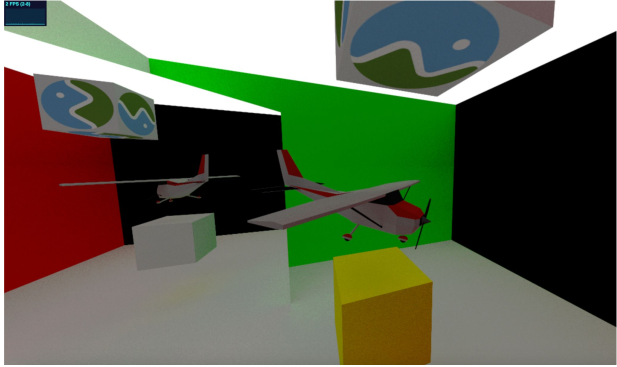
WebGPU Pathtracer + NPR + Cloth Sim

Milestone 2

Alan Lee, Jordan Hochman, Maya Diaz Huizar

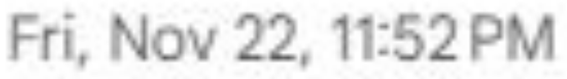
Pathtracer Updates



- GLTF & texture loading and BVH generation integrated to main
- Lambertian and metallic materials fixed
- Sample accumulation added
- NPR pipeline implemented
- Cloth Simulation pipeline in progress

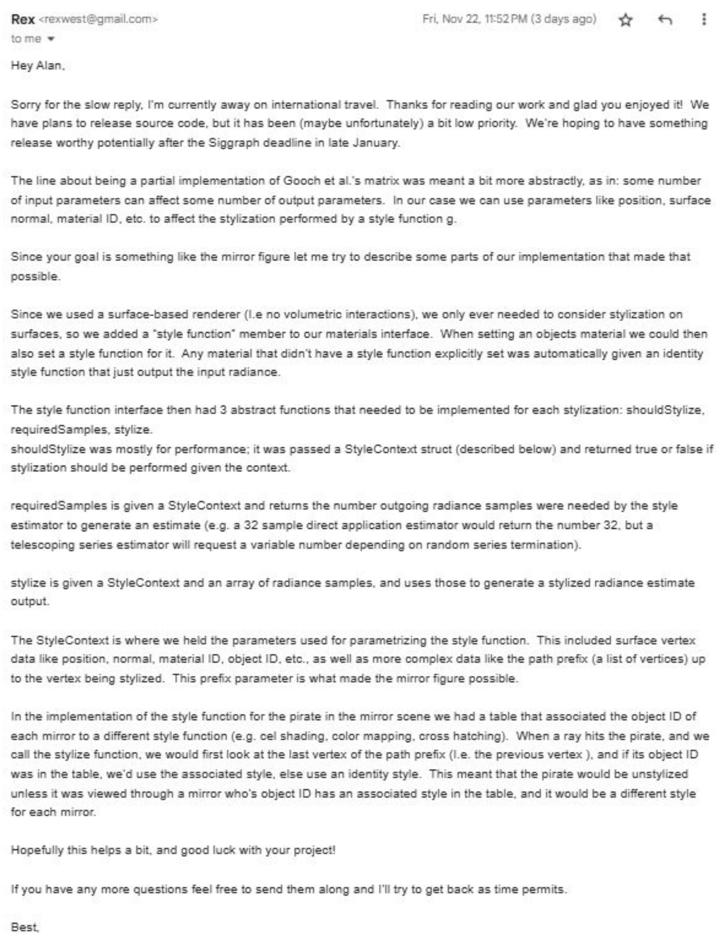
NPR

- Good news, the first author responded to our email
- Bad news, the reply came at Friday midnight



Fri, Nov 22, 11:52 PM

- Had to redo a lot of work but good to know we have a verified working direction



Rex <rexwest@gmail.com>

to me

Fri, Nov 22, 11:52 PM (3 days ago)

Hey Alan,

Sorry for the slow reply, I'm currently away on international travel. Thanks for reading our work and glad you enjoyed it! We have plans to release source code, but it has been (maybe unfortunately) a bit low priority. We're hoping to have something release worthy potentially after the Siggraph deadline in late January.

The line about being a partial implementation of Gooch et al.'s matrix was meant a bit more abstractly, as in: some number of input parameters can affect some number of output parameters. In our case we can use parameters like position, surface normal, material ID, etc. to affect the stylization performed by a style function g.

Since your goal is something like the mirror figure let me try to describe some parts of our implementation that made that possible.

Since we used a surface-based renderer (I.e no volumetric interactions), we only ever needed to consider stylization on surfaces, so we added a "style function" member to our materials interface. When setting an objects material we could then also set a style function for it. Any material that didn't have a style function explicitly set was automatically given an identity style function that just output the input radiance.

The style function interface then had 3 abstract functions that needed to be implemented for each stylization: shouldStylize, requiredSamples, stylize.
shouldStylize was mostly for performance; it was passed a StyleContext struct (described below) and returned true or false if stylization should be performed given the context.

requiredSamples is given a StyleContext and returns the number outgoing radiance samples were needed by the style estimator to generate an estimate (e.g. a 32 sample direct application estimator would return the number 32, but a telescoping series estimator will request a variable number depending on random series termination).

stylize is given a StyleContext and an array of radiance samples, and uses those to generate a stylized radiance estimate output.

The StyleContext is where we held the parameters used for parametrizing the style function. This included surface vertex data like position, normal, material ID, object ID, etc., as well as more complex data like the path prefix (a list of vertices) up to the vertex being stylized. This prefix parameter is what made the mirror figure possible.

In the implementation of the style function for the pirate in the mirror scene we had a table that associated the object ID of each mirror to a different style function (e.g. cel shading, color mapping, cross hatching). When a ray hits the pirate, and we call the stylize function, we would first look at the last vertex of the path prefix (I.e. the previous vertex), and if its object ID was in the table, we'd use the associated style, else use an identity style. This meant that the pirate would be unstylized unless it was viewed through a mirror who's object ID has an associated style in the table, and it would be a different style for each mirror.

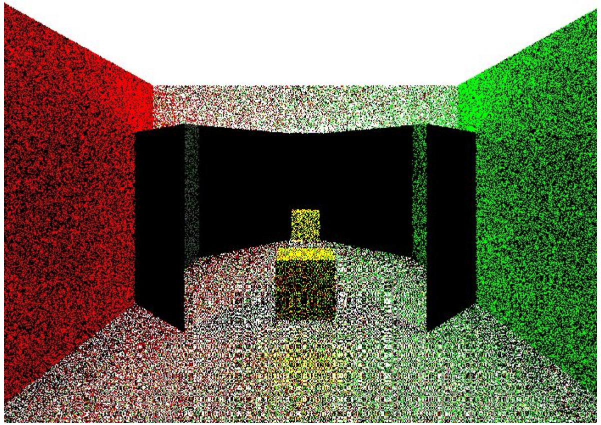
Hopefully this helps a bit, and good luck with your project!

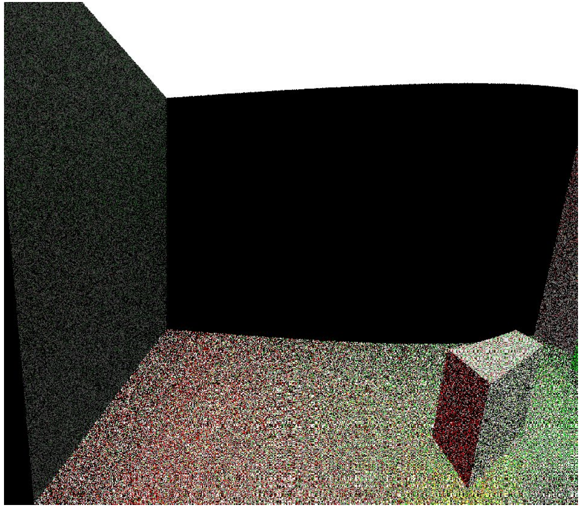
If you have any more questions feel free to send them along and I'll try to get back as time permits.

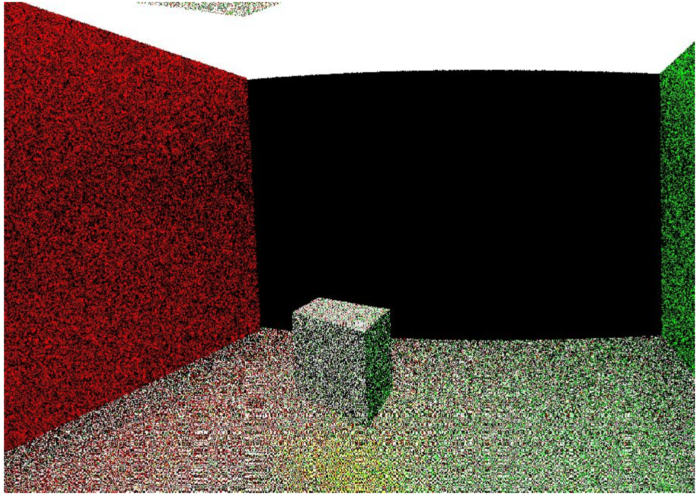
Best,

NPR

- shouldStylize, requiredSamples, stylize
- Not sure the usage of requiredSamples
- Basic implementation, only greyscale so far

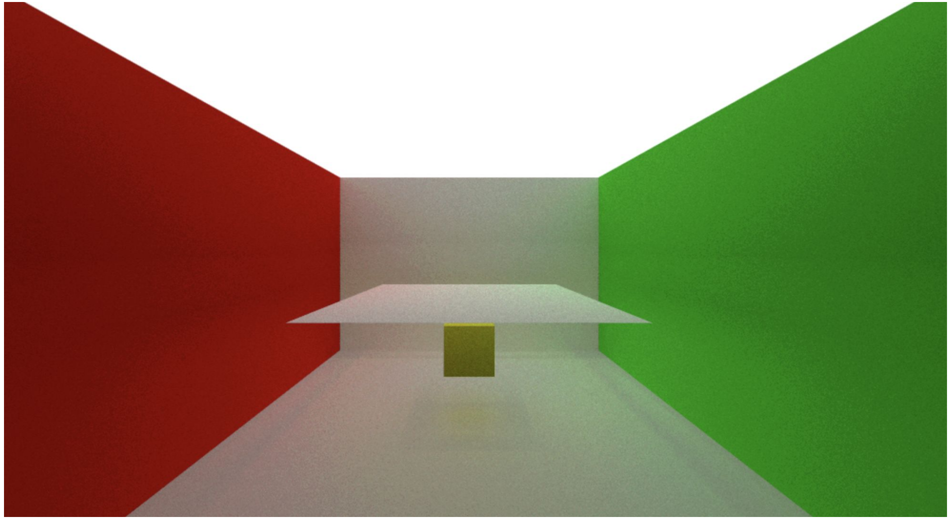






Cloth Simulation

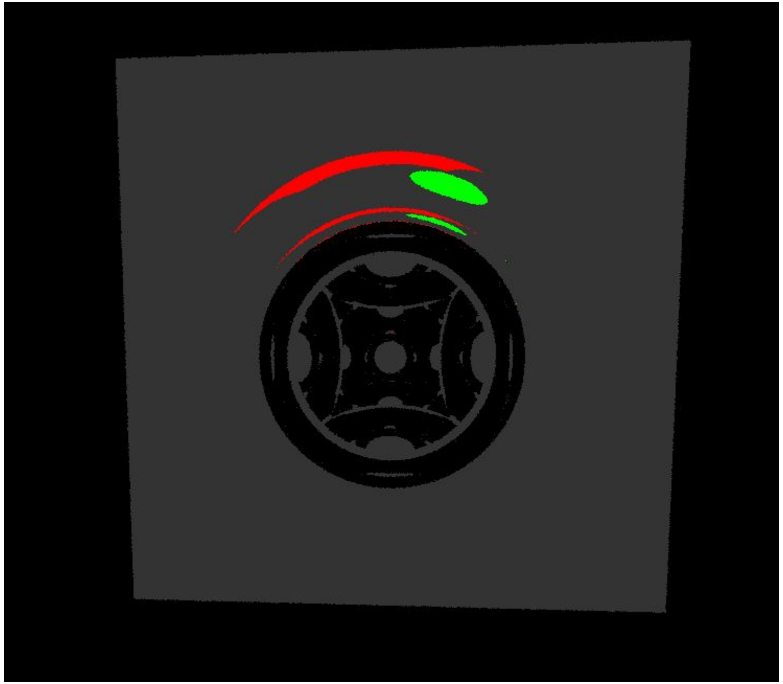
- Progress integrating cloth simulation pipeline
- Compute shader is set up, work to be done still to improve
  - Physics not completed yet
  - Need to shift to progressive dynamic simulation (from the paper)

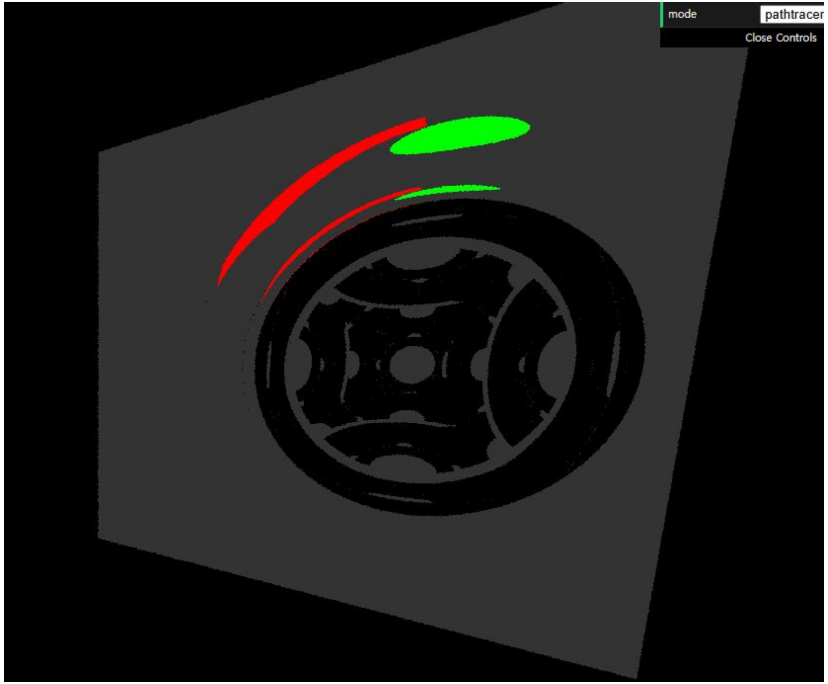


Milestone 3 TODOs

1. Explore more NPR stylization configurations (cross hatching, cel shading, etc)
2. Finish fully progressive cloth simulation
    - Add NPR styles to cloth

Bloopers



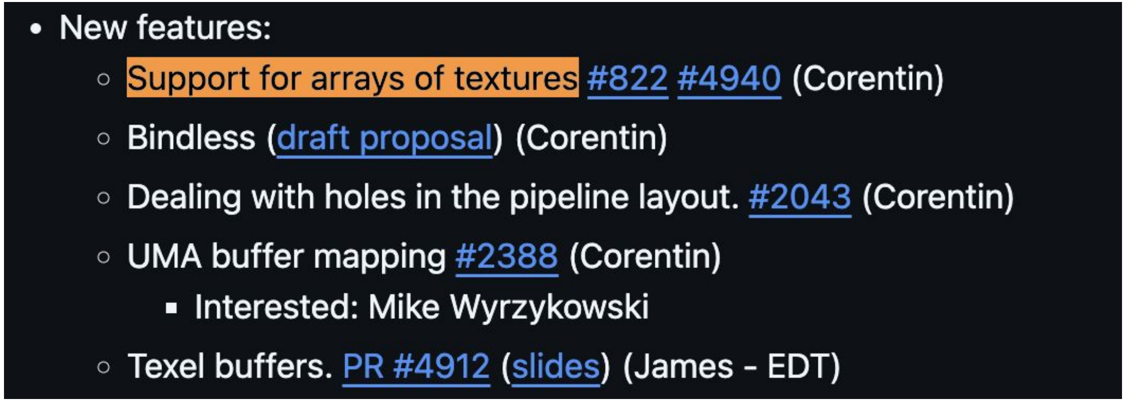


mode: pathtracer

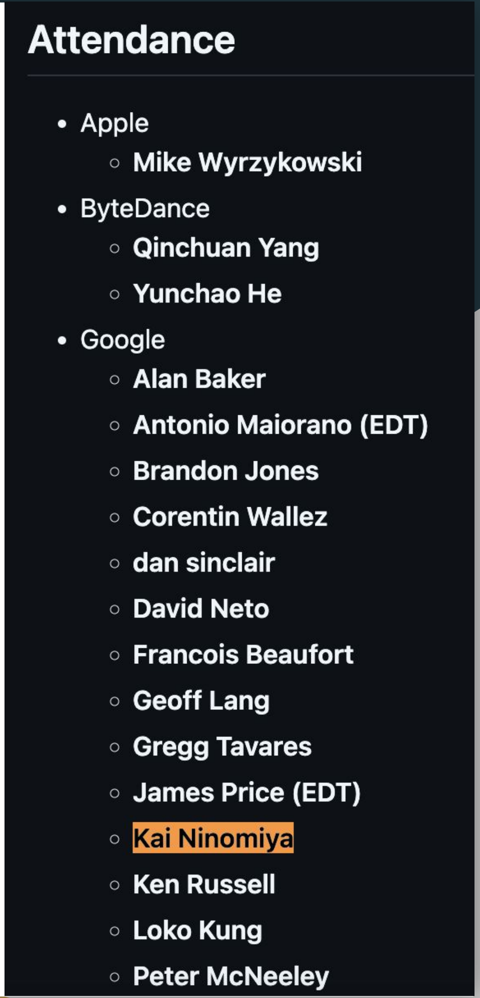
Interesting Extras

WebGPU doesn’t support arrays of textures!

- Had to implement our own texturing system with everything in one buffer
- Support is being added as of 3 days ago!
    - https://github.com/gpuweb/gpuweb/issues/822



- New features:
    - Support for arrays of textures #822 #4940 (Corentin)
    - Bindless (draft proposal) (Corentin)
    - Dealing with holes in the pipeline layout. #2043 (Corentin)
    - UMA buffer mapping #2388 (Corentin)
        - Interested: Mike Wyrzykowski
    - Texel buffers. PR #4912 (slides) (James - EDT)



Attendance

- Apple
    - Mike Wyrzykowski
- ByteDance
    - Qinchuan Yang
    - Yunchao He
- Google
    - Alan Baker
    - Antonio Maiorano (EDT)
    - Brandon Jones
    - Corentin Wallez
    - dan sinclair
    - David Neto
    - Francois Beaufort
    - Geoff Lang
    - Gregg Tavares
    - James Price (EDT)
    - Kai Ninomiya
    - Ken Russell
    - Loko Kung
    - Peter McNeeley