Differentiable Ray Tracing Simulator

Maxime Vandegar, Michael Kagan

January 2021

U.S. DEPARTMENT OF ENERGY
Office of Science

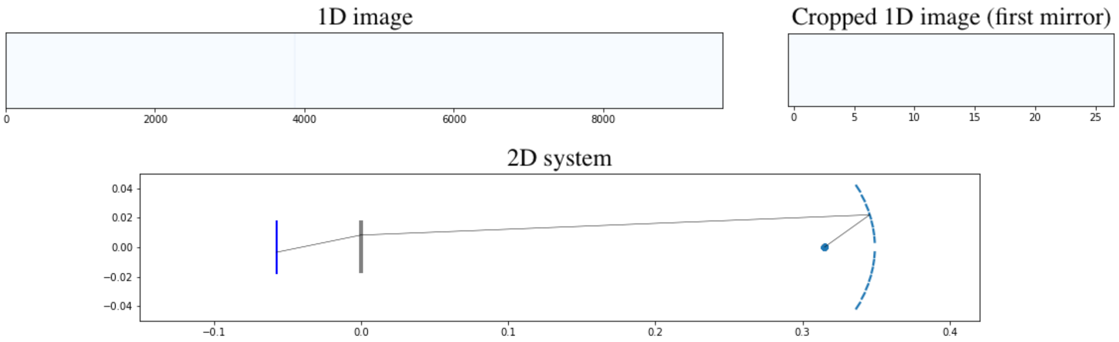
SLAC NATIONAL ACCELERATOR LABORATORY

Plan

- Build a simulator of the MAGIS optical system.
- Insights:
  - Early insights about the system.
    - e.g. what is the impact of noise, quantum efficiency, …
  - Ability to test different systems.
    - e.g. different lenses, numerical apertures, …
- Inference:
  - High-fidelity model of the projection operator that can be used at inference time.
    - $\mathrm{A}(x) = b$.
  - End-to-end differentiable.

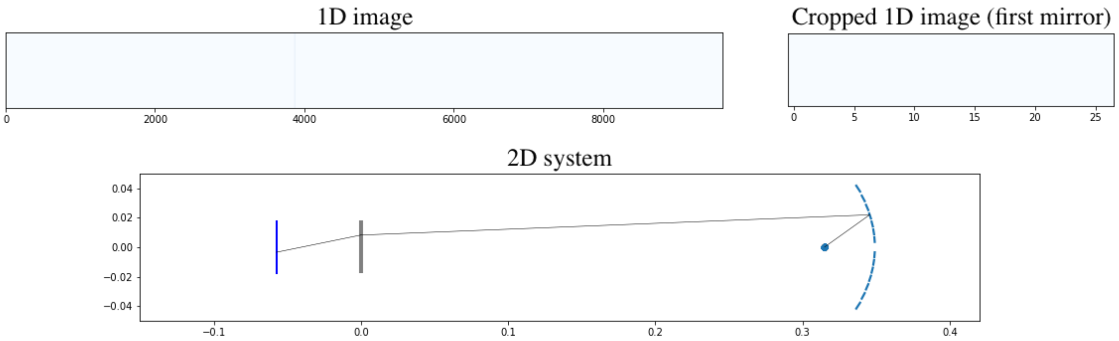
Approach

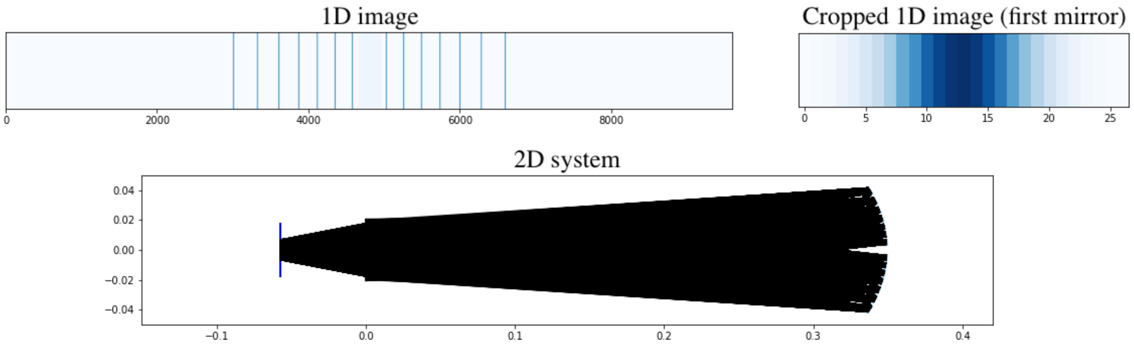
- Sample light rays from the atom cloud and model their interactions with the system (physically based rendering).



Approach

- Sample light rays from the atom cloud and model their interactions with the system (physically based rendering).





Next steps

1. Finish the extension of the simulator to 3D.
2. Complexify the system (noise, quantum efficiency, …).
3. 3D reconstruction & design of experiments.

Backup slides

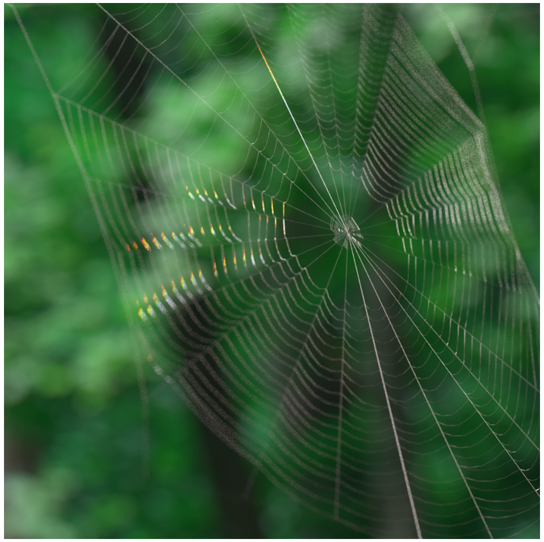
JAX

- Written in JAX (Autograd & XLA).
  - Autograd:
    - Automatically differentiates native Python and Numpy code.
    - Main purpose: gradient-based optimization.
  - XLA:
    - Compiles and runs programs on GPUs and TPUs (fused operations).
- Functional programming paradigm.
- Numpy (Python library) like syntax.
- Automatic parallelization & vectorization.

- Differentiability could also be used for design optimization.

Physically Based Rendering (1)

- Computer graphics approach that render images by modeling the behaviour of light rays in the real world.

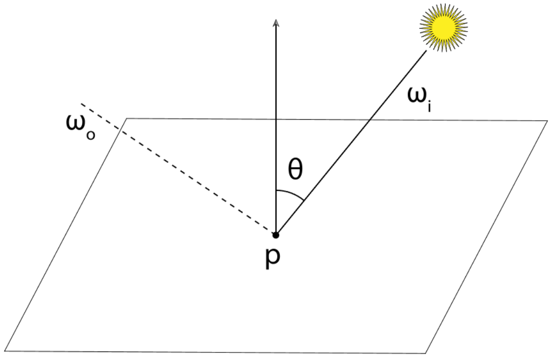


CS348B: Zach DeVito implemented a system for automatic spider web generation and then simulated wavelength dependent refraction through the web's threads.

Credits: https://graphics.stanford.edu/courses/cs348b-competition/cs348b-09/

Physically Based Rendering (2)

- Sample light rays from the camera and trace them until they hit a light source.
- Model the interaction of light rays with materials.
- Need to find which object in the scene a light ray will intersect first.
  - $\mathcal{O}(N)$ *where* N *is the number of objects in the scene.*
    - Not computational efficient.
      - Can be improved with exact or approximated algorithms.
    - We know exactly the setup.
      - Intersections can be computed in $\mathcal{O}(1)$.



Credits: *Physically Based Rendering: From Theory To Implementation.* Matt Pharr, Wenzel Jakob, and Greg Humphreys.