

## 1 物理模拟

万事万物的运动都遵循一定的原理. 经过物理学家们长久以来的努力, 我们已经可以用一系列物理方程描述物体的运动, 但这些方程往往复杂而难以求解. 于是, 人们开始寻求这些方程的近似解, 并且希望这些近似解足够精确, 以至于可以用来模拟现实世界中的物理现象. 这种通过数值方法求解物理方程, 并用计算机进行模拟的过程, 就称为**物理模拟 (Physical Simulation)**.

一般而言, 物理模拟的范围主要是宏观低速物体的运动, 因此主要使用牛顿运动定律进行运动的计算. 牛顿运动定律为

$$\mathbf{f} = m\mathbf{a} = m \frac{d\mathbf{v}}{dt} = m \frac{d^2\mathbf{x}}{dt^2}$$

在上述公式中, 我们的研究对象是一个质点, 并且它仅受一个力. 这个例子实在过于简单, 无法代表一般的情形. 通常而言, 物体包含无穷多粒子, 并且往往存在多个外力以及复杂的内力. 为了将这样的复杂的系统变为数值求解问题, 我们首先需要对物体进行离散化处理.

因此, 物理模拟的过程大致可以分为三个步骤: **空间离散化**, **时间离散化**以及**数值计算**. 下面就来介绍物理模拟在各种模型下的实现方法.

## 2 弹簧质点模型

在本节中, 我们将以弹簧质点模型为例介绍物理模拟的一般方法. 同时, 这一模型也是计算机图形学中常用的物理模拟模型之一, 研究其仿真办法也具有相当的实际意义.

### 2.1 空间离散化

通过一定的方法, 我们可以把物体抽象成由有限个质点组成的系统. 单个质点的状态由其位置  $\mathbf{x}_i$  和速度  $\mathbf{v}_i$ , 而为了考虑其运动状态的变化, 我们还需通过其受的力  $\mathbf{f}_i$  和质量  $m_i$  得出其加速度. 因此, 一个质点模型可以由列表  $\{\mathbf{x}_i, \mathbf{v}_i, \mathbf{f}_i, m_i\}_{i=1}^N$  表示.

在抽象成质点的情形下, 弹簧质点模型假定质点之间存在弹簧以传递内力. 质点  $i$  受的外力可以写做

$$\mathbf{f}_i = \sum_{j \in \Omega(i)} \mathbf{f}_{ij} + \mathbf{f}_i^{\text{ext}}$$

其中  $\mathbf{f}_i^{\text{ext}}$  表示质点  $i$  受到的除去弹簧弹力之外的外力. 质点  $i$  与其邻接的质点  $j \in \Omega(i)$  通过弹簧相连, 弹簧对质点  $i$  的弹力  $\mathbf{f}_{ij}$  可以通过胡克定律计算:

$$\mathbf{f}_{ij} = -k_{ij}(|\mathbf{x}_i - \mathbf{x}_j| - l_{ij}) \frac{\mathbf{x}_i - \mathbf{x}_j}{|\mathbf{x}_i - \mathbf{x}_j|}$$

其中  $k_{ij}$  为弹簧的弹性系数,  $l_{ij}$  为弹簧的原长.

## 2.2 时间离散化

物体的运动状态是随时间连续变化的. 同样地, 我们也需要对时间进行离散化. 一般而言, 我们会在时间区间上均匀采样, 采样间隔为  $\Delta t$ . 我们把  $N$  个质点的位置  $\mathbf{x}$  和速度  $\mathbf{v}$  写做  $3N$  维的堆叠向量的形式, 在第  $k$  次采样  $t_k$  时有:

$$\mathbf{x}(t_k) = \begin{bmatrix} \mathbf{x}_1(t_k) \\ \dots \\ \mathbf{x}_n(t_k) \end{bmatrix}, \quad \mathbf{v}(t_k) = \begin{bmatrix} \mathbf{v}_1(t_k) \\ \dots \\ \mathbf{v}_n(t_k) \end{bmatrix}$$

其中  $\mathbf{x}_i(t)$  表示  $t$  时刻质点  $i$  的位置,  $\mathbf{v}_i(t)$  同理. 根据简单的运动学知识, 我们可以得到下一次采样时的位置矩阵  $\mathbf{x}(t_{k+1})$  和  $\mathbf{v}(t_{k+1})$ :

$$\begin{aligned} \mathbf{x}(t_{k+1}) &= \mathbf{x}(t_k) + \int_{t_k}^{t_{k+1}} \mathbf{v}(t) dt \\ \mathbf{v}(t_{k+1}) &= \mathbf{v}(t_k) + \mathbf{M}^{-1} \int_{t_k}^{t_{k+1}} \mathbf{f}(t, \mathbf{x}(t), \mathbf{v}(t)) dt \end{aligned}$$

其中  $\mathbf{M} = \text{diag}\{m_1, m_1, m_1, \dots, m_N, m_N, m_N\}$  为质点的质量矩阵.

现在, 我们的目标就是求解上述积分方程. 典型的计算时间积分的办法有**显式欧拉 (Explicit Euler)** 积分和**隐式欧拉 (Implicit Euler)** 积分.

### 2.2.1 显式欧拉积分

在复杂的情形中, 上述积分方程可能不存在解析表达. 好在当取样的时间间隔较小时, 我们可以将被积函数视作常数, 取值即为每个时间间隔开始的时刻, 于是即有

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + \mathbf{v}(t_k) \Delta t$$

$$\mathbf{v}(t_{k+1}) = \mathbf{v}(t_k) + \mathbf{M}^{-1} \mathbf{f}(t_k) \Delta t$$

这样, 对于弹簧质点系统的模拟就归结于上述迭代过程. 显式欧拉积分的优点在于计算和逻辑简单, 每一步只需要计算当前时刻的力即可.

然而, 这样的简单近似也有其缺点. 当时间步长较大时,  $\mathbf{v}(t)$  和  $\mathbf{f}(t)$  可能在时间间隔内变化较大, 从而导致解得的  $\mathbf{x}(t)$  和  $\mathbf{v}(t)$  偏离实际情况, 使得系统能量升高. 这又进一步导致  $\mathbf{v}(t)$  和  $\mathbf{f}(t)$  的偏离, 最终可能导致系统发散, 无法继续模拟下去.

解决上述问题的一种办法是使用更小的时间步长, 但这会大大增加计算量, 反而得不偿失. 因此, 我们需要一种更稳定的办法.

### 2.2.2 隐式欧拉积分

**隐式欧拉积分的推导** 与显式欧拉积分不同, 隐式欧拉积分在计算下一个时间步长的状态时, 使用的是每个时间间隔结束的時刻的力. 即有

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + \mathbf{v}(t_{k+1}) \Delta t$$

$$\mathbf{v}(t_{k+1}) = \mathbf{v}(t_k) + \mathbf{M}^{-1} \mathbf{f}(t_{k+1}) \Delta t$$

这样, 我们需要解一个隐式方程组. 将  $\mathbf{v}(t_{k+1})$  代入  $\mathbf{x}(t_{k+1})$  中可得

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + (\mathbf{v}(t_k) + \mathbf{M}^{-1} \mathbf{f}(t_{k+1}) \Delta t) \Delta t$$

为了方便考虑, 我们把力  $\mathbf{f}$  分成内力和外力两部分:

$$\mathbf{f}(t_{k+1}) = \mathbf{f}_{\text{int}}(t_{k+1}) + \mathbf{f}_{\text{ext}}$$

其中  $\mathbf{f}_{\text{ext}}$  作为外力与时间和质点的位置无关. 于是进一步整理可得

$$\mathbf{x}(t_{k+1}) = [\mathbf{x}(t_k) + \Delta t \mathbf{v}(t_k) + (\Delta t)^2 \mathbf{M}^{-1} \mathbf{f}_{\text{ext}}] + (\Delta t)^2 \mathbf{M}^{-1} \mathbf{f}_{\text{int}}(t_{k+1})$$

等号右边前半部分全部为已知量, 不妨记作  $\mathbf{y}(t_k)$ . 于是方程组即为

$$\mathbf{x}(t_{k+1}) - \mathbf{y}(t_k) - (\Delta t)^2 \mathbf{M}^{-1} \mathbf{f}_{\text{int}}(t_{k+1}) = \mathbf{0}$$

接下来, 我们需要将内力  $\mathbf{f}_{\text{int}}(t_{k+1})$  表示成位置  $\mathbf{x}(t_{k+1})$  的函数. 对于质点  $i$  与  $j$  之间的弹簧, 其弹性势能

$$E_{ij} = \frac{1}{2} k_{ij} (\|\mathbf{x}_j - \mathbf{x}_i\| - l_{ij})^2$$

于是

$$\mathbf{f}_{ij} = -\frac{\partial E_{ij}}{\partial \mathbf{x}_i}$$

从而对于质点  $i$ , 其内力  $\mathbf{f}_{i,\text{int}}$  就可以写作

$$\mathbf{f}_{i,\text{int}} = -\sum_{j \in \Omega(i)} \frac{\partial E_{ij}}{\partial \mathbf{x}_i} = -\frac{\partial E}{\partial \mathbf{x}_i}$$

其中  $E$  为弹簧系统的总能量 (与  $i$  不相连的弹簧的势能对  $\mathbf{x}_i$  的梯度为  $\mathbf{0}$ , 因此可以并入求和项中), 其自变量为各质点的位置向量  $\mathbf{x}$ . 将所有质点的内力堆叠起来, 我们就有

$$\mathbf{x}(t_{k+1}) - \mathbf{y}(t_k) + (\Delta t)^2 \mathbf{M}^{-1} \frac{\partial E(\mathbf{x}(t_{k+1}))}{\partial \mathbf{x}} = \mathbf{0}$$

于是, 可以构造辅助函数

$$g(\mathbf{x}) = \frac{1}{2(\Delta t)^2} (\mathbf{x} - \mathbf{y}(t_k))^t \mathbf{M} (\mathbf{x} - \mathbf{y}(t_k)) + E(\mathbf{x})$$

则有

$$\frac{\partial g(\mathbf{x})}{\partial \mathbf{x}} = \mathbf{x}(t_{k+1}) - \mathbf{y}(t_k) + (\Delta t)^2 \mathbf{M}^{-1} \frac{\partial E(\mathbf{x}(t_{k+1}))}{\partial \mathbf{x}} = \mathbf{0}$$

于是, 目标就转化为最小化问题

$$\mathbf{x}(t_{k+1}) = \arg \min_{\mathbf{x}} g(\mathbf{x})$$

**最小化问题的数值求解** 我们采取牛顿法解决上述问题. 牛顿法的基本思想是在每轮迭代中使用一个二次函数逼近目标函数, 然后求解该二次函数的极小值作为下一次迭代的点. 迭代过程每次通过尝试解  $\mathbf{x}^{(i)}$  计算下一次的解  $\mathbf{x}^{(i+1)}$ . 对  $g(\mathbf{x})$  在  $\mathbf{x}^{(i)}$  处进行二阶泰勒展开, 有

$$g(\mathbf{x}) = g(\mathbf{x}^{(i)}) + \nabla g(\mathbf{x}^{(i)}) \cdot (\mathbf{x} - \mathbf{x}^{(i)}) + \frac{1}{2}(\mathbf{x} - \mathbf{x}^{(i)})^t \mathbf{H}_g(\mathbf{x}^{(i)})(\mathbf{x} - \mathbf{x}^{(i)}) + o(\|\mathbf{x} - \mathbf{x}^{(i)}\|^3)$$

其中  $\nabla = \frac{\partial}{\partial \mathbf{x}}$ ,  $\mathbf{H}_g$  为  $g$  的 Hessian 矩阵. 忽略高阶无穷小量, 下一轮迭代的解  $\mathbf{x}^{(i+1)}$  应当为上述二次函数的极小值点. 于是对上式两边求梯度并代入  $\mathbf{x}^{(i+1)}$  可得

$$\mathbf{0} = \nabla g(\mathbf{x}^{(i+1)}) = \nabla g(\mathbf{x}^{(i)}) + \mathbf{H}_g(\mathbf{x}^{(i)})(\mathbf{x}^{(i+1)} - \mathbf{x}^{(i)})$$

于是下一轮迭代的解即为线性方程组

$$\mathbf{H}_g(\mathbf{x}^{(i)})(\mathbf{x}^{(i+1)} - \mathbf{x}^{(i)}) = -\nabla g(\mathbf{x}^{(i)})$$

的解. 对于弹簧质点系统, 我们认为  $g(\mathbf{x})$  的性质比较好, 只需一次迭代就能求得较为准确的解. 于是就有

$$\mathbf{H}_g(\mathbf{x}(t_k))(\mathbf{x}(t_{k+1}) - \mathbf{x}(t_k)) = -\nabla g(\mathbf{x}(t_k))$$

根据  $g(\mathbf{x})$  的定义可知

$$\begin{aligned} \nabla g(\mathbf{x}(t_k)) &= \frac{1}{(\Delta t)^2} \mathbf{M}(\mathbf{x}(t_k) - \mathbf{y}(t_k)) + \nabla E(\mathbf{x}(t_k)) \\ \mathbf{H}_g(\mathbf{x}(t_k)) &= \frac{1}{(\Delta t)^2} \mathbf{M} + \mathbf{H}_E(\mathbf{x}(t_k)) \end{aligned}$$

其中  $\mathbf{H}_E(\mathbf{x})$  为能量函数  $E(\mathbf{x})$  的 Hessian 矩阵.

对于  $\nabla E(\mathbf{x}(t_k))$ , 我们可以把它写成对各个质点的位置  $\mathbf{x}_i$  的梯度  $\nabla_i E(\mathbf{x}) = \frac{\partial E(\mathbf{x})}{\partial \mathbf{x}_i}$  的堆叠向量, 并且我们已经知道质点  $i$  受到的内力之和等于能量的负梯度, 于是

$$\nabla E(\mathbf{x}(t_k)) = \begin{bmatrix} \nabla_1 E(\mathbf{x}) \\ \vdots \\ \nabla_n E(\mathbf{x}) \end{bmatrix}_{\mathbf{x}=\mathbf{x}(t_k)} = - \begin{bmatrix} \mathbf{f}_{1,\text{int}}(t_k) \\ \vdots \\ \mathbf{f}_{n,\text{int}}(t_k) \end{bmatrix}$$

接下来考虑  $\mathbf{H}_E(\mathbf{x}(t_k))$ . 根据弹簧质点模型的定义, 系统的总势能为各个弹簧势能之和:

$$E(\mathbf{x}) = \sum_{(i,j)} E_{ij}(\mathbf{x})$$

其中  $(i, j)$  表示以弹簧相连的质点. 于是

$$\mathbf{H}_E(\mathbf{x}) = \sum_{(i,j)} \mathbf{H}_{E_{ij}}(\mathbf{x})$$

其中  $\mathbf{H}_{E_{ij}}(\mathbf{x})$  为单个弹簧势能  $E_{ij}(\mathbf{x})$  的 Hessian 矩阵. 根据定义  $E_{ij} = \frac{1}{2}k_{ij}(\|\mathbf{x}_j - \mathbf{x}_i\| - l_{ij})^2$  有

$$\mathbf{H}_{ij} := \frac{\partial^2 E_{ij}(\mathbf{x})}{\partial \mathbf{x}_i^2} = k_{ij} \frac{(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^t}{\|\mathbf{x}_i - \mathbf{x}_j\|^2} + k_{ij} \left(1 - \frac{l_{ij}}{\|\mathbf{x}_i - \mathbf{x}_j\|}\right) \left(\mathbf{I} - \frac{(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^t}{\|\mathbf{x}_i - \mathbf{x}_j\|^2}\right)$$

$$\frac{\partial^2 E_{ij}(\mathbf{x})}{\partial \mathbf{x}_j^2} = \mathbf{H}_{ij}, \quad \frac{\partial^2 E_{ij}(\mathbf{x})}{\partial \mathbf{x}_i \partial \mathbf{x}_j} = -\mathbf{H}_{ij}$$

对于  $E_{ij}(\mathbf{x})$  而言, 其值仅与  $\mathbf{x}$  中第  $i$  个向量  $\mathbf{x}_i$  和第  $j$  个向量  $\mathbf{x}_j$  有关. 于是将  $E_{ij}(\mathbf{x})$  的 Hessian 矩阵  $\mathbf{H}_{E_{ij}}(\mathbf{x})$  视作由  $N \times N$  个  $3 \times 3$  子矩阵组成的矩阵, 只有  $(i, i), (j, j), (i, j), (j, i)$  四个子矩阵非零, 且分别为  $\mathbf{H}_{ij}, \mathbf{H}_{ij}, -\mathbf{H}_{ij}, -\mathbf{H}_{ij}$ . 这就求得了  $\mathbf{H}_E$ , 于是再代入前述公式即可得隐式欧拉积分的迭代公式.

**过程总结** 最后, 我们把隐式欧拉积分求解弹簧质点系统的过程总结如下:

1. 遍历系统中的所有弹簧, 然后进行如下两个操作:

a. 根据弹力的计算公式

$$\mathbf{f}_{ij} = -k_{ij}(|\mathbf{x}_i - \mathbf{x}_j| - l_{ij}) \frac{\mathbf{x}_i - \mathbf{x}_j}{|\mathbf{x}_i - \mathbf{x}_j|}$$

计算弹簧所连接的两个质点  $i, j$  的内力  $\mathbf{f}_{ij}, \mathbf{f}_{ji}$ , 并将其累加到质点  $i, j$  的总内力  $\mathbf{f}_{i,\text{int}}, \mathbf{f}_{j,\text{int}}$  上.

b. 根据弹簧的 Hessian 矩阵计算公式

$$\mathbf{H}_{ij} = \frac{\partial^2 E_{ij}(\mathbf{x})}{\partial \mathbf{x}_i^2} = k_{ij} \frac{(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^t}{\|\mathbf{x}_i - \mathbf{x}_j\|^2} + k_{ij} \left(1 - \frac{l_{ij}}{\|\mathbf{x}_i - \mathbf{x}_j\|}\right) \left(\mathbf{I} - \frac{(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^t}{\|\mathbf{x}_i - \mathbf{x}_j\|^2}\right)$$

计算质点  $i, j$  对应的 Hessian 子矩阵, 并将其累加到系统总能量的 Hessian 矩阵  $\mathbf{H}_E$  的对应位置上.

完成上述两步后即可求出  $\nabla E$  和  $\mathbf{H}_E$ .

2. 根据  $\mathbf{y}(t_k)$  的定义

$$\mathbf{y}(t_k) = \mathbf{x}(t_k) + \Delta t \mathbf{v}(t_k) + (\Delta t)^2 \mathbf{M}^{-1} \mathbf{f}_{\text{ext}}$$

求出  $\mathbf{y}(t_k)$ .

3. 根据  $\nabla g(\mathbf{x}(t_k))$  和  $\mathbf{H}_g(\mathbf{x}(t_k))$  的定义

$$\nabla g(\mathbf{x}(t_k)) = \frac{1}{(\Delta t)^2} \mathbf{M}(\mathbf{x}(t_k) - \mathbf{y}(t_k)) + \nabla E(\mathbf{x}(t_k))$$

$$\mathbf{H}_g(\mathbf{x}(t_k)) = \frac{1}{(\Delta t)^2} \mathbf{M} + \mathbf{H}_E(\mathbf{x}(t_k))$$

求出  $\nabla g(\mathbf{x}(t_k))$  和  $\mathbf{H}_g(\mathbf{x}(t_k))$ .

4. 求解线性方程组

$$\mathbf{H}_g(\mathbf{x}(t_k))(\mathbf{x}(t_{k+1}) - \mathbf{x}(t_k)) = -\nabla g(\mathbf{x}(t_k))$$

得到  $\boldsymbol{\delta} = \mathbf{x}(t_{k+1}) - \mathbf{x}(t_k)$ .

5. 根据隐式欧拉的基本公式

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + \boldsymbol{\delta}$$

$$\mathbf{v}(t_{k+1}) = \frac{\boldsymbol{\delta}}{\Delta t}$$

计算更新后的位置  $\mathbf{x}(t_{k+1})$  和速度  $\mathbf{v}(t_{k+1})$ . 如此就完成了在一个时间间隔内的模拟.

## 3 流体模拟

### 3.1 流体的基本性质

**定理 3.1 Navier-Stokes 方程** 流体的运动遵循 Navier-Stokes 方程, 其表达式为

$$\rho \frac{D\mathbf{v}}{Dt} = -\nabla p + \rho \mathbf{g} + \mu \nabla^2 \mathbf{v}$$

其中  $\frac{D\mathbf{v}}{Dt}$  为流体的随体导数, 其定义为

$$\frac{D\mathbf{v}}{Dt} = \frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v} \cdot \nabla) \mathbf{v}$$

**定理 3.2 不可压性质** 流体中的质量守恒方程为

$$\frac{\partial \rho}{\partial t} = -\nabla \cdot (\rho \mathbf{v})$$

一般的流体模拟中总是假设流体不可压缩/膨胀, 于是密度  $\rho$  在各处和任意时刻均为常数. 于是上式即为

$$\nabla \cdot \mathbf{v} = 0$$

即流体的速度场是无散的, 也即流入任意一点的流体体积总是等于流出该点的流体体积.

### 3.2 描述流体的两种方式

上述 N-S 方程是以**欧拉视角**描述流体的运动的. 欧拉视角是指将流体的所有物理量看成空间上的一个场, 然后描述这个场随时间的变化. 这好比在水中插有无限多的木桩, 每个木桩上装有检测水的流速, 密度, 压强等的传感器, 我们描述的是所有传感器上的数值变化.

另一种描述流体的方式是**拉格朗日视角**, 即将流体看成由无数个质点组成, 然后描述每个质点随时间的运动. 这好比在水中放入无数悬浮的小球, 其上也装有传感器, 我们描述的是每个小球的位置, 速度等随时间的变化.

### 3.3 光滑粒子流体

我们首先采用拉格朗日视角来描述流体的运动.

**定理 3.3 光滑粒子流体模型** 光滑粒子流体 (Smoothed Particle Hydrodynamics, SPH) 模型将流体看成有限个流体微团 (视作粒子) 组成的系统. 每个粒子  $i$  具有位置  $\mathbf{x}_i$ , 速度  $\mathbf{v}_i$ , 质量  $m_i$ .

为了根据上述信息计算流体的宏观性质, 例如求解 N-S 方程必不可少的密度场和压强场, 我们采用一种经典的办法: **核函数**.

### 3.3.1 核函数

我们已经经历了很多采样问题以从离散的样本点重建连续函数. 对于流体而言, 如果采取简单的近邻法划分空间并计算目标函数, 往往会在边界上造成函数值的跳变, 因而结果交叉. 因此, 我们需要采取一定的办法使得这种不连续性被平滑. 一种自然的想法就是考虑目标点附近的所有粒子对该点性质的影响, 并且距离越近的粒子影响越大. 这就可以通过引入核函数  $W(\mathbf{r}, h)$  进行加权平均而解决.

一般而言, 核函数  $W$  需要满足归一化性质, 即在  $d$  维空间下有

$$\int_{\mathbb{R}^d} W(\|\mathbf{x}\|) d\mathbf{x} = 1$$

一种常用的核函数是三次样条函数:

$$W(\mathbf{r}, h) = \sigma \begin{cases} 6(q^3 - q^2) + 1, & 0 \leq q \leq 1/2 \\ 2(1 - q)^3, & 1/2 < q \leq 1 \\ 0, & q > 1 \end{cases}$$

其中  $q = \frac{\|\mathbf{r}\|}{h}$ ,  $h$  为核函数的半径;  $\sigma$  为归一化系数, 在三维空间中有  $\sigma = \frac{8}{\pi h^3}$ . 在实践中, 一般取  $h$  为初始状态下粒子平均间距的  $1 \sim 3$  倍左右.

### 3.3.2 流体性质的计算

有了核函数之后, 我们就可以根据粒子的位置和质量计算流体的密度和压强等性质. 流体的密度场可计算如下:

$$\rho(\mathbf{x}) = \sum_i m_i W(\|\mathbf{x} - \mathbf{x}_i\|)$$

类似地, 流体的压强场及其梯度可计算如下:

$$p(\mathbf{x}) = \sum_i p_i \frac{m_i}{\rho_i} W(\|\mathbf{x} - \mathbf{x}_i\|)$$

$$\nabla p(\mathbf{x}) = \sum_i p_i \frac{m_i}{\rho_i} \nabla W(\|\mathbf{x} - \mathbf{x}_i\|)$$

其中  $p_i = k(\rho_i - \rho_0)^\gamma$ ,  $k, \gamma$  为流体的常数,  $\rho_0$  为流体的静密度.  $p_i$  表示粒子  $i$  处的压强, 其随  $\rho_i$  的增加而增加. 于是在 N-S 方程中  $-\nabla p$  项就会将粒子从密度高的区域推向密度低的区域以维持近似不可压的状态.

## 3.4 欧拉网格流体

使用拉格朗日法进行流体模拟十分直观, 便捷且高效, 但其对物理模型做了过于简化的处理, 因此模拟结果往往不够准确也不够真实. 欧拉法则将流体视为一种连续介质, 并且由于 N-S 方程本身的形式就是欧拉视角, 欧拉法也能够更自然地对其进行处理.

### 3.4.1 空间离散化

现在, 我们不再考虑组成流体的粒子的具体运动, 而是将整个容器划分为若干小单元格 (例如边长为  $\Delta x$  的正方形/立方体网格). 场景中的各种物质与物理量均存储在网格中.

**标记网格** 为了表示流体在空间中的分布, 我们可以使用**标记网格 (Marker-and-Cell grid, MAC grid)** 方法. 该方法在每个网格单元的中心存储一个标记变量  $m$ , 用于表示该单元中是否存在流体. 如果存在流体, 则  $m = 1$ ; 否则  $m = 0$ . 通过这种方式, 我们就可以表示流体的形状和位置.

### 3.4.2 数值求解

求解 N-S 方程常用的办法是分裂法. 对于无粘性的流体, 将 N-S 方程拆分成三个更简单的偏微分方程, 在一个时间步内对它们依次求解. 对 N-S 方程变形可得

$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u} \cdot \nabla) \mathbf{u} + \mathbf{g} + \frac{\mu}{\rho} \nabla^2 \mathbf{u} - \frac{1}{\rho} \nabla p$$

于是一个时间步内的步骤即为:

#### 1. 对流 (Advection)

求解  $\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u} \cdot \nabla) \mathbf{u}$  以更新  $\mathbf{u}$ .

#### 2. 外力 (External Force)

求解  $\frac{\partial \mathbf{u}}{\partial t} = \mathbf{g}$  以更新  $\mathbf{u}$ .

#### 3. 粘度和扩散 (Viscosity or Diffusion)

求解  $\frac{\partial \mathbf{u}}{\partial t} = \frac{\mu}{\rho} \nabla^2 \mathbf{u}$  以更新  $\mathbf{u}$ .

#### 4. 压强投影 (Pressure Projection)

求解  $\frac{\partial \mathbf{u}}{\partial t} = -\frac{1}{\rho} \nabla p$  以更新  $\mathbf{u}$ , 并且保证更新后的速度场满足不可压性质  $\nabla \cdot \mathbf{u} = 0$ . 这也是这一步放在最后的原因.

现在, 我们就来考虑每一步的微分方程的求解方法.

**对流** 对流方程可以表示为以下的一般形式:

$$\frac{D\phi}{Dt} = 0, \quad \text{i.e.} \quad \frac{\partial \phi}{\partial t} = -(\mathbf{u} \cdot \nabla) \phi$$

其中  $\phi$  是流体的标量场, 例如速度  $\mathbf{u}$  的各个分量. 求解对流过程的算法可以表示如下:

$$\phi^{(n+1)} = \text{advect}(\mathbf{u}, \Delta t, \phi^{(n)})$$

其中  $\phi^{(n)}$  表示当前的标量场,  $\phi^{(n+1)}$  表示对流之后的标量场.

如果我们直接用差分的方式将上面的微分方程离散化, 会导致严重的问题: 已经证明这样的方法是无条件不稳定<sup>1</sup>的. 因此, 研究者们提出了一个更加直观稳定的办法: 使用拉格朗日视角解决对流问题, 即**半拉格朗日**

<sup>1</sup>即无论时间步长取多小, 数值解都会震荡发散



### 方法 (Semi-Lagrangian Method).

假定我们需要求解标量场  $\phi$  在网格点  $\mathbf{x}_g$  在第  $n+1$  个时间步的值  $\phi^{(n+1)}(\mathbf{x}_g)$ . 在拉格朗日的视角下, 我们可以寻找在前一步时是何处的流体粒子在速度场  $\mathbf{u}$  的作用下运动到了  $\mathbf{x}_g$ . 设该粒子在第  $n$  个时间步的位置为  $\mathbf{x}_p$ . 在拉格朗日视角下, 物理量  $\phi$  由流体粒子携带, 因此有

$$\phi^{(n+1)}(\mathbf{x}_g) = \phi^{(n)}(\mathbf{x}_p)$$

于是目标转化为求解粒子位置  $\mathbf{x}_p$ . 根据粒子的运动方程有

$$\frac{d\mathbf{x}}{dt} = \mathbf{u}$$

在一个时间步长  $\Delta t$  内, 可以用显式欧拉积分求出  $\mathbf{x}_p$ , 即

$$\mathbf{x}_p = \mathbf{x}_g - \Delta t \mathbf{u}(\mathbf{x}_g)$$

为了提高精度, 我们也可以使用二阶 Runge-Kutta 方法, 即在计算  $\mathbf{x}_p$  的过程中将时间步长继续细分:

$$\mathbf{x}_{\text{mid}} = \mathbf{x}_g - \frac{1}{2} \Delta t \mathbf{u}(\mathbf{x}_g), \quad \mathbf{x}_p = \mathbf{x}_{\text{mid}} - \frac{1}{2} \Delta t \mathbf{u}(\mathbf{x}_{\text{mid}})$$

继续增加采样点, 将时间步长划分为更小的间隔也是可行的.

现在, 我们已经获取了  $\mathbf{x}_p$ . 一般而言, 求得的  $\mathbf{x}_p$  并不在网格点上, 因此需要使用插值的方法计算  $\phi^{(n)}(\mathbf{x}_p)$ . 在二维情形下可以使用双线性插值的办法; 而在三维情形下可以使用三线性插值的办法.

如果求得的  $\mathbf{x}_p$  在液体内部, 那么直接使用插值的结果即可; 但有时  $\mathbf{x}_p$  可能在液体的外部. 造成这一意外的可能原因主要有两种, 其解决办法分别如下:

1.  $\mathbf{x}_p$  确实在液体的外部, 这意味着有新的流体流入. 此时, 我们应当知道外部流体的性质, 因此直接用外部流体的性质赋值即可.
2.  $\mathbf{x}_p$  本应在液体的内部, 但由于数值计算的误差等原因导致其落在了外部. 此时, 我们可以寻找流体边界上距离  $\mathbf{x}_p$  最近的点  $\mathbf{x}_b$ , 然后使用插值的方法计算  $\phi^{(n)}(\mathbf{x}_b)$  作为结果.

**外力** 求解外力方程实际上非常简单, 因为外力项  $\mathbf{g}$  通常是一个常量 (例如重力). 因此, 可以直接用差分的形式写出新的速度场

$$\mathbf{u}^{new} = \mathbf{u} + \mathbf{g} \Delta t$$

**粘度** 粘度方程中包含一个拉普拉斯算子  $\nabla^2$ . 我们可以使用有限差分的方法对其进行离散化 (这和图像的泊松编辑是类似的). 在二维情形下, 对于网格点  $(i, j)$  有

$$\mathbf{u}_{ij}^{new} = \mathbf{u}_{ij} + \frac{\mu}{\rho} \Delta t \frac{\mathbf{u}_{(i-1)j} + \mathbf{u}_{(i+1)j} + \mathbf{u}_{i(j-1)} + \mathbf{u}_{i(j+1)} - 4\mathbf{u}_{ij}}{(\Delta x)^2}$$

如果需要更高的精度, 可以将  $\Delta t$  继续细分为更小的时间间隔, 然后按照上述公式多次迭代.

**压强投影** 最后, 我们需要求解压强投影方程. 将其写作对时间离散的形式就有

$$\mathbf{u}^{new} - \mathbf{u}^* = -\frac{\Delta t}{\rho} \nabla p$$

这里  $\mathbf{u}^*$  是经过前三个步骤所得的速度场, 它可能已经失去了无旋的性质. 而求得的速度场  $\mathbf{u}^{new}$  需要满足不可压性质, 于是  $\nabla \mathbf{u}^{new} = \mathbf{0}$ . 于是对上式两边求梯度可得

$$\nabla \mathbf{u}^* = \frac{\Delta t}{\rho} \nabla^2 p$$

然后将上式空间离散化即可得

$$p_{(i-1)j} + p_{(i+1)j} + p_{i(j-1)} + p_{i(j+1)} - 4p_{ij} = \frac{\rho \Delta x}{\Delta t} (u_{(i+1)j} - u_{ij} + v_{i(j+1)} - v_{ij})$$

式中右边的各  $p_{ij}$  是未知量, 而左边的各  $u_{ij}, v_{ij}$  均为已知量. 于是上述方程即可写作线性方程组  $\mathbf{A}p = \mathbf{b}$  的形式. 我们现在来考虑体系的边界条件.

### 1. Neumann 边界条件

对于流体与固体的边界, 我们要求流体不能穿透固体. 设边界处法向量为  $\mathbf{n}$ , 则有

$$\mathbf{u} \cdot \mathbf{n} = 0$$

代入压强投影方程可得

$$\left( \mathbf{u}^* - \frac{\Delta t}{\rho} \nabla p \right) \cdot \mathbf{n} = 0$$

即

$$\nabla p = \frac{\rho}{\Delta t} \mathbf{u}^* \cdot \mathbf{n}$$

这就是 Neumann 边界条件.