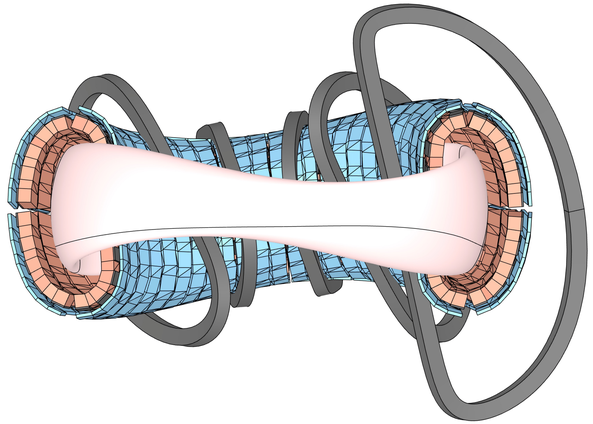
THEA ENERGY

Automated methods for stellarator modelling and simulation

Alex Koen, Soha Aslam, Andrew Cote



Motivation

- The geometry of stellarators is computationally-defined
- Traditional GUI CAD tools are not suited to algorithmic geometry
- Models need to be programmatically generated from design parameters without human intervention

Results

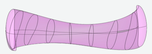
- A three-step workflow for geometry creation, meshing, and simulation
- The development of **Stellarmesh** [1], an MIT-licensed meshing tool suited to generation of DAGMC [2] geometry for Monte-Carlo transport simulation using OpenMC [3]

Step 1. Geometry creation with OpenCASCADE

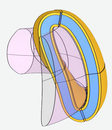
From a set of base parameters, blanket models are generated using **build123d** [4], a python wrapper around the **OpenCASCADE** geometry kernel, with the following steps:

1. Points are extracted from a computationally-defined plasma surface.
2. NURBS surfaces are skinned over the point cloud to generate either full blanket models or individual panels.
3. Blankets are generated by repeatedly offsetting and thickening the plasma surface.
4. Layers are variated to test different designs.

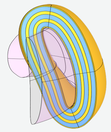
1. Plasma point-cloud



2. NURBS surface fitted to point cloud



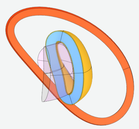
3. Construction of blanket geometry



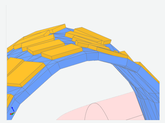
4. Blanket variations

Other components may additionally be generated using either build123d or OpenCASCADE. These include:

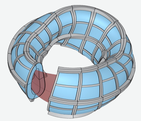
- **Winding surface tiling layout** for planar coil mounting
- Mechanical structure of field-shaping unit (FSU) panels.
- Wiring layouts



Major coils



Winding surface tiling layout for planar coils



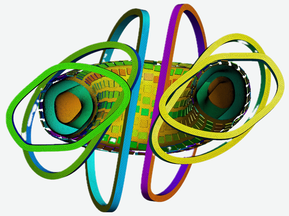
Prototype mechanical structure

Step 2. Meshing with Stellarmesh

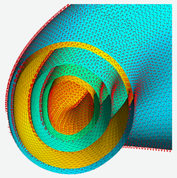
Meshing is a requisite step between geometry creation and simulation, and teams have historically struggled to generate **conformal meshes** i.e. meshes which do not contain overlapping surfaces.

Stellarmesh [1] is Thea's open-source meshing library for **DAGMC** [2] geometry creation:

- MIT-licensed
- Fully implements the DAGMC mesh metadata specification, notably including **surface-sense**
- Supports OpenCASCADE-native and STEP geometry
- Uses the **Gmsh** [5] library for meshing applications
- Supports remeshing



Conformal stellarator blanket and coil mesh generated and visualized with Stellarmesh



Stellarmesh correctly tracks surface normals, allowing DAGMC to properly orient rays during simulation

Step 3. Simulation with OpenMC

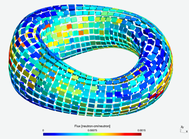
The first iteration of the workflow tools developed in this research focus specifically on Monte-Carlo transport simulation of neutron-blanket interactions.

Blanket designs vary in layer geometry, materials, and presence of gaps/ports and are optimized for:

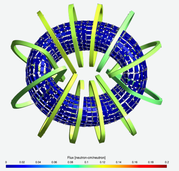
- Tritium-breeding ratio (TBR)
- Leakage fraction

Cell-focused results are visualized using the **post-processing capabilities of Stellarmesh**.

These tools will soon be enhanced to enable additional simulations including finite-element and thermodynamic analysis.



Flux through planar coils visualized with Stellarmesh.



Flux through major coils visualized with Stellarmesh.

References

1. Alex Koen, "Stellarmesh: Make DAGMC geometry from CAD," [Computer Software]. GitHub https://github.com/Thea-Energy/stellarmesh
2. Baptiste Mouginot, Andrew Davis, Patrick Shriwise et. al., "DAGMC - Direct Accelerated Geometry Monte Carlo Toolkit," [Computer Software]. GitHub https://github.com/svalinn/DAGMC
3. Paul K. Romano, Nicholas E. Horelik, Bryan R. Herman, Adam G. Nelson, Benoit Forget, and Kord Smith, "OpenMC: A State-of-the-Art Monte Carlo Code for Research and Development," Ann. Nucl. Energy, 82, 90–97 (2015).
4. Roger Maitland, "Build123d: A Python-based, Parametric, Boundary Representation (BREP) Modeling Framework for 2D and 3D CAD," [Computer software]. GitHub. https://github.com/gumyr/build123d (2023).
5. Geuzaine, Christophe and Remacle, Jean-Francois, "Gmsh," [Computer Software]. http://http://gmsh.info/

65th APS-DPP, October 30-November 3, 2023, Denver, CO
This work is supported by Thea Energy and the University of British Columbia

www.thea.energy
info@thea.energy