Dartmouth COSC 89/189 Physical Computing, Fall 2019

Instructor: Bo Zhu (bo.zhu@dartmouth.edu)

Final Project: Create Your Own Physical World

(Final presentation at 3pm, November 25, 2019)

Project Summary

The goal of the final project is to build a small computational system to solve a specific physical problem, with some concrete application related to animation & simulation, interactive design, or robotic & engineering problems. The workload per person is approximately three times larger than one programming assignment. There are three main requirements for your simulation system:

1) Mathematical Model: The mathematical/physical model you implement should be complex enough and dynamically (visually) appealing to describe the evolution of a physical system. The system could be fluid, soft body, rigid body, coupled system, etc. You need to finish a math/physics note by writing down all the mathematical equations you plan to implement in your code. You may start from a technical paper or you may also extend the mathematical model from one of your homework.
2) Numerical Algorithm: You are expected to implement a complete physical simulation system centered around one (or multiple) numerical algorithm(s) to demonstrate the simulation effects. A complete simulation pipeline includes the following elements: 1) spatial discretization (e.g., particle, grid, ect.); 2) discrete differential operators (e.g., gradient, divergence, curl, Laplacian, etc.); 3) temporal evolution scheme (e.g., explicit, implicit). You are also expected to design acceleration strategies for the performance of your system by leveraging the techniques learned in class (e.g., data structure, high-performance coding, etc). You are expected to put the numerical schemes in your math notes as well.
3) Code Implementation: You will develop the simulation system in C++ to implement your numerical model. The code is expected to run interactively to generate realistic simulation effects. There is no requirement for the dimension of the problem (you may implement 2D code only, if that is enough to demonstrate your physical problem). The code is required to be connected to some visualization tool to show the motion of the system. By default, the visualizer is the OpenGL viewer provided in the starter code. But you can always use more advanced ones (such as Maya, Houdini, etc) for better effects.
4) Examples: You need to demonstrate your system using a number of simulation examples that is related to real-world physics.

Topics

You may refer to the reading list or the implementable paper list on Canvas to get some inspiration for the topics. You may also start from a specific physical phenomena and put together necessary techniques you have learned in class to model the phenomena in a computational environment.

Teamwork

Each team can have up to three members. Each team member is expected to make clear and independent contributions to the implementation of the system. The workload per person is approximately two to three times as the workload of one programming assignment.

Starter Code

You are welcome to use any asset provided with our starter code to build your simulation. The starter code can be downloaded from our GitLab starter code repository.

Report and Final Video

Submit your code and videos on Canvas along with a technical report including the following contents:

1. Simulation overview
   -- One paragraph summary of the theme and the technical contributions of your simulation
2. Implementation details
   -- Describe your math models, algorithms, and code implementations
3. Solved challenges
   -- Describe the challenges you had in your implementation and the way you solved them
4. Contributions
   -- Describe each team member's role and contribution

Grading

The grading policy for the total 35% of this assignment includes:

- Mathematical model (5%)
  - Pick appropriate mathematical models to describe the target physical system
  - Develop a note including all equations and numerical schemes to guide your implementation
- Numerical algorithms (10%)
  - Develop appropriate numerical algorithms for discretization, differential operators, time integrations, and other necessary techniques to implement the physical model
  - Consider the generality and performance in your algorithm design
- Code implementation and system performance (10%)
  - Implement the proposed algorithms in a code environment and visualize the results in a visually attractive way

- Creativity and complexity of the simulation examples (10%)
  - Demonstrate the algorithms by simulating a physical system with enough complexity
  - Create scenes that are visually appealing or dynamically interesting
- Extra credits (up to 5%)
  - Extra credits will be given to students who are tackling challenges beyond the aforementioned requirements. Examples of these challenges include the extraordinary complexities of the mathematical models, difficulties in code implementation, system's performance optimization, outstanding visual effects, the integration of multiple different systems and algorithms, etc.

Checkpoints

We will have three project checkpoints in the next three X-hours. The main goal of each checkpoint is to help you reach the milestone of each part of the project and to give you instant feedback on the next steps you need to take. An early grade will be given to each part of math, numerics, code, and examples in the three checkpoint sessions. Along with the early grade, we will also give you suggestions for next steps to perfect your final version. The checkpoint grade does not count for your final grade but will be a sign of whether your project is going well on the current stage.

Progress Sharing in X-Hour

We encourage teams to share their progress in X-Hours and lead the momentum of the final project development in the class. You may share your math models, preliminary results, the current challenges you are facing, or any thoughts you have with the class. Each student who shares the project progress will get 1% extra credit (in addition to the maximum 5% extra credit).

Timeline

We will have a final presentation at 3pm on November 25, 2019 in the same classroom.

Submit the code and the final report on Canvas before 3pm, November 26.