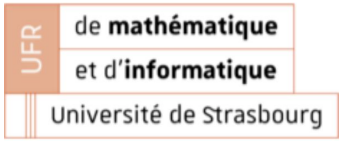
KINETIC AND BUILDING LOD2



Intern: Demuth Axel
Supervisor: Vincent Chabannes, Pierre Alliez, Florent Lafarge

Table of Contents

Introduction

Context

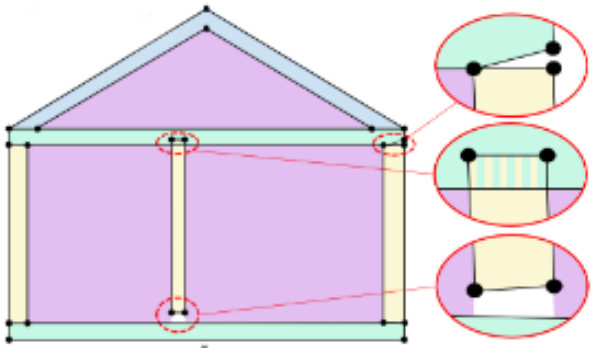


Figure: Mesh with issues

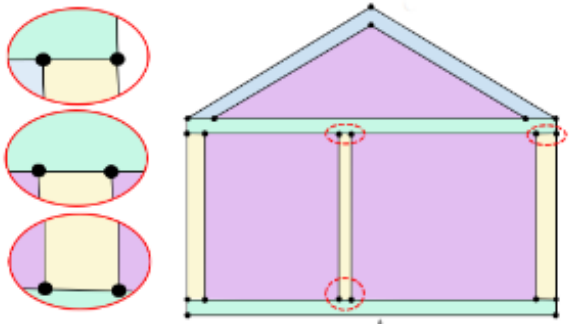


Figure: Mesh without issues

Issue with orientation

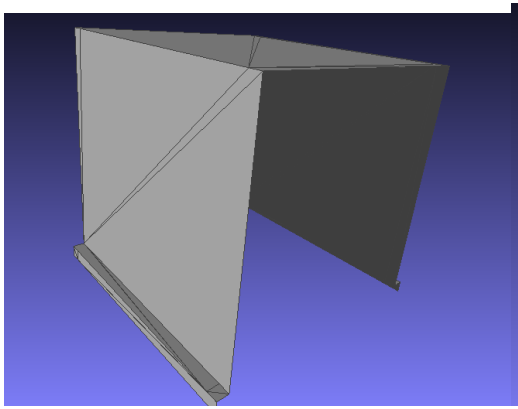


Figure: Cube not oriented

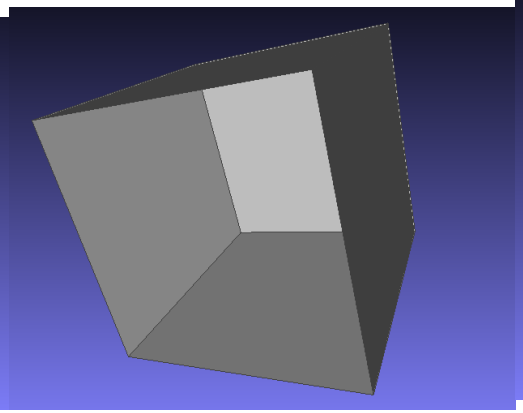


Figure: Cube badly oriented

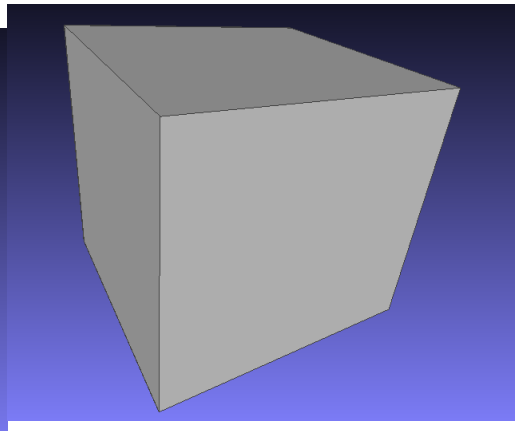


Figure: Cube oriented by CGAL

Self Intersection issue

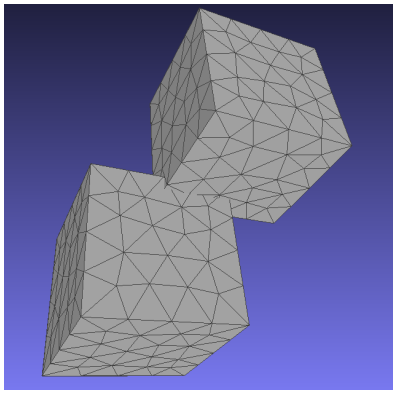


Figure: Two cube self intercting

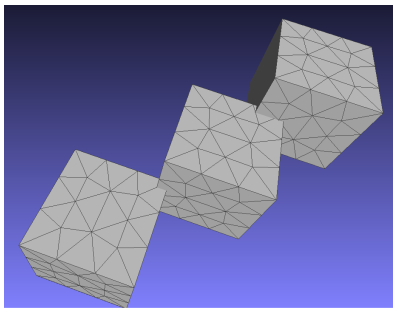


Figure: Two cubes intersecting a third one

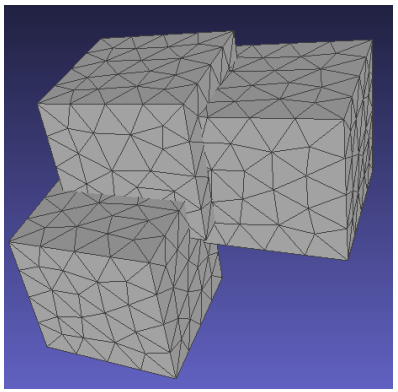


Figure: Three cubes self intersecting

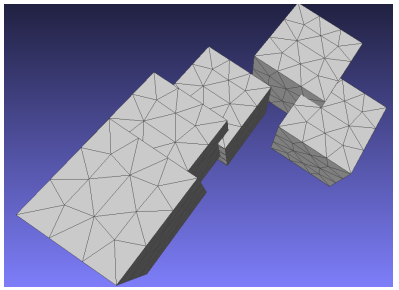


Figure: Five cubes intercting randomly

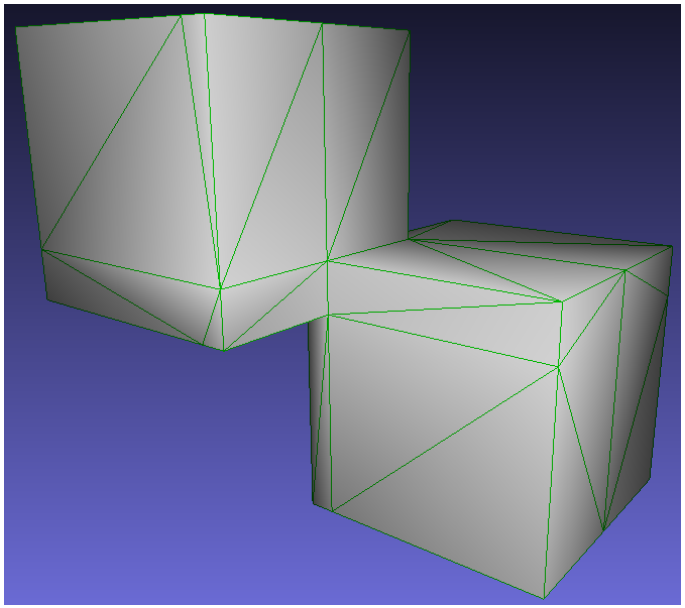


Figure: Two Cubes fixed

All other result in a execution error

Objectives

- Check the validity of the Mesh
- Create a workflow for automatic generation using KSR Algorithm
- Keep the correspondence of surfaces between both meshes
- Run some simulations using the Feel++ library

CGAL

- C++ library for geometric calculations, providing data structures for mesh generation and manipulation.

The main packages utilized are:

- `CGAL::Polygon_mesh_processing`
- `CGAL::Surface_mesh`
- `CGAL::Point_set_processing`
- `CGAL::IO_streams`
- `CGAL::AABB_tree`

File Format

- IFC : Standart for buillding data modeling,similar to class oriented code
- CityGML : 3D format for city modeling with representation of geographic details
- STL : 3D Modeling format
- OBJ :A standard file format for 3D models
- OFF : A file format for 3D mesh data
- PLY : A file format for 3D mesh data,stocking the cloud point of the mesh
- MSH : A file format for mesh data use by GMSH software

Software

- Github : Platforme for collaborating work on a project
- Visual Studio Code : Versatil tools for coding with various extensions
- Paraview : Open-source data analysis and visualisation
- Meshlab : A tool for processing,editing,visualisation of 3D mesh
- GMSH : a 3D finite element mesh generator

Data

The following Data were given by Vincent Chabannes

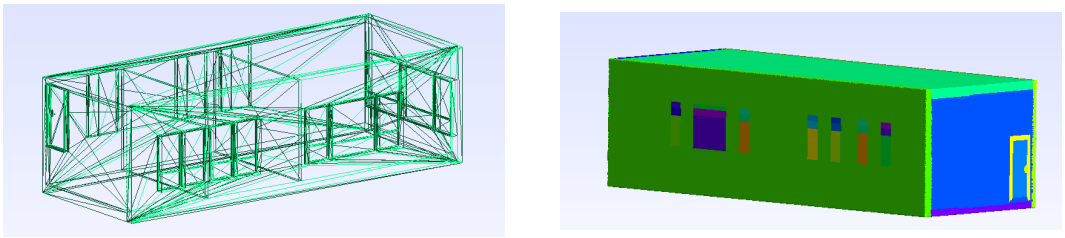


Figure: Three zones mesh

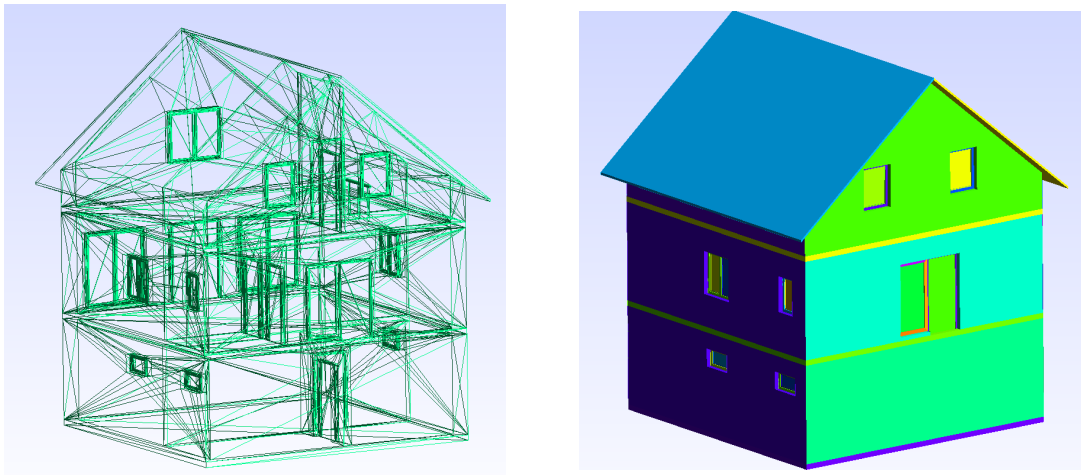


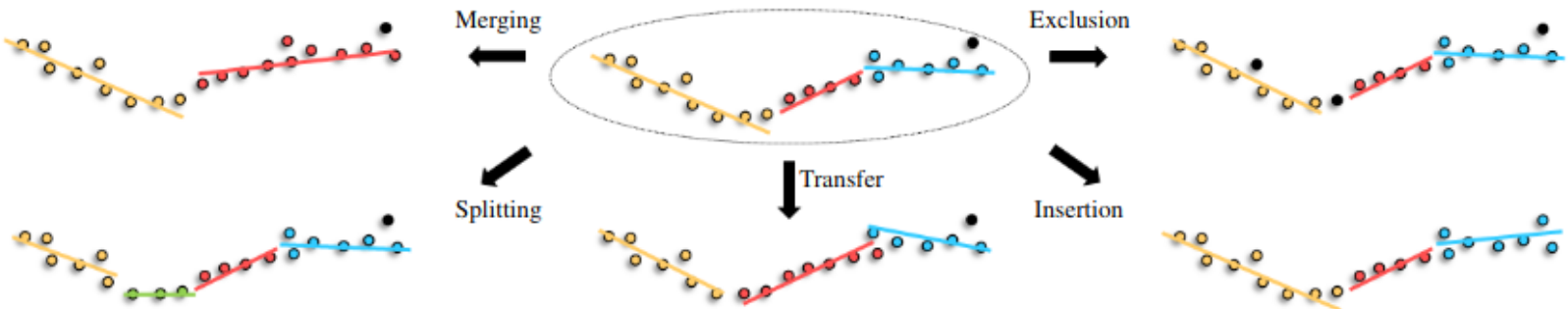
Figure: ACJasmin mesh

Kinetic

We get information from a INRIA report (citer le rapport) Kinetic algorithm is an geometric algorithm generate 3D mesh from a point clouds,it uses geometric primitive with an energy based model to fit the primitives to the model.
Energy formule:

$$U(x) = w_f U_f(x) + w_s U_s(x) + w_c U_c(x)$$

to calculate the best primitive
to fit the mesh. then we have a list of geometric operation on each primitive



preprocessing

To improve Kinetic outcome we pre-process the mesh :

- Isotropic remeshing of the mesh
- Unified and regularize the mesh with grid simplify
- Fix self Intersection
- Calcul normals

Labelling

Issue: Inria developed a method to preserve the semantic information of IFC elements, but it has not yet been implemented in CGAL.
Two potential solutions:

- Modify the Kinetic Solver to recognize and utilize markers on each point used to form a shape.
- Compare the input and output meshes to apply the same markers to the closest faces.

Labelling

Exemple of result of second solutions:

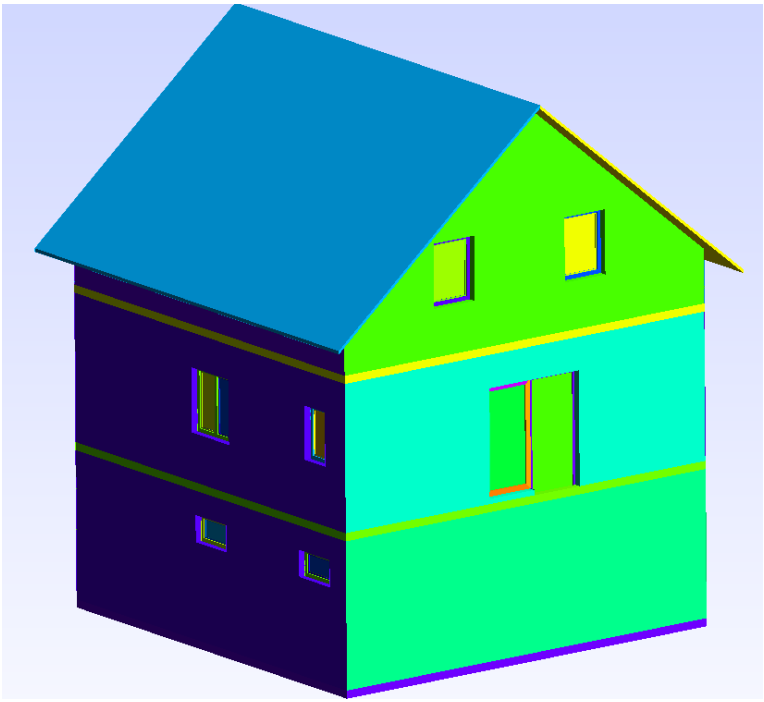


Figure: Input Mesh

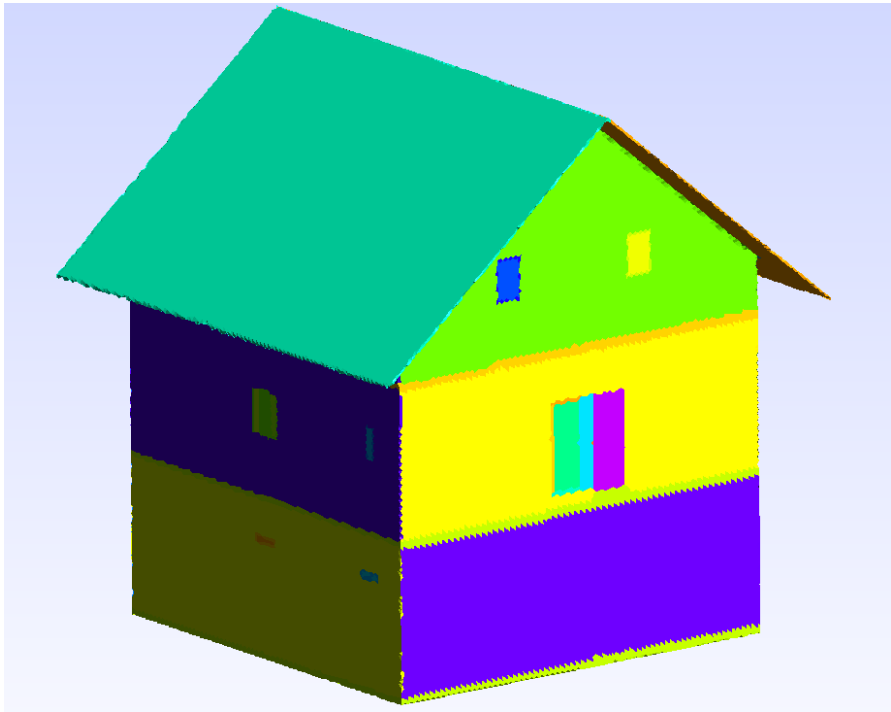


Figure: Output Mesh

Metric

We also want to add method to check the quality off the output mesh

- Properties Check (closed,connected,triangulated...)
- Correspondance between input and output

To check the Correspondance between mesh, we can compare bounding box of each labelled elements.

Table: Bounding Box value

% of marker correct	Three Zones	ACJasmin
<5%	22/57	3/82
between 5 and 10 %	11/57	7/82
between 10 and 20 %	13/57	9/82

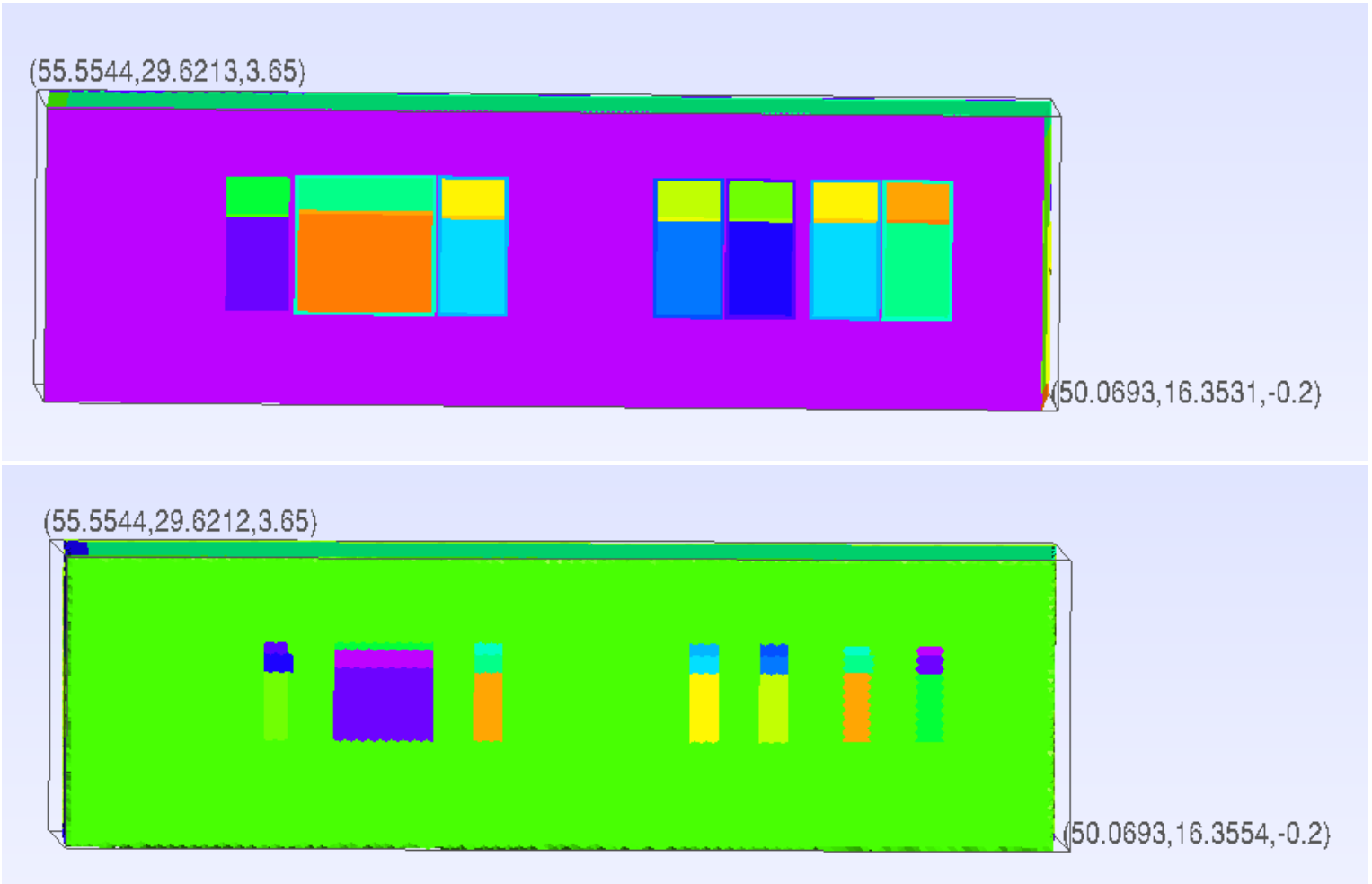


Figure: Three zones Bounding Boxe comparaison

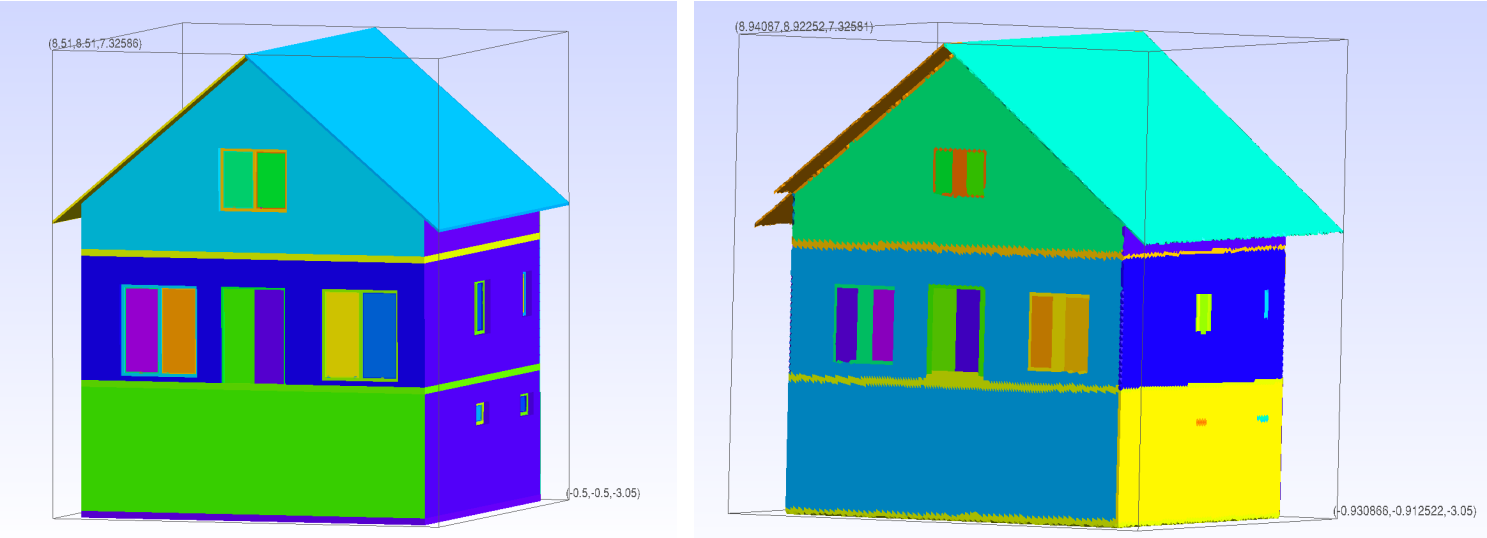


Figure: ACJasmin Bounding Boxe comparaison

Function implemented

- checkProperties
- gridSimplify
- remesh
- KSR

test

- test on Surface Mesh Check
- test on Kinetic algorithm
- test on Point set class and manipulation function

Point cloud

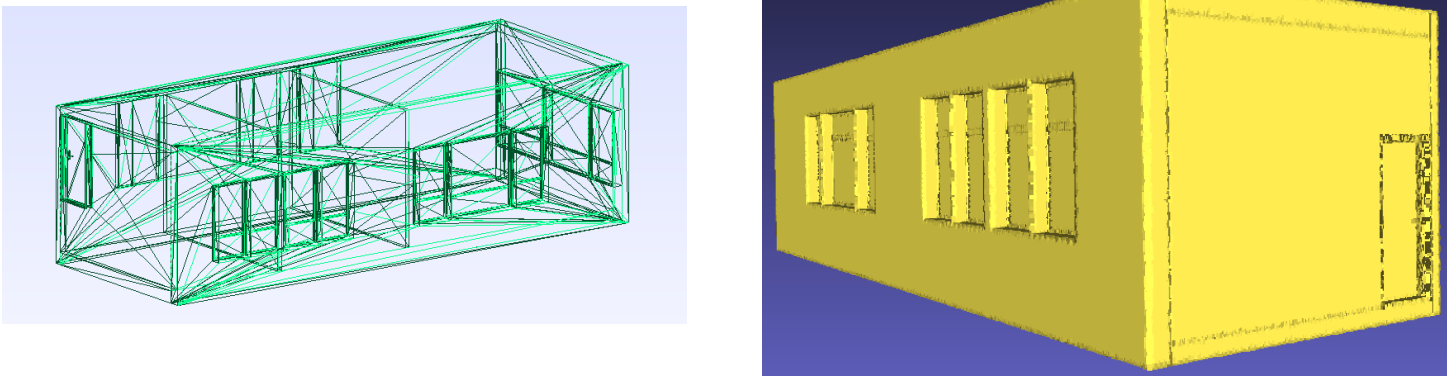
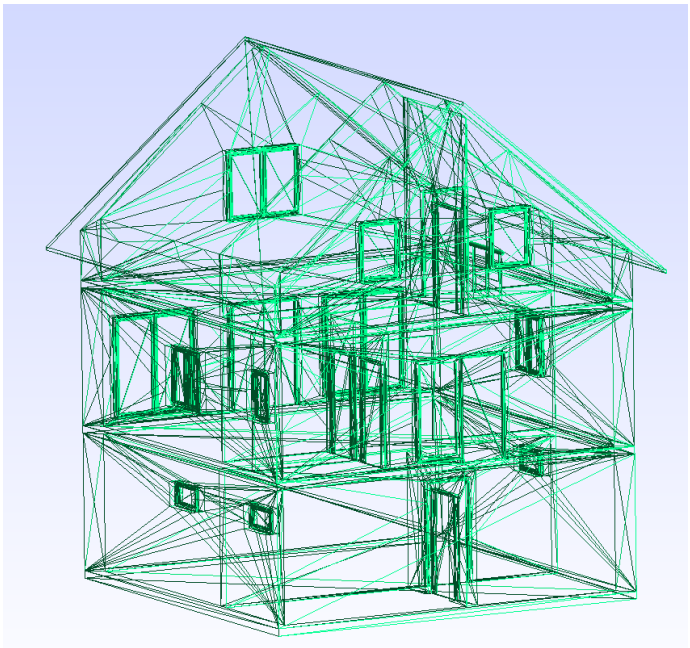


Figure: Three zones mesh point cloud



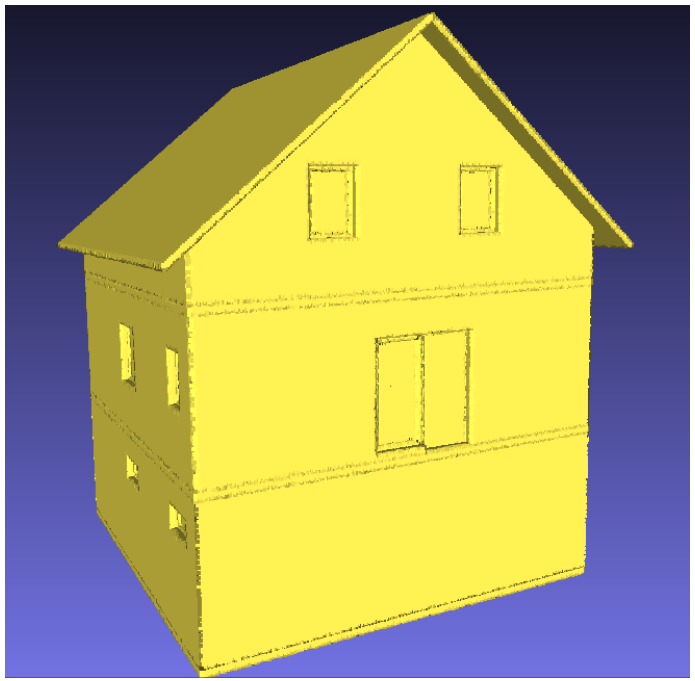


Figure: ACJasmin mesh point cloud

Self Intersection fixing

Performance

Conclusion

bib