# World Space ReSTIR

Jichu Mao Zhiyi Zhou

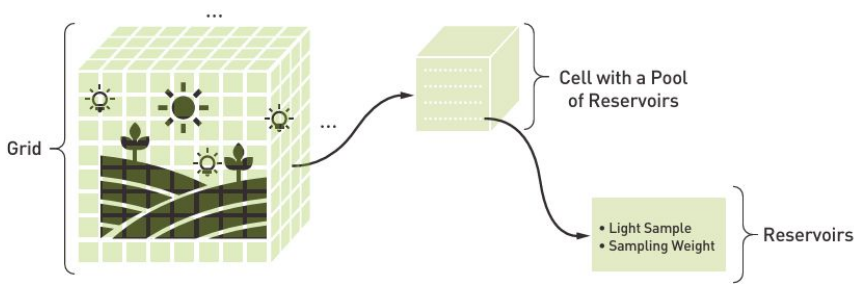
CIS 5650 – Final Project

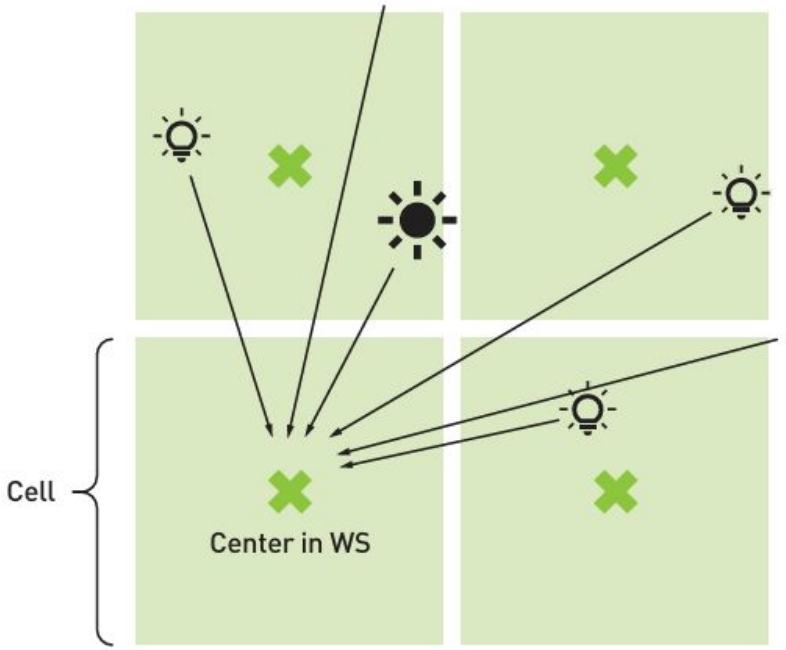
Penn Engineering

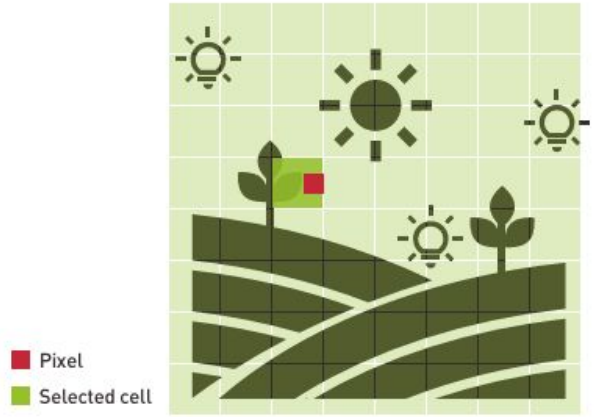
# Background

## What is GRID-BASED RESERVOIRS(ReGIR)?

ReGIR (Reservoir Grid Importance Resampling) is an algorithm designed for **efficiently rendering scenes with many light sources** in real-time ray tracing. It builds upon existing techniques such as ReSTIR (Reservoir Spatiotemporal Importance Resampling) and applies them to world-space sampling using a grid-based structure to optimize light sampling for secondary rays.




Penn Engineering

# Why this Project Matters?

- Original **ReSTIR GI** Operates in **screen space**. It optimizes global illumination sampling for primary rays and uses spatiotemporal resampling to manage indirect illumination. The reservoirs are tied to screen-space pixels. **ReGIR** Uses **world space**. It divides the scene into a 3D grid, with reservoirs distributed across these grid cells.
- We want to learn more about Vulkan as a modern graphics API and leverage hardware-accelerated ray tracing to achieve better performance.

## Goal

- Implement **ReGIR** in Vulkan **to** optimize multi-light rendering and produce high-quality visuals under strict performance constraints.
- Optimization to the origin algorithm, e.g. adaptive grid resolution, time stability enhancement, efficient memory management

# Schedule

Milestone 1(Nov 04 - 13)

- Be familiar with base code and the structure of thesis algorithms
- Implement world space grid construction
- Implement light importance sample

Milestone 2(Nov 14 - 25)

- Implement Reservoir based resampled importance sampling(RIS)
- Implement ReGIR DI

Milestone 3(Nov 26 - Dec 02)

- Implement ReGIR GI

Final Due(Dec 08)

- Complete any remaining items left over from the previous
- The final testing and fixes

# Additional Resources

## API & Platforms

- Vulkan
- Visual Studio 2022

## Third-party Code(planning)

- Base Code of Renderer

## Reference

- ManyLight ReGIR Paper
- Reference Projects

# Project Pitches – Do

- Keep within 1 page (for doc) or 4 slides – **Conciseness is best**
- Have **clear goals and outcomes**
- **Clear application of GPU programming**
- 1-2 paragraphs about **why this project matters**
- A schedule for Milestones 1-3, and Final
- APIs, Platforms etc. you are planning to use
- Include references, links to content/inspiration, picture is fine too
- Any third-party code you are planning to use



Penn Engineering

# Project Pitches – Don't

- Do not include math equations/code etc
- Do not exceed 2 pages / 5 slides
- Don't bring repeat ideas from previous years
- Don't come under-prepared – be ready to defend why you think your project should be accepted

Penn Engineering

Penn Engineering