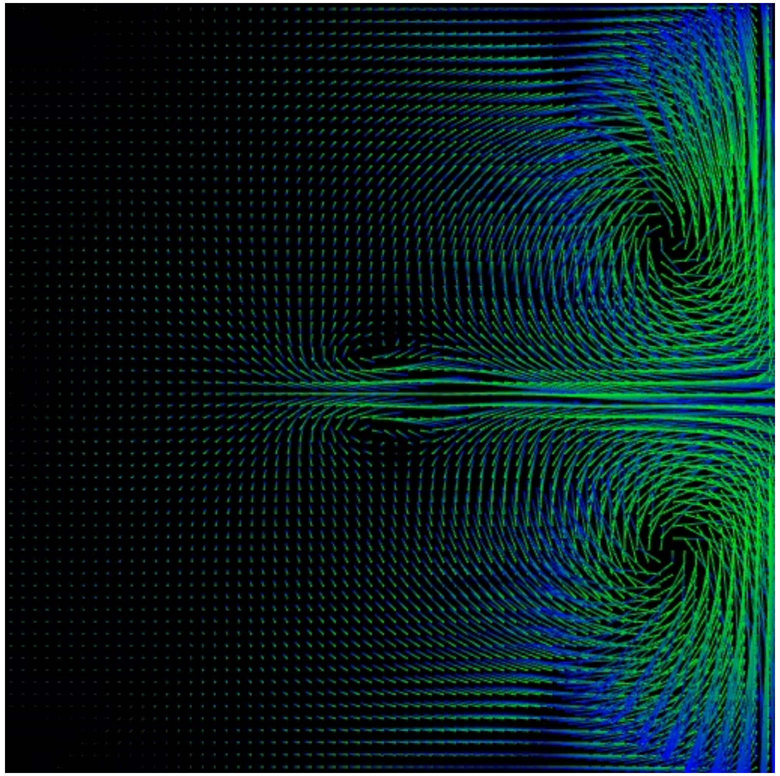
- 煙の密度場の更新 (dens_step)
  - 外部ソースの加算
  - 拡散
  - 移流

煙の密度場 $d(\mathbf{x})$ の方程式

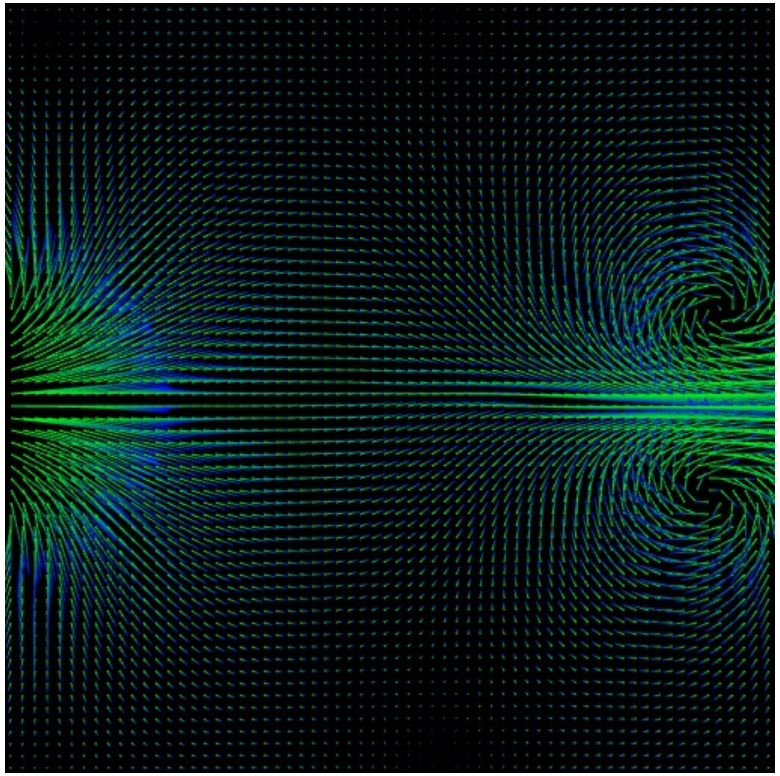
$$\frac{\partial d}{\partial t} = -(\mathbf{u} \cdot \nabla)d + \nu \Delta d + s$$

境界条件の設定

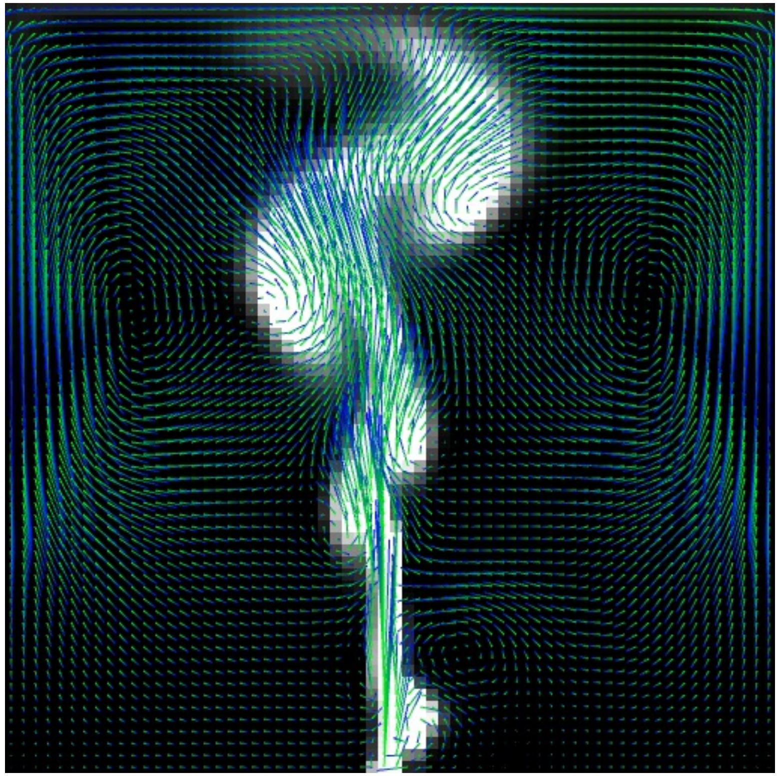
速度の壁方向の成分をゼロにする



左右と上下の壁を連続させる



一定の値を与え続ける

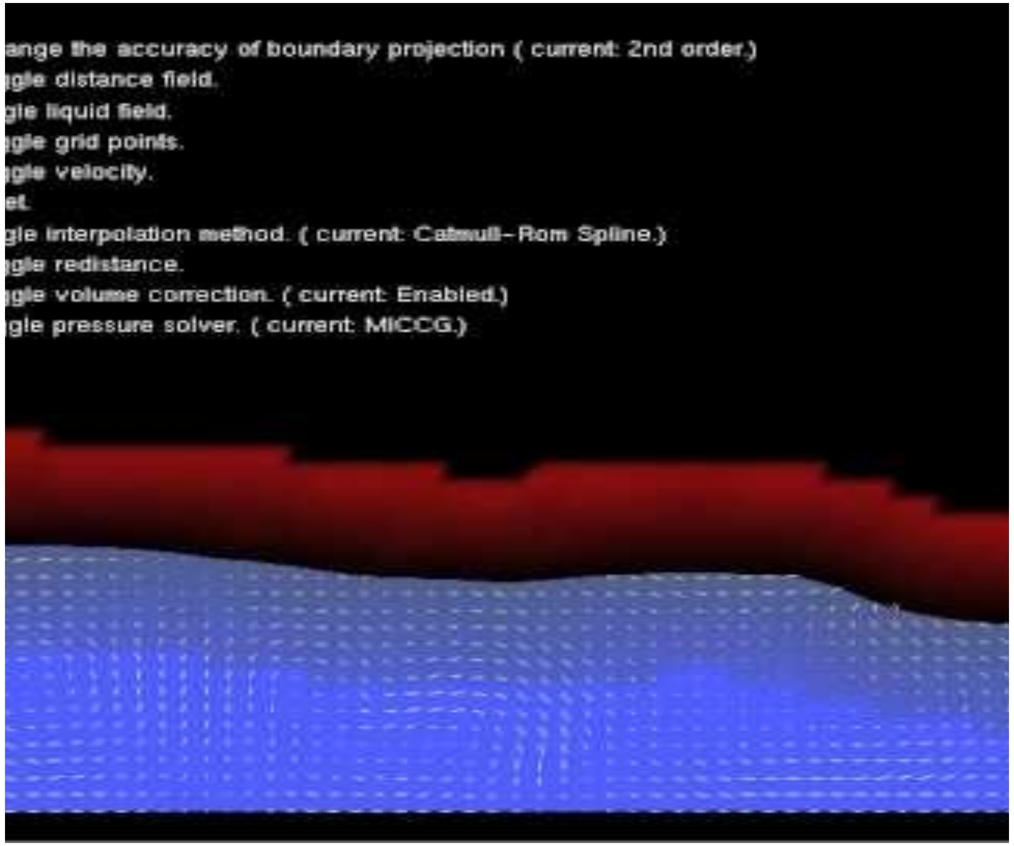


- より複雑なケースを扱うためには、高度な技術が必要
  - 丸みを帯びた形状、格子幅よりも薄いシート、etc

発展的な話題

レベルセット法による水面の表現

- 水面までの符号付き距離場 $\phi(\mathbf{x})$ を導入
  - $\phi(\mathbf{x}) < 0$ なら液体、$\phi(\mathbf{x}) > 0$ なら空気
  - 初期状態を適当に与える
- 速度場に従って $\phi(\mathbf{x})$ を移流
- 圧力計算の際、水面 $\phi(\mathbf{x}) = 0$ において $p(\mathbf{x}) = 0$ という境界条件を設定



https://www.youtube.com/watch?v=Ss89OpQ_u54
http://code.google.com/p/levelset2d/

Practical animation of liquids [Foster SIGGRAPH01]

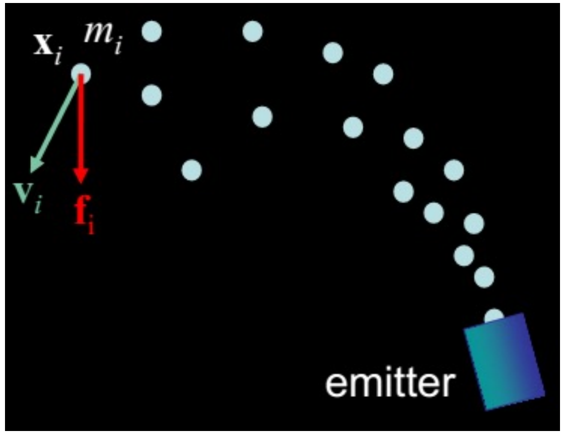
いろいろな移流アルゴリズム

- Semi-Lagrangian
- Upwind
- MacCormack
- WENO5
- QUICK

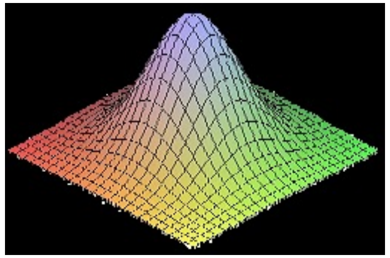
https://www.youtube.com/watch?v=UYu8JgbbHpE
http://code.google.com/p/2dsmoke/

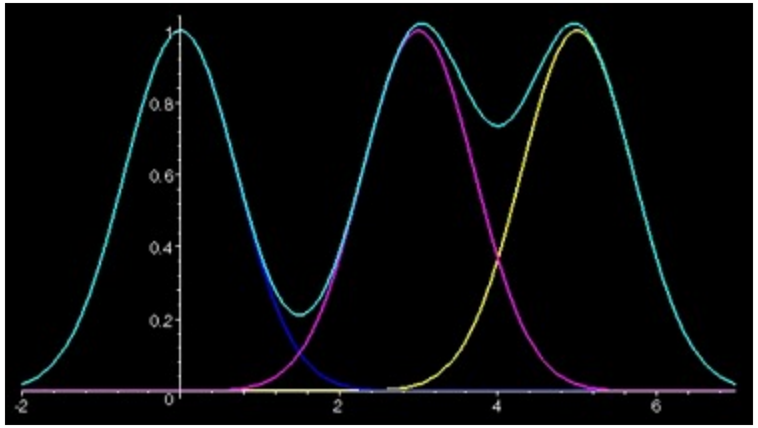
https://en.wikipedia.org/wiki/MacCormack_method
https://en.wikipedia.org/wiki/Upwind_scheme

Smoothed Particle Hydrodynamics



- Lagrangian 法の代表格
- 質量を持った粒子を速度に従って動かす
- Smoothing kernel により粒子群から連続的な場を定義
  - $W(r) = \frac{315}{64\pi h^9}(h^2 - r^2)^3$
  - 密度場： $\rho(\mathbf{x}) = \sum_j m_j W(\|\mathbf{x} - \mathbf{x}_j\|)$
  - 速度場： $\mathbf{u}(\mathbf{x}) = \sum_j \frac{m_j}{\rho(\mathbf{x}_j)} \mathbf{u}_j\, W(\|\mathbf{x} - \mathbf{x}_j\|)$





- 一階微分と二階微分は ∇W と ΔW から求まる

Particle-Based Fluid Simulation for Interactive Applications [Muller SCA03]
https://slideplayer.com/slide/4790539/

Smoothed **P**article **H**ydrodynamics

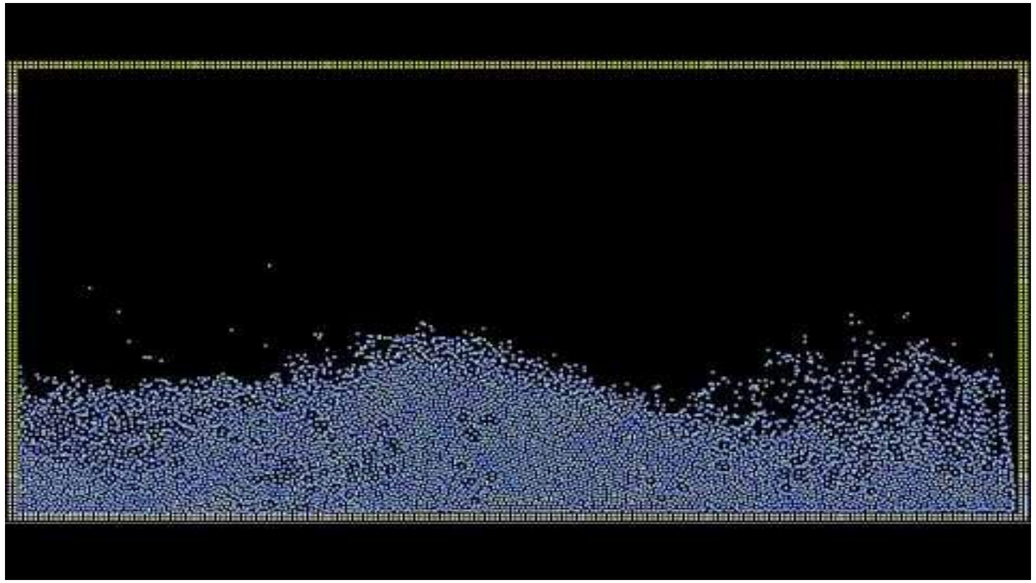
- 粒子に働く力：

$$\underbrace{-\frac{1}{\rho(\mathbf{x}_i)}\boldsymbol{\nabla} p(\mathbf{x}_i)}_{\text{圧力}} + \underbrace{\nu\Delta\mathbf{u}(\mathbf{x}_i)}_{\text{粘性}} + \underbrace{\mathbf{f}}_{\text{外力}}$$

- 理想気体の状態方程式 $pV = NRT$ より、圧力は密度に比例：

$$p(\mathbf{x}) = k\,\rho(\mathbf{x})$$

➔ ポアソン方程式を解く必要が無い！



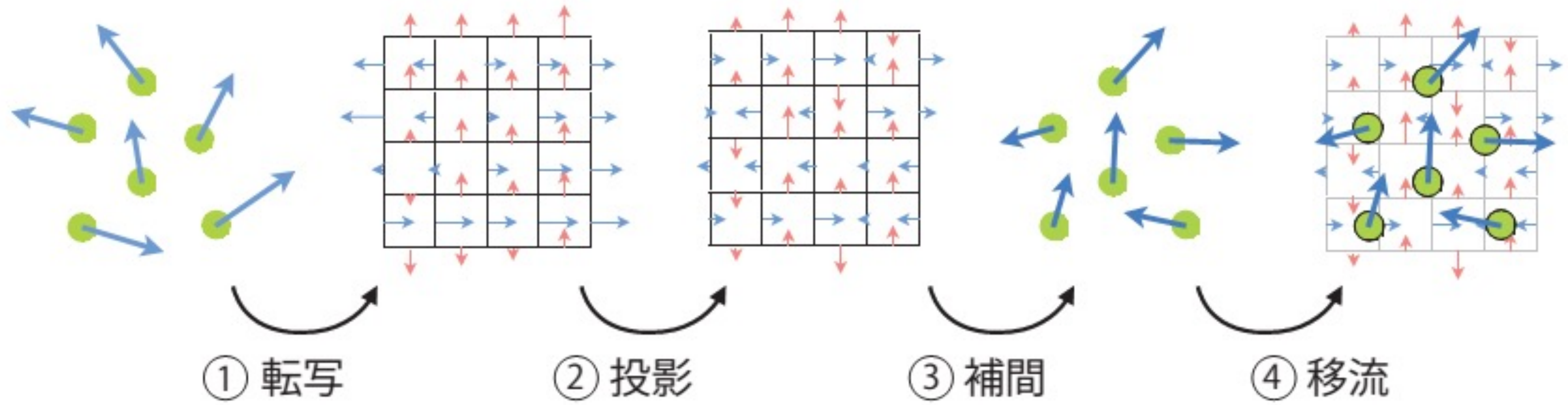
https://www.youtube.com/watch?v=M8WPINWAWPY

Particle-Based Fluid Simulation for Interactive Applications [Muller SCA03]
https://slideplayer.com/slide/4790539/

格子法と粒子法のハイブリッド

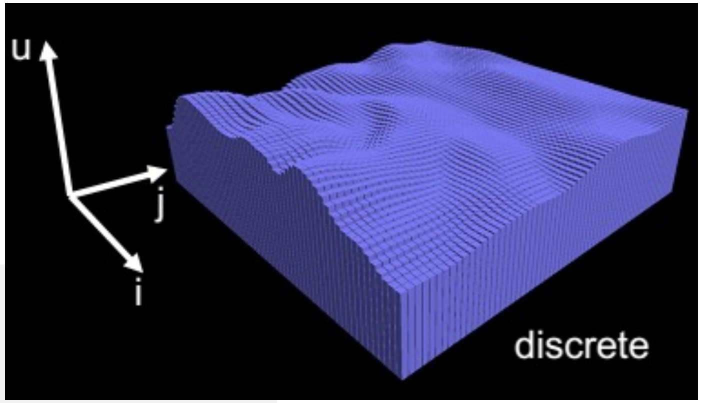
	格子法	粒子法
移流計算	数値拡散する ☹	数値拡散しない ☺
圧力計算	正確 ☺	不正確 ☹

- PIC (**P**article **I**n **C**ell) 法と FLIP (**FL**uid **I**mplicit **P**article) 法



Animating sand as a fluid [Zhu,Bridson,SIGGRAPH05]

Height field による水面の近似




```
initialize u[i,j] as you like
set v[i,j] = 0
loop
    v[i,j] +=(u[i-1,j] + u[i+1,j] + u[i,j-1] + u[i,j+1])/4 – u[i,j]
    v[i,j] *= 0.99
    u[i,j] += v[i,j]
endloop
```

Real Time Fluids in Games (by Matthias Müller)
https://slideplayer.com/slide/4790539/

- WebGL実装
  - http://madebyevan.com/webgl-water/
  - http://dblsai.github.io/WebGL-Fluid/

参考情報

- JavaScriptによる実装
  - http://www.ibiblio.org/e-notes/webgl/gpu/fluid.htm
  - https://nerget.com/fluidSim/
  - http://dev.miaumiau.cat/sph/
  - http://p.brm.sk/fluid/
- C++による実装
  - http://code.google.com/p/flip3d/
  - http://code.google.com/p/levelset2d/
  - http://code.google.com/p/smoke3d/
  - http://code.google.com/p/2dsmoke/
  - http://www.cs.ubc.ca/~rbridson/download/simple_flip2d.tar.gz
- Shiokaze Framework: https://shiokaze.ryichando.graphics/
- 書籍
  - Fluid Simulation for Computer Graphics, by R. Bridson, 2008
  - The Art of Fluid Animation, by Jos Stam, 2015
  - 安東遼一氏による Computer Graphics Gems JP 2012 の記事
    - Chapter 13: ベクタ形式で出力可能な美しいマーブリング模様の生成法
    - Chapter 14: FLIP法による格子&粒子のハイブリッド流体シミュレーション
    - 付録コード：http://book.borndigital.jp/support/CGGems2012/CGGems2012.zip