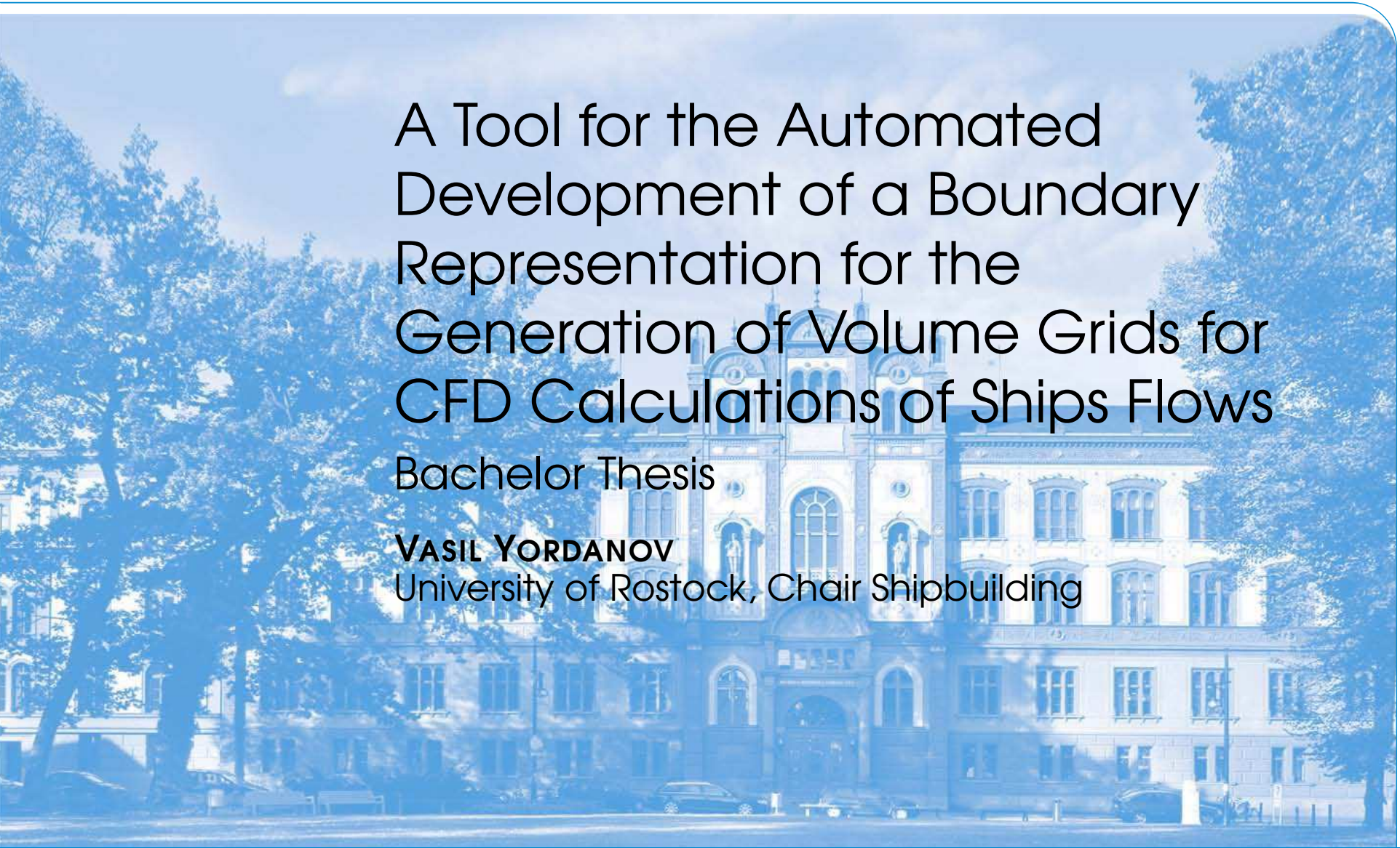
Universität Rostock

Traditio et Innovatio



A Tool for the Automated Development of a Boundary Representation for the Generation of Volume Grids for CFD Calculations of Ships Flows

Bachelor Thesis

Vasil Yordanov
University of Rostock, Chair Shipbuilding

Table of Contents

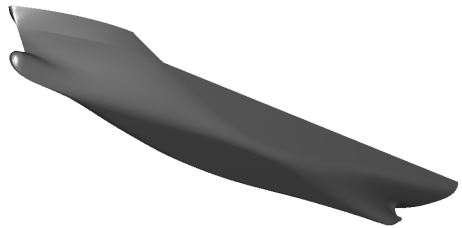
Table of Contents

Table of Contents

Table of Contents

Problem Definition

An overview of the operations prior to a CFD simulation

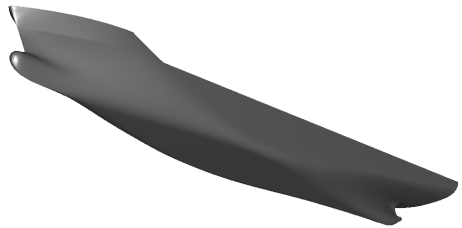


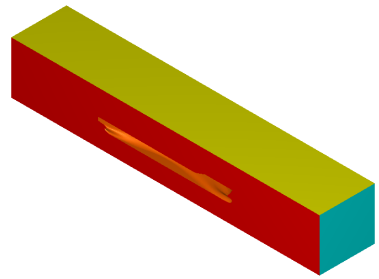
Pre-CFD stages

1. Defining of the ship geometry *This step is done by using 3D Ship Design Software*

Problem Definition

An overview of the operations prior to a CFD simulation



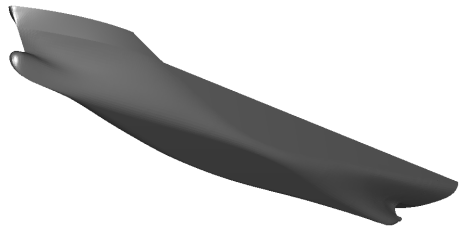
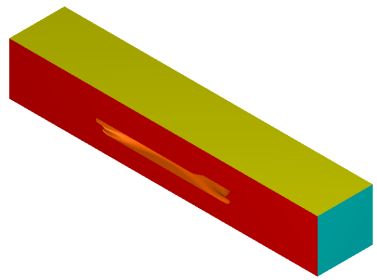
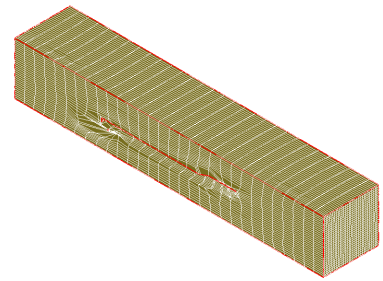


Pre-CFD stages

1. Defining of the ship geometry *This step is done by using 3D Ship Design Software*
2. Defining of a boundary box of the fluid domain *This step is done by using CAD Software*

Problem Definition

An overview of the operations prior to a CFD simulation

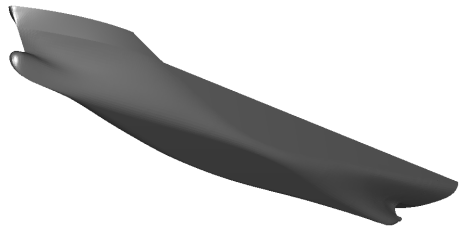
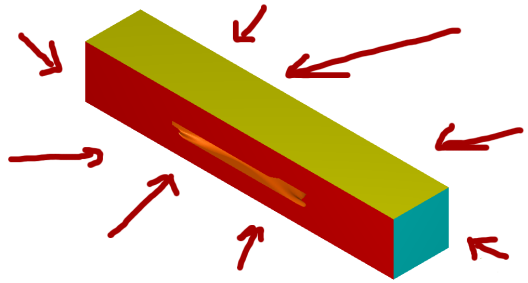
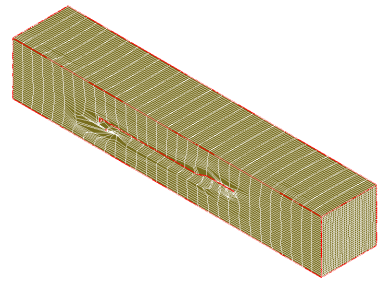




Pre-CFD stages

1. Defining of the ship geometry *This step is done by using 3D Ship Design Software*
2. Defining of a boundary box of the fluid domain *This step is done by using CAD Software*
3. Building a volume mesh *This step is done by using Numeca meshing module HEXPRESS*

Problem Definition

An overview of the operations prior to a CFD simulation

Pre-CFD stages

1. Defining of the ship geometry *This step is done by using 3D Ship Design Software*
2. Defining of a boundary box of the fluid domain *This step is done by using CAD Software*
3. Building a volume mesh *This step is done by using Numeca meshing module HEXPRESS*

Main objective

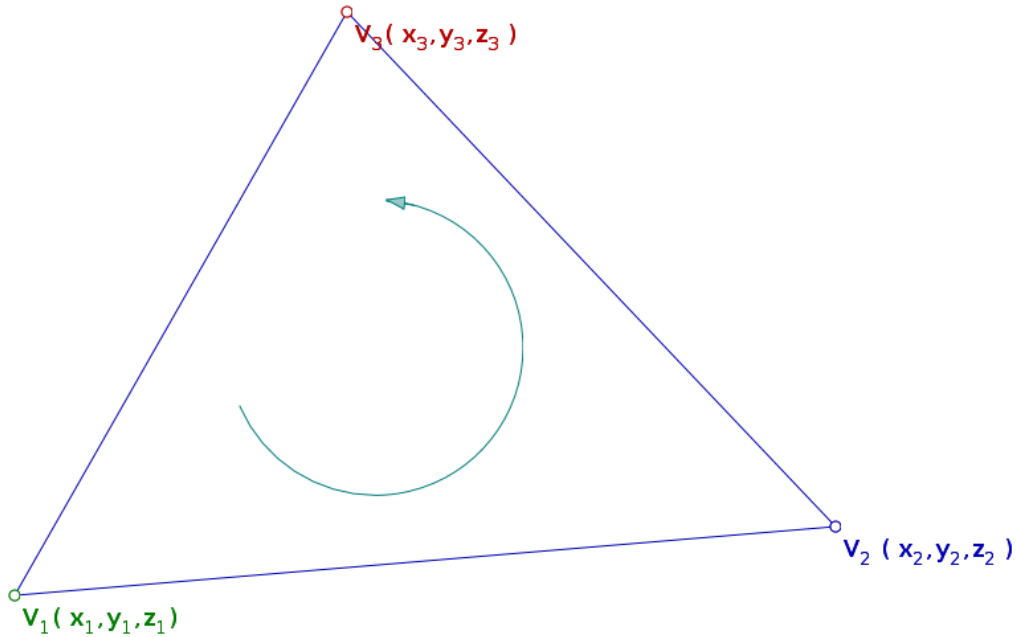
The main aim of the thesis is to provide a tool which builds the boundary box automatically without using any third-party CAD software.

Technical Specifications

Information about the used hull representation

STL file - structure

```
solid name
  ...
  facet normal  n_i   n_j   n_k
  outer loop
    vertex  x_1   y_1   z_1
    vertex  x_2   y_2   z_2
    vertex  x_3   y_3   z_3
  endloop
  endfacet
  ...
endsolid name
```

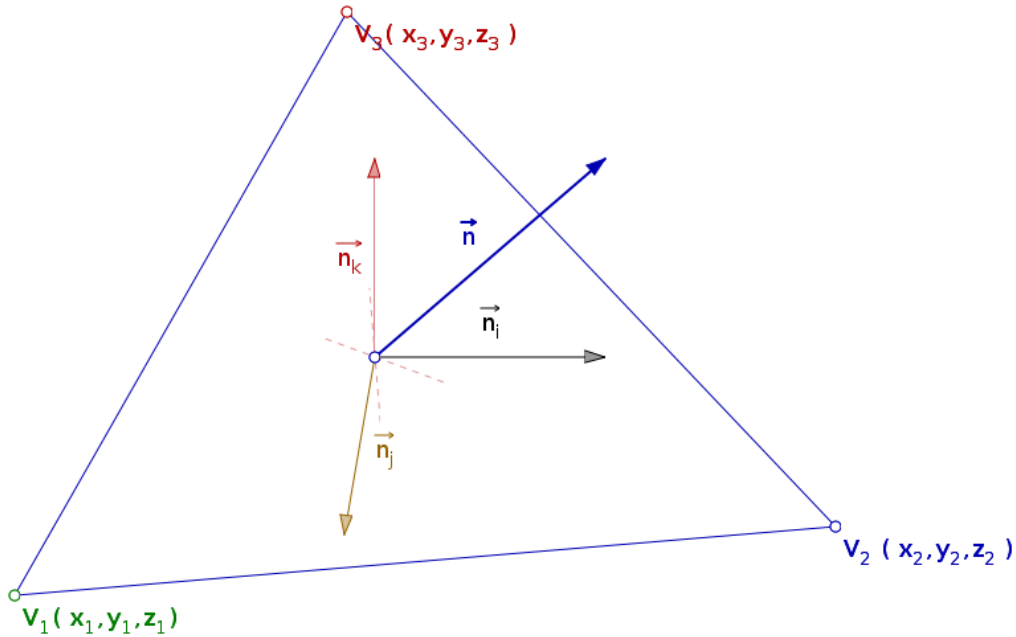


Technical Specifications

Information about the used hull representation

STL file - structure

```
solid name
  ...
  facet normal  ni   nj   nk
  outer loop
    vertex  x1   y1   z1
    vertex  x2   y2   z2
    vertex  x3   y3   z3
  endloop
  endfacet
  ...
endsolid name
```



Technical Specifications

Information about the used hull representation

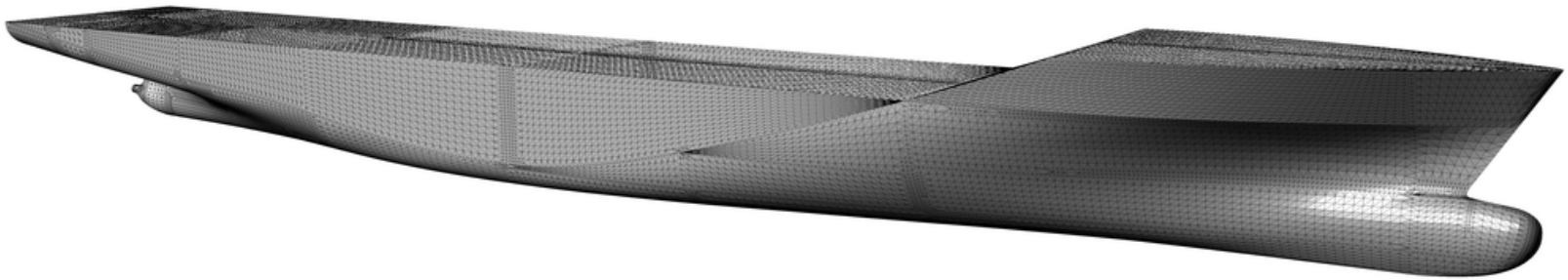


Figure: STL representation of the initial ship geometry

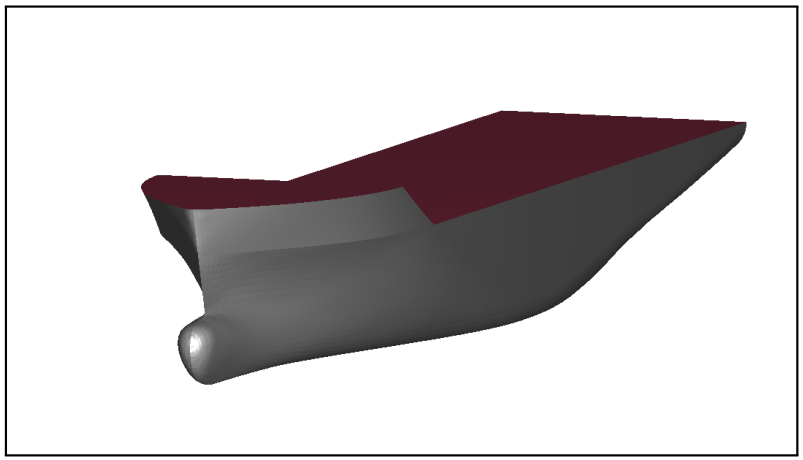
Disadvantages

- No topology information is presented in this format

Advantages

- This is the most simplified representation possible

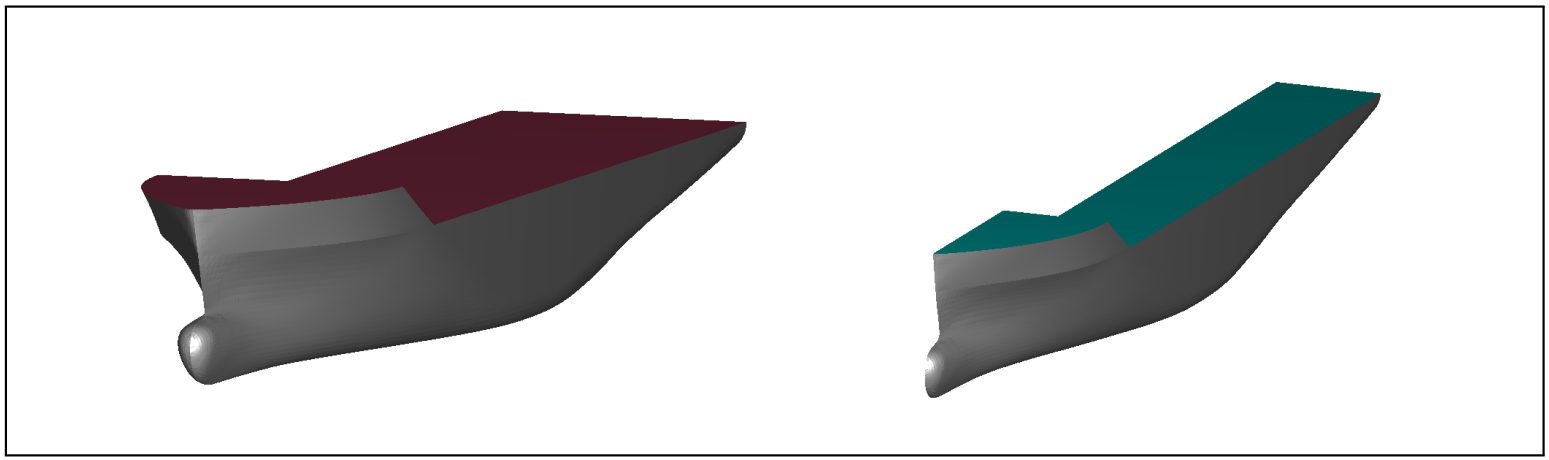
Goals to be achieved



Process sequence

1. Reading the initial ship triangulation from STL file

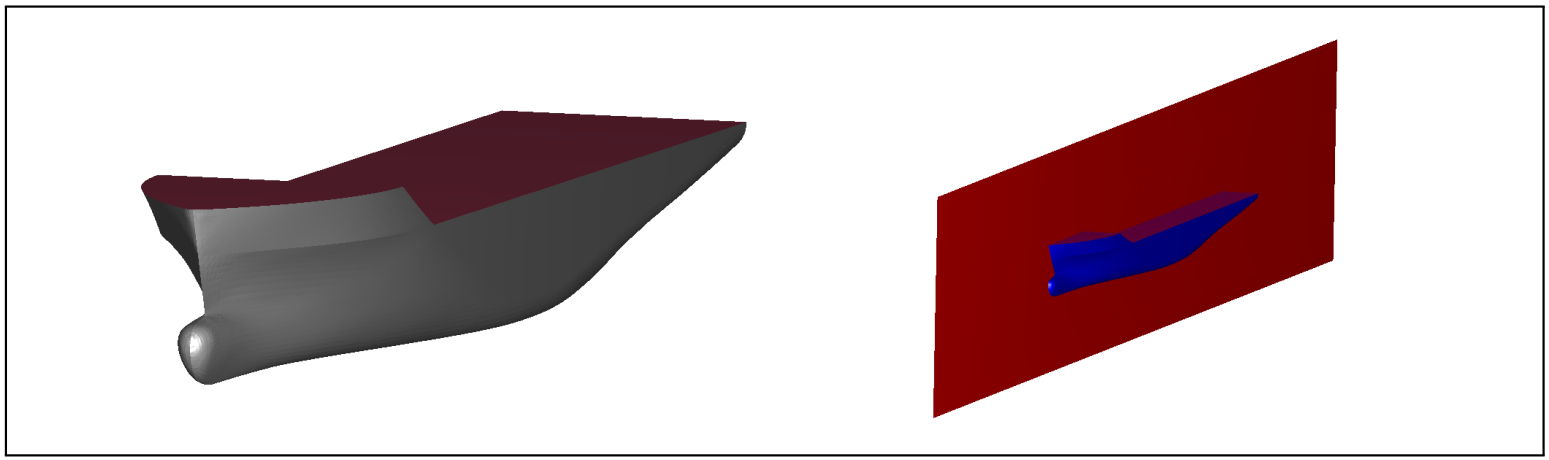
Goals to be achieved



Process sequence

1. Reading the initial ship triangulation from STL file
2. Trimming the ship hull

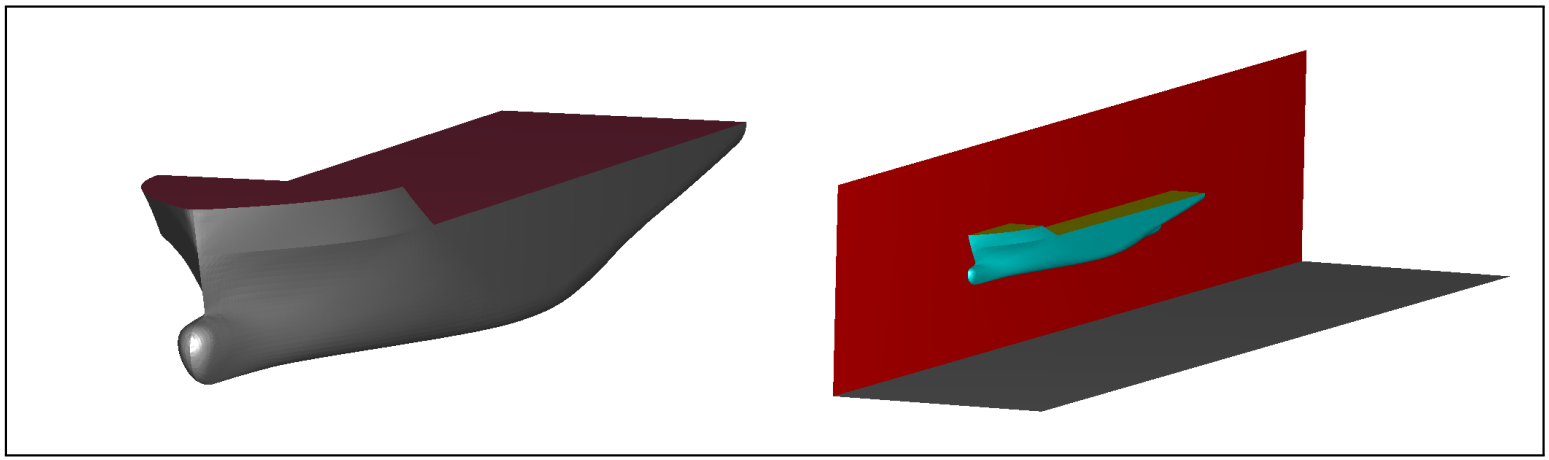
Goals to be achieved



Process sequence

1. Reading the initial ship triangulation from STL file
2. Trimming the ship hull
3. Building boundary planes

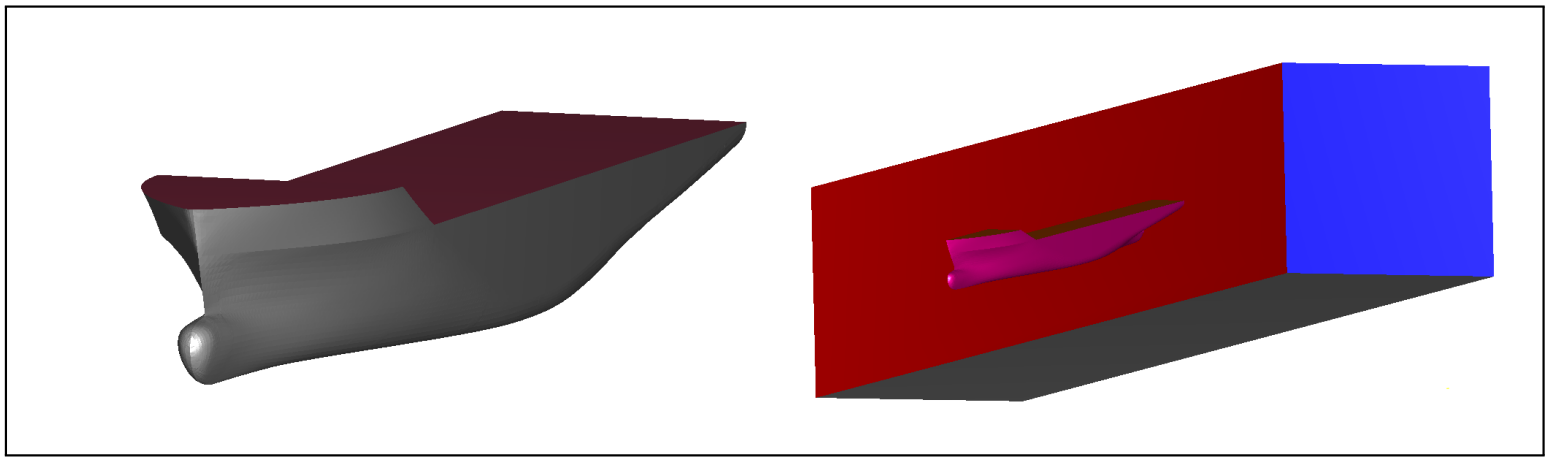
Goals to be achieved



Process sequence

1. Reading the initial ship triangulation from STL file
2. Trimming the ship hull
3. Building boundary planes

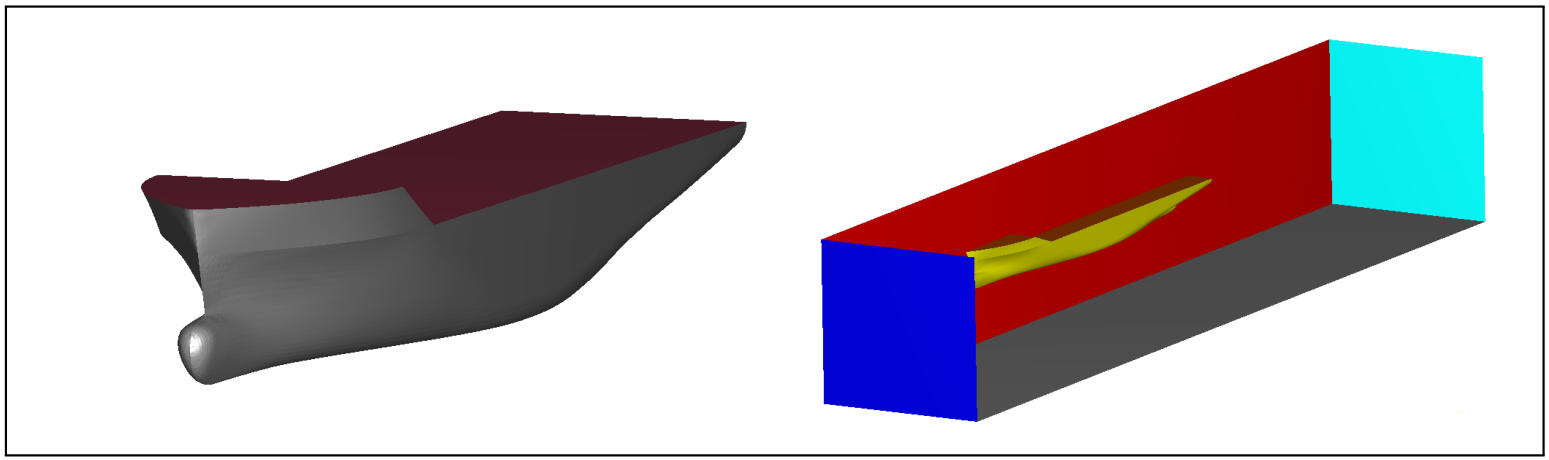
Goals to be achieved



Process sequence

1. Reading the initial ship triangulation from STL file
2. Trimming the ship hull
3. Building boundary planes

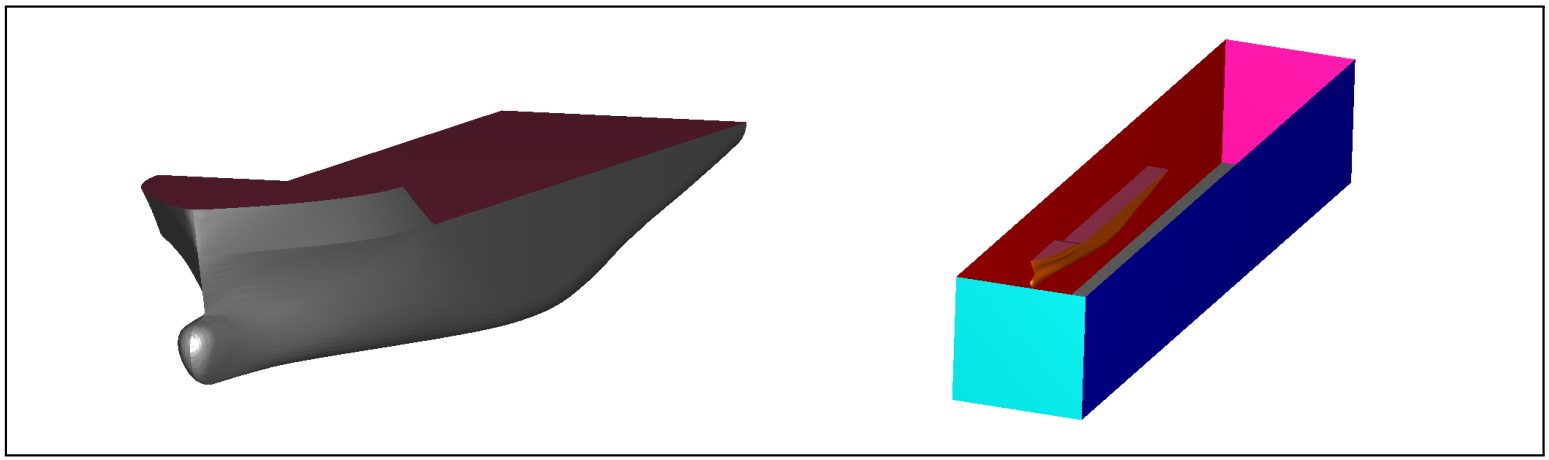
Goals to be achieved



Process sequence

1. Reading the initial ship triangulation from STL file
2. Trimming the ship hull
3. Building boundary planes

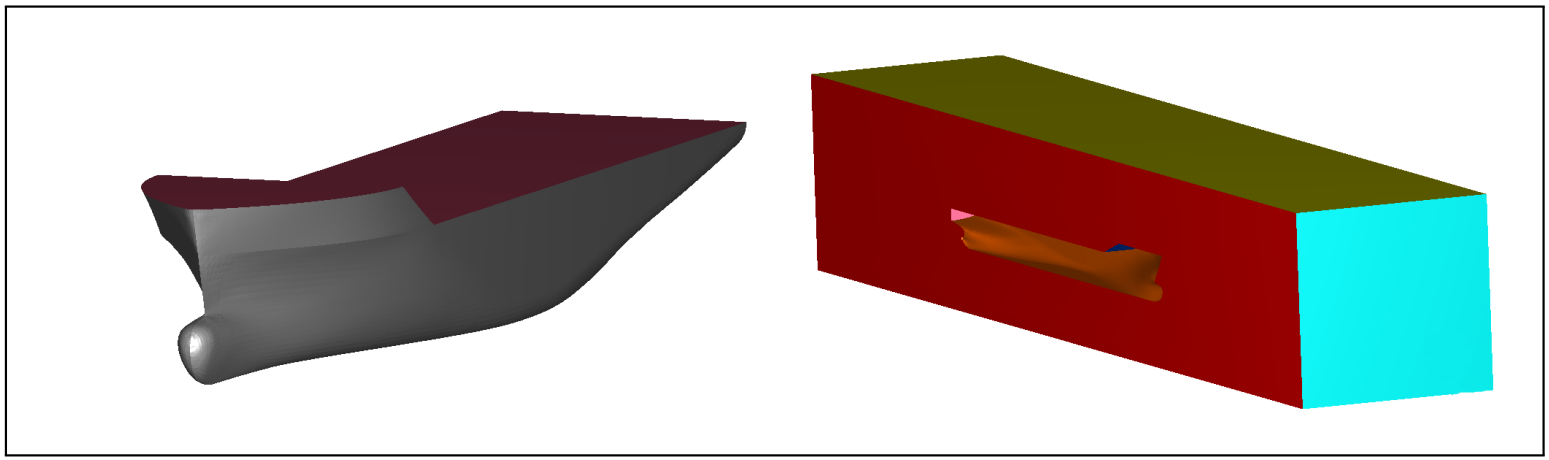
Goals to be achieved



Process sequence

1. Reading the initial ship triangulation from STL file
2. Trimming the ship hull
3. Building boundary planes

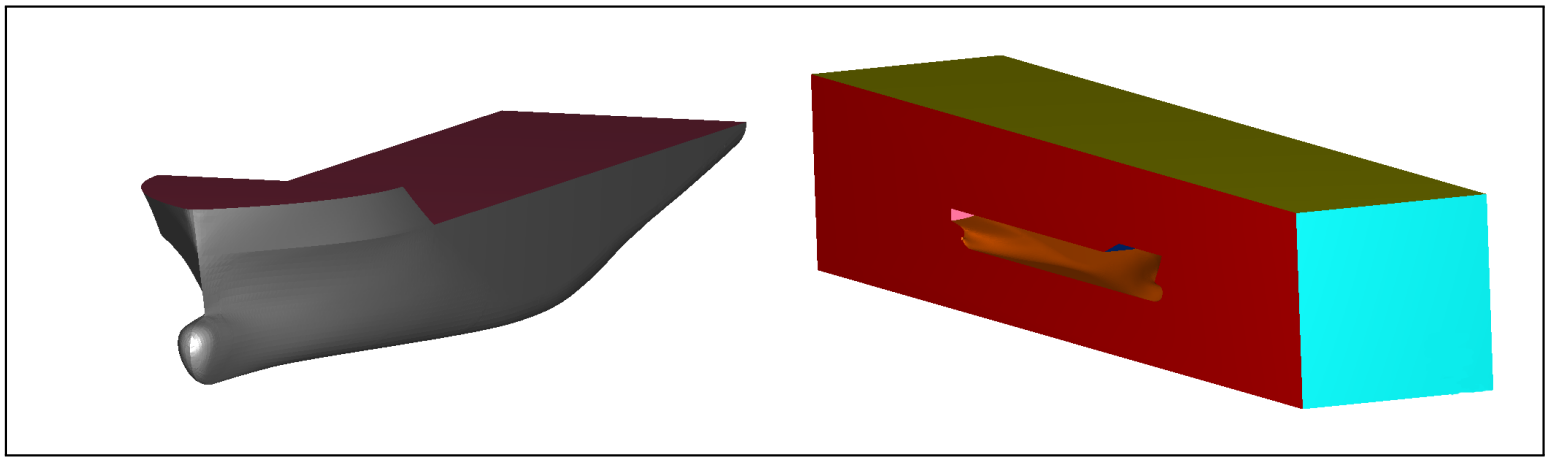
Goals to be achieved



Process sequence

1. Reading the initial ship triangulation from STL file
2. Trimming the ship hull
3. Building boundary planes

Goals to be achieved



Process sequence

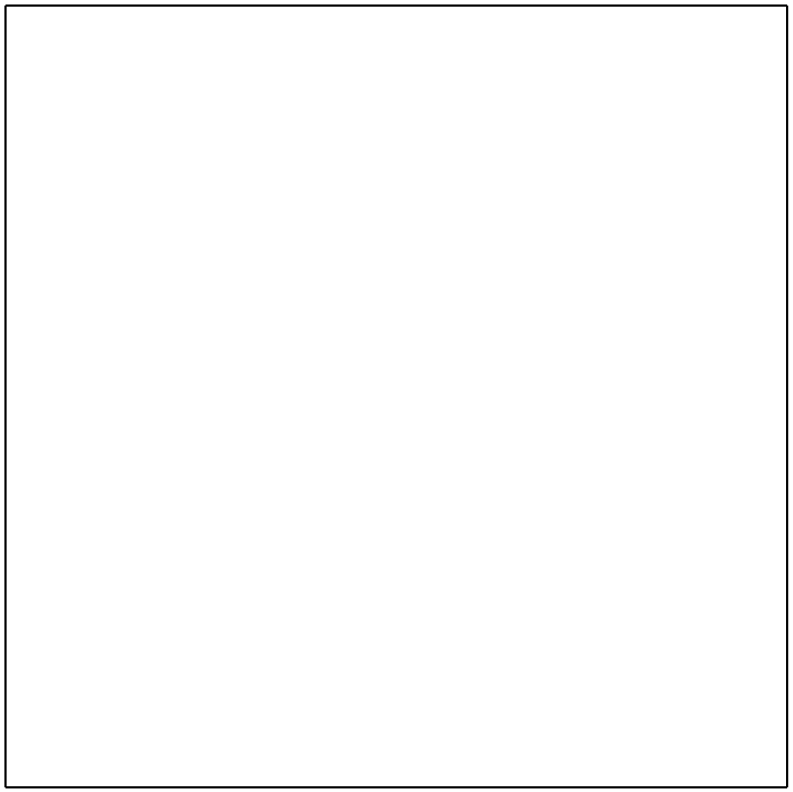
1. Reading the initial ship triangulation from STL file
2. Trimming the ship hull
3. Building boundary planes
4. Writing the boundary to new STL file

Simple Boundary Surfaces Generation

Simple Surfaces

- Inlet Plane
- Outflow Plane
- Bottom Plane
- Port side Plane

Simple Boundary Surfaces Generation



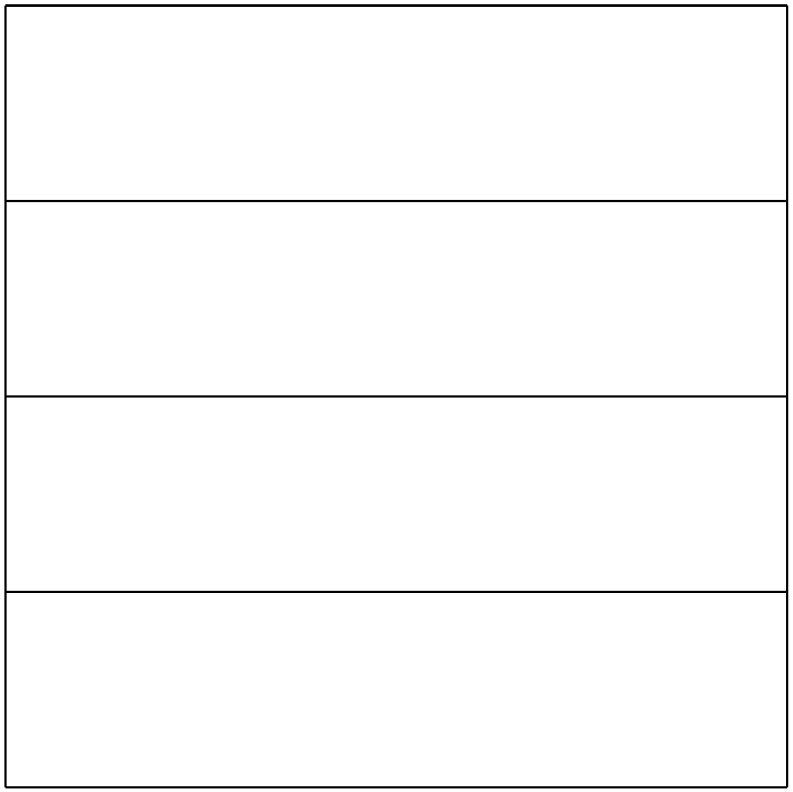
Simple Surfaces

- Inlet Plane
- Outflow Plane
- Bottom Plane
- Port side Plane

Generation Logic

1. Defining of the surface boundaries

Simple Boundary Surfaces Generation



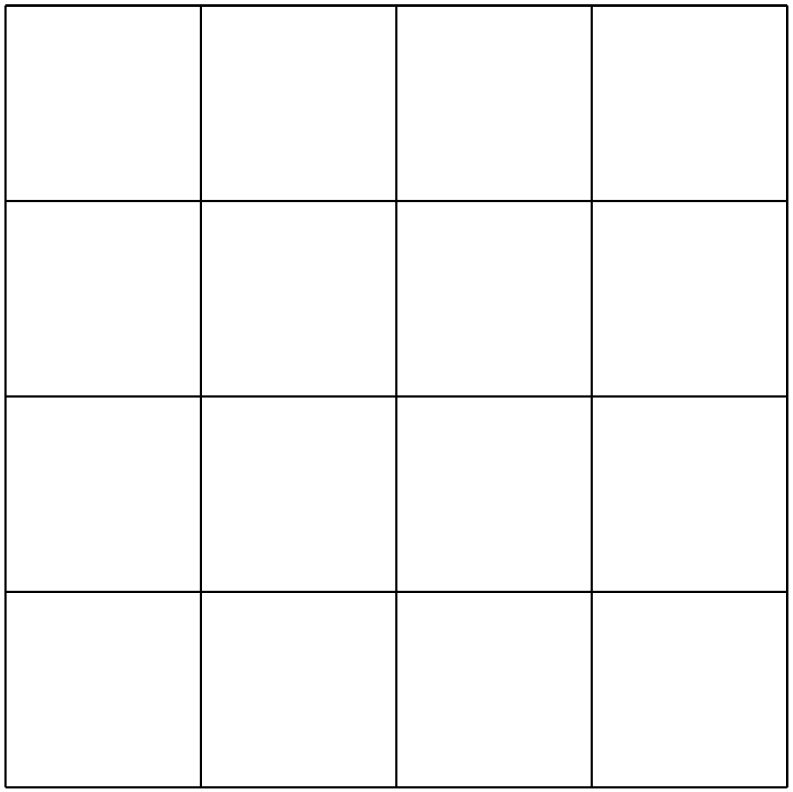
Simple Surfaces

- Inlet Plane
- Outflow Plane
- Bottom Plane
- Port side Plane

Generation Logic

1. Defining of the surface boundaries
2. Dividing the surface in the first direction

Simple Boundary Surfaces Generation



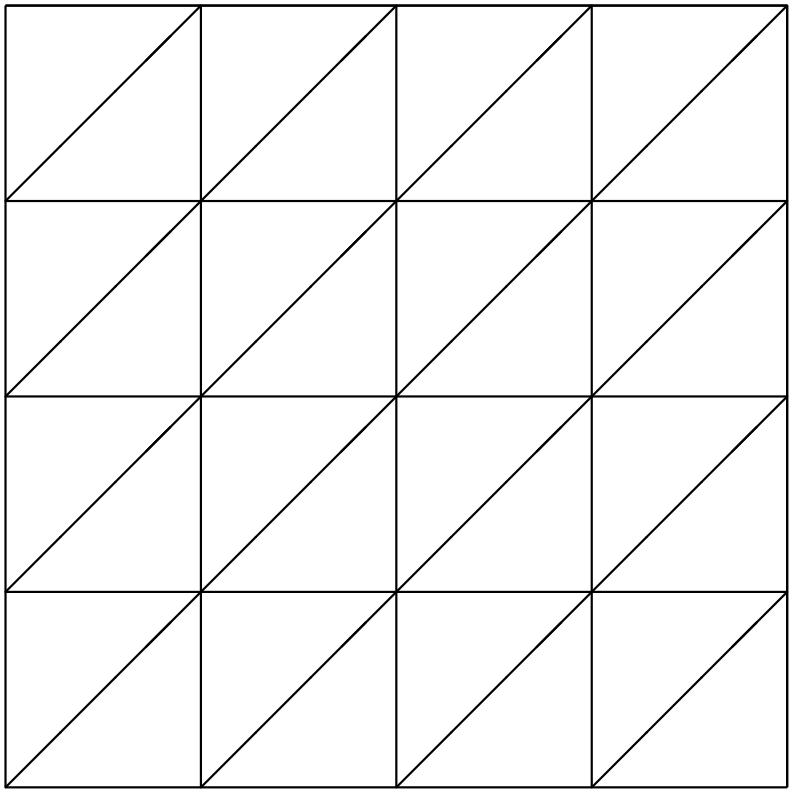
Simple Surfaces

- Inlet Plane
- Outflow Plane
- Bottom Plane
- Port side Plane

Generation Logic

1. Defining of the surface boundaries
2. Dividing the surface in the first direction
3. Dividing the surface in the second direction

Simple Boundary Surfaces Generation



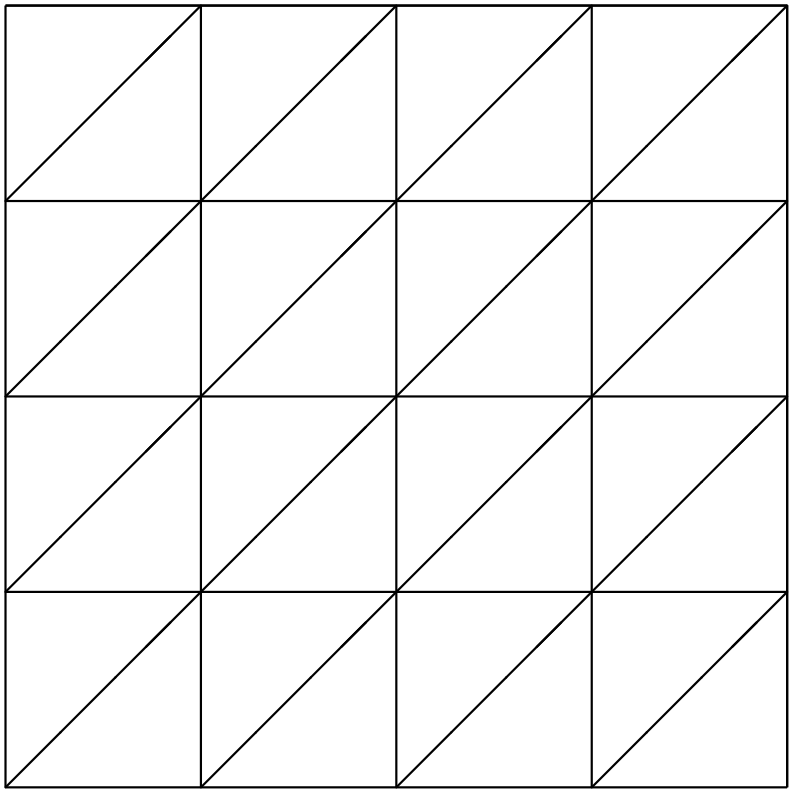
Simple Surfaces

- Inlet Plane
- Outflow Plane
- Bottom Plane
- Port side Plane

Generation Logic

1. Defining of the surface boundaries
2. Dividing the surface in the first direction
3. Dividing the surface in the second direction
4. Dividing the rectangles into triangles

Simple Boundary Surfaces Generation



Simple Surfaces

- Inlet Plane
- Outflow Plane
- Bottom Plane
- Port side Plane

Generation Logic

1. Defining of the surface boundaries
2. Dividing the surface in the first direction
3. Dividing the surface in the second direction
4. Dividing the rectangles into triangles
5. Writing these triangles to STL file

Complex Boundary Surfaces Generation

Complex Surfaces

Waterline and Centerline surfaces are assumed as complex

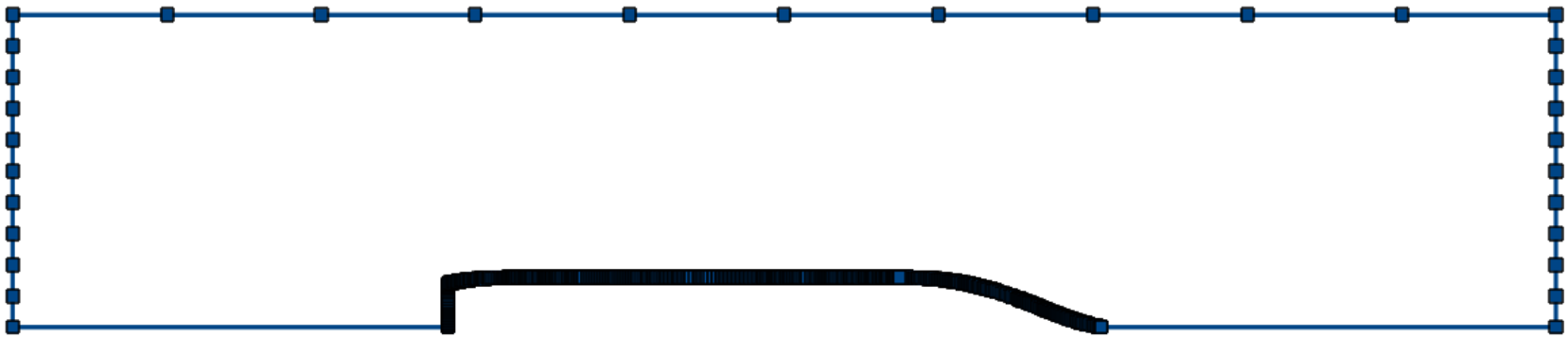
Complex Boundary Surfaces Generation

Complex Surfaces

Waterline and Centerline surfaces are assumed as complex

Used Approach

1. Defining complex surface contour



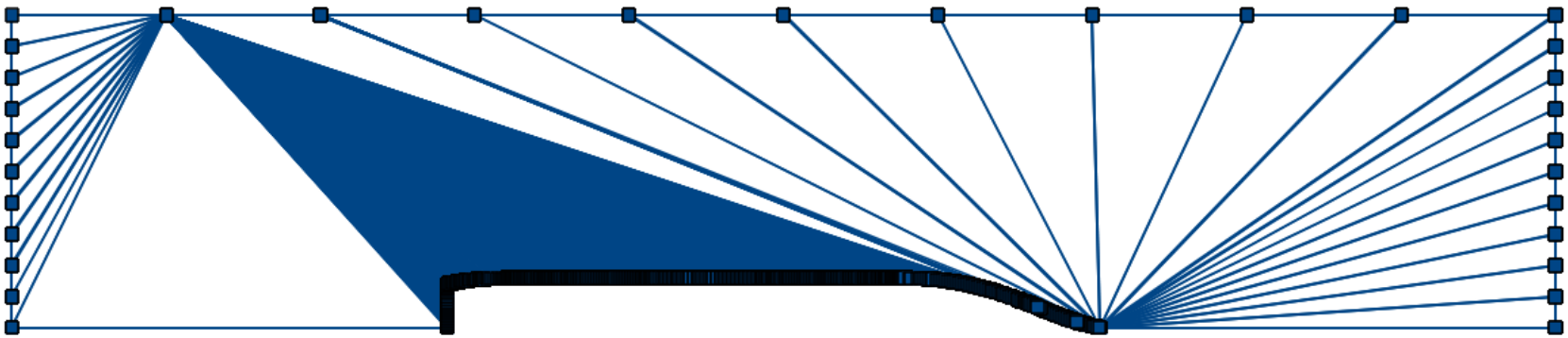
Complex Boundary Surfaces Generation

Complex Surfaces

Waterline and Centerline surfaces are assumed as complex

Used Approach

1. Defining complex surface contour
2. Adding all non-intersecting diagonals



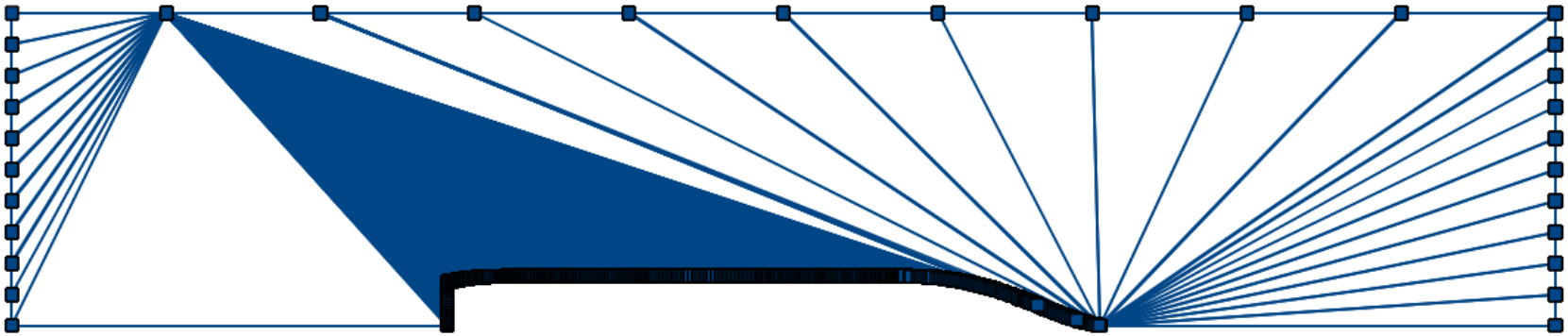
Complex Boundary Surfaces Generation

Complex Surfaces

Waterline and Centerline surfaces are assumed as complex

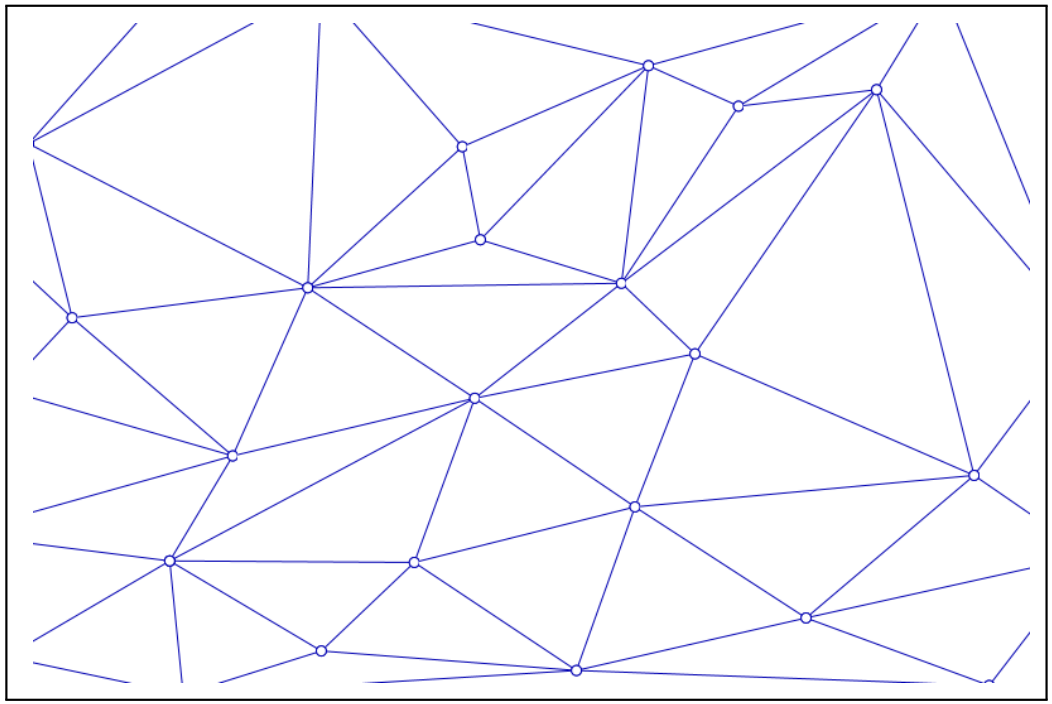
Used Approach

1. Defining complex surface contour
2. Adding all non-intersecting diagonals
3. Writing result triangles to STL file



Hull Cutting

Procedure for trimming the initial hull with waterline and/or centerline planes

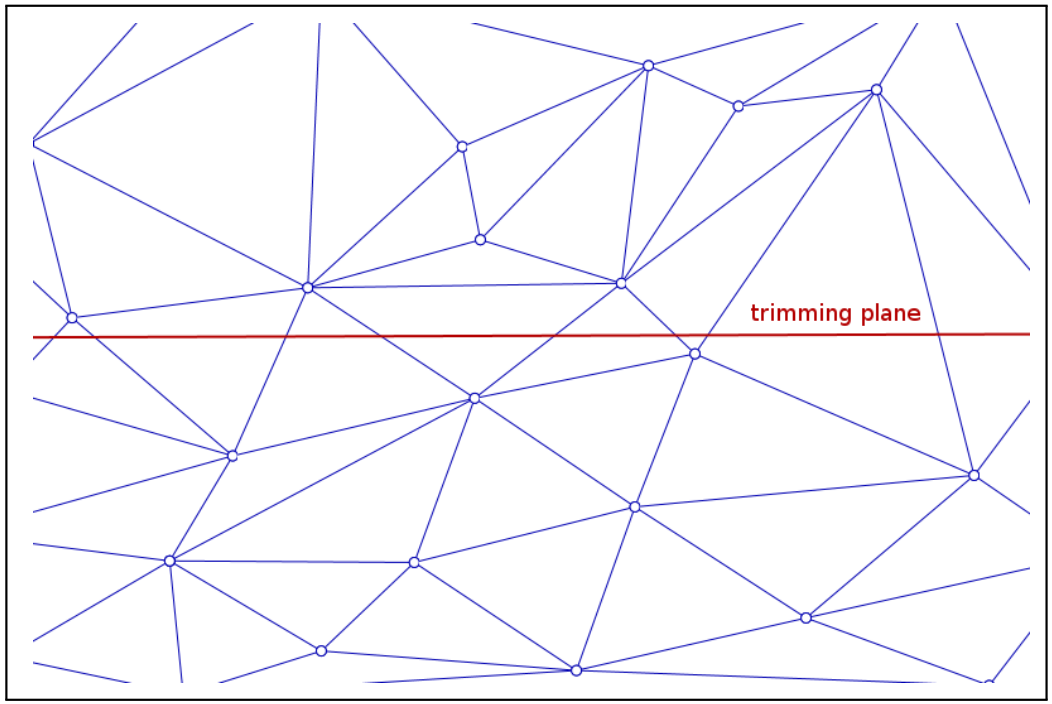


Algorithm logic

1. Defining the initial hull triangulation

Hull Cutting

Procedure for trimming the initial hull with waterline and/or centerline planes

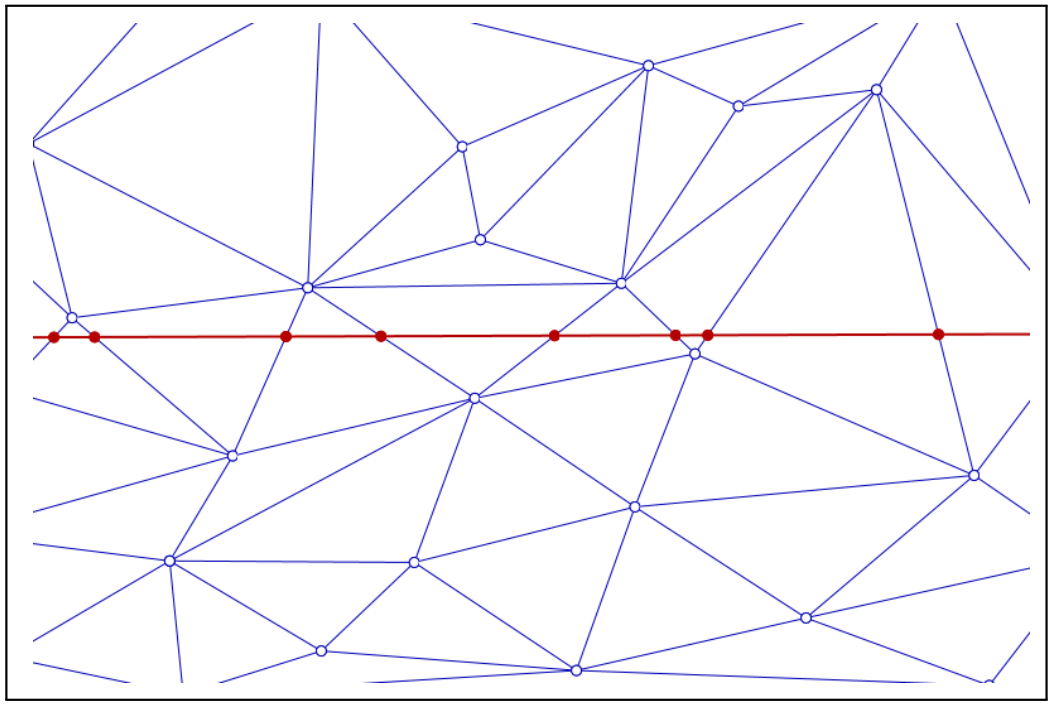


Algorithm logic

1. Defining the initial hull triangulation
2. Defining the cutting plane

Hull Cutting

Procedure for trimming the initial hull with waterline and/or centerline planes

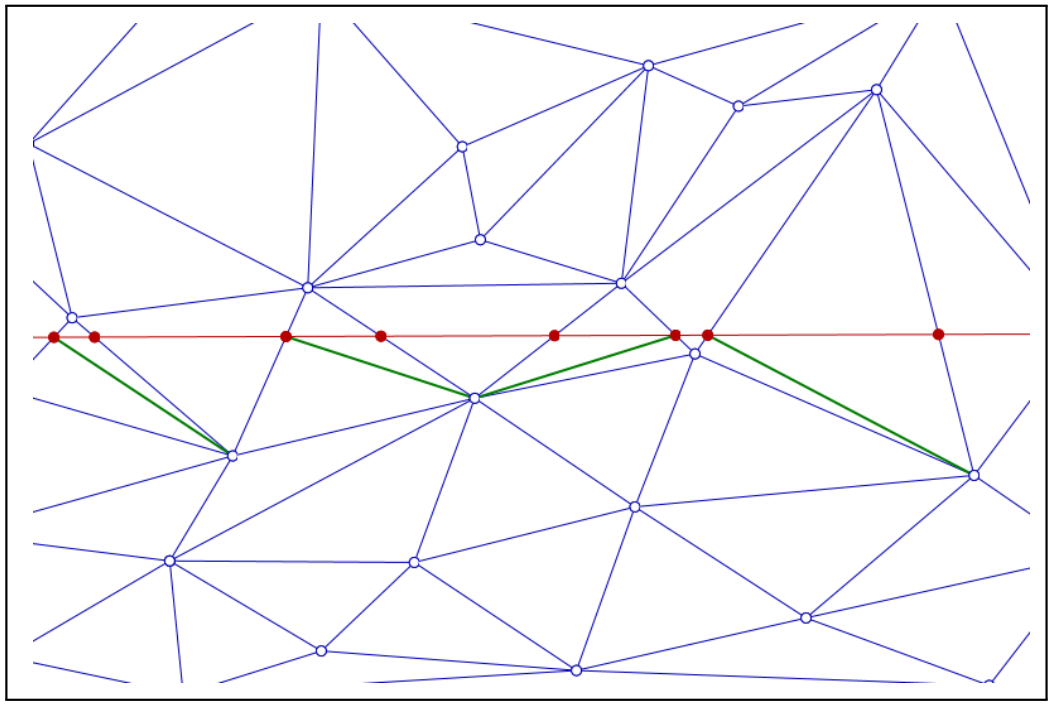


Algorithm logic

1. Defining the initial hull triangulation
2. Defining the cutting plane
3. Calculating all intersections of initial triangulation edges with the cutting plane

Hull Cutting

Procedure for trimming the initial hull with waterline and/or centerline planes

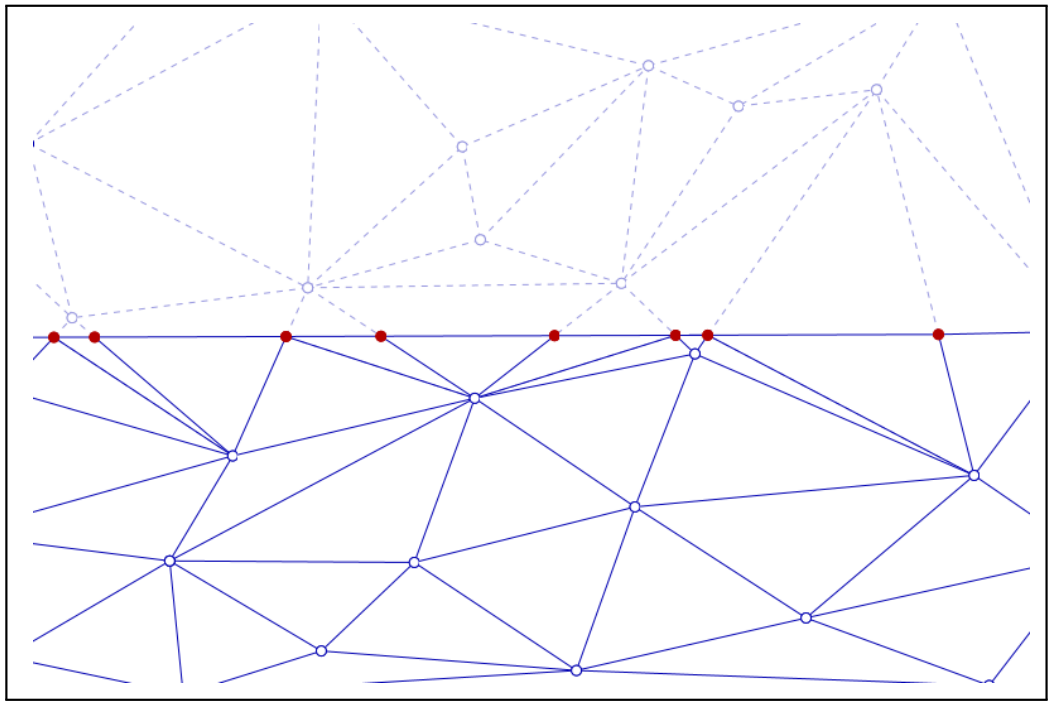


Algorithm logic

1. Defining the initial hull triangulation
2. Defining the cutting plane
3. Calculating all intersections of initial triangulation edges with the cutting plane
4. Converting of all trapezoids to triangles

Hull Cutting

Procedure for trimming the initial hull with waterline and/or centerline planes

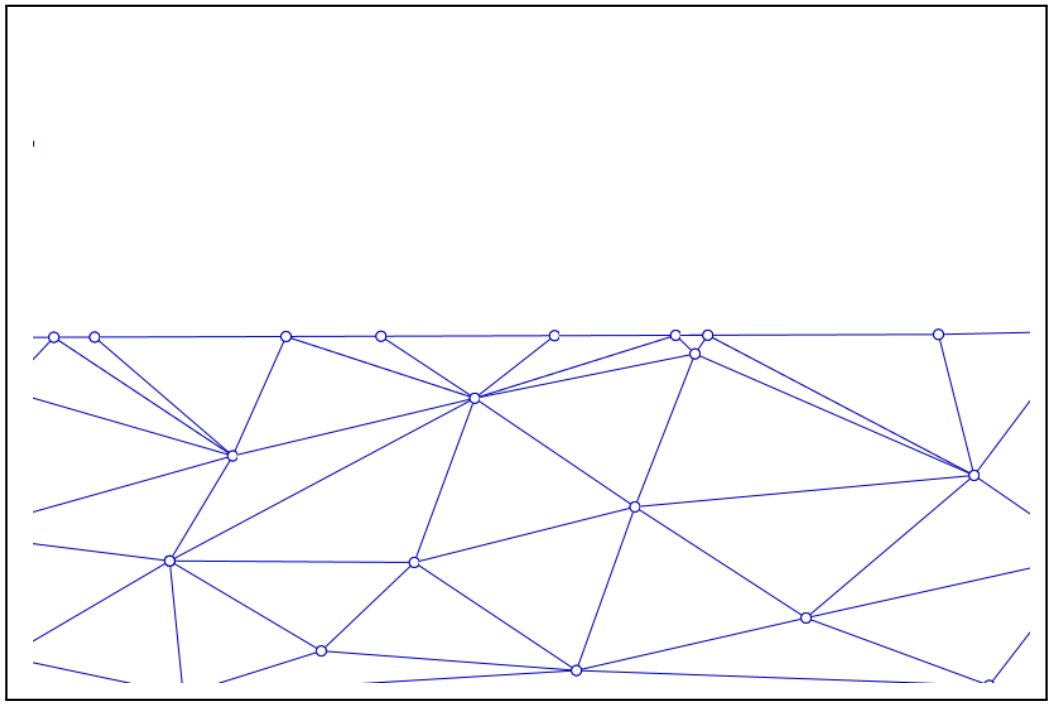


Algorithm logic

1. Defining the initial hull triangulation
2. Defining the cutting plane
3. Calculating all intersections of initial triangulation edges with the cutting plane
4. Converting of all trapezoids to triangles
5. Sorting of triangles depending on their position in the plane

Hull Cutting

Procedure for trimming the initial hull with waterline and/or centerline planes



Algorithm logic

1. Defining the initial hull triangulation
2. Defining the cutting plane
3. Calculating all intersections of initial triangulation edges with the cutting plane
4. Converting of all trapezoids to triangles
5. Sorting of triangles depending on their position in the plane
6. Writing the result triangulation to STL file

Topology Detection

Procedure for detection of rapid geometry changes

Nature of the algorithm

The topology detection algorithm is implemented as a sorting algorithm with special separation criteria.

Topology Detection

Procedure for detection of rapid geometry changes

Nature of the algorithm

The topology detection algorithm is implemented as a sorting algorithm with special separation criteria.

Separation Criteria

Normal Vector Components			
Region	x - component	y - component	z - component
Transom	$x \neq 0$	$y \approx 0$	$z \approx 0$
Flat Boards	$x \approx 0$	$y \neq 0$	$z \approx 0$
Flat Bottom	$x \approx 0$	$y \approx 0$	$z \neq 0$

Topology Detection

Procedure for detection of rapid geometry changes

Nature of the algorithm

The topology detection algorithm is implemented as a sorting algorithm with special separation criteria.

Separation Criteria

Normal Vector Components			
Region	x - component	y - component	z - component
Transom	$x \neq 0$	$y \approx 0$	$z \approx 0$
Flat Boards	$x \approx 0$	$y \neq 0$	$z \approx 0$
Flat Bottom	$x \approx 0$	$y \approx 0$	$z \neq 0$

Disadvantages

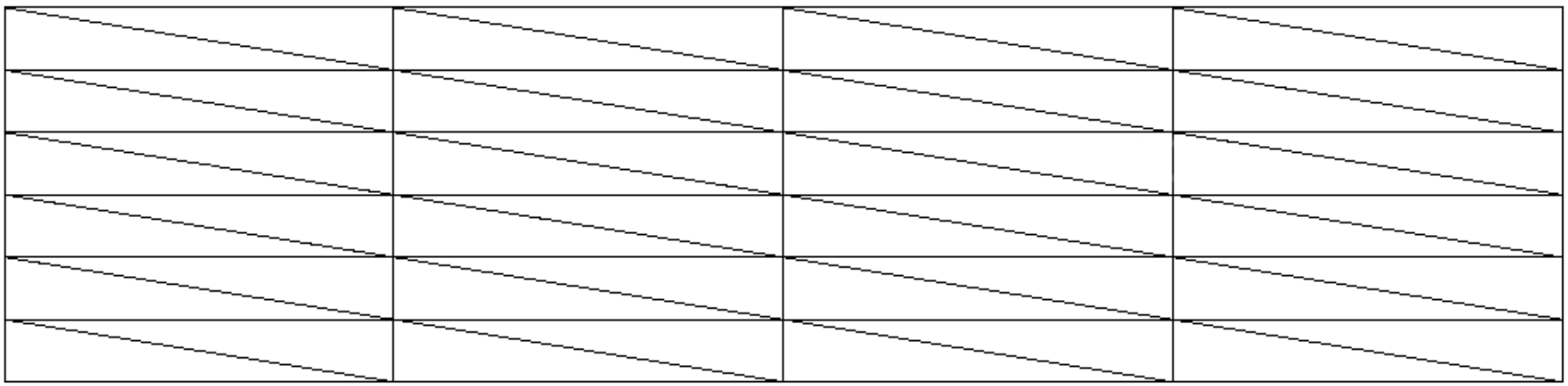
- Works only for planes parallel to the base planes

Boolean Intersection

Another approach for dealing with the complex surfaces

Algorithm Logic

1. All domain boundary surfaces are built as simple

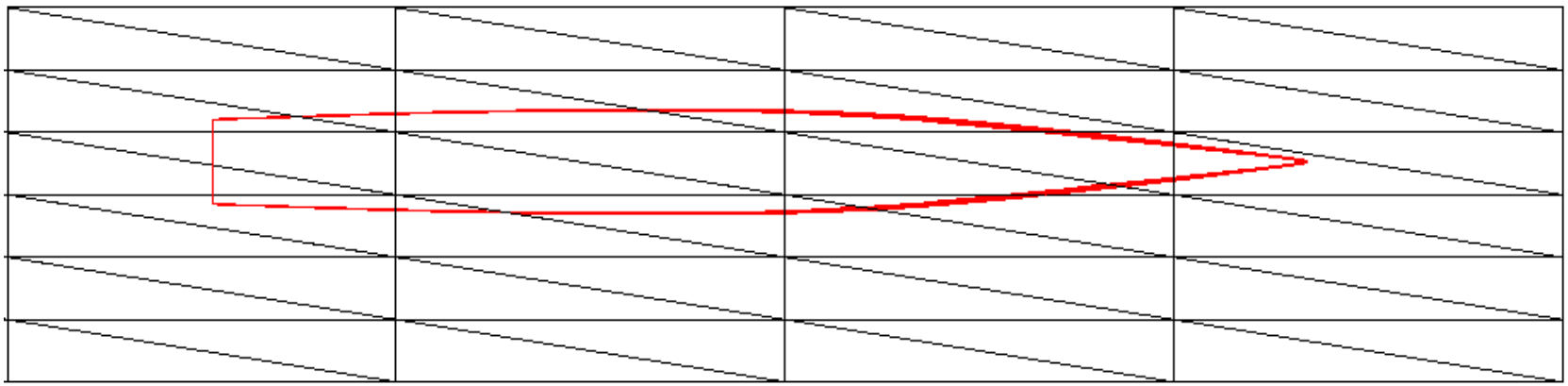


Boolean Intersection

Another approach for dealing with the complex surfaces

Algorithm Logic

1. All domain boundary surfaces are built as simple
2. Hull boundary edges are sorted in native order

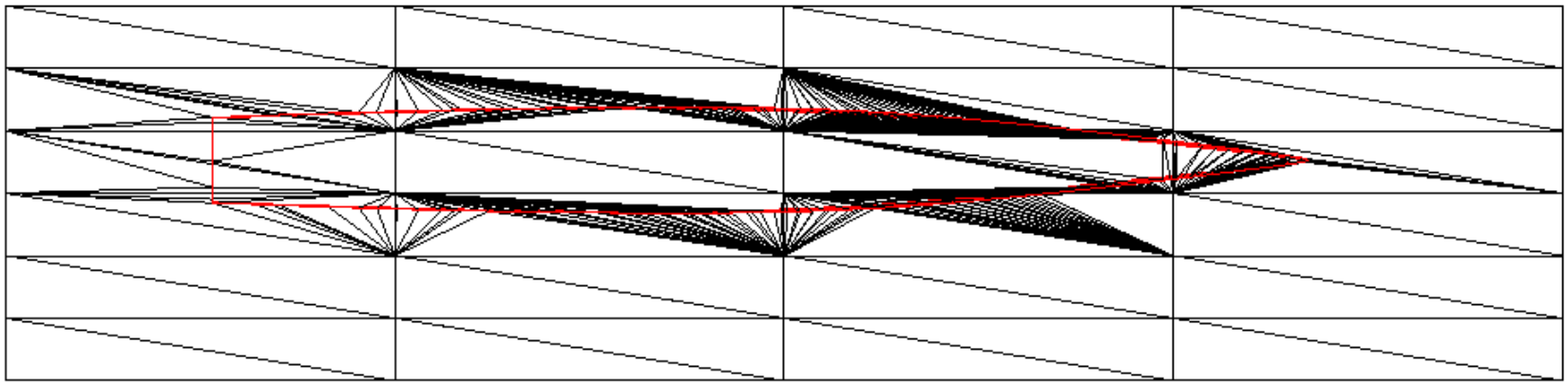


Boolean Intersection

Another approach for dealing with the complex surfaces

Algorithm Logic

1. All domain boundary surfaces are built as simple
2. Hull boundary edges are sorted in native order
3. Adaptation of the boundary surface triangulation to the hull boundary edges

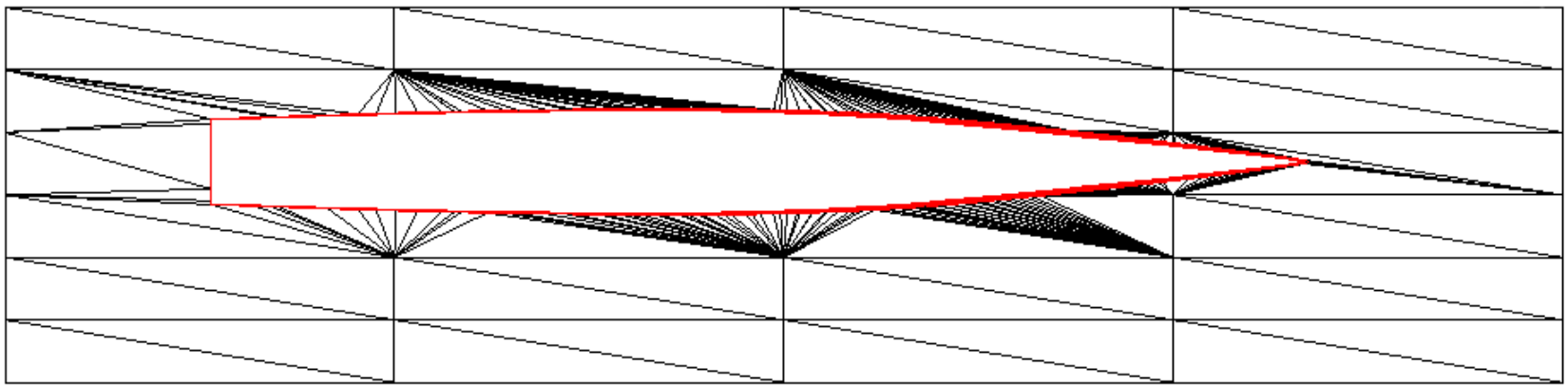


Boolean Intersection

Another approach for dealing with the complex surfaces

Algorithm Logic

1. All domain boundary surfaces are built as simple
2. Hull boundary edges are sorted in native order
3. Adaptation of the boundary surface triangulation to the hull boundary edges
4. Removing of the enclosed by the hull contour triangles

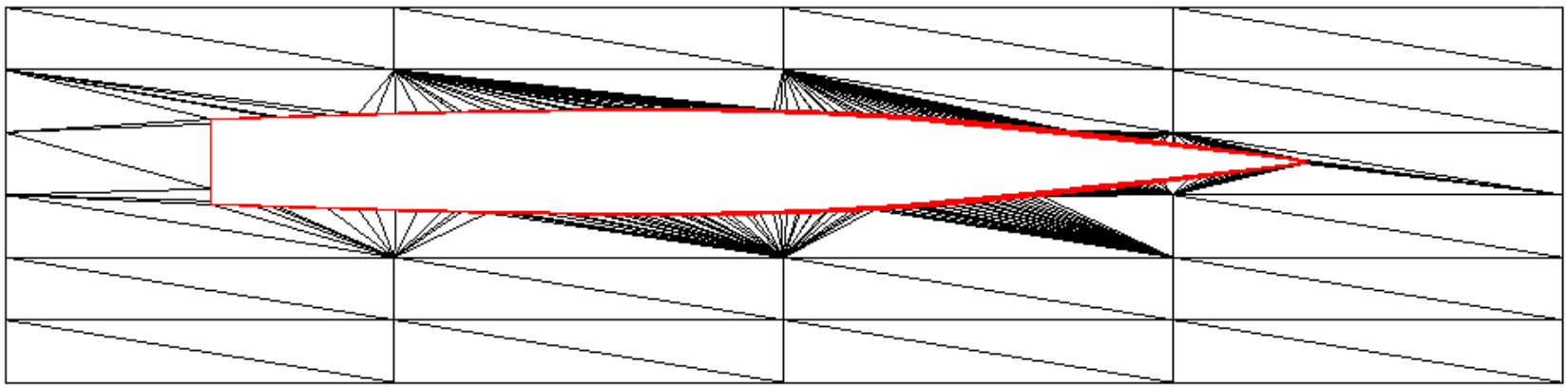


Boolean Intersection

Another approach for dealing with the complex surfaces

Algorithm Logic

1. All domain boundary surfaces are built as simple
2. Hull boundary edges are sorted in native order
3. Adaptation of the boundary surface triangulation to the hull boundary edges
4. Removing of the enclosed by the hull contour triangles
5. Writing the result triangulation to STL file



MOERI KCS Container Ship

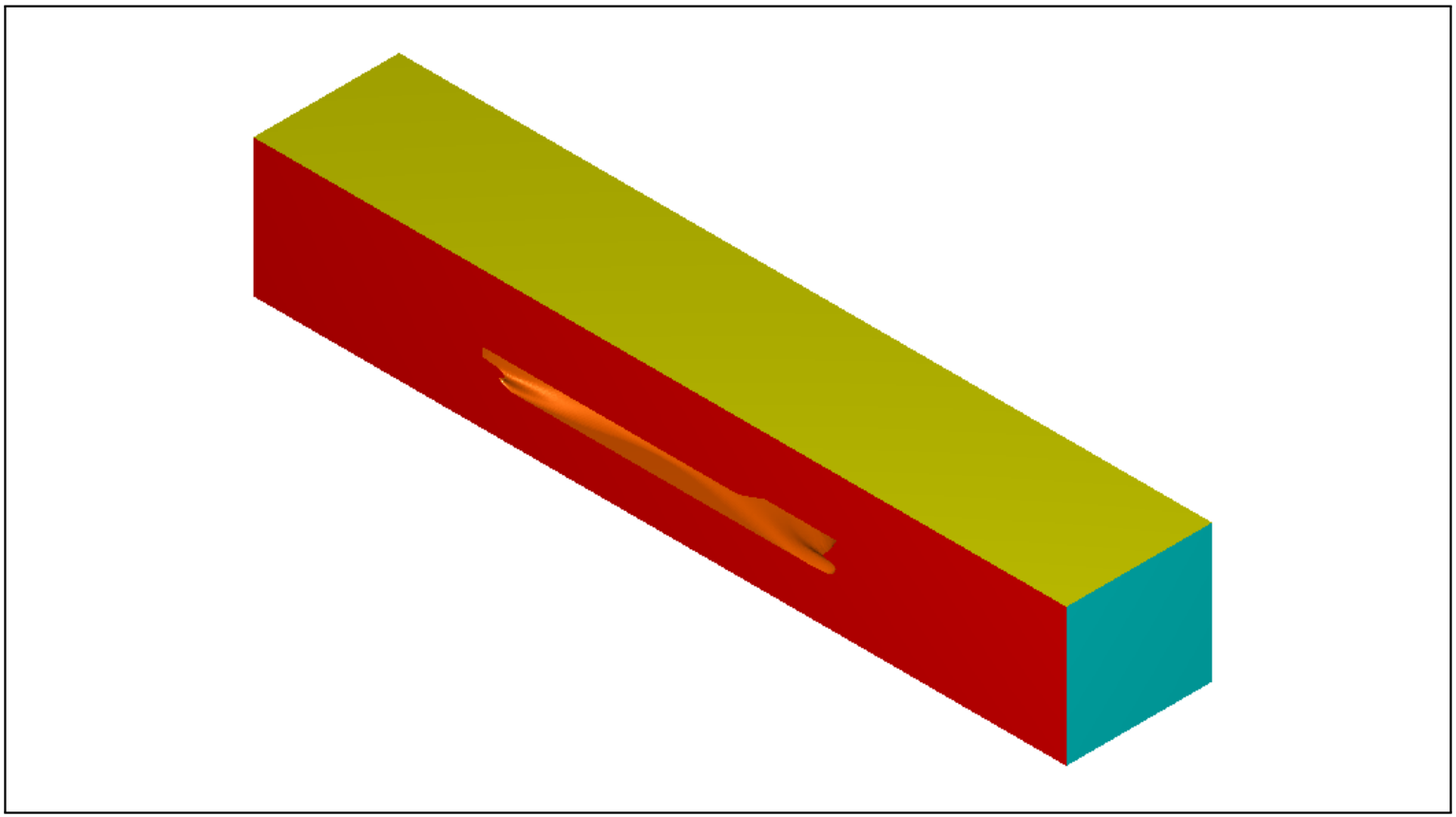


Figure: Perspective view of the entire domain in HEXPRESS

MOERI KCS Container Ship

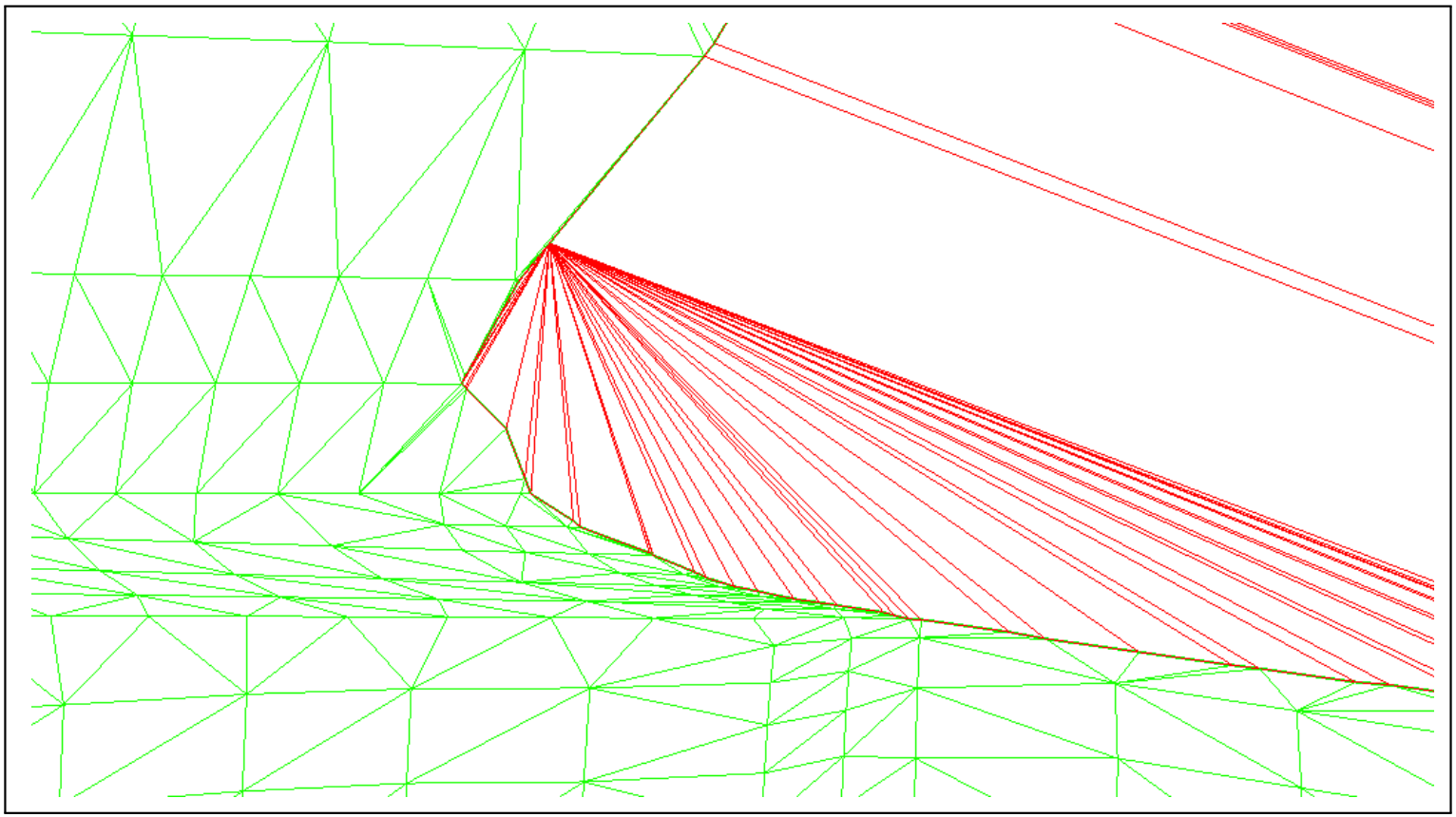


Figure: Detailed view of the bow region in HEXPRESS

MOERI KCS Container Ship

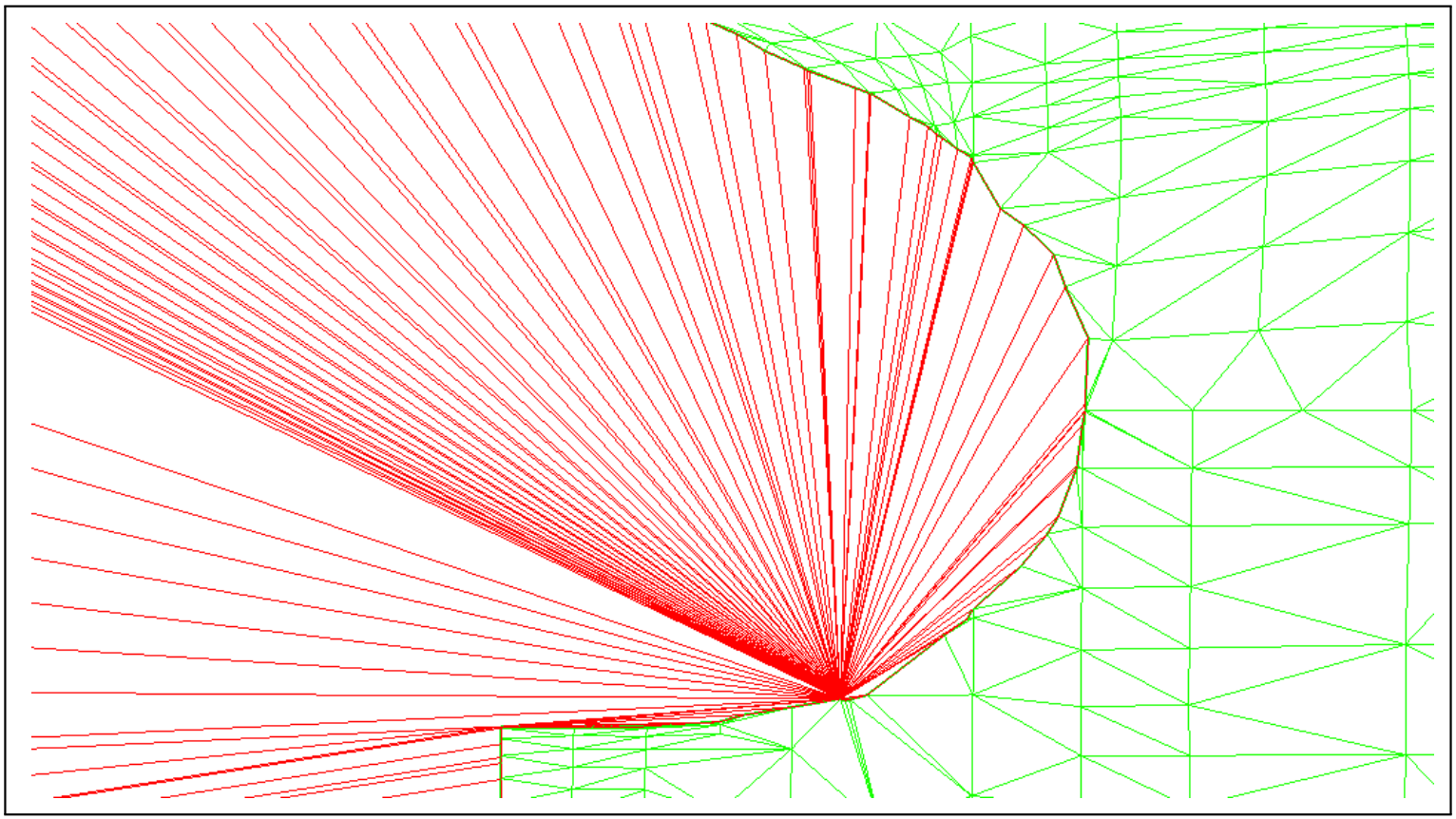


Figure: Detailed view of the stern region in HEXPRESS

Sample Submarine

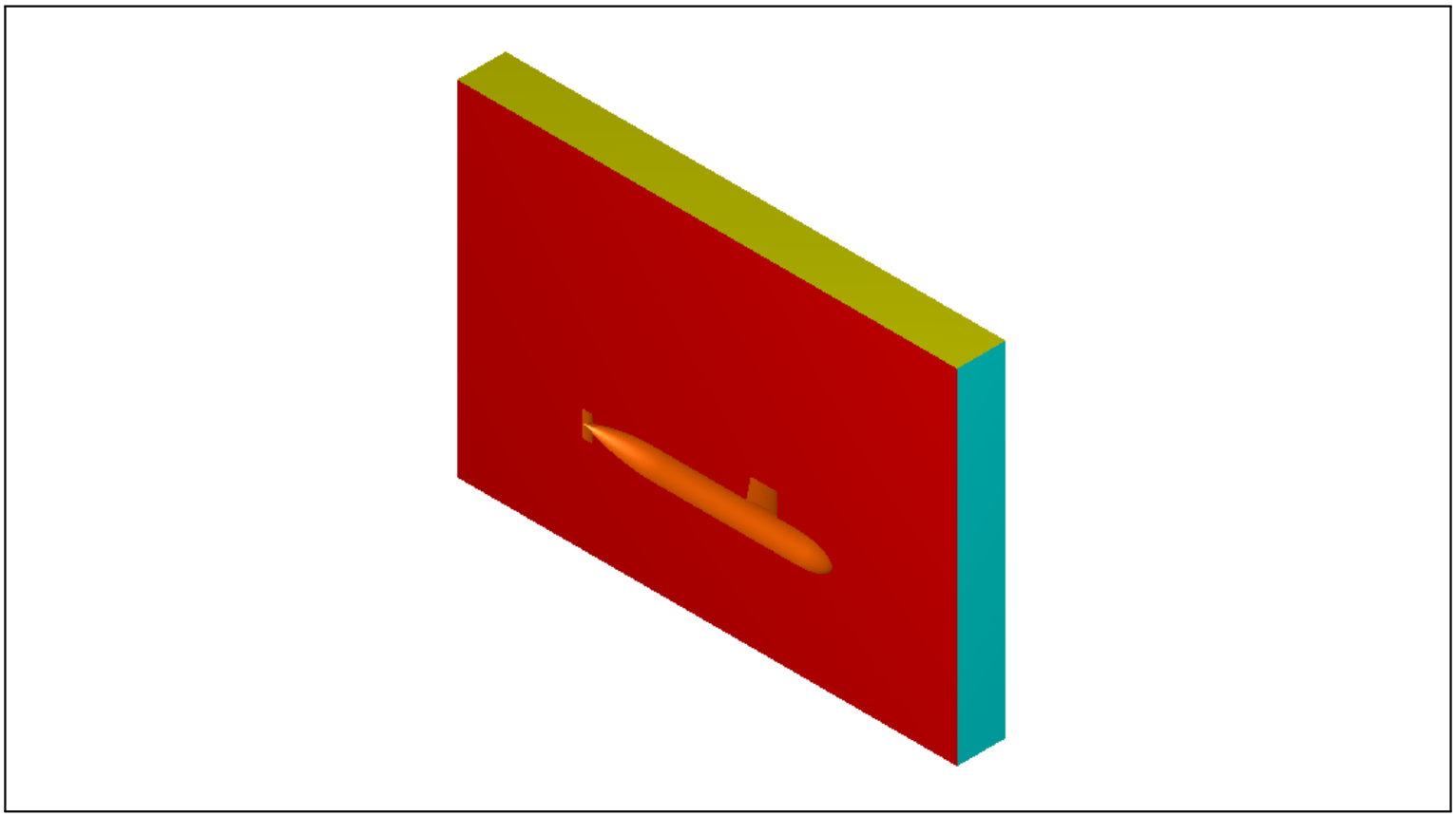


Figure: Perspective view of the entire domain in HEXPRESS

Sample Submarine

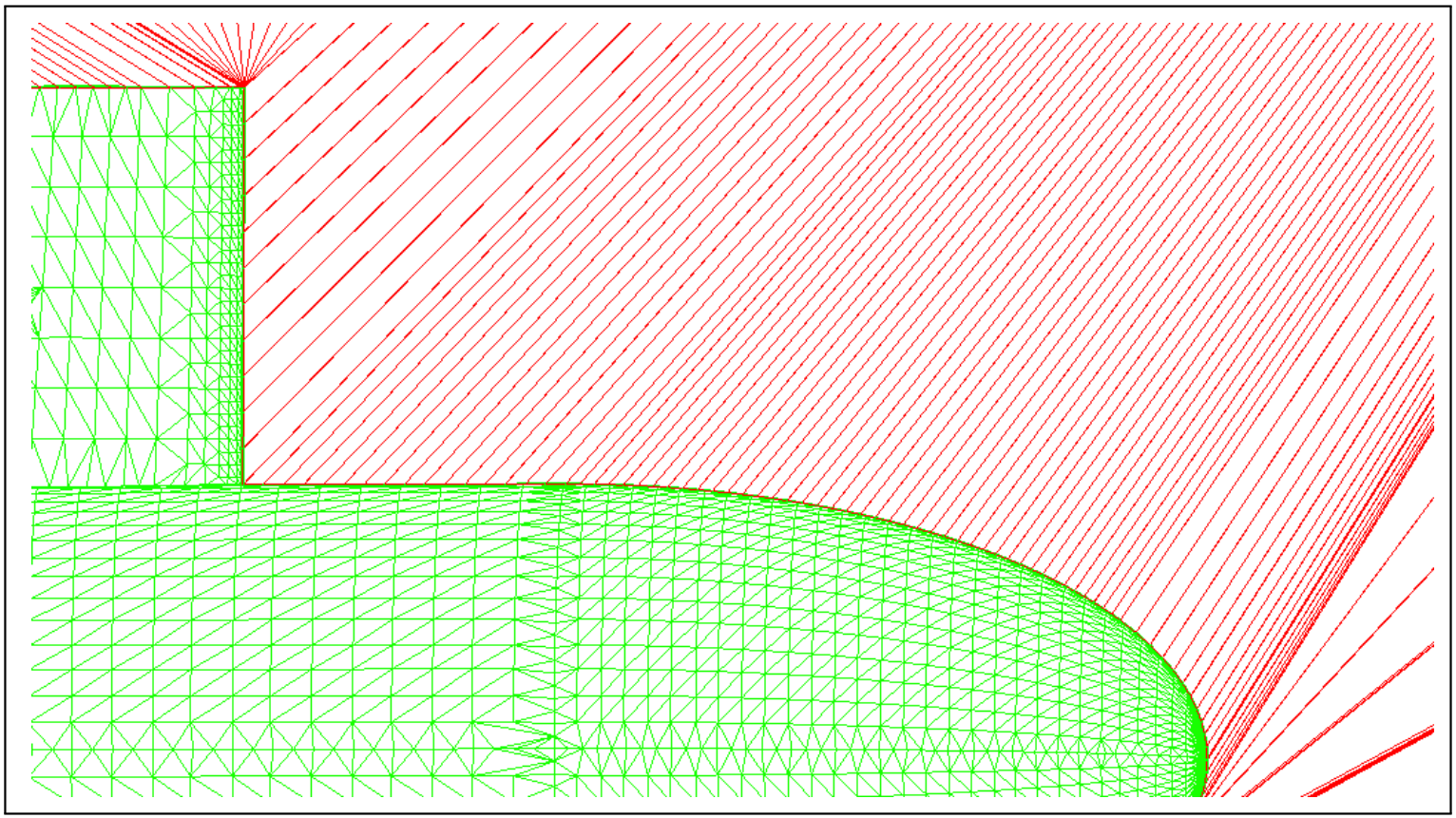


Figure: Detailed view of the bow region in HEXPRESS

Sample Submarine

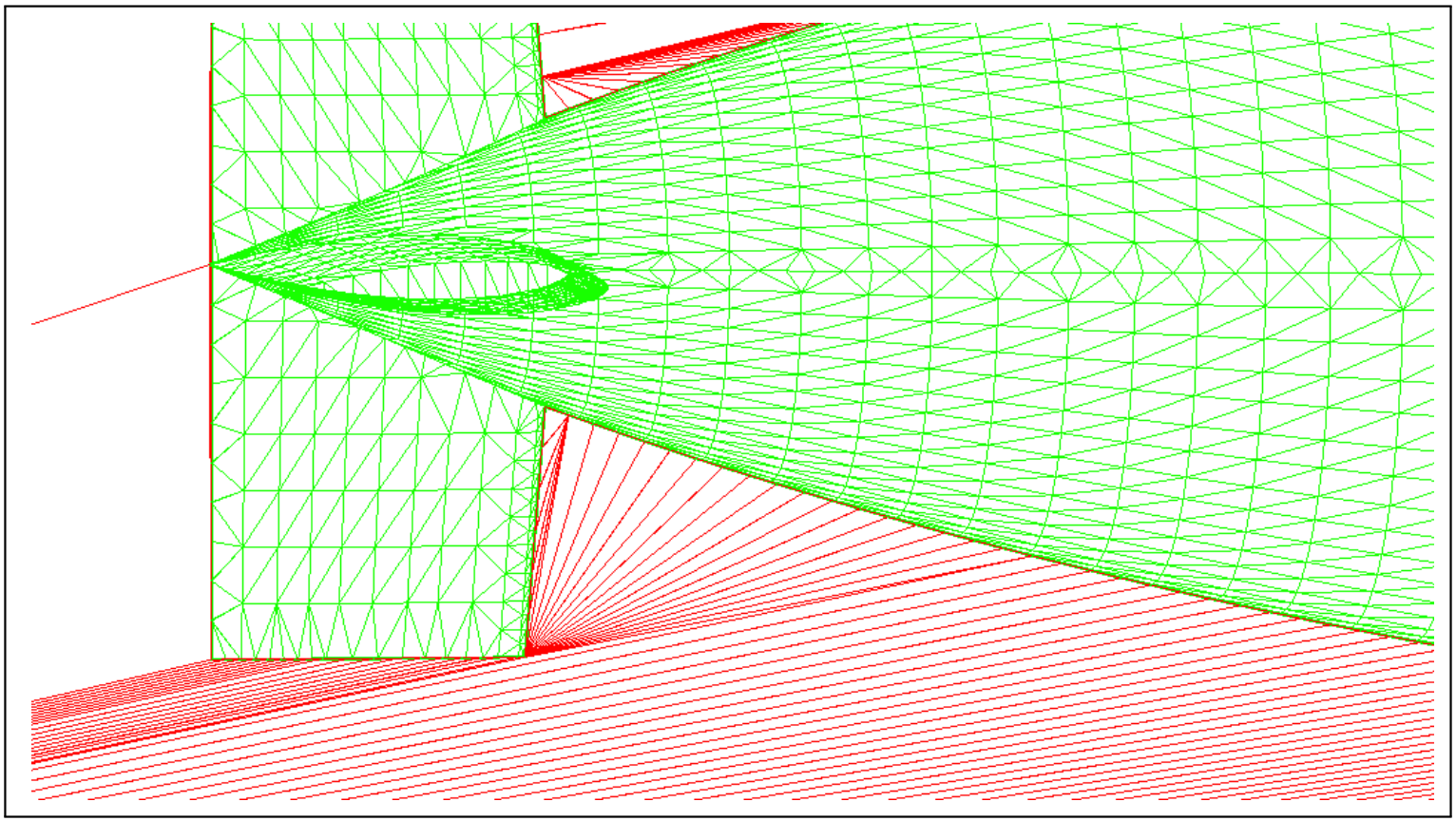


Figure: Detailed view of the stern region in HEXPRESS

Topology Detection

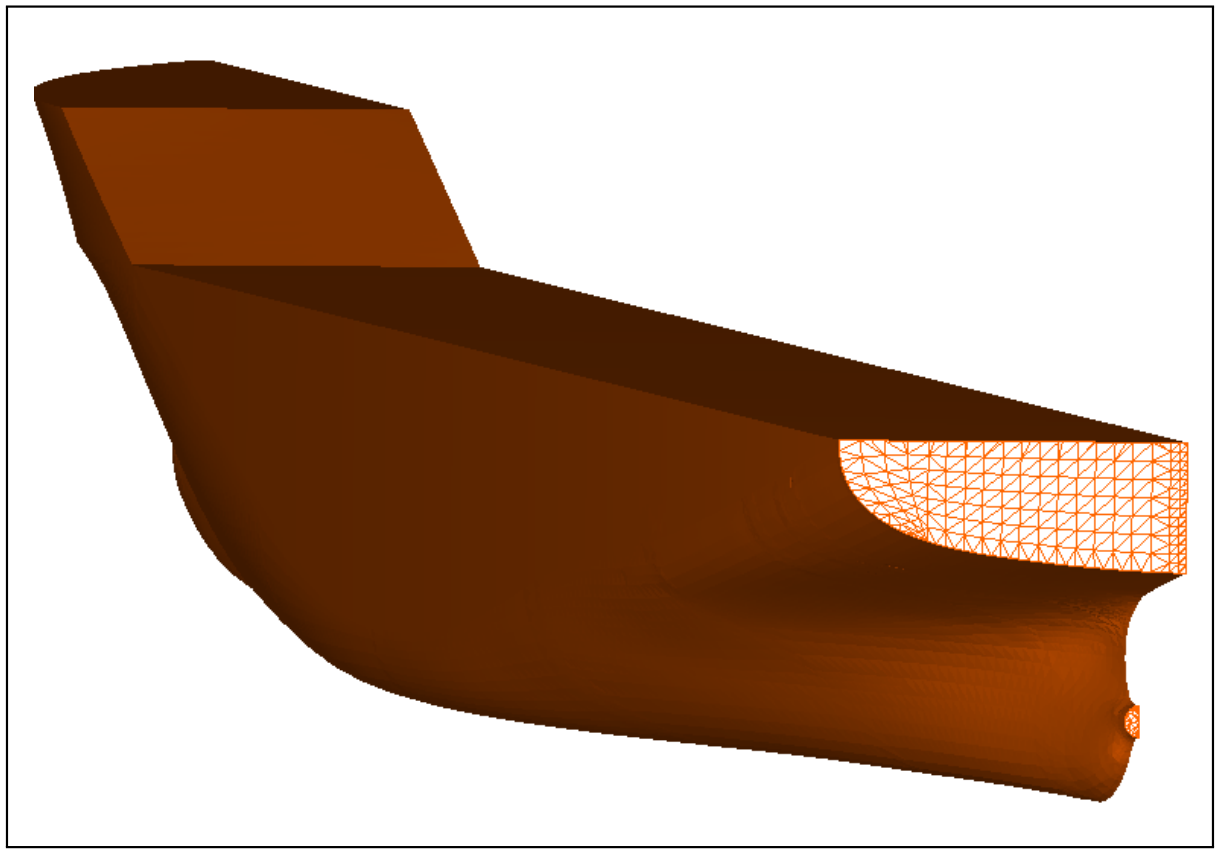


Figure: Transom Detection

Topology Detection

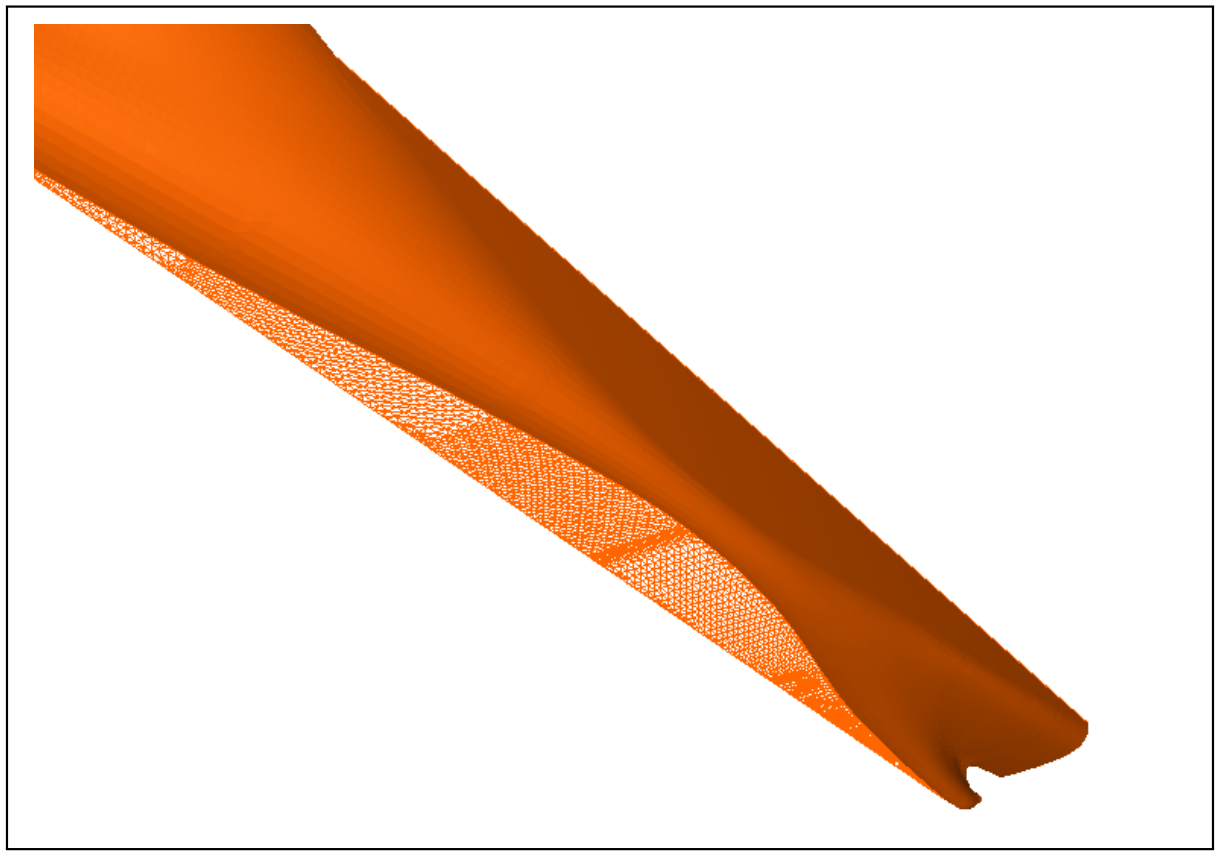


Figure: Flat Bottom Detection

Topology Detection

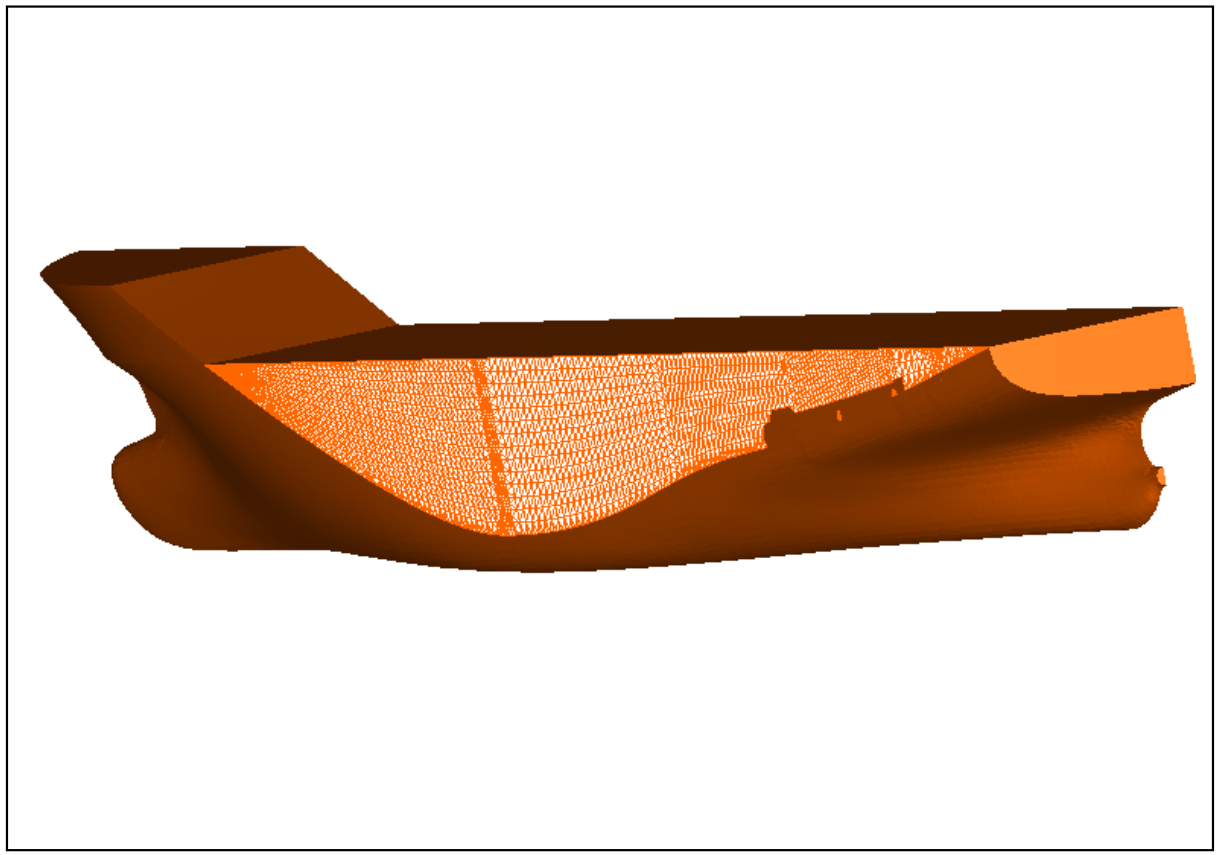


Figure: Flat Boards Detection

Conclusions

- The application works with wide range of ships and scenarios
- Created boundary domain is completely watertight and compatible with HEXPRESS
- The topology detection algorithm works when the ship geometry is defined accurately

Conclusions

- The application works with wide range of ships and scenarios
- Created boundary domain is completely watertight and compatible with HEXPRESS
- The topology detection algorithm works when the ship geometry is defined accurately

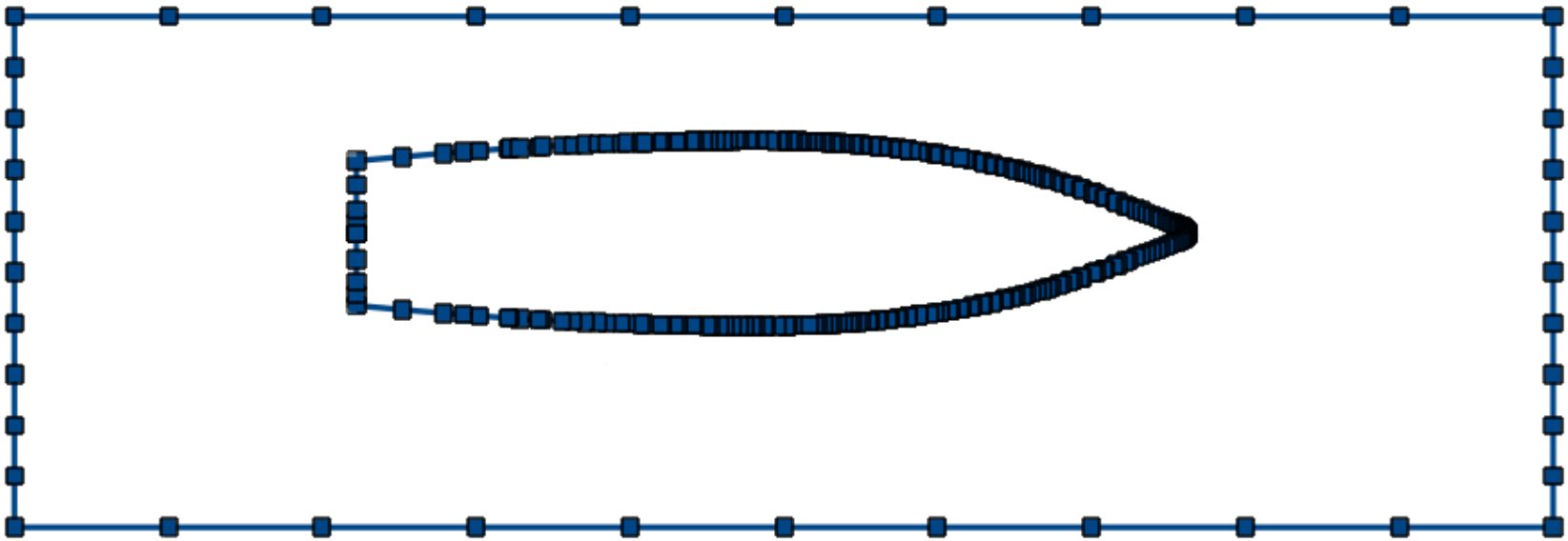
Conclusions

- The application works with wide range of ships and scenarios
- Created boundary domain is completely watertight and compatible with HEXPRESS
- The topology detection algorithm works when the ship geometry is defined accurately

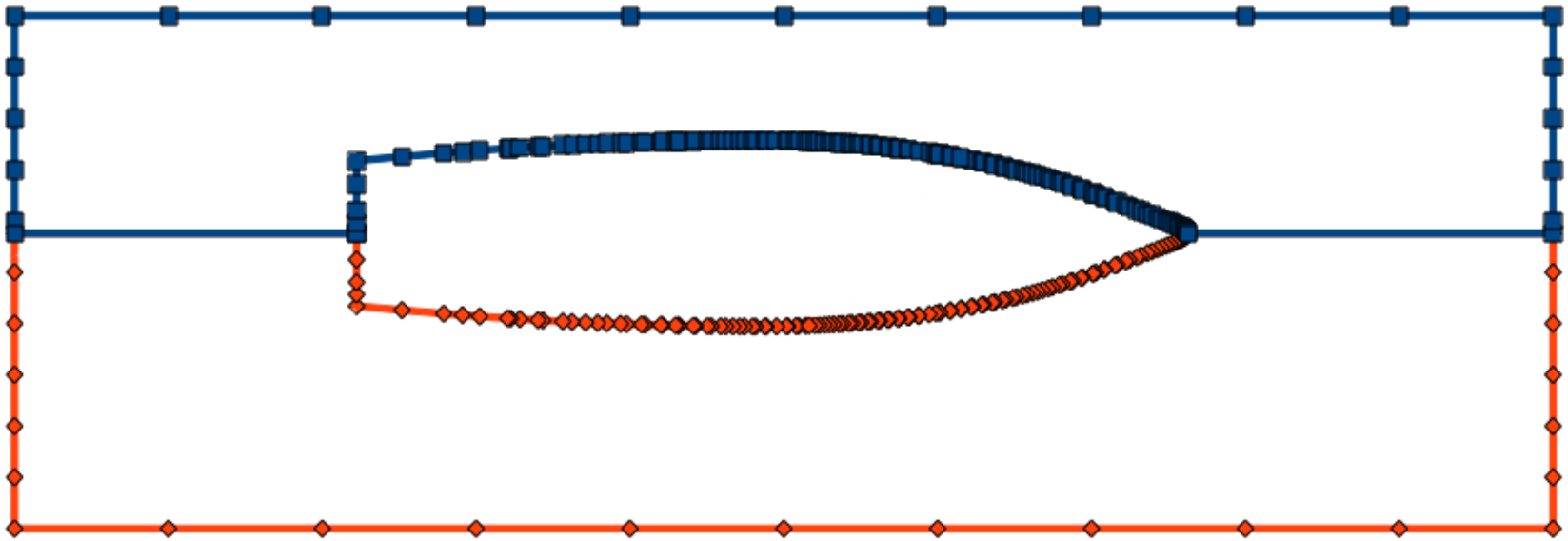
Thank you for your attention !

- Triangulation by leaf removal
- Entry Data
- Domain Simplification Submarine
- Domain Simplification Catamaran

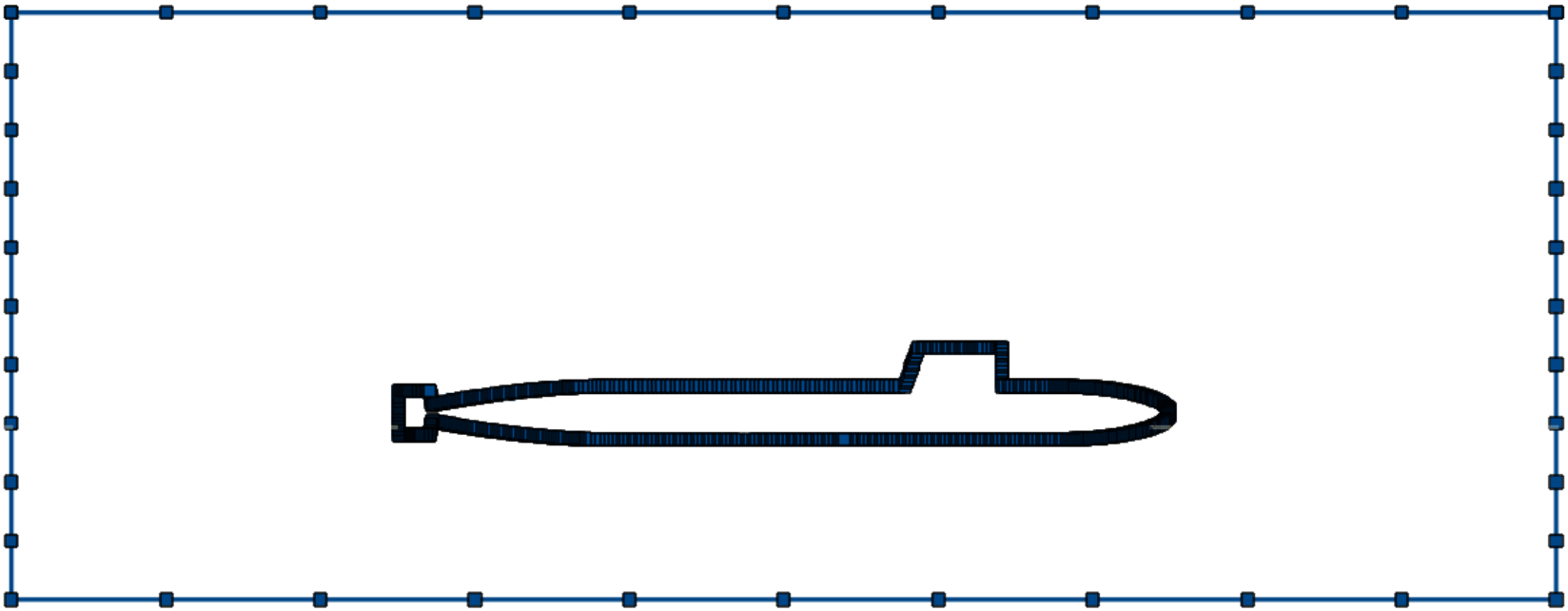
Back



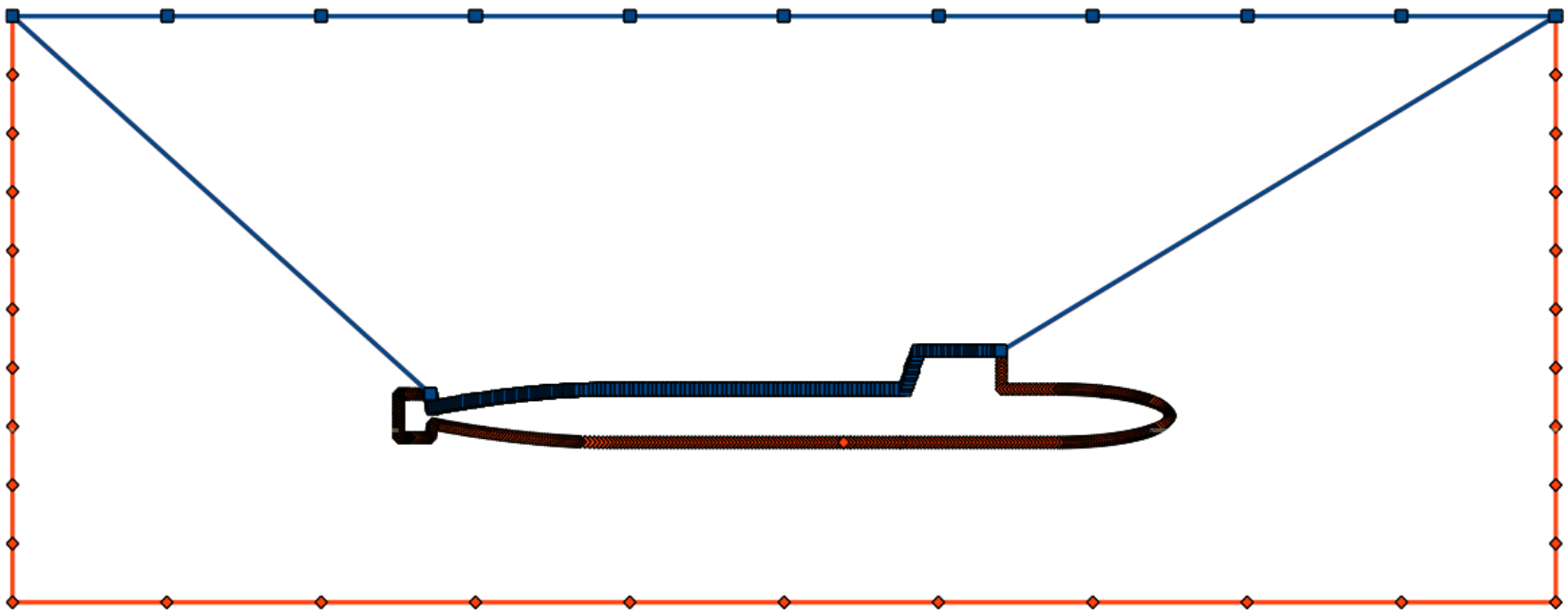
Back



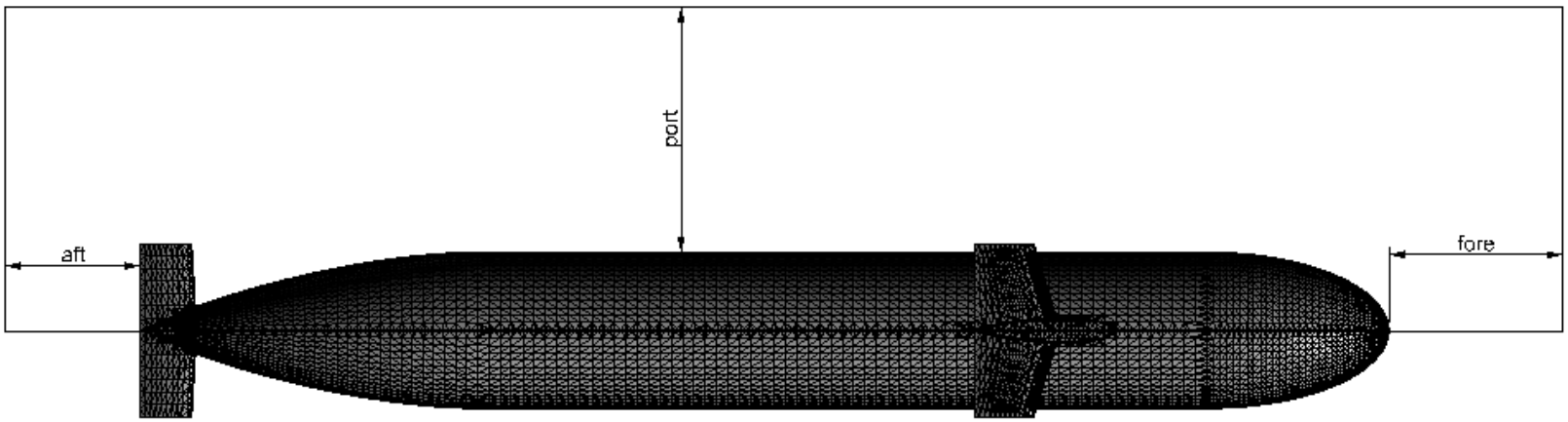
Back



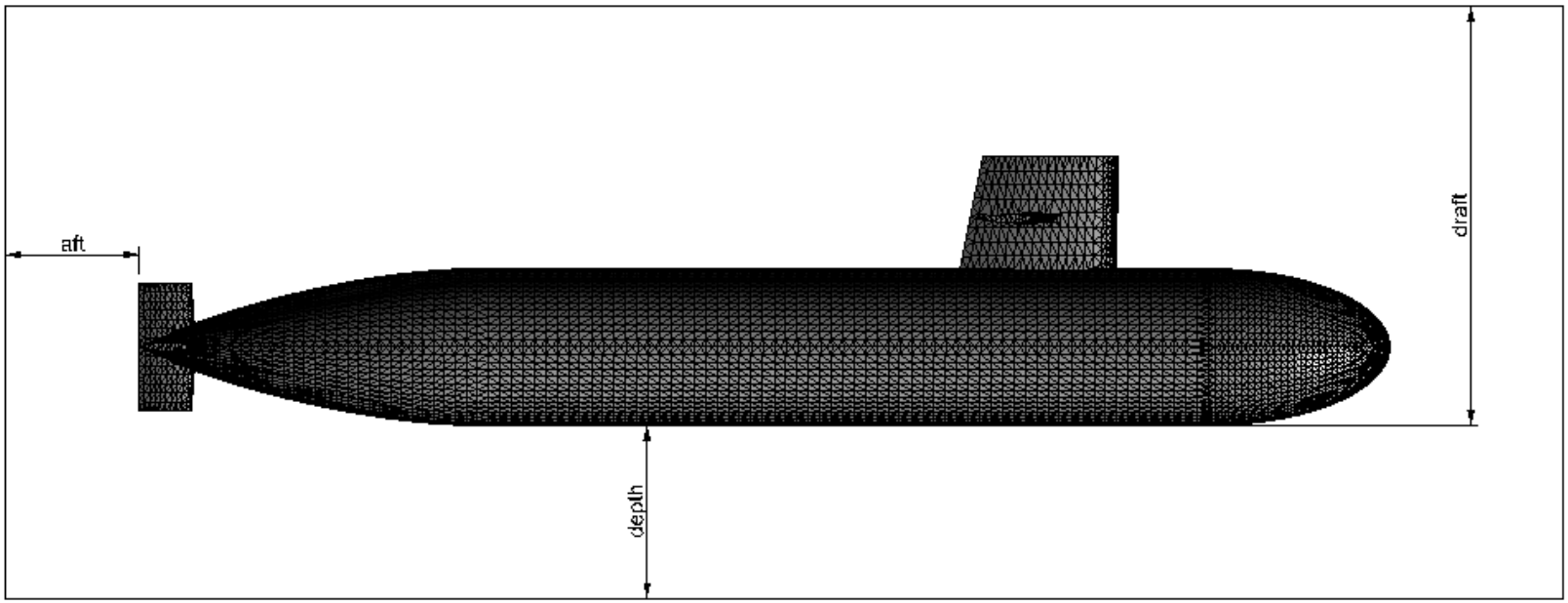
Back

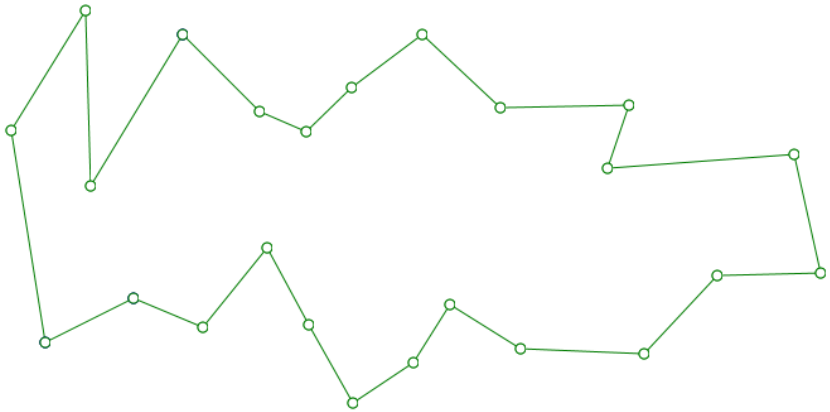


Back



Back

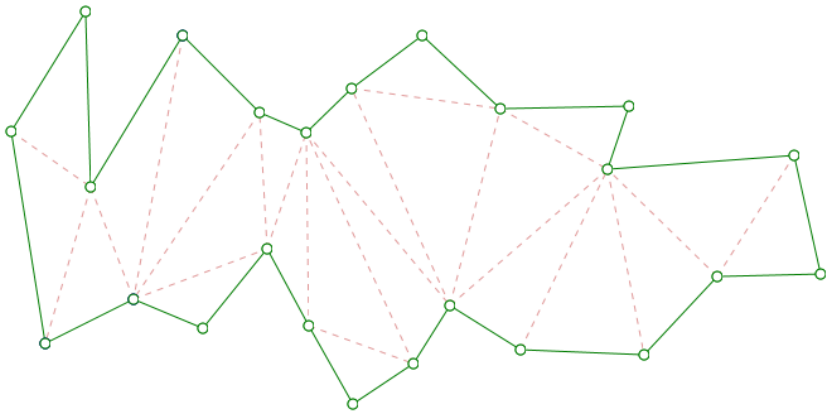




- Defining the polygon

Implementation

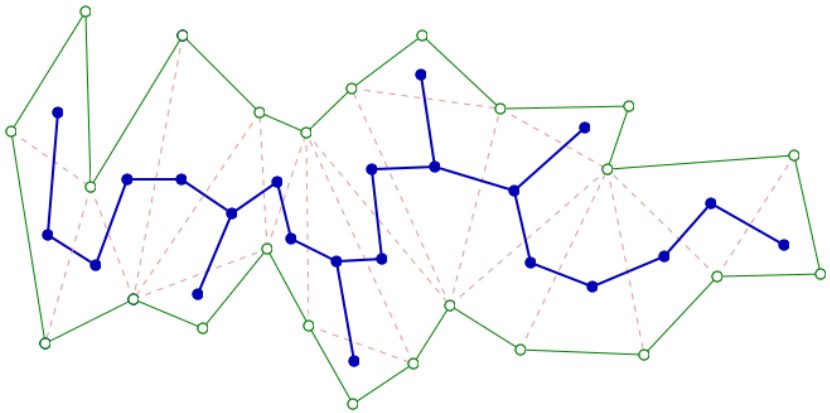
Back



- Defining the polygon
- Defining the polygon triangulation

Implementation

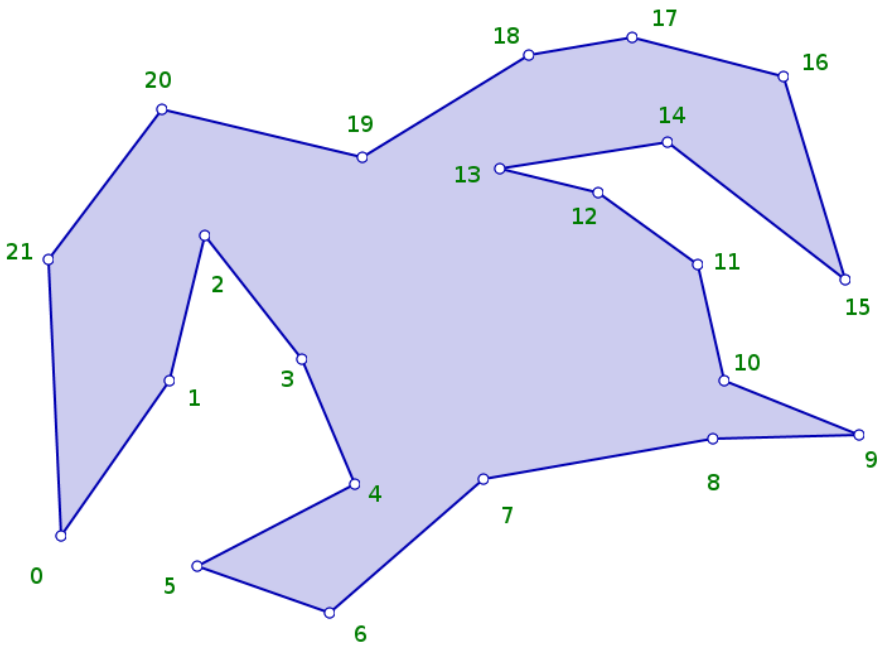
Back



- Defining the polygon
- Defining the polygon triangulation
- Presenting the triangulation as a tree / non-circular graph /

Implementation

Back

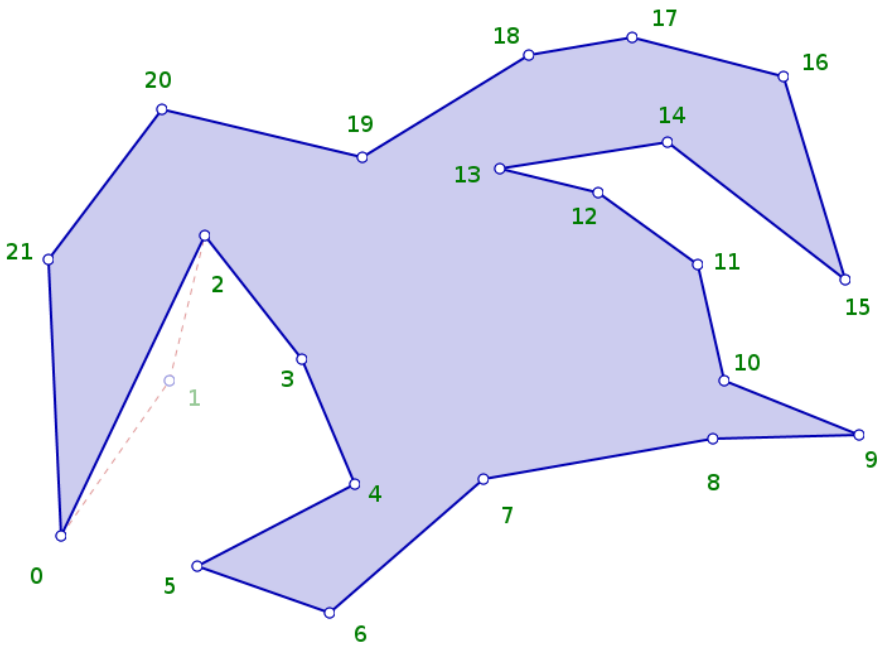


- Defining the polygon
- Defining the polygon triangulation
- Presenting the triangulation as a tree / non-circular graph /

Implementation

1. We have to enumerate all the vertices of this polygon

Back

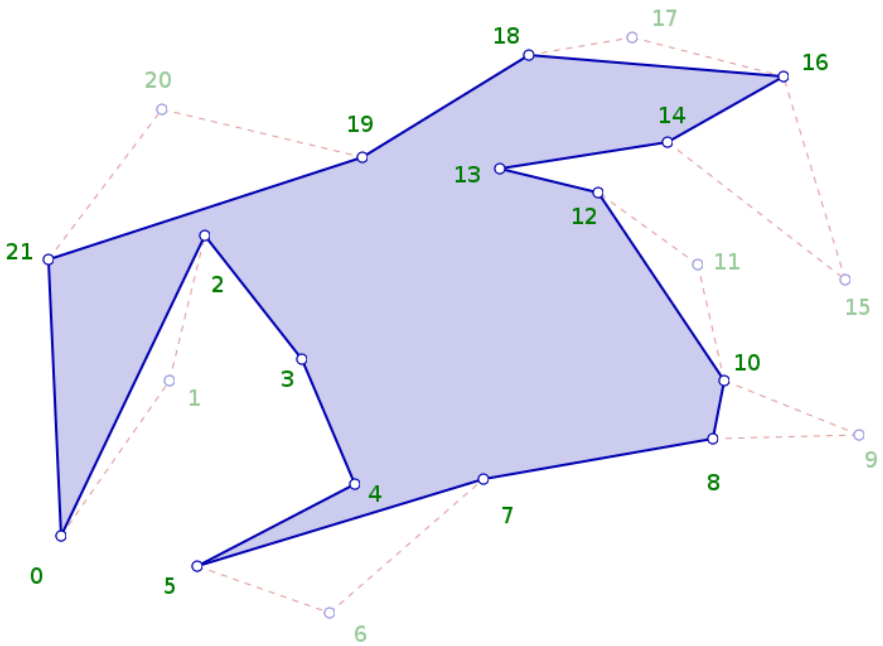


Back

- Defining the polygon
- Defining the polygon triangulation
- Presenting the triangulation as a tree / non-circular graph /

Implementation

1. We have to enumerate all the vertices of this polygon
2. Testing for every three ordered points if they define a diagonal and if so then adding the triangle to STL file
3. Looping through remaining points until only 3 remain



Back

- Defining the polygon
- Defining the polygon triangulation
- Presenting the triangulation as a tree / non-circular graph /

Implementation

1. We have to enumerate all the vertices of this polygon
2. Testing for every three ordered points if they define a diagonal and if so then adding the triangle to STL file
3. Looping through remaining points until only 3 remain