# 使用 taichi 仿真

陈柯

2022 年 7 月 15 日

## 1 目的

用 Taichi 框架制作游戏“TaiCro”。

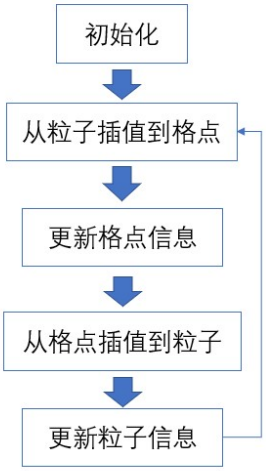
## 2 Taichi 框架介绍

Taichi 框架是 MIT 博士生 Yuanming Hu（胡渊鸣）开发的计算机图形学代码库，在此基础上可以实现很多物理模拟算法。

我们只学习使用 88 行实现冰雪奇缘中的 2D 版本的代码。

## 3 MLS-MPM 方法

算法流程图如下：



### 3.1 从粒子插值到格点

插值函数为:

$$N(x)=\begin{cases}\frac{1}{2}|x|^3-x^2+\frac{2}{3} & 0\le|x|\le 1\\ -\frac{1}{6}|x|^3+x^2-2|x|+\frac{4}{3} & 1\le|x|\le 2\\ 0 & 2\le|x|\end{cases}$$

B 样条函数性质:

$$\sum_i w_{ip}=1,\sum_i w_{ip}\boldsymbol{x}_i=\boldsymbol{x}_p$$

格点质量计算:

$$\text{质量插值：}\ m_i^n=\sum_p w_{ip}^n m_p^0$$

$$\text{质量守恒：}\ \sum_i m_i^n=\sum_i\sum_p w_{ip}^n m_p^0=\sum_p\left(\sum_i w_{ip}^n\right)m_p^0=\sum_p m_p^0$$

格点速度计算:

$$\text{动量插值：}\ m_i^n\boldsymbol{v}_i^n=\sum_p w_{ip}^n m_p^0(\boldsymbol{v}_p^n+\boldsymbol{C}_p^{n+1}(x_i^n-x_i^p))$$

$$\boldsymbol{C}_p=\boldsymbol{B}_p\boldsymbol{D}_p^{-1}$$

$$\boldsymbol{B}_p=\sum_i w_{ip}\boldsymbol{v}_i(x_i-x_p)^T$$

$$\boldsymbol{D}_p=\sum_i w_{ip}\boldsymbol{v}_i(x_i-x_p)(x_i-x_p)^T$$

$$\text{计算格点速度：}\ \boldsymbol{v}_i^n=\frac{m_i^n\boldsymbol{v}_i^n}{m_i^n}$$

我们可以验证这里的动量守恒。

### 3.2 更新格点信息

格点受力计算:

$$\boldsymbol{f}_i^n=-\frac{\partial\Psi^n}{\partial\boldsymbol{x}_i^n}=-\sum_p N_i(x_p^n)V_p^0M_p^{-1}\frac{\partial\psi_p^n(F_p^n)}{\partial\boldsymbol{x}_i^n}(x_i^n-x_i^p)$$

格点速度更新:

$$v_i^{n+1}=v_i^n+\frac{\boldsymbol{f}_i^n+\boldsymbol{f}_{iext}^n}{m_i^n}\Delta t$$

碰撞处理：仅对在边界附近的格点进行处理。

### 3.3 从格点插值到粒子

粒子速度:

$$\text{速度插值：}\ \boldsymbol{v}_p^{n+1}\sum_i w_{ip}^n\boldsymbol{v}_i^{n+1}$$

此时粒子质量不必插值。

### 3.4 更新粒子信息

粒子位置更新:

$$\boldsymbol{x}_p^{n+1} = \boldsymbol{x}_p^n + \boldsymbol{v}_p^{n+1}\Delta t$$

形变梯度更新:

$$F_p^{n+1} = (I + \Delta t \boldsymbol{C}_p^{n+1})F_p^n$$

粒子形变梯度添加约束。

## 4 游戏设计思路

我想制作的就是类似大家比较熟知的小游戏：“小鳄鱼爱洗澡”。所以我将游戏名取为：TaiCro。为了实现交互，我将 imgui 框架和 taichi 框架融合在了一起。用户可以通过鼠标点击来使得沙子消失，从而引导水进入水管，从而游戏通关。

## 5 制作过程中遇到的困难

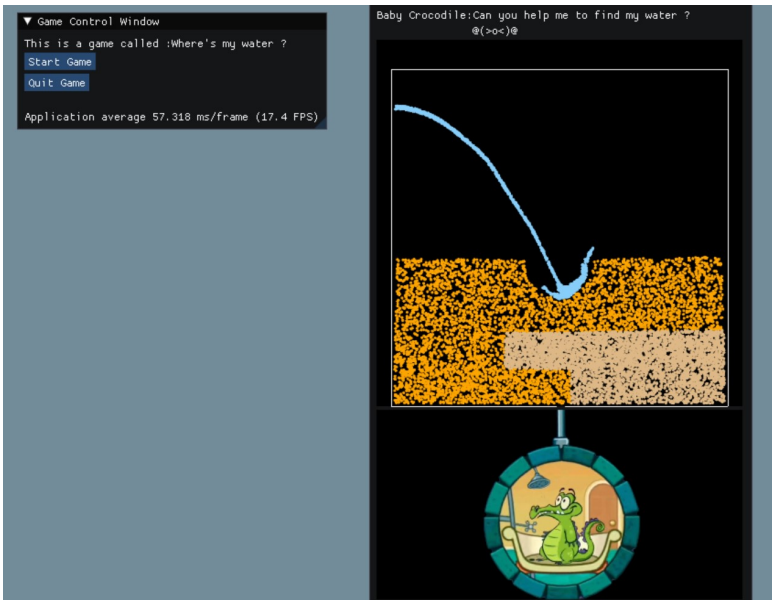
1. imgui 需要结合一个后端来实现，我一开始选了 opengl3 作为后端，发现 taichi.h 需要用多字符集编译，而 glfw.lib 需要用 Unicode 字符集编译，两者冲突。后来选了 DirectX10，结果发现官方说明里只有 DirectX10 没给实现 imgui 加载图片的示例，而 DirectX9、DirectX11、DirectX12 都有示例，随后我转为用 DirectX11，问题解决。

2. 要注意要用 x64 编译，taichi.h 不支持 x86 编译。而且用 debug 会很慢，用 release 比较好。

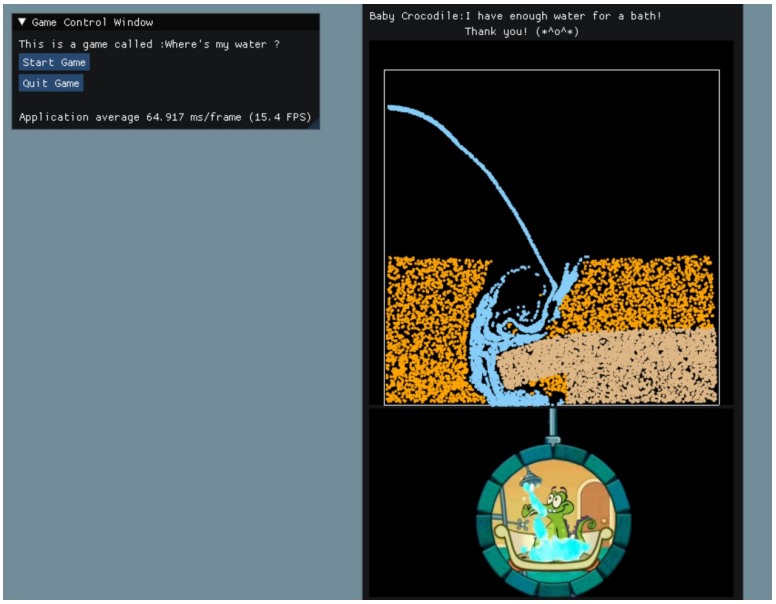
3. 因为我是把 taichi.h 和 main.cpp 文件拿出来重新构建了一个 vs 项目，所以在编译时还要注意在预处理器里加上 _CRT_SECURE_NO_WARNINGS，这样就不会出现 fopenf 不安全等报错。

# 6 实现结果

以下是游戏刚开始的界面：




以下是游戏通过时的界面：




游戏的实现结果请看随本报告附带的视频。

# 7 不足

1. 没有实现更多功能，只实现了最基本的交互和相关功能。

2. 由于渲染的窗口不是用 taichi 自带的，而是用 imgui，所以实际上游戏帧率很低，仿真的效果较差。

# 8 参考文献

[1]Stomakhin et al. "A Material Point Method for Snow Simulation." ACM Transactions on Graphics (SIGGRAPH 2013)

[2]Hu et al. "A Moving Least Squares Material Point Method with Displacement Discontinuity and Two-Way Rigid Body Coupling." ACM Transactions on Graphics (SIGGRAPH 2018)

[3]Hu et al .Taichi: A Language for High-Performance Computation on Spatially Sparse Data Structures. ACM Transactions on Graphics (SIGGRAPH Asia 2019)