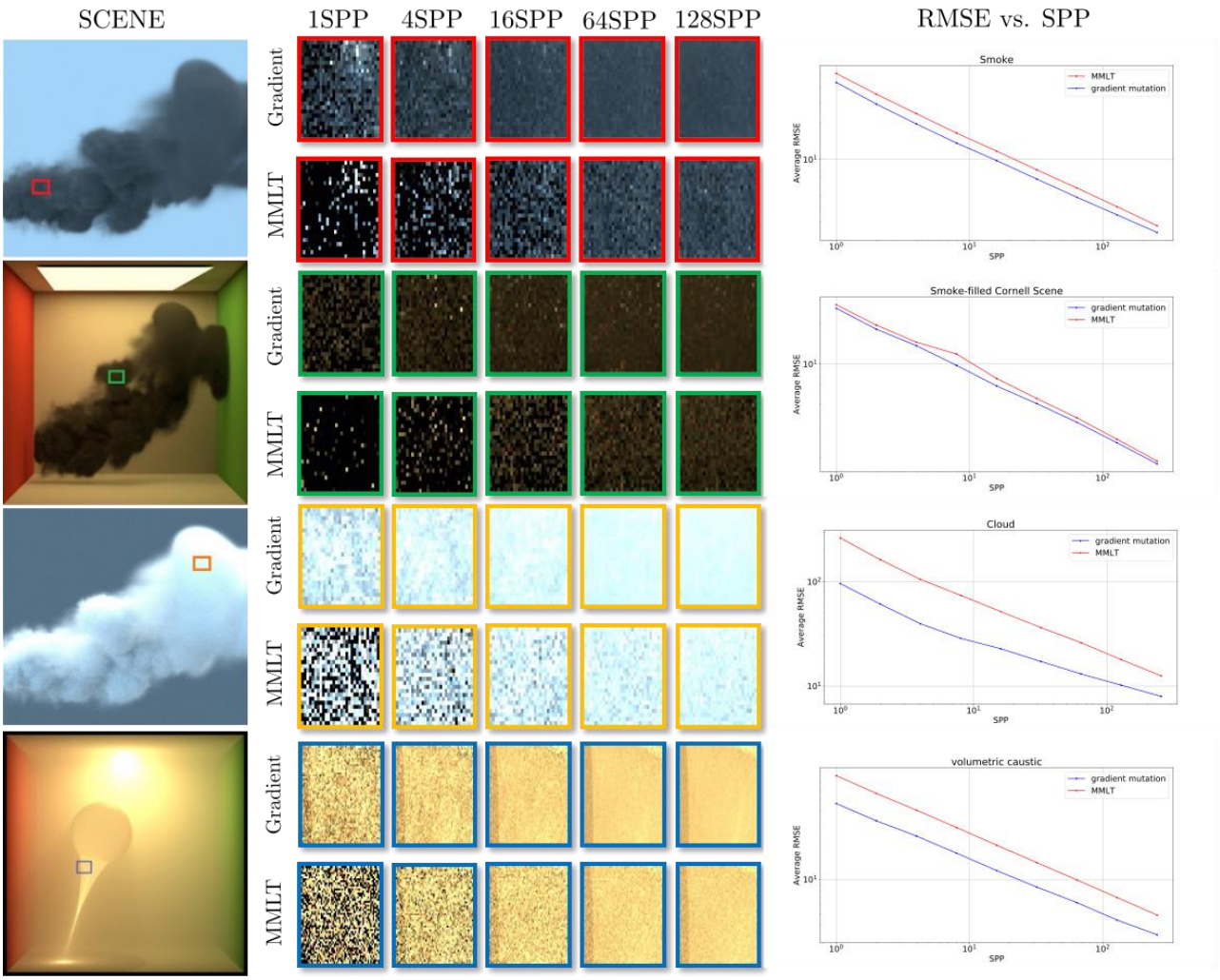
(Peter)G.Lu

Project portfolio

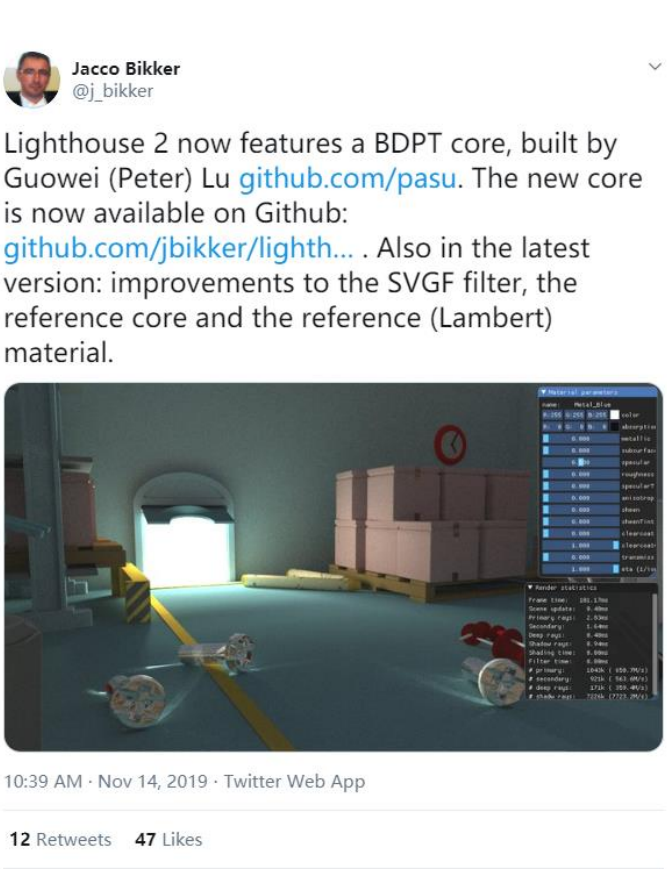
Master's thesis project
Gradient-Domain Volume Rendering

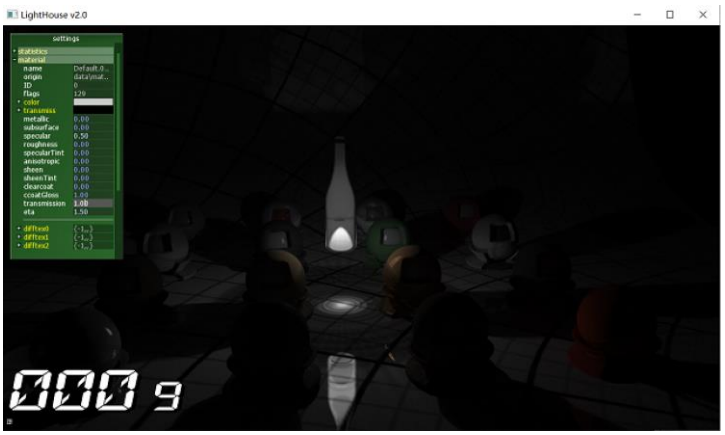


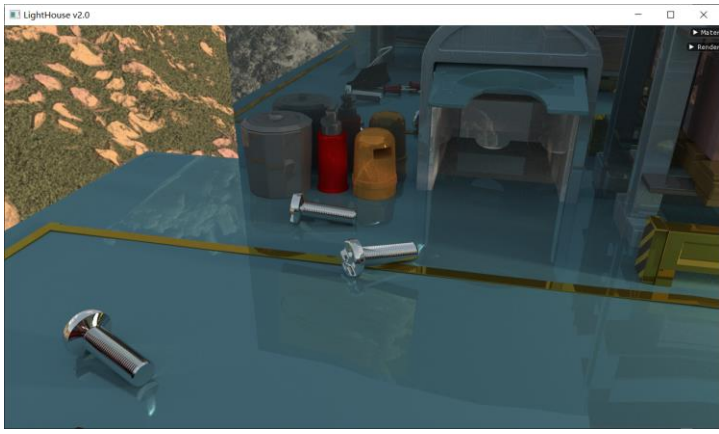
My master thesis 'Gradient-Domain Volume Rendering' revolves around computing the path-space gradient of the light path with respect to one vertex in the medium and using this gradient to estimate the contribution of correlated paths in Coherent Metropolis Light Transport. We can do so with limited computation, as we avoid the expensive construction and radiance evaluation of entirely new paths.

GitHub PDF Slides

Small project
Streaming BDPT based on Light House 2







Light House 2 is a rendering framework for real-time ray tracing. I built this streaming bidirectional path tracing render system based on Light House 2, which supports energy conservation, caustic, and a wavefront pipeline under the Optix framework.

Major skills include:

- C++
- CUDA
- OptiX
- Wavefront
- Batching Ray for visibility test

GitHub PDF Video

Path Tracer (GPU&WebGPU)

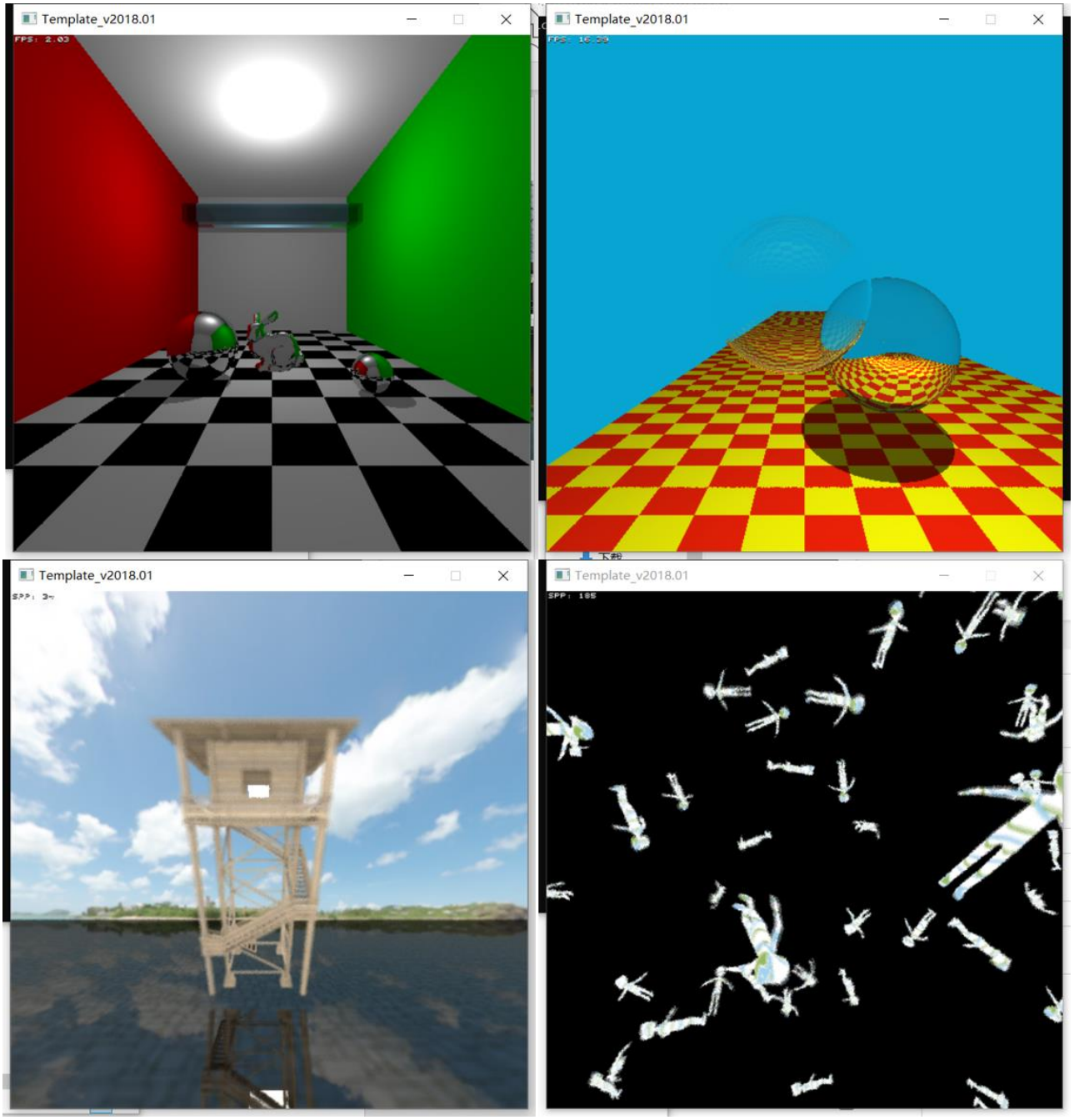
C++/JS/Compute Shader

A path tracing toy written in C++, JS and compute shaders. You can build the BVH of the scene and save it as binary on the desktop, then load this binary and view the scene interactively on the Browser. It supports wavefront and megakernel.

GitHub Demo

Path Tracer (CPU)

C++



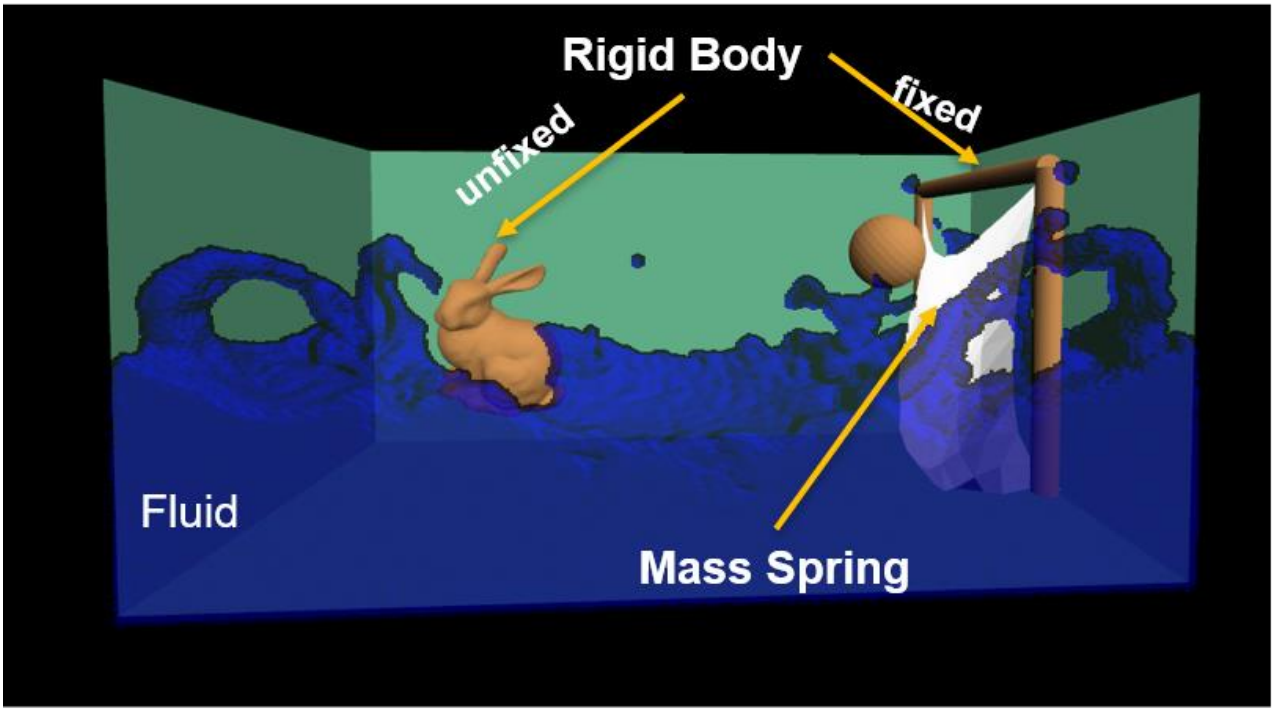
This is a fully featured ray tracer for the assignment in the course 'Advanced Graphics'. Major features are:

- BVH (SAH + Top-Level + SIMD Intersection)
- NEE+MIS, Photon Mapping (Simple), Ray Packets
- Depth Field, Motion Blur
- Multithreading, Filter

GitHub Video

Position Based Fluid Simulation

C++/Compute shader/OpenGL



Particals	Time Per Iteration	Frame Rate
64K	1.3ms	110fps
128K	2.0ms	70fps

This is the Mini-project in the course 'Game Physics', which was selected into the **Game Physics Hall of Fame 2019** (2nd place). The key idea is enforcing incompressibility using position constraints. It includes fluid-rigid body, fluid-cloth interaction and fluid surface reconstruction.

GitHub Video

Rigid and Soft Body Simulation

C++

This is the assignment in the course ‘Game Physics’. It includes three parts:

- Rigid bodies and collision
  - multiple interpenetrations and collisions
  - impulse-based collision resolution
- Constraints
- Finite-Element Soft-Body Deformation

Rigid bodies and collision: GitHub

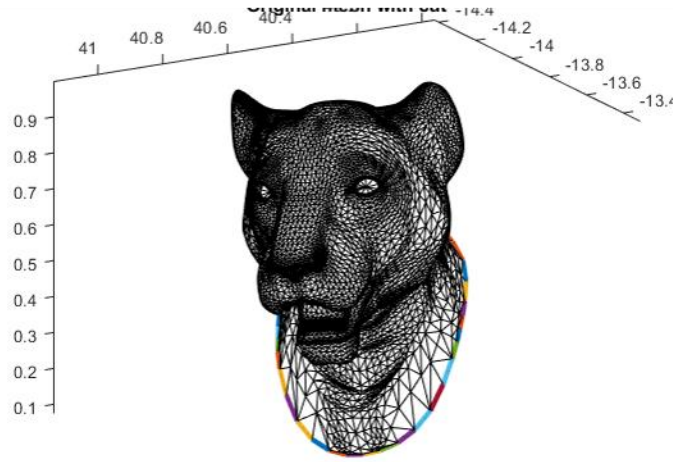
Constraints : GitHub

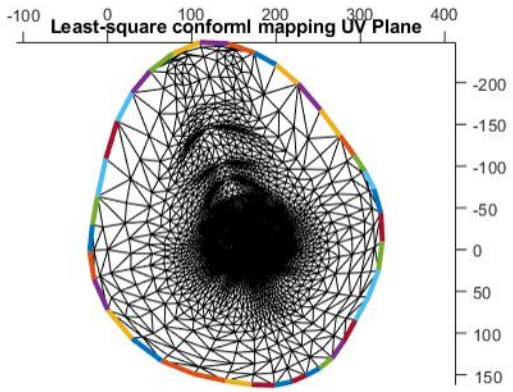
Soft-Body Deformation: GitHub

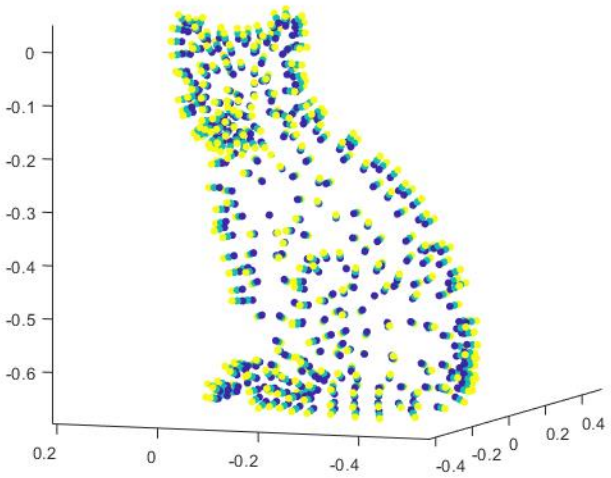
Video

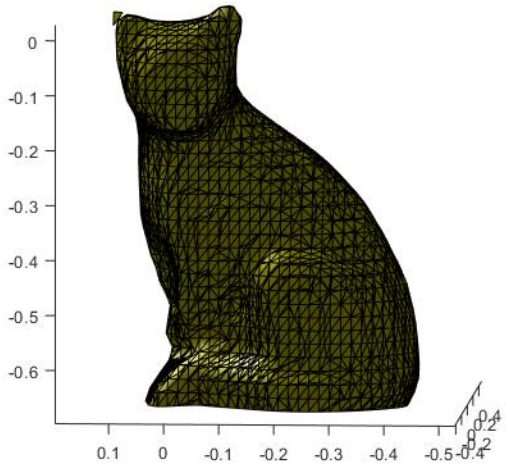
3D Model

matlab





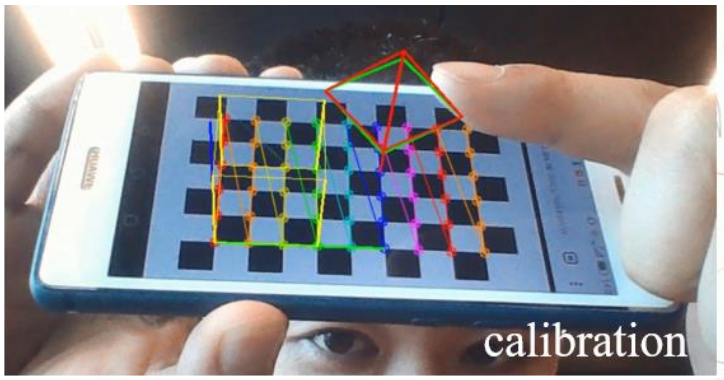


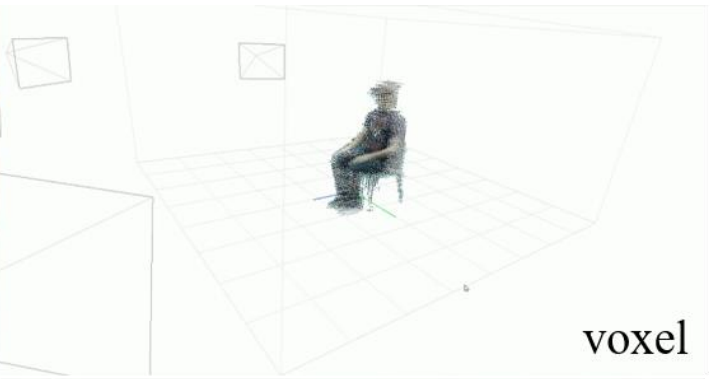


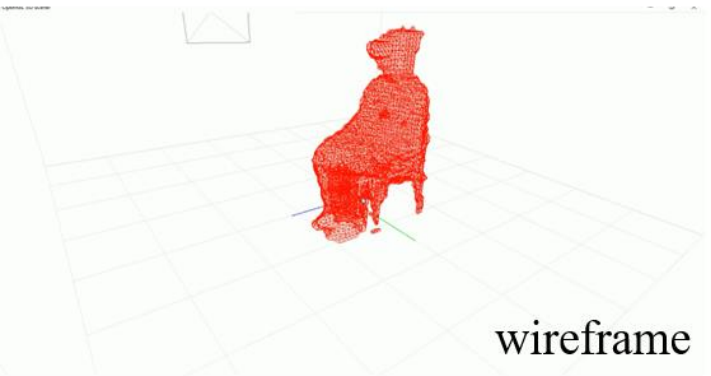
I self-studied the course '3D model' and did these practices including linear least-squares system using sparse matrices, a simple version of the moving least squares, and Least-Squares Conformal Mapping algorithm. **Academic integrity**: I did the Least-Squares Conformal Mapping practice under the help of reference codes. Strictly speaking, this part is not my own code.

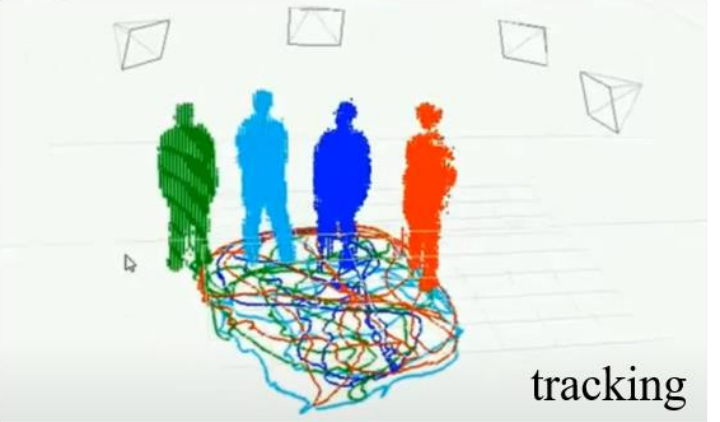
Video

Voxel-based 3D Reconstruction and Tracking C++/OpenCV







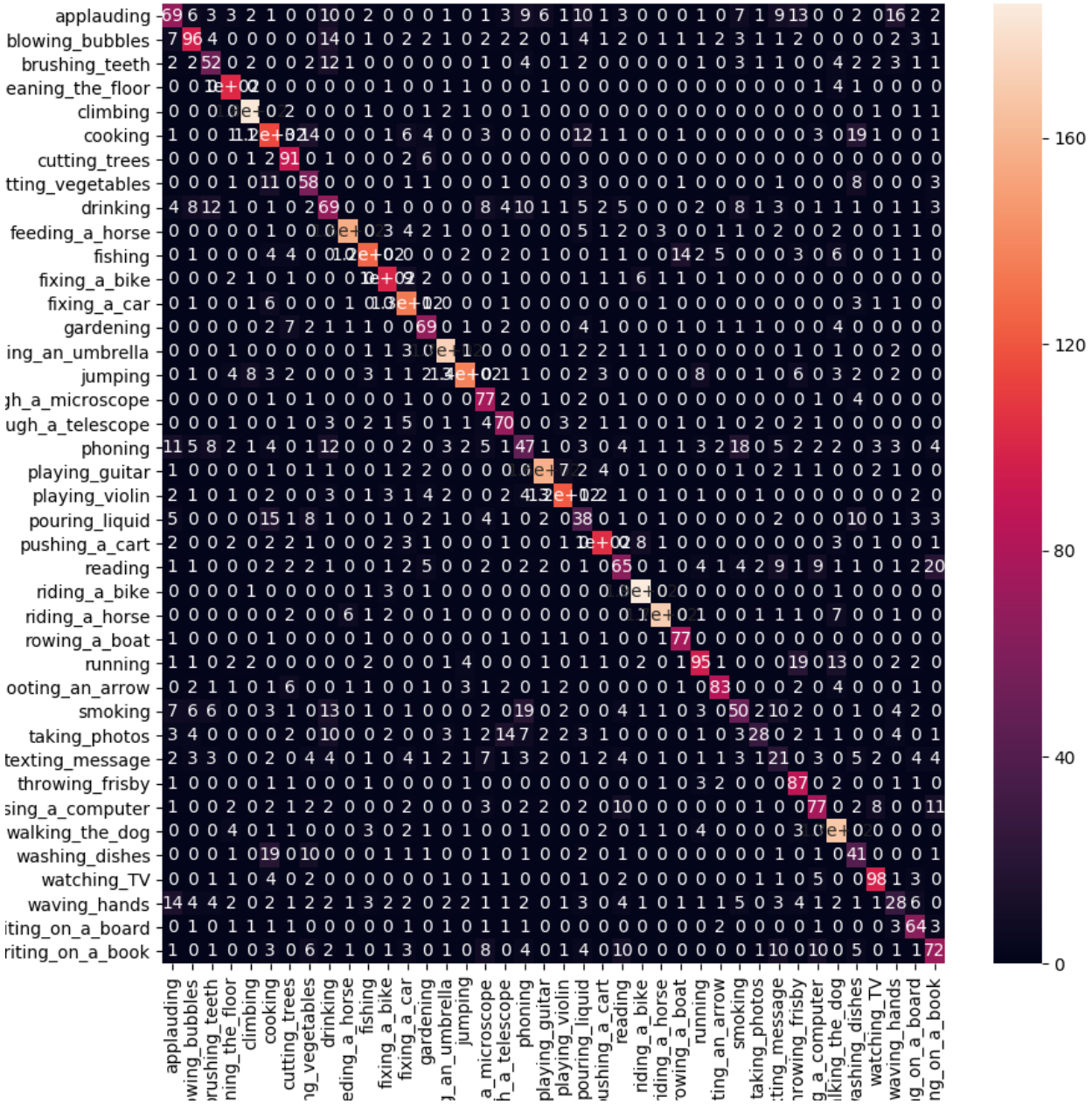


This is the assignment in the course ‘Computer Vision’, which includes the camera calibration, voxel-based 3D reconstruction and tracking. The major skills are:

- Marching cube
- K-means

Video

Action Recognition with Automatic Model Search
Python/Keras/Tensorflow

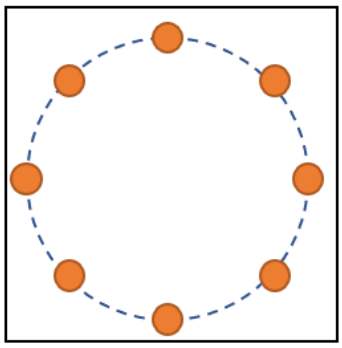


This is the project in the course ‘Computer Vision’. We created a Neural Network architecture capable of classifying human actions of the Stanford-40 dataset and optimized it by using methods such as Transfer Learning, weight decay, and custom learning rates.

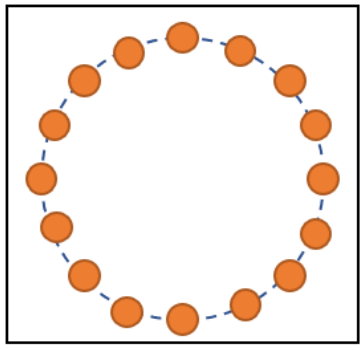
GitHub PDF

A Comparative Study of Collision Avoidance Algorithms

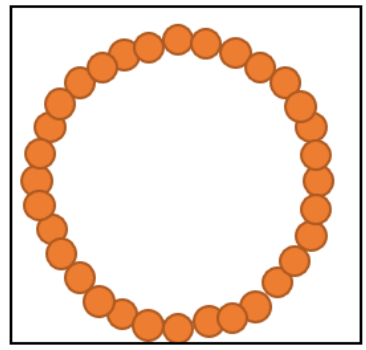
C++



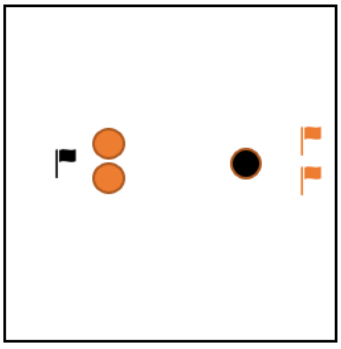
8-way-confusion



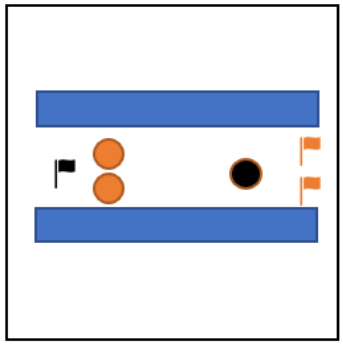
16-way-confusion



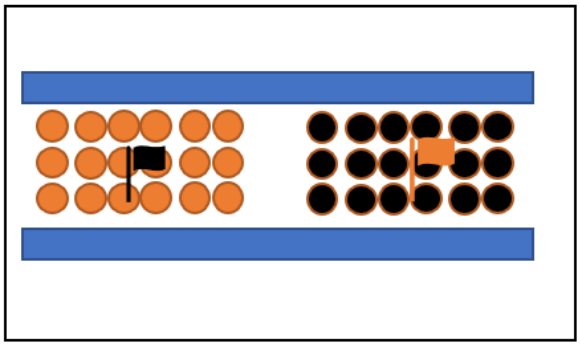
32-way-confusion



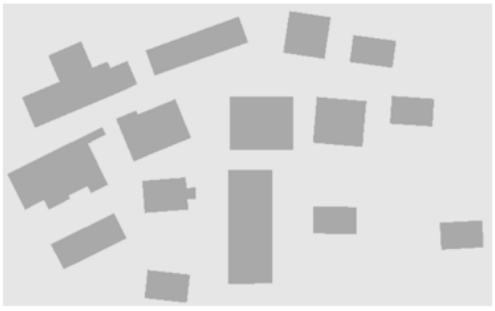
crossing



Squeeze crossing



Group intersection

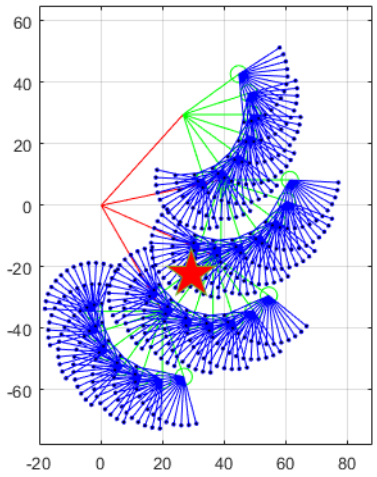


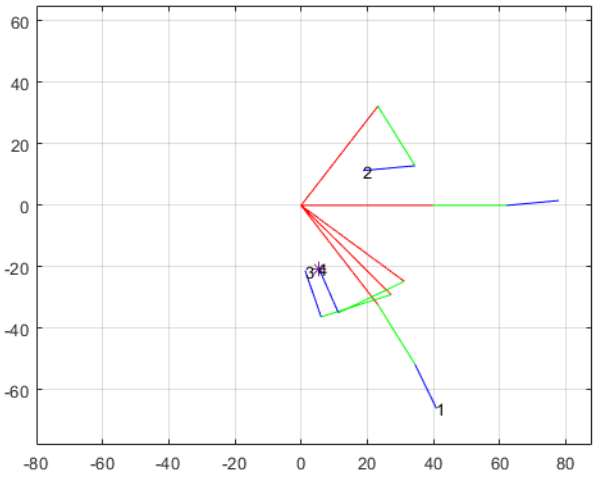
Large scale scenario

This is the project in the course 'Crowd Simulation', we added the *implicit crowd algorithm* to the UUCS(Utrecht University Crowd Simulation) framework. UUCS is a closed source project. We compared this algorithm with other similar algorithms in several test scenes.

PDF Video

Inverse Kinematics for Human Fingers
matlab





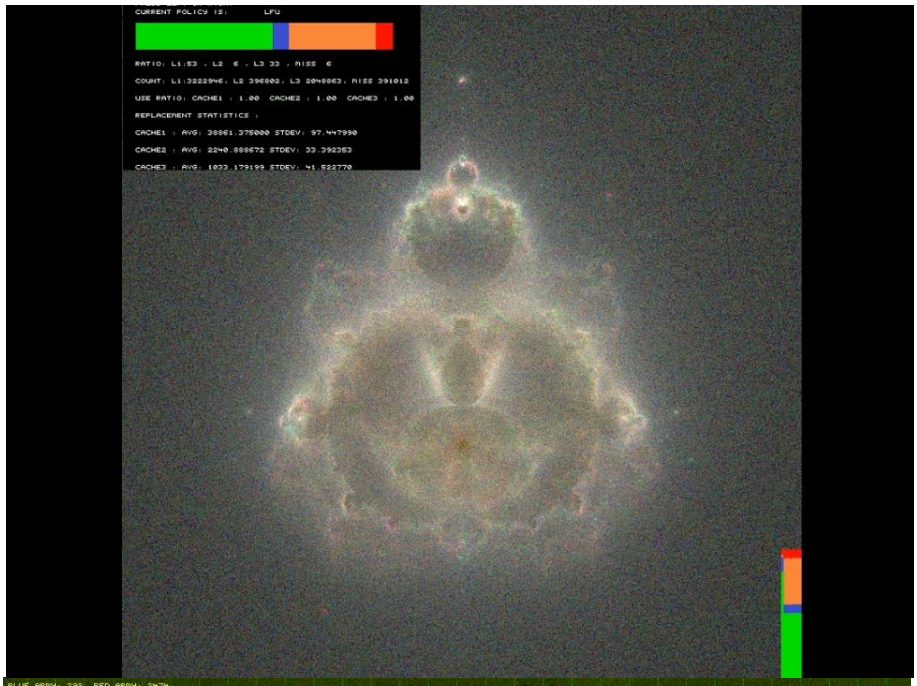
This is the project in the course 'Motion and Manipulation'. First, we simulate the positions of one finger with three joints where it can reach. And we use the inverse Kinematics method to compute the angle of each joint iteratively to reach one given position with the constrained joint angles. We provided three solvers:

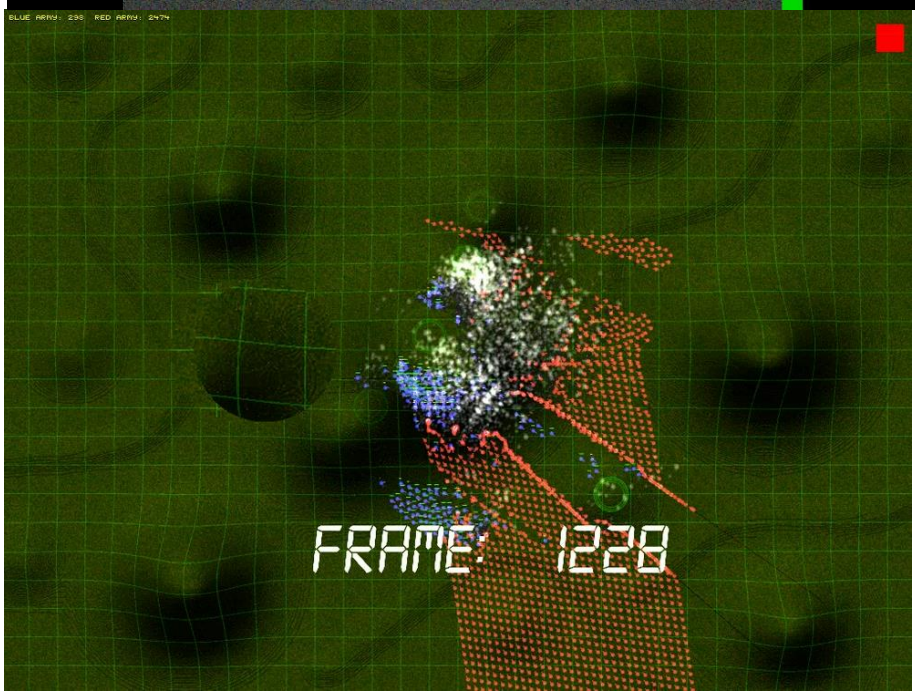
- Pseudo Inverse
- Pseudo Inverse with Optimization Derivation
- Extended Jacobian Method

GitHub PDF Video

CPU Cache Simulator and Optimization

C++





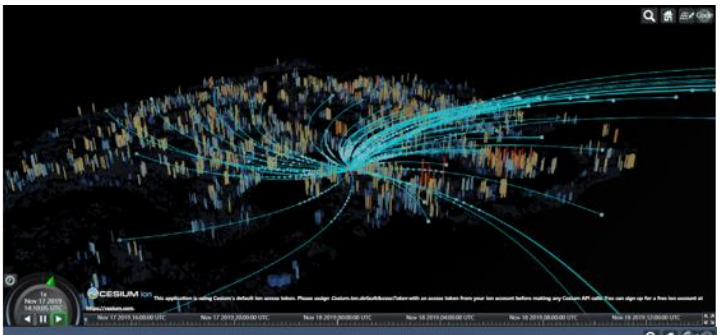
This is the assignment in the course ‘Optimization and Vectorization’. In the cache simulator, there are three levels cache, and we provide six eviction policies. A tool is provided to monitor the hit ratio. Finally, we can use all the skills we learned to optimize the performance of a small game. I speedup 25 times

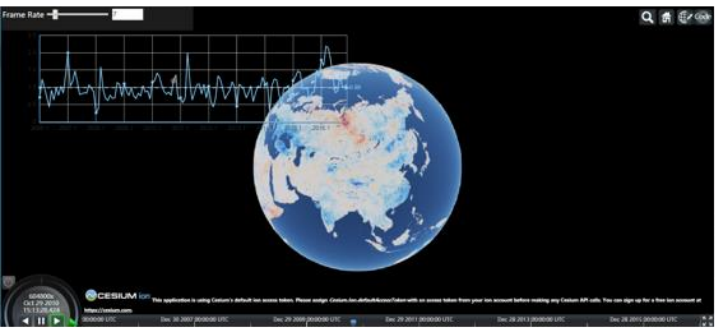
GitHub Video

Examples for Cesium

JS/WebGL

GitHub | Watch 42 | Star 320 | Fork 135



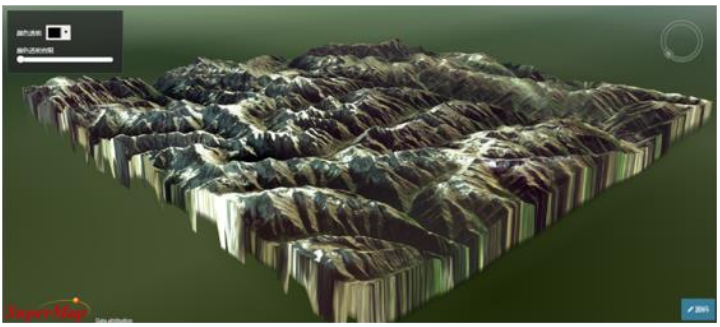


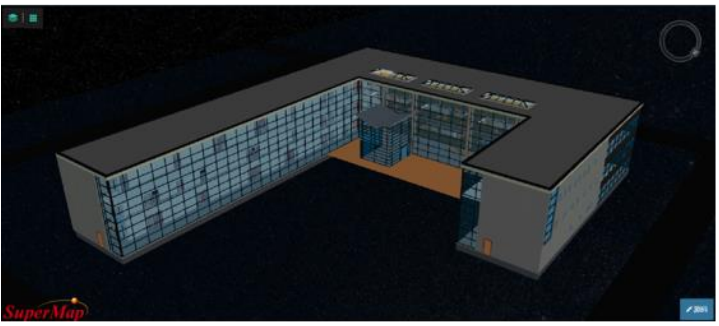
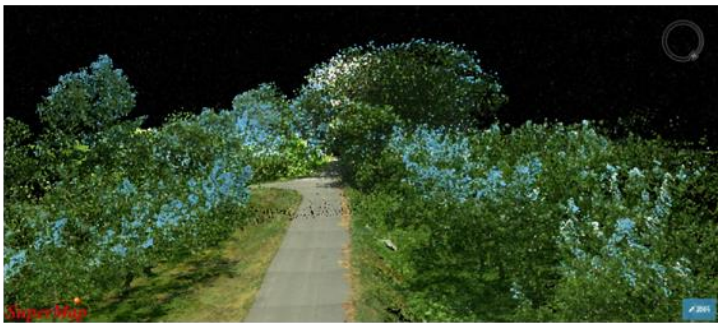
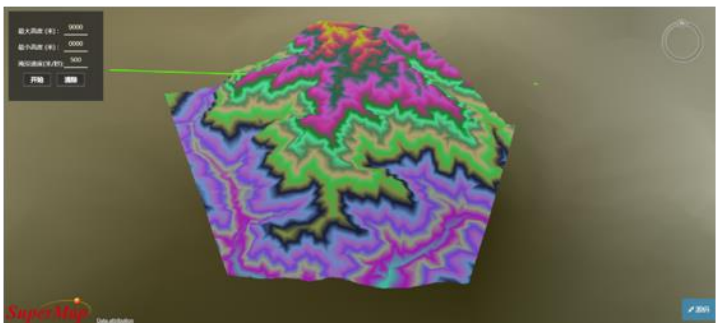
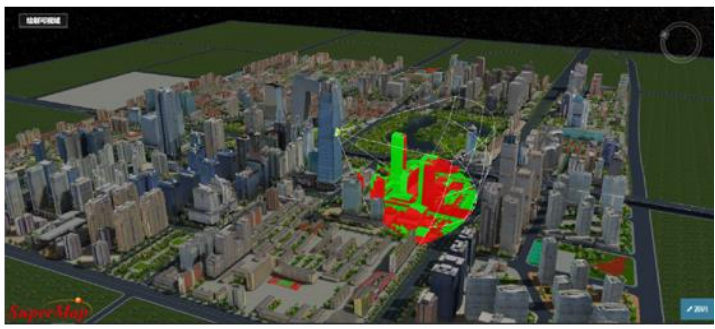
CesiumJS is an open-source JavaScript library for creating world-class 3D globes and maps. I create several visualization demos based on this library, such as real-time global AQI(air quality index), dynamic tax flow, ten years of global temperature change, and global terrain. I am also a little contributor to this library.

GitHub | Video

S3M

C++/JS/WebGL



S3M is one specification (T/CAGIS 1-2019) in China for the transmission and loading of massive meshes, including oblique photographs, BIM and point clouds. We can do the analysis and visualization on this Spatial geographic data. I was the core engineer to design and develop this format, including the data generation, loading and rendering on the Browser.

GitHub