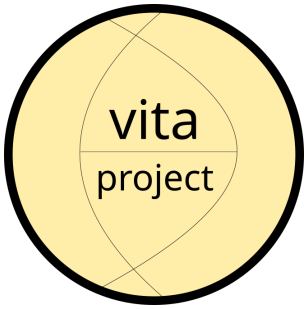
Vitaproject

Release 0.1

Dec 15, 2020

Contents:


vita
project

CHAPTER 1

Welcome

Welcome to Vita! The Vitaproject is part of a comprehensive tool-set for divertor modeling.

1.1 Need Help?

Please post a question on the github issue queue.

1.1.1 Description

Introduction and functionality

Vitaproject is part of a comprehensive tool-set for divertor modeling. The function and objectives of each tool, as shown in Fig. 1.1, vary depending on the operating phase. Vitaproject aims to deliver the following functionality:

Design, assessment and preparation for operations: The objective of the modeling in this stage is to have a good estimate of the plasma loading effect, in order to assess compliance to the Design Criteria and to define the Operating Limits. The uncertainty at this stage is large so a large number of sensitivity analysis shall be required in order to establish the range of acceptable plasma parameters, as well as the risk of non-compliance.

Pulse monitoring: validated 2D nonlinear diffusion models are used for real-time temperature estimation. This synthetic diagnostic complements other protection measures such as thermography and thermocouple measurements.

Post-pulse processing: A Virtual Thermal Map (VTM) may be used based on processing protection IR camera data. A backup for recreating the surface and bulk temperatures shall be provided through quick analysis of hidden components, cross checking the VTM in case of dubious hotspots, and overall analysis in the case of IR camera malfunctioning.

Sensitivity, condition and change request assessment: any change on divertor components may be checked to actual experimental conditions, in order to evaluate the impact of any deviation from nominal geometry and properties as well as to assess major design modifications. This stage differs from the first one in that the model will have been already validated under nominal conditions and the workflow uses experimental data.

The workflow implemented bridges the Physics design, operational database and the engineering design and assessment.

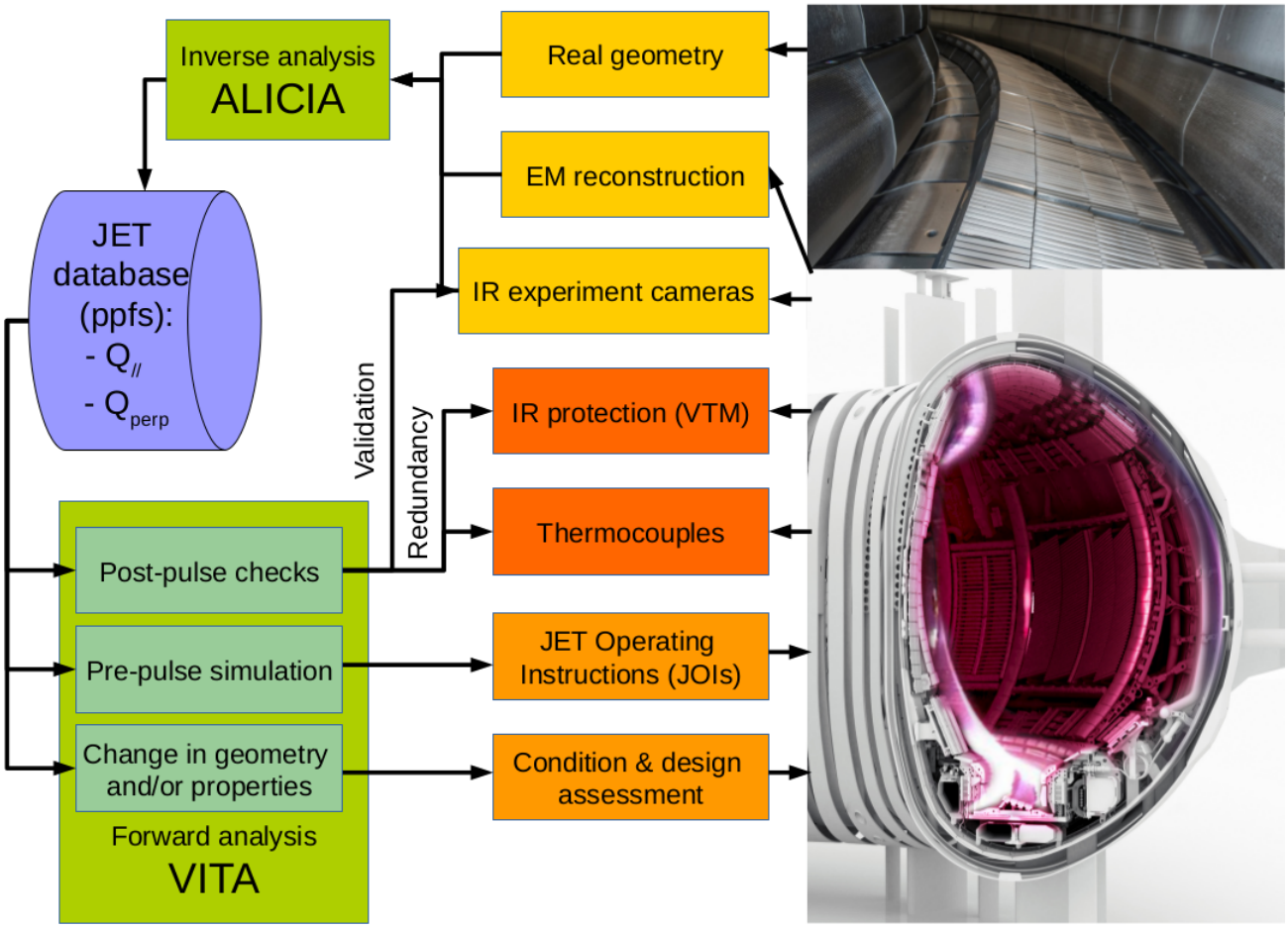


Fig. 1.1: Overall workflow scheme

Description

Virtual Thermal Assessments is a forward simulation code featuring a GUI for ease of use. Its main goal is to allow both quick and accurate analysis of divertor tiles to users by setting global machine parameters, recreating previous stored pulses, or a mix of both. The time varying boundary conditions and integration parameters are automatically set, therefore not requiring the user to deal with numerical details.

It is designed for design, pulse preparation activities, post-pulse checks, and integrity assessments of in-vessel components. It may also be used to test alternative divertor configurations under experimental conditions. It includes the following capabilities:

- Several ways for defining pulse parameters and automatically setting the simulation model and its boundary conditions.
- Connection to the experimental database for the readout of diagnostic measurements, typically temperatures.
- Selection of the wall segment or divertor tile with different accuracy on the thermal model.
- Direct plotting of diagnostic synthetic signals.

- Tabulated output of maximum temperature at the surface and thermocouple measurement locations, along with energy values.

The parameters that define the pulse can be grouped as follows:

Input power parameters: The total power input to the plasma arrives from either resistive heating, NBI or RF sources. Each of the three signals can be defined as a constant value or a table from a file allowing complex manual load inputs. In the case where an experimental pulse is to be recreated, each of these values can be read from their corresponding signal in the JET database.

Plasma parameters: The total power arriving to the divertor at any moment in time corresponds to the total minus the radiated power. This is taken into account as a factor in the range $[0 - 1]$ called the radiated fraction. The outboard-inboard power ratio is typically estimated in single null configurations as $1/3$ inboard, $2/3$ outboard, and $1/10$ inboard, $9/10$ outboard in double null configurations. The footprint can be defined using different functions: - A pure exponential function is the simplest way of defining the shape of the SOL power density around the plasma.

When information about the far-SOL is known, a double exponential function may be used. Only the falloff length is needed for defining the footprint, allowing for a rough estimation of the power footprint at any PFC surface.

- A square distribution may be used for fast transients simplified modeling of limited plasmas.
- The convolution of an exponential with a Gaussian has been proven in [**?**] to be the best fit to the experimental observations for diverted plasma configurations. This function defines the profile of the scrape-off layer (SOL) at the equatorial plane. The parameters defining this function correspond to the power fall-off width, λ, and the spreading factor, S. Their values can be manually fixed or estimated—as defined in [**?**]—as a function of the plasma current, I_p, toroidal field, B_t, integrated density, n_e, SOL power, P_{SOL}, ELM frequency, f_{ELM}, and the standard deviation of the radial field current, σ_{RF}.

Magnetic parameters: In the latter case, the power density needs to be projected from the equatorial to the divertor plane. By default the flux expansion is used, but an option is available for performing a 3D magnetic projection using the magnetic field components and the equilibrium reconstruction provided by the Flush code [**?**] at each calculation time step. A second option allows the magnetic shadowing of the surrounding tiles to be taken into account.

The strike point position can be defined manually as a fixed location, or a regular sweep across it. It is also possible to input its evolution as a table or read it directly from an stored signal in the experimental database.

Analysis parameters: Once the physical quantities which define the loading conditions have been set, the Diritchlet boundary conditions are automatically defined in the model. The power density footprint is combined with the strike point time evolution, defining the power at each boundary point. The use of analytical functions for the heat flux profile allows calculating the exact power density at every surface node in an energy consistent manner (i.e. eliminating interpolation errors). In addition, the application of meshfree C^∞ shape functions greatly increases the accuracy of surface temperature simulation. In the case where the loading parameters have been manually specified, the duration of the heating stage can be defined by the pulse time. Finally, the total simulation time is input using the analysis duration parameter.

The accuracy of VITA has been tested to experimental data with satisfactory results. Fig. 1.2 compares the response of two H-mode medium and high power pulses with the IR camera signal used for experiment data analysis, which is much more accurate than the ones used for the protection of the JET-ILW [**?**]. Due to the large number of signals used for recreating the loading conditions, there is of course an overall associated uncertainty. The total error has been bounded to 10% of the measured temperatures, being comparable to the mismatch observed between the machine protection and experimental camera systems. The differences in amplitude during the sweeping of the strike point position is mostly due to the IR being measured in a tile extension instead of the full length tile. This short extension has a local shadow which amplifies the temperature oscillations. During the upcoming campaign, a normal length tile will be diagnosed. This will allow the specific testing of VITA against the alarms of the protection system. As the oscillation of the IR will be reduced, and the alarms are set to trigger when 200ms overheating events are detected [**?**]—in line with the response time of VITA models—, lower errors are expected.

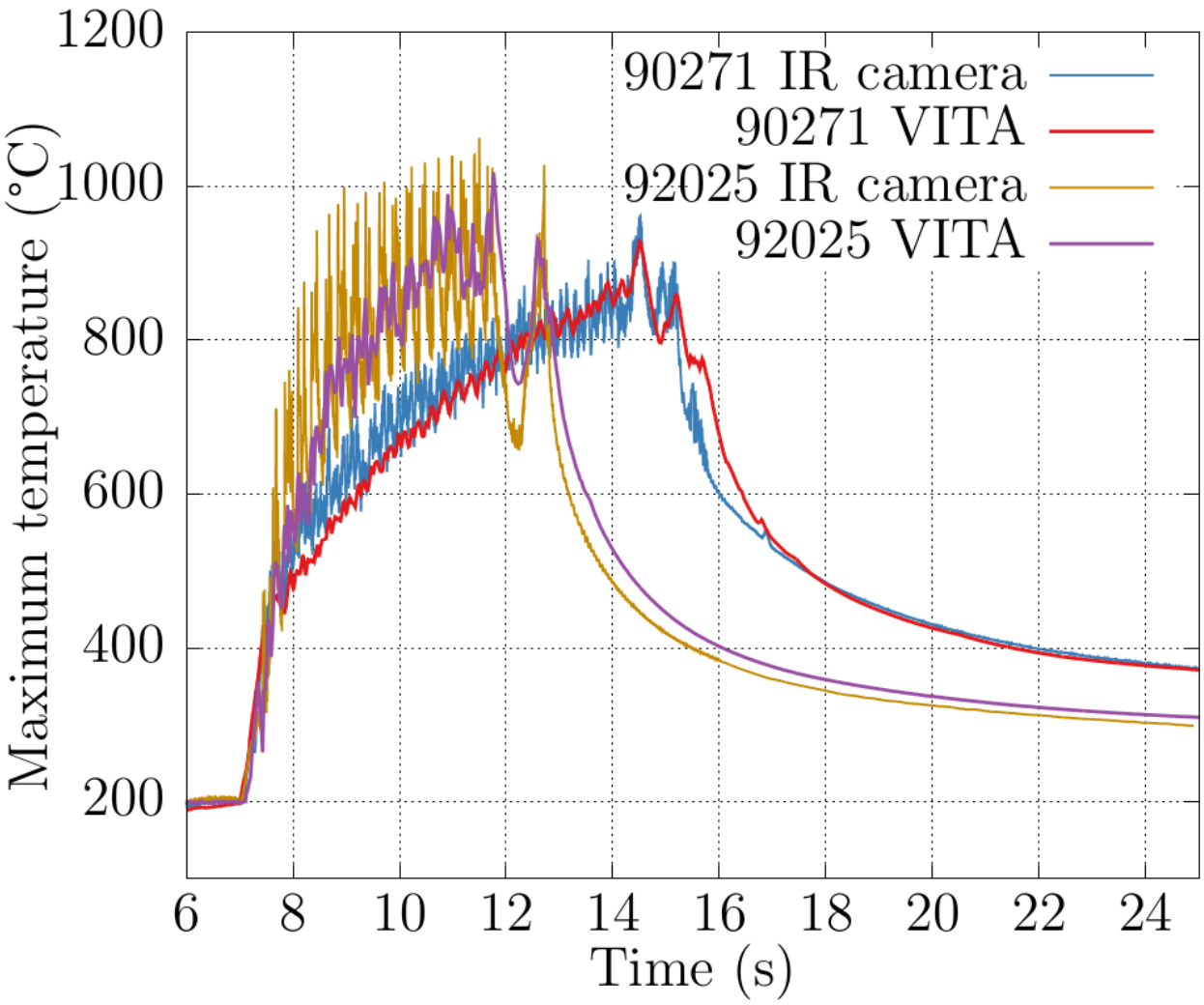


Fig. 1.2: VITA synthetic reconstruction of maximum temperature IR signal compared to experiment IR camera measurement for two H-mode pulses with medium (90271), and high power (92025) input power.

Input data

Equilibrium

Static equilibrium reading them from equilibrium files in FIESTA or EQDSK formats.

Sweeping applies a displacement to the heat load along the divertor target.

Multiple equilibria uses several input files for defining a transient plasma load.

Plasma Parameters

coming soon.

1.1.2 Theory

Objectives, formulation and models

In order to provide the functionality needed, each of the tools tackles one specific phase. As opposed to a typical analysis workflow, the main objective is maximizing the final user's productivity. Their design therefore hides any numerical complexity, and allows their operation using machine and experimental parameters. Several models are provided as a black-box, which is previously validated by analyst experts, but its source code can be inspected, audited, and extended at any time.

The formulation used for this first implementation is based on the thermal equilibrium using the Principle of Virtual Power. The contributions to the power virtual variation, $\delta\dot{\Pi}$, are calculated from the numerical integration of the following residual equation:

$$\delta\dot{\Pi} = \delta\dot{\Pi}_{capacitance} - \delta\dot{\Pi}_{external} - \delta\dot{\Pi}_{conduction} = 0$$

Each of the previous contribution terms can be expressed in the reference configuration [**?**] as:

$$\delta\dot{\Pi}_{capacitance} = \int_{\mathcal{B}} \rho c_p \frac{dT}{dt} \delta T \, dV$$

$$\delta\dot{\Pi}_{external} = \int_{\partial B} \mathbf{q}\delta T \cdot \mathbf{n} \, dS$$

$$\delta\dot{\Pi}_{conduction} = \int_{\mathcal{B}} (\kappa \nabla T) \cdot \nabla \delta T \, dV$$

Where the conductivity tensor κ and the specific heat capacity c_p, are temperature dependent, $f(T)$, properties of the material. The density ρ is considered constant.

Fully nonlinear Finite Element (FE) approximations are used for all analyses, with some Galerkin meshfree enhancements [**?**] when applicable. Several de-featuring levels are applied when speed is a concern. Initial implementation uses 2D models shown in Fig. 1.3, but design is extensible to 3D in the future. Orthotropic effects, as well as Planck radiation or convection cooling are also foreseen.

Coatings and deposits can be modelled with exact properties, by means of a proper layer formulation which is available for all the applications. Usual parameters for the JET divertor tiles range from 10 - 20 um thickness for the W coating on CFC tiles, to 50 um node separation in direction normal to the surface for modelling ELMs accurately in bulk W tiles.

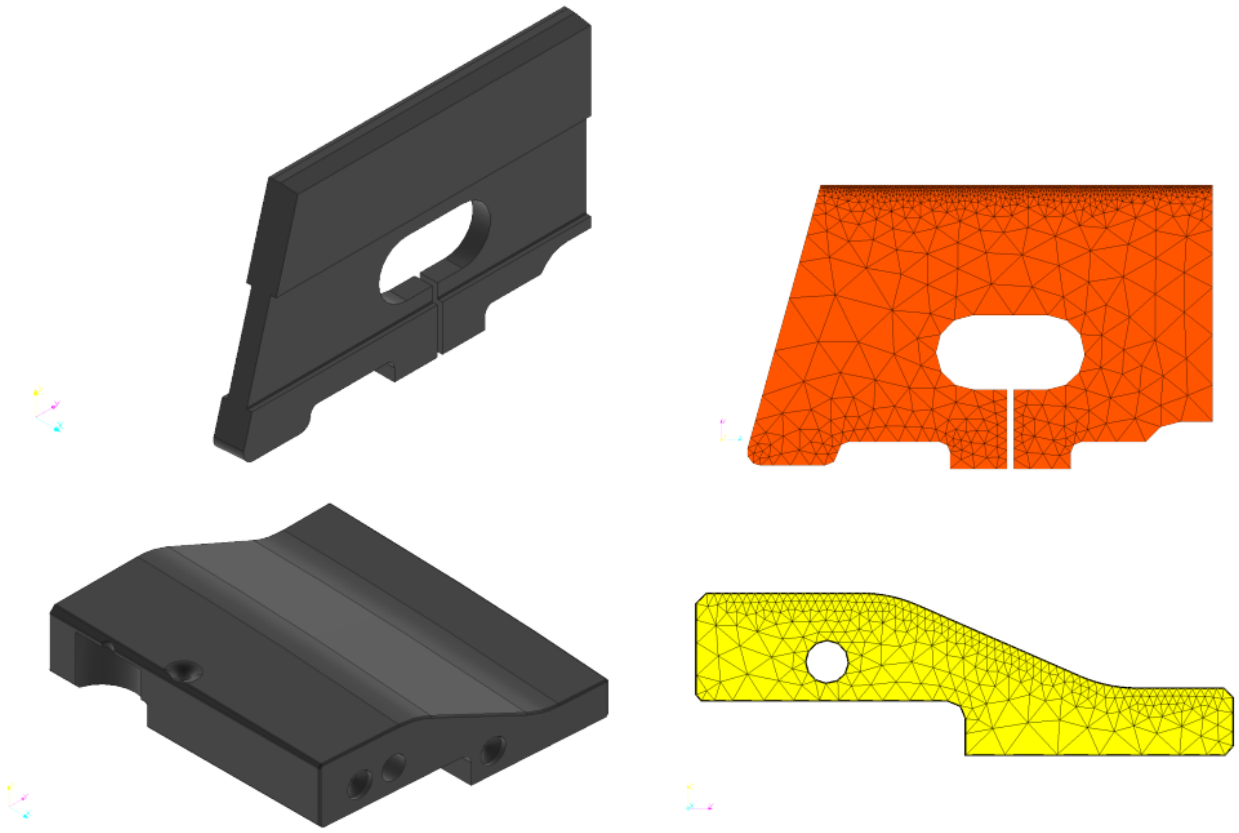


Fig. 1.3: 3D CAD (left) and 2D numerical discretization (right) of divertor components: Tile 5 (top) and tile 6 (bottom).

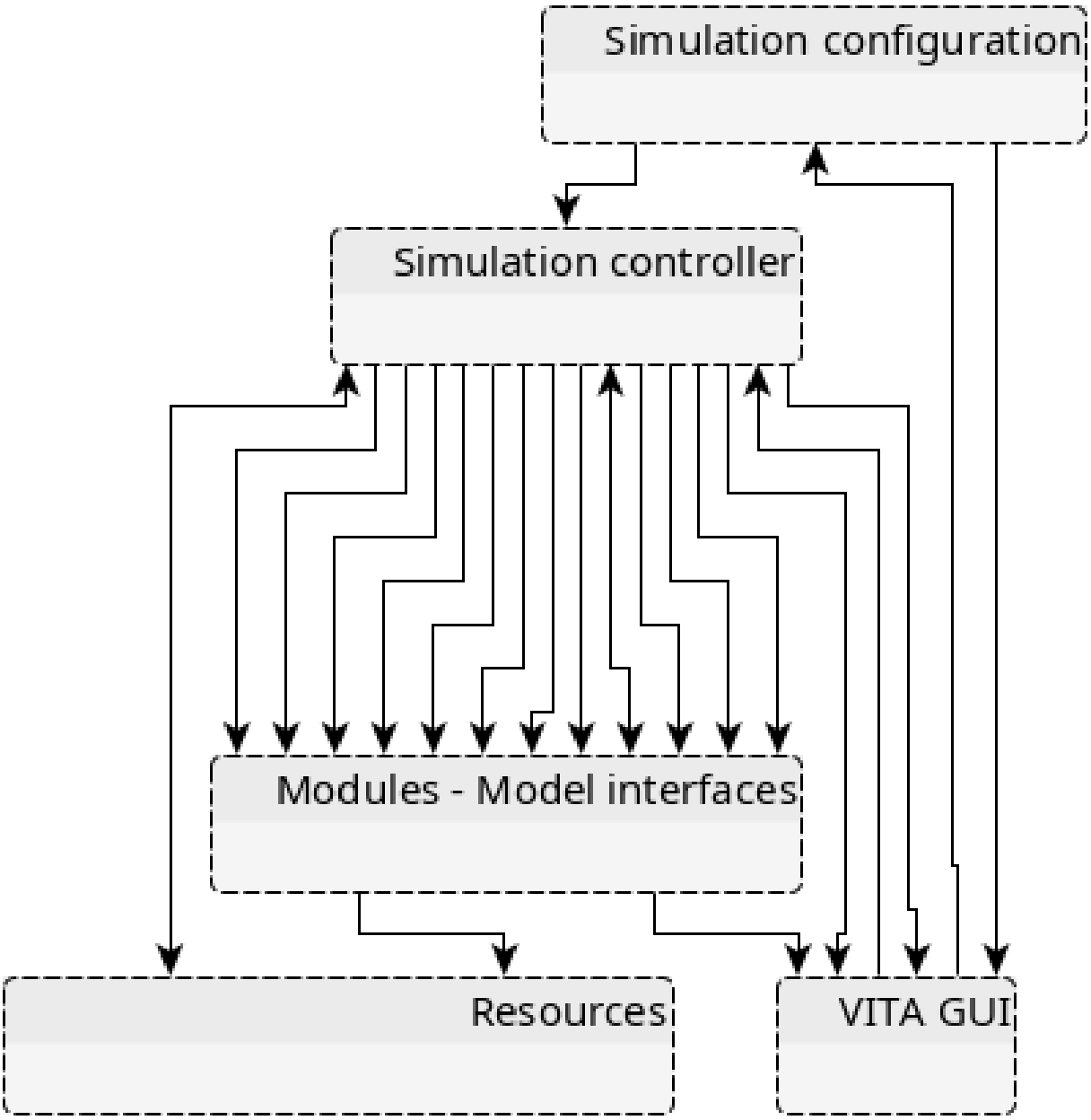


Fig. 1.4: Top level scheme of Vitaproject.

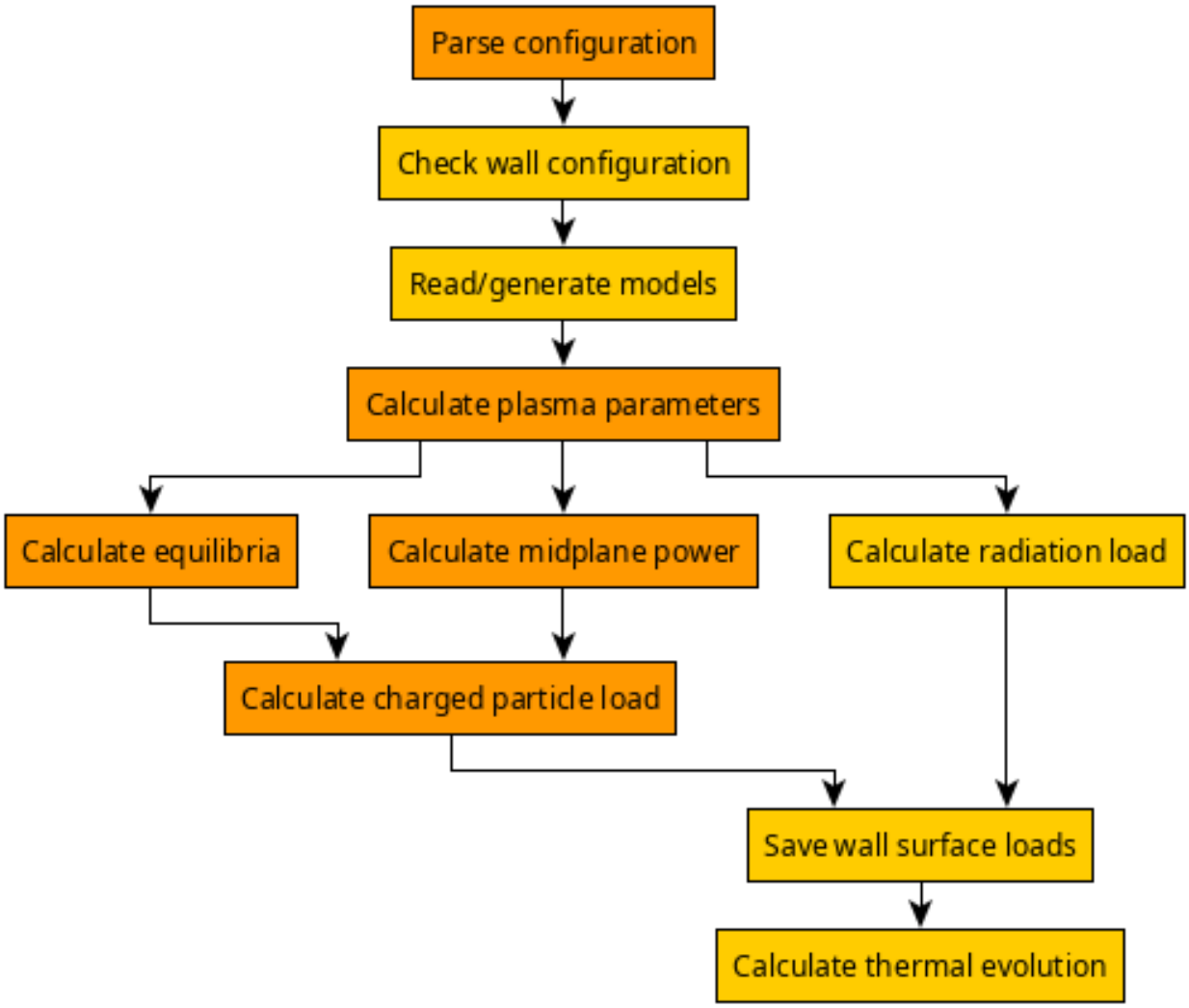


Fig. 1.5: Scheme Vitaproject simulation controller.

1.1.3 Workflow

Coming soon.

1.1.4 Demonstrations

1.1.5 Equilibrium

Fiesta

class vita.modules.equilibrium.fiesta.**Fiesta**(*filename*)

Class for tracing the magnetic field lines given a FIESTA equlibrium

Parameters **filename** (*str*) – the path to the FIESTA MATLAB save file.

Variables ***b_field*** (*VectorFunction3D*) – A 3D vector function of the magnetic field.

b_field

Function for getting the magnetic a 3D vector function of the magnetic field.

get_midplane_lcfs(*psi_p=1.0005*)

Function for getting the inner and outer radial position of the LCFS at the midplane

input: self, a reference to the object itself psi_p, the flux surface of the LCFS, standard is psi_p = 1.005 (otherwise the field-line is located inside the LCFS)

return: Rcross, a list with the outer and inner radial position of the mid-plane LCFS

read_fiesta_model()

Function for reading the FIESTA equilibrium data from a .mat file

input: self, a reference the object itself

output: self.r_limiter, a numpy array with the radial coordinates of the vessel limits self.z_limiter, a numpy array with the vertical coordinates of the vessel limits self.r_vec, a numpy array with the radial grid coordinates self.z_vec, a numpy array with the vertical grid coordinates self.psi, a numpy array with the flux coordinates self.psi_n, a numpy array with the normalised flux coordinate self.psi_axis, self.psi_lcfs, self.mag_axis, self.b_r, a numpy array with the radial magnetic field component self.b_z a numpy array with the vertical field component self.b_phi, a numpy array with the toroidal field component self.b_theta, a numpy array with the poloidal magnetic field component self.b_vac_radius, a self.b_vac, self.i_rod, a float with the current in the rod self.x_points, self.f_profile, self.q_profile, self.lcfs_polygon,

to_cherab_equilibrium()

Function for converting this Fiesta object to a CHERAB equilibrium.

rtype: EFITEquilibrium

Equdsk

Coming soon.

1.1.6 Projections

2D Projections

vita.modules.projection.projection2D.**particle_path_projection**
alias of *vita.modules.projection.projection2D.particle_path_projection*

vita.modules.projection.projection2D.psi_map_projection.**map_psi_omp_to_divertor**(*x_axis_omp*, *divertor_coords*, *fiesta*, *location='lfs'*)

Function mapping the normalised psi from the specified coordinates at the OMP to the specified coordinates at the divertor. Currently the divertor is assumed to be represented by a 1D polynomial function, y = ax + b.

Parameters

- **x_axis_omp** (*np.ndarray*) – Numpy array with the radial coordinates we wish to map at the OMP
- **fiesta** (Fiesta) – A Fiesta object with the 2D equilibrium we wish to map
- **divertor_coords** (*np.ndarray*) – A 2-x-2 numpy array containg the corner points of the divertor in the 2D projection
- **location** (*string*) – a string with the location to evaluate, either 'hfs' or 'lfs'. Default is 'lfs'

Return type dict

Returns

A dictionary containing:

"R_div" [an n-x-1 array] with the R-coordinates at the divertor tile corresponding to the same psi_n as at the OMP

"Z_div" [an n-x-1 array] with the Z-coordinates at the divertor tile corresponding to the same psi_n as at the OMP

"Angles" [an n-x-1 array] with the angles between the field lines and the divertor tile corresponding to the same psi_n as at the OMP

"Flux_expansion" [an n-x-1 array] with the flux expasion at the divertor tile corresponding to the same psi_n as at the OMP

vita.modules.projection.projection2D.project_heat_flux.**project_heat_flux**(*x_pos_omp*, *heat_flux_profile*, *map_dict*)

Function for mapping the heat flux from the OMP to the divertor. The heat flux at a different position is given by:

$$q_{parallel\_surf} = \frac{R_{omp}}{R_{surf}} * \frac{q_{parallel\_omp}}{(f_x/\cos(\alpha))},$$

where R_{omp}, is the radial coordinate at the OMP, R_{surf} is the radial coordinates at the surface, $q_{parallel\_omp}$ is the parallel heat flux at the OMP, α is the incidence angle of the field-lines with respect to the normal of the surface, and f_x is the flux expansion:

$$f_x = R_{omp} * B_{pol}(R_{omp}, Z_{omp})/(R_{surf} * B_{pol}(R_{surf}, Z_{surf})),$$

where Z_{omp} is the vertical position of the OMP (usually 0), Z_{surf} is the vertical position of the surface, and B_{pol} is the poloidal magnetic field and the given coordinates.

Parameters

- **x_pos_omp** (*np.ndarray*) – Radial coordinates at the OMP
- **heat_flux_profile** (*np.ndarray*) – Parallel heat flux at the given coordinates
- **map_dict** (*dict*) – a python dictionary with:

  keys: float x_pos_omp[i], each position at the omp has a corresponding mapped position

  values: dictionary

  dictionary with keys: "R_pos" : float, radial position of the surface "Z_pos" : float, vertical position of the surface "f_x" : float, flux expansion at the given R, Z position "alpha" : float, incidence angle with respect to the normal

  of the surface

Return type np.ndarray

Returns q_surf, the parallel heat flux at the surface position

3D Projections (CHERAB)

Coming soon.

1.1.7 Sol Heat Flux

1.1.8 Utils

1.2 Indices and Tables

- genindex
- modindex
- search

Index