Innovation project
supported by

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Swiss Confederation

Innosuisse – Swiss Innovation Agency

PSI

Particles Trajectories in a Poloidal Magnetic System with Tunnels

Semesterproject

in Computational Science and Engineering

Department of Mathematics

ETH Zurich

written by

Anna Hutter

supervised by

Dr. A. Adelmann (ETH)

March 31, 2025

Abstract

Nuclear fusion is a promising solution for sustainable energy production, offering an almost limitless energy supply with minimal waste and no greenhouse gas emissions. As the demand for clean energy increases, fusion presents a viable alternative to current energy sources. However, achieving practical nuclear fusion remains a major scientific and engineering challenge. The **Tokamak**, the currently leading fusion reactor design, has demonstrated significant progress but faces limitations, including plasma pressure constraints and current-driven instabilities.

The **Polomac**, an alternative design proposed by Filippo Elio [1], allows higher plasma pressures and mitigates some of the instabilities.

This project investigates the Polomac's design by implementing a numerical simulation of particle trajectories in the reactor. It identifies weak points in the current design and contributes to improvements that bring the development closer to the first physical prototype.

Contents

1 Introduction **1**
1.1 Scope of the Project 2

2 Theory **3**
2.1 Maxwell's Equations in Plasma Physics 3
2.1.1 Maxwell's Equations in the Simulated System 3
2.2 Lorentz Force and Particle Motion 4
2.2.1 Numerical Integration Using the Boris Pusher 4
2.3 Magnetic Field Lines, Magnetic Surfaces and Particle Movement 4
2.3.1 Magnetic Field Lines and Surfaces 4
2.3.2 Particle Movement in a Magnetic Field 5

3 Particle Trajectory Simulation Tool **6**
3.1 Overview of Simulation Libraries 6
3.1.1 Independent Parallel Particle Layer (IPPL) 6
3.1.2 The Visualization Toolkit (VTK) 6
3.2 Implementation Details 6
3.2.1 Field Map Handling 6
3.2.2 Field Interpolation 8
3.2.3 Particle Initialization and Tracking 8
3.3 Energy Conservation 9

4 Results **10**
4.1 Magnetic Field Consistency with Maxwell's Equations 10
4.2 Particle Trajectories and Confinement 10
4.2.1 Test cases 10
4.2.2 Particles exiting trough the Tunnel 11
4.2.3 Mirroring 12
4.2.4 Confinement time 16

5 Conclusion and Future Work **18**
5.1 Summary of Key Findings 18
5.2 Limitations of the Study 18
5.3 Possible Improvements and Next Steps 18

A Resources **21**
A.1 Resources 21
A.2 IPPL Build Instructions 21
A.3 Running Particle Trajectory Simulation Tool 22
A.3.1 Preparing input data 22
A.3.2 Running the Simulation Tool 22
A.3.3 Reproducing Plots 23

B Examples of Output Data Plots **24**

List of Figures

1.1 Reproduced from [2]. The plasma (orange) surrounded by coils (blue). The right cutting plane goes trough a tunnel, the left one is between the tunnels 1

3.1 Illustration of the provided grid section. The highlighted region represents the one-eighth portion of the full circular domain for which the grid data is available 7
3.2 Kinetic Energy Difference Relative to Initial Energy 9

4.1 Histogram log plot of the grad and curl of the magnetic field for the new and the old mesh 11
4.2 Particle initialization regions in the simulation. 12
4.3 Magnetic field lines in the simulation domain (Image courtesy of Filippo Elio). . 12
4.4 Random distribution of initial partices 13
4.5 Initial particles distributed on a regular grid 13
4.6 Initial particles uniformly distributed in R and θ 13
4.7 Grid with colored boundary regions 13
4.8 Initial position of particles distributed on a regular grid colored by exit region in the grid boundary 14
4.9 Particle trajectories exhibiting mirroring effects in different regions. 14
4.10 Visualization of B field gradient leading to radial drift (Image courtesy of Filippo Elio) 15
4.11 Log plot of confinement time with randomly distributed particles 17
4.12 Log plot of confinement time with uniformly distributed particles 17
4.13 Log plot of confinement time with particles distributed on grid 17

B.1 Initial position of particles distributed on a regular grid colored by exit time . . . 24
B.2 Initial position of particles distributed randomly, only particles have a confinement time over 2ms colored by exit region in the grid boundary 25
B.3 Initial position of particles uniformly distributed in R and θ colored by exit time 26

Chapter 1

Introduction

There are two primary approaches to achieve magnetic confinement in fusion reactors. The first involves bending a solenoid into a toroidal shape, leading to the Tokamak, which is the state-of-the-art design. The Tokamaks faces several challenges, including density and pressure limits, as well as instabilities and disruptions caused by the plasma current.

An alternative approach is the Polomac [1]. The goal of this design is to develop a fusion reactor suitable for industrial applications that can be built within a reasonable time frame.

The design of the Polomacs fig. 1.1 consists of a central dipol coil (solenoid) inside the plasma, held by support structures that pass through magnetic tunnels and magnetic shielding to protect the in-vessel coil supports.

The concept aims to guide magnetic field lines from the north pole of the solonoid to the south pole using external coils, which creates a confinement system that allows for higher plasma pressures than in the Tokamak and may offer better stability by reducing current-driven instabilities.

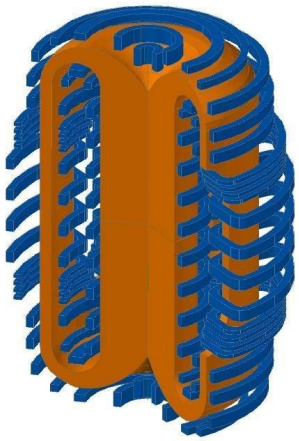


Figure 1.1: Reproduced from [2]. The plasma (orange) surrounded by coils (blue). The right cutting plane goes trough a tunnel, the left one is between the tunnels

An issue are the particles that are lost on the symmetry plan of the tunnels and the weak field regions above them. The goal of this project is to investigate the amount of particles landing in this region.

In [1] a small prototype is introduced, that has a magnetic field produced by water-cooled copper coils in the range of 0.2-0.3T is heated by microwaves and should reach ion temperatures of 100eV.

Numerical simulations play an important role in this process, as building a physical fusion reactor prototype is extremely expensive. With the help of simulations, we can optimize the design and save costs by catching errors before building the prototype.

1.1 Scope of the Project

The focus of this project is to investigate the prototype and design of the Polomac, finding weak points and identifying the particles exiting trough the tunnels and causing damage in the support structures. To achieve this, we developed a simulation tool that tracks the particle trajectories within the vacuum region of a provided field map. The results of the simulation are compared to the existing simulation [1], with the aim of refining the design based on the identified issues

Chapter 2

Theory

2.1 Maxwell's Equations in Plasma Physics

Maxwell's equations describe the fundamental relationships between electric fields, magnetic fields, and their sources. In differential form, Maxwell's equations are given by:

- **Gauss's Law for Electricity**:

$$\nabla \cdot \mathbf{E} = \frac{\rho}{\epsilon_0} \tag{2.1}$$

- **Gauss's Law for Magnetism**:

$$\nabla \cdot \mathbf{B} = 0 \tag{2.2}$$

- **Faraday's Law of Induction**:

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \tag{2.3}$$

- **Ampère's Law with Maxwell's Addition**:

$$\nabla \times \mathbf{B} = \mu_0 \mathbf{J} + \mu_0 \epsilon_0 \frac{\partial \mathbf{E}}{\partial t} \tag{2.4}$$

where $\mathbf{E}$ is the electric field, $\mathbf{B}$ is the magnetic field, $\mathbf{J}$ is the current density, ρ is the charge density, ϵ_0 is the electric permittivity of free space and μ_0 is the magnetic permeability of free space.

2.1.1 Maxwell's Equations in the Simulated System

For the system simulated in this project, the electric field is zero and the magnetic field is static (constant in time). The current density is zero, as the magnetic field exists in a vacuum where there are no free charges or currents. As a consequence the curl of the magnetic field is zero outside any current-carrying sources, such as the coils, and non-zero only in regions near the coils, where a current exists.

2.2 Lorentz Force and Particle Motion

The Lorentz force governs the motion of a charged particle in an electromagnetic field. It is given by the equation:

$$\mathbf{F} = q(\mathbf{E} + \mathbf{v} \times \mathbf{B}) \tag{2.5}$$

where q is the charge of the particle, $\mathbf{E}$ is the electric field, $\mathbf{B}$ is the magnetic field, and v is the velocity of the particle.

When the electirc field is zero, the Lorentz Force simplifies to:

$$\mathbf{F} = q\mathbf{v} \times \mathbf{B} \tag{2.6}$$

2.2.1 Numerical Integration Using the Boris Pusher

The Boris algorithm is widely used to numerically integrate the motion of a charged particle in a magnetic field. The algorithm updates the velocity in the following steps:

1. Compute the half- step velocity update due to the electric field:

$$\mathbf{v}^- = \mathbf{v} + \frac{q\Delta t}{2m}\mathbf{E} \tag{2.7}$$

2. Rotate the velocity due to the magnetic field using the following transformations:

$$\mathbf{t} = \frac{q\Delta t}{2m}\mathbf{B}, \ \mathbf{s} = \frac{2\mathbf{t}}{1 + |\mathbf{t}|^2} \tag{2.8}$$

$$\mathbf{v}^+ = \mathbf{v}^- + (\mathbf{v}^- \times \mathbf{t}), \qquad \mathbf{v}^* = \mathbf{v}^- + (\mathbf{v}^+ \times \mathbf{s}). \tag{2.9}$$

3. Compute the final velocity update due to the electric field:

$$\mathbf{v}^{n+1} = \mathbf{v}^* + \frac{q\mathbf{E}\Delta t}{2m} \tag{2.10}$$

4. Update the position:

$$\mathbf{x}^{n+1} = \mathbf{x}^n + \mathbf{v}^{n+1}\Delta t \tag{2.11}$$

The Boris pusher is a stable method that ensures the conservation of energy over long simulations, making it a great choice in plasma physics and electromagnetic simulations.

In the absence of an electric field, the first and third step of the algorithm are skipped.

2.3 Magnetic Field Lines, Magnetic Surfaces and Particle Movement

2.3.1 Magnetic Field Lines and Surfaces

Magnetic field lines are a graphical representation of the magnetic field direction at different points in space. Magnetic surfaces consist of regions formed by multiple magnetic field lines with the same magnetic field strength. These surfaces are used to describe the structure of the field.

2.3.2 Particle Movement in a Magnetic Field

The motion of a charged particle in a magnetic field can be understood by decomposing its velocity into two components: $v_\perp$, which is perpendicular to the magnetic field, and $v_\parallel$, which is parallel to the field direction.

The Lorentz force acts only on the perpendicular component of velocity because it is always perpendicular to both the velocity and the magnetic field. As a result, this force does not change the speed of the particle but only its direction. This leads to circular motion around the magnetic field lines, with a radius known as the Larmor radius, given by:

$$r_L = \frac{mv_\perp}{qB} \tag{2.12}$$

where m is the particle's mass, q its charge, and B the magnetic field strength.

Since the Lorentz force does not act on $v_\parallel$, the particle moves freely along the field lines. If $v_\parallel$ is nonzero, the combined motion results in a helical trajectory, where the particle spirals along the magnetic field while simultaneously undergoing circular motion in the perpendicular plane.

Chapter 3

Particle Trajectory Simulation Tool

3.1 Overview of Simulation Libraries

3.1.1 Independent Parallel Particle Layer (IPPL)

Independent Parallel Particle Layer [3] is a performance portable C++ library for Particle-Mesh methods. IPPL makes use of Kokkos [4] [5], HeFFTe [**?**], and MPI (Message Passing Interface) to deliver a portable, massively parallel toolkit for particle-mesh methods. IPPL supports simulations in one to six dimensions, mixed precision, and asynchronous execution in different execution spaces (e.g. CPUs and GPUs).

3.1.2 The Visualization Toolkit (VTK)

The Visualization Toolkit [6] is open source software for manipulating and displaying scientific data. It comes with state-of-the-art tools for 3D rendering, a suite of widgets for 3D interaction, and extensive 2D plotting capability.

3.2 Implementation Details

3.2.1 Field Map Handling

The magnetic field in this project is represented as a vector field stored in a `vtkUnstructuredGrid`, a VTK library class that supports point location and field value interpolation in the grid.

The grid consists of cells, specifically hexahedra and wedges. Each cell has an associated `cell ID` and a `boundaryType` string that indicates whether it belongs to the interior of the grid or one of the boundary regions. The `cell ID` and `boundaryType` are stored as `vtkDataArray` within the cell data of the grid.

Field Representation and Storage

The magnetic field vectors are stored in a `vtkDataArray` within the point data of the `vtkUnstructuredGrid`.

Grid Structure

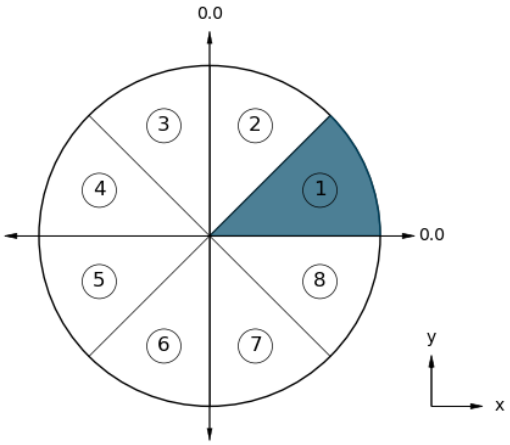


Figure 3.1: Illustration of the provided grid section. The highlighted region represents the one-eighth portion of the full circular domain for which the grid data is available

The grid is only given for one-eighth of the full circular region and includes only the upper half where R_z is positive (see fig. 3.1). The full domain can be obtained by mirroring the grid across the entire circular region and both the positive and negative z-regions.

To map a position $\mathbf{R}$ from the full simulation domain to the provided grid region, a reference position $\mathbf{R}_{\text{ref}}$ is determined that lies within octant (1) (see fig. 3.1). This transformation is computed as follows:

$$R_{\text{ref},i} = |R_i| \quad \text{for} \quad i \in \{x, y, z\}$$

$$(R_{\text{ref},x}, R_{\text{ref},y}, R_{\text{ref},z}) = \begin{cases} (R_{\text{ref},y}, R_{\text{ref},x}, R_{\text{ref},z}), & \text{if } |R_x| < |R_y| \\ (R_{\text{ref},x}, R_{\text{ref},y}, R_{\text{ref},z}), & \text{otherwise} \end{cases}$$

After obtaining the corresponding grid values $\mathbf{V}_{\text{grid}}$ at $\mathbf{R}_{\text{ref}}$, the original values in the full simulation domain are reconstructed by applying the appropriate symmetry transformations as follows:

$$(V_{\text{orig},x}, V_{\text{orig},y}, V_{\text{orig},z}) = \begin{cases} (V_{\text{grid},y}, V_{\text{grid},x}, V_{\text{grid},z}), & \text{if } |R_x| < |R_y| \\ (V_{\text{grid},x}, V_{\text{grid},y}, V_{\text{grid},z}), & \text{otherwise} \end{cases}$$

$$V_{\text{orig},x} = \text{sgn}(R_x) * \text{sgn}(R_z) V_{\text{orig},x}, \quad V_{\text{orig},y} = \text{sgn}(R_y) * \text{sgn}(R_z) V_{\text{orig},y}, \quad V_{\text{orig},z} = \text{sgn}(R_z) V_{\text{orig},z}$$

3.2.2 Field Interpolation

The `vtkStaticCellLocator` is used to efficiently locate the grid cell containing a given point. A cell locator is a spatial search structure in VTK that quickly locates cells in 3D.

The function `vtkStaticCellLocator::FindCell` determines the cell in the `vtkUnstructuredGrid` that contains a given point. It returns:

- The `cell ID` of the located cell.
- The `parametric coordinates` inside the cell.
- The `interpolation weights` for the cell's corner points.

The interpolation weights w_i are computed such that they satisfy the following equation:

$$\mathbf{R} = \sum_i N_i(\boldsymbol{\xi}) \mathbf{R}_{cell,i}$$

where $\mathbf{R}$ is the position of the point inside the cell, $\mathbf{R}_{cell,i}$ are the positions of the cell's corner points, $N_i(\boldsymbol{\xi})$ are the shape functions that depend on the cell type.

The weights w_i correspond directly to $N_i(\boldsymbol{\xi})$ and satisfy $\sum_i w_i = 1$. Once determined, these weights are used to calculate the magnetic field at the given point.

Interpolation weight issues

An issue observed during the simulation is the occurrence of negative interpolation weights returned by `vtkStaticCellLocator::FindCell`. In most cases, the function correctly identifies the cell and provides valid interpolation weights. However, in rare instances, some weights are slightly negative, with the smallest observed value being approximately -5×10^{-4}.

To further investigate this behavior, interpolation was tested at grid points to examine edge cases within the domain. While negative weights still occur in some cases, they were significantly bigger (never below -10^{-10}).

A post was made on the VTK support discourse to seek clarification on this issue, but no response has been received so far. To ensure accurate field interpolation and avoid errors in the results, particles with interpolation weights smaller than -10^{-10} are thrown out of the simulation.

3.2.3 Particle Initialization and Tracking

Initialization

The simulation reads particle data from a CSV file, which can be provided in either cylindrical or cartesian coordinates. The first letter in the first line of the file specifies the coordinate system: `"R"` for cylindrical coordinates (R, Θ, Z), otherwise Cartesian coordinates (X, Y, Z). The position is given in meter (m).

The kinetic energy is read in electron volts (eV) and converted to joules for the simulation.

Instead of directly specifying the velocity components, the input file provides velocity split factors along each coordinate direction. These values indicate the relative proportion of the total velocity in each direction. The actual velocity magnitude is computed later based on the kinetic energy. The final velocity components are obtained by scaling the velocity splits accordingly.

Tracking

At each time step of the simulation the following information about each particle is written to a CSV file:

- **Time** $[s]$
- **Particle ID**
- **Position** $[m]$: X, Y, Z
- **Cell ID:** cell in which particle is located
- **Velocity** $[m/s]$**:** V_x, V_y, V_z
- **Kinetic Energy** $[J]$
- **Magnetic Field** $[T]$**:** $B_x, B_y, B_z, |B|$
- **Curl of the Magnetic Field** $[T/s]$**:** $[\nabla \times \mathbf{B}]_x, [\nabla \times \mathbf{B}]_y, [\nabla \times \mathbf{B}]_z, |\nabla \times \mathbf{B}|$
- **Interpolation Weights**

A particle stays in the simulation as long as it stays in the region of the provided grid. For each particle exiting the mesh, the following data is written to an other CSV file:

- **Time** $[m]$
- **Particle ID**
- **Last Cell:** Cell Id of the last cell particle was located at before exiting grid
- **Boundary_type:** `boundaryType` of Last Cell

This recorded data provides detailed insights into the particle trajectories, energy evolution, and interaction with the magnetic field.

3.3 Energy Conservation

The energy is well conserved in the simulation, as can be seen in fig. 3.2. This is a good indicator of the stability of the simulation.

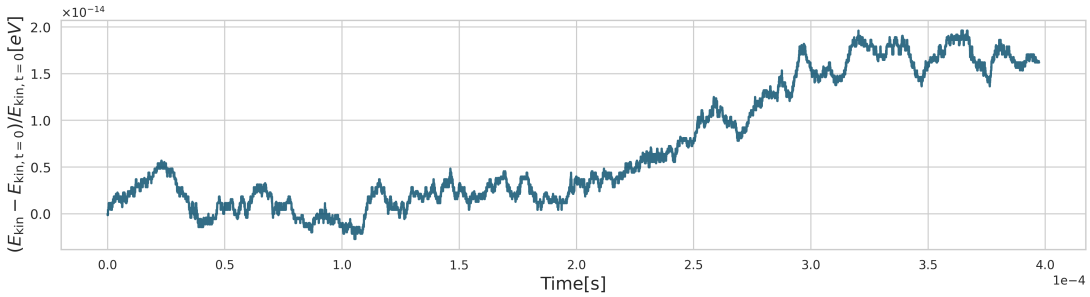


Figure 3.2: Kinetic Energy Difference Relative to Initial Energy

Chapter 4

Results

4.1 Magnetic Field Consistency with Maxwell's Equations

Given a discretized field map B, Maxwell's equations are a good measure for assessing the quality of the magnetic field data. Since the field is provided for a region in vacuum away from the coil,eq. (2.1) and eq. (2.4) apply and the gradient as well as the curl should be exactly zero section 2.1.1. However, for a field map in a numerical grid, discretization errors cause $\nabla \cdot \mathbf{B}$ and $\nabla \times \mathbf{B}$ to be non zero.

Comparing the gradient and curl of the old mesh and the newer mesh using a histogram section 4.1, we can see that the newer mesh contains fewer values deviating far from zero, for both the gradient as well as the curl. This is a clear increase in field quality from the old to the new mesh.

4.2 Particle Trajectories and Confinement

4.2.1 Test cases

For the runs of the simulation we choose initial particles distributed in three distinct regions: one in the outer region of the mesh and two in the inner region (see fig. 4.2).

These regions are selected based on the structure of the magnetic field lines. Particles starting on a magnetic line follow along it through the grid. In fig. 4.3 we see the magnetic lines at $\theta = 1$ and $\theta_{min} = 44°$. They are already close to the boundary of the grid. Magnetic lines closer to the boundary would lead to particles exiting trough the boundary out of the grid really early in the simulation. This analysis leads to the choice of two inner region from $R_{min} = 0.005m$ to $R_{max} = 0.144m$, $\theta_{min} = 0$ to $\theta_{max} = 45$ at $z = 0$ and $z = 0.2$ and an outer region at $R_{min} = 0.318m$ to $R_{max} = 0.332m$, $\theta_{min} = 20$ to $\theta_{max} = 45$ at $z = 0$ (see fig. 4.3), from where the particles are started.

For each region, three different spatial distributions were used to initialize the particles:

- **Random Distribution:** A specified number of particles are placed randomly within the region.
- **Uniform Radial and Angular Distribution:** Particles are distributed uniformly over the radial and angular coordinates.

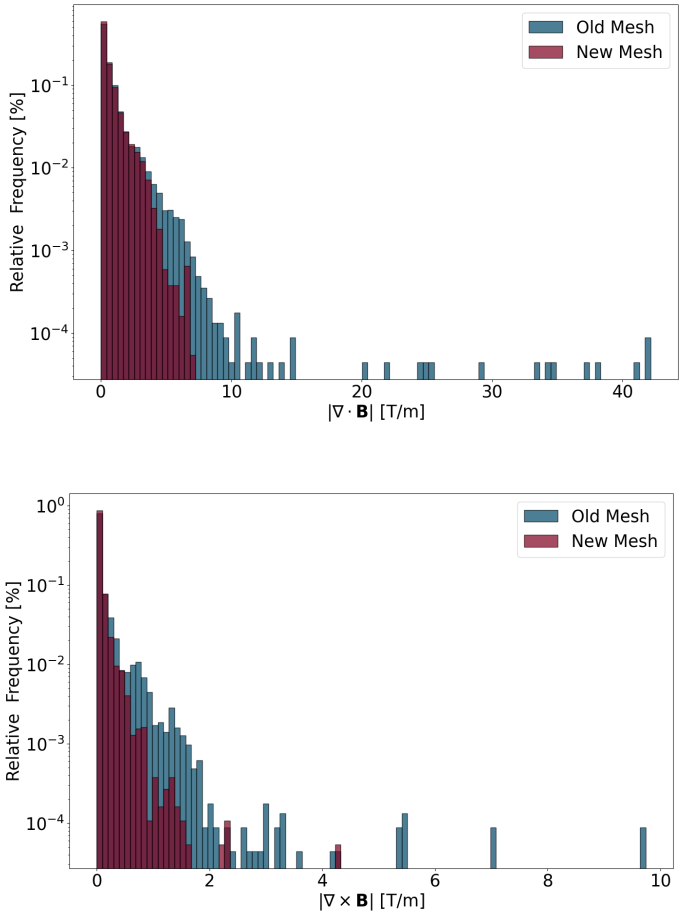


Figure 4.1: Histogram log plot of the grad and curl of the magnetic field for the new and the old mesh

- **Grid-Based Distribution:** A uniform grid is placed on the region, and particles are initialized at the grid points.

4.2.2 Particles exiting trough the Tunnel

The bar plots section 4.2.2 shows the percentage of particles exiting trough each boundary region for particles initialized with three distributions section 4.2.1. The results of the random and grid-based distributions are very similar, with approximately 25% of the particles exiting through the tunnel. This is a issue, as particles leaving the grid in this region can cause damage to the support structure of the inner coil.

The uniform distribution plot shows a similar distribution with a key difference, the percentage of particles exiting through `Tip_E` is almost doubled. Since particles are distributed uniformly over both radius and angle, a higher number of particles is placed in regions where the radius is small. Looking at fig. 4.8 it becomes clear that particles in the regions with a small radius

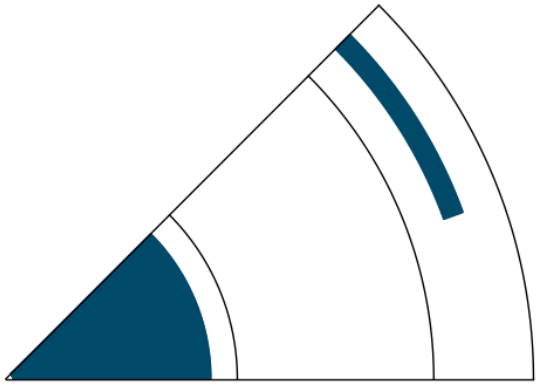


Figure 4.2: Particle initialization regions in the simulation.

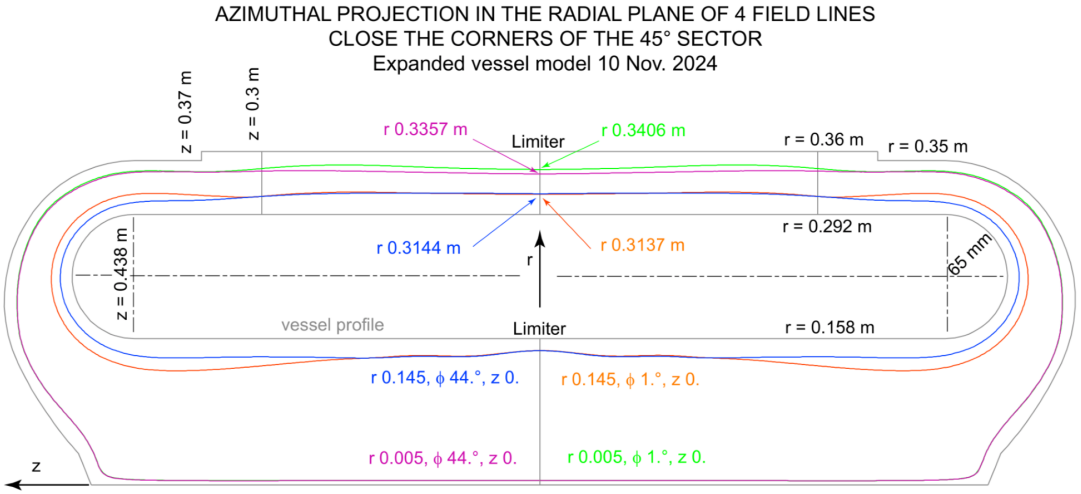


Figure 4.3: Magnetic field lines in the simulation domain (Image courtesy of Filippo Elio).

have a high chance of leaving the mesh at `Tip_E`, this explains the spike. Aside from this localized difference, the overall exit distribution closely resembles that of the other two initialization methods.

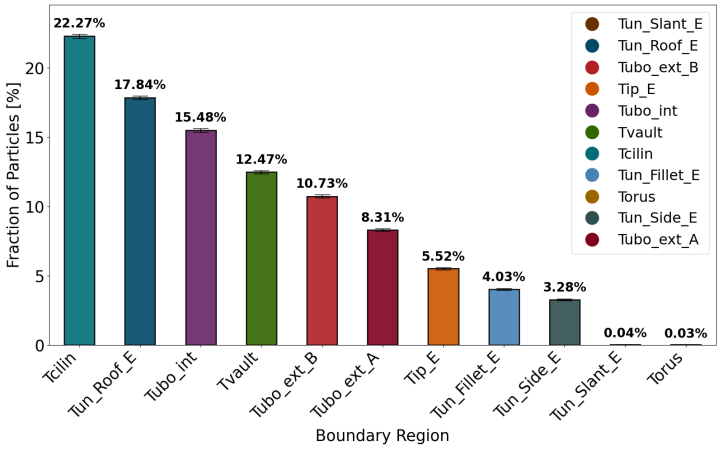
For all imput distribution between 22% and 25% of the particles exit the grid trough the Tunnel. To avoid this, the B field map has to be adjusted.

4.2.3 Mirroring

In particle simulations, *mirroring* or *inversion* refers to a phenomenon in which a charged particle reverses their trajectory due to interactions with increasing magnetic field strength. fig. 4.9

When a charged particle moves into a region with a stronger magnetic field, its velocity component $v_{\parallel}$ decreases while $v_{\perp}$ increases.

If the magnetic field strength increases enough, $v_{\parallel}$ can reach zero, causing the particle to reverse direction and return to a region of lower field strength.



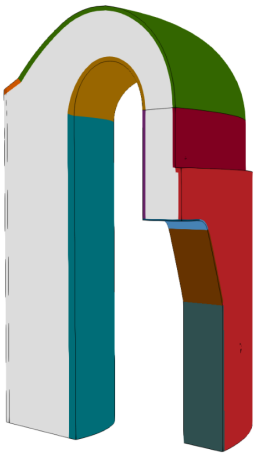
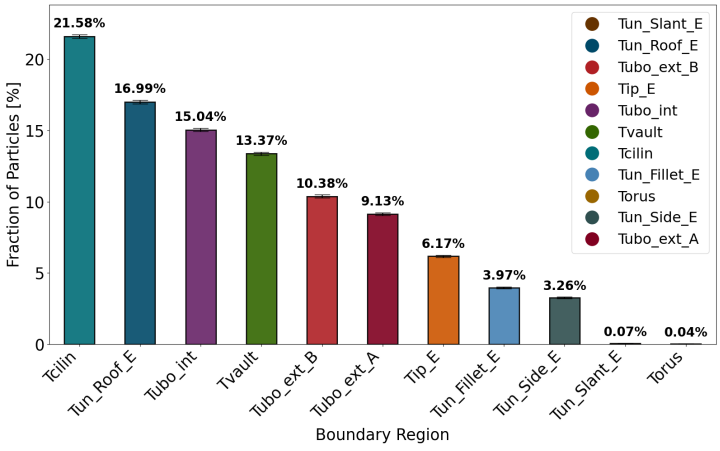


Figure 4.4: Random distribution of initial partices



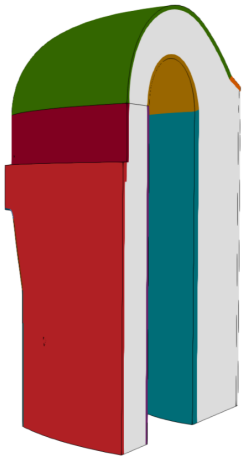


Figure 4.5: Initial particles distributed on a regular grid

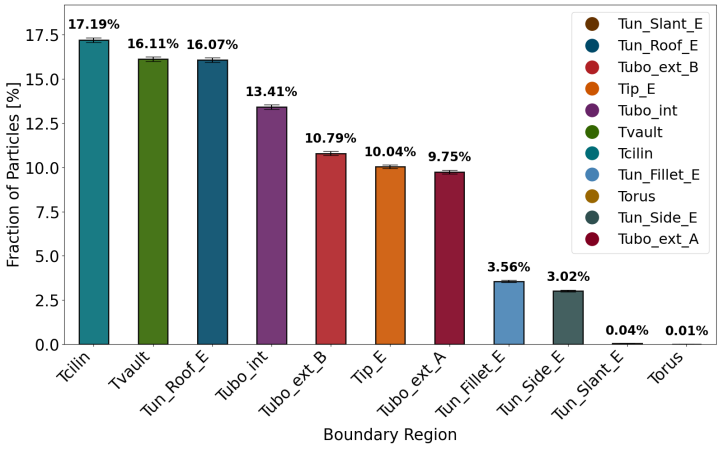


Figure 4.6: Initial particles uniformly distributed in R and θ

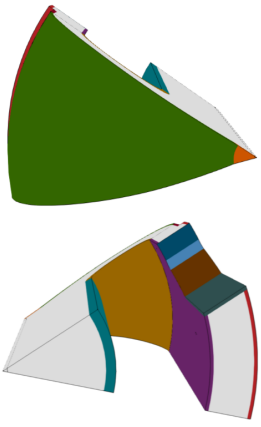


Figure 4.7: Grid with colored boundary regions

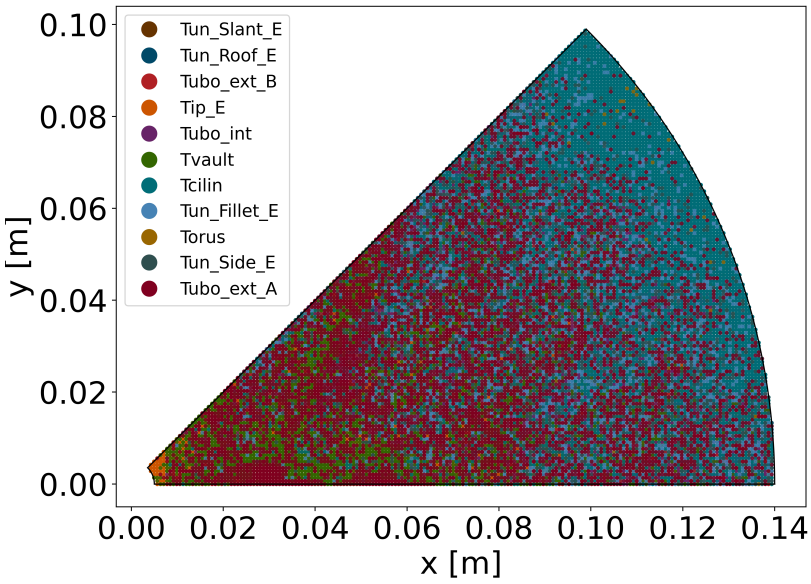


Figure 4.8: Initial position of particles distributed on a regular grid colored by exit region in the grid boundary

Figure 4.9: Particle trajectories exhibiting mirroring effects in different regions.

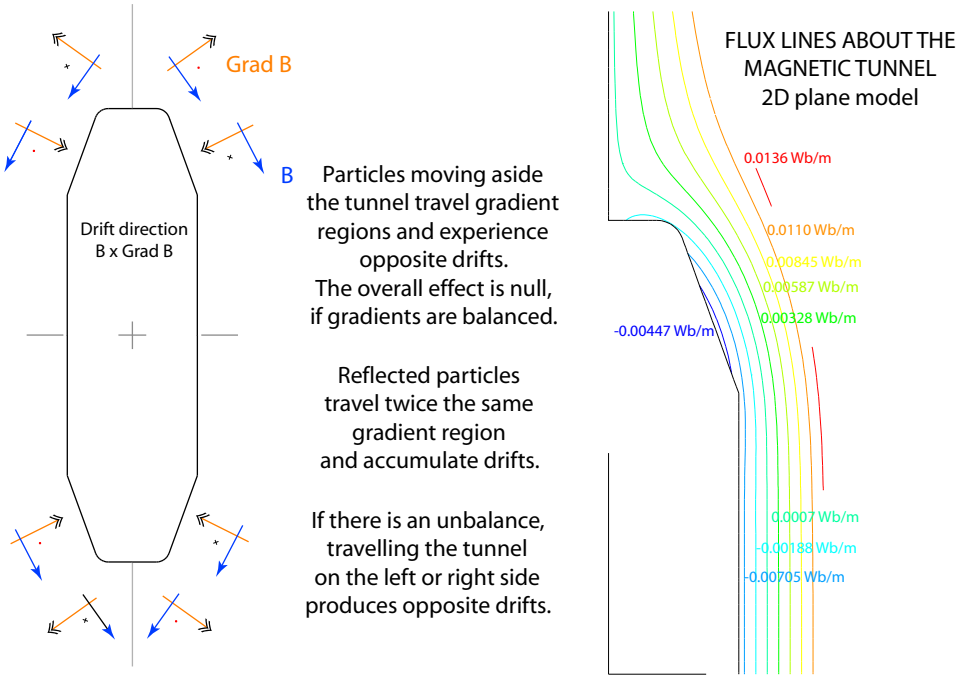


Figure 4.10: Visualization of B field gradient leading to radial drift (Image courtesy of Filippo Elio)

In the provided field map, the inner region has a lower magnetic field strength than to the outer region. As a result, particles moving outward can experience mirroring.

Impact on Particle Dynamics

In discussion with Filippo Elio, we came to the following observation about the impact of the mirroring on the simulation:

Different mirroring locations lead to distinct trajectory modifications (see fig. 4.10):

- Path inversion above or below the tunnel increases the particle's time spent in a region where $\nabla \cdot B$ is not zero (see fig. 4.10), causing radial drift to accumulate and the particle to jump to a different magnetic surface, either inward or outward, depending on its location and the strength of the field gradient. Top inversions produce outwards jumps, bottom inversions produce inwards jumps.
- Path inversions between tunnels where $\nabla \cdot B$ is parallel to B (higher field without change of directions) do not produce any jumps of the proton to a different magnetic surface.
- Path inversion in low-field regions above/below the tunnels or near the Z axis at top/bottom also produce jumps to other magnetic surfaces.

In a fusion reactor, particles changing magnetic surfaces is a problem because it disrupts the containment and control of the plasma. The magnetic field is designed to confine charged particles within specific surfaces, preventing them from escaping and maintaining stable plasma conditions. When particles jump between magnetic surfaces, they can escape the intended confinement regions, possibly collide with the reactor walls, leading to loss of energy and potential damage to the reactor walls and support structures. To maintain plasma stability and improve confinement times, it is therefore critical to change the magnetic field such that the mirroring effects are minimized.

4.2.4 Confinement time

The confinement time of particles provides insight into the stability and the effectiveness of magnetic confinement. Figure 4.13 shows the distribution of particle confinement times on a logarithmic scale for the three particle distributions over the whole input area.

Around 50% of the particles are lost within the first 0.14 ms independent of the initial distribution of particles, and only a very small fraction remain confined for more than 4 ms. This is significantly lower than the confinement times achieved in advanced fusion reactors, where plasmas can be sustained for over a minute.

One reason for this is the mirroring discussed in the previous subsection, solving this problem will likely increase the confinement time significantly.

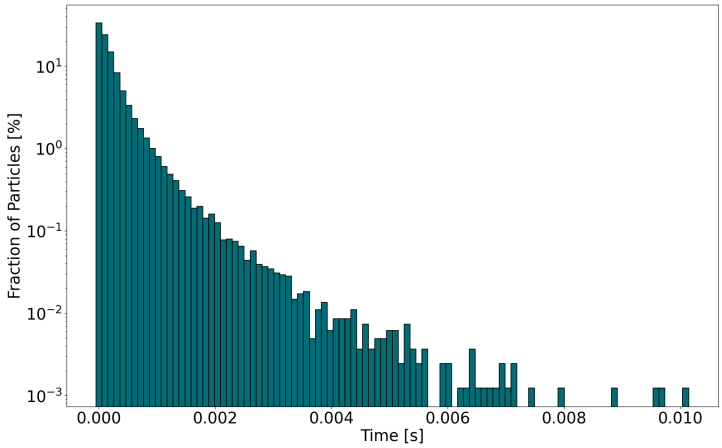


Figure 4.11: Log plot of confinement time with randomly distributed particles

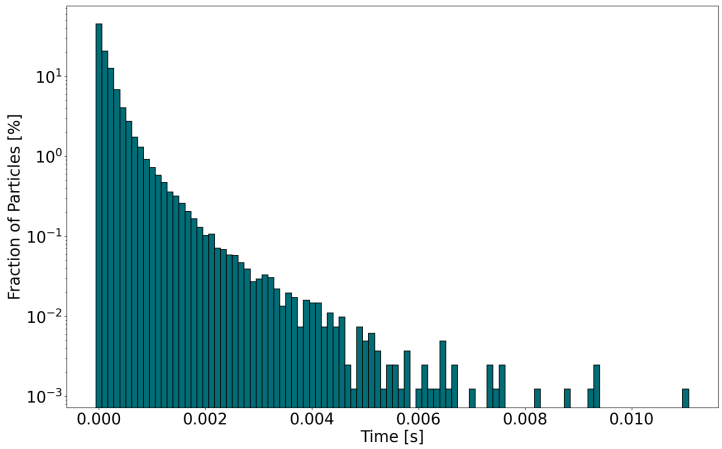


Figure 4.12: Log plot of confinement time with uniformly distributed particles

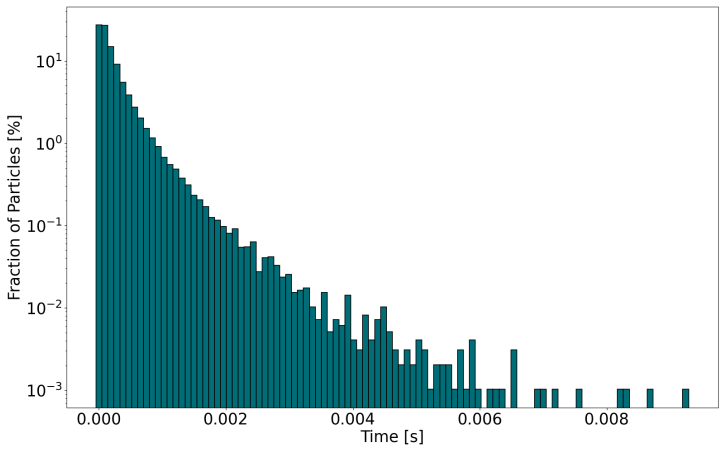


Figure 4.13: Log plot of confinement time with particles distributed on grid

Chapter 5

Conclusion and Future Work

5.1 Summary of Key Findings

The transition from the first mesh to the second mesh has led to a noticeable improvement in fulfilling Maxwell's conditions. Moreover, adjustments to the field map of the second mesh are necessary to prevent mirroring effects and unintended particle transitions between magnetic surfaces (see section 4.1).

Filippo Elio is already working on the next design iteration. Influenced by these findings, the difference in magnetic field strength between the inner and outer plasma regions are reduced and the upper shape of the confinement tunnel will be modified into a more rounded form.

As part of this semester project, a simulation tool was developed to track particle trajectories through the grid, analyze their paths, and determine their exit positions. This tool provides valuable insights into particle behavior and the field quality in future meshes. Several types of output data that can be further analyzed is produced by each simulation, examples are given in the Appendix (see appendix B).

5.2 Limitations of the Study

A key limitation of this study is the issue with interpolation weights (section 3.2.2), which causes approximately 10% of initialized particles to be excluded from the simulation. This occurs even after long simulation times, resulting in unnecessary computational time and potential gaps in the results for some input positions.

5.3 Possible Improvements and Next Steps

Explore alternative libraries to handle the B field map, which may solve the interpolation weight issue.

Utilize the developed simulation tool to conduct an uncertainty quantification analysis, assessing the sensitivity of particle trajectories with respect to initial position (already started in this project), magnetic field and energy variations.

Acknowledgements

I would like to thank Dr. Andreas Adelmann for his valuable input, guidance and encouragement. I would also like to thank Filippo Elio and Deutelio for the great collaboration and the many discussions that contributed significantly to this project.

Bibliography

[1] F. Elio, "Revisiting the poloidal magnetic confinement," *Fusion Engineering and Design*, vol. 89, no. 6, pp. 806–811, 2014.

[2] F. Elio, F. Elio, T. Fulceri, M. G. Leone, and C. Sborchia, "The polomac approach to fusion energy," *The Journal of Technological and Space Plasmas*, vol. 5, pp. 172–180, Oct. 2024.

[3] S. Muralikrishnan, M. Frey, A. Vinciguerra, M. Ligotino, A. J. Cerfon, M. Stoyanov, R. Gayatri, and A. Adelmann, "Scaling and performance portability of the particle-in-cell scheme for plasma physics applications through mini-apps targeting exascale architectures," in *Proceedings of the 2024 SIAM Conference on Parallel Processing for Scientific Computing (PP)*, pp. 26–38, SIAM, 2024.

[4] H. C. Edwards, C. R. Trott, and D. Sunderland, "Kokkos: Enabling manycore performance portability through polymorphic memory access patterns," *Journal of Parallel and Distributed Computing*, vol. 74, no. 12, pp. 3202 – 3216, 2014. Domain-Specific Languages and High-Level Frameworks for High-Performance Computing.

[5] C. R. Trott, D. Lebrun-Grandié, D. Arndt, J. Ciesko, V. Dang, N. Ellingwood, R. Gayatri, E. Harvey, D. S. Hollman, D. Ibanez, N. Liber, J. Madsen, J. Miles, D. Poliakoff, A. Powell, S. Rajamanickam, M. Simberg, D. Sunderland, B. Turcksin, and J. Wilke, "Kokkos 3: Programming model extensions for the exascale era," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 4, pp. 805–817, 2022.

[6] W. Schroeder, K. Martin, and B. Lorensen, *The Visualization Toolkit (4th ed.)*. Kitware, 2006.

Appendix A

Resources

A.1 Resources

- IPPL repository (GitHub): https://github.com/IPPL-framework/ippl
- IPPL project branch: https://github.com/anekslen/ippl/tree/semesterproject
- Student Repository branch: https://github.com/anekslen/semesterproject/tree/main

A.2 IPPL Build Instructions

1. Clone the IPPL project branch repository

   ```
   git clone https://github.com/anekslen/ippl.git
   ```

2. Switch to semesterproject branch:

   ```
   cd ippl
   git checkout semesterproject
   ```

3. Make sure the following prerequisites are installed:

   - CMake (https://cmake.org/)
   - A compatible C++ compiler:
     - GCC (https://gcc.gnu.org/)
     - Clang (https://clang.llvm.org/)
   - MPI, for example OpenMPI (https://www.open-mpi.org/)
   - VTK (https://vtk.org/)f

4. Setup a build directory:

   ```
   mkdir build
   cd build
   ```

5. Build the project:

   - Serial build:

     ```
     cmake .. -DCMAKE_BUILD_TYPE=Release -DCMAKE_CXX_STANDARD=20 -DENABLE_VTK=ON
     ```

- OpenMPI build:

```
cmake .. -DCMAKE_BUILD_TYPE=Release -DCMAKE_CXX_STANDARD=20 -DENABLE_VTK=ON
-DIPPL_PLATFORMS=openmp
```

6. Compile:

```
make
```

A.3 Running Particle Trajectory Simulation Tool

A.3.1 Preparing input data

The simulation needs input data for each particle as described in section 3.2.3.

In the folder **semesterproject/inputdata/reactormesh** you can find the files **pointdataInner.txt** and **pointdataOuter.txt** defining the region, number of particles, velocity splits and energy for the particles used in the runs discussed in the result section.

The python scripts **GridInput.py**, **RandomInputData.py** and **UniformInputData.py** will read from these two files and create input files for the simulation.

A.3.2 Running the Simulation Tool

Change in to the directory **build/semesterproject** and create a folder **data**.

The simulation can either save the data for each particle at each time step, or only save the data for the time step before the particle exits the simulation. The first version is done, if the whole particle trajectories are needed. The second version decreases the run time significantly and is used when only the exit region or confinement time of the particles are needed.

To run the simulation use the following command:

```
./FusionReactor <Np> <Nt> <step_method> <grid_file_name> <input_particles_file_name> <dt>
    <output_folder> <writeData> --info 10
```

Where:

- Np = Total number of particles in the simulation
- Nt = Number of time steps
- step_method = Time-stepping method used e.g. Boris
- grid_file_name = path to the grid file in VTK format
- input_particles_file_name = path to the file containing the initial particle positions
- dt = time step size
- output_folder = path to the folder where the output files will be written
- writeData = true if the data should be written for each time step, false if only for the last position of each particle

A.3.3 Reproducing Plots

The python scripts to reproduce the plots found in this report can be found in the folder **semesterproject/visualization**.

- The file **Boundarys/BoundaryRegion.py** creates the plots for section 4.2.2, fig. 4.13, fig. 4.8, fig. B.1, fig. B.2 and fig. B.3.
- The file **MeshRegion.py** creates the plots for fig. 4.2 and fig. 3.1.
- The file **CurlBField/HistPlot.py** creates the plots for fig. 4.1.
- The file **Energy/PlotEnergy.py** creates the plots for fig. 3.2.

Appendix B

Examples of Output Data Plots

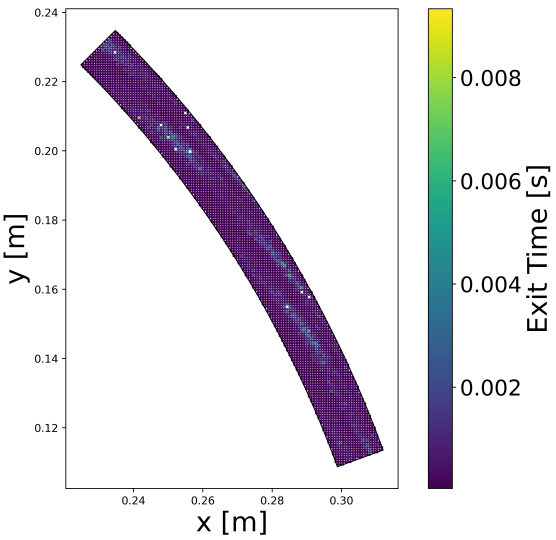


Figure B.1: Initial position of particles distributed on a regular grid colored by exit time

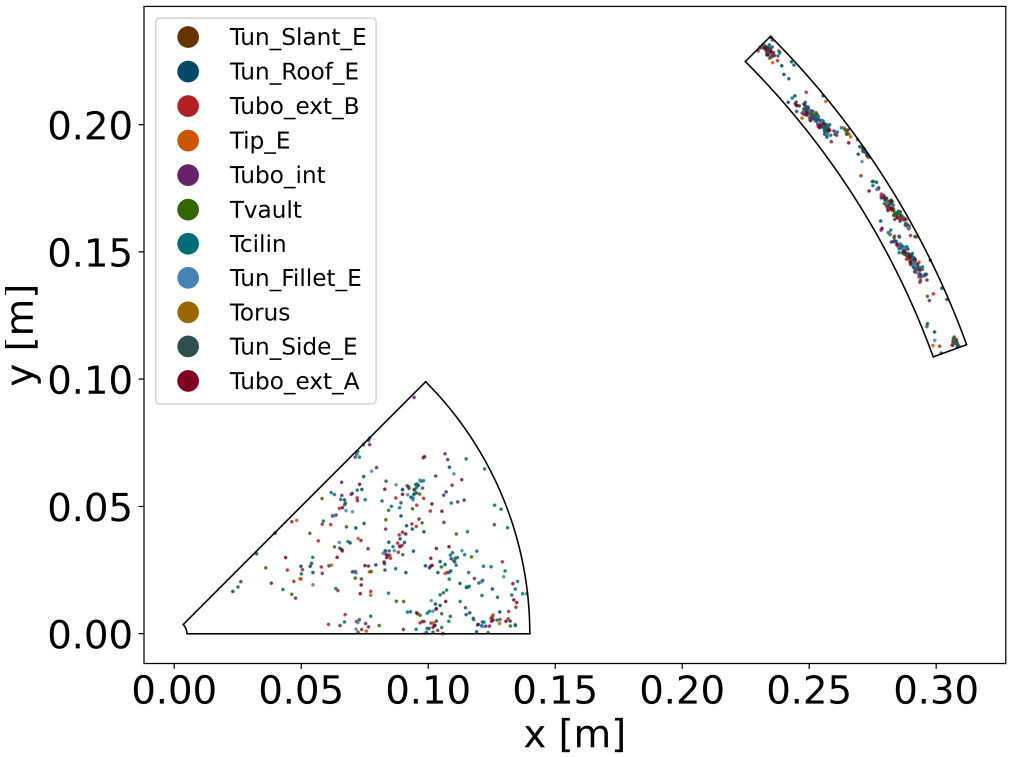


Figure B.2: Initial position of particles distributed randomly, only particles have a confinement time over 2ms colored by exit region in the grid boundary

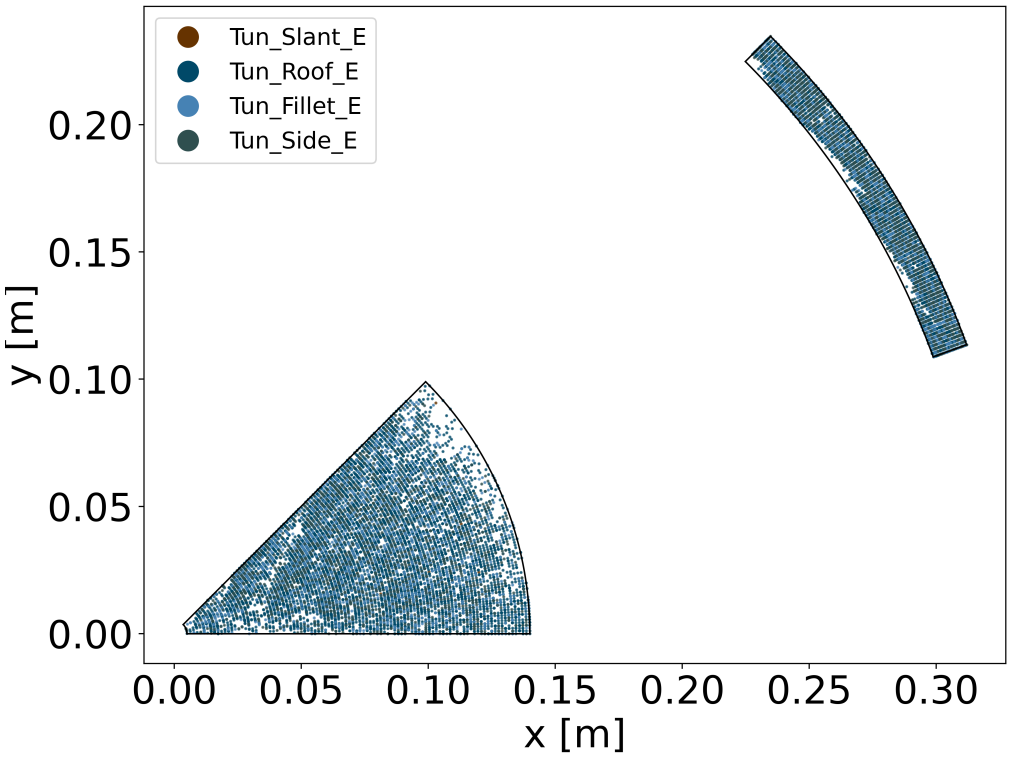


Figure B.3: Initial position of particles uniformly distributed in R and θ colored by exit time

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of Originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of Master's thesis (in block letters):

Particles Trajectories in a Poloidal Magnetic System with Tunnels

Authored by (in block letters):

For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**
Hutter	Anna

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**
31.03.2025	

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.