universität freiburg

SWIFT-HEP 2025

Optical Photon Propagation

Andrew Gait[1], Ian Hinder[1], Marco Gersabeck[2], Conor Fitzpatrick[3]

[1]Research Software Engineering, Research IT, University of Manchester

[2]Physikalisches Institut, Universität Freiburg

[3]Department of Physics, University of Manchester

11th April 2025

universität freiburg

MANCHESTER 1824
The University of Manchester

RSE

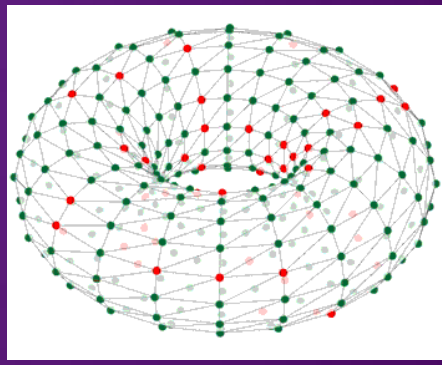
About me

- PhD in Geophysics (University of Leeds)
- Been at University of Manchester since 2007 working across multiple disciplines on various large software projects including applied mathematics, medical imaging, computer science and neural modelling

oomph-lib

VXL

SpiNNaker
Biologically Inspired Massively Parallel Architectures



- Joined the (central) RSE team in 2023, but have technically really been an RSE since 2007
- Worked on this photon propagation project since June 2024

universität freiburg

MANCHESTER 1824
The University of Manchester

RSE

Outline

- Optical Photon Propagation: motivation and previous work
- Mitsuba3
- Performance and scaling
- Future work: coupling Geant4 with Mitsuba3

universität freiburg

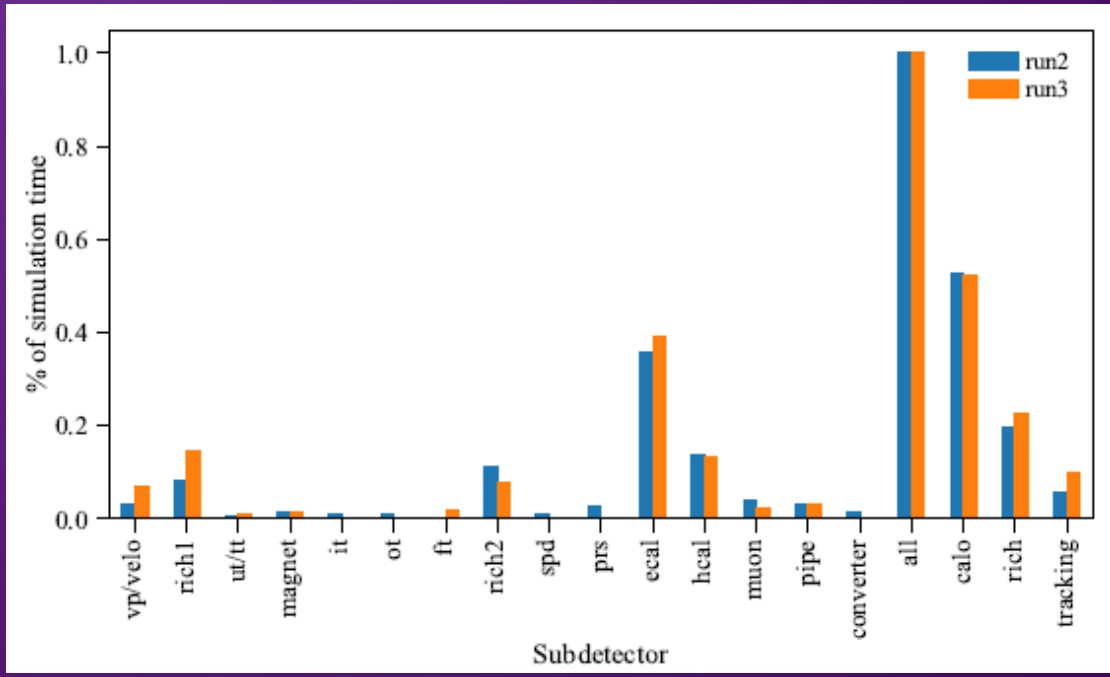
MANCHESTER 1824
The University of Manchester

RSE

Optical Photon Propagation: motivation

- In LHCb simulations in Geant4 involving RICH (Ring Imaging CHerenkov detector), photon propagation is computationally expensive
  - General problem inside and outside HEP (e.g. CGI)
  - Geant4 does not currently have GPU capabilities

- Photon propagation is computationally expensive but lends well to parallelisation as photons are independent
- Need to work with both GPU and CPU to maximise compatibility and measure performance



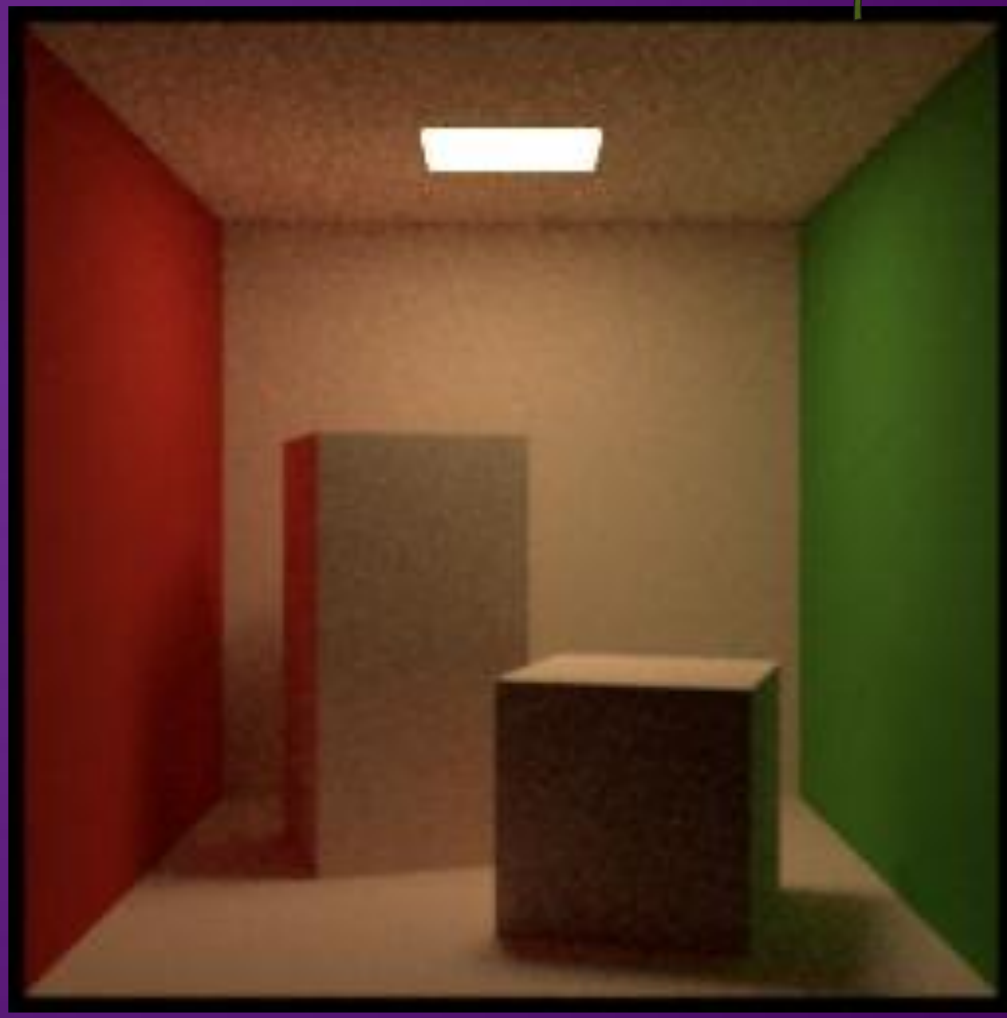
universität freiburg

MANCHESTER 1824
The University of Manchester

RSE

Mitsuba3

- Physics based renderer, represents light as rays
- Uses MonteCarlo sampling
- Widely used, open source, under active development
- Parallelised using LLVM and CUDA
  - CUDA uses OptiX
  - LLVM uses Embree
- https://mitsuba.readthedocs.io/en/stable/



https://mitsuba.readthedocs.io/en/stable/src/quickstart/mitsuba_quickstart.html

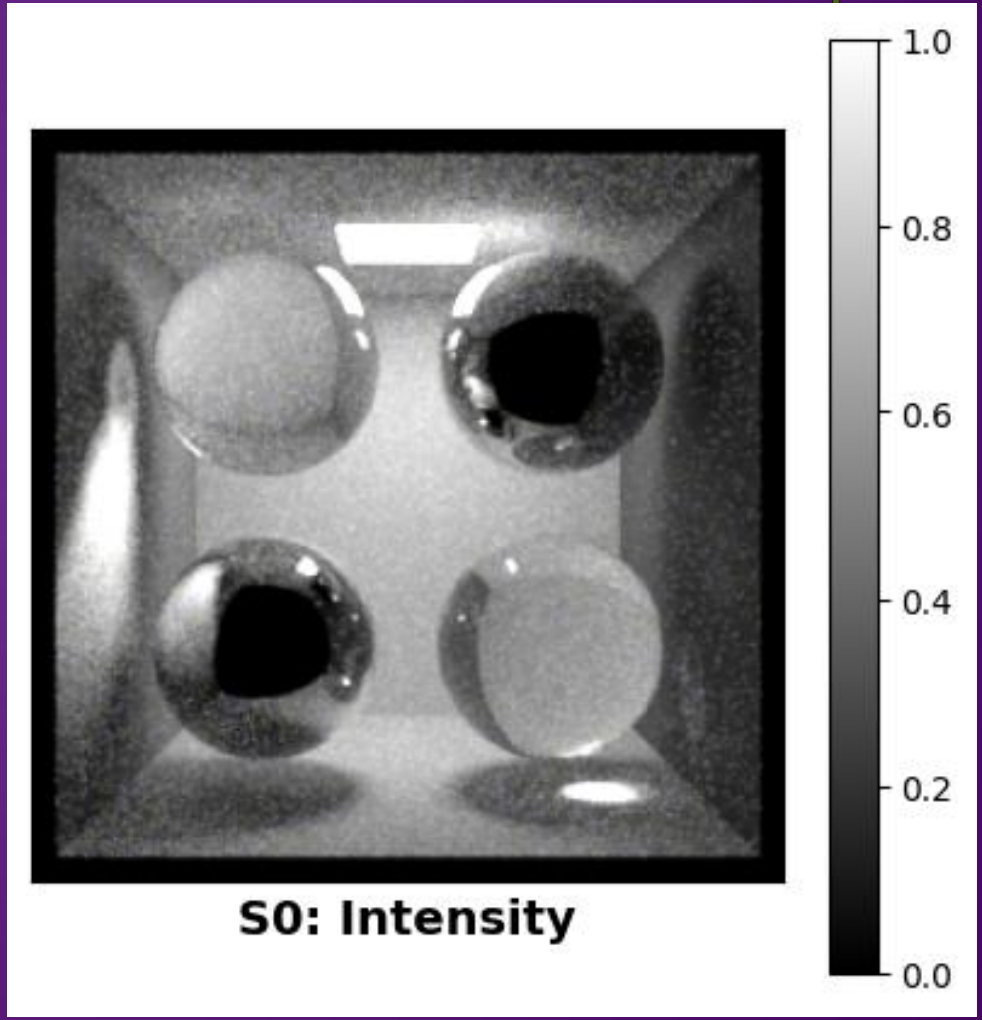
universität freiburg

MANCHESTER 1824
The University of Manchester

RSE

Mitsuba3

- Input
  - XML or Python dictionary
- Rendering
  - Parses geometry (scene)
  - Compiles optimised kernel (DrJIT)
  - CUDA + OptiX
  - LLVM + Embree
  - Emits and propagates (traces) photons (rays)
  - Constructs images applying camera effects
- Output
  - EXR image, Numpy arrays



https://mitsuba.readthedocs.io/en/stable/src/rendering/polarized_rendering.html

universität freiburg

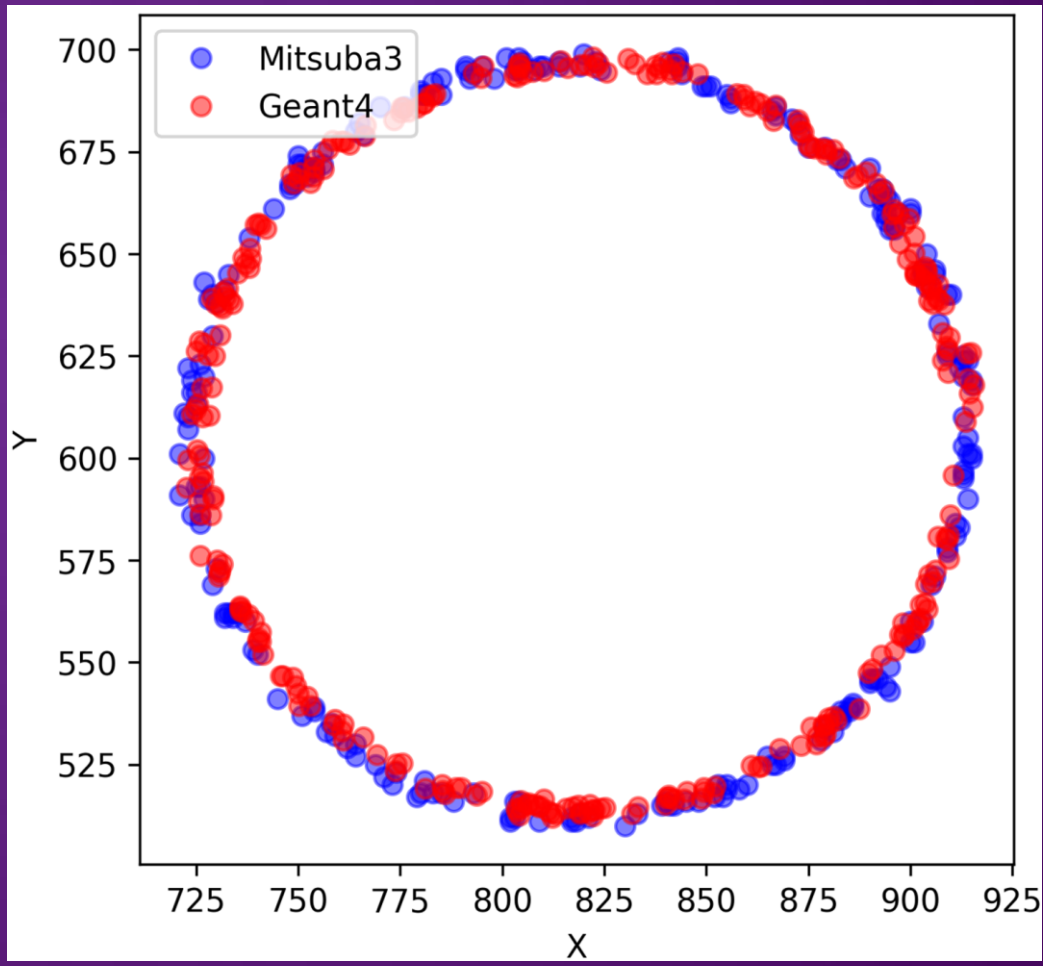
MANCHESTER 1824
The University of Manchester

RSE

Optical Photon Propagation: previous work with Mitsuba3

- Cherenkov rings match well to Geant4
- Differences are due to image processing
  - Mitsuba3 hits are recorded as pixel values → threshold
- Slight difference because Mitsuba3 hits are projections
- Thresholding
  - Mitsuba simulates ”bloom” like effects creating more hits than photons
  - Set a minimum pixel value such that:

    Hits(Mitsuba) = Hits(Geant4)



From Davis et al:
https://arxiv.org/pdf/2309.12496

universität freiburg

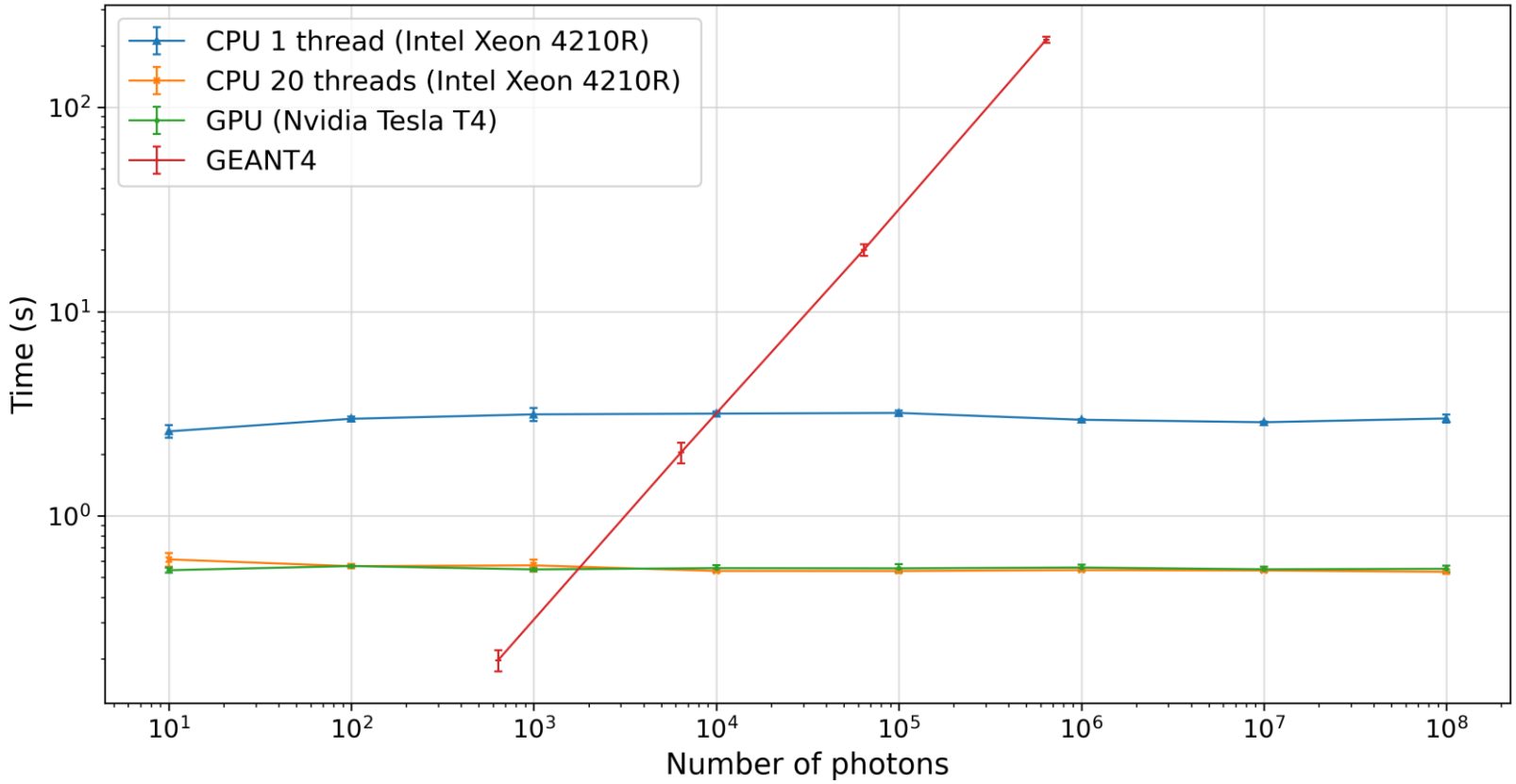
MANCHESTER 1824
The University of Manchester

RSE

Optical Photon Propagation: previous work with Mitsuba3

- Resampled Geant4 photons (~10^6) to simulate larger numbers
- Run on CPU and GPU with multiple repeats, faster than Geant4

From Davis et al: https://arxiv.org/pdf/2309.12496



universität freiburg

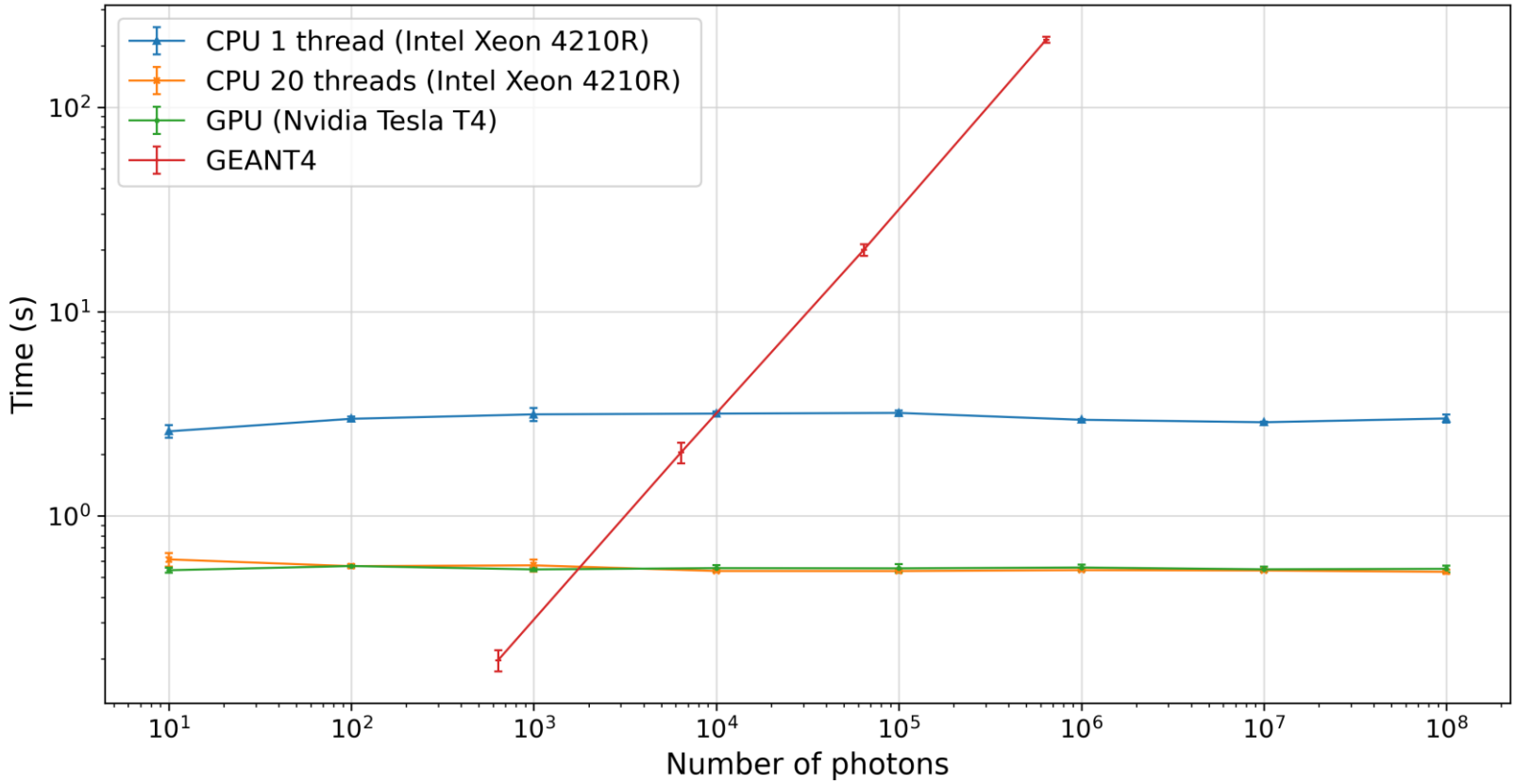
MANCHESTER 1824
The University of Manchester

RSE

Optical Photon Propagation: previous work with Mitsuba3

- Mitsuba render() time does not increase with photon count?
- Similar CPU and GPU timings due to JIT compilation overhead

From Davis et al: https://arxiv.org/pdf/2309.12496



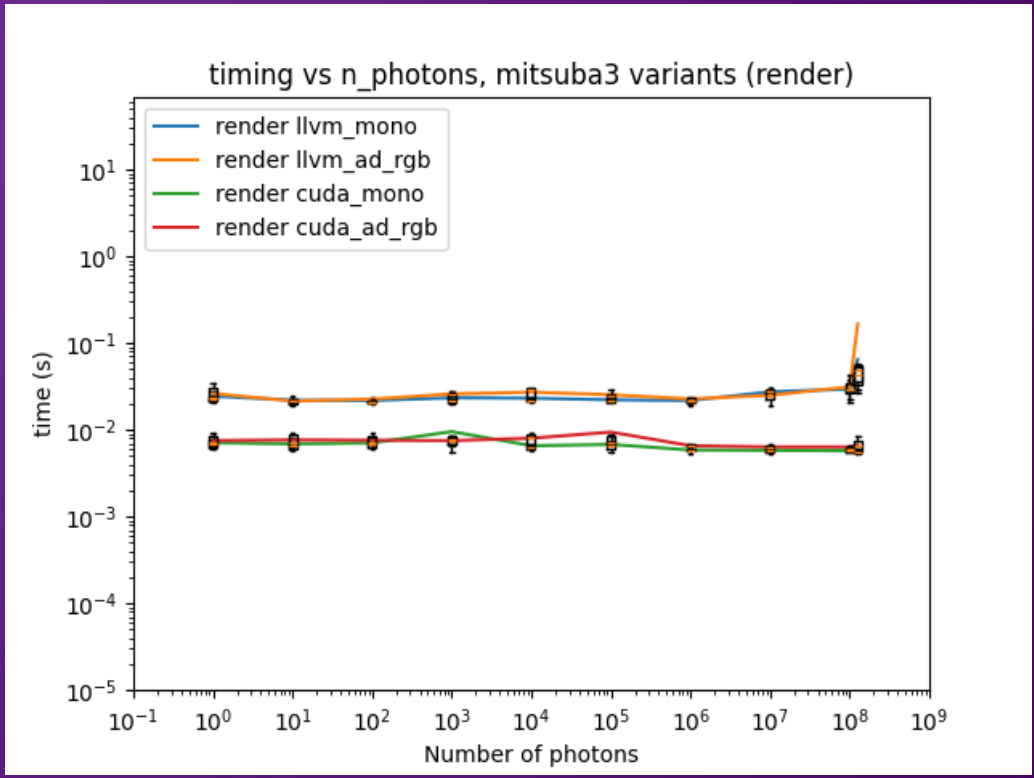
universität freiburg

MANCHESTER 1824
The University of Manchester

RSE

Mitsuba3: Performance and scaling

- More recent work to try to understand render() scaling
- Ran a similar Cherenkov-style experiment to previous work, but using Python input rather than command-line XML input (checked for no differences in code timings when doing this)
- To start with, results corroborated with previous work, with no scaling with increased numbers of photons



universität freiburg

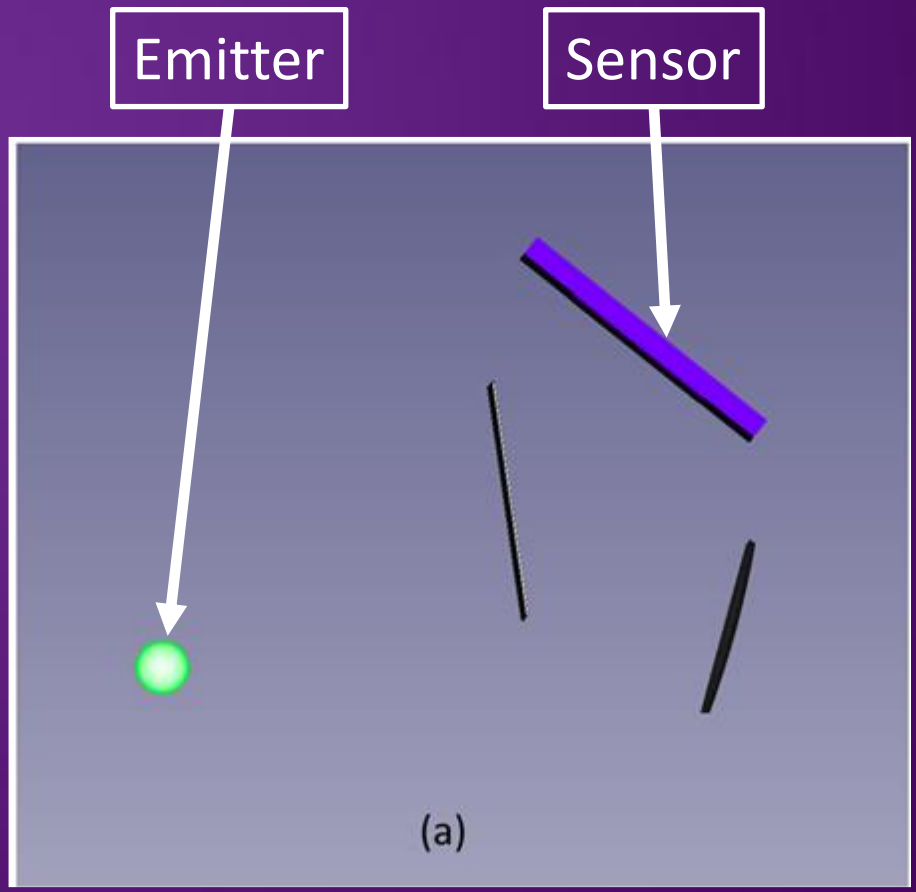
MANCHESTER 1824
The University of Manchester

RSE

Mitsuba3: Performance and scaling

- Further investigation of the code with help from Mitsuba3 dev team pinpointed why no scaling was occurring in our example

- The number of rays to be rendered depends solely on two parameters: the **size of the sensor (S_xy)** and the **sample_count (SC)** specified in the sensor (both these parameters can be set by the user)
- This will uniformly sample **S_xy * SC** photons from our list of photons, which is constant (so for large numbers of photons, only a fraction of them were sampled)



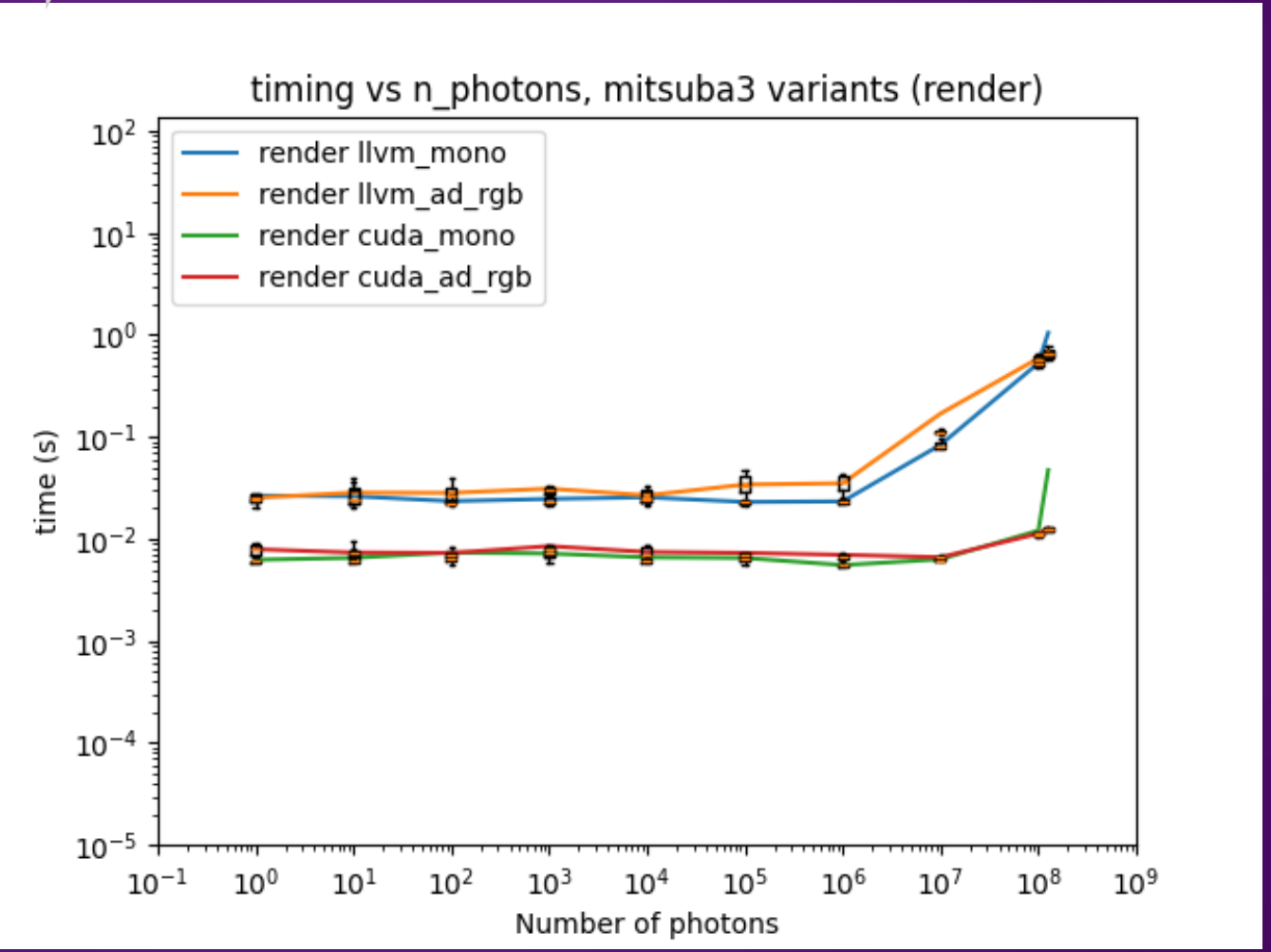
universität freiburg

MANCHESTER 1824
The University of Manchester

RSE

Mitsuba3: Performance and scaling

- Re-run the experiments with **sample count** set so that **all photons** are sampled
- Sensor size is 1024^2
- New results show scaling (n_photons > sensor size)



- At small sizes computation time dominated by JIT compilation

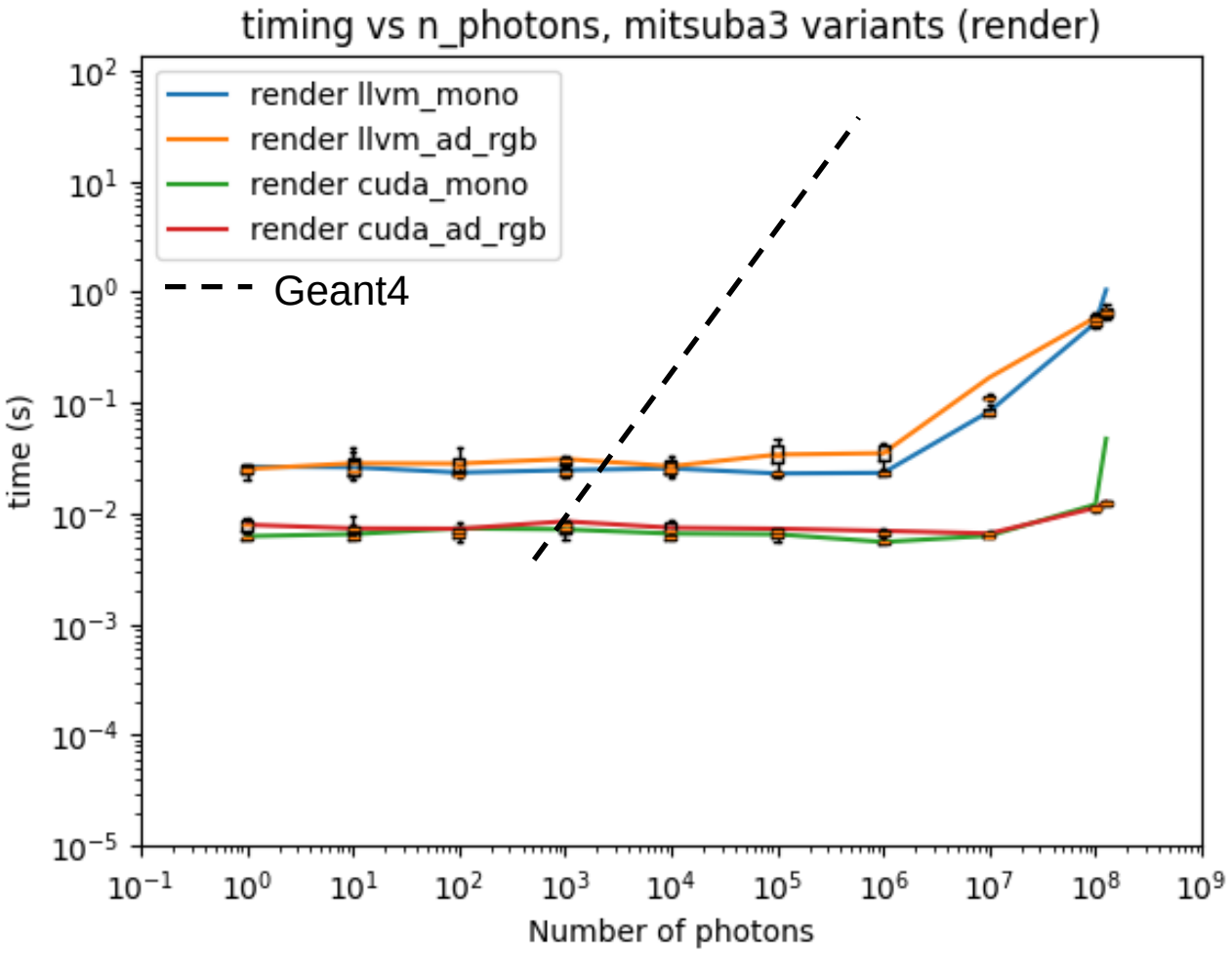
universität freiburg

MANCHESTER 1824
The University of Manchester

RSE

Mitsuba3: Performance and scaling

- Scaling for **CPU (LLVM)** variants with 16 threads when number of photons larger than sensor size
- Slight scaling on **GPU (CUDA)** too though less clear



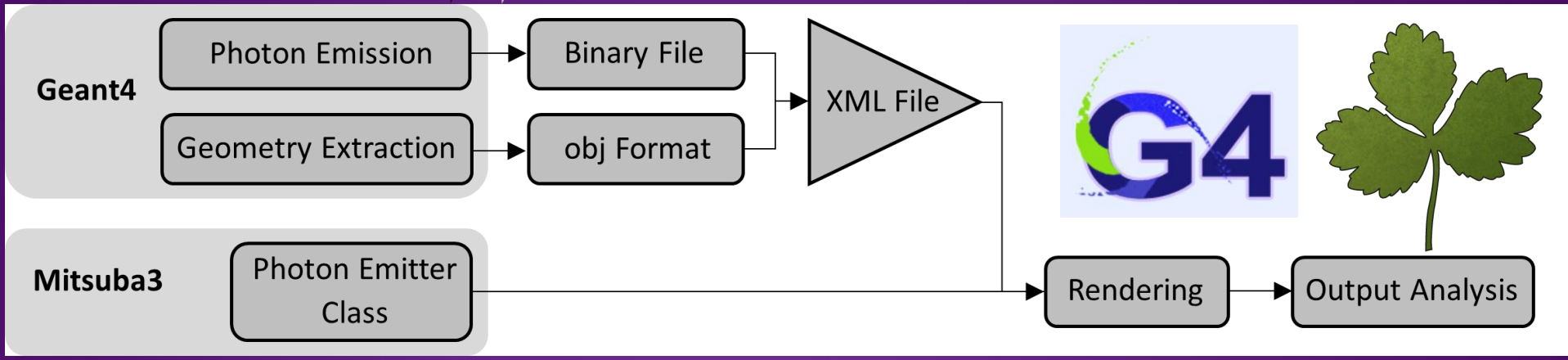
- Future: Run previous (XML) experiment with adjusted sample count?

universität freiburg

MANCHESTER 1824
The University of Manchester

RSE

Future work: coupling Geant4 with Mitsuba3



- Pipeline from Geant4 to Mitsuba3 seems reasonably clear from previous work
- Will both need to run simultaneously together, or can Mitsuba3 run asynchronously?
- Future work will also look at the pipeline from Mitsuba3 back into Geant4
- The result will be some form of application or workflow

universität freiburg

MANCHESTER 1824
The University of Manchester

RSE

Conclusion

Work on Mitsuba3:

- https://github.com/UoMResearchIT/mitsuba3-manchester
- If you're interested in collaboration then please ask!

Thanks to:

- Keith Evans
- Nicolas Roussel
- Wenzel Jakob

Any questions?

universität freiburg

MANCHESTER 1824
The University of Manchester

RSE

What is an RSE?

- Basically, creating software for researchers to use; effectively what I've been doing for 17 years

- The name was first developed in the early 2010s as software developers at universities considered a way to create career pathways for those working in research but not interested in a traditional academic career

- See https://society-rse.org/about/history/

- I first became aware of it in 2017 when I was asked at the start of a new position which job title I wanted: "Research Associate" or "Research Software Engineer"

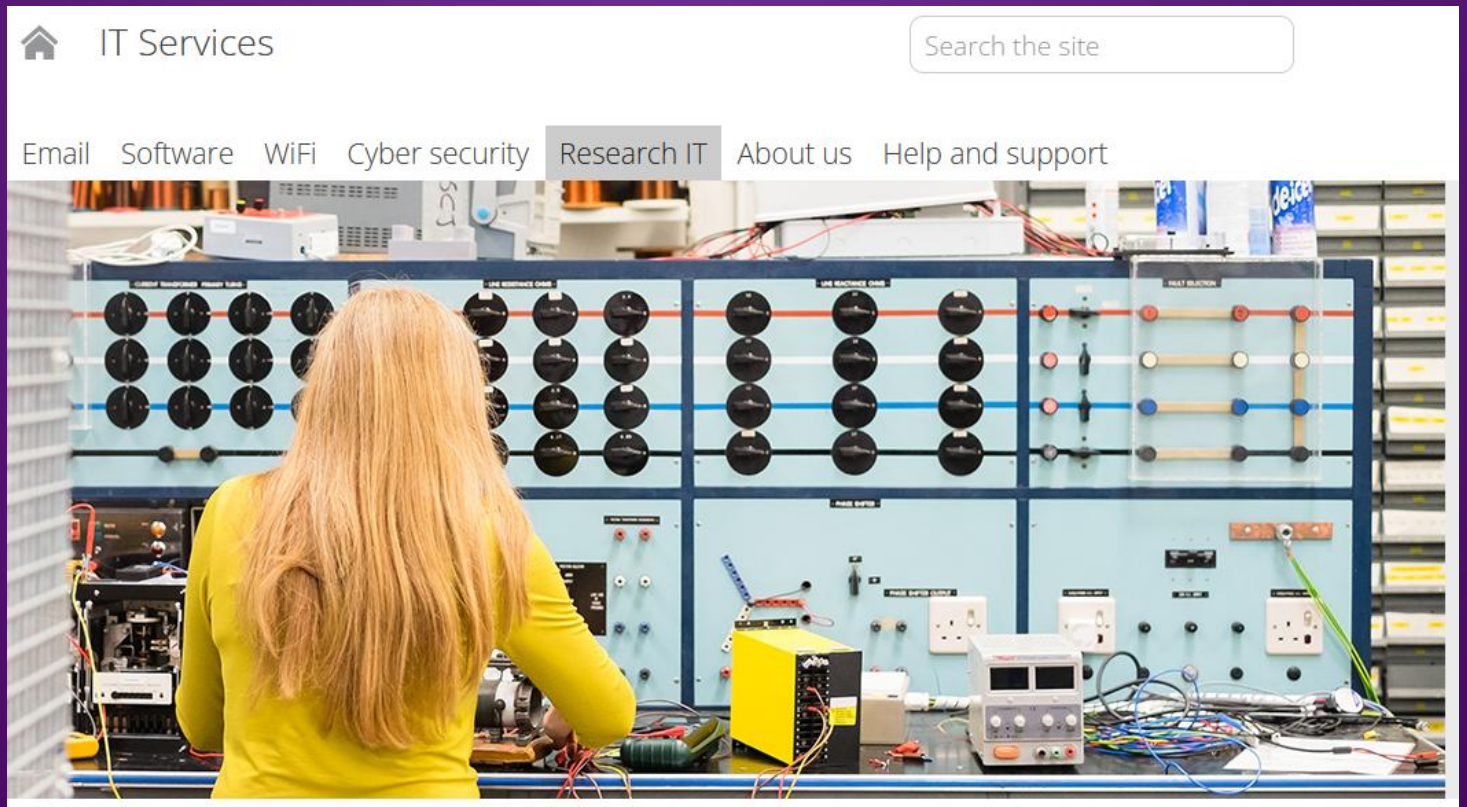
universität freiburg

MANCHESTER 1824
The University of Manchester

RSE

RSE team in Manchester

- One of the largest RSE teams in the UK
- Around 50 employees working across about 60-70 projects (oversubscribed)
- Opportunities for career growth, development & management



universität freiburg

MANCHESTER 1824
The University of Manchester

RSE

RSE team in Manchester

- Projects can take place anywhere where software is required to perform research
  - Across all university faculties and most university departments
  - Mainly across scientific/medical research, but not limited to that
- We also take a lead on teaching courses relating to software for research (programming languages, software lifecycles)

- Part of the wider Research IT community which also includes infrastructure teams, teams which support teaching, and teams which manage platforms and resources

- There are RSE teams across the country at different universities / research centres who all work in different ways