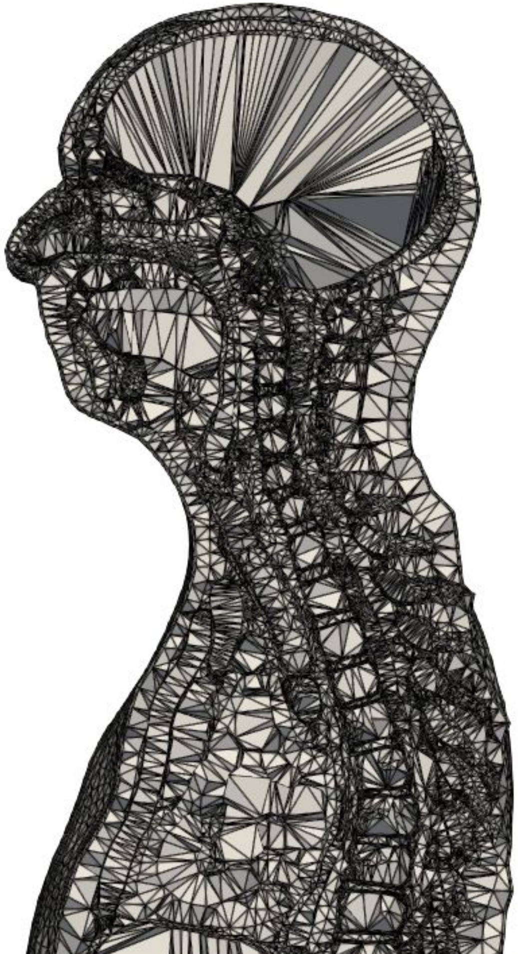
EGS_Mesh: accurate radiation transport simulations in CAD meshes with EGSnrc

Max Orok, University of Ottawa

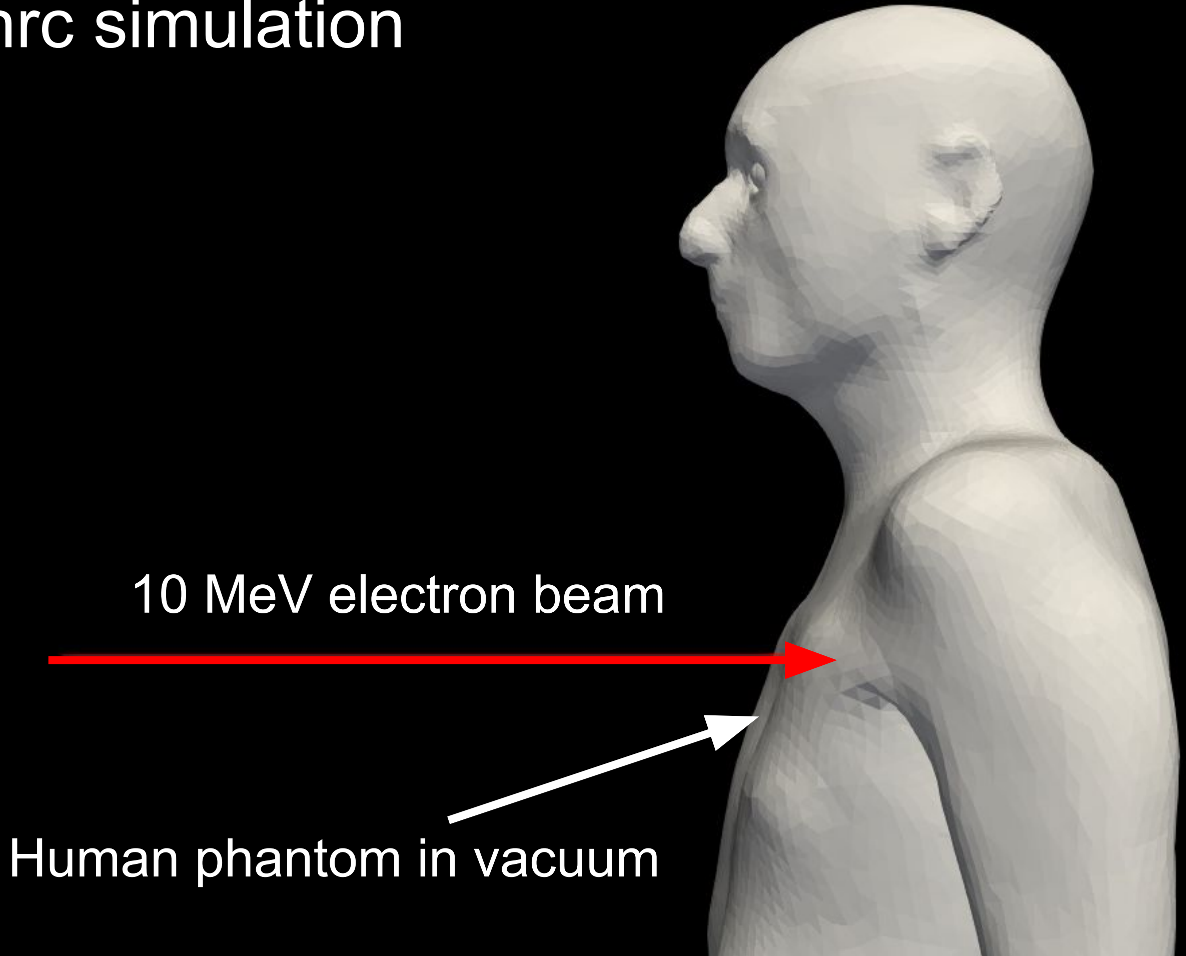
CIRMS 2022
April 12th, 2022



Presentation overview

1. Introduce the radiation transport code EGSnrc
2. Present EGS_Mesh, a tetrahedral mesh geometry for EGSnrc
3. Preliminary verification results:
   - Compared to other EGSnrc geometries,
   - Compared to other transport codes that can simulate tetrahedral meshes,
   - **Not part of this presentation**:
     - Theoretical tests (EGS_Mesh has passed so far),
     - Simulation performance (preliminary results show EGS_Mesh competitive with other codes).

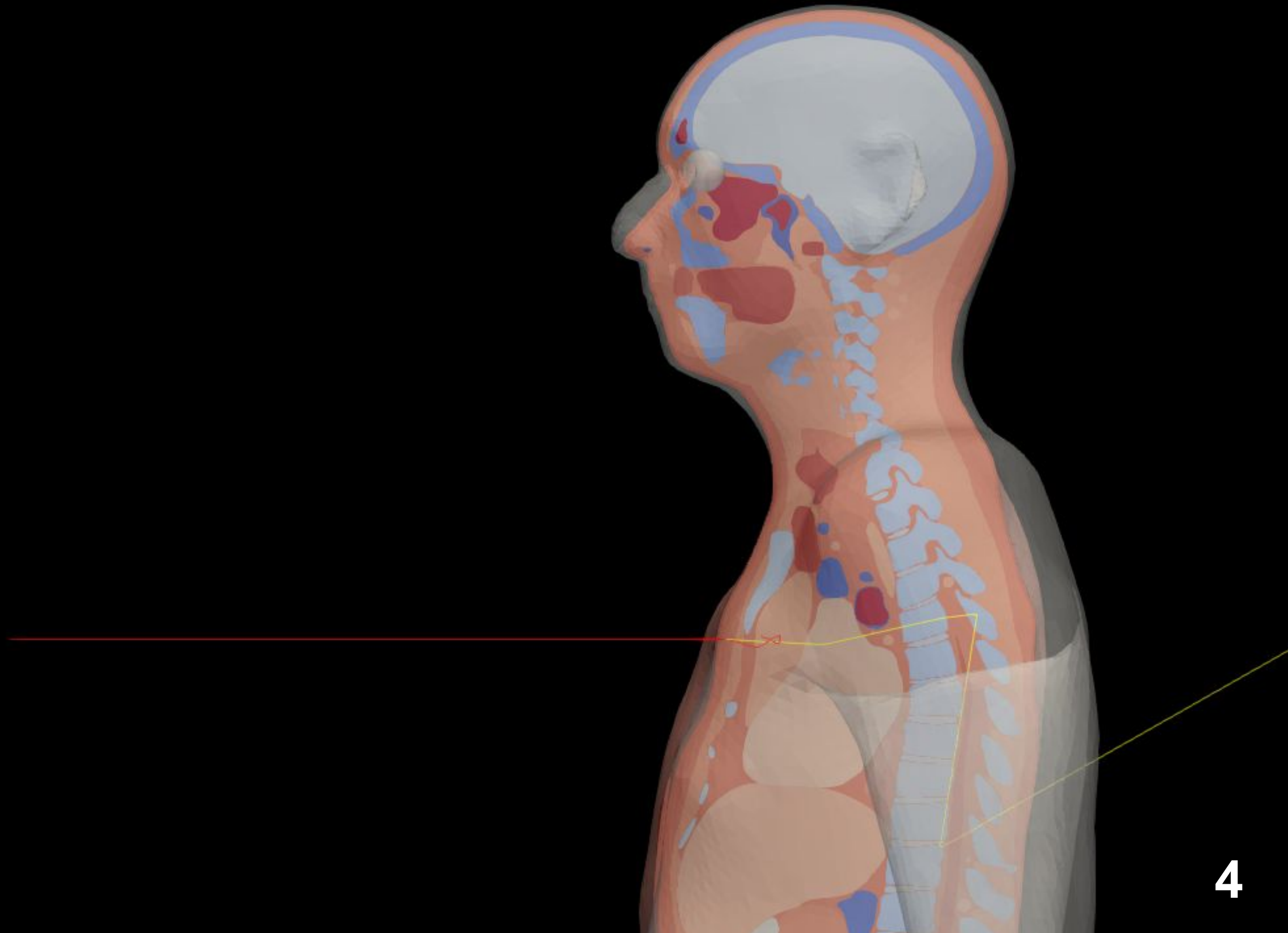
Example EGSnrc simulation



Electrons

Photons

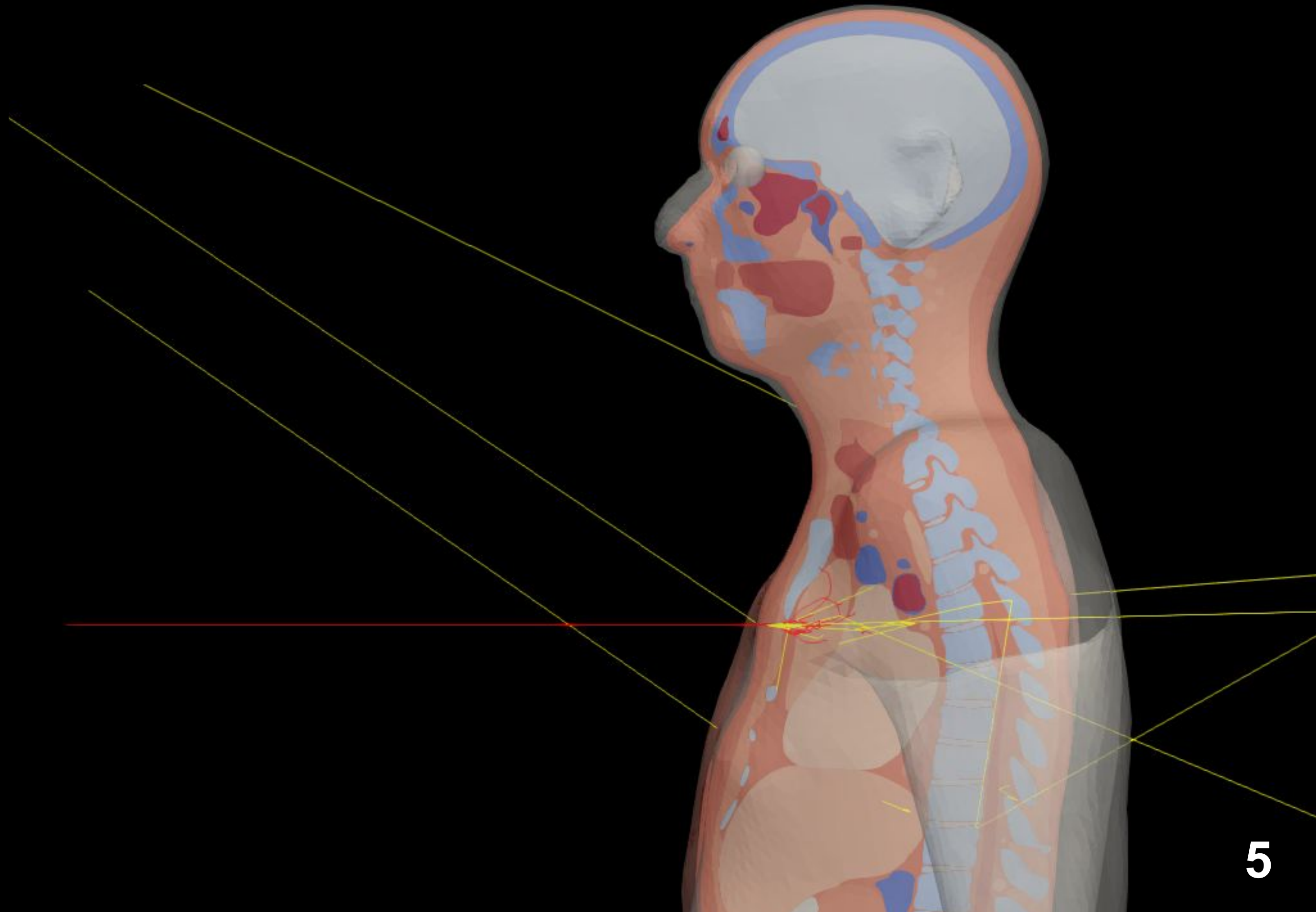
1 particle…



Electrons

Photons

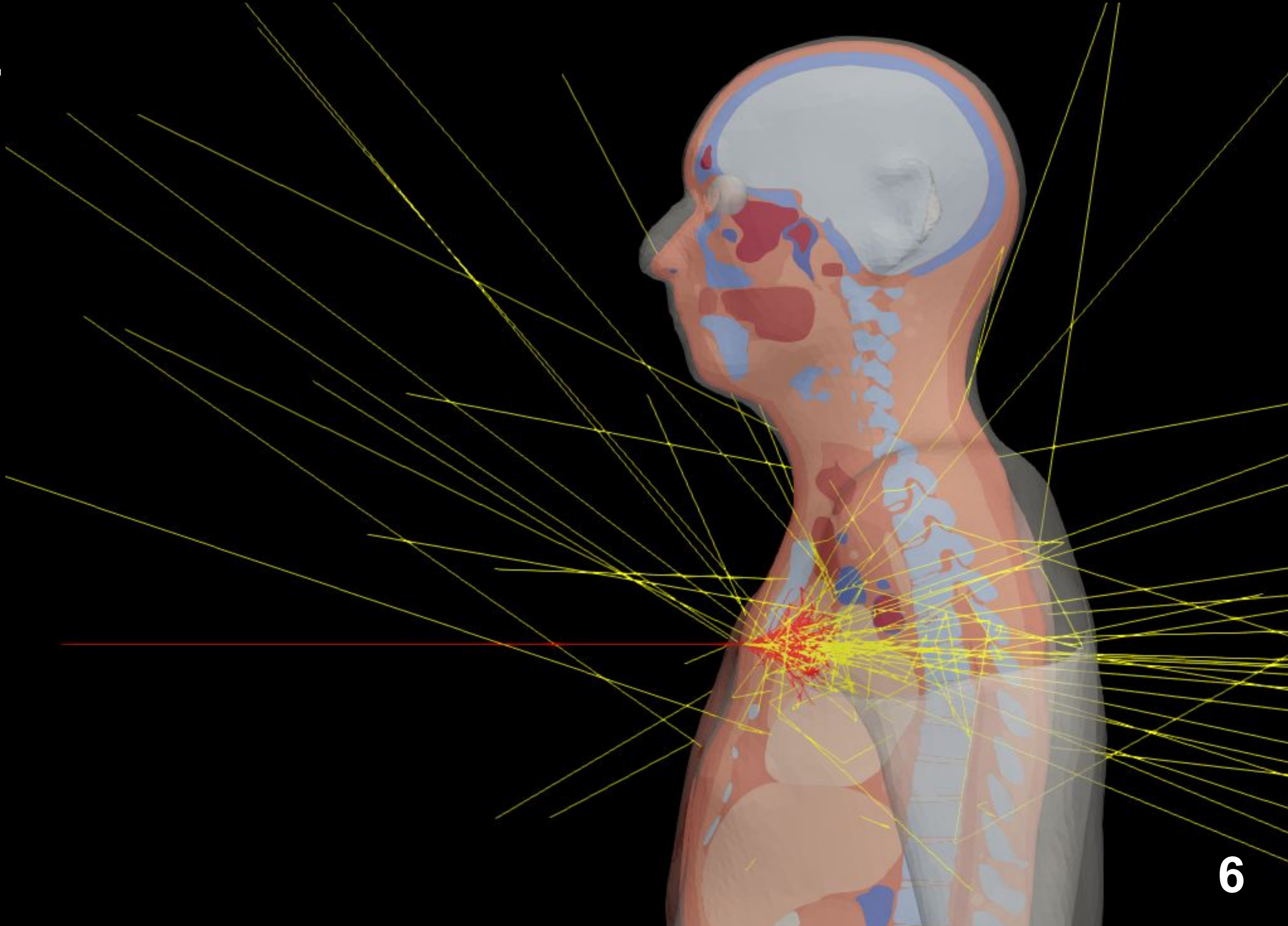
10 particles…



Electrons

Photons

100 particles…



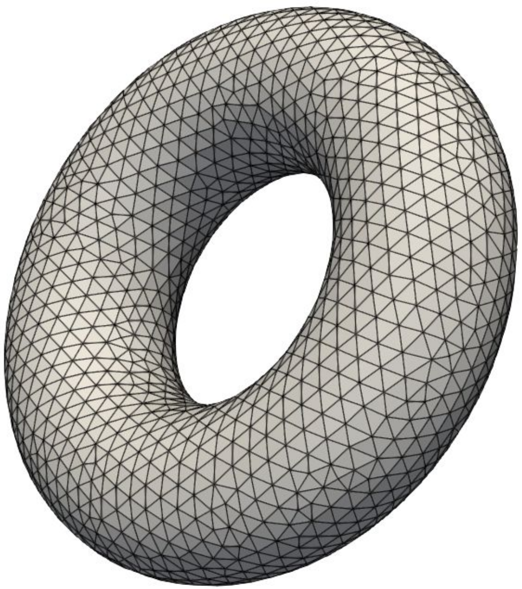
Electrons

Photons

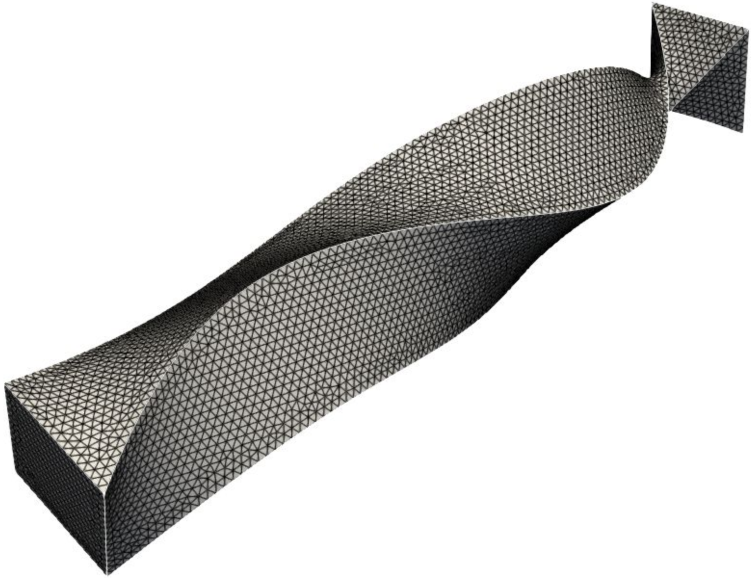
EGS_Mesh was built to simulate CAD using EGSnrc

- From 2017 to 2019, Mevex, a sterilization equipment manufacturer, created a tetrahedral mesh geometry library for EGSnrc and donated it to the Canadian National Research Council.
- The work presented here, EGS_Mesh, is a from-scratch rewrite of that original mesh library with substantial performance improvements.
- Other codes that can simulate tetrahedral mesh phantoms include Geant4, MCNP6, and PHITS.

Tetrahedral meshes: one geometry to model them all



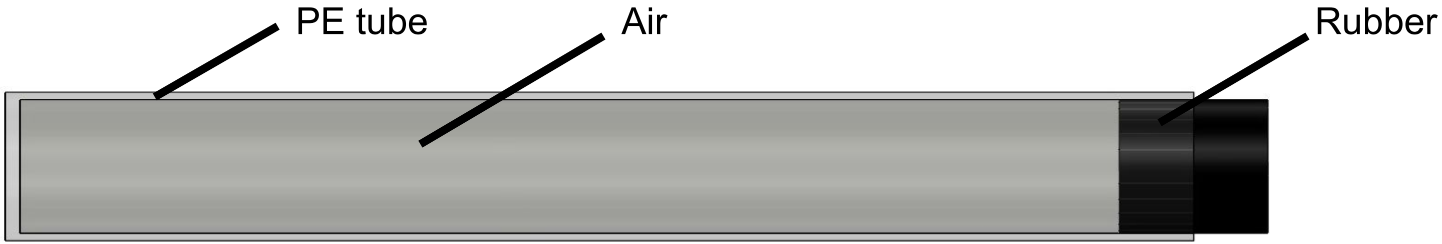
Meshed torus

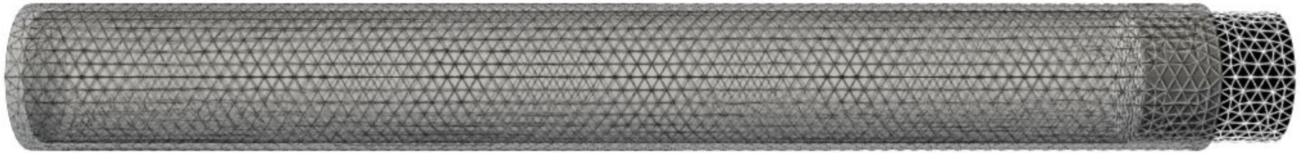


Meshed loft geometry

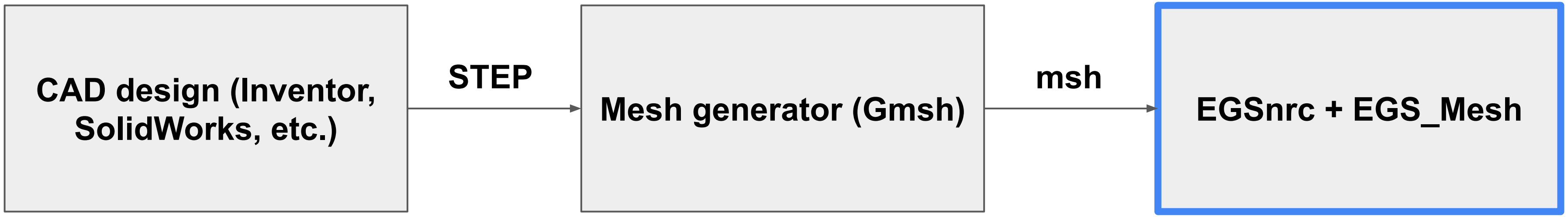
Mesh use case: accurate dose uniformity ratios

- Sample use case: calculate dose uniformity ratios for sterilization
- Example: test tube CAD model and mesh



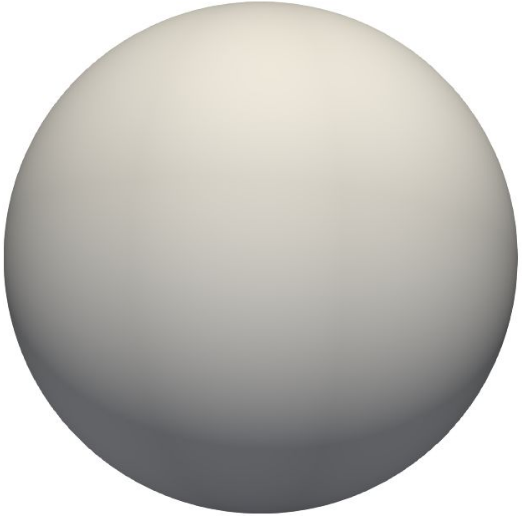


Tetrahedral mesh simulation process



- Performance is usually 2-3 times slower than a rectilinear voxel grid with the same number of elements. But, using tetrahedrons means complex phantoms have fewer elements.
- **At the same time**, the modelling community is moving towards tetrahedral meshes for complex phantoms.

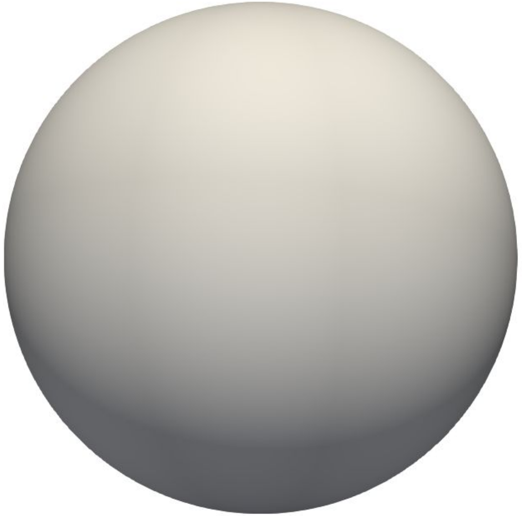
Generations of simulation phantoms: modelling a head



“Stylized” phantoms: simple shapes

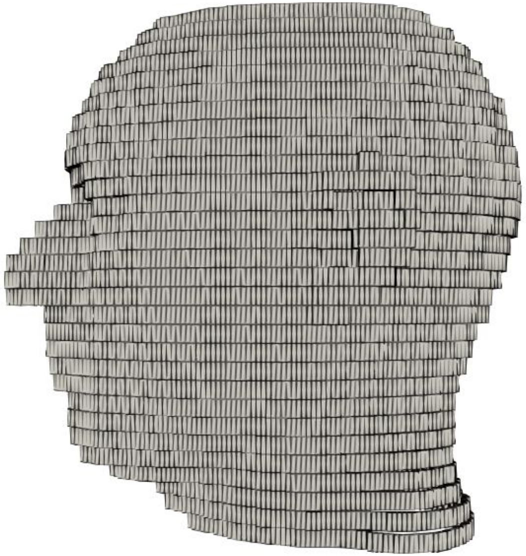
~1960

Generations of simulation phantoms: modelling a head



“Stylized” phantoms: simple shapes

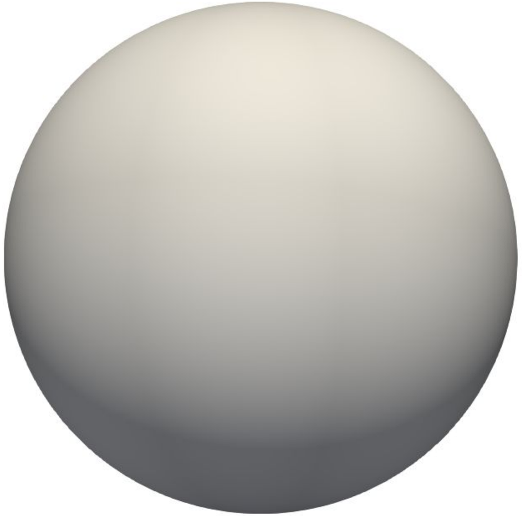
~1960



“Voxel” phantoms: hexahedrons with the same resolution

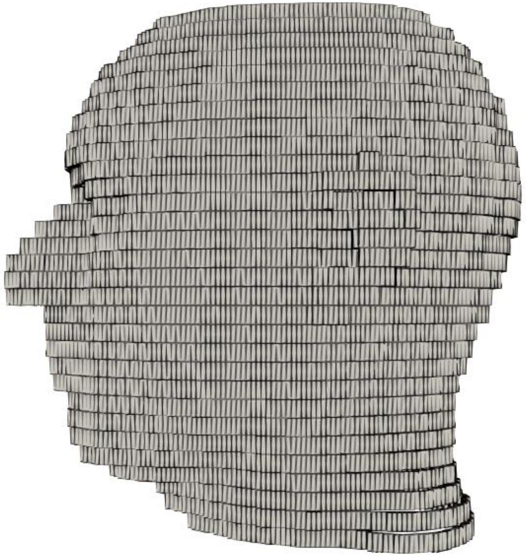
~1980

Generations of simulation phantoms: modelling a head



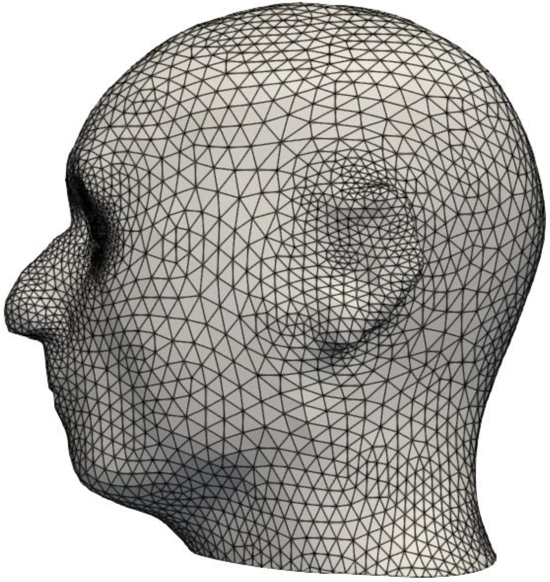
"Stylized" phantoms: simple shapes

~1960



"Voxel" phantoms: hexahedrons with the same resolution

~1980



"Mesh" phantoms: tetrahedrons with varying resolution

~2010

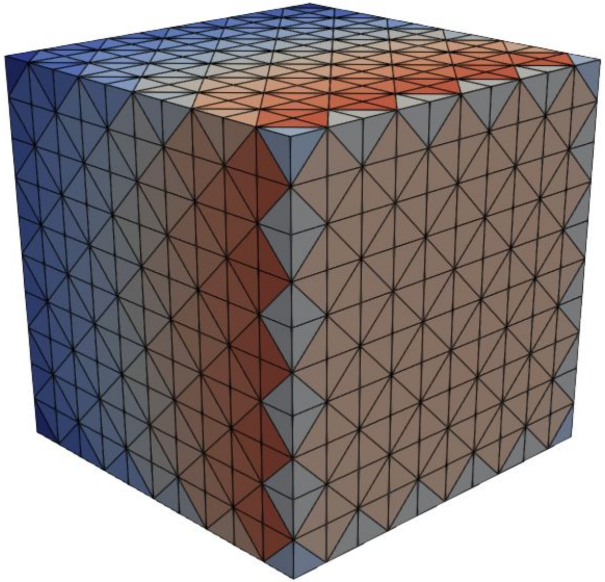
Tetrahedral mesh geometry summary

- Similar to voxel models but without uniform resolution limitations.
- Can model CAD geometries using a mesh representation.
- Simulation community is moving towards tetrahedral mesh phantoms because of their increased modelling power.

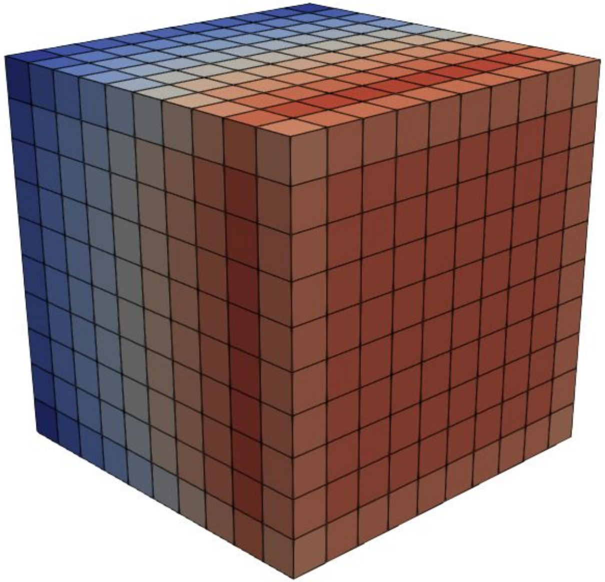
Verification work

1. Comparison to EGSnrc voxel geometry results
2. Comparison to results from other Monte Carlo codes
    - In 2020, the International Commission on Radiological Protection released two adult tetrahedral mesh reference phantoms (ICRP publication 145). Researchers have used other codes to simulate these phantoms.

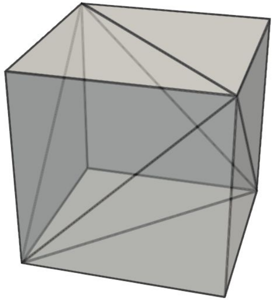
Comparison against EGS_XYZ: split voxels into tetrahedrons



EGS_Mesh

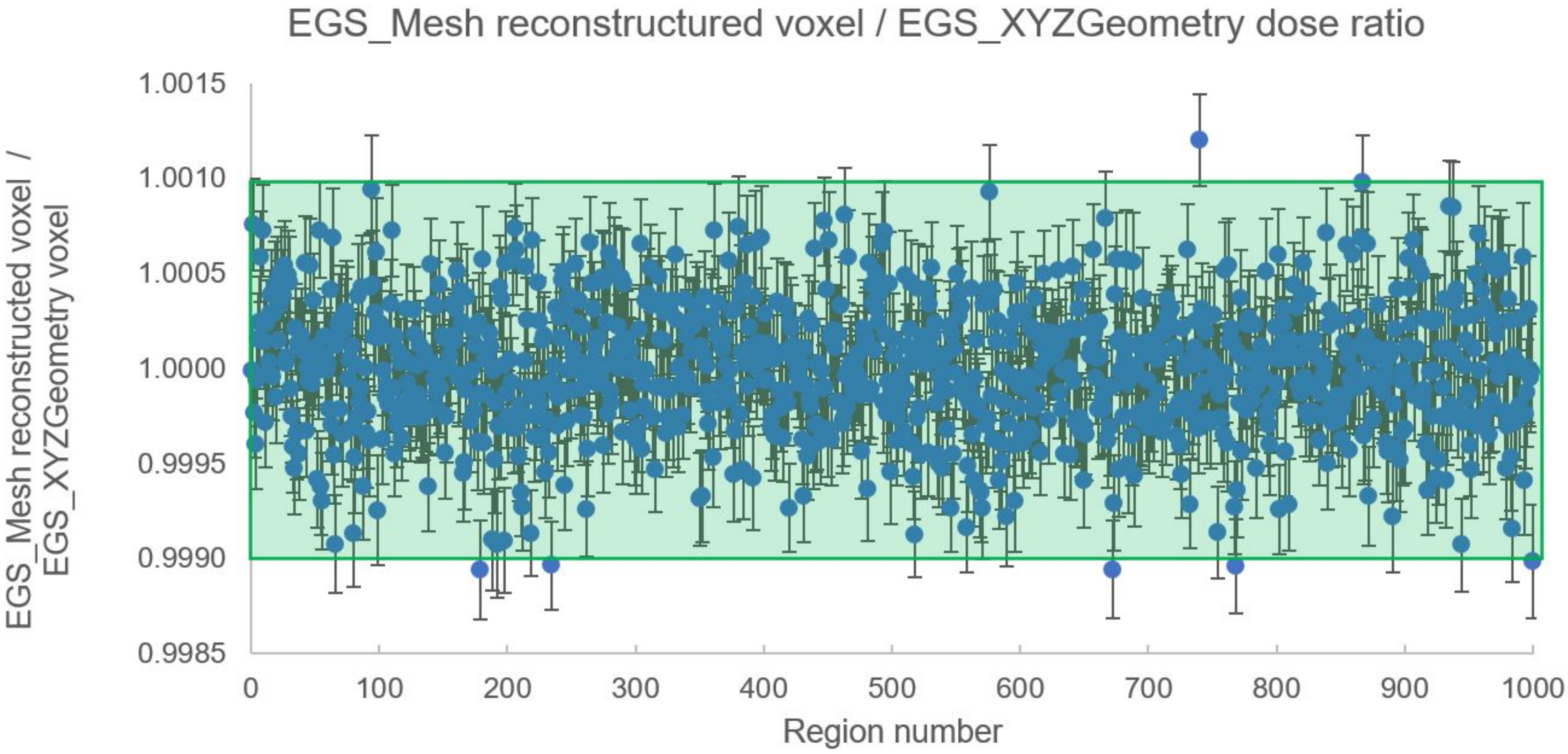


EGS_XYZGeometry



Absorbed dose, 1000cm^3 cube of water, broad parallel 1MeV photon beam

Combined tetrahedron dose results compared to EGS_XYZ

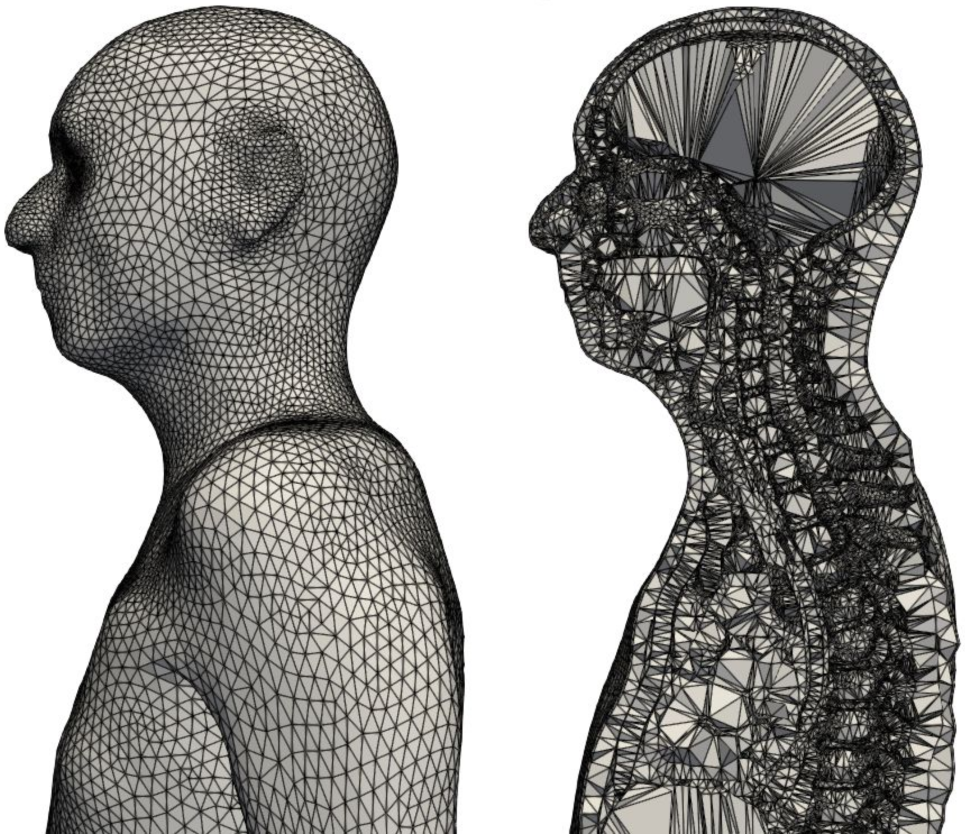


ICRP 145 reference adult mesh phantoms (2020)

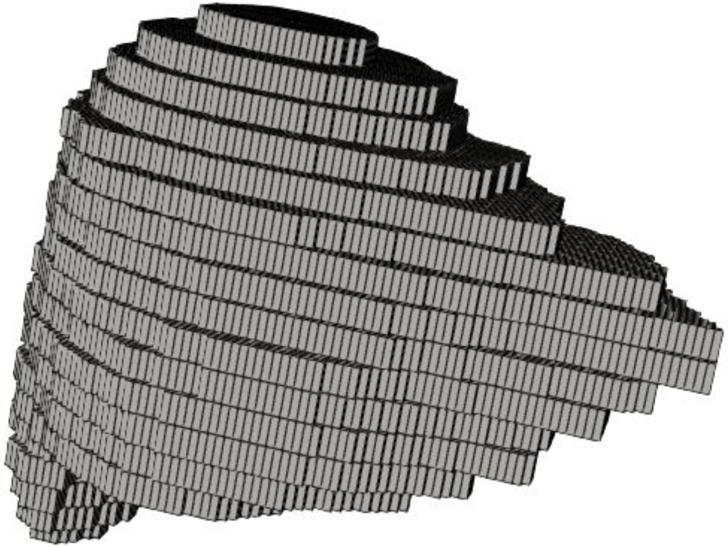
- > 8 million tetrahedrons
- ~190 organs
- Thin tissues ~µm

- Next generation after ICRP 110 voxel phantoms (2009)

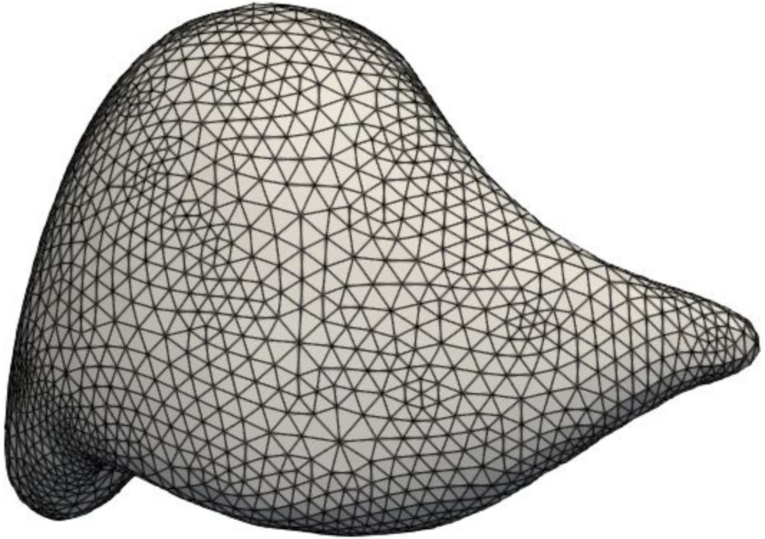
- Some of the most complex meshes ever made



Voxel vs mesh reference phantom liver

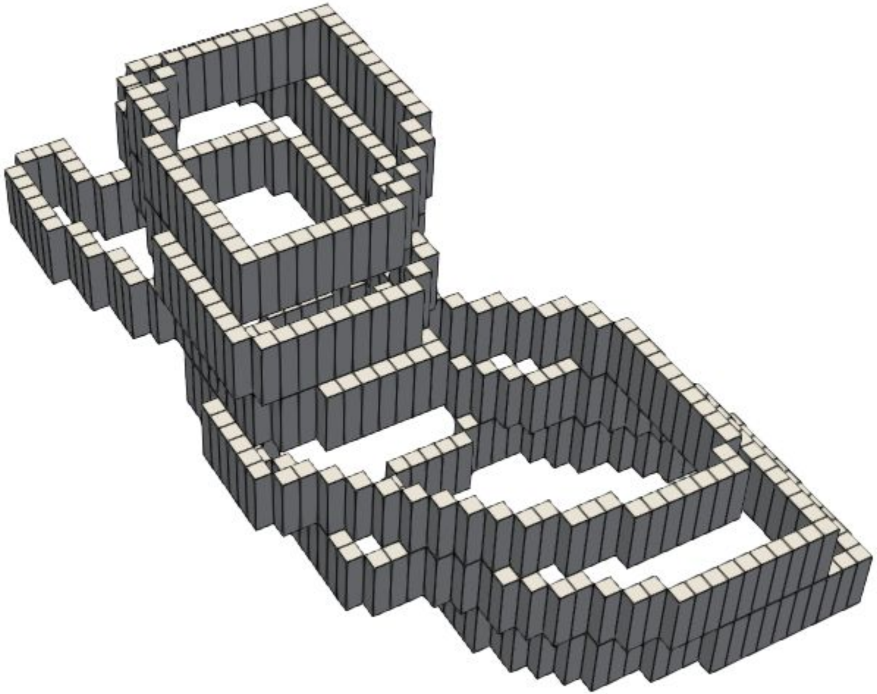


ICRP 110

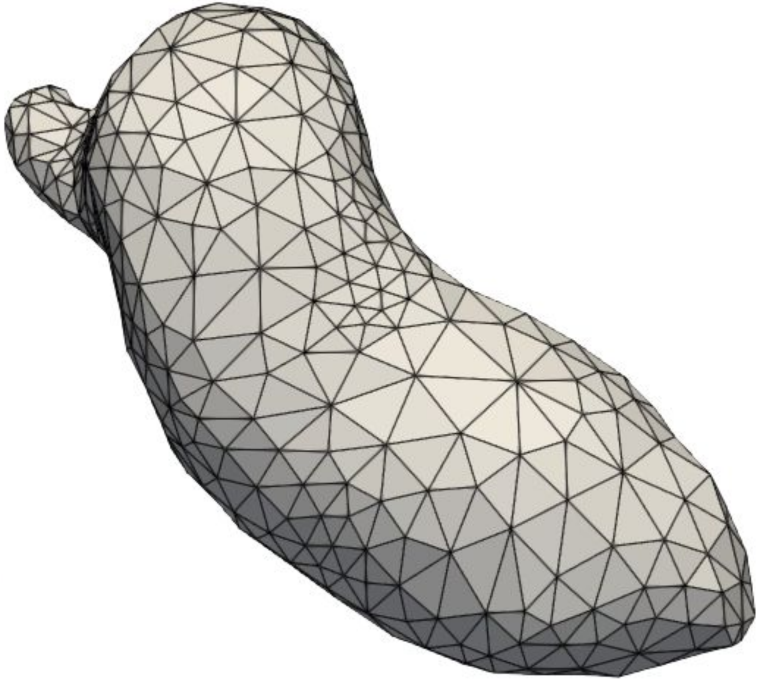


ICRP 145

Voxel vs mesh reference phantom gallbladder wall



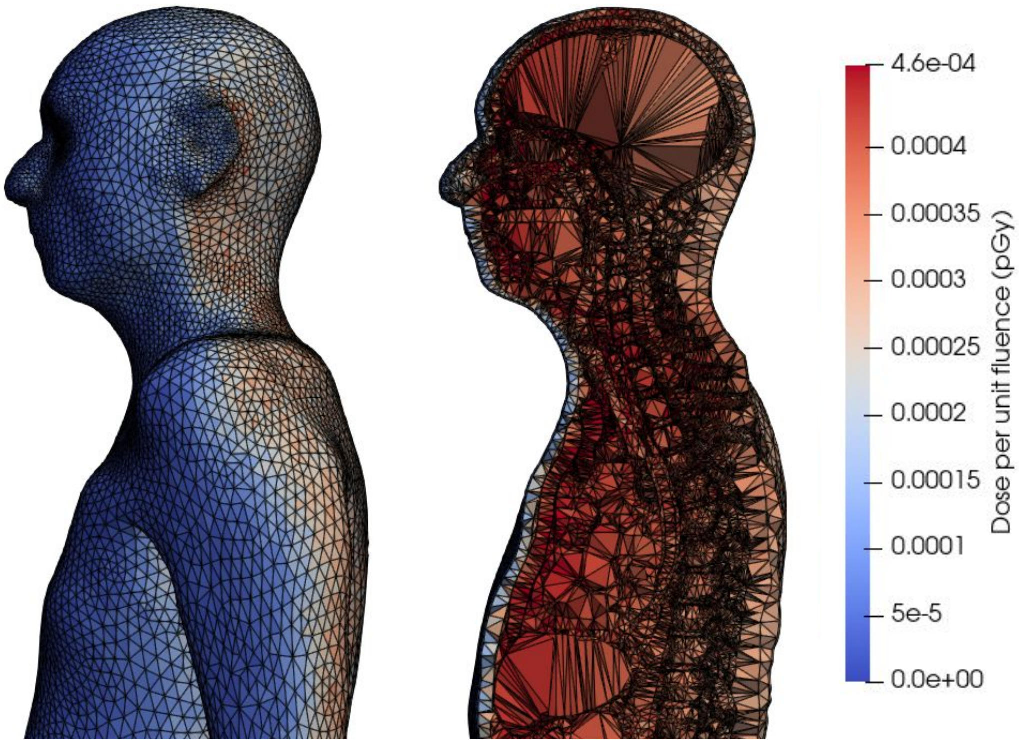
ICRP 110



ICRP 145

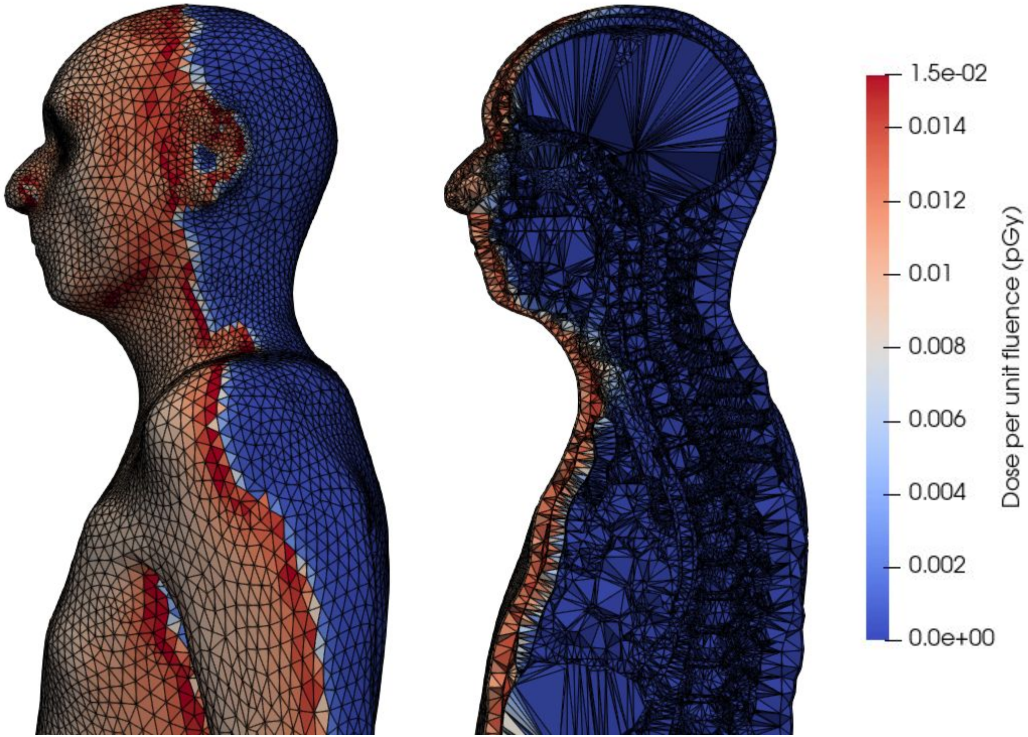
5 MeV photon broad parallel beam incident to the front of the phantom in vacuum

- Results for 1 billion histories
- ~33 cpu hours
- ~2GB memory

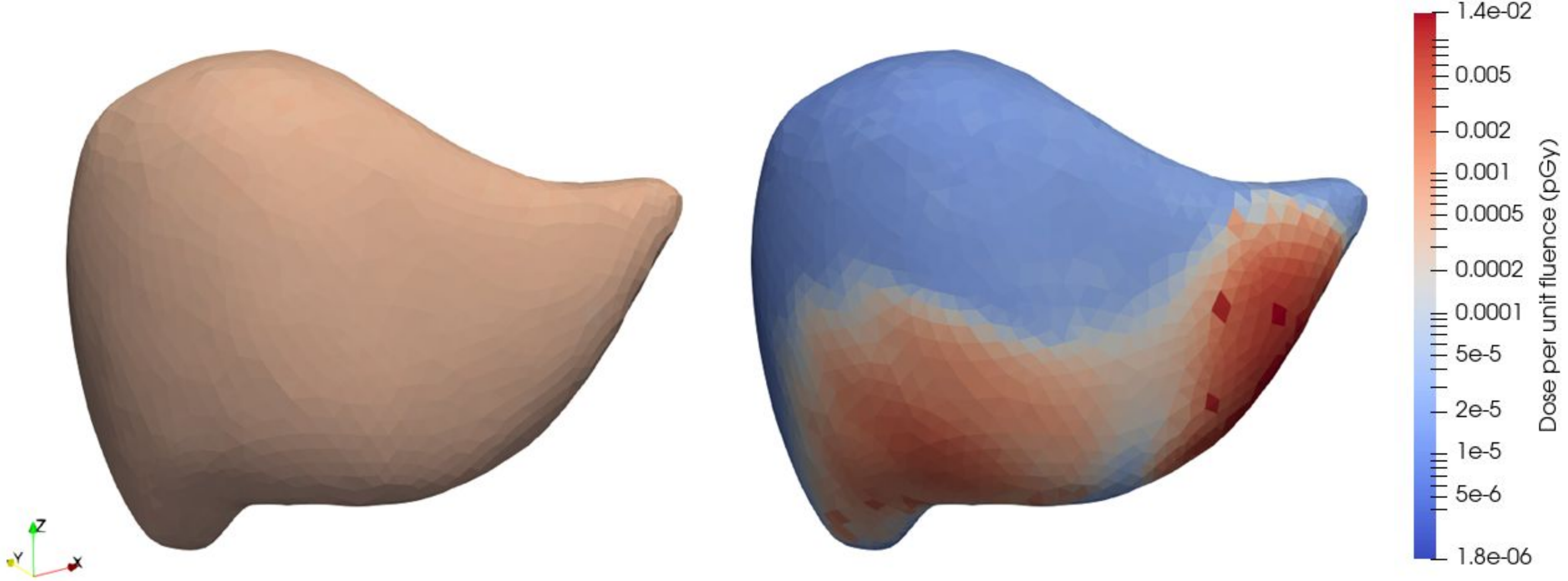


5 MeV electron broad parallel beam incident to the front of the phantom in vacuum

- Results for 1 billion histories
- ~73 cpu hours
- ~2GB memory



5 MeV photon and electron liver dose

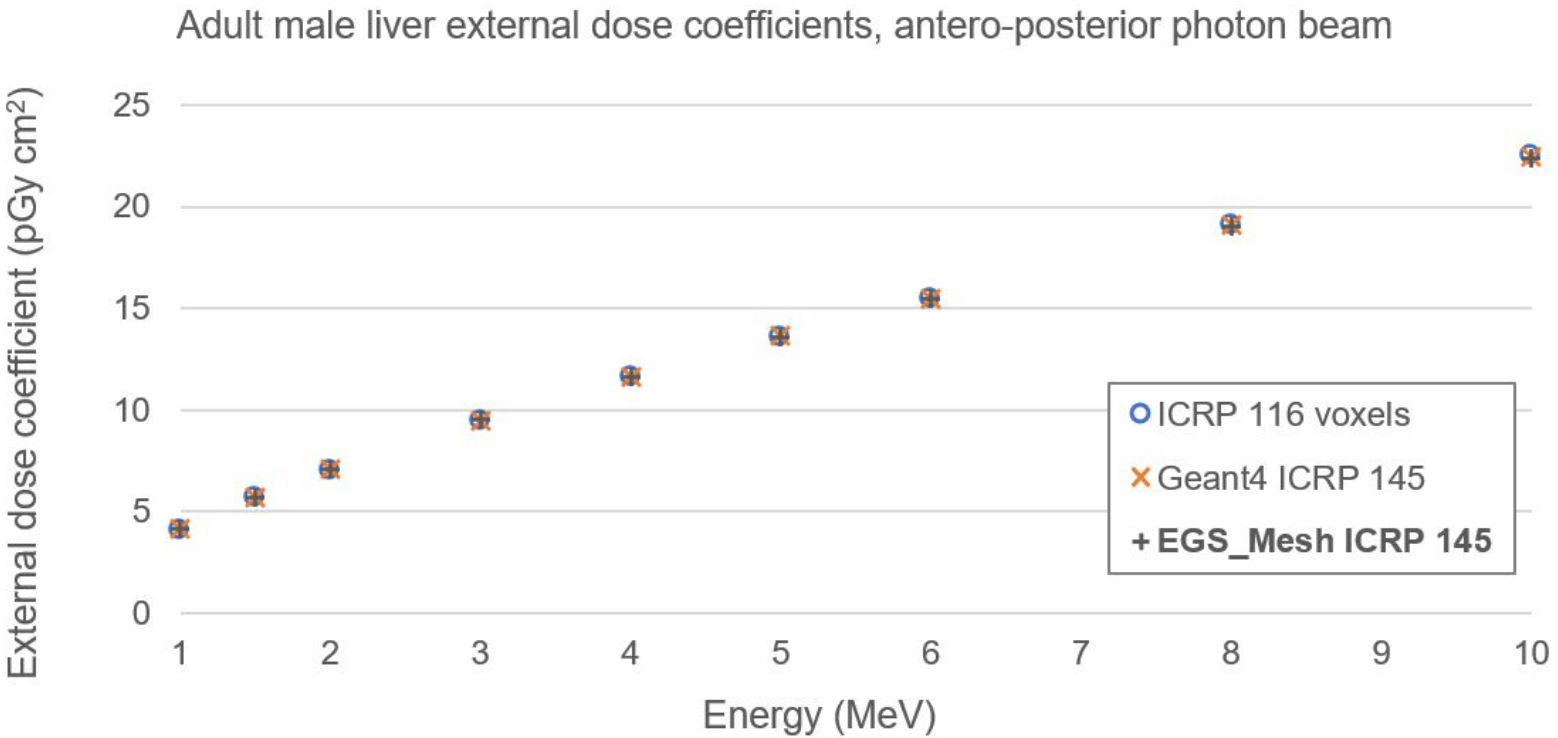


Photons

Electrons

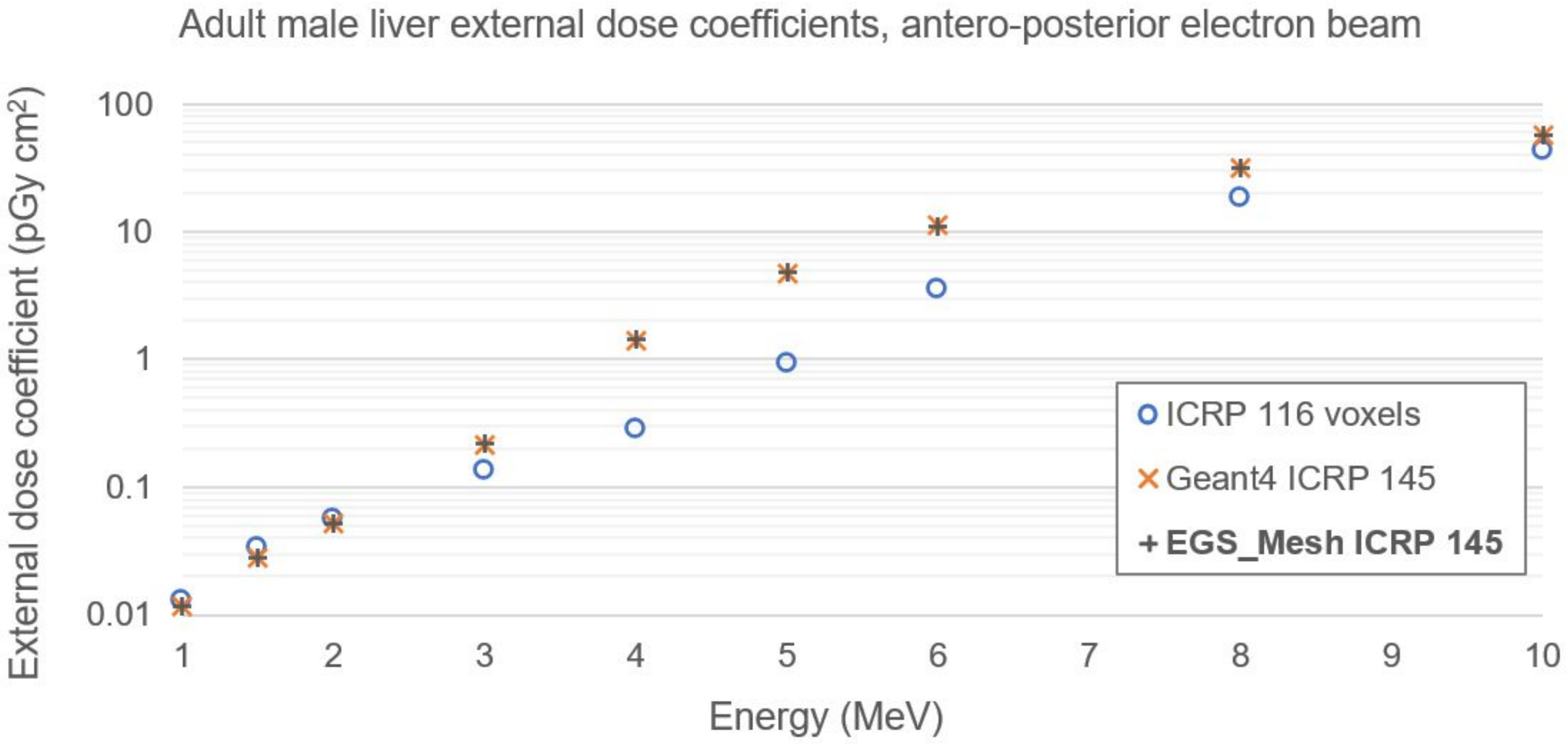
Results for a broad parallel beam along +Y axis. Graph uses a log scale.

Comparison to ICRP 116 organ doses: liver, photons



All uncertainties under 1%. Geant4 results from Yeom et al. 2019. “Dose coefficients of mesh-type ICRP reference computational phantoms for idealized external exposures of photons and electrons”.

Comparison to ICRP 116: liver, electrons



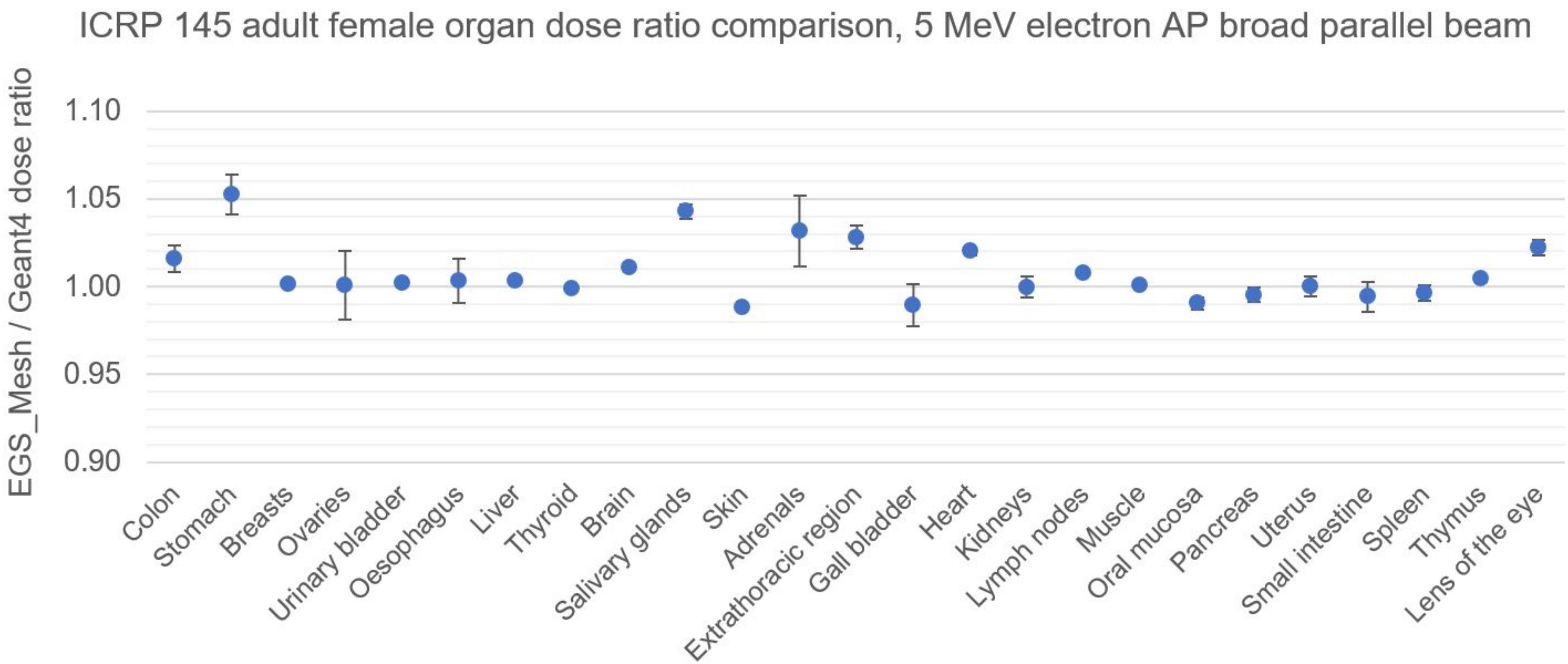
*All uncertainties less than 1%.

ICRP 145 mesh vs voxel phantom summary

- The ICRP 145 report (and these results) show mesh organ doses are close to the voxel results for penetrating radiation such as photons[1].
- But for weakly penetrating radiation like electrons, the mesh phantoms offer more realistic results[1].
- Report: “***MRCPs [mesh phantoms] will be used in all other future calculations****….*”[2]

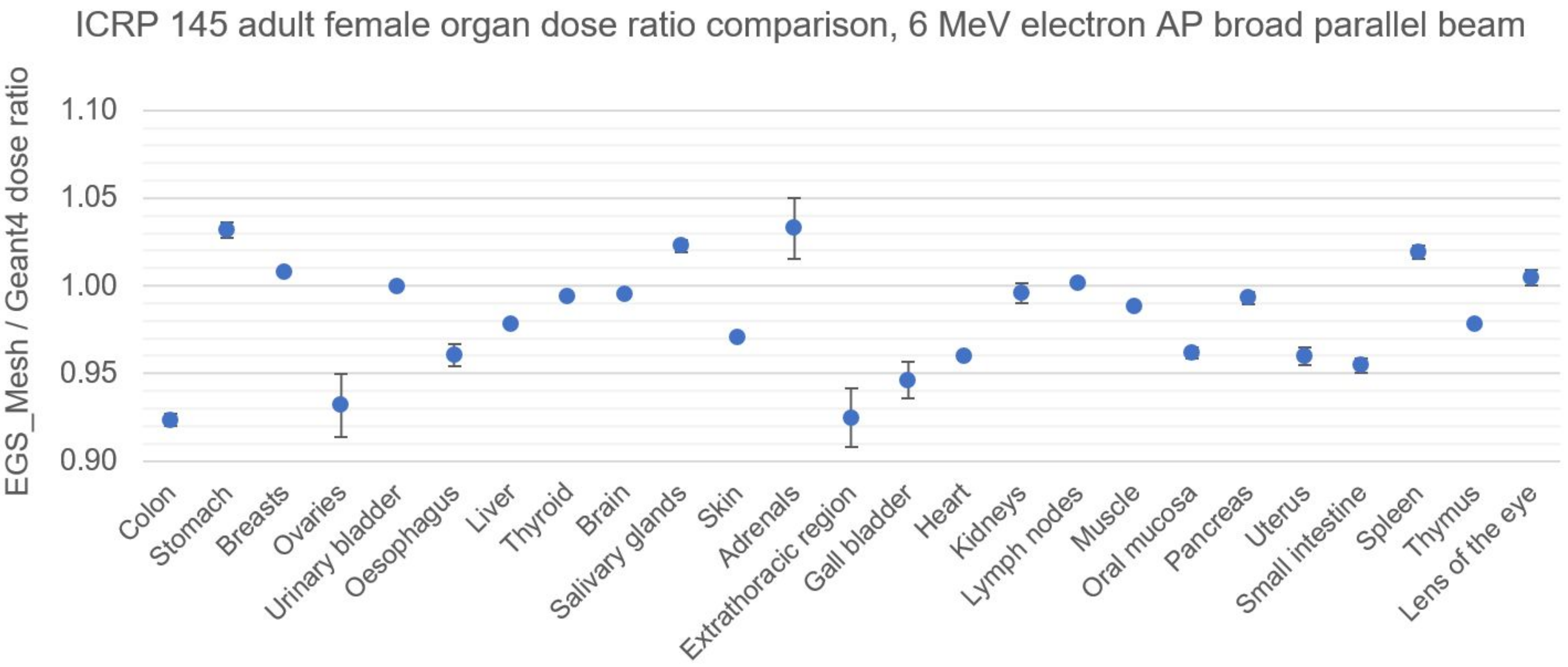
1. IRCP. “Adult Mesh-type Reference Computational Phantoms”. In: Annals of the ICRP 49 (2020). ICRP Publication 145. p. 20, bullet 15.
2. IRCP. “Adult Mesh-type Reference Computational Phantoms”. In: Annals of the ICRP 49 (2020). ICRP Publication 145. p. 15.

Comparison with Geant4 organ doses*, 5 MeV electrons



*Some organs excluded (red bone marrow, lungs, endosteum). Geant4 results from Yeom et al. 2019. “Dose coefficients of mesh-type ICRP reference computational phantoms for idealized external exposures of photons and electrons”. Note: Geant4 organ dose uncertainties under 1%.

Comparison with Geant4 organ doses*, 6 MeV electrons



*Some organs excluded (red bone marrow, lungs, endosteum). Geant4 results from Yeom et al. 2019. “Dose coefficients of mesh-type ICRP reference computational phantoms for idealized external exposures of photons and electrons”. Note: Geant4 organ dose uncertainties under 1%.

Conclusions

- Tetrahedral meshes are here: the next generation of complex, high-quality radiation transport phantoms.
- EGS_Mesh allows users to simulate tetrahedral meshes including CAD meshes using the trusted code EGSnrc.
- Preliminary results using EGS_Mesh and Geant4 tetrahedral meshes agree within 10%. Comparing independent codes on identical meshes enables stronger validation studies.

Thank you to Dave Macrillo, Matt Ronan, Nigel Vezeau, Lou Thompson, Dave Brown and Emily Craven at Mevex. Thanks also to Dr. James McDonald and Dr. Frédéric Tessier and Dr. Reid Townson at the NRC.

Future work

1. Finish comparison against ICRP 145 mesh phantom report and other published results.
2. Internal mesh sources (e.g. use the thyroid as a radiation source)
3. Upcoming ICRP paediatric mesh phantoms
4. Aim to include EGS_Mesh in EGSnrc 2022 this spring for a beta release.

References

Tetrahedral mesh phantoms

- IRCP. “Adult Mesh-type Reference Computational Phantoms”. In: Annals of the ICRP 49 (2020). ICRP Publication 145.
- R. L. Martz. “MCNP6 Unstructured Mesh Initial Validation and Performance Results”. In: Nuclear Technology 180.3 (2012), pp. 316–335.
- T. Furuta et al. “Implementation of tetrahedral-mesh geometry in Monte Carlo radiation transport code PHITS”. In: Physics in Medicine & Biology 62.12 (2017), pp. 4798–4810.
- Y. S. Yeom et al. “Tetrahedral-mesh-based computational human phantom for fast Monte Carlo dose calculations”. In: Physics in Medicine & Biology 59.12 (May 2014),pp. 3173–3185.
- Y. S. Yeom et al. “Dose coefficients of mesh-type ICRP reference computational phantoms for idealized external exposures of photons and electrons”. In: Nuclear Engineering and Technology 51.3 (2019), pp. 843–85.
- W. Kainz et al. “Advances in Computational Human Phantoms and Their Applications in Biomedical Engineering - A Topical Review”. eng. In: IEEE transactions on radiation and plasma medical sciences 3.1 (2019), pp. 1–23.
- Y. Yeom et al. “Computation Speeds and Memory Requirements of Mesh-Type ICRP Reference Computational Phantoms in Geant4, MCNP6, and PHITS”. English. In: Health Physics 116.5 (May 2019), pp. 664–676.
- C. Choi et al. “Development of paediatric mesh-type reference computational phantom series of International Commission on Radiological Protection”. In: Journal of Radiological Protection 41.3 (Aug. 2021), S160–S170.

Voxel reference phantoms

- IRCP. “Adult Reference Computational Phantoms”. In: Annals of the ICRP 39 (2009). ICRP Publication 110.
- IRCP. “Conversion Coefficients for Radiological Protection Quantities for External Radiation Exposures”. In: Annals of the ICRP 40 (2010). ICRP Publication 116.

References (continued)

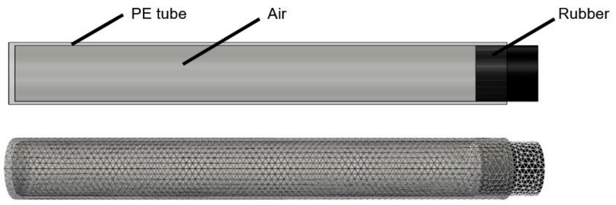
Applications

- G.P. Fonseca et al. The use of tetrahedral mesh geometries in Monte Carlo simulation of applicator based brachytherapy dose distributions. Phys Med Biol. 2014 Oct 7;59(19):5921-35.
- H. Lee et al. Percentile-specific computational phantoms constructed from ICRP mesh-type reference computational phantoms (MRCPs). Phys Med Biol. 2019 Feb 5;64(4):045005.
- B.-W. Cheon et al. “Development of a novel program for conversion from tetrahedral-mesh-based phantoms to DICOM dataset for radiation treatment planning: TET2DICOM”. In: Journal of Applied Clinical Medical Physics 23.1 (2022).
- L. M. Carter et al. “Patient Size-Dependent Dosimetry Methodology Applied to 18F-FDG Using New ICRP Mesh Phantoms”. In: Journal of Nuclear Medicine 62.12 (2021), pp. 1805–1814.
- T. G. Primidis et al. “3D chest tomosynthesis using a stationary flat panel source array and a stationary detector: a Monte Carlo proof of concept”. In: Biomedical Physics & Engineering Express 8.1 (Nov. 2021), p. 01500.
- C. Choi et al. “Body-size-dependent phantom library constructed from ICRP mesh-type reference computational phantoms”. In: Physics in Medicine & Biology 65.12 (June 2020), p. 125014.
- Y. S. Yeom et al. “Posture-dependent dose coefficients of mesh-type ICRP reference computational phantoms for photon external exposures”. In: Physics in Medicine & Biology 64.7 (Apr. 2019), p. 075018.
- Y. S. Yeom et al. “INVESTIGATION OF THE INFLUENCE OF THYROID LOCATION ON IODINE-131 S VALUES”. In: Radiation Protection Dosimetry 189.2 (Apr. 2020), pp. 163–171.
- Y. Yin et al. “Physical dosimetric reconstruction of a case of large area back skin injury due to overexposure in an interventional procedure”. In: Radiation Medicine and Protection (2022).

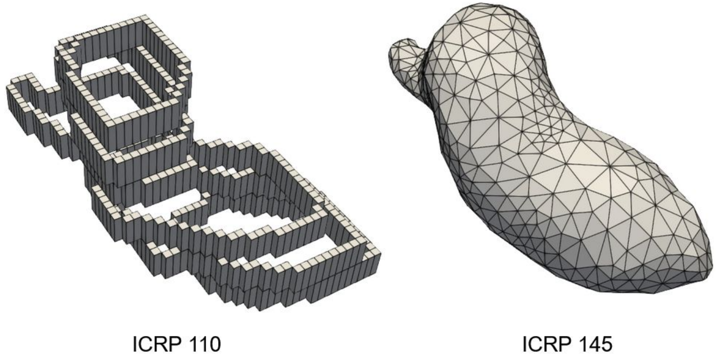
EGS_Mesh summary

Mesh use case: accurate dose uniformity ratios

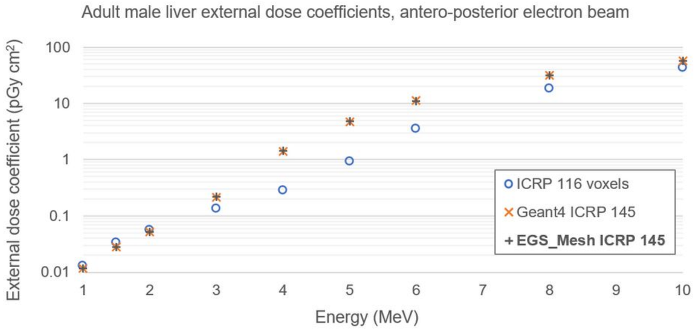
- Sample use case: calculate dose uniformity ratios for sterilization
- Example: test tube CAD model and mesh



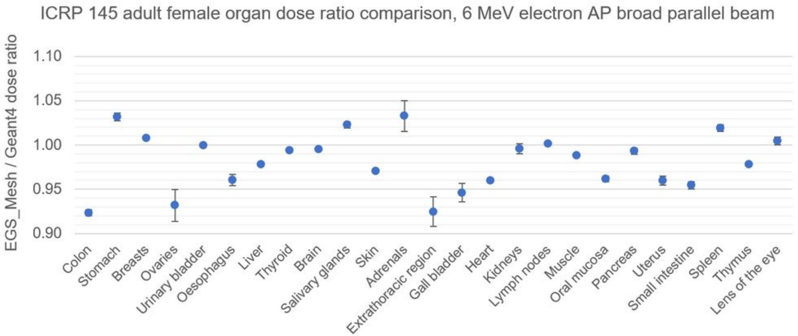
Voxel vs mesh reference phantom gallbladder wall



Comparison to ICRP 116: liver, electrons



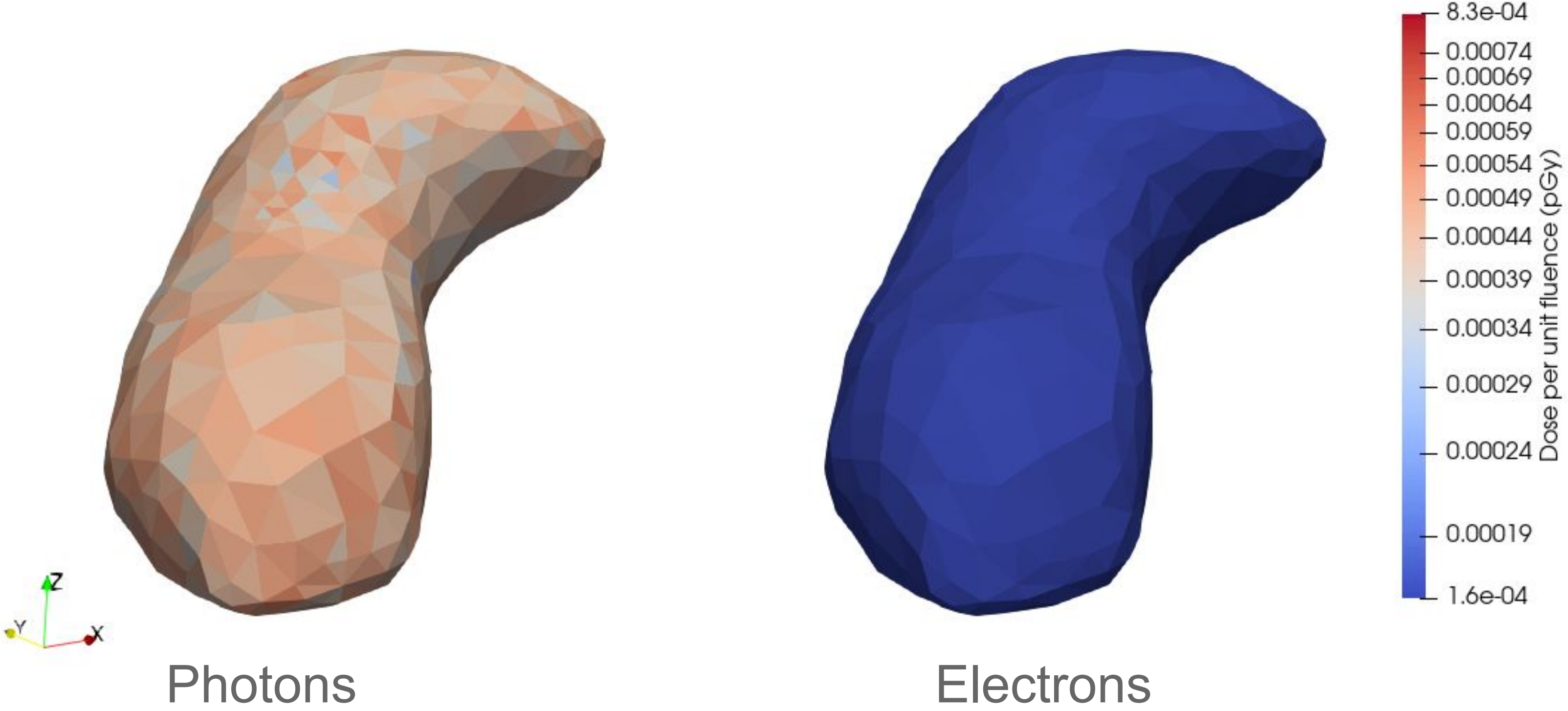
Comparison with Geant4 organ doses*, 6 MeV electrons



*All uncertainties less than 1%.

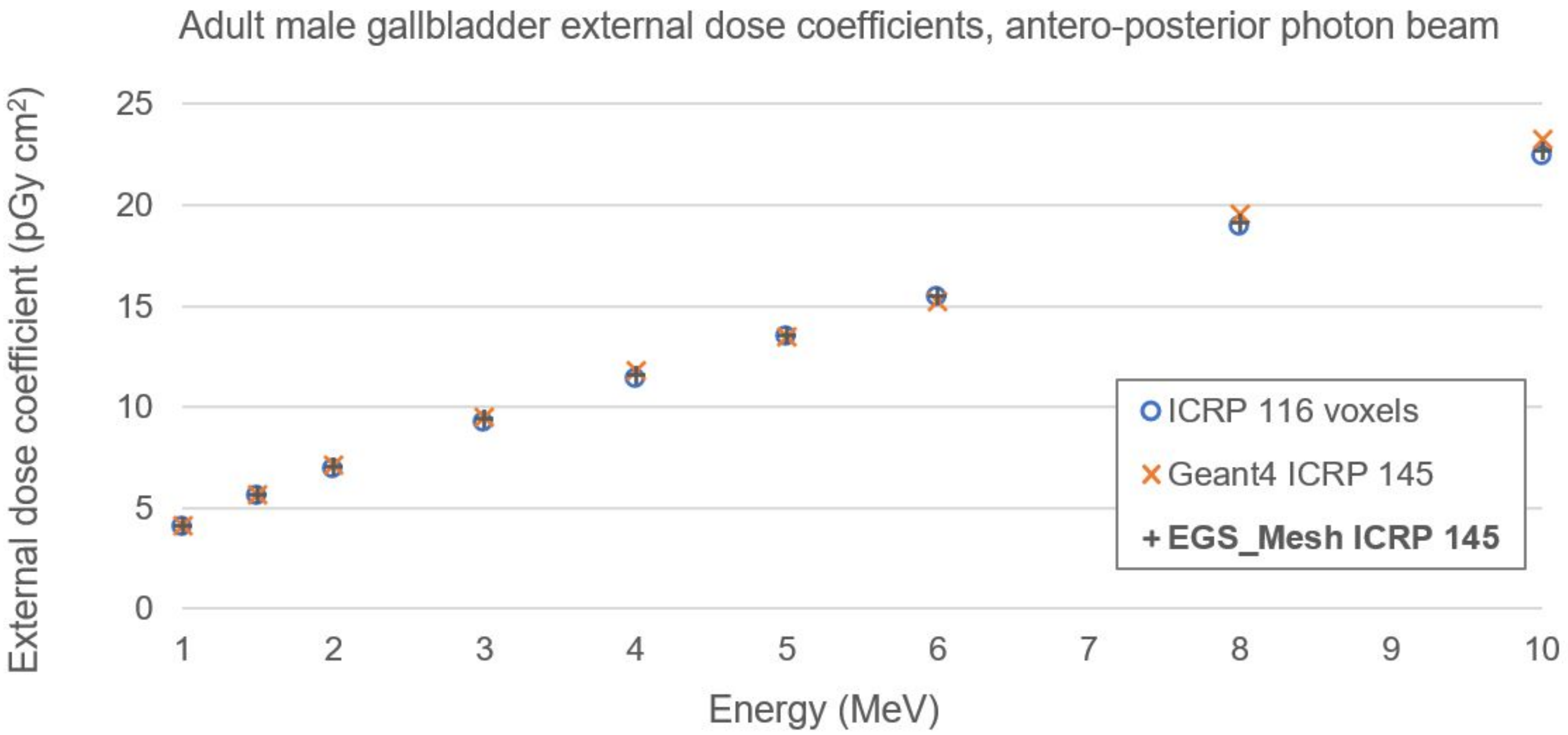
Backup slides

5 MeV photon and electron gallbladder dose



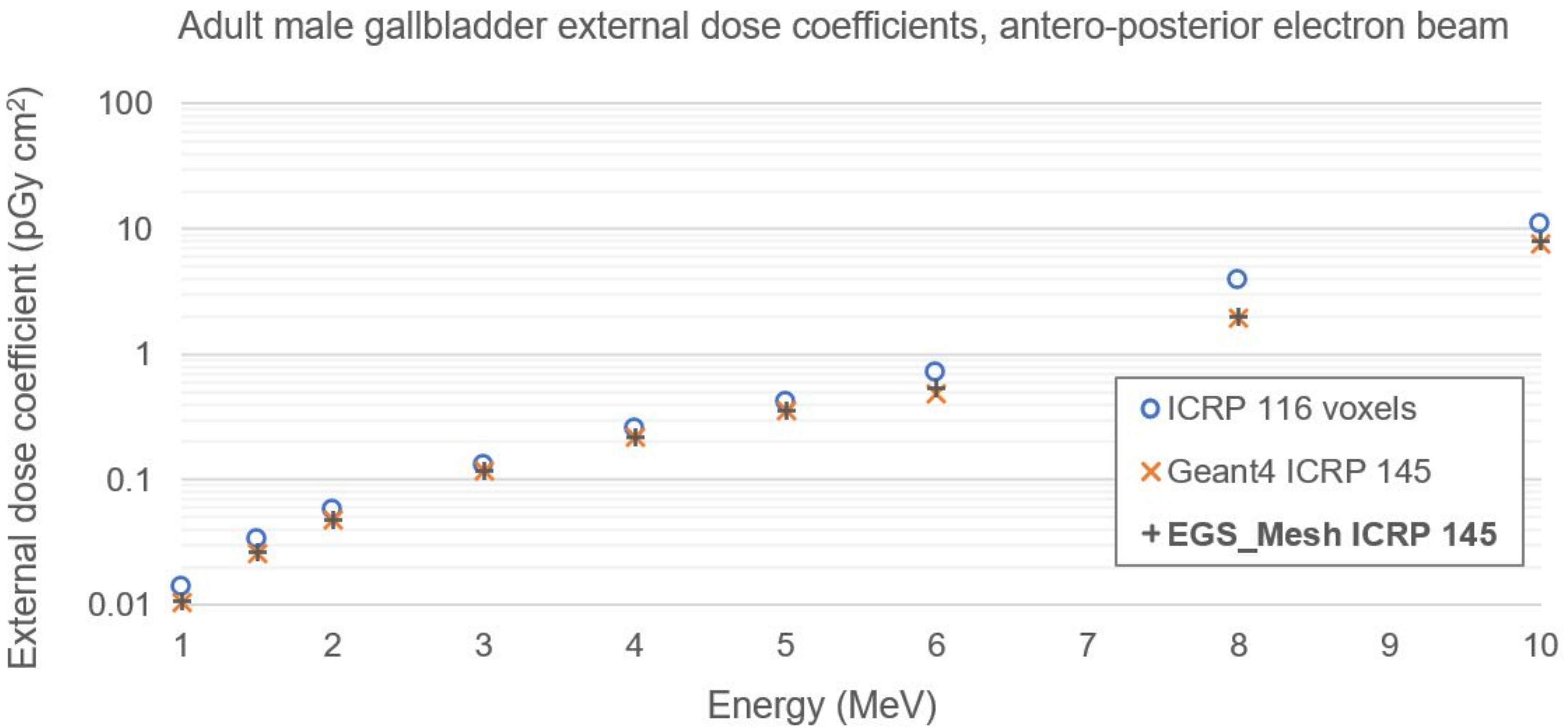
Results for a broad parallel beam along +Y axis. Graph uses a log scale.

Comparison to ICRP 116: gallbladder, photons



*All uncertainties less than 1%.

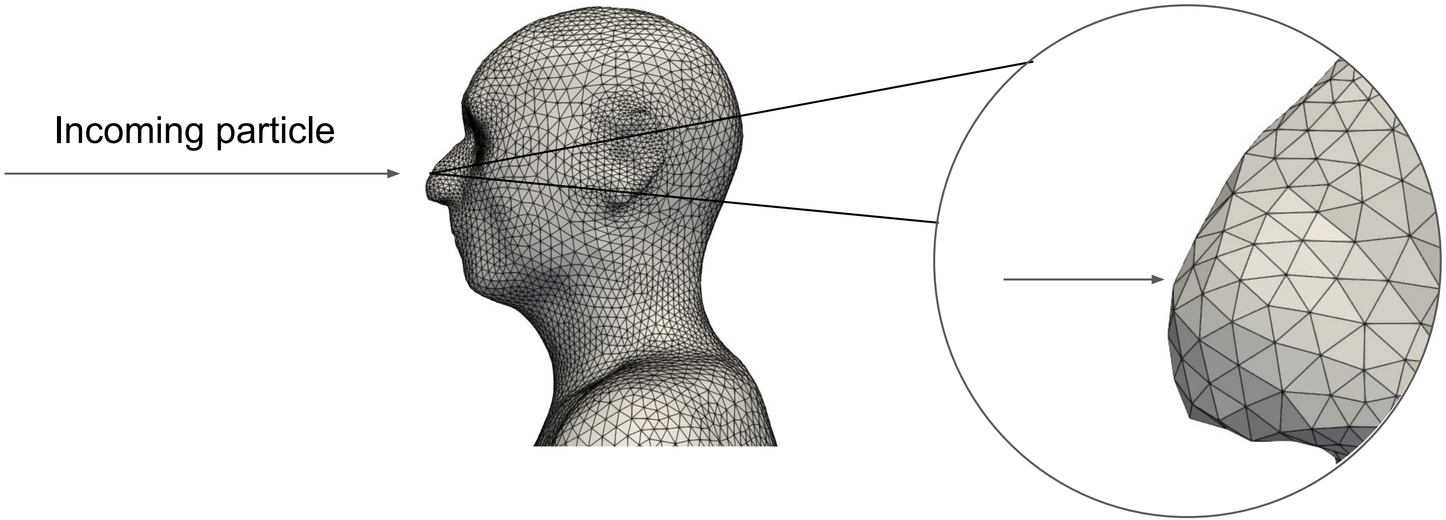
Comparison to ICRP 116: gallbladder, electrons



*All uncertainties less than 1%.

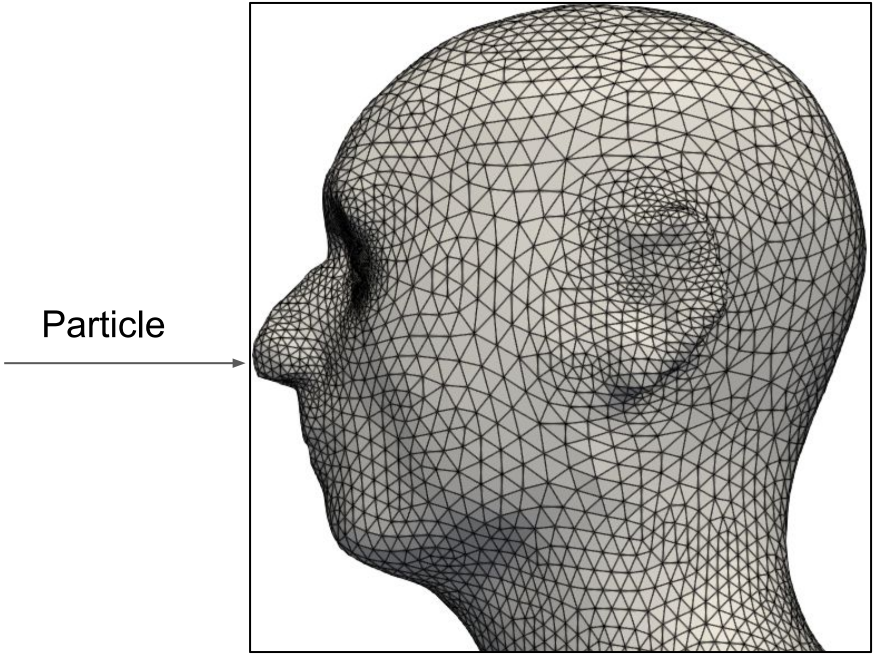
How tetrahedral mesh transport works

- Each tetrahedron has 4 triangular faces.
- Calculating mesh distances = distance to a triangle.
- Brute force search is extremely slow… need a faster way



Transport acceleration

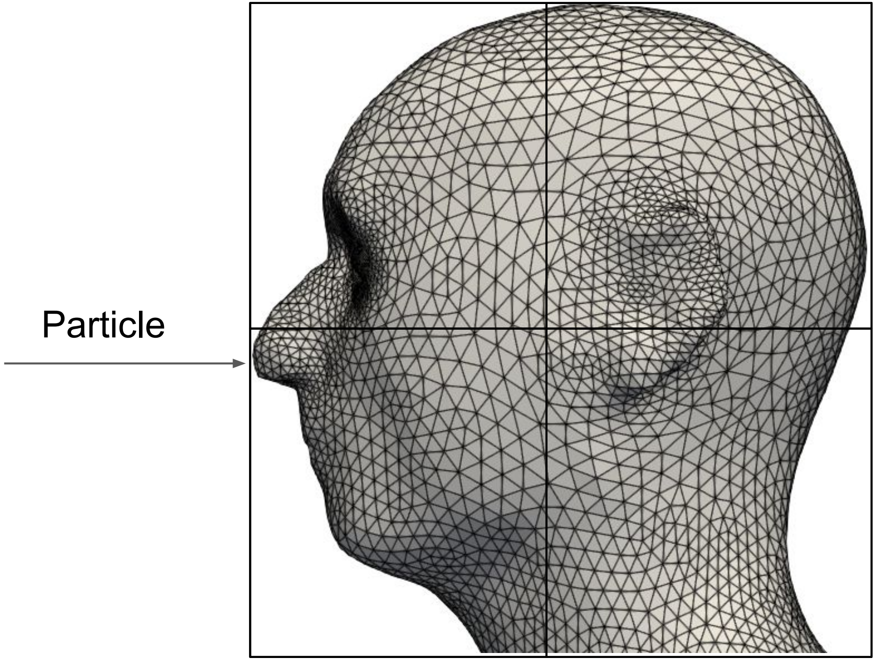
- Octree partitioning[1]: instead of searching all elements, only search in the partition where the intersection could possibly occur.



1. Idea from PHITS implementation: Furuta et al. 2017. “Implementation of tetrahedral-mesh geometry in Monte Carlo radiation transport code PHITS”. Octrees are very common in computer graphics.

Transport acceleration

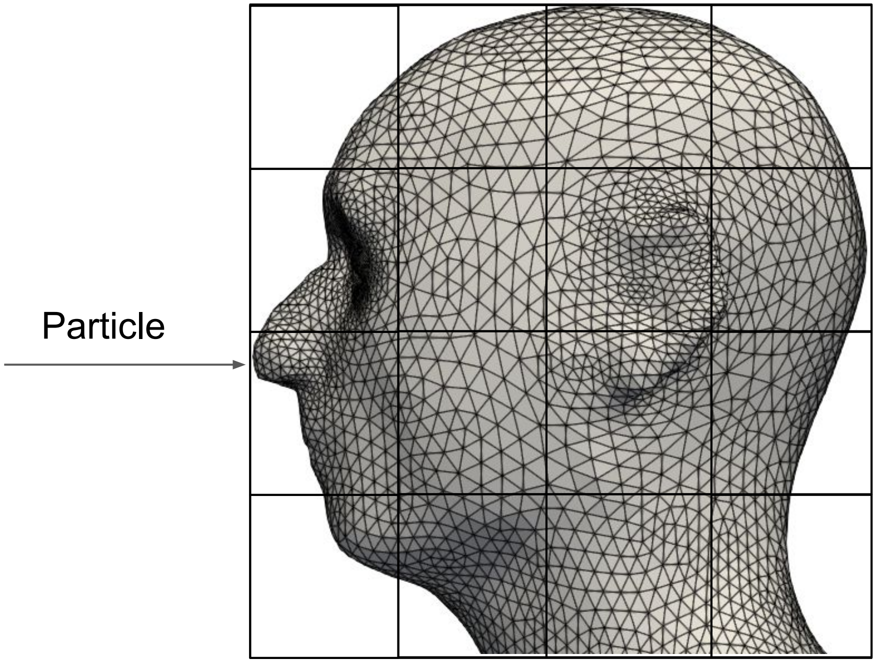
- Octree partitioning[1]: instead of searching all elements, only search in the partition where the intersection could possibly occur.



1. Idea from PHITS implementation: Furuta et al. 2017. “Implementation of tetrahedral-mesh geometry in Monte Carlo radiation transport code PHITS”. Octrees are very common in computer graphics.

Transport acceleration

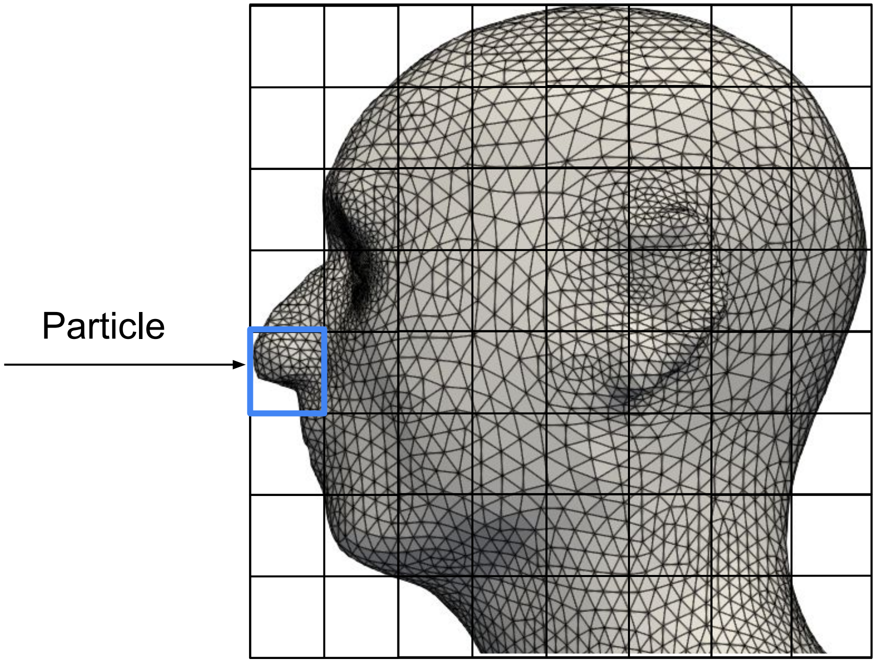
- Octree partitioning[1]: instead of searching all elements, only search in the partition where the intersection could possibly occur.



1. Idea from PHITS implementation: Furuta et al. 2017. “Implementation of tetrahedral-mesh geometry in Monte Carlo radiation transport code PHITS”. Octrees are very common in computer graphics.

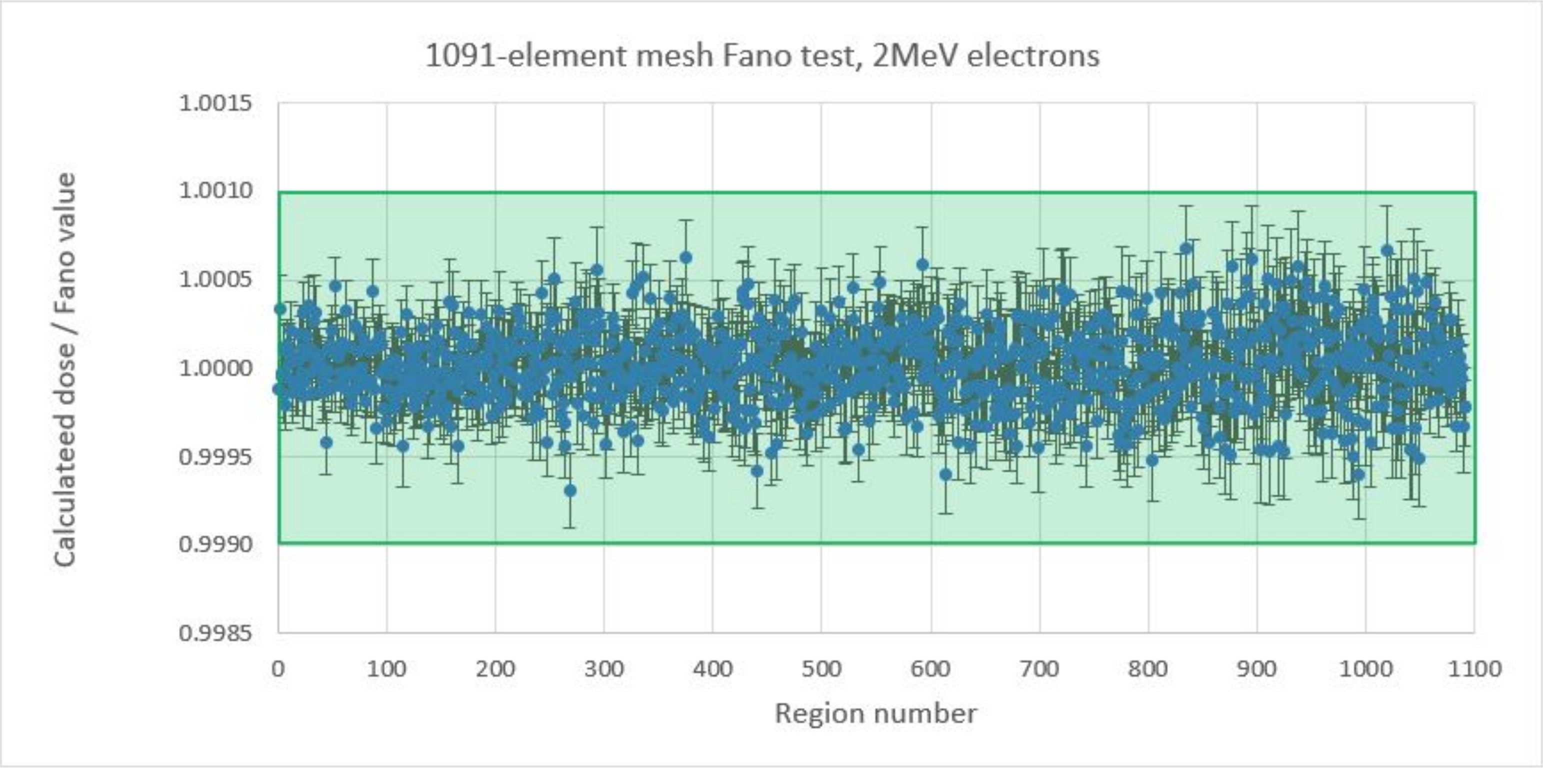
Transport acceleration

- Octree partitioning[1]: instead of searching all elements, only search in the partition where the intersection could possibly occur.
- Partitioning ends when there is a small enough number of elements in each octant (e.g. 100-200)[1].



1. Idea from PHITS implementation: Furuta et al. 2017. “Implementation of tetrahedral-mesh geometry in Monte Carlo radiation transport code PHITS”. Octrees are very common in computer graphics.

Theory: Fano test results for a ~1000 element mesh



EGS_Mesh metrics (using ICRP 145 male phantom)

1. Histories per second for 1MeV-10MeV range: **1k to 10k**, competitive with other codes (to be confirmed as part of verification work)

Table 2. Memory usages of adult male MRCP, adult male VRCP, and voxelized phantoms in Geant4, MCNP6, and PHITS (unit: GB).

Phantoms	Geant4	MCNP6	PHITS
Adult male MRCP	10.6	13.7	1.2

2. Memory use: **2GB**

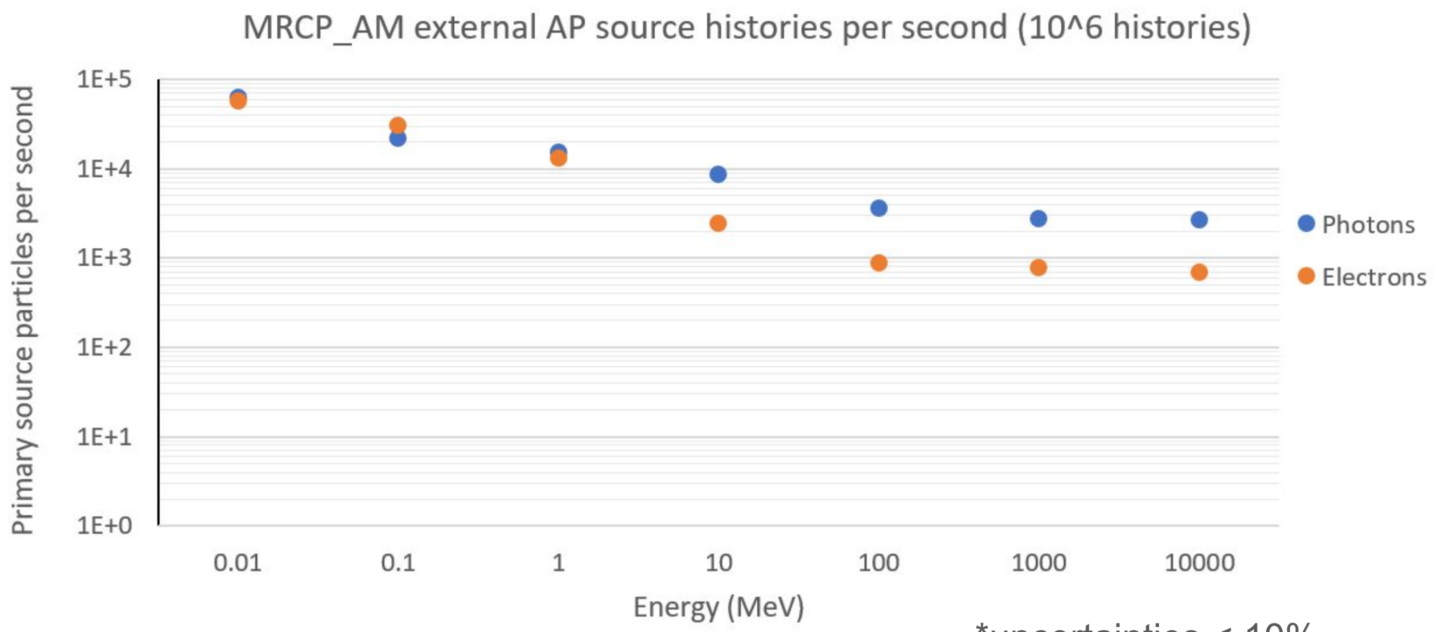
Table 3. Initialization times of adult male MRCP, adult male VRCP, and voxelized phantoms in Geant4, MCNP6, and PHITS (unit: minutes).

Phantoms	Geant4	MCNP6	PHITS
Adult male MRCP	3.3	2.3	0.2

3. Initialization time: **3 minutes**

Tables from "Computation Speeds and Memory Requirements of Mesh-Type ICRP Reference Computational Phantoms in Geant4, MCNP6, and PHITS" Yeom et al. (2019)

Simulation performance of ICRP 145 adult male phantom



*uncertainties < 10%

CPU information: Single core of Intel(R) Xeon(R) CPU E5-2683 v4 @ 2.10GHz (125 GiB RAM), CentOS 7.

Documentation!

https://mxxo.github.io/egs_mesh/docs/quickstart.html

EGS_Mesh

`EGS_Mesh` is an unstructured tetrahedral mesh library for EGSnrc.

```
:start geometry:
    library = egs_mesh
    name    = my_mesh
    file    = # your mesh here
:stop geometry:
```

If you're new to `EGS_Mesh`, you can work through a guided example. You can also consult the `egsinp` syntax reference. If you want to simulate STEP files using `EGS_Mesh`, check out the STEP file example.

Overview

`EGS_Mesh` offers modelling flexibility over traditional voxel-based simulations. Instead of building up a geometry using a constructive-solid approach, `EGS_Mesh` uses tetrahedral mesh files as input. CAD geometries can be meshed and then simulated directly.

Users can generate meshes using the standalone tool Gmsh, or create a mesh file from their data.

Example simulation

https://mxxo.github.io/egs_mesh/docs/example/example.html

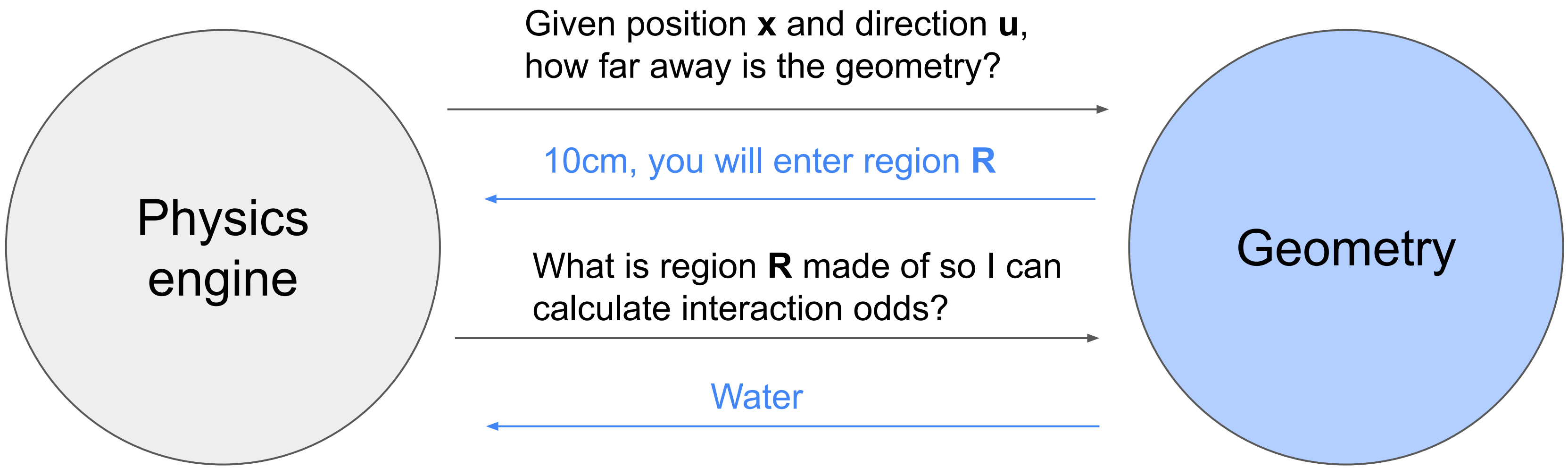
Guided simulation

This guide goes through a complete `EGS_Mesh` simulation step by step. By the end, you'll be ready to conduct your own simulations. Along the way we'll cover:

- Fundamental EGSnrc and `EGS_Mesh` concepts
- Gmsh basics
- Running EGSnrc simulations
- Viewing `EGS_Mesh` simulation results in Paraview

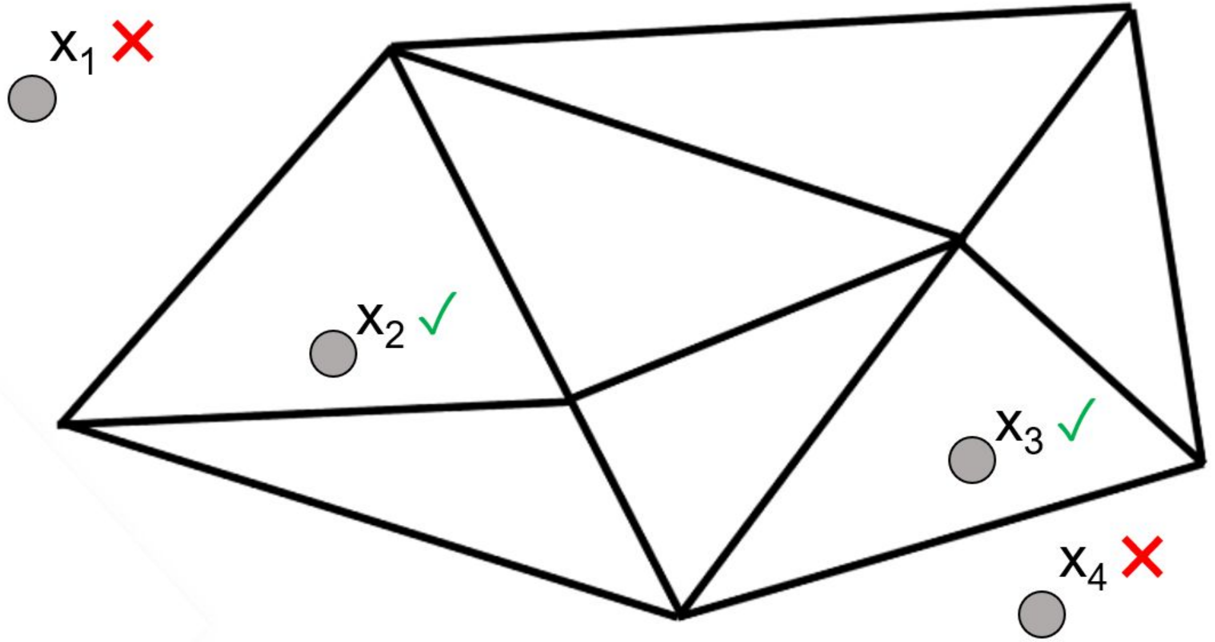
EGSnrc simulation overview

- Roughly speaking, EGSnrc = physics + geometry. The geometry (EGS_Mesh) is in charge of calculating particle intersections and other geometric queries.
- EGSnrc handles everything else.



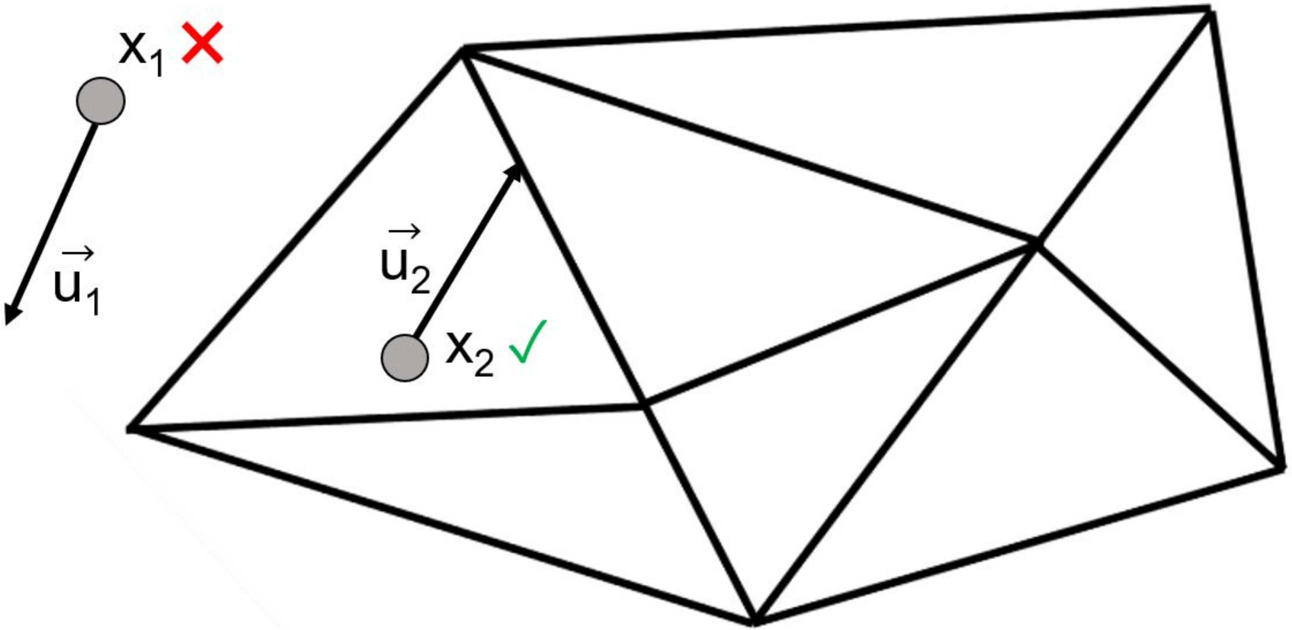
Geometry implementation requirements (1)

- `isWhere`: Is the particle with position **x** inside a tetrahedron?



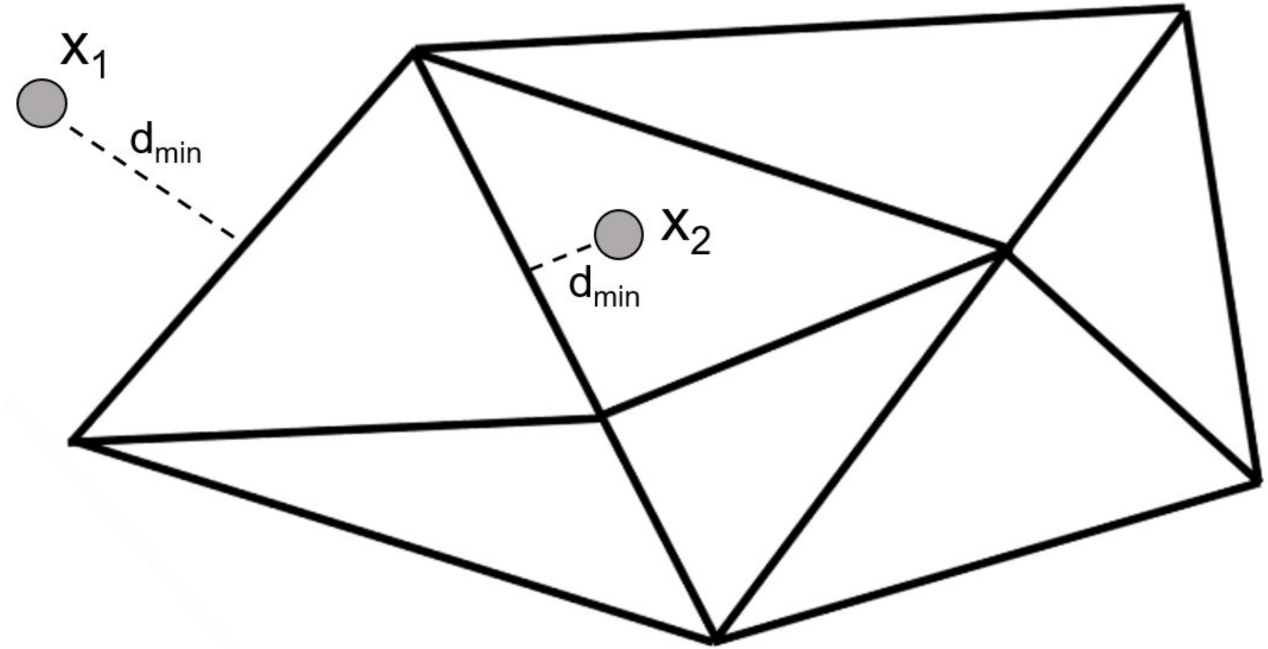
Geometry implementation requirements (2)

- `howfar`: Given a particle with position **x** and direction **u**, find the next intersection with a geometry boundary.



Geometry implementation requirements (3)

- **`hownear`**: What is the minimum distance to a boundary for a particle with position **x**?



Geometry implementation requirements (4)

- By the way… the mesh geometry must be resilient to edge cases.
- The geometry routines are called many times during a simulation. If you're simulating 1 billion particles, there will be X billion calls to **`howfar`** and friends.
- A naive implementation will almost certainly get stuck in an infinite loop due to floating point issues.
- The implementation has to be robust, even when the geometry math returns garbage.