Scientific Rendering and Ray-Tracing

Matic Brank, University of Ljubljana

24.09.2024

Session overview

- Digital twin and synthetic diagnostics:
  - Field line tracing
  - Heat transfer
  - Ray tracing

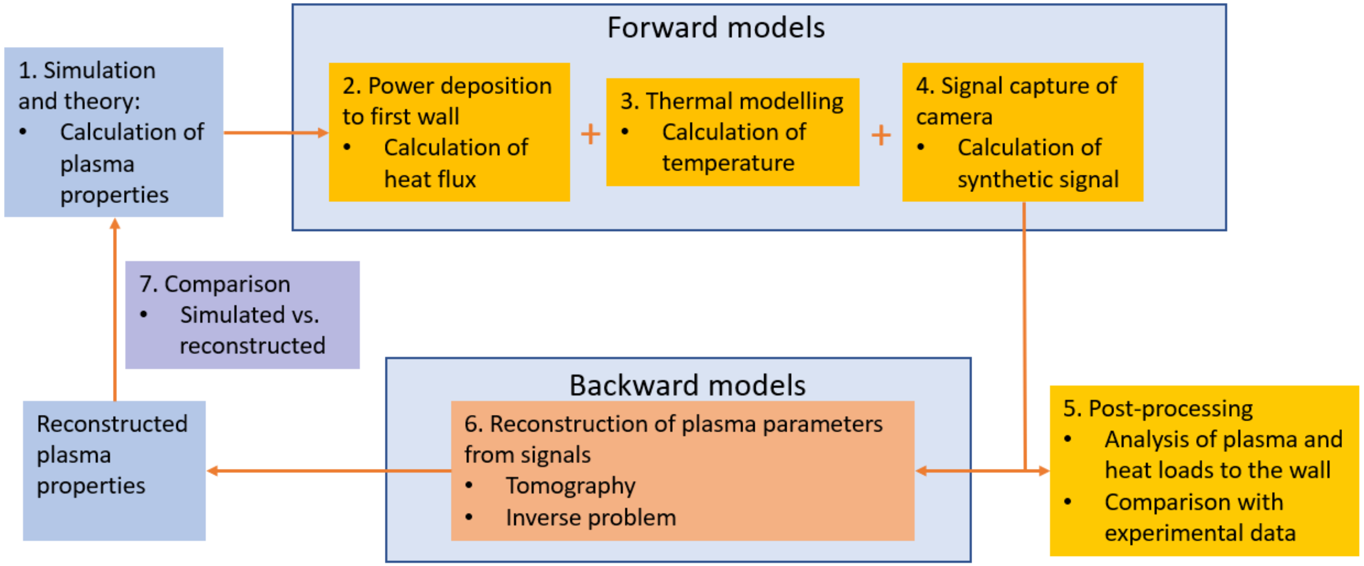
- Ray tracing for scientific computing:
  - Rendering equation
  - Raysect/CHERAB ray tracing packages

Digital twin and synthetic diagnostics

- In magnetically confined devices the plasma facing components need to be protected from excessive heat loads. This is usually achieved through physics-based simulations coupled with experimental results from similar tokamak devices.

- Overall, diagnostic systems that provide insight into tokamak are important to monitor and characterize the plasma as well as the PFCs under high heat fluxes. Disruptive instabilities and all-metal walls, disturbance phenomena such as reflections from surfaces, inaccuracy of emissivity affect the interpretation of measurements and can potentially endanger machine safety. The measurements are thus exposed to high levels of noise which need to be taken into an account as well.

- This makes it challenging to correctly detect plasma properties and heat loads to first wall. One of the approaches is to develop ''synthetic diagnostic'' measurements inside tokamak environment.



Synthetic diagnostics

- Magnetic fusion experiments
  - Expensive and thus scarce
  - Numerical simulations are an alternative

- Simulation of plasma phenomena and response of measurement systems:
  - Calculation of plasma power distribution in vacuum vessel
  - Simulation of plasma power deposition on PFCs
  - Assessment of temperature on PFCs through thermal modelling and assessment of other optical dependent properties
  - Optical simulation to obtain synthetic signal from diagnostic systems

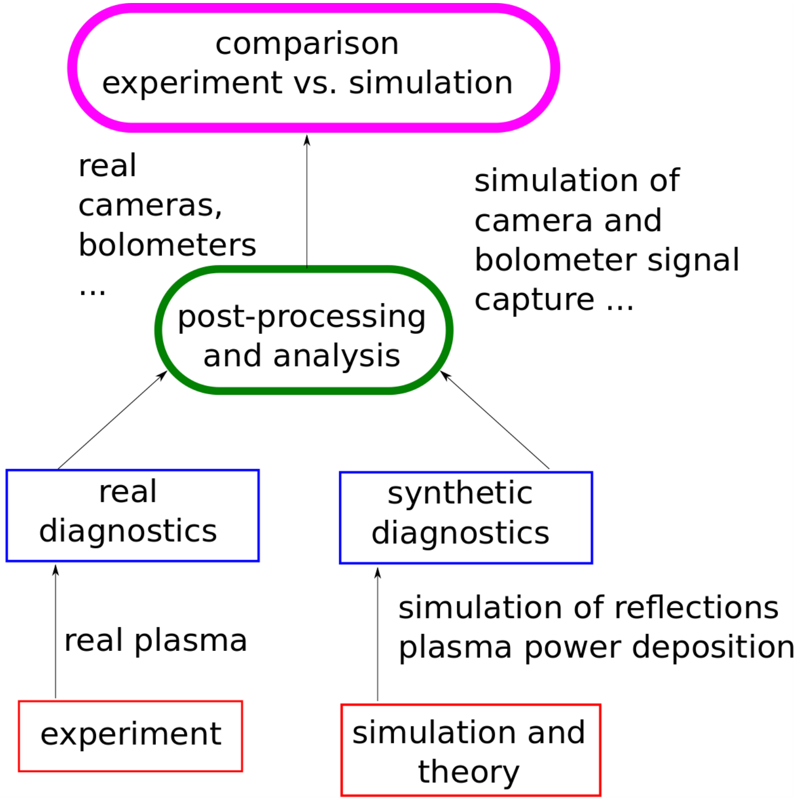


Fig. 1: Full procedure of synthetic diagnostic. The final result of such simulation is measurement signal, e.g. IR image which is then compared to actual experimental result.

Synthetic diagnostics within Excellerat P2

Development of full scale 3D simulation package for synthetic diagnostics in nuclear fusion reactors:

Field line tracing (FLT)
(Following the magnetic field lines in the tokamak)

↓ Heat flux

Thermal model
(calculate temperatures from heat fluxes)

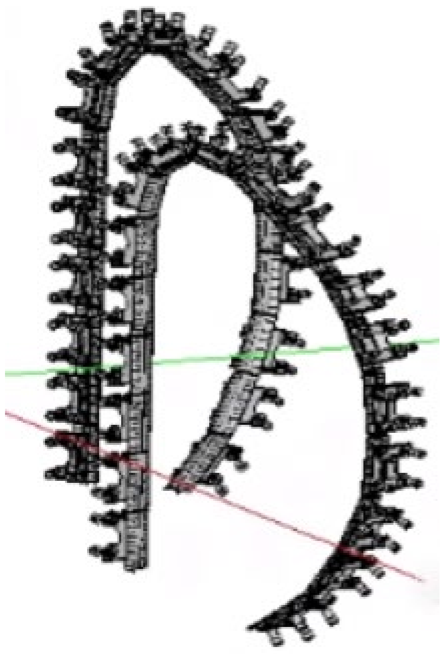
↓ Temperature

Optical simulations
(calculate camera signal)

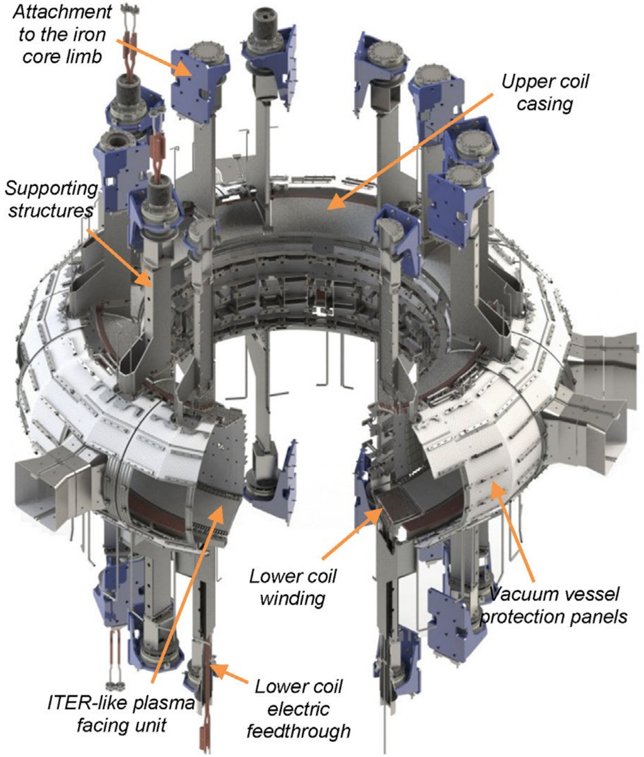
Synthetic camera signal

Three tokamak configurations are available:

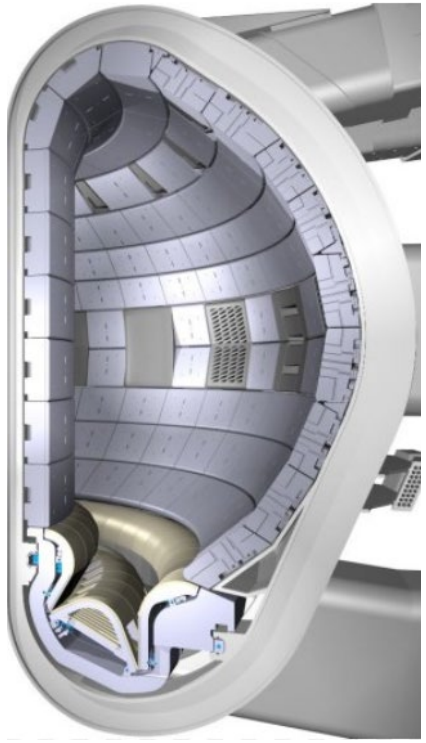
ITER tokamak – temporary limiter



WEST tokamak (CEA)



ITER tokamak – first wall (cross-section)



Field line tracing

Based on particle tracing through magnetic field

In terms of magnetic flux Ψ, the magnetic field components in cylindrical polars are

$$B_R = -\frac{1}{R}\frac{\partial \Psi}{\partial Z}$$

$$B_T = -\frac{I}{R}$$

$$B_Z = \frac{1}{R}\frac{\partial \Psi}{\partial R}$$

The standard field-line equation is given as

$$\dot{x} = \frac{dx}{dt} = B(x)$$

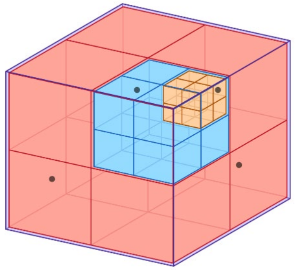
Here dot denotes differentiation with respect to pseudo-time t measured along the field line. The equation is solved using Runge-Kutta schemes (RK2 or RK-Fehlberg scheme) with step adaptation by the Cash-Karp algorithms [1].

[1] L. Shampine and H. Watts, “The art of writing a Runge-Kutta code. II.” Applied Mathematics and Computation, vol. 5, no. 2, pp. 93–121, 1979.

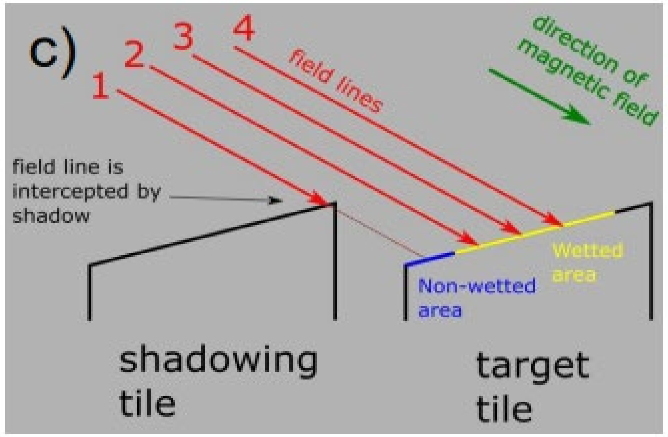
Field line tracing

Field line tracing (FLT)
(Following the magnetic field lines in the tokamak)

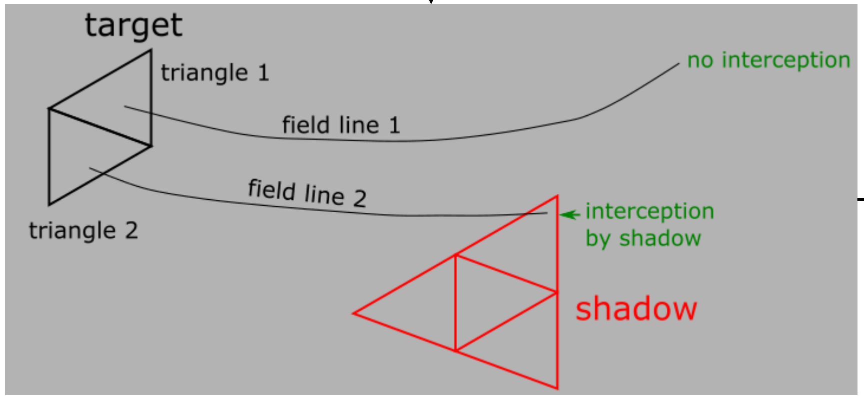
- Runge-Kutta-Fehlberg 4 integration of Magnetic field
  - Starting from target tile
  - To obtain field line of the particle
  - In each step, intersection with the shadowing tiles is checked
- Octree space partitioning
- Bounding box partitioning



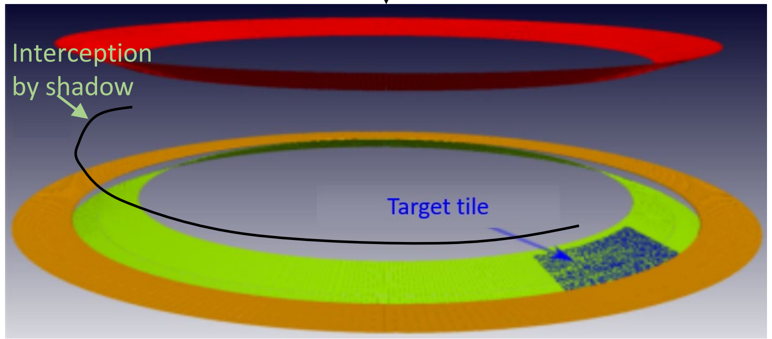
- Multiple FLT codes:
  - PFCFlux
  - Smardda (OpenMPI parallel)
  - L2G (OpenMP)



Both target and shadowing tile are triangular meshes



Algorithm is applied to real data



Field line tracing – WEST tokamak

Field line tracing (FLT)
(Following the magnetic field lines in the tokamak)

- Target -> 1 baffle tile in WEST ~ 9k triangles
- Shadow -> lower baffle, lower, upper target ~450k triangles
- Time: 123 s (36 processes) (Smardda code)

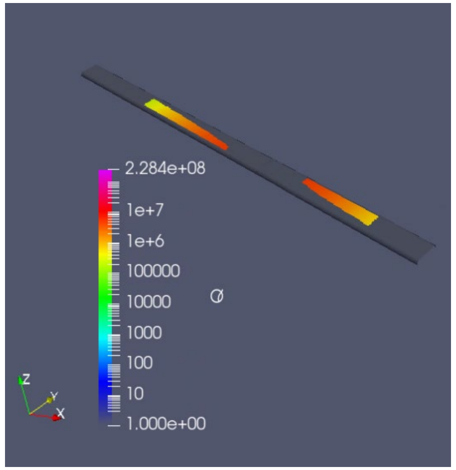


Fig. 1: Heat fluxes on baffle tile

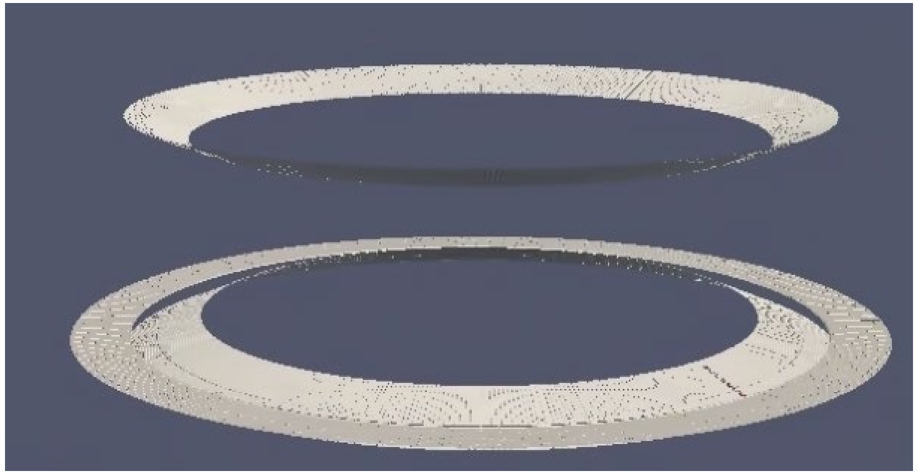


Fig. 2: Shadow

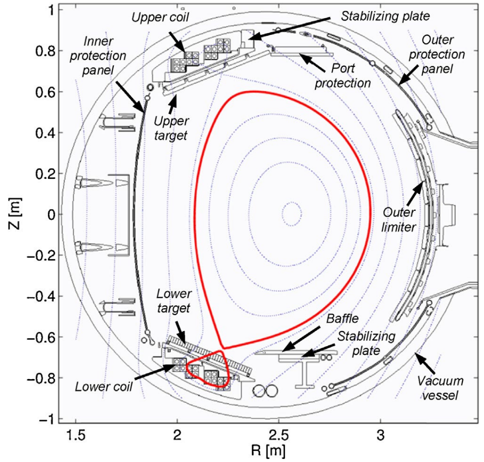


Fig. 2: Cross section of WEST

Thermal model

Thermal model
(calculate temperatures from heat fluxes)

- Finite Volume/Element Method to solve diffusion equation in time
- Unstructured grid -> tetrahedral mesh
- Power deposition from FLT mapped to 3D model and given as heat flux boundary condition

Temperature gradient on the surface is expressed with Fourier's law [2]:

$$\nabla T = -\frac{q_\perp}{k(T)}$$

3D transient heat equation can be then expressed as:

$$\frac{\partial T}{\partial t} = \frac{k(T)}{\rho(T) c_p(T)} \nabla^2 T$$

Density ρ, thermal conductivity k and thermal capacity c_p are temperature dependent material properties.

Codes:

- OpenFOAM (OpenMPI), Elmer FEM, etc.

Thermal model

- Heat loads come mostly from transported particles of magnetic field lines
- Example : ITER tokamak
  - All PFCs are actively cooled
  - Three types of subcomponents in ITER tokamak:
    - Divertor monoblock tiles,
    - Enhanced heat flux panels ($q_{max} = 4.5\ MW/m^2$),
    - Normal heat flux panels ($q_{max} = 2\ MW/m^2$)
- The results are used as input for simulating IR images

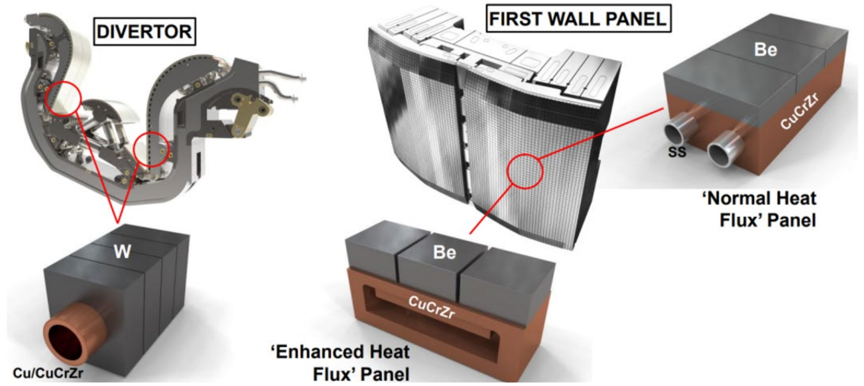


Fig. 1: Three main plasma facing components of ITER [1].

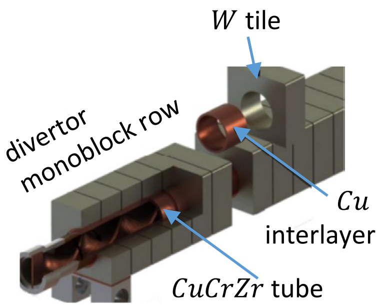
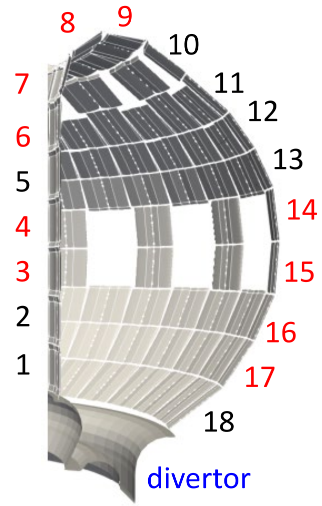


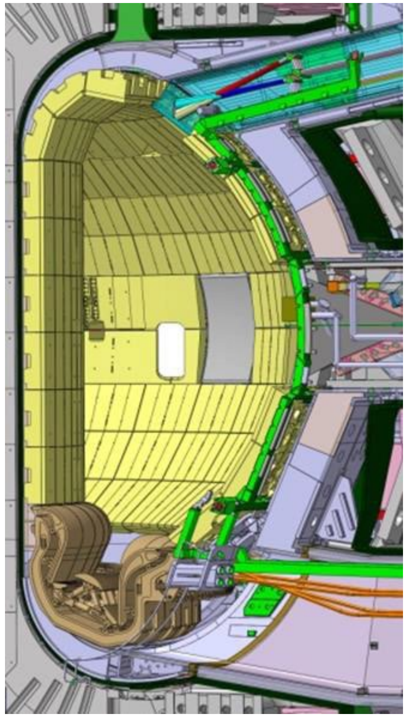
Fig. 2: Decomposition of divertor monoblock [2].

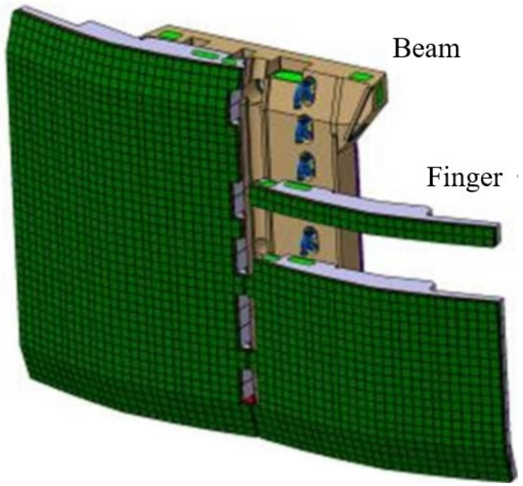


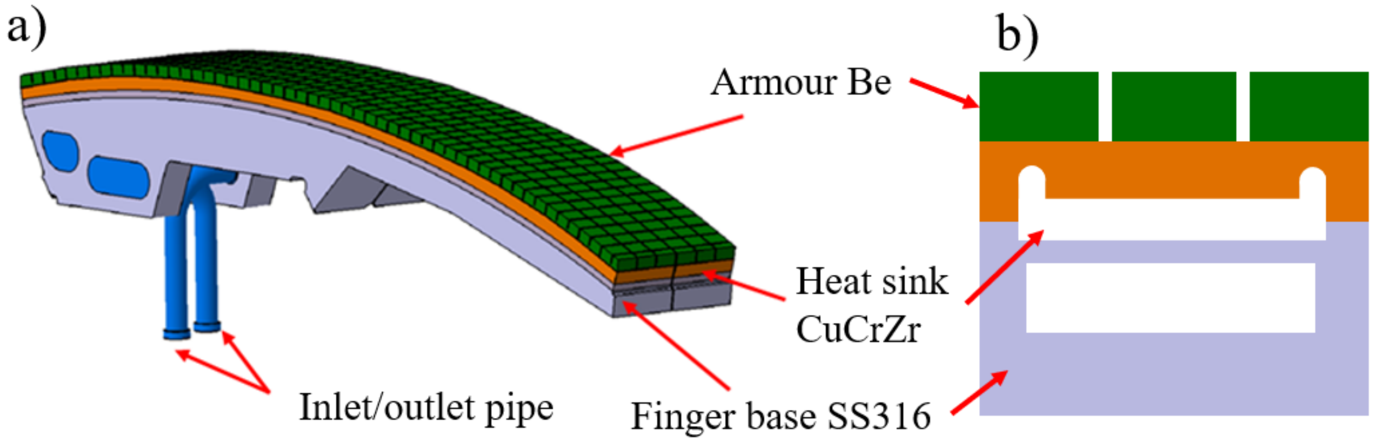
[1] I. Mazul, Preparation to manufacturing of ITER plasma facing components in Russia
[2] X. Bonnin, ITER divertor plasma response to time-dependent impurity injection

Thermal model

3D geometry for ITER first wall consist of cca. 440 panels which cover in total cca. $600\ m^2$. Each panel consist of cca 100 beryllium cubes (depending on the location and variant of panel) positioned on the front surface (depending on the variant)



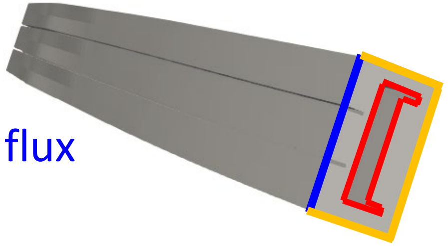




(left) panels in ITER (middle): 1 panel with beryllium cubes (left-a) one finger (left-b) cross-section of finger

Thermal model – simplified model

Heat flux coefficient and external temperature set as boundary condition

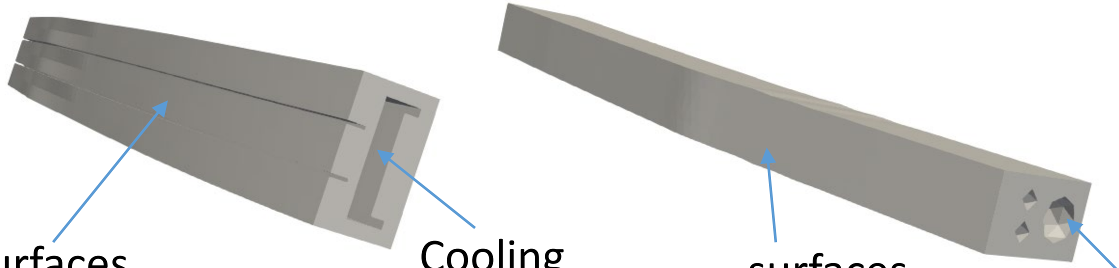


Plasma heat flux

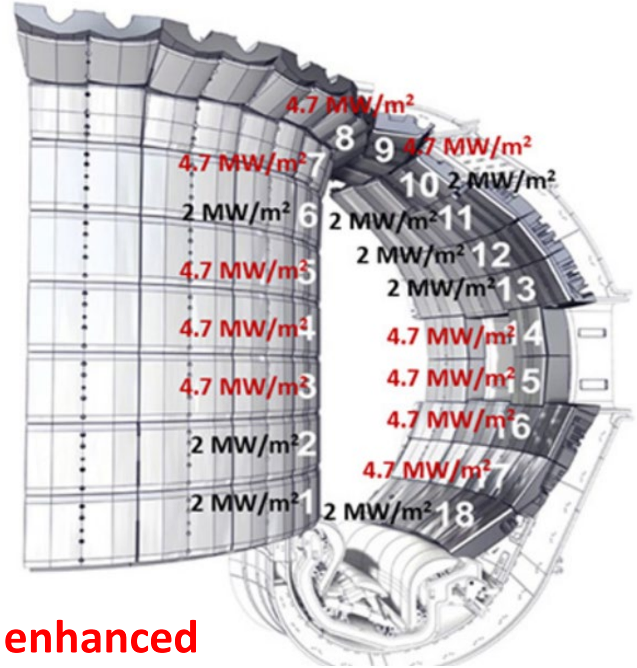
Adiabatic BC

Heat transfer coefficient

Simplified geometry for both enhanced and normal heat flux panel



ITER – first wall
(cross-section)



enhanced

normal

Thermal model

ITER temporary limiter

Using OpenFOAM:
- 48 different timesteps were prepared with 0.04 s step
- Mesh -> ~300k tetrahedrons (1 segment)
- Time -> 1m22s (serial)

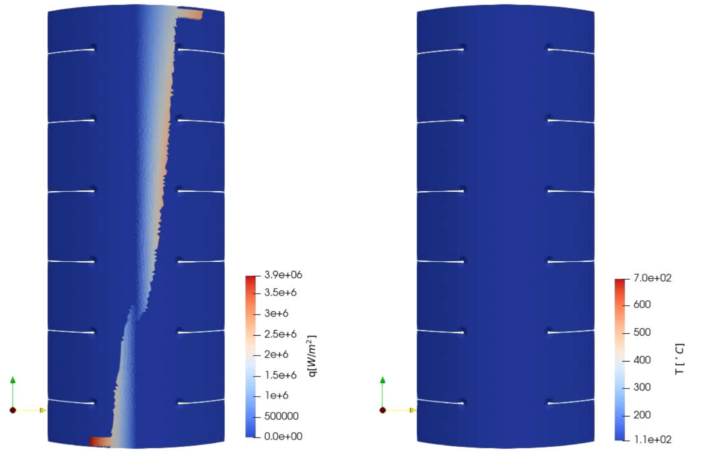


Fig. 1: (left) heat flux (right) temperature

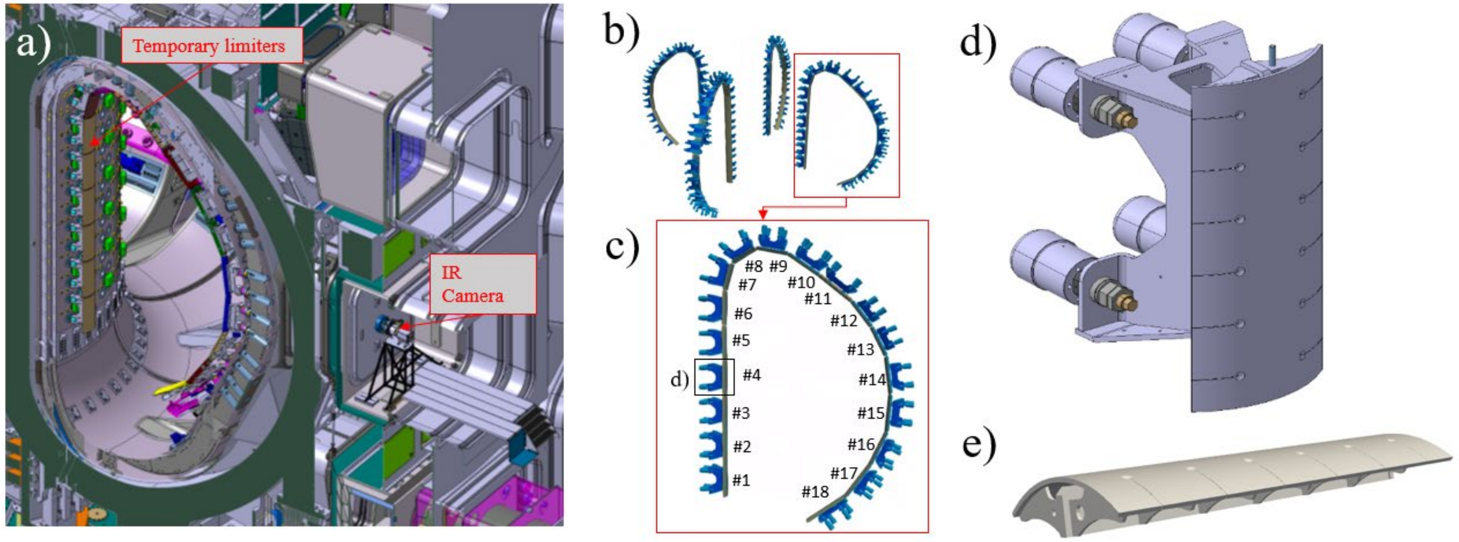


Fig. 2: Temporary limiter breakdown.

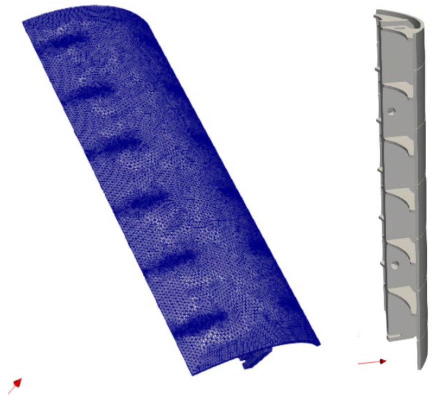


Fig. 3: Left: 3D mesh for thermal model. Right: Side view of TL front structure.

Thermal model

WEST divertor baffle

OpenFOAM:

- 6183k tetrahedrons, 3 timesteps -> 2min19s (serial)

a) b)

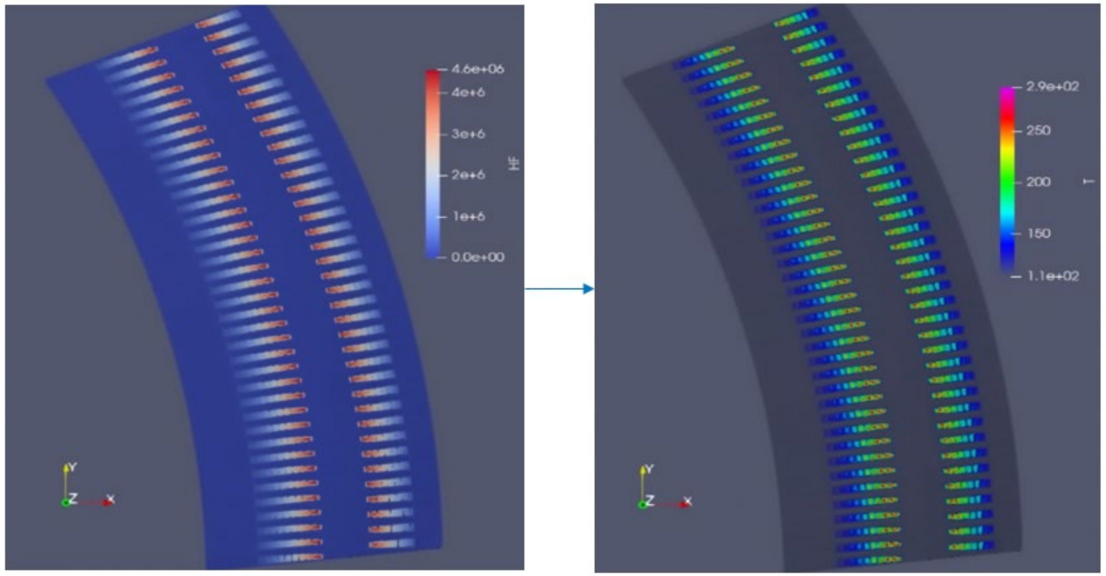


Fig. 1: (a) Heat fluxes (b) Corresponding temperatures (OpenFOAM)

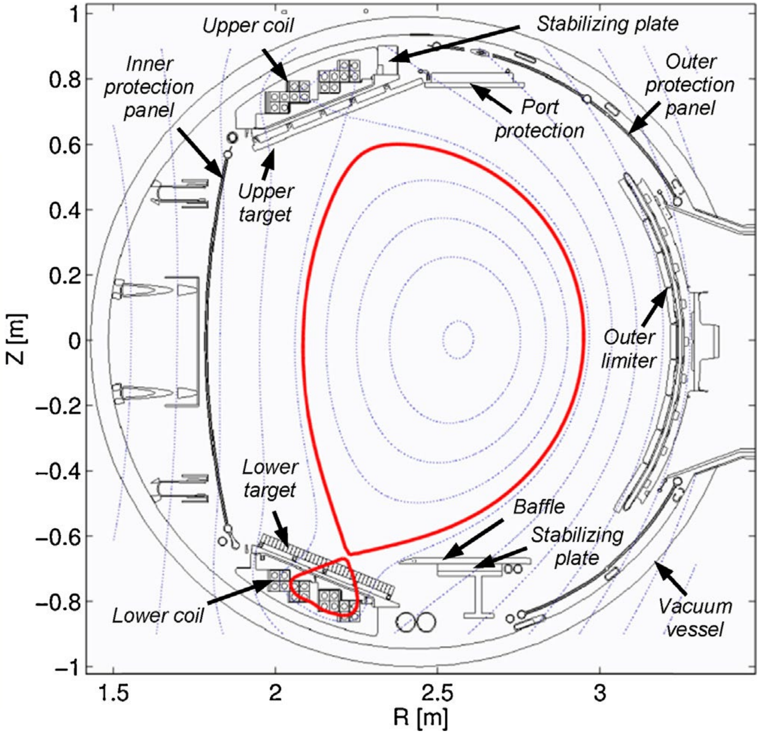


Fig. 3: Poloidal cross-section of WEST tokamak with contours of magnetic flux and main inner components of the tokamak.

Fig. 3: Interior of WEST tokamak with baffle slabs (Photo courtesy of IRFM CEA)

Thermal model

ITER – FWP 4

a) b)

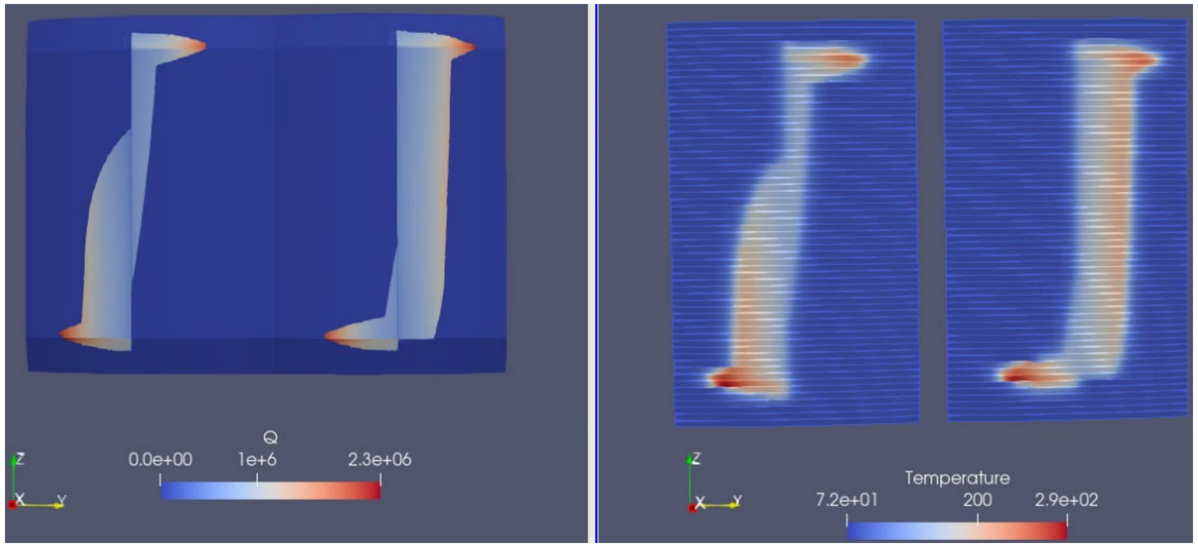
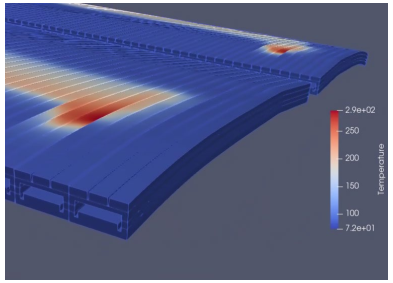
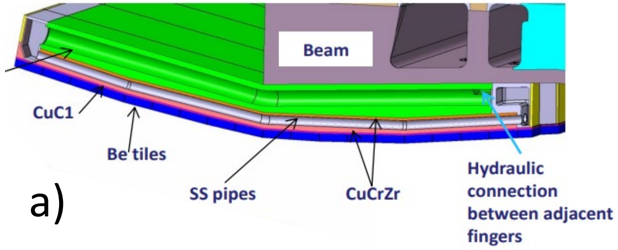


Fig. 1: (a) Heat fluxes (b) Corresponding temperatures (OpenFOAM): 1 panel ~1400k tetrahedrons

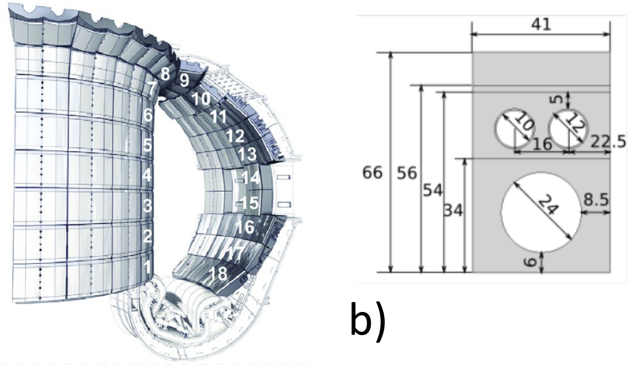


ITER first wall

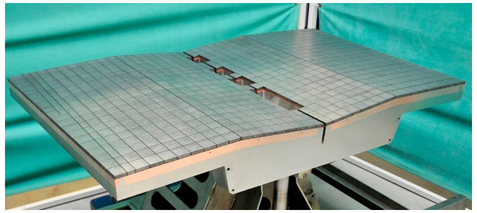
Fig. 2: (a) cross-section of ITER first wall (b) corresponding dimensions of ITER frst wall (c) 1 panel



a)



b)

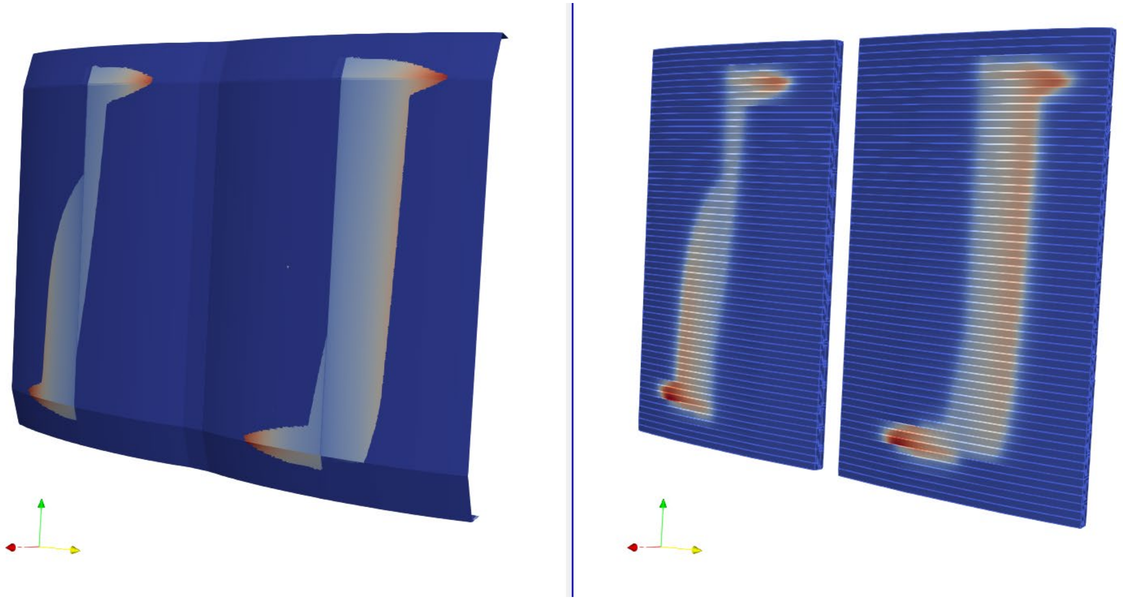


In total over 300 panels!

Thermal model

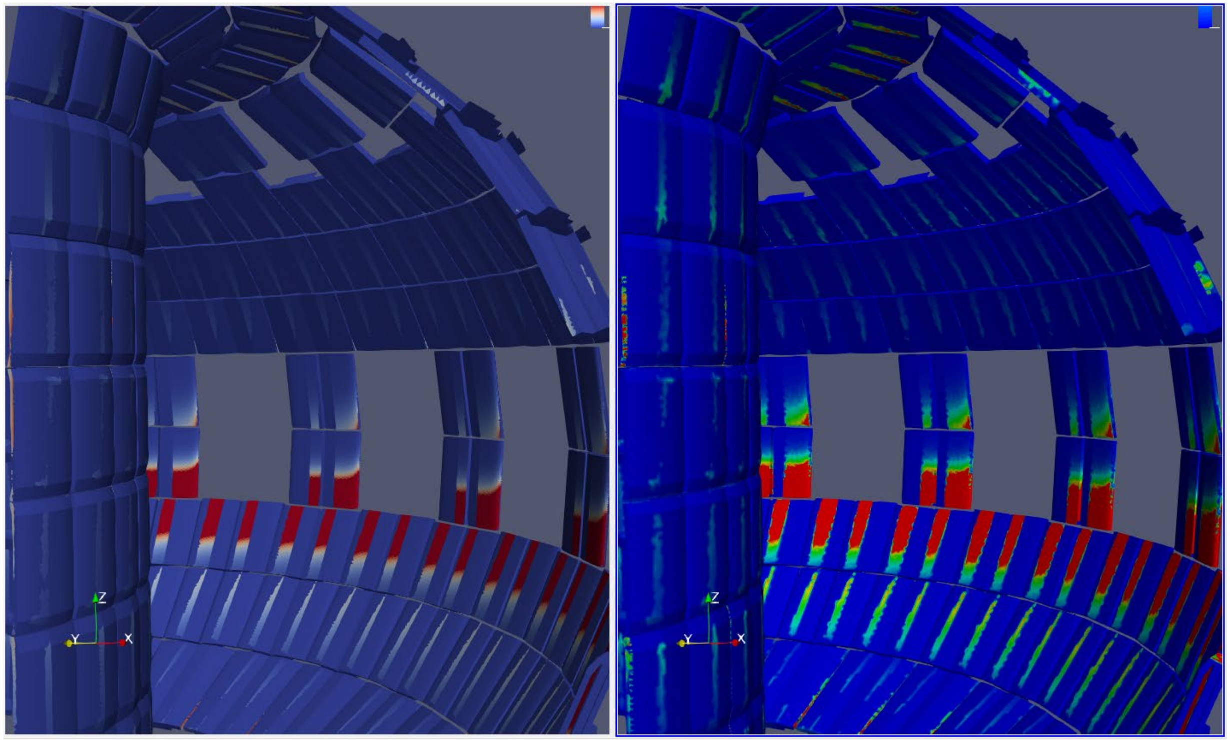
2D model
(field line tracing)

3D model
(heat transfer)



Heat flux map

Corresponding temperatures



Ray tracing – optical simulations

- Take into account full 3D field of distributed temperatures and calculate corresponding power/radiance on the IR cameras/bolometers in fusion reactors

ITER temporary limiter

- Taking into account the camera resolution 640 x 512
- FoV = 21,7° X 17,5°
- Pixel size =15 µm

Codes:
Raysect (Cython - 1 node, multiple cores), Mitsuba (c++ - GPU), Embree, Pov-ray, and many others, …

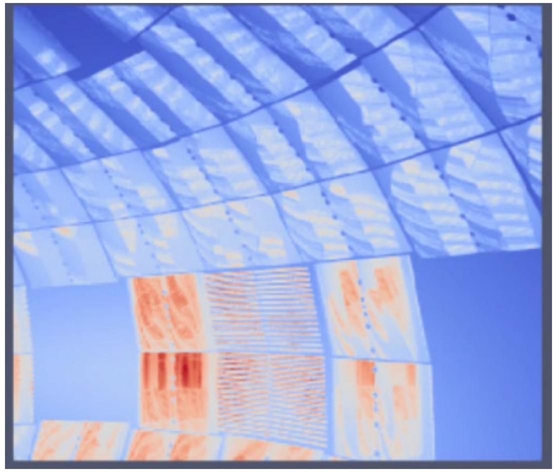


Fig. 2:Example of camera view in the ITER tokamak

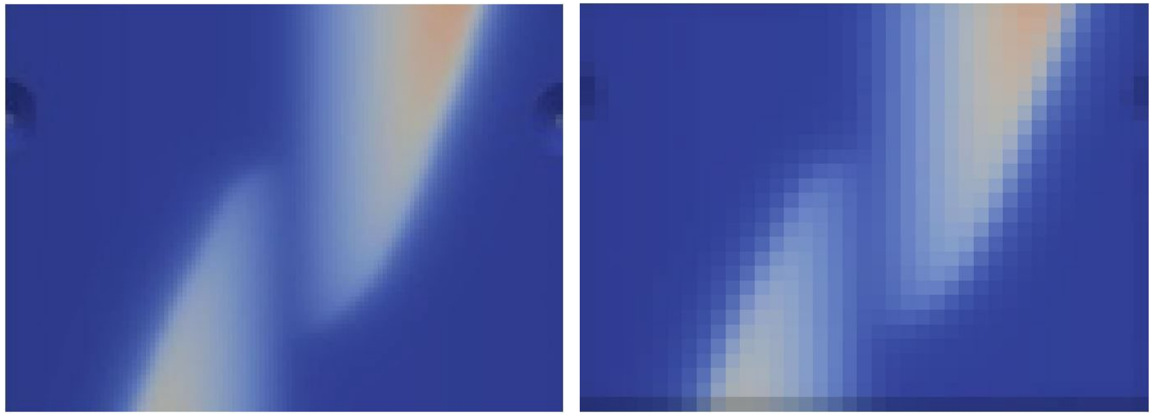


Fig 1: Temperature result from simulation (left) and result as seen from camera (right) for a given first plasma scenario.

Full workflow – ITER temporary limiter

Small scale run: Temporary limiter

Field line tracing (FLT)
(Following the magnetic field lines in the tokamak)

Thermal model
(calculate temperatures from heat fluxes)

Optical simulations
(calculate camera signal)

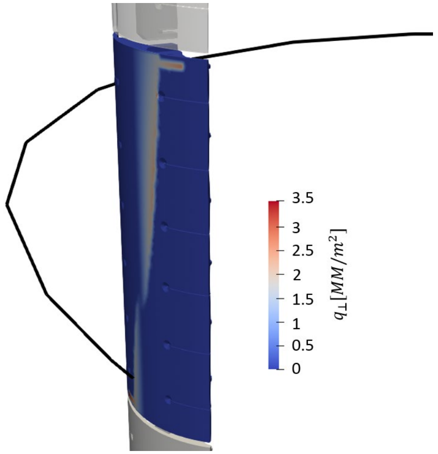


Fig. 1: Heat fluxes on baffle tile

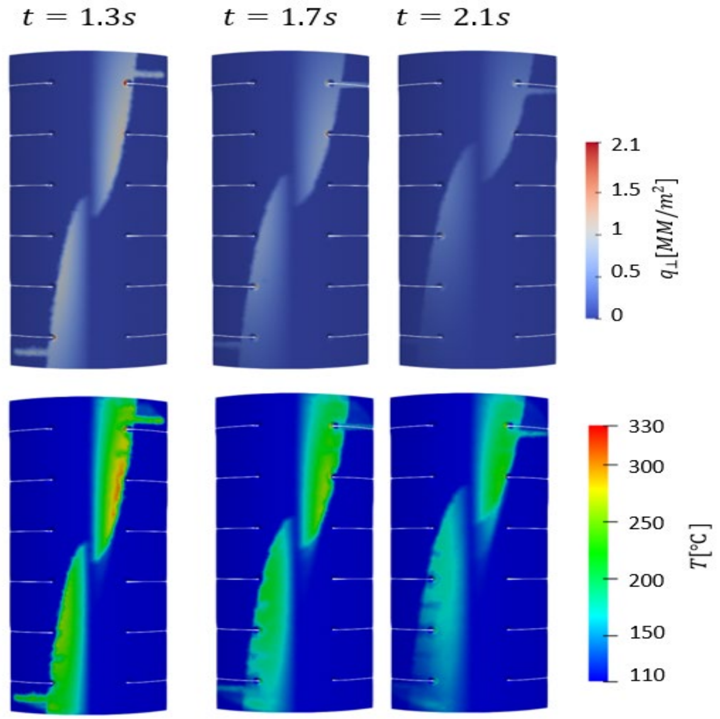


Fig. 2: (top) heat flux (bottom) temperature

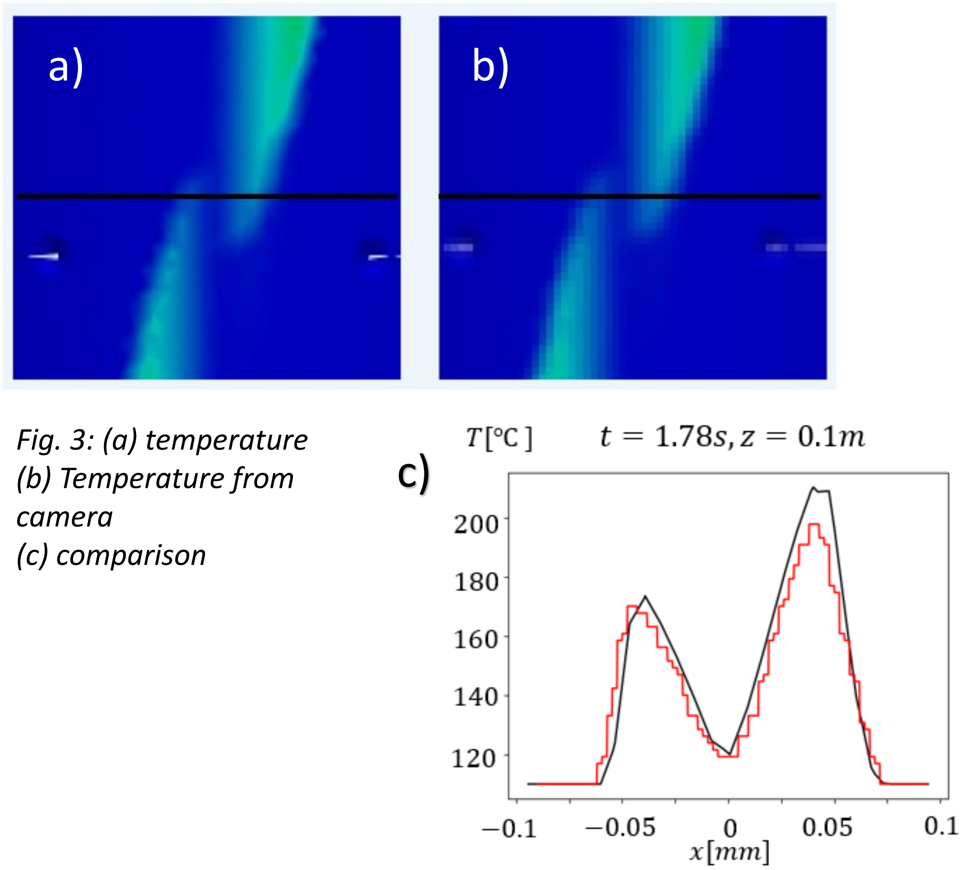


Fig. 3: (a) temperature
(b) Temperature from camera
(c) comparison

Scalability

- Field line tracing/Ray tracing*
  - ITER temporary limiter
    - 18 x 18 segments
    - 1 segment cca 300k elements
    - Total 21 million elements
  - WEST
    - Currently full mesh with 5 million elements, however density will probably need to be increased
  - ITER first wall
    - 18 x 18 panels
    - 1 panel cca 270 k triangles
    - Total ~87 million elements

* Estimation is that for ray tracing, test case will be smaller in terms of elements but similar in terms of computational time

- Thermal modelling
  - ITER temporary limiter
    - 18 x 18 segments
    - 1 segment cca 300k tetrahedrons
    - Total 21 million elements
  - WEST
    - Not yet known, but it will be between ITER temporary limiter and first wall
  - ITER first wall
    - 18 x 18 panels
    - 1 panel cca 1.5 million tetrahedrons
    - Total ~486 million elements

Output generation

Scalability of digital twin of processes in fusion reactor:

- Test case is given for integration of 3D magnetic field to calculate stream flow of particles
- Thermal modelling of large 3D structures taking into accout time and space dependent heat fluxes from plasma
- Optical simulations (ray-tracing) of millions of rays to generate camera output based on surface radiation and reflections

Synthetic diagnostic strives towards more detailed simulation setups and more efficient workflow execution to acquire simulation results that will better explain physical processes around the inner wall during the fusion reactions and improved workflow execution that will drastically shorten time to solution of this computations and therefore pave the path towards real-time control of fusion reaction, which is needed for stable fusion.

Ray-tracing

- Algorithms for simulating the propagation of light. Light is modelled as a bundle of rays which travel through the scene. Rays follow a straight line until they intersect with the objects in the scene. Based on the object material and optical properties, their contribution to emission is added to the scene.

- A computer rendering technique for generating an image by tracing the path of light as pixels in a space of interest and simulation of the effects of its encounters with virtual objects. In terms of physical simulations, ray tracing is capable of simulating a variety of optical effects, such as reflection and refraction, scattering, and dispersion phenomena (such as chromatic aberration).

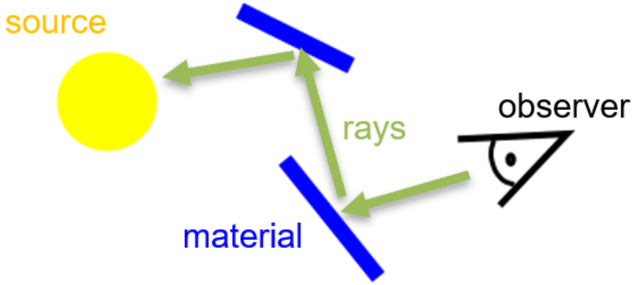


Fig. 1: Simple scheme of backwards ray tracing. Ray is generated on the observer and traced backwards into space.

Ray-tracing

- Simulation of variety of optical effects
- A light source emits a ray of light which travels, eventually, to a surface that interrupts its progress. One can think of this "ray" as a stream of photons traveling along the same path. In a perfect vacuum this ray will be a straight line (ignoring relativistic effects). Any combination of four things might happen with this light ray: absorption, reflection, refraction and fluorescence.
- Technique for generating an image by tracing the path of light through the space of interest and simulation of encounters with virtual objects
- For a number of generated rays a Monte Carlo integral estimator is used to solve the lightning equation for targeted surfaces (observers)
- Reflections are simulated through bidirectional reflectance distribution function (BRDF)

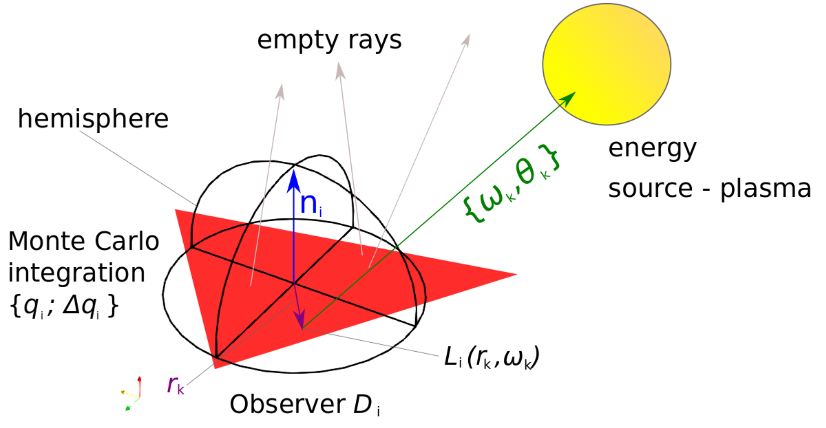


Fig.1 : Scheme of ray tracing simulation and ray interception with energy source.

Rendering equation

$$P = \int_A \int_\Omega \int L_i(x,\omega,\lambda)\cos(\theta)\, d\lambda d\omega dA$$

L_i … incident radiance arriving at point x
ω … incident angle on the surface
A … surface
Ω … solid angle

Ray-tracing

Bidirectional reflectance distribution function

Weighting function f_r that describes redistribution of light into outgoing reflections and absorption into the material

Two ideal limits of BRDF:

- Specular reflection
  - Behaves like mirroring surface (perfect reflection into one specular angle)
- Diffuse/Lambertian reflection
  - Even distribution of incident light into outgoing reflections

In reality, BRDF is a combination of both limits.

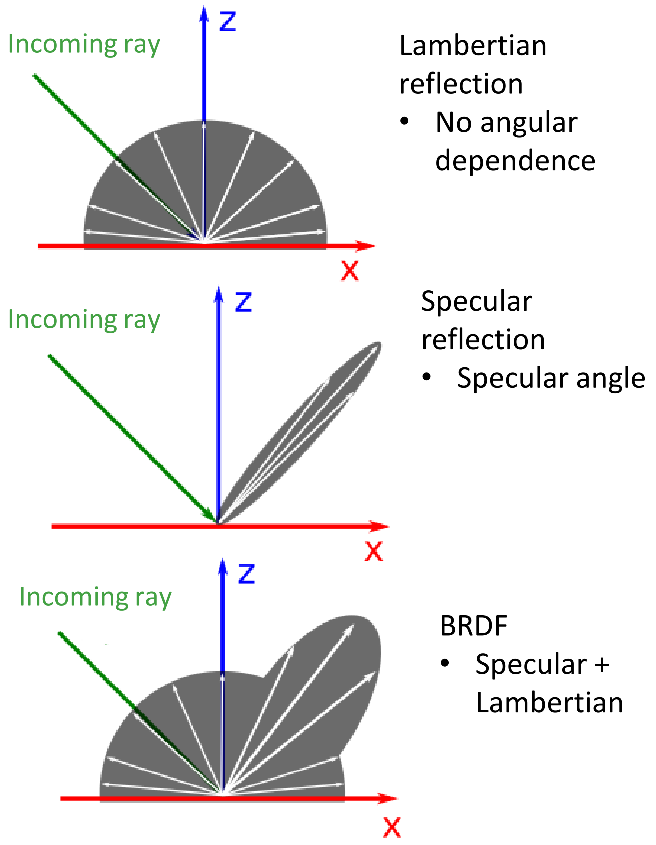


Fig. 1: Reflections for BRDF modelling.

Ray-tracing

Bidirectional reflectance distribution function

How to correctly determine BRDF?

Thermal optical properties of materials (samples of tungsten and beryllium) need to be assessed experimentally, taking into account

- Temperature
- Wavelength
- Roughness
- Direction

Measurement with gonioreflectometers and spectrophotometry to assess reflectivity and emissivity of samples.

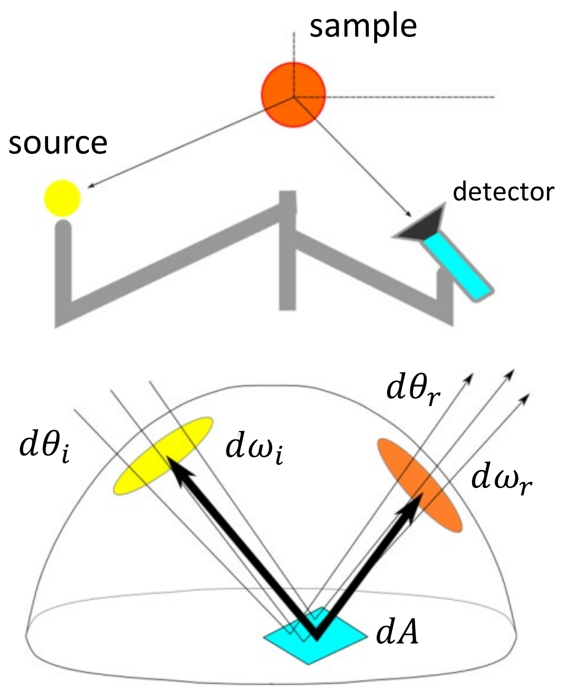


Fig. 1: Outline of gonioreflectometer, a device for measuring BRDF. Light source is illuminating the measured sample and a sensor captures light from the material.

Raysect/CHERAB

Raysect

Raysect is an open-source python framework for geometrical optical simulations. It is designed to be a physically realistic ray-tracer generally applicable to research problems in science and engineering. A core philosophy at the heart of Raysect's design is that scientific robustness and flexibility takes precedence over speed. Raysect has been designed to be easy to extend by a physicist/engineer.

Raysect feature set:

- Path tracer for efficient configurable ray-tracing.
- Full scenegraph for managing complex geometry and coordinate transforms.
- A complete set of geometric primitives, lens types, meshes and CSG operations.
- A wide range of simulated physical observers such as CCDs, cameras, fibreoptics, etc.

Raysect/CHERAB

- Advanced optical material models, an associated material library (BRDFs), metals, glasses.
- Supports serial or multi-core operation. MPI not currently supported.

Structure/Architecture

- Raysect is an OOP framework written in a combination of Python and Cython. All major functionality is accessible from python. It is possible to extend all components from python, however to gain full speed, the Cython API should be used.
- The core of Raysect is general and can be used for other ray-tracing applications such as neutron transport, etc.
- The core of Raysect is a generalised kernel for calculating interactions with rays and or volumes onto which physics models that require ray-tracing (such as geometric optics) can be built.

Raysect/CHERAB

Cherab

A python library for forward modelling diagnostics based on spectroscopic plasma emission.

Cherab is a large code framework consisting of a main core package and many smaller feature packages. The overall structure of the framework is best visualised as onion shells. The full project structure is most relevant for researchers in fusion plasma physics.

Ray-tracing Engine: Cherab was built on top of a ray-tracing engine to provide all the core ray-tracing functionality, mesh handling, etc. Raysect is currently the only ray-tracing engine supported, but the framework has been designed such that this component is interchangeable. Support for other ray-tracers may be added in the future.

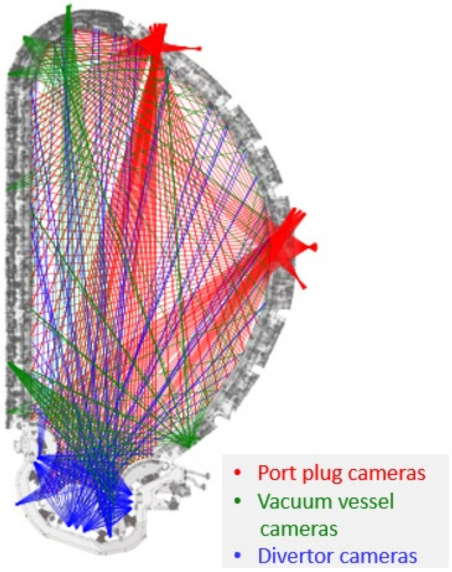
Raysect/CHERAB

Core API Module: The cherab package defines all the API interfaces for the whole framework. It defines all the core functionality such as: how plasmas are defined and the properties that can be computed from them; the types of atomic data that can be used in spectroscopic calculations; and the types of plasma emission that Cherab can calculate. This package is strictly managed by the Cherab development team.

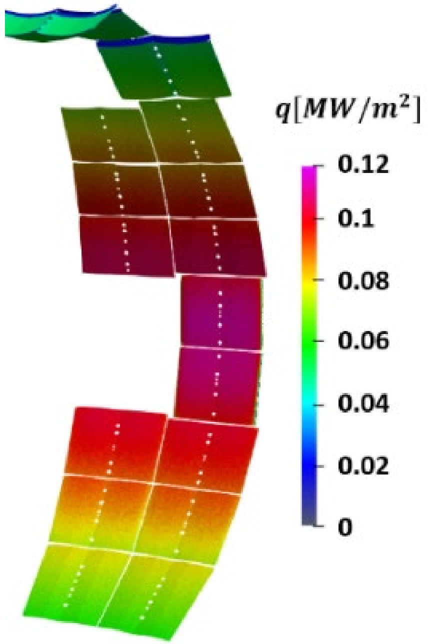
Machine Specific Packages: These components are for functionality associated with a specific fusion experiment. For example, each experiment tends to have its own systems for loading and saving experiment/simulation data. These packages allow the local facility experts to provide models and diagnostic settings that are unique to their experiments.

What can be accomplished with Raysect?

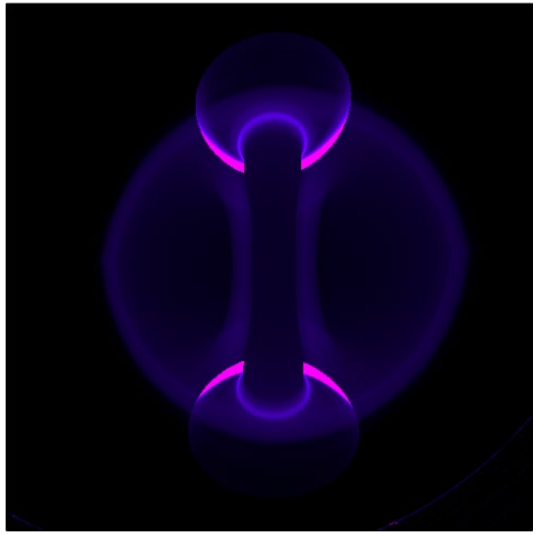
Bolometry



Radiation loads

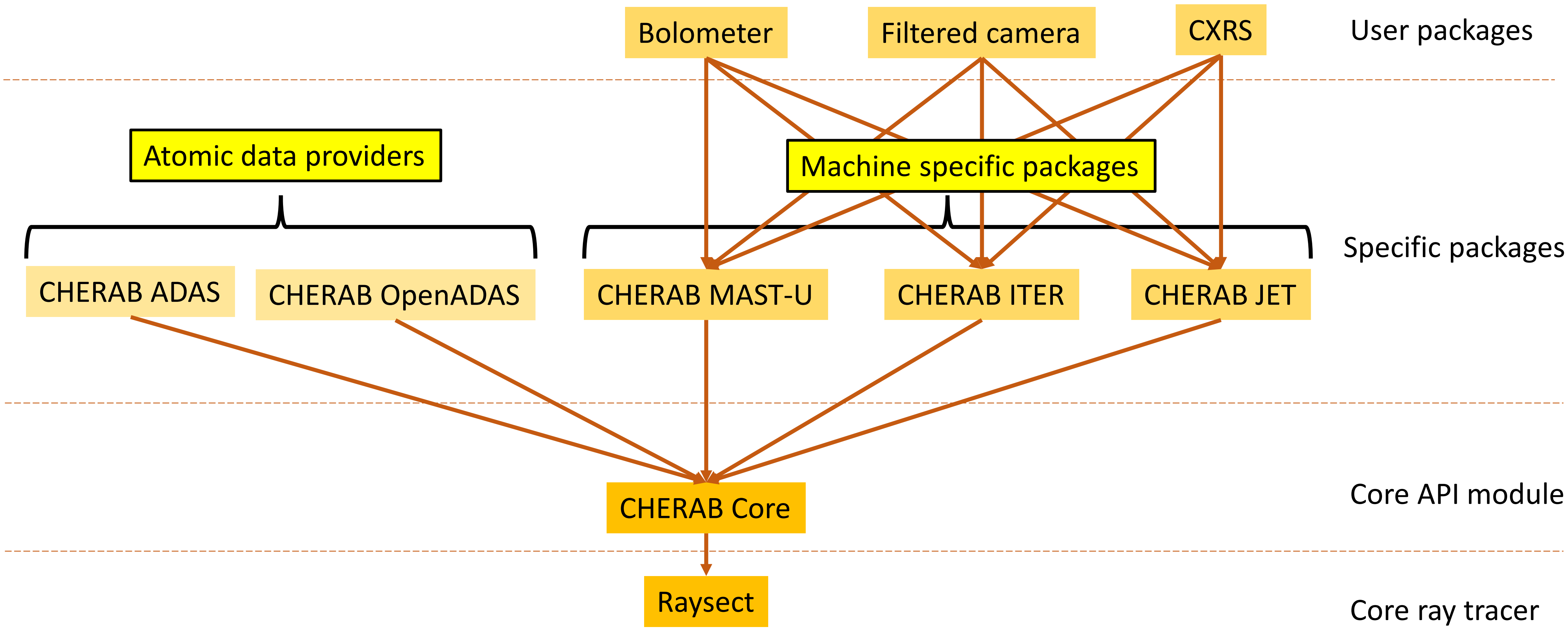


Filtered camera imaging



What can be accomplished with Raysect?

Overall code structure:



How to install Raysect/CHERAB

- Easiest way to install CHERAB is from the python package repository pypi through Pip. Unless you are developing a new emission model, this is the recommended installation path for new users.

```
$> pip install cherab cherab-openadas --user
```

- If you prefer to install it from source download the source repositories and install with setup.py.

```
$> python setup.py install --user
```

- If you are a developer we recommend using pythons develop environment. This installs the package with symbolic links to your development directory.

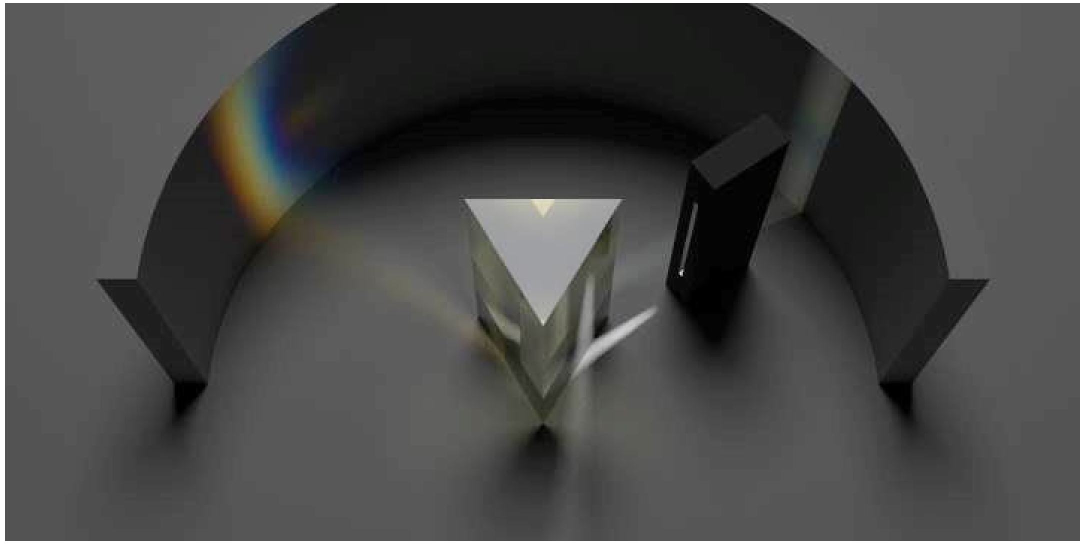
```
$> python setup.py develop --user
```

- **Make sure you configure some atomic data!!!**
  To download the default data set:

```
>>> from cherab.openadas.repository import populate
>>> populate()
```

Raysect

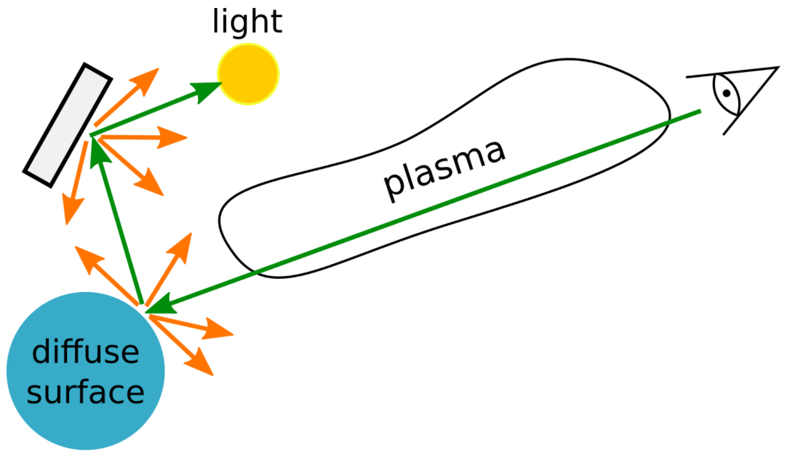
- Fully spectral, supports arbitrary wavelength resolution when paired with appropriate emission models.
- Scientifically accurate reflections from materials through physically principled BRDFs such as the Cook-Torrance BRDF.
- Support for a number of CAD geometry formats.
- Powerful observer framework, from simple visualisation with pinhole cameras through to real CCD sensor and optic geometry. Also support non-imaging optics such as fibre optics and power collection at surfaces.



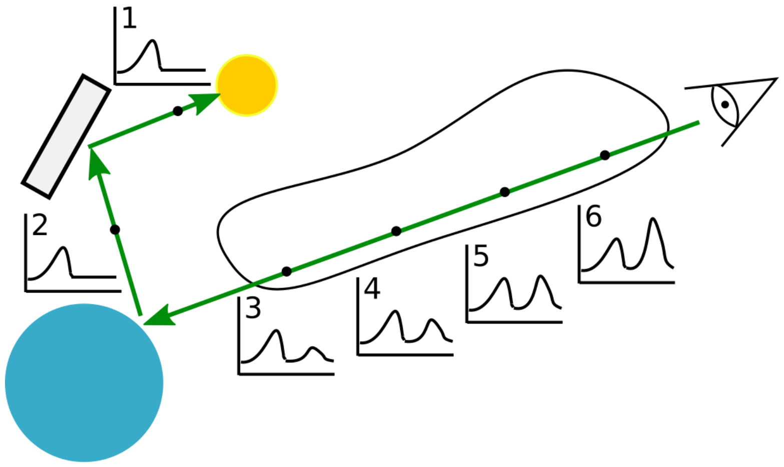
Simulation of a camera looking at the classic prism dispersion experiment

Raysect – work principle

1. Calculate a path by recursively tracing rays until termination criteria reached. Terminations can be due to absorption or Russian roulette style ray-extinction.
2. Sample the emission back up the ray stack (forward direction). Each sample add its contribution (or subtracts if absorbing) to the ray's accumulated spectrum.



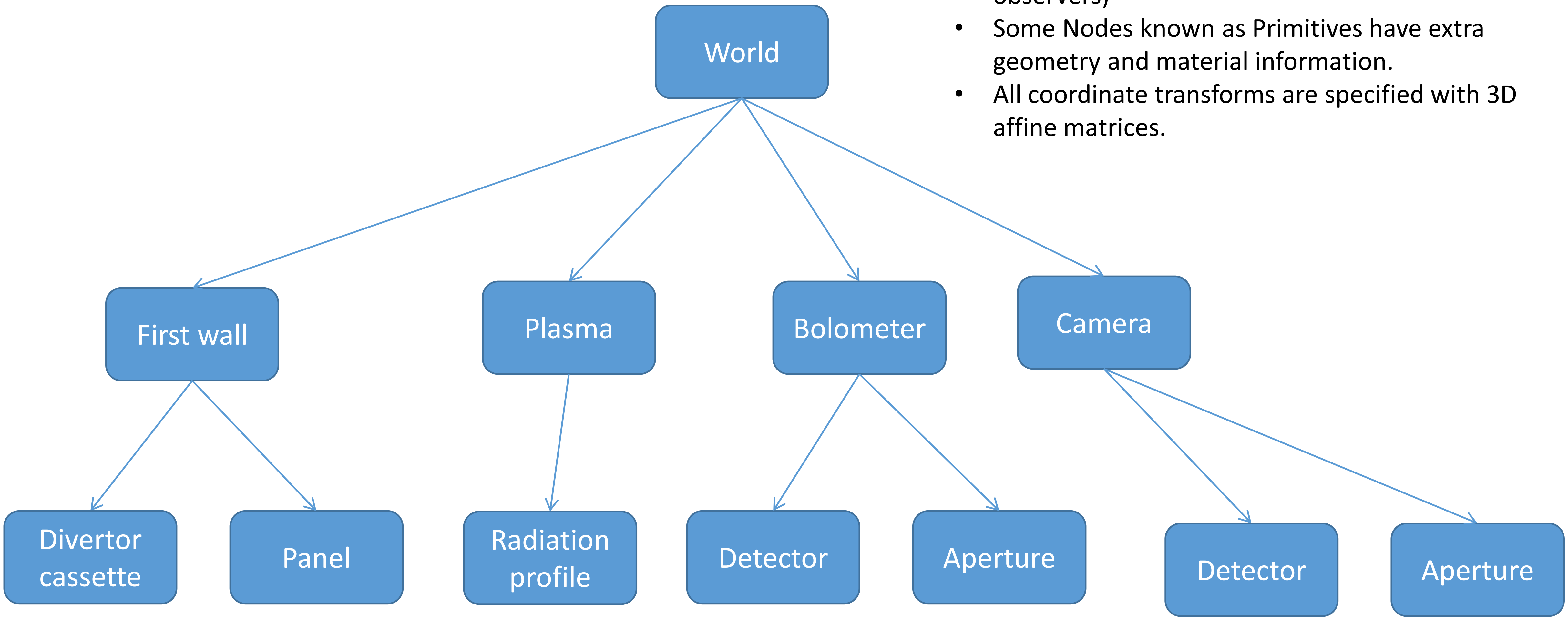
a) Generate paths (reverse tracing)



b) Integrate emission in forward direction

Raysect – Scene definition

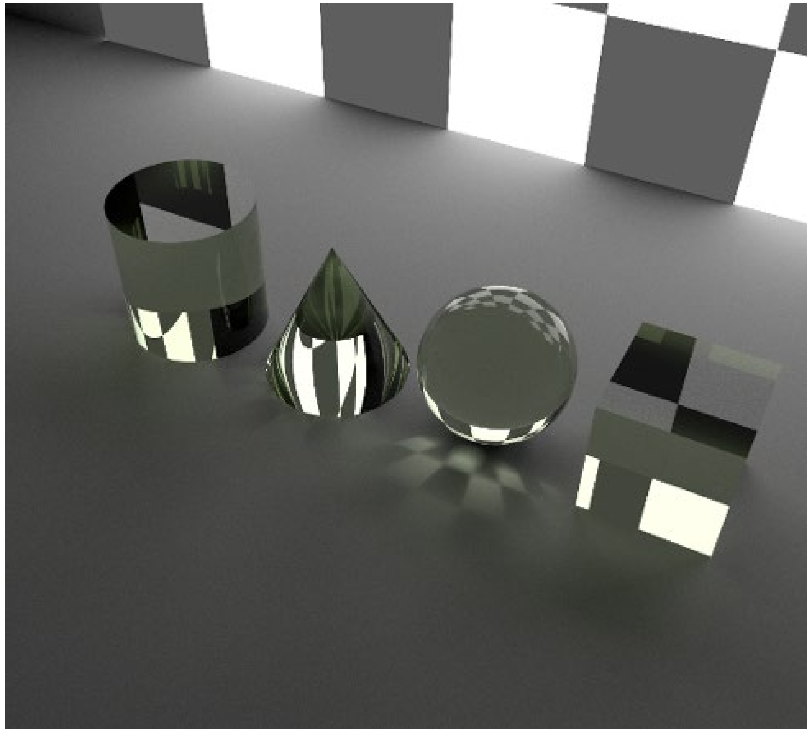
- All nodes have their own local coordinate system relative to their parent.
- Some Nodes are virtual (i.e. groups and observers)
- Some Nodes known as Primitives have extra geometry and material information.
- All coordinate transforms are specified with 3D affine matrices.



Raysect – Primitives

- Primitives are Nodes that define their geometric extent and material properties.

Geometric Primitives



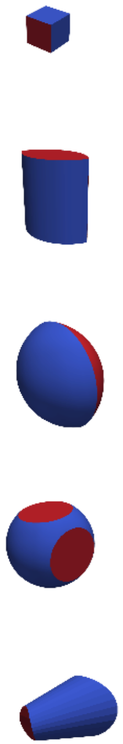
Mesh based Primitives

Raysect – Primitives

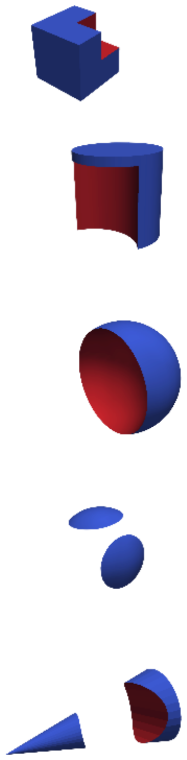
- Geometrical operations on primitives

Union

Intersection

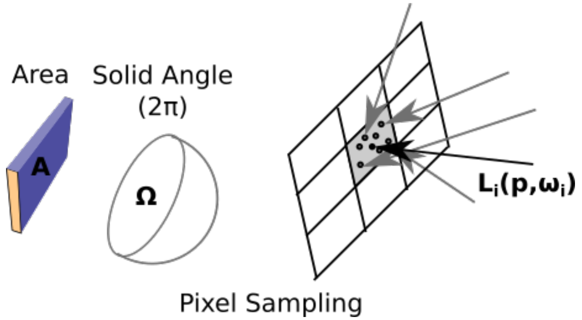


Difference



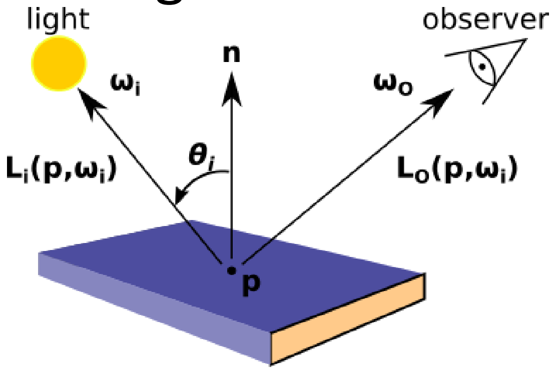
Raysect – Light transport equation (rendering equation)

- The total intensity measured by an observing pixel is given be the integral of the incident emission over the collecting solid angle and surface area.



$$I = \int_A \int_\Omega L_i(p, \omega_i) \times \cos(\theta_i) d\omega_i \, dA$$

- The amount of light that reaches a camera from a point on an object is given by the sum of light emitted and reflected from other sources.



$$L_0(p, \omega_0) = \int_\Omega L_i(p, \omega_i) \times f(\omega_i, \omega_0) \times \cos(\theta_i) d\omega_i$$

Outgoing emission

Incident emission

BRDF

Surface geometry

Raysect – Ideal BRDF materials

- Bidirectional reflectance distribution function (BRDF) is a weighting function that describes the redistribution of incident light into outgoing reflections and transmission/absorption.
- For ideal specular reflection (mirror) the incoming light is perfectly reflected into one angle.

$$f_s(\omega_i, \omega_0) = \rho_s(\omega_i)\delta(\omega_i, \omega_0)$$

specular reflectance delta function

- An ideal diffuse surface (matte paper) will evenly redistribute incident light across all directions.

$$f_d(\omega_i, \omega_0) = \frac{\rho_d}{\pi}$$

diffuse reflectance

- Ideal materials can be constructed by blending specular and diffuse reflection.

Raysect – Ideal Metals

- The Fresnel equations describe the amount of light reflected from a surface, they are the solution to Maxwell's equations for a smooth surface. Raysect uses simplified versions for dielectrics and metals assuming no polarisation.

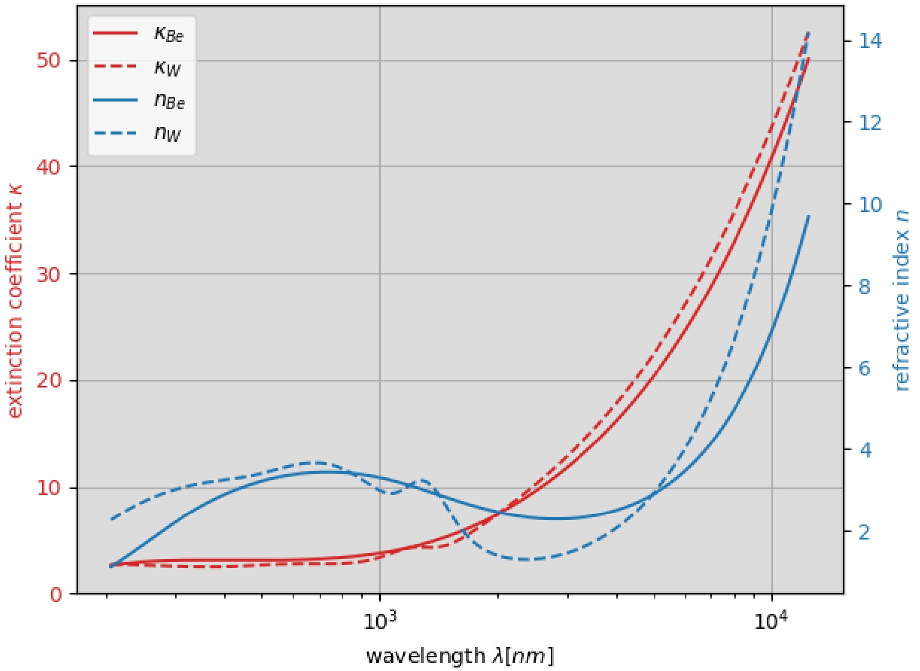
$$r_{\parallel}^2 = \frac{(n^2 + k^2)\cos\theta_i^2 - 2n\cos\theta_i + 1}{(n^2 + k^2)\cos\theta_i^2 + 2n\cos\theta_i + 1}$$

$$r_{\perp}^2 = \frac{(n^2 + k^2) - 2n\cos\theta_i + \cos\theta_i^2}{(n^2 + k^2) + 2n\cos\theta_i + \cos\theta_i^2}$$

$$F_r(\omega_O) = \frac{1}{2}(r_{\parallel}^2 + r_{\perp}^2)$$

Parallel polarisation

Perpendicular polarisation



Raysect –Rough surfaces

- Microfacet surface models are described by a function that gives the distribution of facet normals $\mathbf{n_f}$ with respect to the surface normal **n**.
- Increasing roughness increases the distribution of facet normals.

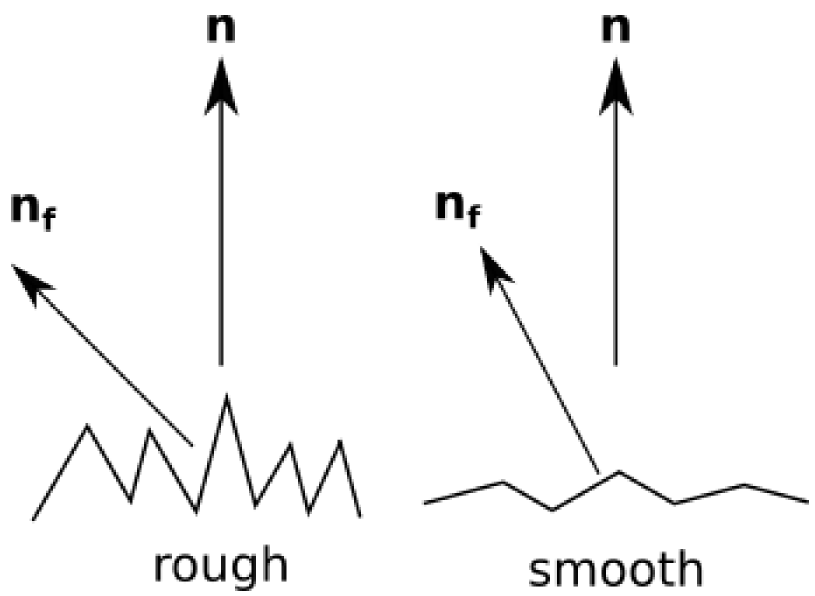
$$f_r(\omega_i, \omega_0) = \frac{F(\theta)}{\pi} * \frac{D(\theta_h)G(\omega_i, \omega_0)}{\cos(\omega_i)\cos(\omega_0)}$$

- $F(\theta)$ is the ideal Fresnel reflection factor for the entire surface.
- $D(\theta_h\ )$ is the microfacet distribution.

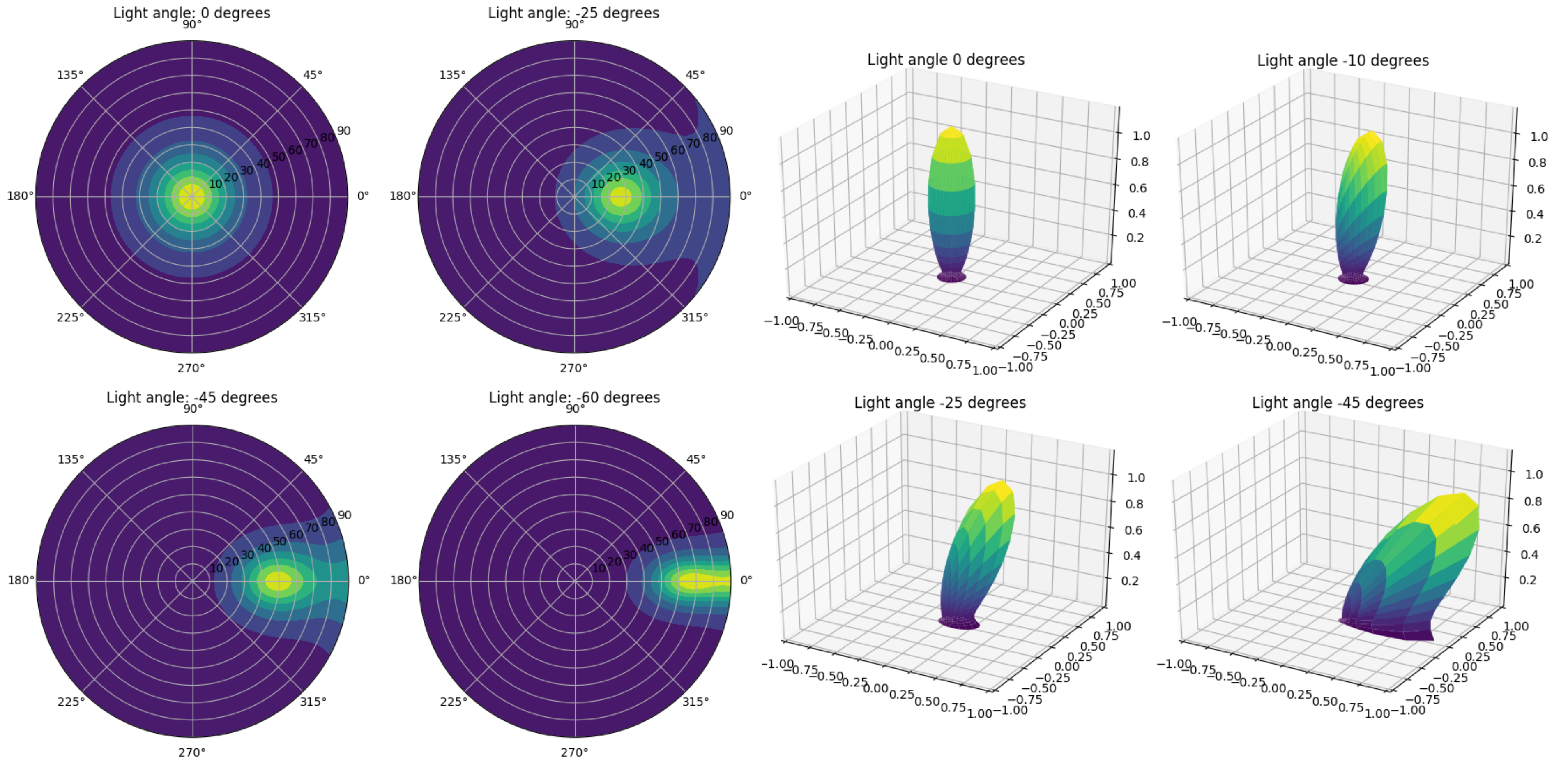
$$D(\theta_h) = \frac{1}{m^2\cos(\delta^2)} exp\left(\frac{\cos(\delta^2) - 1}{m^2\cos(\delta^2)}\right)$$

- $G(\omega_i,\omega_0)$ is a geometric shadowing factor, it expresses the ratio of light that is occluded due to masking/shadowing.

$$G(\omega_i, \omega_0) = min\left\{1, \frac{2(n \cdot \theta_h)(n \cdot \omega_0)}{(\theta_h \cdot \omega_0)}, \frac{2(n \cdot \theta_h)(n \cdot \omega_i)}{(\theta_h \cdot \omega_0)}\right\}$$



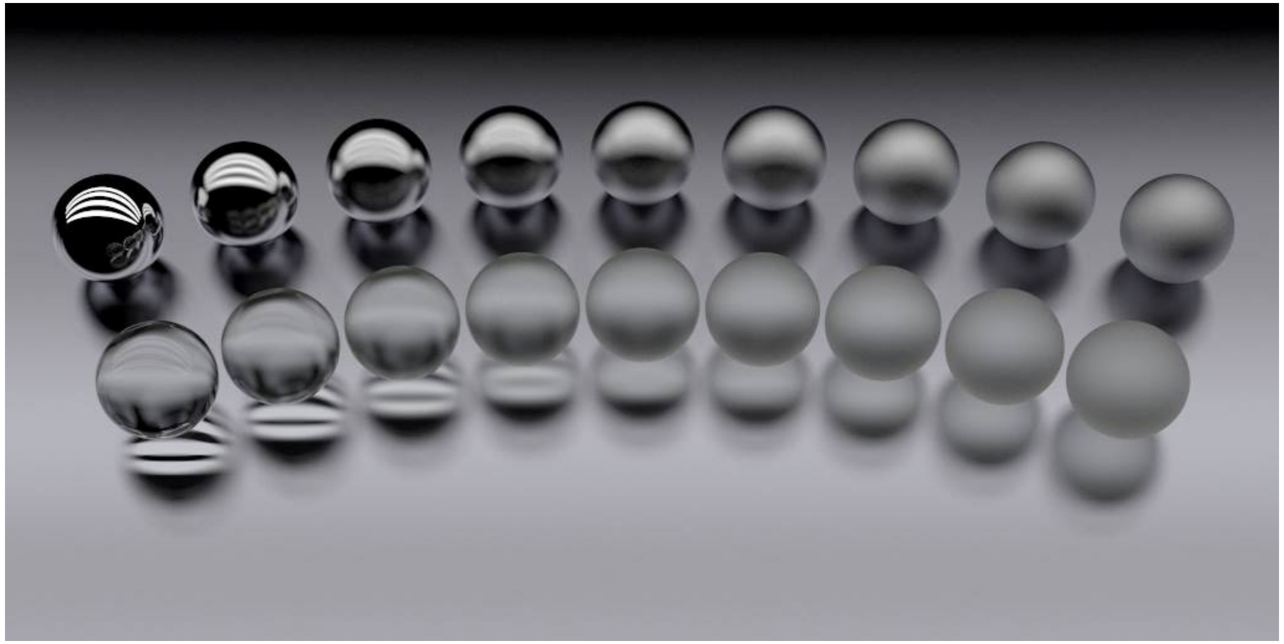
Sampling Raysect BRDFs



Rough surfaces

metal : n, k

glass : n, k



smooth ↔ rough

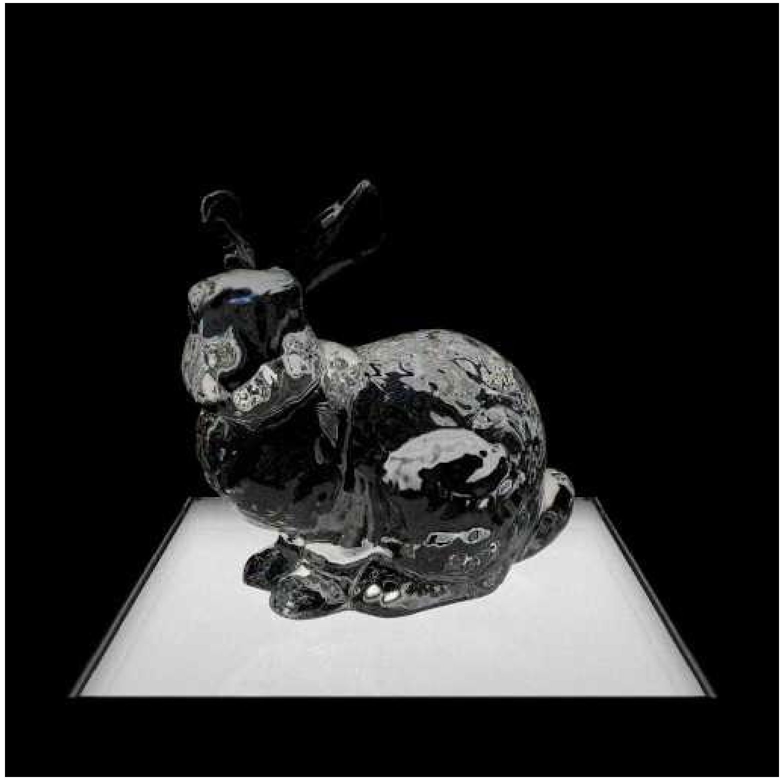
- The material colour and absorption are defined through n and k as a function of wavelength.
- The roughness is an additional tuneable parameter.

Raysect examples gallery

- Materials can have surface properties, volume properties or both.

Bunny mesh with gold metal

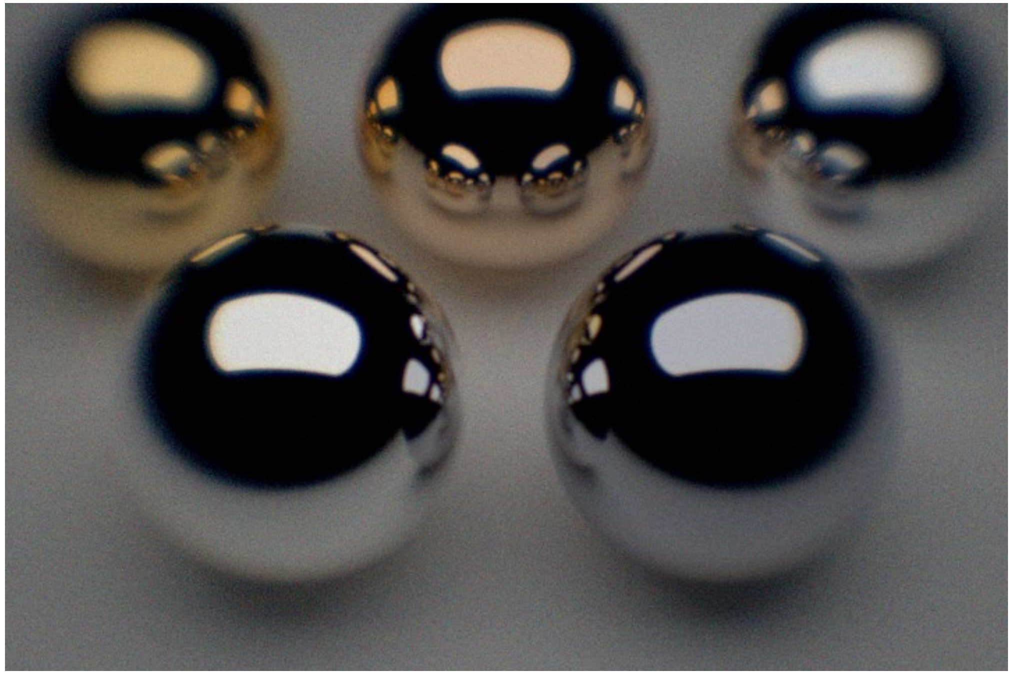
Bunny mesh with glass



Raysect examples gallery

Raysect examples gallery

Starting with an ideal pinhole camera

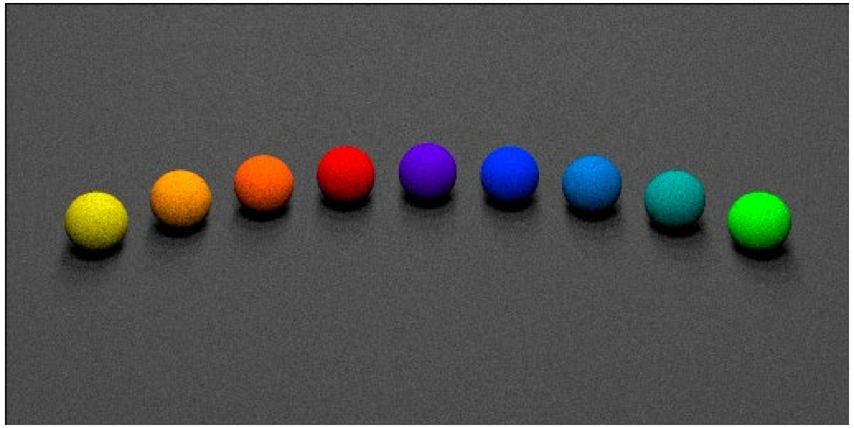


Adding some lenses and playing with the focus

Key Raysect concepts: Primitives -> Materials

emissive materials

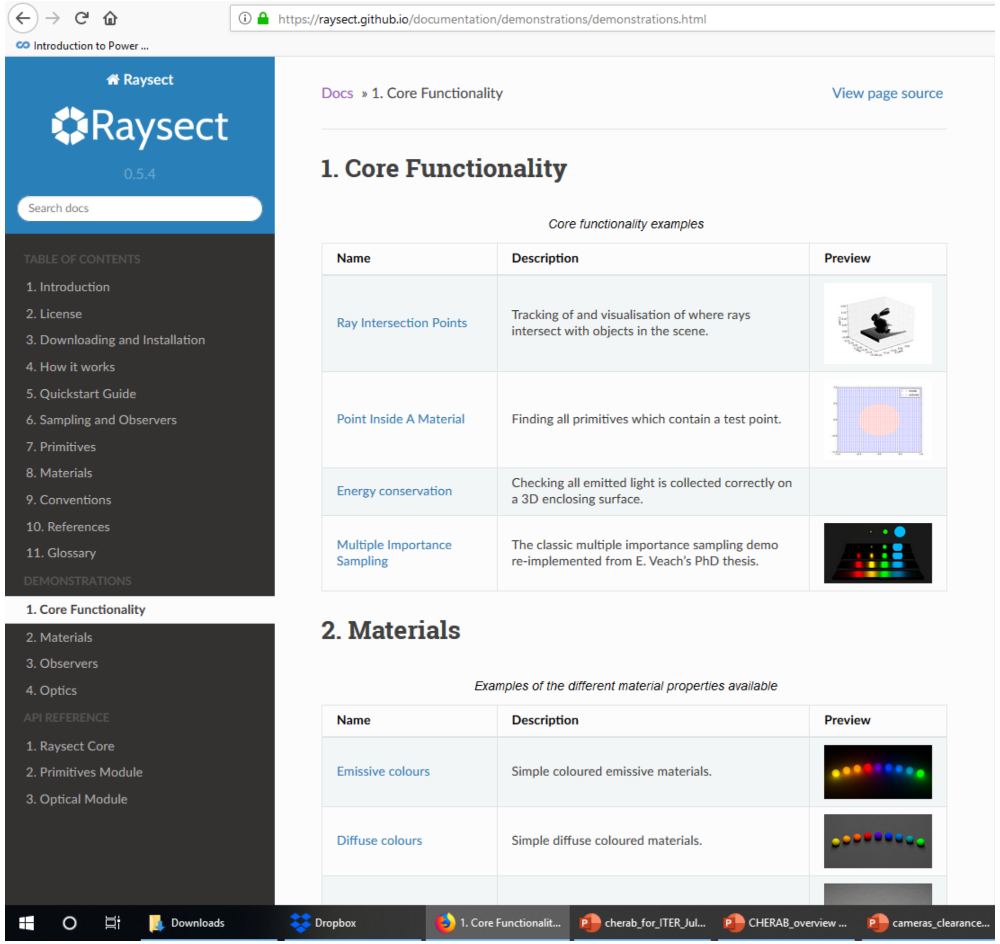
diffuse materials



etc.

Raysect Documentation

www.raysect.org


https://raysect.github.io/documentation/demonstrations/demonstrations.html

Introduction to Power ...

Raysect

Raysect

0.5.4

Search docs

TABLE OF CONTENTS

1. Introduction
2. License
3. Downloading and Installation
4. How it works
5. Quickstart Guide
6. Sampling and Observers
7. Primitives
8. Materials
9. Conventions
10. References
11. Glossary

DEMONSTRATIONS

1. Core Functionality
2. Materials
3. Observers
4. Optics

API REFERENCE

1. Raysect Core
2. Primitives Module
3. Optical Module

Docs » 1. Core Functionality

View page source

1. Core Functionality

Core functionality examples

Name	Description	Preview
Ray Intersection Points	Tracking of and visualisation of where rays intersect with objects in the scene.	
Point Inside A Material	Finding all primitives which contain a test point.	
Energy conservation	Checking all emitted light is collected correctly on a 3D enclosing surface.	
Multiple Importance Sampling	The classic multiple importance sampling demo re-implemented from E. Veach's PhD thesis.	

2. Materials

Examples of the different material properties available

Name	Description	Preview
Emissive colours	Simple coloured emissive materials.	
Diffuse colours	Simple diffuse coloured materials.	

Downloads | Dropbox | 1. Core Functionalit... | cherab_for_ITER_Jul... | CHERAB_overview ... | cameras_clearance...

CHERAB

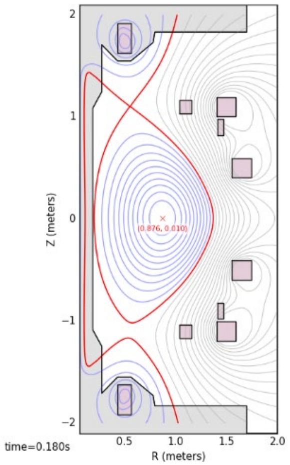
Plasma definition: Through Distribution Functions

- Plasmas are composed of Species objects, which identify an ion, its ionisation stage and 6D distribution function.
- Every property of the Plasma is accessed through a distribution function. This design pattern means we don't care how a plasma was actually defined.

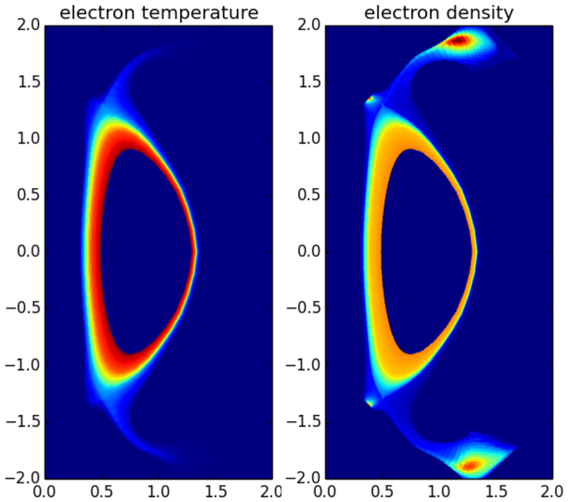
analytic function

$T_i(x, y, z) = x^2$

1D flux function



2D mesh interpolation



EURO

SLING

REPUBLIC OF SLOVENIA
MINISTRY OF HIGHER EDUCATION, SCIENCE AND INNOVATION

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 101101903. The JU receives support from the Digital Europe Programme and Germany, Bulgaria, Austria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, Greece, Hungary, Ireland, Italy, Lithuania, Latvia, Poland, Portugal, Romania, Slovenia, Spain, Sweden, France, Netherlands, Belgium, Luxembourg, Slovakia, Norway, Türkiye, Republic of North Macedonia, Iceland, Montenegro, Serbia.