Finite Element Modeling of Vibrating Elastic Structures – Athens March 2023

FEM Analysis of Bus Door Using Julia

Candidates:

Claudia **Crini**

Lorenzo **Lucarelli**

Mirko **Tresoldi**

Alberto **Castro**

José **Oura**

Our goal was to realize a finite element analysis of a bus plug-sliding door subjected to a static load, to find maximal displacements and stresses.

We based our work on the resources found on the GitHub page (https://github.com/ziolai/ventura-modeling), coding suggestions from other students who faced a similar project (https://github.com/natan-ber/FEM-bus-door/) and articles like "A Study on Finite Element Analysis of Electric Bus Frame for Lightweight Design" (by Tan Wei et al., 2012) and "Design of The Vehicle Door Structure Based on Finite Element Method" (by Zhuo Yang et al., 2018).

The problem we faced was simplified to a 2D problem, described by a Newton type of equation, which is also a partial differential equation, of the form:

$$\begin{cases} u_{tt} + \alpha u_t - \Delta u = f(x,y,t) \\ +InitialConditions + BoundaryConditions \end{cases}$$

where αu_t is a damping factor and $f(x,y,t)$ is a forcing component which represents and external impact or vibration source.

First, we tried to solve a static version of the problem, in particular a case in which a static load was applied: we analysed two cases, a load on the top beam of the open door and a load falling on the door while the bus is in standstill. These cases allow to remove the time dependency from the problem:

$$\begin{cases} -\Delta u = f(x,y) \\ +BC \end{cases} \quad \text{with } (x,y) \in \Omega.$$

2D

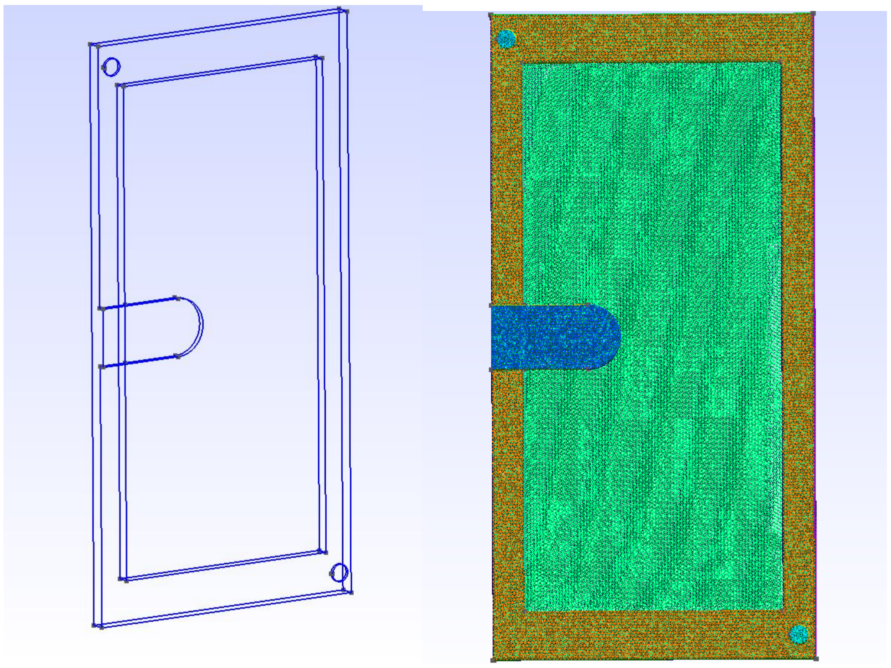
In order to simplify the process of analysing the plug-sliding door, the group created a 2D model representing it. That model was constructed using two rectangles, each with a specific material associated to it. In this representation of the door, the group was able to apply constrains to the whole extent of its top part (the top line) and to one of its inferior vertices (the one on the right), simulating the constrains applied in a real situation. By developing different codes in the Julia software, we were able to:

- Initially, do a mesh of this 2D model, that could be visualized on the Gmsh software (thus, creating a connection between Julia and Gmsh);
- Secondly, using the previous mesh, simulate the load of a “x” and “y” force, calculating therefore the value of “u”, having in account that, in this case, the is no distinction between the properties of the two different materials (the boundary conditions in the interacting surfaces were defined, but the analysis program didn’t recognise the different properties of the materials);
- Finally, simulate the same load as previously, having the interaction between the two domains (aluminium and glass) in account. In this case, the physical distinction between the materials was defined and appeared on the output of the program, having each of them its own properties and, consequently, different reactions to the loads.

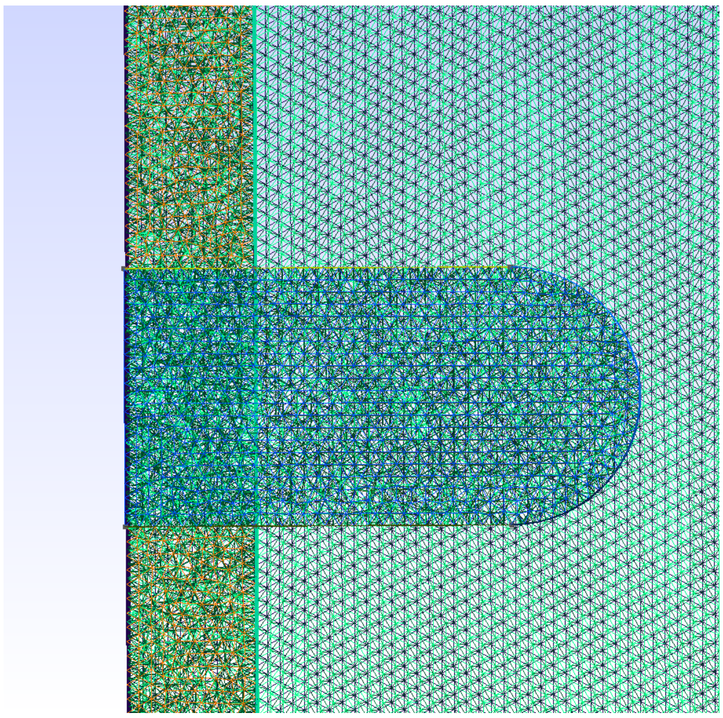
The code was then rewritten in the Jupyter Notebook software, which provides a faster execution and better visualisation of its output. After running the code, the user would be able to visualize a representation of displacement “u” on the mesh, through a 3D graph (where the x and y represent the coordinates of the 2D model and the coordinate z represents the value of “u” in each point).

3D:

We tried to solve the problem in the 3D domain as well, first by creating a 3D model of the door, with varying thicknesses in the frame+door handle part and in the glass part, on a CAD software (Autodesk Inventor):



We defined different volumes with the Gmsh function, so that we could distinguish different materials, and generated a mesh, setting 5 smoothing steps and an element size factor of 0.05.



Then we used the following code in Julia, with packages necessary to allow interface between Gmsh and Julia, to define the material properties (E, ν, ρ, then converted to Lamè constants μ=G and λ), and the boundary conditions:

```
using Gridap

using GridapGmsh

model = GmshDiscreteModel("C:/Users/…./door_new.msh")

writevtk(model,"model")


using Gridap.Geometry

labels = get_face_labeling(model)

dimension = 3

tags = get_face_tag(labels,dimension)


const glass_tag = get_tag_from_name(labels,"glass")

order = 1

reffe = ReferenceFE(lagrangian,VectorValue{3,Float64},order)

V0 = TestFESpace(model,reffe;

  conformity=:H1,

  dirichlet_tags=["boundary1","handle","boundary2"],

  dirichlet_masks=[(true,true,true), (true,false,false), (true,true,true)])


g1(x) = VectorValue(0.0,0.0,0.0)

g2(x) = VectorValue(0.0,0.0,0.0)

g3(x) = VectorValue(0.0,0.0,-0.005)


U = TrialFESpace(V0,[g1,g2,g3])


function lame_parameters(E,ν)

  λ = (E*ν)/((1+ν)*(1-2*ν))

  μ = E/(2*(1+ν))

  (λ, μ)

end
```

```
const E_glass = 100e9
const ν_glass = 0.37
const (λ_glass,μ_glass) = lame_parameters(E_glass,ν_glass)
const E_alu = 69.0e9
const ν_alu = 0.33
const (λ_alu,μ_alu) = lame_parameters(E_alu,ν_alu)

function σ_bimat(ε,tag)
  if tag == glass_tag
    return λ_glass*tr(ε)*one(ε) + 2*μ_glass*ε
  else
    return λ_alu*tr(ε)*one(ε) + 2*μ_alu*ε
  end
end
```

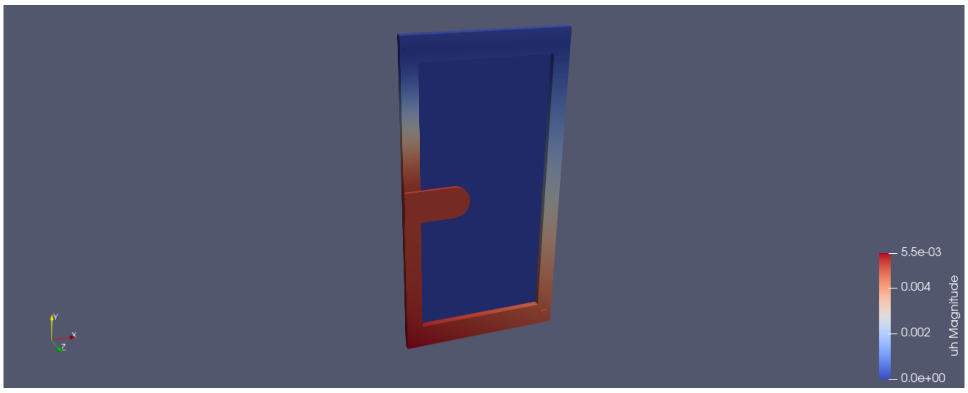
We applied the weak form of the problem as:

```
degree = 2*order
Ω = Triangulation(model)
dΩ = Measure(Ω,degree)
a(u,v) = ∫( ε(v) ⊙ (σ_bimat∘(ε(u),tags)) )*dΩ
l(v) = 0
```

Now we could finally build the FE simulation where the load was applied to the door handle, by imposing a 5mm displacement of the handle which could simulate for example a passenger pushing on it, and run it using the commands:

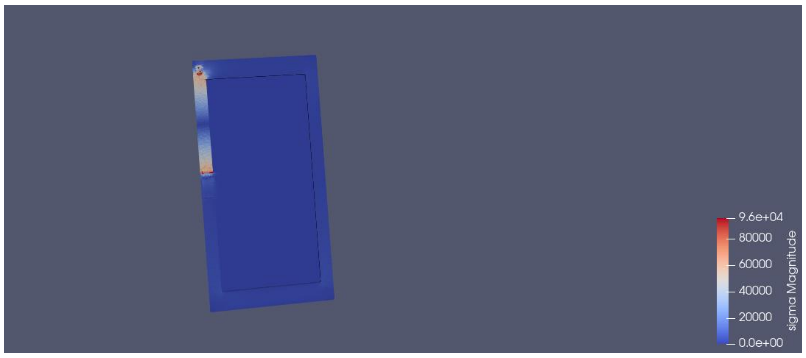
```
op = AffineFEOperator(a,l,U,V0)
uh = solve(op)
writevtk(Ω,"results",cellfields=
 ["uh"=>uh,"epsi"=>ε(uh),"sigma"=>σ_bimat∘(ε(uh),tags)]
```

To see the results, we used the open-source post-processing app called Paraview:

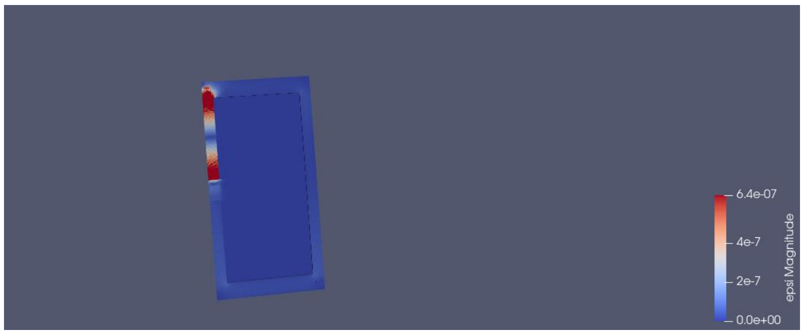


Now we can go through the results in 'Coloring' tab. Selection of 'uh' show us total deformation, which is 5 mm in the area of the handle as expected.

We can inspect the deformations in directions of x, y and z axis in the tab next to it. By going into 'sigma' and 'epsi' tabs we can review the stress and strain, which were: 96kPa for the highest stress, which is significantly less than the tensile yield strength of aluminium (276 MPa), and 6.4*10^-7 for the highest strain.



Stresses



Deformation

In conclusion, we could run the 3D simulation of a bus door using various open-source software. However, the transition between the two materials should be improved, as we couldn't find a way to define their different properties while making sure that the volumes were connected, as is evident when observing the results in Paraview, where stresses and deformations are only visible on the aluminium frame.