GAMES 201

Advanced Physics Engines 2020: A Hands-on Tutorial

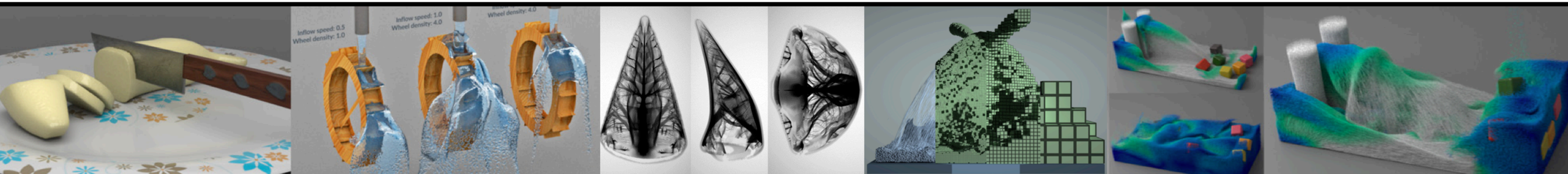
高级物理引擎实战2020 （基于太极编程语言）

第十讲：总结

Yuanming Hu 胡渊鸣 MIT CSAIL

麻省理工学院 计算机科学与人工智能实验室

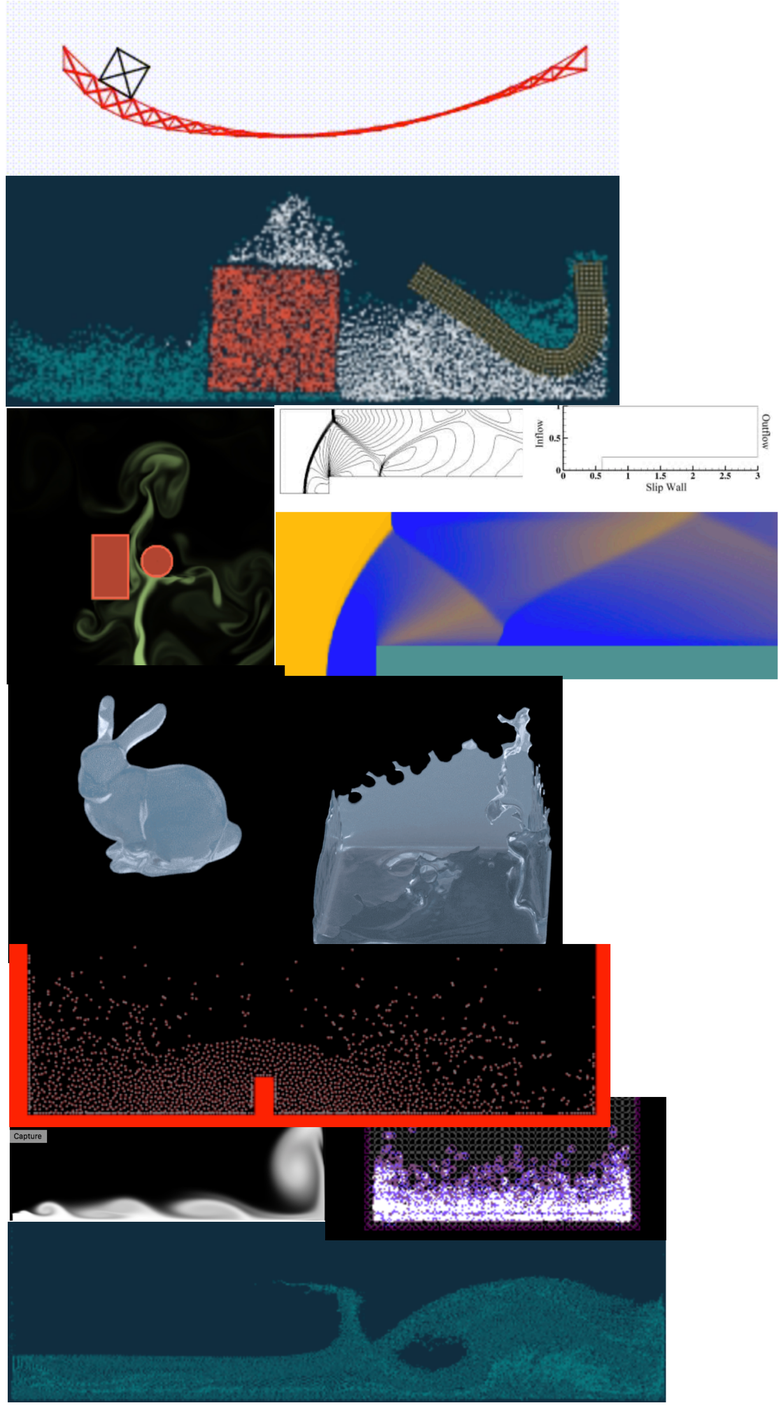
Taichi Programming Language



Homework 2 获奖作品

- Wimaxs & g1n0st: Taichi小游戏合集
- BillXu2000：整合mpm_lagrangian_forces和mpm_99
- JYLeeLYJ: OOP Taichi and Eulerian Fluid Engine Design
- hejob: 2D FVM Compressible CFD Solver for multiblock structured mesh
- Timber: MPM 3D Liquid
- huahuo: PBF-driven game (Water gun)
- lqxu: 3D PBF
- JerryYan：Eulerian Fluid -- Jacobi method and CG method
- linyaodong：3D SPH with Grid-based Neighborhood Search
- robs: 2D Free Surface Fluid Simulation (Eluerian/PIC/FLIP/APIC)
- limengfan: MPM 的三种边界条件

- 课程结束后欢迎继续提交!（但是可能得等比较久才能收到纪念品...）



ChinaVR Taichi竞赛

- 一分钟完成报名：https://www.wjx.top/jq/88136866.aspx
- 特等奖1名、一等奖3名、二等奖6名、三等奖和优秀奖若干
- 竞赛网站：https://www.chinavr.info/creative.html

https://www.chinavr.info/

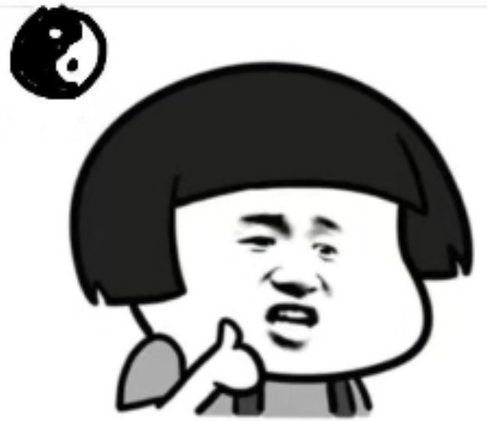
中国计算机学会 China Computer Federation

中国图象图形学学会 CHINA SOCIETY OF IMAGE AND GRAPHICS

中国仿真学会 China Simulation Federation

恭喜各位！

- 收获了新技能：Taichi编程语言 刚体 液体 烟雾 弹塑性体 PIC/FLIP法 Krylov-子空间求解器 预条件 无矩阵法 多重网格 弱形式与有限元 隐式积分器 拓扑优化 物质点法 现代处理器微架构 内存层级 并行编程 GPU编程 稀疏数据结构 可微编程…
- **课程难度不小，能坚持到最后的都是勇者！**



Overview

- **What I learned from building physics engines and Taichi**
  - Simplicity is good! （“简单”好！）
- **物理引擎的未来**
- **我自己在课程中的收获**
- **Acknowledgments**

Simplicity is good

- **考试：**证明我会某种方法　　**做研究、工程：**我能找到好方法
- **用哪种方法不重要，重要的是系统性地解决问题**
- **一天只有24个小时**
  - “最聪明的人都是一样聪明的”
  - 用简单可行的方法解决大部分简单的问题，才有时间去解决真正复杂的问题
- **有影响力的创新往往伴随着复杂性的减少**
  - “不要给他人带来麻烦”

Complexity is bad. Why?

- **No one else understands your ideas (limited impact)**
- **Slow to implement (won't fail quickly)**
- **Error-prone and hard to analyze**
- **Hard to implement efficiently (by hands and by compilers)**
  - Sparse data structures (as C++ libraries)
  - As a result: slow physical solvers
- **...**

Why do people still go complicated?

Why do people still go complicated?

Because it's way easier to be complicated than to be simple.

Why do people still go complicated?

- **To prove smartness**
  - Maybe a good idea at school…
  - … but almost never a good idea when you need to interact with people
  - In fact if you look at the bigger picture it's often better not to be too smart
- **To show off the amount of work**
  - so that you have more pages in your paper—not a great idea either
- **To hide potential errors**
  - "Nobody can prove I'm wrong because nobody understands what I'm doing, so I'm right."
  - "科学的表述在于其是可证伪的" - 卡尔·波普尔
- **To differentiate from prior work (and have the paper accepted)**
  - "Simple ideas have been done by previous people, so I have to do a more complex one."
- **Being lazy to simplify**
- **Audience being too nice to tolerate**

Use a simple solution != being an idiot

- ✦The smartest people know the importance of keeping things simple.

“**Everything should be made as simple as possible, but not simpler**.”

—— Albert Einstein

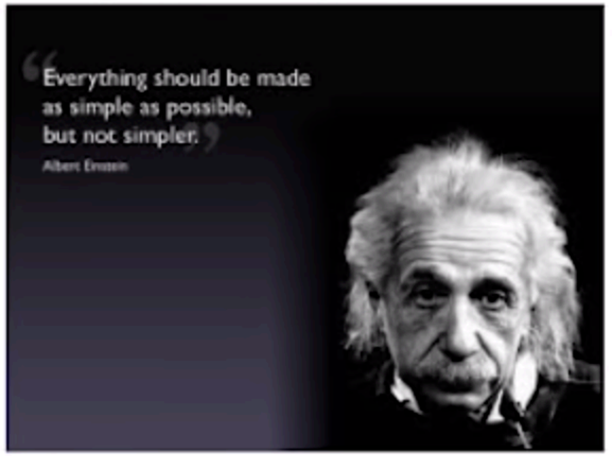
cppcon

Father of C++

Morgan Stanley

Make simple things simple!

- What does “simple” mean?



- Make the code directly express intent
  - Simple code is easier to understand
  - Simple code is often beautiful
  - Simple code is often fast
  - Not all code can be simple
    - Hide the complexity behind a simple interface

MAKE SIMPLE TASKS SIMPLE!

Bjarne Stroustrup

Being too clever is not clever

- BWK (1976)
  - "debugging is twice as hard as writing the program in the first place. So, if you're as clever as you can be when you write it, how can you debug it?"
- Design for clarity
  - Essential complexity
    - What is needed to defeat to solve the problem
    - Try to find the simplest solution
  - Accidental complexity
    - introduced by the expressions of the solution
    - Minimize it by using C++ well
  - Don't over-abstract
    - Base generalizations on concrete solutions
  - Hide complexity
    - Good interfaces are essential

MAKE SIMPLE TASKS SIMPLE!

Bjarne Stroustrup

"There are two ways of constructing a software design: One way is to make it so simple that there are obviously no deficiencies, and the other way is to make it so complicated that there are no obvious deficiencies. *The first method is far more difficult*." — C.A.R. Hoare

"有两种软件设计的方法: 第一种是使得软件足够简单以至于**显然没有错误**，第二种是使得软件足够复杂以至于**没有显然的错误。 第一种方法难得多**。"

——Tony Hoare (1980年图灵奖得主，快速排序发明人)

✦ How to Have a Bad Career by David Patterson (Turing Award 2017)

David Patterson

How to Have a Bad Career

Bad Career Move #2: Let Complexity Be Your Guide (Confuse Thine Enemies)

- Best compliment:
  "Its so complicated, I can't understand the ideas"
- Easier to claim credit for subsequent good ideas
  - If no one understands, how can they contradict your claim?
- It's easier to be complicated
  - Also: to publish it must be different; N+1st incremental change
- If it were not unsimple then how could distinguished colleagues in departments around the world be positively appreciative of both your extraordinary intellectual grasp of the nuances of issues as well as the depth of your contribution?

The story of ChainQueen [ICRA 2019]

- **ChainQueen aims to be a differentiable MPM solver**
  - But we didn't have a tailored AutoDiff system at that time
  - I had to manually derive the gradients of MPM
  - Fortunately we have MLS-MPM, which is a lot **simpler** than traditional MPM
- **Thanks to the simplicity of MLS-MPM, we had a ICRA paper finished in 25 days (otherwise it won't be doable)**
- **DiffTaichi further simplifies differentiable MPM - no manual gradient derivation anymore!**

ChainQueen's supplemental material...

(J, G2P) For $\mathbf{x}_p^n$, we have

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \mathbf{v}_p^{n+1} \tag{85}$$

$$\mathbf{v}_p^{n+1} = \sum_i N(\mathbf{x}_i - \mathbf{x}_p^n)\mathbf{v}_i^n \tag{86}$$

$$\mathbf{C}_p^{n+1} = \frac{4}{\Delta x^2}\sum_i N(\mathbf{x}_i - \mathbf{x}_p^n)\mathbf{v}_i^n(\mathbf{x}_i - \mathbf{x}_p^n)^T \tag{87}$$

$$\mathbf{p}_i^n = \sum_p N(\mathbf{x}_i - \mathbf{x}_p^n)\left[m_p\mathbf{v}_p^n + \left(-\frac{4}{\Delta x^2}\Delta t V_p^0 \mathbf{P}_p^n \mathbf{F}_p^{nT} + m_p \mathbf{C}_p^n\right)(\mathbf{x}_i - \mathbf{x}_p^n)\right] \tag{88}$$

$$m_i^n = \sum_p N(\mathbf{x}_i - \mathbf{x}_p^n) m_p \tag{89}$$

$$\mathbf{G}_p := \left(-\frac{4}{\Delta x^2} V_p^0 \Delta t \mathbf{P}_p^n \mathbf{F}_p^{nT} + m_p \mathbf{C}_p^n\right) \tag{90}$$

$$\Longrightarrow \tag{91}$$

$$\frac{\partial L}{\partial \mathbf{x}_{p\alpha}^n} = \left[\frac{\partial L}{\partial \mathbf{x}_p^{n+1}}\frac{\partial \mathbf{x}_p^{n+1}}{\partial \mathbf{x}_p^n} + \frac{\partial L}{\partial \mathbf{v}_p^{n+1}}\frac{\partial \mathbf{v}_p^{n+1}}{\partial \mathbf{x}_p^n} + \frac{\partial L}{\partial \mathbf{C}_p^{n+1}}\frac{\partial \mathbf{C}_p^{n+1}}{\partial \mathbf{x}_p^n} + \frac{\partial L}{\partial \mathbf{p}_i^n}\frac{\partial \mathbf{p}_i^n}{\partial \mathbf{x}_p^n} + \frac{\partial L}{\partial m_i^n}\frac{\partial m_i^n}{\partial \mathbf{x}_p^n}\right]_\alpha \tag{92}$$

$$= \frac{\partial L}{\partial \mathbf{x}_{p\alpha}^{n+1}} \tag{93}$$

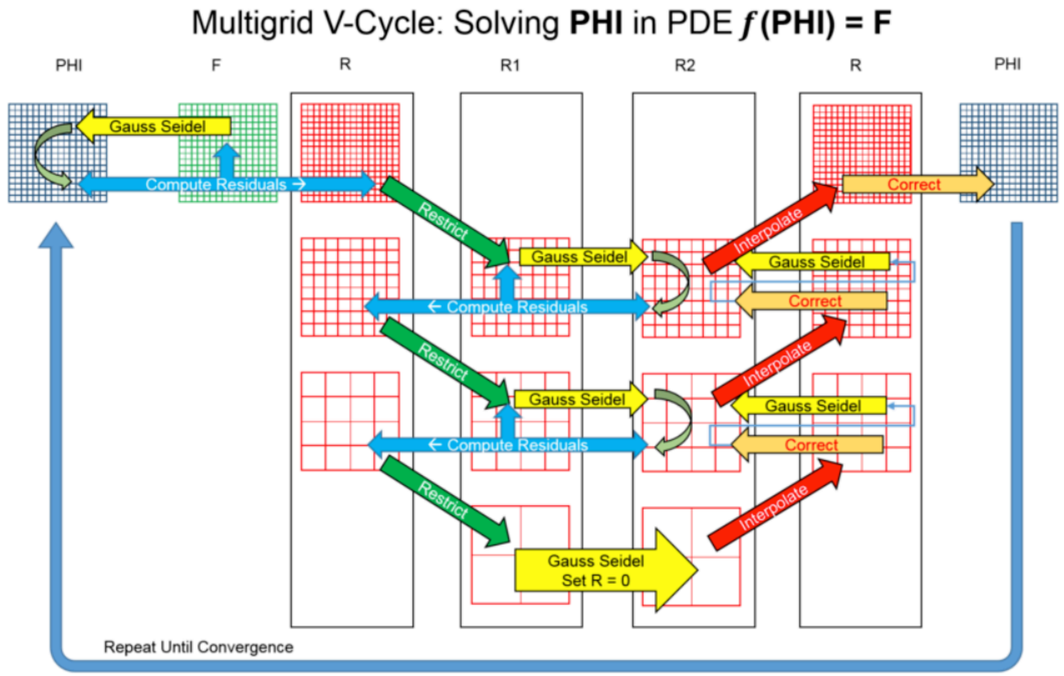
$$+ \sum_i \sum_\beta \frac{\partial L}{\partial \mathbf{v}_{p\beta}^{n+1}}\frac{\partial N(\mathbf{x}_i - \mathbf{x}_p^n)}{\partial \mathbf{x}_{i\alpha}}\mathbf{v}_{i\beta}^n \tag{94}$$

$$+ \sum_i \sum_\beta \frac{4}{\Delta x^2}\left\{-\frac{\partial L}{\partial \mathbf{C}_{p\beta\alpha}^{n+1}}N(\mathbf{x}_i - \mathbf{x}_p^n)\mathbf{v}_{i\beta} + \sum_\gamma \frac{\partial L}{\partial \mathbf{C}_{p\beta\gamma}^{n+1}}\frac{\partial N(\mathbf{x}_i - \mathbf{x}_p^n)}{\partial \mathbf{x}_{i\alpha}}\mathbf{v}_{i\beta}(\mathbf{x}_{i\gamma} - \mathbf{x}_{p\gamma})\right\} \tag{95}$$

$$+ \sum_i \sum_\beta \frac{\partial L}{\partial \mathbf{p}_{i\beta}^n}\left[\frac{\partial N(\mathbf{x}_i - \mathbf{x}_p^n)}{\partial \mathbf{x}_{i\alpha}}\left(m_p\mathbf{v}_{p\beta}^n + [\mathbf{G}_p(\mathbf{x}_i - \mathbf{x}_p^n)]_\beta\right) - N(\mathbf{x}_i - \mathbf{x}_p^n)\mathbf{G}_{p\beta\alpha}\right] \tag{96}$$

$$+ m_p \sum_i \frac{\partial L}{\partial m_i^n}\frac{\partial N(\mathbf{x}_i - \mathbf{x}_p^n)}{\partial \mathbf{x}_{i\alpha}} \tag{97}$$

Another example: Taichi and MGPCG



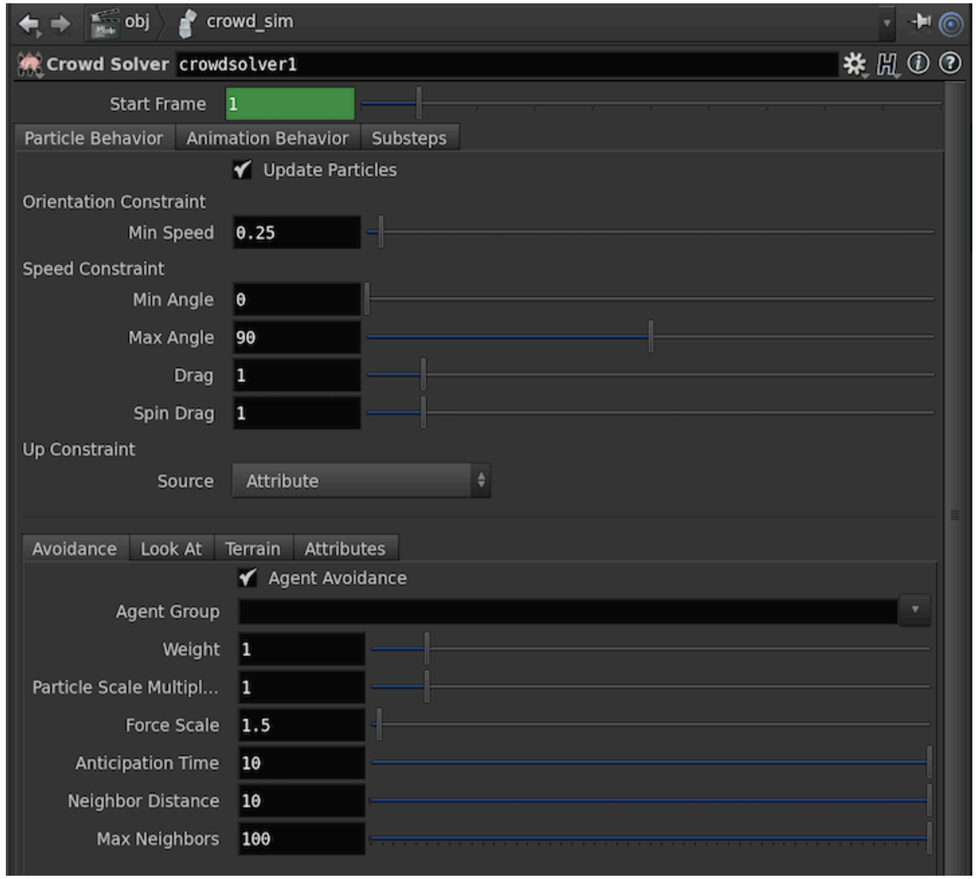
- MGPCG (multigrid preconditioned conjugate gradients) is hard to implement.
- With Taichi, a programming system that aims to be simple, I can implement it in 80 minutes and 300 LoC(according to git log).
- After GAMES 201, now everyone knows how to write a multigrid solver!
- 很少有人能正确实现 -> 我能轻易正确实现 -> 大家都能轻易正确实现 (大家好才是真的好)

Recall: our ambitious goal is to …

Simulate the world in your computer!

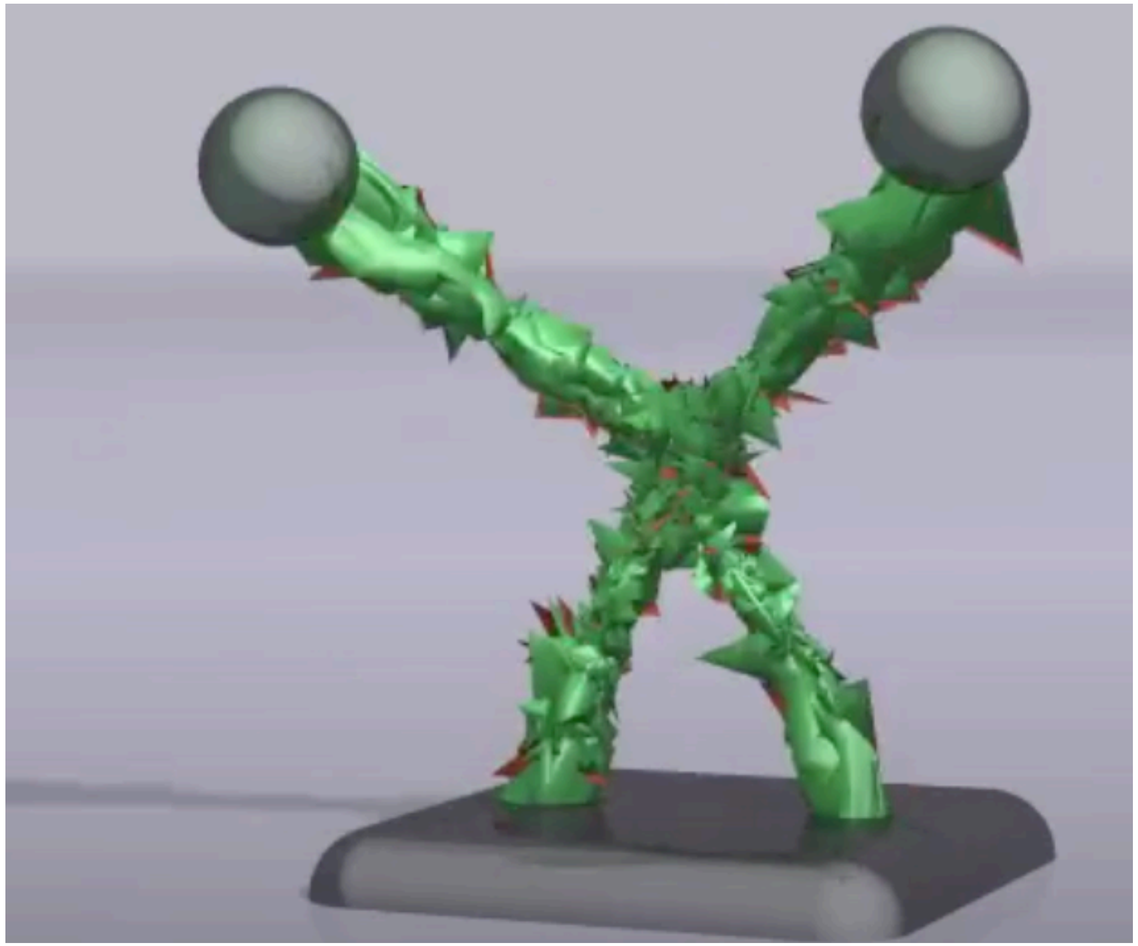
物理引擎的未来

- **易用性**
  - 如何快速、正确设置参数?
- **健壮性 (robustness)**
  - 手抖一下模拟炸了
- **完备性**
  - 多材料耦合
- **真实感**



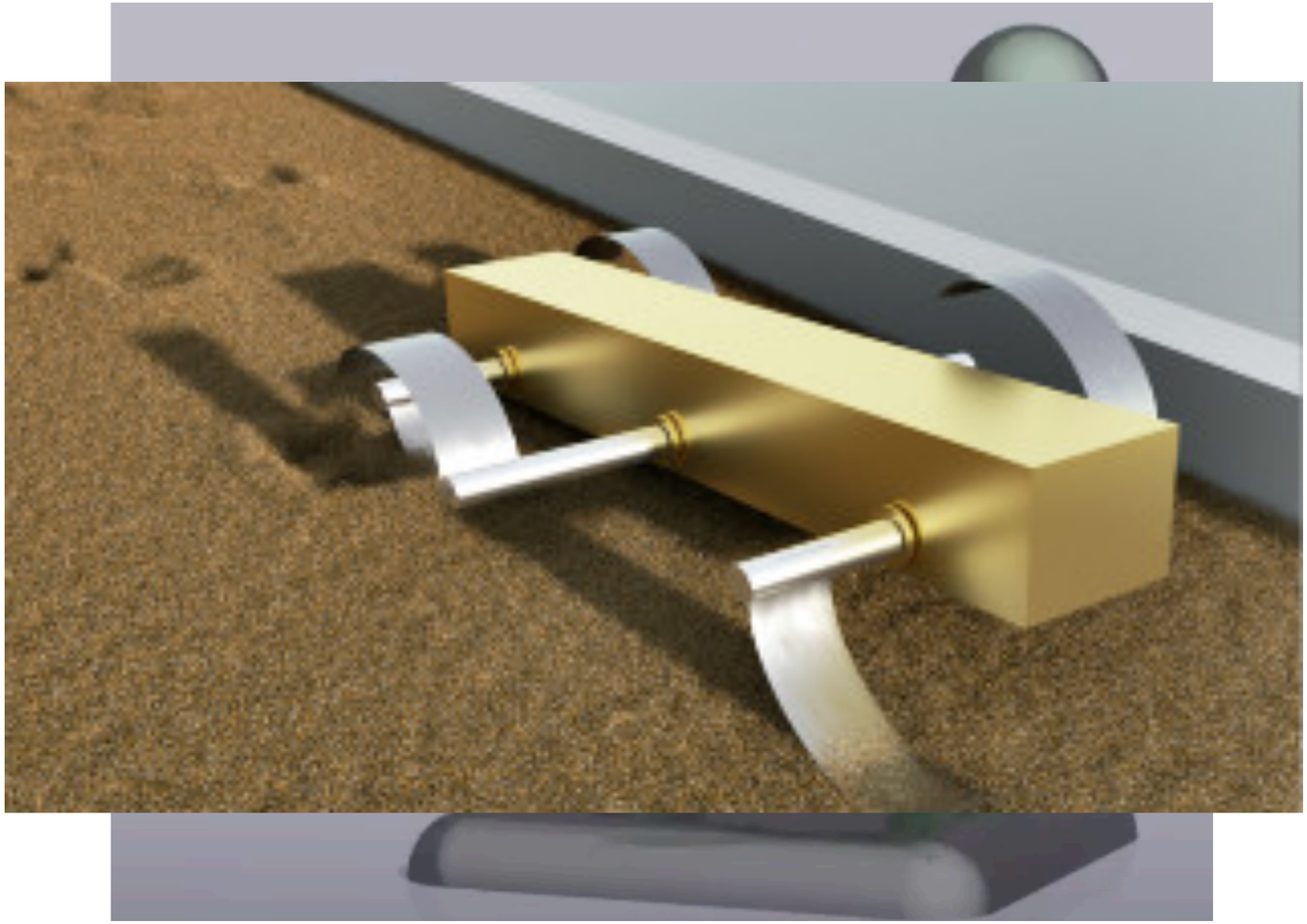
物理引擎的未来

- **易用性**
  - 如何快速、正确设置参数?
- **健壮性 (robustness)**
  - 手抖一下模拟炸了
- **完备性**
  - 多材料耦合
- **真实感**



物理引擎的未来

- **易用性**
  - 如何快速、正确设置参数?
- **健壮性 (robustness)**
  - 手抖一下模拟炸了
- **完备性**
  - 多材料耦合
- **真实感**



物理引擎的未来

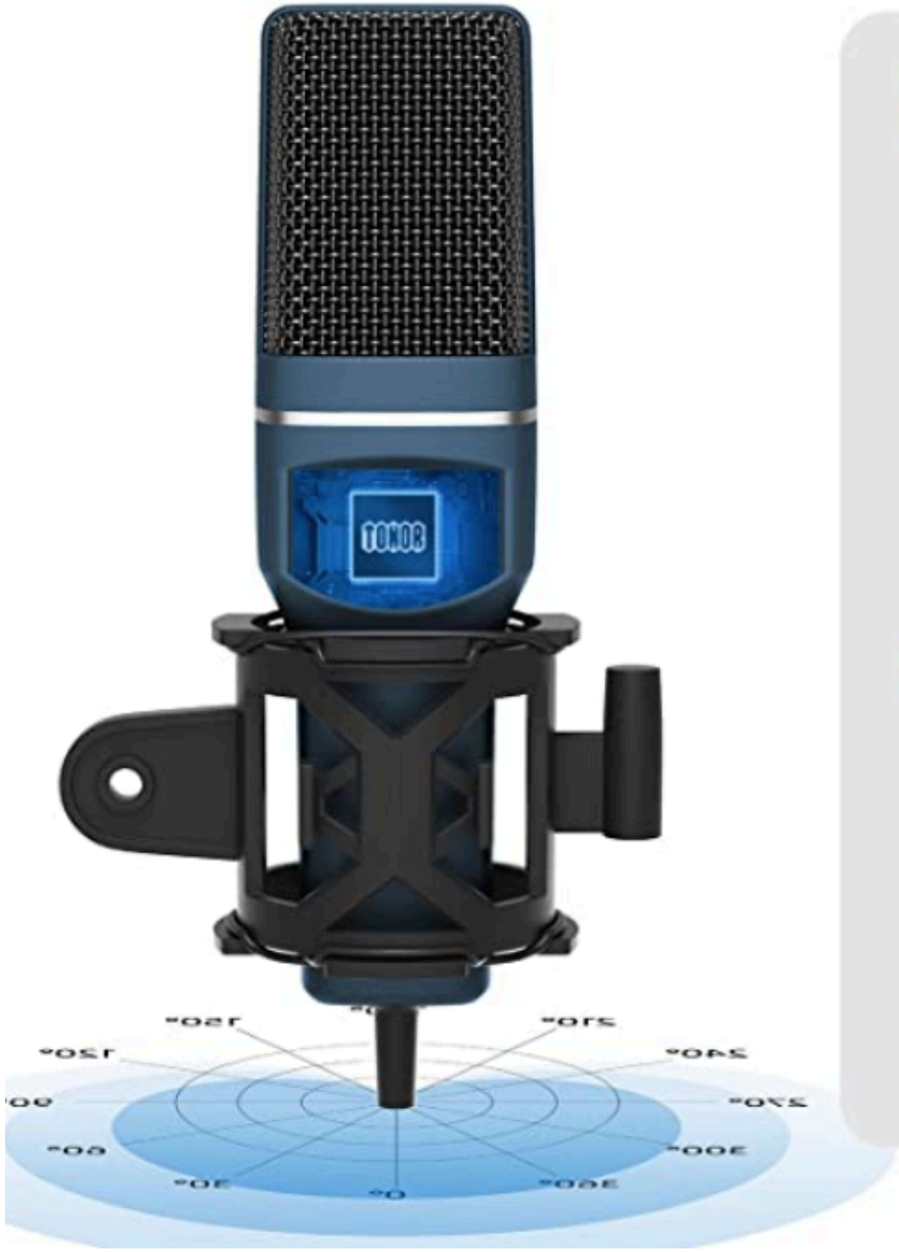
- **易用性**
  - 如何快速、正确设置参数?
- **健壮性 (robustness)**
  - 手抖一下模拟炸了
- **完备性**
  - 多材料耦合
- **真实感**

物理引擎的未来

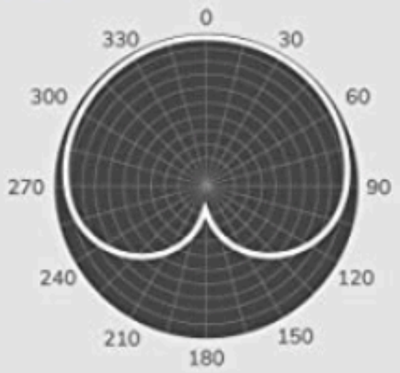
- **性能和引擎制造的生产力是关键**
  - 高性能计算、领域特定编程语言和编译器 (Taichi) 必不可少
  - 自动adaptive grids/多GPU/多node simulation
  - Simplicity, simplicity, simplicity
- **Learning-for-Simulation**
- **Simulation-for-Learning**

我自己在课程中的收获

- **同学们的作业（每天醒来第一件事...）**
- **世界上第一次用中文开设的物理引擎课?**
  - 近千名同学参加，近五十份引擎项目提交
  - Taichi的贡献：复制，粘贴，运行!
  - 同学们通过GAMES 201和Taichi快速掌握了曾今的我折腾了好久才弄懂的知识
  - 第一次普及了MGPCG, MLS-MPM, APIC等实用算法
- **讲课还是挺不容易的**
  - 周末备课像打仗（“一周只工作五天原来是有道理的！”）
  - 自己懂了 != 能流利的讲出来
  - LaTeX真难用 （但是公式排版效果还是好）
- **直播技术升级了。学会了正确使用麦克风。**



1 Cardioid pickup



2 Built-in quality chipset

鸣谢

- **GAMES社区：** 刘利刚教授(USTC)， 闫令琪教授(UCSB)，周晓巍教授(ZJU); 陈凌昊、张远青、董峻廷同学
- **Professors who (directly or indirectly) taught me so much on physical simulation:**
  - 蒋陈凡夫教授 (UPenn)
  - 朱博教授 (Dartmouth)
  - Prof. Eftychios Sifakis (UWisconsin-Madison)
  - Prof. Joseph M. Teran (UCLA)
  - Prof. Ron Fedkiw (Stanford)
- **Ph.D. advisors at MIT:**
  - Prof. Fredo Durand
  - Prof. Bill Freeman

鸣谢

- **每一位来上课的同学**
  - 特别是交了作业的同学 :-)
- **Taichi论坛管理员：woclass**
- **Taichi核心开发组：archibate, k-ye, xumingkuan, Rullec, FantasyVR, TH3CHARLie, rexwangcc, KLozes, Eydcao, …**
- **Taichi中文社区：StephenArk30, archibate, ArkhamWJZ, isdanni, rexwangcc, liaopeiyuan, zhiyaluo**
- **后勤部：Linan Zhou**

感谢助教同学们的鼎力相助！

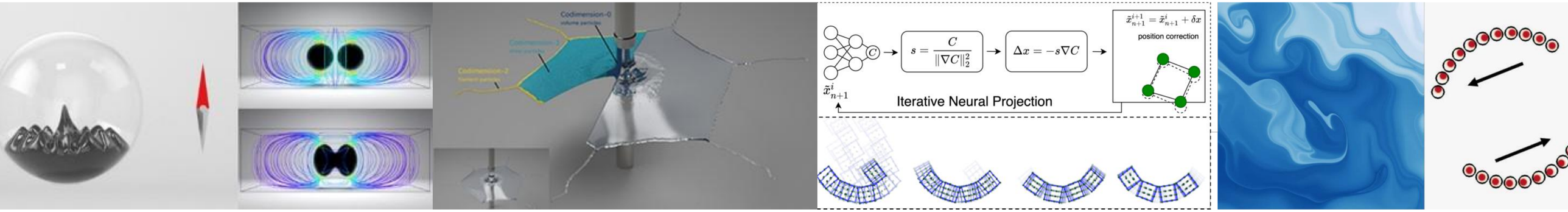
- 曹亚帝
- 禹鹏
- 杨玄达
- 冯旭东
- 翟骁
- 夏一鸣
- 史雨宸
- 袁宇杰

- **PLY dumper,**
- **compiler bugs,**
- **ti.field,**
- **Export video,**
- **v0.6.7->v0.6.27,**
- **…**

Bo Zhu @ Dartmouth

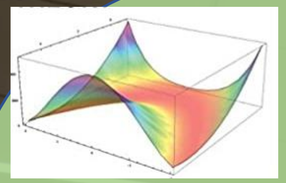
DARTMOUTH

- Research Interests: Physics-based Simulation, Computational Fluid Dynamics, Scientific Machine Learning
- Grabbing my research and teaching materials at: http://www.dartmouth.edu/~boolzhu/
- If you are interested in pursuing a PhD/MS/Intern/Visitor at Dartmouth, please reach out via: bo.zhu@dartmouth.edu

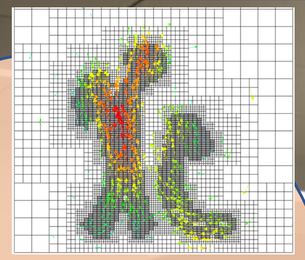


热烈祝贺GAMES 201结课！！！附宾夕法尼亚大学JIANG GROUP招生宣传

Physics-Based Modeling and Simulation for :



Numerical Optimization

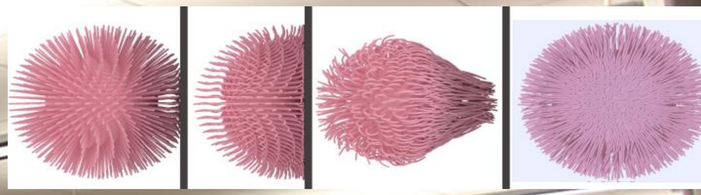


Scientific Computing

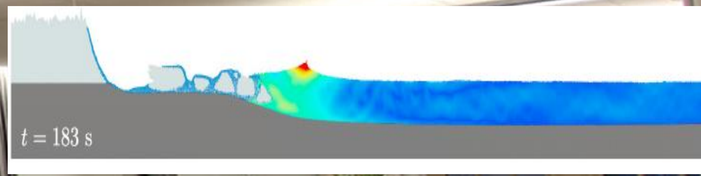
Machine Learning

High Performance Computing

1. Animation and Visual Applications

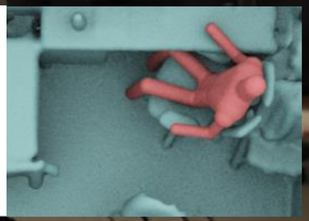


2. Computational Physics



3. Robotics

4. Visual Scene Understanding



5. And more for you to 开拓…!

欢迎PhD申请、访问学者/联合培养研究生、大三暑假的本科生、远程合作者、以及相信基于物理的模拟与数字孪生将促进第四次工业革命的你

https://www.seas.upenn.edu/~cffjiang/

More resources for graduate study

- **Toshiya Hachisuka's Graduate Study Survival Guide**
- **Fredo Durand's advices**
  - http://people.csail.mit.edu/fredo/student.html
  - How to Write a Bad Paper
- **Bill Freeman's advices**
  - How to do research

课程预告：GAMES 102 - 几何建模与处理

- 主讲：刘利刚，中国科学技术大学
  - http://staff.ustc.edu.cn/~lgliu
- 预期时间：2020年10-12月
- 课程定位：几何建模与处理的基础入门
- 内容概要：
  - 离散微分几何基础理论
  - 几何建模：重建、网格化、编辑、变形等
  - 几何处理：去噪、光顺、简化、压缩、参数化、修复等
  - 形状理解和分析基础
- 前序课程：
  - 计算机图形学、C++编程

谢谢大家！

谢谢大家！

「孔子曰：三人行，则必有我师。是故弟子不必不如师，师不必贤于弟子，闻道有先后，术业有专攻，如是而已。」

——韩愈《师说》

谢谢大家！

「孔子曰：三人行，则必有我师。是故弟子不必不如师，师不必贤于弟子，闻道有先后，术业有专攻，如是而已。」

——韩愈《师说》

「师父领进门，修行在个人。」

谢谢大家！

「孔子曰：三人行，则必有我师。是故弟子不必不如师，师不必贤于弟子，闻道有先后，术业有专攻，如是而已。」

——韩愈《师说》

「师父领进门，修行在个人。」

期待十年以后，你的物理引擎课程！