G-RIPS Sendai 2026

Daikin Group

Final Report

Authors:
Anastasiia Rudenko[1]
Davis Funk[2]
Itsuki Imai[3]
Shota Nakagawa[3]

Mentors:
Yuji Ando[+]
Satoru Takanezawa[*]
Keiji Tomioka[*]
Yuki Eda[*]
Ryota Shiba[*]

[1] University of Pittsburgh,
[2] University of California Los Angeles,
[3] Keio University,
[+] Tohoku University,
[*] Industrial Mentor

August 6, 2026

Contents

1 Introduction **3**
- 1.1 Background . . . 3
- 1.2 Research Significance . . . 3

2 Problem Statement **3**
- 2.1 Pre-thesis . . . 3
- 2.2 Research Direction and Open Issues . . . 5

3 Mathematical Formulation **5**
- 3.1 Weak Form of Daikin's Equation Set . . . 5
- 3.2 Euler-Bernoulli Beam Theory . . . 6
- 3.3 Timoshenko Beam Theory . . . 8
- 3.4 Optimization . . . 10
- 3.5 Additional Mathematics . . . 12
  - 3.5.1 Damped Timoshenko . . . 12
  - 3.5.2 Expanding the General Theory . . . 12

4 Methodology **14**
- 4.1 Explicit Embedding Approach . . . 14
- 4.2 Spline Polynomial Approach . . . 15
- 4.3 Numerical Simulation . . . 17
  - 4.3.1 Forward Solver . . . 17
  - 4.3.2 Temporal Discretization . . . 17
  - 4.3.3 Rayleigh Damping . . . 20
  - 4.3.4 Optimization Iteration . . . 22

5 Route Refinement Algorithm **23**
- 5.1 Purpose of Our Algorithm . . . 23
  - 5.1.1 Diversity of Routes . . . 23
  - 5.1.2 Efficiency . . . 23
  - 5.1.3 For Practical Use . . . 23
- 5.2 Overall Flow . . . 23
- 5.3 Detailed Description of Each Step . . . 24

6 Results **28**
- 6.1 Experiment Description . . . 28
- 6.2 Analysis of Results . . . 30
  - 6.2.1 Forward Solver . . . 30
  - 6.2.2 Optimization . . . 32
  - 6.2.3 Caveats . . . 34

7 Conclusion **36**
- 7.1 Status of Deliverables and Limitations . . . 36
- 7.2 Summary of Contributions . . . 36

8 Future Work **37**
- 8.1 Short-term Tasks / Remaining Implementations . . . 37

8.2 Long-term Research Directions 37

A Pseudo-code of the route refinement algorithm **38**

1 Introduction

1.1 Background

DAIKIN Industries, Ltd. (further addressed as Daikin) is a globally leading AC supplier. It has business development in more than 170 countries, with 83% of its sales being in regions outside of Japan. The company aims to solve the heating and cooling problems of countries and regions across the world, as well as to grow its business. Currently, the company is focusing on developing a novel method to aid in the design of AC units, and assist designers when selecting a pipe routing to reduce the stress and noise a unit experiences. With that goal in mind, Daikin conducted research regarding mathematically modeling the pipe system and how to mathematically optimize the shape of a pipe to account for stress. The purpose of this project is to build on the previous research done by Daikin and to improve and mathematically prove the well posedness of the model, to simulate it, and then possibly expand the model into one that has less limitations.

1.2 Research Significance

The current manufacturing process has room for improvement. Designers currently use CAE software to model pipes and then perform vibration simulations to determine whether the designed pipe can achieve desirable stresses. If desired performance metrics are not met, then designers must manually modify the pipe's shape. In this process, decisions regarding pipe shape are wholly reliant on the designer's knowledge and experience; however, since the optimal pipe shape remains a "black box," the process of redesigning lacks efficiency. This process can take an indeterminate amount of iterations as a result, and when combined with long simulation times, can greatly lengthen the product design process. By mathematically optimizing the pipe shape, it becomes possible to automate the determination of pipe shapes with a simulation software — which today depends entirely on the designer — with high precision. Since mathematical modeling cannot strictly account for all physical constraints, combining it with the designer's practical expertise and experience is expected to efficiently identify more desirable pipe shapes. A successful mathematical modeling software, combined with the expertise of an experienced designer, can both shorten the time it takes to design new products, and free up designers to use their time more productively.

2 Problem Statement

In this section, the problem is specified, and the previous work of Daikin (aka the pre-thesis) is introduced.

2.1 Pre-thesis

Discovering the optimal shape of an elastic pipe is a critical challenge. To solve a problem where the physical shape of an object is constantly changing — in our case, this is due to the pipes experiencing vibration due to the motion of the compressor — we cannot effectively compute our physics on a moving grid. Instead, we decompose the problem by introducing a fixed, static domain called the "reference space", which we denote mathematically by M. We can think of a reference space as the perfectly straight pipe acting as a blueprint of our actual pipe. Meanwhile, our actual physical pipe — that ultimately will vibrate and experience stress — exists in the "physical space", that we denote as Ω. To connect the reference space with the reality, we introduce a continuous mapping function $\iota(\cdot)$. This mapping takes a point in the reference image of our pipe, $y \in M$, and moves it to a new

location, $x \in \Omega$, to create our desired actual pipe shape. The figure below showcases this scenario Takiwaki [unpublished].

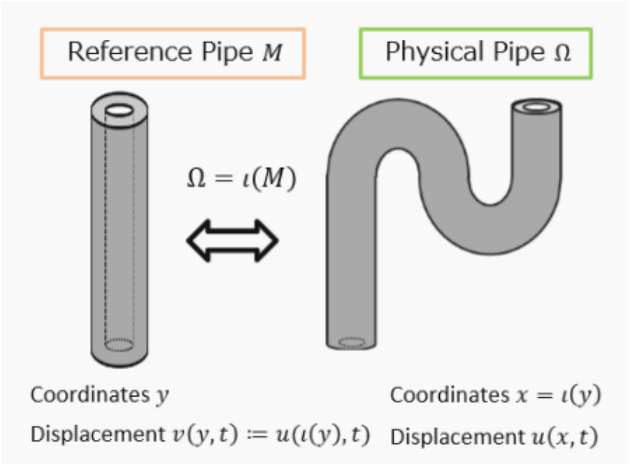


Figure 1: Representation of reference and physical space.

In Daikin's previous work, they applied the infinitesimal stress theory to this problem, to develop a set of seven equations that would need to be solved. First, they calculated the Lagrangian of the physical system, and computed its first variation with respect to the reference pipe's displacement, v. Through the appropriate application of tensor/vector calculus identities, and setting the resulting equation equal to 0, the equation of motion (1) for the system was found. The boundary condition (3) was also found from this formula. Equations (2) and (4) are prescribed boundary and initial conditions respectively. (2) defines the application of the vibration, $f(y,t)$, and (4) defines the initial state of the pipe. Then, the maximum squared von-Mises stress, approximated by the KS function (to be discussed in section 3), was minimized by taking the first variation with respect to the embedding map, ι. This procedure resulted in (5), the shape optimization condition, as well as the boundary condition (6). Equation (7) is a prescribed boundary condition, fixing the embedding map at some interface/boundary with the compressor, defined by a shape $g(y)$. This full set of equations can be seen below, with the appropriate tensors, $[V]$ and $[G]$. Below, v is the displacement of the reference pipe, μ and λ are the Lame parameters, F is the shape gradient $\frac{\partial \iota}{\partial y}$, and A is the gradient defined in the last equation below. In all of these calculations, a critical assumption that $\det F = 1$ is made. This assumption is nonphysical, as it states that the embedding map ι must preserve volume, greatly limiting the possible choices of ι.

$$\rho \frac{\partial^2 v}{\partial t^2} = \nabla \cdot [V]^T \text{ on } M, \tag{1}$$

$$v(y,t) = f(y,t) \text{ on } D_1, \tag{2}$$

$$[V]\hat{n} = 0 \text{ on } D_2, \tag{3}$$

$$v(y,0) = v_0(y) \text{ on } M, \tag{4}$$

$$a[G]^T \nabla (M[\iota, v]^2) + \nabla \cdot [G]^T = 0 \text{ on } M, \tag{5}$$

$$[G]\hat{n} = 0 \text{ on } D_2, \tag{6}$$

$$\iota(y) = g(y) \text{ on } D_1, \tag{7}$$

$$[V] = \mu \frac{\partial v}{\partial y} + \mu F \left(\frac{\partial v}{\partial y} \right)^T F + \lambda \text{tr} \left(\frac{\partial v}{\partial y} F^T \right) F,$$

$$[G] = 6\mu^2 A^T \frac{\partial v}{\partial y} + 6\mu^2 A \frac{\partial v}{\partial y} - 4\mu^2 \mathrm{tr}(A) \frac{\partial v}{\partial y}, \quad A = \frac{\partial v}{\partial y} F^T.$$

2.2 Research Direction and Open Issues

Building on this prior progress, the group will first conduct a thorough mathematical analysis of this system of equations. The weak formulation of these equations will be derived first. This weak formulation will then be used as the starting point for developing numerical simulations. A primary goal for this project is to provide concrete numerical simulations that prove the shape optimization method outlined in the previous research is effective. In order to develop these simulations, the group will need to devise an appropriate method to discretize the set equations, determine a viable functional form of ι, and determine a method to numerically enforce real world constraints on the pipe layout. As simulations are being prepared, simultaneous work will be done to analyze the $\det F > 0$ case, and build a theoretical basis for that problem.

3 Mathematical Formulation

This section summarizes the mathematical work that was completed during this project, in order to support the development of the desired Minimum Viable Product (MVP). The first mathematical task that was undertaken by the team was to review and verify the previous work done by Daikin Takiwaki [unpublished]. Minor typos were corrected during this process, and once verification was finished, the mathematical background could be constructed. First, the weak formulation of the equations Daikin provided to the group will be discussed. Due to the complexity of the resulting weak formulations, and the time limit of the G-RIPS program, the group decided to use a simplified model of the physics of the problem. Multiple theories will be discussed, along with the relevant mathematical work, and the reasoning behind the group's decision to use the Timoshenko theory. The final weak formulation of Timoshenko will be derived and explained. The mathematics of the optimization procedure, including forming the adjoint equations, as well as calculating the gradient update will be discussed. Finally, additionally developed theory, unrelated to the MVP, will be explained.

3.1 Weak Form of Daikin's Equation Set

The team's original first objective in this project was to derive the weak formulation of the set of PDE's that Daikin had provided us with. First, the equation of motion (EOM) was studied, in order to develop the underlying mathematics that would be used in the FEM simulation. To begin, the EOM was multiplied by an appropriate test function, w. The resulting equation was then integrated over the problem's spatio-temporal domain, leading to the following equation:

$$\int_{t_0}^{t_1} \int_M \rho \frac{\partial^2 v}{\partial t^2} \cdot w dy dt = \int_{t_0}^{t_1} \int_M (\nabla \cdot [V]^T) \cdot w dy dt.$$

By applying integration by parts, and applying the appropriate boundary conditions for this PDE, the following weak formulation was derived:

$$\int_{t_0}^{t_1} \int_M \mathrm{tr}\left([V]^T \frac{\partial w}{\partial y}\right) + \rho \frac{\partial^2 v}{\partial t^2} \cdot w dy dt = \int_{t_0}^{t_1} \int_{D_1} f(y,t) \cdot [V] \hat{n} dS dt.$$

Note that the forced vibration, $f(y,t)$ acts on the pipe system through the boundary condition, and its action on the boundary section D_1 in particular. By applying common tensor calculus identities,

we can note that the trace term in this expression is equivalent to a double dot product between the $[V]$ tensor, and the spatial derivative of the test function. $[V]$ is itself a combination of terms involving the spatial derivatives of the displacement v, and this double dot product quantifies the strain energy of the system, and can be used to derive the stiffness matrix. The other term represents kinetic energy, and can be used to derive the mass matrix. The shape optimization PDE was then multiplied by an appropriate test embedding function, κ. The system was then integrated over the reference space, resulting in the following equation:

$$\int_M a\left([G]^T\nabla(M^2[\iota,v])\right)\cdot\kappa+(\nabla\cdot[G]^T)\cdot\kappa dy=0$$

Again, after applying integration by parts, and utilizing the appropriate boundary conditions, the following PDE set was derived.

$$\int_M a\left(\nabla(M^2[\iota,v])\cdot[G]\kappa\right)-\text{tr}\left([G]^T\frac{\partial\kappa}{\partial y}\right)dy=\int_{D_1}g\cdot[G]\hat{n}dS$$

As with the weak form for the EOM, this weak formulation is immensely difficult to study. After discussing this situation with the industry mentors, the group decide to move forward with a simplified model of the physics, treating the pipe shape as being composed of 1D beam elements. This simplifies the physics of the problem, and also matches the standard practice of using simplified elements in FEM to tackle specific problems. While this is a simplification, it does not necessarily represent a downgrade in the quality of our model. Beam elements are commonly utilized in modern FEM to make efficient simulations that still capture an appropriate amount of detail in the physics. While each beam element is thought of as being 1D, 3D information is still encoded through cross sectional properties of the pipe, such as the area A and moments of inertia I. It should be noted that these properties are treated as constants mathematically. In the physical world, cross-sectional areas deform during vibration, but this effect is negligible and does not majorly affect the model's accuracy. These physical constants, which will appear in our physics equations, encode information about the 3D nature of the pipe into our beam element model. This simplification also allows us to think of our reference space as a 1D interval $[0,1]$, which is mapped into a 3D pipe curve. After constructing the above weak forms, the shape optimization PDE was modified to account for the time dependence of the matrix $[G]$ and the von-Mises stress, M. The new set of equations is:

$$\int_{t_0}^{t_1}(a[G]^T\nabla(M[\iota,v]^2)+\nabla\cdot[G]^T)dt=0\text{ on }M \tag{8}$$

$$\int_{t_0}^{t_1}w[G]\hat{n}=0\text{ on }D_2 \tag{9}$$

$$\iota(y)=g(y)\text{ on }D_1 \tag{10}$$

Due to the project's time constraint, and a lack of need for the equations, the weak formulation of this new set of shape optimization conditions has not been derived. The new time dependence of the equation further complicates the study and analysis an already difficult set of equations.

3.2 Euler-Bernoulli Beam Theory

The first beam element theory that the group would attempt to work with was the Euler-Bernoulli Beam Theory (EBT). EBT is the simplest beam theory, and is the first many engineering students encounter when beginning to study structures. This theory assumes that the cross-section of the beam remains plane (i.e. does not rotate), eliminating the possibility of shear deformation Anderson

and Nielson [2008]. However, it is key to note that we are mapping a fully 3D beam into a 1D reference pipe, meaning we are mapping full 3D physics into a 1D space. As a result, the 1D reference pipe will experience the full 3D physics. This leads to a set of four partial differential equations that fully describe the physics of the pipe. One equation describes the axial displacement of the pipe due to normal forces, two describe bending over the two possible bending planes, and one describes the torsional rotation of the pipe. Due to the assumption to neglect shear deformation, torsion will be considered as the only source of rotation. Noting that we treat force as a boundary condition, we arrive at the set of four PDE's:

$$\rho A\frac{\partial^2 u_y}{\partial t^2}+\frac{\partial^2}{\partial s^2}\left(EI_z\frac{\partial^2 u_y}{\partial s^2}\right)=0,\quad \rho A\frac{\partial^2 u_z}{\partial t^2}+\frac{\partial^2}{\partial s^2}\left(EI_y\frac{\partial^2 u_z}{\partial s^2}\right)=0,$$
$$\rho A\frac{\partial^2 u_x}{\partial t^2}-\frac{\partial}{\partial s}\left(EA\frac{\partial u_x}{\partial s}\right)=0,\quad \rho I_p\frac{\partial^2 \theta}{\partial t^2}-\frac{\partial}{\partial s}\left(GJ\frac{\partial \theta}{\partial s}\right)=0.$$

In the above equation, $u_x, u_y,$ and u_z, represent our translational displacements, and θ is the twist due to torsion. $A, \rho, E, I, G,$ and J represent the relevant physical quantities: cross-sectional area, density, elastic modulus, moments of inertia, shear modulus, and polar moment of inertia. These equations are written with respect to the local frame of reference of the pipe, parameterized by the arc length coordinate, s. As our embedding map $\iota(\xi)$, maps a 1D space into a full 3D curve, when we take its shape derivative, F, we find that the norm of this vector has the following convenient property:

$$ds=||F||d\xi\rightarrow\frac{\partial}{\partial s}=\frac{1}{||F||}\frac{\partial}{\partial \xi}.$$

This gives us a relatively simple coordinate transform between the arc length coordinate and the reference coordinate, ξ, that will be applied to the above set of strong equations while transforming to the weak formulation. The detail of this is omitted as it is a straightforward, but tedious calculation. After converting each individual equation to its weak form, and applying the coordinate transform, we sum each equation to get the following weak formulation for the EBT:

$$\int_{t_0}^{t_1}\int_0^1 ||F||\left(\rho A\left(\frac{\partial^2 v}{\partial t^2}\cdot\bar{w}\right)+\rho I_p\frac{\partial^2\phi}{\partial t^2}\bar{\psi}\right)d\xi dt+\int_{t_0}^{t_1}\int_0^1\frac{1}{||F||}\left(EA\frac{\partial v_x}{\partial\xi}\frac{\partial\bar{w}_x}{\partial\xi}+GJ_t\frac{\partial\phi}{\partial\xi}\frac{\partial\bar{\psi}}{\partial\xi}\right)d\xi dt$$
$$+\int_{t_0}^{t_1}\int_0^1\frac{EI_z}{||F||}\left(\frac{\partial}{\partial\xi}\left(\frac{1}{||F||}\frac{\partial v_y}{\partial\xi}\right)\right)\left(\frac{\partial}{\partial\xi}\left(\frac{1}{||F||}\frac{\partial\bar{w}_y}{\partial\xi}\right)\right)d\xi dt$$
$$+\int_{t_0}^{t_1}\int_0^1\frac{EI_y}{||F||}\left(\frac{\partial}{\partial\xi}\left(\frac{1}{||F||}\frac{\partial v_z}{\partial\xi}\right)\right)\left(\frac{\partial}{\partial\xi}\left(\frac{1}{||F||}\frac{\partial\bar{w}_z}{\partial\xi}\right)\right)d\xi dt=0.$$

As above, $v_x, v_y,$ and v_z represent translational displacement, mapped backed to the reference frame. Similarly, ϕ is the pullback of the twist. Here, $\bar{w}$ and $\bar{\psi}$ are the pullbacks of our test functions. This equation is much more straightforward to implement than the full continuum set of weak equations that had been previously discovered. Unfortunately, there is a major caveat to this equation that complicates the implementation of the above equations into FEM code. This weak formulation requires us to differentiate the Jacobian, which can be computationally difficult, and requires second derivatives. This increases the necessary regularity requirement for our shape functions, meaning we need cubic shape functions to satisfy our equation. These are much more burdensome computationally, and are difficult to implement. As a result of this regularity requirement, the group then decided to work with the Timoshenko Beam Theory, whose advantages will be discussed in the next section.

3.3 Timoshenko Beam Theory

The Timoshenko Beam Theory (TBT), removes the assumption of neglecting shear deformation that EBT utilized. This introduces two new rotational displacements to keep track of, but has the advantage of replacing the fourth order bending PDE's from EBT with two pairs of coupled second order PDE's Anderson and Nielson [2008]. As we will see, the lowered order of the derivatives in the strong equations will result in a weak formulation that is easier to implement numerically, and will only require linear shape functions. This exact property leads to Timoshenko being used prolifically in actual FEM simulations in industry. We now must keep track of 6 DOF in this problem (technically 12 for each beam element in the FEM code, as each element has 2 nodes). These DOF are u_x, u_y, u_z, the translational displacements, and $\theta_x, \theta_y, \theta_z$, our rotational displacements. We will have six PDE's in our strong form: the same axial and torsional equations as EBT, and the new coupled bending equations. The strong form for 3D Timoshenko is:

$$\rho A\frac{\partial^2 u_x}{\partial t^2} - \frac{\partial}{\partial s}\left(EA\frac{\partial u_x}{\partial s}\right) = 0, \quad \rho I_p\frac{\partial^2 \theta_x}{\partial t^2} - \frac{\partial}{\partial s}\left(GJ\frac{\partial \theta_x}{\partial s}\right) = 0,$$
$$\rho A\frac{\partial^2 u_y}{\partial t^2} + \frac{\partial}{\partial s}\left(\kappa_y AG\left(\frac{\partial u_y}{\partial s} - \theta_z\right)\right) = 0, \quad \rho I_z\frac{\partial^2 \theta_z}{\partial t^2} - \frac{\partial}{\partial s}\left(EI_z\frac{\partial \theta_z}{\partial s}\right) + \kappa_y AG\left(\frac{\partial u_y}{\partial s} - \theta_z\right) = 0,$$
$$\rho A\frac{\partial^2 u_z}{\partial t^2} + \frac{\partial}{\partial s}\left(\kappa_z AG\left(\frac{\partial u_z}{\partial s} + \theta_y\right)\right) = 0, \quad \rho I_y\frac{\partial^2 \theta_y}{\partial t^2} - \frac{\partial}{\partial s}\left(EI_y\frac{\partial \theta_y}{\partial s}\right) + \kappa_z AG\left(\frac{\partial u_z}{\partial s} + \theta_y\right) = 0.$$

While the physical constants in the above equations are largely similar to those in EBT, we have a few new ones: κ_y, κ_z, the Timoshenko shear coefficients which often depend on Poisson's ratio, ν. We again multiply by test functions, and integrate to convert things to the weak form. After applying this procedure to each PDE, and utilizing our coordinate transformation, we get the following four weak formulations:

$$\text{Axial: } \int_{t_0}^{t_1}\int_0^1 \rho A\frac{\partial^2 v_x}{\partial t^2}\bar{w}_x||F||d\xi dt + \int_{t_0}^{t_1}\int_0^1 \frac{EA}{||F||}\frac{\partial v_x}{\partial \xi}\frac{\partial \bar{w}_x}{\partial \xi}d\xi dt = 0$$
$$\text{Torsion: } \int_{t_0}^{t_1}\int_0^1 \rho I_p\frac{\partial^2 \phi_x}{\partial t^2}\bar{\psi}_x||F||d\xi dt + \int_{t_0}^{t_1}\int_0^1 \frac{GJ_t}{||F||}\frac{\partial \phi_x}{\partial \xi}\frac{\partial \bar{\psi}_x}{\partial \xi}d\xi dt = 0$$
$$\text{Bending (XY): } \int_{t_0}^{t_1}\int_0^1 \left(\rho A\frac{\partial^2 v_y}{\partial t^2}\bar{w}_y + \rho I_p\frac{\partial^2 \phi_z}{\partial t^2}\bar{\psi}_z\right)||F||d\xi dt + \int_{t_0}^{t_1}\int_0^1 \frac{EI_z}{||F||}\frac{\partial \phi_z}{\partial \xi}\frac{\partial \bar{\psi}_z}{\partial \xi}d\xi dt$$
$$+ \int_{t_0}^{t_1}\int_0^1 \kappa_y AG\left(\frac{1}{||F||}\frac{\partial v_y}{\partial \xi} - \phi_z\right)\left(\frac{1}{||F||}\frac{\partial \bar{w}_y}{\partial \xi} - \bar{\psi}_z\right)||F||d\xi dt = 0$$
$$\text{Bending (XZ): } \int_{t_0}^{t_1}\int_0^1 \left(\rho A\frac{\partial^2 v_z}{\partial t^2}\bar{w}_z + \rho I_p\frac{\partial^2 \phi_y}{\partial t^2}\bar{\psi}_y\right)||F||d\xi dt + \int_{t_0}^{t_1}\int_0^1 \frac{EI_y}{||F||}\frac{\partial \phi_y}{\partial \xi}\frac{\partial \bar{\psi}_y}{\partial \xi}d\xi dt$$
$$+ \int_{t_0}^{t_1}\int_0^1 \kappa_z AG\left(\frac{1}{||F||}\frac{\partial v_z}{\partial \xi} + \phi_y\right)\left(\frac{1}{||F||}\frac{\partial \bar{w}_z}{\partial \xi} + \bar{\psi}_y\right)||F||d\xi dt = 0$$

Note in the above equations that the two coupled equations have been combined into a single bending equation for each possible bending plane. As in EBT, v_x, v_y, v_z our the reference translational displacements, and ϕ_x, ϕ_y, ϕ_z are the reference rotational displacements. The components of $\bar{w}$ and $\bar{\psi}$ are our test functions, mapped back to the reference frame. While these formulas are technically more complicated than those in EBT due to the number of terms, we can see only first order spatial derivatives are present. This means that when implementing the equations in our FEM simulation, we can use linear shape functions, that will simplify the computation greatly. With the new formulas derived for Timoshenko, the shape optimization condition would again need to be

rewritten to account for the simplified physics and geometry. For the full continuum problem, the maximum von-Mises stress was calculated using the following KS Function:

$$\tilde{M}_a[v,\iota] = \frac{1}{a}\log\left(\frac{1}{\mathrm{vol}(M\times[t_0,t_1])}\int_{t_0}^{t_1}\int_M \exp\left(aM^2[v,\iota]\right)dydt\right)$$

Under our assumptions, this formula becomes:

$$\tilde{M}_a[v,\iota] = \frac{1}{a}\log\left(\frac{1}{L(t_1-t_0)}\int_{t_0}^{t_1}\int_0^1 \exp\left(a\sigma_{VM}^2\right)||F||d\xi dt\right), \quad L=\int_0^1 ||F||d\xi$$

It should be noted here that in the previous formulation, the assumption $\det F = 1$ was utilized. As a result, the volume normalization constant in the original formulation would not change. In this new formulation, that condition has been relaxed, and $||F||$ is not necessarily 1. We also now integrate over the physical space, mapped to the reference coordinate. As a result of these two changes, this volume coefficient can change, and we must consider the length integral above. To derive the shape optimization condition from this formula, we take the first variation of this equation with respect to the embedding and set it equal to 0. After some simplification, we arrive at the following form:

$$\delta_\iota \tilde{M}_a[v,\iota] = \frac{1}{a}\int_{t_0}^{t_1}\int_0^1 w(\xi,t)(\delta_\iota||F|| + a\delta_\iota(\sigma_{VM}^2)||F||)d\xi dt - \frac{1}{aL}\int_0^1 \delta_\iota||F||d\xi.$$

In the above equation, w is a weight function defined as follows:

$$w(\xi,t) = \frac{\exp\left(a\sigma_{VM}^2\right)}{\int_{t_0}^{t_1}\int_0^1 \exp a\sigma_{VM}^2 d\xi dt}.$$

The variation of the squared mises stress has a relatively simple form, as the stress is proportional to $\frac{1}{||F||}$:

$$\delta_\iota(\sigma_{VM}^2) = 2\sigma_{VM}\delta_\iota(\sigma_{VM}) = -\frac{2\sigma_{VM}^2}{||F||}\delta_\iota||F||.$$

With this expression, we can write everything as:

$$\delta_\iota \tilde{M}_a[v,\iota] = \int_{t_0}^{t_1}\int_0^1 w(\xi,t)\left(\frac{1}{a}-2\sigma_{VM}^2\right)\delta_\iota||F||d\xi dt - \frac{1}{aL}\int_0^1 \delta_\iota||F||d\xi.$$

Some simple calculus shows that:

$$\delta_\iota||F|| = \delta_\iota\sqrt{\frac{\partial\iota}{\partial\xi}\cdot\frac{\partial\iota}{\partial\xi}} = \frac{F}{||F||}\cdot\frac{\partial}{\partial\xi}(\delta_\iota\iota)$$

By applying this formula, and using integration by parts to transfer the derivative from the variation of ι, we arrive at our new shape optimization condition:

$$\frac{d}{d\xi}\left[\frac{F}{||F||}\left(\int_{t_0}^{t_1} w(\xi,t)\left(\frac{1}{a}-2\sigma_{VM}^2\right)dt - \frac{1}{aL}\right)\right] = 0.$$

The above formula provides a stationary condition for our optimization procedure, but actually constructing the optimization mathematically is a more difficult task that is covered in the next section.

3.4 Optimization

Before diving into the specific optimization mathematics that will be used in this project, some basic optimization theory will be discussed as a precursor. Gradient based methods are a widely employed technique to solve shape optimization problems Sokołowski [2026]. Due to their relative simplicity compared to other optimization methods, and the better familiarity of the group with gradient methods, the group chose to employ a gradient-based optimization method to update the shape. To start, the control points of the 3D splines were treated as the only optimization parameter. As stated in section 2.3, our goal in this optimization problem is to minimize the maximum stress found in the KS approximation of the maximum von-Mises stress. To create a gradient update, we must calculate the total derivative of this functional with respect to the control points. Using standard optimization notation where J is our objective function, u is the state variable, and p is the optimization variable, we can write the formula for this problem as follows:

$$\frac{dJ}{dp} = \frac{\partial J}{\partial p} + \frac{\partial J}{\partial u}\frac{du}{dp}.$$

The two partial derivatives of J are relatively straightforward and efficient to calculate. However, the term $\frac{du}{dp}$ can be very difficult to calculate, as it quantifies how each individual component of displacement will change as an individual control point changes. As a result, by introducing an auxiliary equation $g \equiv 0$, and an arbitrary scalar, λ, which is our adjoint variable, we define the quantity $\mathcal{L} = J - \lambda^T g$ Bradley [2024]. Through the chain rule, and a clever choice of lambda, we reduce our problem into two relatively straightforward equations:

$$\frac{\partial J}{\partial u} = \lambda^T \frac{\partial g}{\partial u}, \quad \frac{dJ}{dp} = \frac{\partial J}{\partial p} - \lambda^T \frac{\partial g}{\partial p}.$$

This first equation is the adjoint equation, that we can solve to uncover the critical value of lambda that simplifies the calculation. This second equation can now be used as a more efficient method to calculate our necessary shape gradient. By letting $A[v, \lambda]$ denote our weak formulation, with the test function selected to be λ, we can reformulate this for our problem as follows:

$$A[v, \lambda] = \int_{t_0}^{t_1} \int_0^1 M(\ddot{v}, \lambda) + K(v, \lambda) d\xi dt = 0,$$

$$\frac{d\tilde{M}_a}{d\iota} = \frac{\partial \tilde{M}_a}{\partial \iota} - \frac{\partial}{\partial \iota}(A[v, \lambda]), \quad \frac{\partial \tilde{M}_a}{\partial v} = \frac{\partial}{\partial v} A[v, \lambda].$$

To calculate our gradient at each step, we must first solve the adjoint equation, then calculate the gradient using the results of that solve, and the original forward FEM solve. We begin by discussing the adjoint solve. In this project, the discrete adjoint method will be utilized. To understand implementing this method, an explanation of the FEM solver helps: see section 4.3 for a detailed discussion of the FEM solver. We can conceptualize our FEM formula as follows:

$$M\ddot{v}_n + C\dot{v}_n + Kv_n = F_n \leftrightarrow K_{eff} v_n = F_n.$$

By noting that for each time step n, $K_{eff} v_n - F_n = 0$, we can utilize the following discrete version of A:

$$A[v_n, \lambda_n] = \sum_{n=0}^{N_t} \lambda_n^T (K_{eff} v_n - F_n) = 0.$$

By taking the derivative of this form with respect to v_n, we can arrive at the following expression for the adjoint system that must be solved:

$$\frac{\partial}{\partial v_n} A = K_{eff}^T \lambda_n = \frac{\partial \tilde{M}_a}{\partial v_n} + \text{history}.$$

There are a few things to note here. First, K and M are symmetric matrices, meaning K_{eff} will be as well. Once the derivative of $\tilde{M}_a$ is calculated and the history terms are added, it is very efficient to solve this matrix equation, as we already have the inverse of K_{eff} from the forward solve. At each time step, we will have a few extra history terms from the last step, due to the way the Newmark-β method is structured. While the forward solve goes from $t = 0$ to $t = T$, the adjoint solve works backward from $t = T$ to $t = 0$. To finish, we must calculate $\frac{\partial \tilde{M}_a}{\partial v_n}$ at each step. Differentiating our stress function with respect to v_n, we reach the following form after some simplification:

$$\frac{\partial \tilde{M}_a}{\partial v_n} = \int_{t_0}^{t_1} \int_0^1 w(\xi, t) \frac{\partial \sigma_{VM}^2}{\partial v_n} ||F|| d\xi dt = c_n \Delta t \int_0^1 w(\xi, t) \left(2\sigma_{xx} \frac{\partial \sigma_{xx}}{\partial v_n} + 6\tau \frac{\partial \tau}{\partial v_n} \right) ||F|| d\xi.$$

In the final expression, the time-dependent integral has been expressed in a form suitable for numerical evaluation. The coefficients c_n represent the temporal weighting associated with this discretization. The resulting expression provides the contribution of the stress sensitivity at each time level to the subsequent sensitivity calculation. While a little elaborate, these equations are relatively straightforward to calculate, as we only need to perform a linear algebra calculation, and no numerical differentiation will be necessary. With this formula, the adjoint load vector can be calculated, and we can now solve the adjoint equations at each time step. With λ_n known for each time, we can now move on to the task of calculating the gradient update. By applying our discrete formulation, and explicitly differentiating with respect to control point position, we arrive at the following equation for the derivative:

$$\begin{aligned} \frac{d\tilde{M}_a}{dP_i} &= \frac{\partial \tilde{M}_a}{\partial P_i} - \frac{\partial}{\partial P_i} \sum_{n=0}^{N_t} \lambda_n^T (K_{eff} v_n - F_n) \\ &= \frac{\partial \tilde{M}_a}{\partial P_i} + \sum_{n=0}^{N_t} \lambda_n^T \left(\frac{\partial F_n}{\partial P_i} - \frac{\partial K_{eff}}{\partial P_i} v_n \right) \end{aligned}$$

This equation gives the gradient of the objective function $\tilde{M}_a$ with respect to the position P_i of the i-th control point.

The first term, $\partial \tilde{M}_a / \partial P_i$, represents the direct influence of the control point position on the objective function. The summation term represents the indirect influence through the discretized governing equations. Here, λ_n denotes the adjoint variable at time step n, K_{eff} is the effective system matrix used in the forward calculation, v_n is the corresponding displacement vector, and F_n is the load vector.

The derivatives of F_n and K_{eff} with respect to P_i describe how the control point position affects the load vector and the effective system matrix, respectively. The summation over n accounts for their cumulative influence over all time steps.

Thus, the equation expresses the gradient as the sum of a direct contribution and an adjoint-based contribution, without explicitly evaluating the sensitivity of the forward displacement vector v_n to each control point position.

The detailed derivation and implementation of these terms are omitted here. In practice, the forward and adjoint equations are solved at each optimization iteration, and the resulting gradient is used to update the control point positions. By repeating this process, the pipe geometry can be optimized with respect to the prescribed objective function.

3.5 Additional Mathematics

3.5.1 Damped Timoshenko

One request that the team received in the feedback to their midterm presentation was to add damping to the Timoshenko weak form. Currently, damping is handled in the code through the use of Rayleigh damping, which constructs a damping matrix C as a linear combination of the mass and stiffness matrices derived from the weak form. See section 4.3 for a discussion of Rayleigh damping. One could initially try to model damping in the Timoshenko equation by adding a $c\frac{\partial v}{\partial t}$ term to each weak form, but this result in a rather simplistic matrix, C, that is diagonal. Since the mass and stiffness matrices are sparse but not diagonal, constructing C using Rayleigh damping results in a more robust dependence on the velocities, so we will use that as a starting point for incorporating stiffness. using Rayleigh damping as a guide, we arrive at the following updated weak form:

$$\text{Axial: } \int_{t_0}^{t_1}\int_0^1 \rho A\frac{\partial^2 v_x}{\partial t^2}\bar{w}_x||F||d\xi dt + \int_{t_0}^{t_1}\int_0^1 \left[\alpha||F||\left(\rho A\frac{\partial v_x}{\partial t}\bar{w}_x\right) + \beta\left(\frac{EA}{||F||}\frac{\partial^2 v_x}{\partial\xi\partial t}\frac{\partial\bar{w}_x}{\partial\xi}\right)\right]d\xi dt$$
$$+\int_{t_0}^{t_1}\int_0^1 \frac{EA}{||F||}\frac{\partial v_x}{\partial\xi}\frac{\partial\bar{w}_x}{\partial\xi}d\xi dt = 0$$

$$\text{Torsion: } \int_{t_0}^{t_1}\int_0^1 \rho I_p\frac{\partial^2\phi_x}{\partial t^2}\bar{\psi}_x||F||d\xi dt + \int_{t_0}^{t_1}\int_0^1 \left[\alpha||F||\left(\rho I_p\frac{\partial\phi_x}{\partial t}\bar{\psi}_x\right) + \beta\left(\frac{GJ_t}{||F||}\frac{\partial^2\phi_x}{\partial\xi\partial t}\frac{\partial\bar{\psi}_x}{\partial\xi}\right)\right]d\xi dt$$
$$+\int_{t_0}^{t_1}\int_0^1 \frac{GJ_t}{||F||}\frac{\partial\phi_x}{\partial\xi}\frac{\partial\bar{\psi}_x}{\partial\xi}d\xi dt = 0$$

$$\text{Bending (XY): } \int_{t_0}^{t_1}\int_0^1 \left(\rho A\frac{\partial^2 v_y}{\partial t^2}\bar{w}_y + \rho I_p\frac{\partial^2\phi_z}{\partial t^2}\bar{\psi}_z\right)||F||d\xi dt + \int_{t_0}^{t_1}\int_0^1 \left[\alpha||F||\left(\rho A\frac{\partial v_y}{\partial t}\bar{w}_y + \rho I_p\frac{\partial\phi_z}{\partial t}\bar{\psi}_z\right)\right.$$
$$\left.+\beta\left(\frac{EI_z}{||F||}\frac{\partial^2\phi_z}{\partial\xi\partial t}\frac{\partial\bar{\psi}_z}{\partial\xi} + \kappa_y AG||F||\left(\frac{1}{||F||}\frac{\partial^2 v_y}{\partial\xi\partial t} - \frac{\partial\phi_z}{\partial t}\right)\left(\frac{1}{||F||}\frac{\partial\bar{w}_y}{\partial\xi} - \bar{\psi}_z\right)\right)\right]d\xi dt$$
$$+\int_{t_0}^{t_1}\int_0^1 \frac{EI_z}{||F||}\frac{\partial\phi_z}{\partial\xi}\frac{\partial\bar{\psi}_z}{\partial\xi}d\xi dt + \int_{t_0}^{t_1}\int_0^1 \kappa_y AG\left(\frac{1}{||F||}\frac{\partial v_y}{\partial\xi} - \phi_z\right)\left(\frac{1}{||F||}\frac{\partial\bar{w}_y}{\partial\xi} - \bar{\psi}_z\right)||F||d\xi dt = 0$$

$$\text{Bending (XZ): } \int_{t_0}^{t_1}\int_0^1 \left(\rho A\frac{\partial^2 v_z}{\partial t^2}\bar{w}_z + \rho I_p\frac{\partial^2\phi_y}{\partial t^2}\bar{\psi}_y\right)||F||d\xi dt + \int_{t_0}^{t_1}\int_0^1 \left[\alpha||F||\left(\rho A\frac{\partial v_z}{\partial t}\bar{w}_z + \rho I_p\frac{\partial\phi_y}{\partial t}\bar{\psi}_y\right)\right.$$
$$\left.+\beta\left(\frac{EI_y}{||F||}\frac{\partial^2\phi_y}{\partial\xi\partial t}\frac{\partial\bar{\psi}_y}{\partial\xi} + \kappa_z AG||F||\left(\frac{1}{||F||}\frac{\partial^2 v_z}{\partial\xi\partial t} + \frac{\partial\phi_y}{\partial t}\right)\left(\frac{1}{||F||}\frac{\partial\bar{w}_z}{\partial\xi} + \bar{\psi}_y\right)\right)\right]d\xi dt$$
$$+\int_{t_0}^{t_1}\int_0^1 \frac{EI_y}{||F||}\frac{\partial\phi_y}{\partial\xi}\frac{\partial\bar{\psi}_y}{\partial\xi}d\xi dt + \int_{t_0}^{t_1}\int_0^1 \kappa_z AG\left(\frac{1}{||F||}\frac{\partial v_z}{\partial\xi} + \phi_y\right)\left(\frac{1}{||F||}\frac{\partial\bar{w}_z}{\partial\xi} + \bar{\psi}_y\right)||F||d\xi dt = 0$$

3.5.2 Expanding the General Theory

The most natural first step to expand the continuum theory would be to remove the restriction that $\det F = 1$. This limits the possible set of embedding maps greatly, and needs to be removed to create a more useful theory. The main caveat of removing this, is that F is no longer unitary, meaning $F^T \neq F^{-1}$. We should note we have effectively removed this restriction in the previous Timoshenko

theory, as this is relatively straightforward for our mapping. The most basic change removing this assumption makes is to our shape gradient expression:

$$\frac{\partial v}{\partial y} = \frac{\partial u}{\partial x}F \Leftrightarrow \frac{\partial u}{\partial x} = \frac{\partial v}{\partial y}F^{-1} = A.$$

Due to the frequency with which the matrix A appears in our expressions for strain energy, $U[v, \iota]$, and von-Mises stress, $M[v, \iota]$, this change in the definition of A is the main driver of the increased complexity we see in the resulting equations. Overall formulas for stress and strain remain unchanged, as removing this assumption on F does not fundamentally alter the physics of the problem. Instead now, we will see a more complicated dependence on the shape gradients in the resulting formulas. First, we note the revised expressions for stress and strain with weakened assumptions:

$$\sigma = \mu\left(F^{-T}\left(\frac{\partial v}{\partial y}\right)^T + \frac{\partial v}{\partial y}F^{-1}\right) + \lambda \text{tr}\left(\frac{\partial v}{\partial y}F^{-1}\right)I,$$

$$\varepsilon = \frac{1}{2}\left(F^{-T}\left(\frac{\partial v}{\partial y}\right)^T + \frac{\partial v}{\partial y}F^{-1}\right).$$

With these revised expression, the strain energy similarly changes to become:

$$U[u] = \frac{1}{2}\mu \text{tr}\left(\left(\frac{\partial v}{\partial y}F^{-1}\right)^2\right) + \frac{1}{2}\mu \text{tr}\left(\left(\frac{\partial v}{\partial y}F^{-1}\right)^T\left(\frac{\partial v}{\partial y}F^{-1}\right)\right) + \frac{1}{2}\lambda \text{tr}\left(\frac{\partial v}{\partial y}F^{-1}\right)^2.$$

Due to this difference in the strain energy term, our Lagrangian for the system is also changed, and when we go to re-derive the Equation of Motion for our system by minimizing the action, the overall equation keeps the same form:

$$\rho\frac{\partial^2 v}{\partial t^2} = \nabla \cdot [V]^T \text{ on } M,$$

$$[V]\hat{n} = 0 \text{ on } D_2,$$

but the expression for the tensor $[V]$ contains the changed terms:

$$[V]^T = \mu F^{-1}A^T + \mu F^{-1}A + \lambda \text{tr} AF^{-1}.$$

Our expression for von-Mises stress is unchanged in its dependence on A, much like with the strain energy term, but A itself has changed. Similarly, the KS-stress function actually remains unchanged. We are still integrating over the same fixed reference domain as in the case where F is more limited. As a result, we recover the same expressions for the KS stress and the von-Mises stress. When we go to take the variation of this stress, the overall expression remains the same up to the point where we derive the variation for the matrix A. This expression is notably different, and will take the form:

$$\delta A = -A\delta F F^{-1}.$$

Like with the matrix $[V]$, when we go to derive the expression for the matrix $[G]$, its expression will be more complex:

$$[G]^T = -6\mu^2 F^{-1}A^2 - 6\mu^2 F^{-1}A^T A + 2\mu^2 \text{tr}(A)F^{-1}A.$$

With $[G]$ redefined, the final resulting expressions are unchanged:

$$\int_{t_0}^{t_1} (a[G]^T \nabla (M[\iota, v]^2) + \nabla \cdot [G]^T) dt = 0 \text{ on } M \tag{11}$$

$$\int_{t_0}^{t_1} w[G]\hat{n} = 0 \text{ on } D_2 \tag{12}$$

$$\iota(y) = g(y) \text{ on } D_1 \tag{13}$$

Please note that we have maintained the revision to the original work, where the time dependence of M and $[G]$ is explicitly incorporated into the resulting PDE's.

4 Methodology

This section describes the numerical approach used to achieve the MVP. Before simulations could be developed in the Python environment, the group needed to develop a method to represent the embedding map, ι. Two ideas for this mapping will be discussed, that were tested by the group in Python. Following this, the numerical physics simulation, developed from the previously discussed mathematics will be explained. Depending on the success of the code, the finished work in solving the adjoint and constructing the optimization step will be discussed.

4.1 Explicit Embedding Approach

The first approach that the group attempted to implement was the explicit embedding approach. In this approach, the reference space, which we shall call I, is divided into multiple sub intervals I_i, such that $\cup I_i = I$. We then treat the embedding ι as being piecewise, such that each sub interval is mapped to a unique shape. To imitate a pipe-like shape, each sub interval is mapped into elbows, or rotated appropriately, and then linked together to form the physical shape.

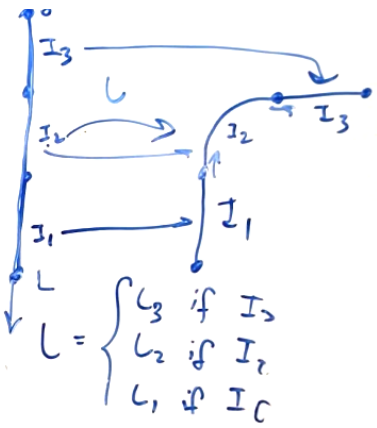


Figure 2: Whiteboard diagram of the explicit embedding approach.

The figure above shows a rough sketch of this process drawn on a whiteboard. This method was implemented in python, resulting in the following simulation:

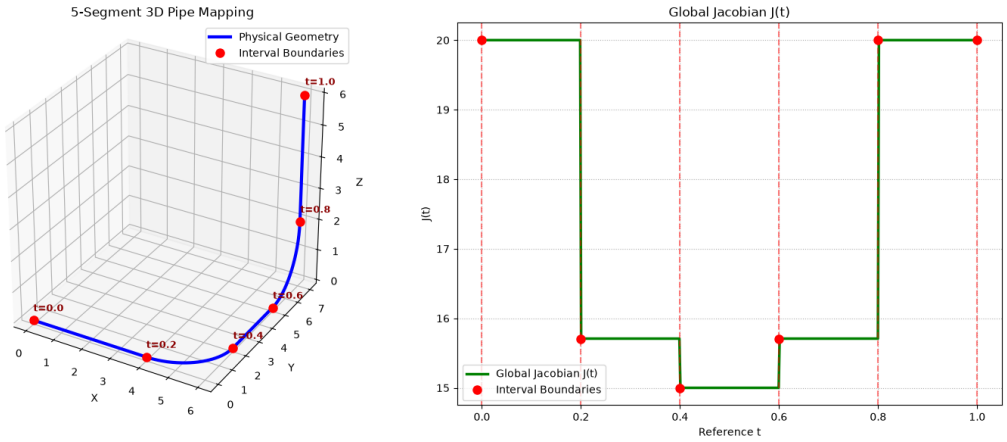


Figure 3: Implementation of the explicit approach, with Jacobian graphed.

From the above figure, we can see that even though this approach nicely creates industrially realistic pipe shapes, our Jacobian is very discontinuous. This is an undesirable property, as we must multiply and integrate against this function multiple times in our mathematical formulation. We must also take the derivatives of this gradient, which is a further headache. One could attempt to fix this by smoothing the gradient, but this both harms the quality of the physics engine, and creates a second derivative that contains many sharp sections. This can be seen in the figure below, with the smoothing implemented.

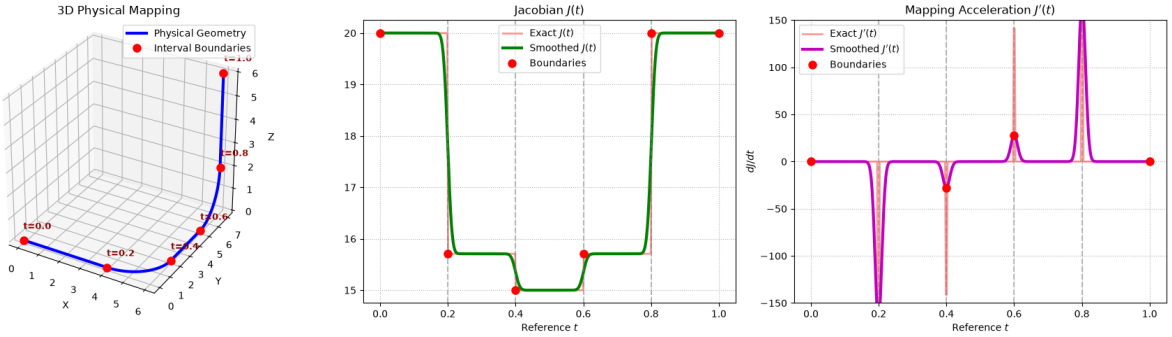


Figure 4: Implementation of the explicit approach, with the Jacobian smoothed.

4.2 Spline Polynomial Approach

The second approach the group implemented was the spline approach, where iota was constructed as a spline polynomial function. For this project, the group just focused on determining the shape of the pipe's centerline, and ended up mapping the 1D reference space into physical space by parameterizing each spline basis function on the reference frame. This approach was tested as splines are very common in shape optimization in industry, and possess desirable continuity and regularity properties. One draw back of splines is that they do not naturally conform to industrially realistic shapes. However, more realistic shapes can be achieved by locally refining the number of control points at elbows and bends. The figure below shows the results of initial testing with splines.

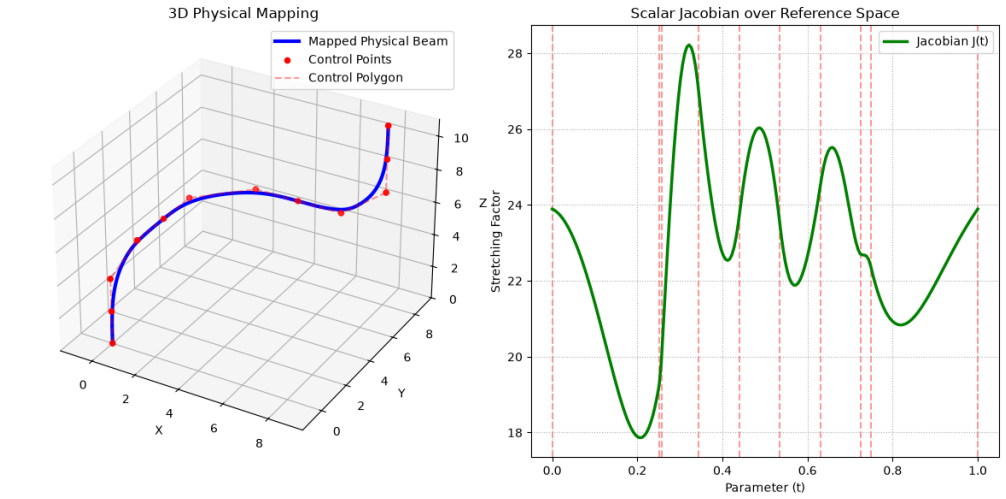


Figure 5: A test with cubic splines.

We can see in the test above that critically, the splines have a smooth, nicely behaved Jacobian. This is a desirable property. The figure below shows the results of experiments with additional control points, with an aim at achieving industrial pipe shapes. Toy physics are also implemented in this simulation, but the results should be neglected as they are inaccurate.

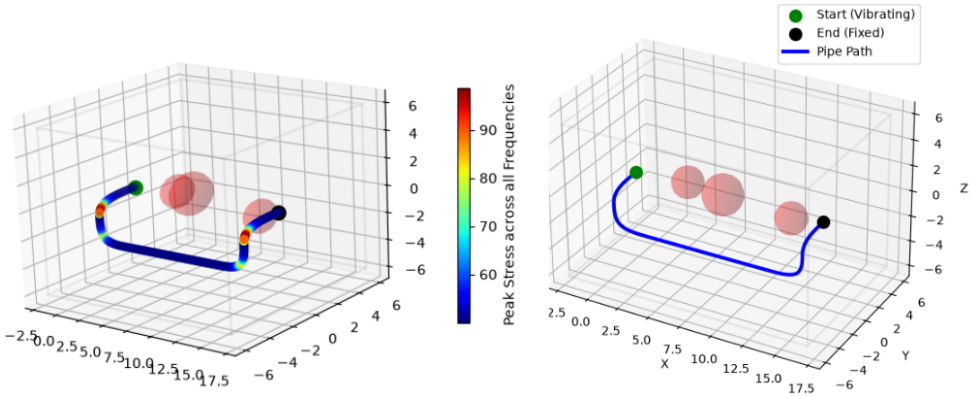


Figure 6: Splines avoiding obstacles, with additional control points to enforce better shape.

After examining both approaches, the group decided to go with the spline approach. The mathematical theory of splines is much more refined, and they are widely utilized in modern CAE software and shape optimization software. Furthermore, splines achieve much more desirable continuity and regularity. On the optimization side, splines can be controlled simply by modifying the position of control points in 3D space. This focus our optimization problem into one of position control points correctly. One should note that additional restrictions/algorithms may be necessary to ensure the splines maintain industrially viable shapes when updating the geometry.

4.3 Numerical Simulation

The numerical simulation of this PDE shape optimization problem naturally splits into two computational sub-parts: the dynamic physics state solver, namely the forward solver, and the geometric optimization loop. The forward solver maps the one-dimensional reference domain into three-dimensional Euclidean space, constructs the discrete differential operators corresponding to Timoshenko beam theory, integrates the equations of motion over time, and ultimately recovers the localized stress tensor. Next, the optimization loop evaluates the dynamic stress response, aggregated over both space and a frequency spectrum, and computes the necessary updates to the geometric mapping parameters. Because the objective landscape of the max vibrational stress is highly non-convex, with sharp resonance peaks, the optimization loop relies on smooth, differentiable approximation achieved by Kreisselmeier-Steinhauser (KS) functional. The success of this framework depends on the mathematical consistency and numerical stability of the forward solver, which we talk in more detail in subsequent subsections.

4.3.1 Forward Solver

The main challenge of optimizing a structure under dynamic loading is the necessity of repeatedly evaluating the state variables over a continuously deforming spatial domain. In shape optimization, the common problem is that the algorithm frequently tests extreme or highly distorted geometries as it searches the parameter space. These intermediate shapes often contain severe localized stiffness or concentrated mass, which make the resulting ordinary differential equations (ODEs) numerically stiff. For the optimization loop to successfully converge, the forward solver must reliably handle these intermediate shapes without crashing or returning false physics.

Traditional 3D shape optimization often relies on 3D volume remeshing, which can be computationally expensive and also has to be handled with lots of details and precision to avoid breaking the gradient calculations required by the optimizer. We avoid this by projecting the physics onto a 1D curve mapped into 3D space. This mapping is defined by a set of control points using piecewise quadratic and linear splines interchangeably.

The time-integration scheme must also be unconditionally stable to behave well during the optimization loop. To satisfy this, we propose a framework that combines a 3D Timoshenko beam finite element formulation solved dynamically via the implicit Newmark-β method.

The mass and stiffness matrices M_e and K_e are initially constructed in a local coordinate system, assuming the pipe segment points straight down a localized axis. To assemble the full pipe, these local matrices must be rotated into the global 3D room coordinates. We achieve this by calculating the geometric tangent, normal, and binormal vectors for each element. These vectors form a 12×12 rotation matrix R_{12} which smoothly transforms the local physical properties into the global system without distorting the underlying mechanics.

4.3.2 Temporal Discretization

Once the spatial properties of the pipe are mapped and discretized, the individual element matrices are assembled into a single global system. When stretching a static spring, the equation is easy, the force is just stiffness times displacement: $F = kx$. Meanwhile, in dynamic moving pipe, the force has the stiffness matrix K which takes the force to bend the pipe, the damping matrix C which takes the force to push through the friction, and the inertia matrix M which takes force to accelerate the mass. All of this dynamic behavior is governed by the fundamental second-order ordinary differential

equation of motion:

$$M\ddot{u}(t) + C\dot{u}(t) + Ku(t) = F(t)$$

In physical terms, this equation dictates that at any given moment t, the external forces applied to the pipe F must be balanced by three internal reactions: the inertia of the pipe's mass $M\ddot{u}$, the energy dissipation $C\dot{u}$ (also known as damping), and the structural elastic resistance to bending and stretching Ku. To evaluate the maximum stress, we must integrate this equation through time to find the global displacement history $u(t)$.

To step the system forward in time, one must choose a numerical integration scheme. Under the time discretization, we step the system forward in time from a known state at t_n to an unknown future state at t_{n+1}, where $t_{n+1} = t_n + \Delta t$.

A common approach in simple simulations is an explicit method, which calculates the next time step directly from the current one. However, explicit methods are only conditionally stable. If the optimization algorithm generates a pipe shape that is extremely stiff or light, the natural frequencies of the system will spike, and an explicit method would require an impossibly small time step to capture those frequencies without mathematically blowing up. This phenomena is also known as CFL condition Richard Courant [1928]. To explain what is essentially happening, let's Taylor expand around the current time t_n:

$$u_{n+1} = u_n + \Delta t\dot{u}_n + \frac{\Delta t^2}{2}\ddot{u}_n + \frac{\Delta t^3}{6}\dddot{u}_n + \dots$$

$$\dot{u}_{n+1} = \dot{u}_n + \Delta t\ddot{u}_n + \frac{\Delta t^2}{2}\dddot{u}_n + \dots$$

A purely explicit numerical method truncates these series by assuming the acceleration remains constant over the entire time step. However, because explicit methods only use past information to predict the future, they mathematically diverge, and hence blow up, if the time step Δt is too large relative to the highest natural frequency of the structure. Since our optimization algorithm actively manipulates the pipe to make it stiffer and lighter, an explicit method might break the optimization loop.

So, to advance the system in time, we choose to use the implicit Newmark-β method, as this method is unconditionally stable. The reason this method work whereas the classic explicit methods do not is because instead of truncating the Taylor series and ignoring the higher-order derivative terms, Newmark-β method implicitly approximate those higher-order terms by combining the known acceleration at the current step $\ddot{u}_n$ with the unknown acceleration at the future step $\ddot{u}_{n+1}$.

Newmark introduced two weighting parameters γ and β to control it, rewriting the Taylor series as:

$$\dot{u}_{n+1} = \dot{u}_n + \Delta t\left[(1-\gamma)\ddot{u}_n + \gamma\ddot{u}_{n+1}\right]$$

$$u_{n+1} = u_n + \Delta t\dot{u}_n + \frac{\Delta t^2}{2}\left[(1-2\beta)\ddot{u}_n + 2\beta\ddot{u}_{n+1}\right]$$

The method is therefore implicit. We cannot simply calculate the future state directly. Instead, we must enforce dynamic equilibrium at the future time step:

$$M\ddot{u}_{t+\Delta t} + C\dot{u}_{t+\Delta t} + Ku_{t+\Delta t} = F_{t+\Delta t}$$

Our goal is to find the future displacement $u_{t+\Delta t}$. The problem is we do not know the future acceleration $\ddot{u}_{t+\Delta t}$ or velocity $\dot{u}_{t+\Delta t}$ either. Following our choice of time-discretization scheme, the

Newmark-β method uses finite difference approximations to express future velocity and acceleration entirely in terms of the future displacement and known historical values $(u_{\Delta t}, \dot{u}_{\Delta t}, \ddot{u}_{\Delta t})$ from the previous time step. The Newmark-β method relates the future kinematics to the future displacement in the following way:

$$\dot{u}_{n+1} = \dot{u}_n + \Delta t\left[(1-\gamma)\ddot{u}_n + \gamma\ddot{u}_{n+1}\right]$$

$$u_{n+1} = u_n + \Delta t\dot{u}_n + \frac{\Delta t^2}{2}\left[(1-2\beta)\ddot{u}_n + 2\beta\ddot{u}_{n+1}\right]$$

Focusing on the displacement equation first, we move all known historical terms to the left side:

$$u_{n+1} - u_n - \Delta t\dot{u}_n - \frac{\Delta t^2}{2}(1-2\beta)\ddot{u}_n = \beta\Delta t^2\ddot{u}_{n+1}$$

Dividing through by $\beta\Delta t^2$ isolates the future acceleration:

$$\ddot{u}_{n+1} = \frac{1}{\beta\Delta t^2}(u_{n+1} - u_n) - \frac{1}{\beta\Delta t}\dot{u}_n - \left(\frac{1}{2\beta} - 1\right)\ddot{u}_n$$

Notice that future acceleration now only depends on the future displacement u_{n+1} and historic values $u_n, \dot{u}_n, \ddot{u}_n$.

Next, we substitute our new expression for future acceleration directly into the Newmark velocity equation:

$$\dot{u}_{n+1} = \dot{u}_n + \Delta t(1-\gamma)\ddot{u}_n + \gamma\Delta t\left[\frac{1}{\beta\Delta t^2}(u_{n+1} - u_n) - \frac{1}{\beta\Delta t}\dot{u}_n - \left(\frac{1}{2\beta} - 1\right)\ddot{u}_n\right]$$

Expanding the brackets and grouping the terms by their historical variables gives the isolated future velocity:

$$\dot{u}_{n+1} = \frac{\gamma}{\beta\Delta t}(u_{n+1} - u_n) + \left(1 - \frac{\gamma}{\beta}\right)\dot{u}_n + \Delta t\left(1 - \frac{\gamma}{2\beta}\right)\ddot{u}_n$$

which again, only depends on the future displacement u_{n+1} and historic values $u_n, \dot{u}_n, \ddot{u}_n$. We now substitute both of these expressions back into the equation of motion:

$$\begin{aligned} M\left[\frac{1}{\beta\Delta t^2}u_{n+1} - \frac{1}{\beta\Delta t^2}u_n - \frac{1}{\beta\Delta t}\dot{u}_n - \left(\frac{1}{2\beta} - 1\right)\ddot{u}_n\right] + \\ +C\left[\frac{\gamma}{\beta\Delta t}u_{n+1} - \frac{\gamma}{\beta\Delta t}u_n + \left(1 - \frac{\gamma}{\beta}\right)\dot{u}_n + \Delta t\left(1 - \frac{\gamma}{2\beta}\right)\ddot{u}_n\right] + \\ +Ku_{n+1} = F_{n+1} \end{aligned}$$

To transform this into a computationally solvable linear system $Ax = b$, we factor out the single unknown vector u_{n+1} and move every strictly known historical term to the right-hand side of the equality. Factoring out u_{n+1} on the left-hand side defines the so-called Effective Dynamic Stiffness Matrix K_{eff}:

$$\underbrace{\left[K + \frac{1}{\beta\Delta t^2}M + \frac{\gamma}{\beta\Delta t}C\right]}_{K_{eff}}u_{n+1} = F_{eff,n+1}$$

Note that the right-hand side is the Effective Force Vector $F_{eff,n+1}$. It consists of the actual external physical load applied at the future time step, plus the momentum carried forward by the mass and

damping from the previous time step. By carefully moving the historical terms across the equals sign, we arrive at the following algebraic formulation used in the computational solver:

$$\begin{aligned} F_{eff,n+1} = F_{n+1} + \\ + M\left[\frac{1}{\beta\Delta t^2}u_n + \frac{1}{\beta\Delta t}\dot{u}_n + \left(\frac{1}{2\beta} - 1\right)\ddot{u}_n\right] + \\ + C\left[\frac{\gamma}{\beta\Delta t}u_n + \left(\frac{\gamma}{\beta} - 1\right)\dot{u}_n + \frac{\Delta t}{2}\left(\frac{\gamma}{\beta} - 2\right)\ddot{u}_n\right]. \end{aligned}$$

By undertaking this algebraic transformation, we have converted a complex, coupled, second-order differential equation into a simple, static-equivalent linear system

$$K_{eff}u_{n+1} = F_{eff,n+1},$$

where the effective dynamic stiffness matrix is defined as:

$$K_{eff} = K + \frac{1}{\beta\Delta t^2}M + \frac{\gamma}{\beta\Delta t}C.$$

Because the geometry of the pipe, the mass, damping, and stiffness matrices remains completely constant throughout a single simulation, K_{eff} is constructed exactly once at $t = 0$. To step forward in time, the solver only needs to compute the history vector $F_{eff,n+1}$ and perform a highly efficient back-substitution, allowing the physics to be solved rapidly during the optimization loop.

Mathematical stability analysis of the Newmark method proves that unconditional stability is guaranteed if and only if the parameters satisfy the inequality

$$\gamma \geq 0.5 \quad \text{and} \quad \beta \geq \frac{1}{4}\left(\gamma + \frac{1}{2}\right)^2$$

Setting $\gamma > 0.5$ would introduce spurious numerical damping, causing the numerical scheme itself to artificially bleed energy out of the dynamic system, which does not correspond to the physical reality of our problem. Hence, $\gamma = 0.5$ is selected; this way we guarantee that the Newmark method preserves the exact physical energy of the elastodynamic system. The common choice for β is 0.25, which can be obtained by plugging $\gamma = 0.5$ into the stability criteria for β and set an equality (also know as Trapezoid Rule).

4.3.3 Rayleigh Damping

In solid mechanics, internal viscosity and structural damping are mathematically the exact same thing. Physically, when a steel pipe bends rapidly, it generates micro-heat; as a result, the internal material viscosity dissipates energy, preventing the pipe from vibrating forever (other reasons for energy dissipation are micro-plasticity, atmospheric drag, and more). Meanwhile, because the selected Newmark-β integration parameters $\beta = 0.25, \gamma = 0.5$ strictly enforce zero algorithmic damping, the numerical scheme perfectly preserves the total energy of the system. As a result, as no energy leaves the system, and the stress non-physically accumulates in the pipe over time, leading to incorrect testing. We can see in the Fig. 7 below how max von Mises stress of the pipe would accumulate, preventing us to correctly test it under different frequencies.

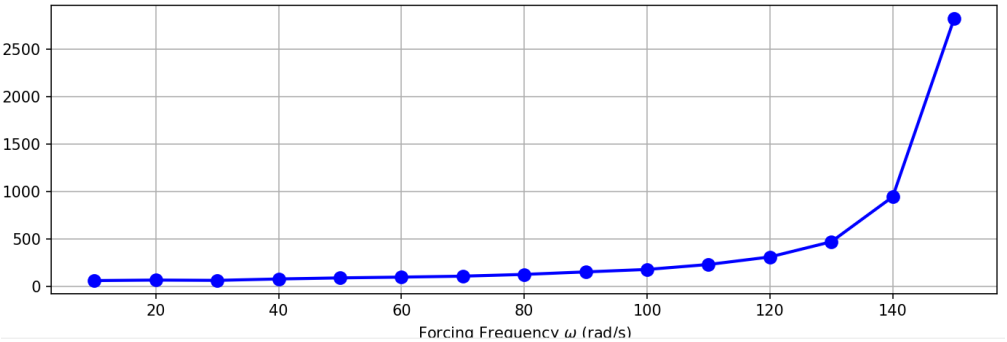


Figure 7: Max von Mises stress vs. Vibration frequency ω, plotted without the dumping term.

Therefore, to capture the physics of energy dissipation and allow the simulation to reach a bounded steady-state resonance, an explicit viscous damping matrix C was introduced into the global equations of motion.

Mass matrix M and stiffness matrix K are deterministically derived from known local material properties, such as density ρ, Young's modulus E, and other, and are integrated over the finite element domain. Unlike them though, localized damping coefficients are rarely known. Furthermore, constructing an arbitrary damping matrix C can mathematically ruin the system by corrupting the dynamic analysis.

To resolve this, we employ Rayleigh damping, also known as proportional. This classical approach defines the global damping matrix as a linear combination of the mass and stiffness matrices:

$$\mathbf{C} = \alpha M + \beta_R K$$

Remark: note that we use β_R instead of β to differentiate between the weighing coefficient β in the Newmark-β method and the stiffness term coefficient in the damping.

This formulation is nice because when M and K can be simultaneously diagonalized, any linear combination of the two will also be diagonalized by the exact same eigenvectors. This modal orthogonality property guarantees that the introduction of damping term does not artificially alter the shapes of the vibrating pipe.

When talking about damping, it is also helpful to look at the effective modal damping ratio ζ_n, which can be thought of as the percentage of critical damping. For any natural frequency ω_n of the system, in the Rayleigh damping, it is known to be:

$$\zeta_n = \frac{\alpha}{2\omega_n} + \frac{\beta_R \omega_n}{2}$$

The term $\frac{\alpha}{2\omega_n}$ is the mass-proportional term. It is inversely proportional to frequency and mimics external viscosity, such as the pipe moving through an air friction.

The term $\frac{\beta_R \omega_n}{2}$ is the stiffness-proportional term. It is directly proportional to frequency and mimics internal material viscosity, such as friction inside the steel.

When the harmonic forcing function $F(t)$ is first applied to the stationary pipe at $t = 0$, the sudden impact generates a massive, low-frequency sharp isolation. If it is left undamped, this transient state takes a long time to decay, forcing the numerical solver to run for extra time steps before the

true steady-state resonance emerges. We set $\alpha = 2.0$ to apply the aggressive damping specifically to these low frequencies. This acts as a mathematical shock absorber, reducing the initial transient startup motion and forcing the pipe into its steady-state rhythm faster. We can see the resulting motion of the vibrating end of the pipe over time in our simulation in the Fig. 12 at a later section.

Spatial discretization via finite elements inherently introduces non-physical, high-frequency displacements into the simulation; essentially microscopic numerical waves bouncing between the nodes. Because the stiffness-proportional damping scales linearly with frequency, we set $\beta_R = 10^{-4}$ to apply low damping to our realistic frequencies between $\omega \approx 5$ and $\omega \approx 40$ rad/s, while applying high damping to the unnaturally high frequencies.

4.3.4 Optimization Iteration

The optimization wrapper is being implemented following the mathematics derived for it. Meanwhile, that being said, multiple complications had been faced when implementing adjoint method; hence, the finite differences have been implemented instead.

One of the problems faced with the adjoint method was that the adjoint equation runs backwards in time ($t : T \to 0$). To calculate the adjoint λ_n, the adjoint operator requires to know the forward state u_n and tangent stiffness matrix $K(u_n)$. The problem arises through a time complexity: under many time steps in the simulations, saving the entire forward trajectory $u(t)$ to memory is hard. As a result, due to the limited resources of our project, we have to sacrifice the number of time steps, which limits the accuracy.

Another issue encountered in applying the adjoint method concerns the treatment of the sign(ε_a) function in the sensitivity calculation. This function appears in several parts of the implementation and takes the values $+1$ or -1. Its derivative is not defined at zero and may become unbounded there. Because the relevant quantities can pass through zero during dynamic simulations, this issue requires careful treatment in the adjoint formulation.

So, as mentioned above, to adjust to our limited time left in the program, we pivoted to using the finite differences in the implementation. The finite differences method is implemented in python and is more accessible for a time-limited project. Limitation is that finite difference is slow, as it attempts to evaluate all the directions of the gradient, while adjoint method simply points in the correct direction for optimizer to go to. We want to point out that the adjoint method is still preferred. This pivot was made due to the complexity of the implementation of adjoint and time limitations of this project.

The optimization code includes all of the implemented physics described in the prior section. One difference that had been made is the implementation of the splines: instead of cubic splines, the implementation uses the merge of the linear and quadratic splines to ensure the resulting optimized pipe is industrially looking. On top of that, it calculates the KS function in the `objective_ks_function`. In the main body, it defines the obstacle through two values of the 3D location of the center `OBS_CENTER` of the obstacle and the radius `OBS_RADIUS`, the initial guess through the array of control points in the `INITIAL_CPS`, and then runs the optimization using finite differences. It uses python implemented `minimize()` function with the `objective_ks_function` as the function we are optimizing and the `'L-BFGS-B'` method.

5 Route Refinement Algorithm

5.1 Purpose of Our Algorithm

In this section, we explain our algorithm for automated simulations. For a given evaluate function, the algorithm aims to obtain a set of optimal routes [1] satisfying several practical restrictions with keeping both of coverage and efficiency. Also, the process and outputs are required to match practical purposes.

5.1.1 Diversity of Routes

The algorithm has to generate distinct routes. For example, we know some existing route searching algorithms providing the shortest route between two endpoints such as A*. Certainly, it is a good tool for finding appropriate routes in a way. However, being the shortest does not mean being optimal in the context of pipe routing since the stress may be reduced by having extra parts. In this way, it has a risk of overlooking to generate only specific routes having mostly same properties. Moreover, outputs are also required to be different among themselves. In the designing process, several niche conditions would be added for production. If we have only routes giving similar pipes, these can easily rejected by them. Therefore, the algorithm needs to keep diversity of outputs, and thus it also be kept during the steps of selection.

5.1.2 Efficiency

It is clear that the computational resources are limited and checking plenty of routes drastically consumes the resources. This is the reason why the algorithm is simultaneously required to be efficient in terms of computation.
In our algorithm design, we suppose that most of the computational costs follow calling the evaluate functions, then we focus on reducing usage of them. Besides, entire evaluation costs much more than local evaluations does, thus the former can be used less as much as possible. [2]

5.1.3 For Practical Use

For more practically appropriate outputs, our algorithm has several features and takes some restrictions into account. Inserting U-turn segments and generating non axis-parallel legs enable us to create realistic pipe candidates. Moreover, outputs certainly satisfies the two conditions, avoiding obstacles and map walls (with room), and following fixed endpoint vectors.

5.2 Overall Flow

The algorithm consists of the following steps:

- Step1-1 Category route generation: Generate a large number of candidate routes for each combination of generation parameters such as numbers of curves.
- Step1-2 Diversity-Based Thinning: Mitigate the bias between parameter combinations that arises during generation, using diversity-based thinning via Farthest-Point Sampling (FPS) Gonzalez [1985].

[1] A route consists of control points, and the spline method makes the actual pipe shape.
[2] Of course, the local evaluation can not completely replace the entire evaluation. Then note that there is a trade-off between efficiency and accuracy.

- Step1-3 Upfront U-turn insertion: Insert several U-turn-shaped slack-segment variants into each candidate for reduce of stress, using stratified sampling to intentionally introduce diversity.
- Step1-4 Archive registration: Using the CVT-MAP-Elites method Vassiliades et al. [2018], generate "cells" based on set of each feature vector summarizing each route, and assign each route to one cell. During the step, only the best route is kept per cell, and thus an "archive" consists of the eventual cells.
- Step2-1, 2-2 Iterative optimization loop and Periodic U-Turn Re-Insertion: Repeatedly perturb vertex positions, and periodically try inserting new U-turns. Simultaneously, evaluate each attempt with the evaluation function and accept improvements as updates of the archive.
- Step3 Final selection: Output the score-sorted list of the final archive.

5.3 Detailed Description of Each Step

Step1-1: Category route generation

Given a map, which includes a start point, start vector, goal point, goal vector, and set of obstacles, this step generates multiple candidate routes for each combination of parameters that determine a route's geometric skeleton. Here, we have two approaches having different parameters.

In Angle approach, positions and angles of intermediate "bend points" are determined. The classification parameters are the following two:

- bend count: The total number of bend points.
- lateral spread factor: A coefficient controlling how the bend points are far from the main axis, which connects the start vector and the goal vector.

In Grid approach, maps are gridded, and every route consists of points at centers of grid cells, and thus it is restricted to only axis-aligned (X/Y/Z) lines. The classification parameters are the following:

- bend count: The total number of bend points.
- nominal detour count: The number of obstacle-avoidance detours anticipated at generation time.
- axis traversal order: The sequence specifying which of the X/Y/Z axes each leg [3] follows, and in what order (e.g., X→Y→X→Z).
- leg-length allocation profile: The shape describing how route length is allocated across legs.

Step1-2: Diversity-Based Thinning

The candidate pool from Step1-1 tends to be biased toward parameter combinations that are more likely to be generated. This step mitigates that bias using the following feature values that summarize each route's leg-length allocation, together with diversity-based thinning via greedy FPS.

- middle ratio: What fraction of the route's total leg length is occupied by the middle segments, excluding the legs adjacent to the start and goal.

[3] Here, a leg means a subsection between two bend points (or start or goal point).

- variation of leg lengths: The coefficient of variation of leg lengths (standard deviation divided by the mean).

Step1-3: Upfront U-turn insertion

This step inserts U-turn-shaped slack segments (hereafter, U-turns) for reducing stress into each candidate route. A single U-turn consists of four vertices: *entry*, V_1, V_2, and *exit*. *entry* and *exit* lie on the leg's main axis and mark where the U-turn's outbound and inbound paths branch off from, and rejoin, the original leg; V_1 and V_2 form the turn-back portion, offset from the main axis in the transverse direction. Figure 1 illustrates this structure and the variables that determine it.

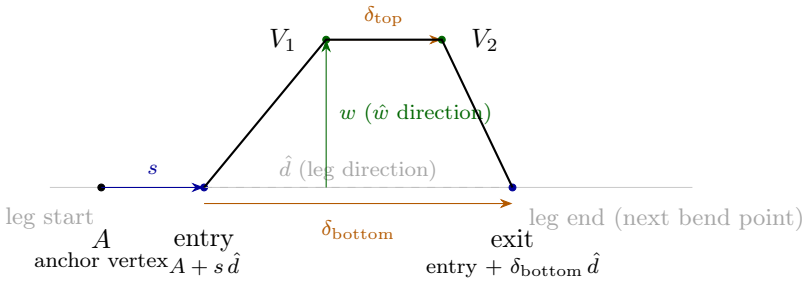


Figure 8: U-turn geometry. The trapezoid entry–V_1–V_2–exit lies in the 2D plane spanned.

- A: The anchor vertex (a discrete variable chosen from the existing vertex list); the origin from which the U-turn is constructed.
- $\hat{d}$: The unit direction vector of the leg into which the U-turn is inserted.
- s: The offset from anchor A along $\hat{d}$, defining entry $= A + s\,\hat{d}$. $s = 0$ is allowed (the U-turn then originates at the anchor vertex itself).
- δ_{bottom}: The spacing between *entry* and *exit* along $\hat{d}$ (the bottom-edge length of the U-turn), defining exit $=$ entry $+ \delta_{\text{bottom}}\,\hat{d}$.
- w: The transverse offset width.
- δ_{top}: The top-edge length, i.e. the spacing between V_1 and V_2 along $\hat{d}$. $\delta_{\text{top}} \le \delta_{\text{bottom}}$ yields a trapezoid that widens toward the base; $\delta_{\text{top}} = \delta_{\text{bottom}}$ yields a rectangle.
- shift (not drawn): How far the midpoint of the top edge (V_1–V_2) is offset along $\hat{d}$ from the midpoint of the bottom edge (entry–exit), i.e. asymmetry along the main axis. shift $= 0$ gives a U-turn symmetric about the entry–exit midpoint.
- ϕ (not drawn): The rotation angle of $\hat{w}$, the transverse direction of w about the axis of $\hat{d}$, parameterizing the U-turn's orientation in 3D.

Step1-4: Archive registration

This step constructs the "archive" in the Quality-Diversity optimization framework. MAP-Elites Mouret and Clune [2015] partitions the search space using an artificially defined discrete grid of "behavior descriptors" [4] and keeps only the best individual per cell, but suffers from an exponential growth in cell count as the number of behavior-descriptor dimensions increases. This pipeline

[4] In this case, these are the following 9 (or 8) parameters.

addresses that weakness by adopting CVT-MAP-Elites Vassiliades et al. [2018], which replaces the discrete grid with a Centroidal Voronoi Tessellation (CVT) over a continuous feature space, controlling the cell count to a fixed number k that does not grow with dimensionality.
Concretely, a CVT partitions the feature space into k regions, called *cells*, each represented by a centroid; a route is assigned to whichever cell's centroid is nearest to its feature vector. The set of centroids is learned once, by clustering the feature vectors of the candidate pool, and is then kept fixed for the rest of the algorithm, so that a given feature vector always maps to the same cell whole the algorithm. The *archive* is the data structure that stores, for each cell, the best-scoring route assigned to it so far: a newly evaluated candidate replaces the one currently stored in its cell only if its score is better, and cells that have not yet received any candidate simply remain empty. Because the number of cells is fixed at k regardless of how many feature dimensions are used, the archive's size does not grow as more descriptive features are added — this is the key advantage of CVT-MAP-Elites over a naively discretized grid, whose cell count would otherwise grow exponentially with dimensionality.

The continuous feature vector summarizing each route has 9 dimensions for the Angle approach, and 8 for the Grid approach (which lacks the notion of spread). [5]

1. n_{bends}: Number of bends
2. spread: Lateral spread factor, Angle approach only
3. middle_ratio: Ratio of middle-leg length
4. cv: Coefficient of variation of leg lengths
5. uturn_count: Total number of U-turns
6. uturn_position_mean: Mean position of all U-turns along the route
7. uturn_anchoring_mean: Mean anchoring degree [6] across all U-turn
8. uturn_size_mean: Mean size w across all U-turns
9. uturn_phi_concentration: Circular concentration of ϕ across all U-turns

Step2-1: Iterative Optimization Loop (Emitter Scheme)

This step performs repeated perturbation and evaluation of the candidates registered in the Archive, using an emitter-style optimization loop unified on CMA-ME. CMA-ME Fontaine et al. [2020] combines the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) Hansen and Ostermeier [2001] — an optimization method that repeatedly samples candidate solutions from a multivariate normal distribution and updates that distribution's mean and covariance toward the better-scoring samples — with the cell-based archiving of MAP-Elites, so that each archive cell is refined by its own self-adapting, increasingly directed search rather than by memoryless random perturbation. The loop has a nested structure of five levels, illustrated in the below Figure.

- Level-1 Round loop: The outermost repeated unit of Step2-1. The archive and the pool of emitters carry over from one round to the next without being reset, so progress made by an emitter persists across Rounds.

[5] All these variables are normalized.
[6] Anchoring degree: The number of bend points used as an entry or an exit for a U-turn.

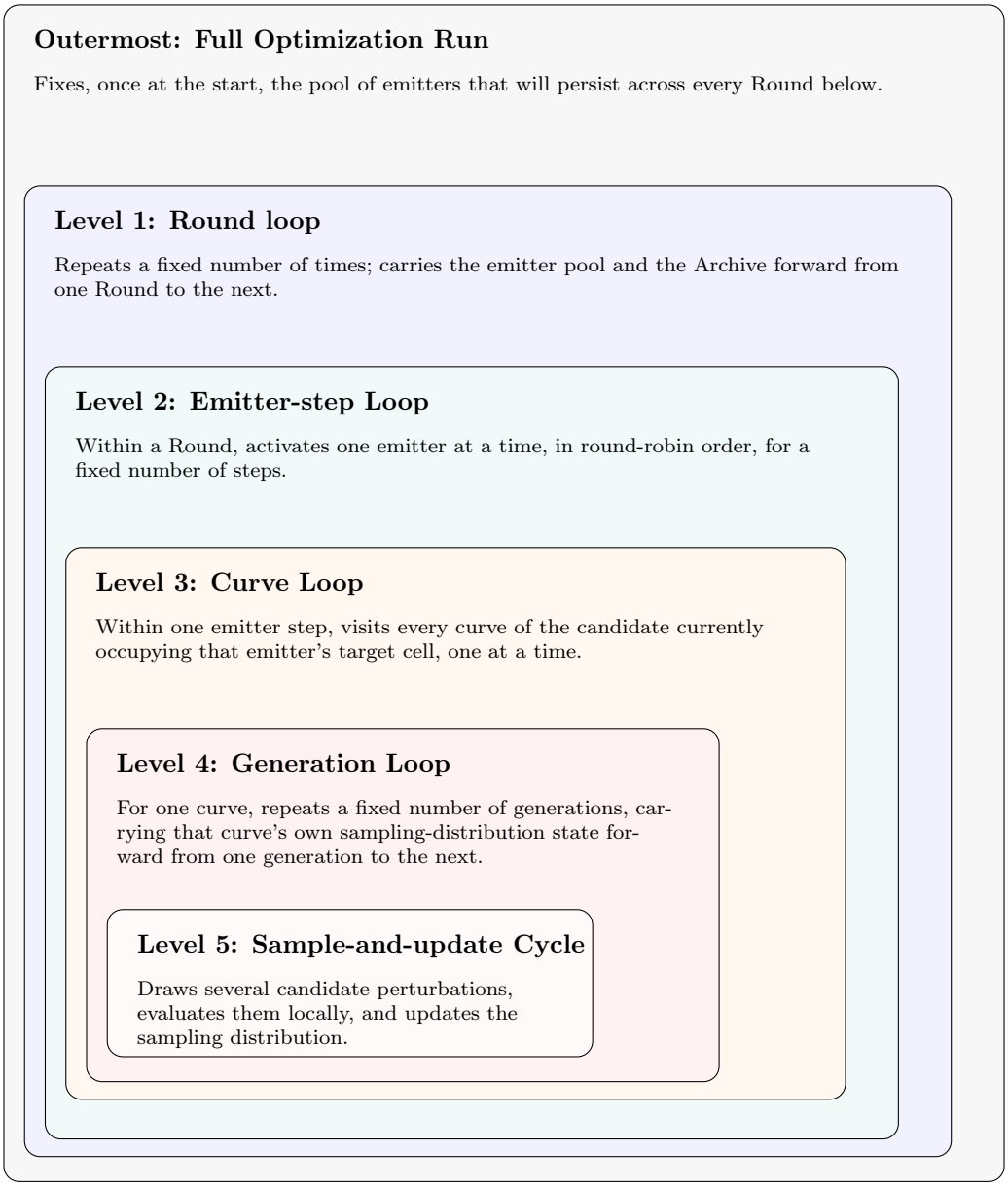


Figure 9: Nested structure of Step2-1. Each level is repeated a fixed number of times.

- Level-2 Emitter-step loop: Within a Round, emitters are activated one at a time in round-robin order. Each emitter is bound to a fixed archive cell and perturbs whichever candidate currently occupies that cell.
- Level-3 Curve loop: Within an emitter's step, every curve of the target candidate is visited in turn. Each curve keeps its own sampling-distribution state (mean, step size, covariance), so different curves can be perturbed independently, in whatever direction and at whatever scale has worked for them before. A curve's state is reset if it is missing or no longer matches that curve's current degrees of freedom.
- Level-4 Generation loop: For a single curve, a fixed number of generations run the sample-and-update cycle below, with the distribution from one generation carried into the next so that sampling concentrates on directions that previously reduced local stress. The best result across all generations is kept.
- Level-5 Sample-and-update cycle: The innermost step. Several perturbations are sampled, each scored by local stress plus a smooth penalty for obstacle or boundary violations, and the distribution's mean and covariance are updated toward the better-scoring samples. After all generations, the perturbed candidate is re-evaluated as a whole and placed into whichever archive cell it now belongs to, replacing that cell's occupant if it scores better.

Step2-2: Periodic U-Turn Re-Insertion (Optional)

Optionally, a periodic phase for inserting new U-turns is activated between fixed rounds. This is designed to enable targeted U-turn placement guided by evaluation feedback, going beyond what the upfront insertion in Step1-3 alone can capture — motivated by the design principle that stress-concentration locations cannot be known in advance without evaluation feedback.

Step3: Final Selection

From the final Archive, this step outputs a candidate list sorted by score via evaluation. The outputs provide structural information about the routes. The Archive itself is also retained separately, as a diverse pool of good candidates.

6 Results

6.1 Experiment Description

As explained earlier, the MVP consists of two parts:

- (1) the implementation of all the physics, and
- (2) the implementation of the optimization iteration.

First thing, let's talk about forward solver.

The earlier versions of the forward solver simulation use cubic splines as an embedding shape function for their mathematically nice nature. In the later versions of the forward solver that is used with the optimization iteration, we use piecewise linear and quadratic coupled splines to unsure that the resulting spline has a more realistic industrial pipe shape, incorporating straight segments and turns interchangeably.

The code consists of the `PipeProperties` class and functions that evaluate splines, find 3D matrices, and compute von Mises stress, along with the solver, constants, and visualization modules.

In the solver, under the discretized time, we apply the vibration test function of the form $\sin(\omega t)$ and then using the history of displacements, we evaluate the stress of the shape of the pipe.

To ground the numerical simulation in physical reality, the model is parameterized using the values for steel. The pipe is modeled with an outer diameter $D_{out} = 0.5$m and an inner diameter $D_{in} = 0.45$m, with thickness of 0.05m= 5cm and a hollow circular cross-section. The material properties are defined by Young's modulus $E = 200$GPa, shear modulus $G = 77$GPa, and mass density $\rho = 7850\text{kg/m}^3$.

Unlike in Euler-Bernoulli theory, Timoshenko beam theory includes the shear correction factors κ_y and κ_z, although the initial mathematical derivation assumes the shear stress is distributed uniformly across the cross-section. Physically, this is false. Shear stress acting on a cross-section must be zero at the surface and reaches a maximum at the neutral axis, creating a parabolic distribution. If we use the uniform shear assumption, we mathematically overestimate the pipe's shear stiffness. The shear correction factor κ is a dimensionless constant that scales the shear strain energy to account for this non-uniform parabolic distribution. For a thin-walled hollow circular cross-section, the exact theoretical value depends on Poisson's ratio, which is a measure of how a material deforms in width when you stretch or compress it in length. For most metals, the standard assumption for the Poisson's ratio $\nu \approx 0.34$. The sheer correctness factor is known to be

$$\kappa = \frac{2(1+\nu)}{4+3\nu} = \frac{2(1+0.34)}{4+3\times 0.34} \approx 0.534.$$

In our code, we approximate κ to be 0.5, and so we set $\kappa_y = \kappa_z = 0.5$, ensuring the finite element formulation captures the true transverse flexibility of the physical pipe.

Correspondingly, the Young's and Shear Modulus are related to each other and to Poisson's ratio as

$$E = 2 \times G(1+\nu) \quad \text{or} \quad G = \frac{E}{2(1+\nu)}.$$

Force is applied to the system as a boundary condition, that acts on the system through the vector F_{n+1}. For these initial experiments, the function that is used is a simple sine function of the form:

$$f(t) = 50000 \sin(\omega t),$$

where ω is the frequency that the user inputs.

6.2 Analysis of Results

6.2.1 Forward Solver

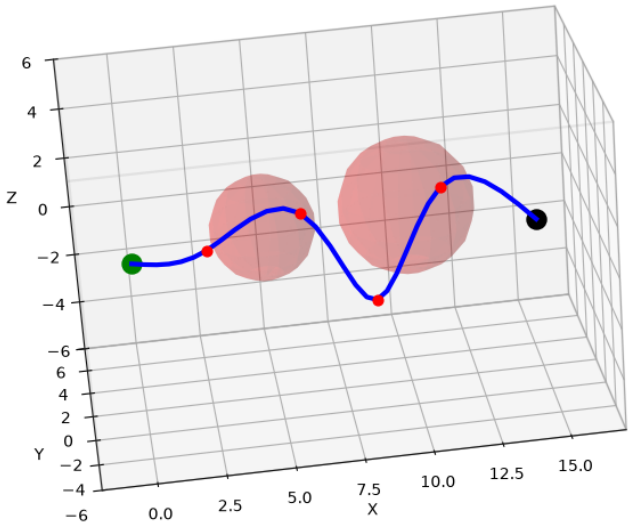


Figure 10: The 3D visualization of the tested pipe shape defined through cubic splines and 4 inner control points.

At the current stage, all of the physics has been successfully implemented in code. The implementation uses the weak form derived in Section 3.3. Following the simulation description above, it applies a vibration function to evaluate the stress of the structure and test the physics of the pipe.

In the Fig. 11 below, we can see the evaluated local stress of the pipe under the applied vibration.

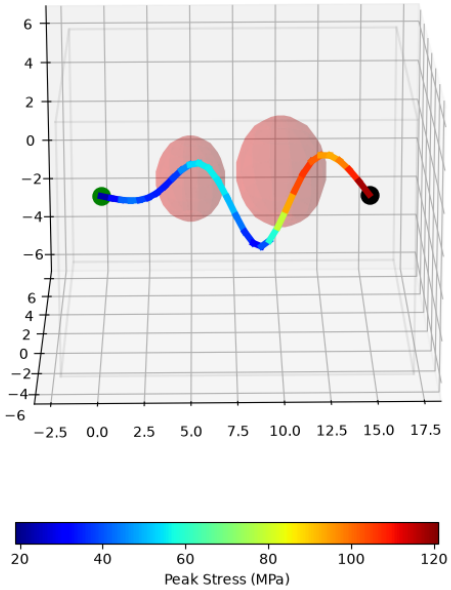


Figure 11: The 3D heatmap visualization of the local von-Mises stress of the tested pipe. The green end of the pipe is the vibrated end. The black end of the pipe is the fixed end.

As we can see, the stress accumulates closer to the fixed point of the pipe. This makes sense physically, because the rigidness of the fixed end has to absorb the vibration and therefore increases the tension in the structure.

It is important to note that while the 1D Timoshenko mapping provides an efficient and reasonably accurate evaluation of dynamic displacements and internal forces, it also inevitably introduces some abstractions. In reality, when a hollow pipe is bending, the cross-section does not remain perfectly circular; the inner curve and the outer curve compress, forcing the cross-section to flatten into an oval. This physical phenomenon significantly increases local flexibility and causes intense localized stress concentrations within pipe elbows.

Because our finite element formulation projects the geometry onto a 1D reference line, the cross-section is mathematically assumed to remain a rigid circle. Consequently, the forward solver strictly calculates the nominal bending stress and does not inherently resolve the 3D local stress concentrations found at the bends. For the purposes of this shape optimization framework, the nominal stress was utilized to maintain mathematical smoothness in the gradients. The nominal 1D stress serves as a perfectly valid, robust proxy for driving the structural topology away from resonance. For detailed strain analysis of the finalized shape, localized 3D shell modeling or SIF scaling can be applied as a post-processing step; although it is not currently part of the project and will be regarded as a potential for the future work.

As we can see in the Fig. 12 below, the solver successfully captures the transition from transient initial vibration impact to a steady-state vibration, demonstrating that the chosen Rayleigh damping coefficients γ and β properly isolated the steady-state resonant amplitude required by the optimizer.

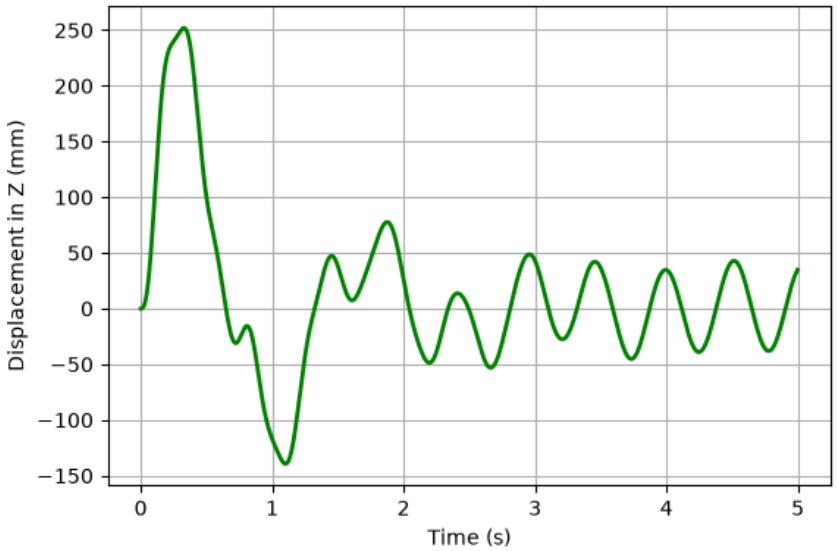


Figure 12: Displacement of the vibrating end of the pipe over time. We can observe the transient state of rapid oscillations for the first 2.5-3 sec before the convergence to the steady-state vibration.

In the Fig. 13 below, we can see our pipe being tested under different vibration frequencies ω. We observe that the max von Mises stress first dips down under the safety limit threshold; then it spikes up at the structure resonance frequency that happened to be around $\omega \approx 29$ rad/s. After which is goes back down under the safety limit threshold.

This is exactly the expected physical behavior of the vibrating structure, which once again confirms the correctness of our implementation of all physics of the given problem.

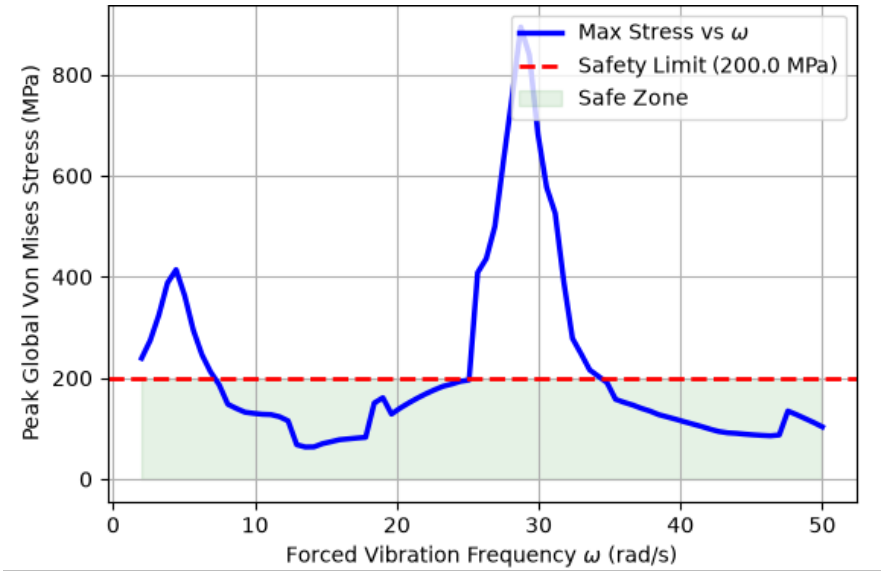


Figure 13: The max von Mises stress vs. vibration frequencies ω.

6.2.2 Optimization

Here we will discuss the results of the optimization iteration implementation. The details of how the simulation was implemented can be found in Sec. 4.3.4.

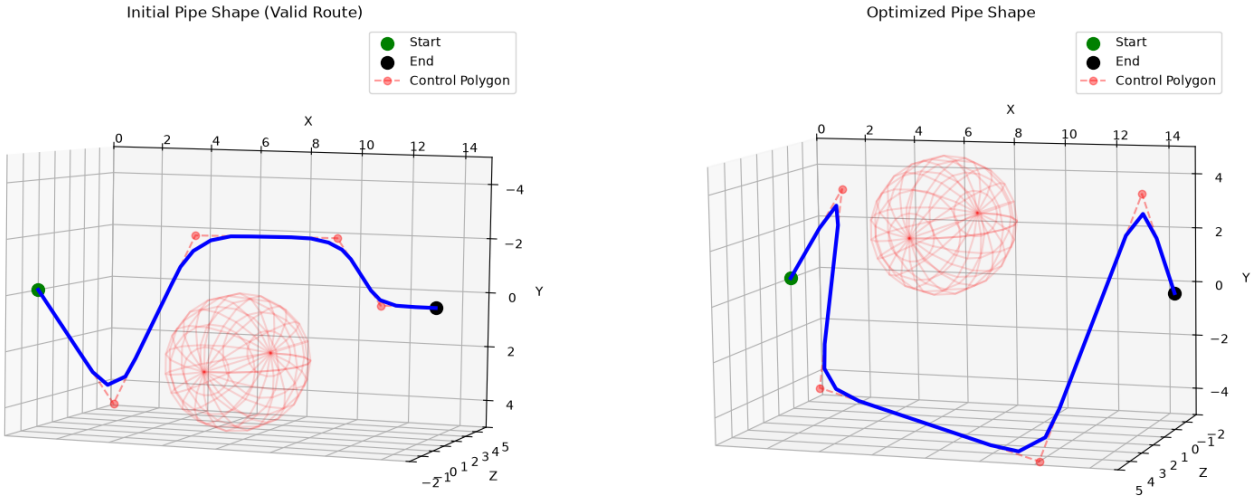


Figure 14: The 3D shape of the initial guess (on the left) and the optimized pipe shape (on the right).

Above you can see a figure of the initial guess and the optimized pipe. Current version of the optimization cycle successfully constructs a new pipe that avoids obstacles, which in the simulation has been implemented as a one big obstacle mimicking the compressor of the AC unit.

From the Fig. 15 below, we can see the heat map of the optimized pipe shape. We can observe natural accumulation of the stress closer to the fixed end of the pipe (black point), as well as at the elbows. This is the expected phenomenon which indicates that our simulation works correctly and along with the laws of physics.

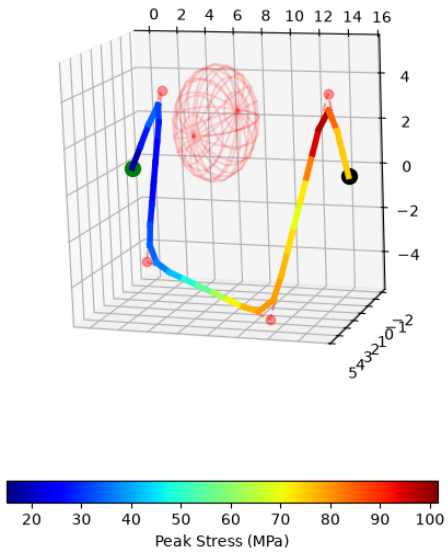


Figure 15: The peak von Mises stress heatmap (at $\omega = 12$ rad/s).

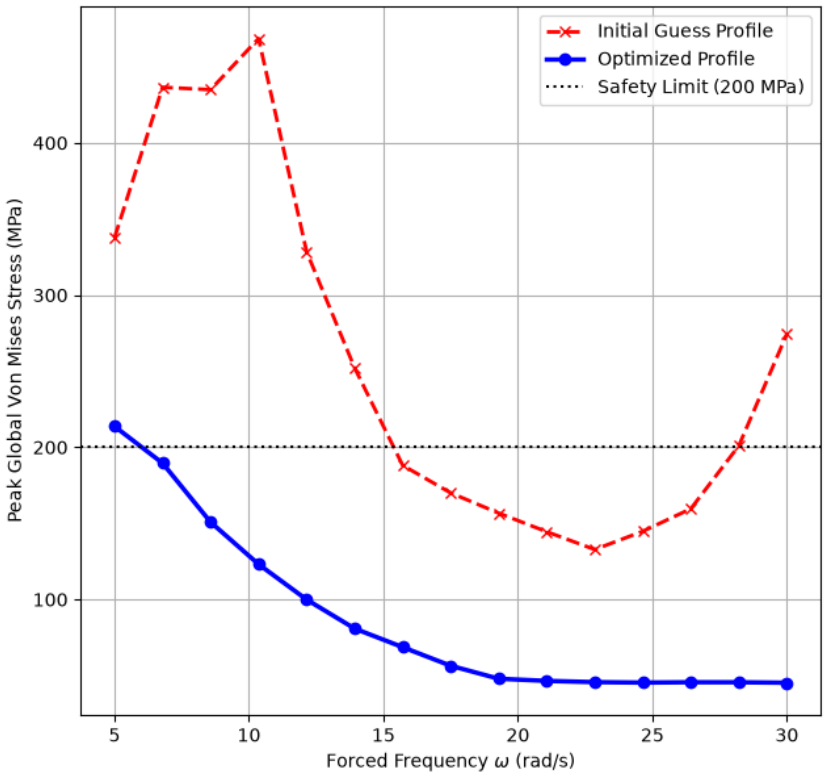


Figure 16: The max von Mises stress vs vibration frequencies ω graphs: comparison of the initial guess (red line) and the optimized shape (blue line).

Here you can see the graph of the frequency testing, where pipe is being vibrated under different frequencies ω. The red dashed line is the graph of the maximum von Mises stress of the initial guess pipe shape, and the blue line is the graph of the maximum von Mises stress of the optimized pipe shape. As you can see, the red line spikes up above the safety threshold at the resonance point, which again is an expected physical behavior of the solid structure under vibration. Meanwhile, the stress for the optimized pipe shape (the blue graph in Fig. 16) dips down and stays low under the safety threshold. This is an excellent result for the optimized pipe, showing that our simulation successfully found a shape that minimizes the stress of the pipe.

We want to point out that the optimized pipe is a suggestion for the physical pipe, not an actual pipe that must be built. We recognize that there are naturally industrial limitations to the pipes that can be manufactured. Specifically, we want to point out that the slight sharpness of the optimized pipe shape in the Fig. 14 is not ideal; this happened due to the limitations of mathematics and what a computer can simulate. With more control points and a finer space mesh, the simulation can achieve smoother curves for the optimized pipe. Similarly to the obstacle avoidance feature, additional constraints can be added to the simulation to ensure the optimized pipe does not have sharp turns.

6.2.3 Caveats

The first caveat we want to notice is that occasionally the optimizer flattens the pipe shape and the pipe becomes 2D in a 3D space.

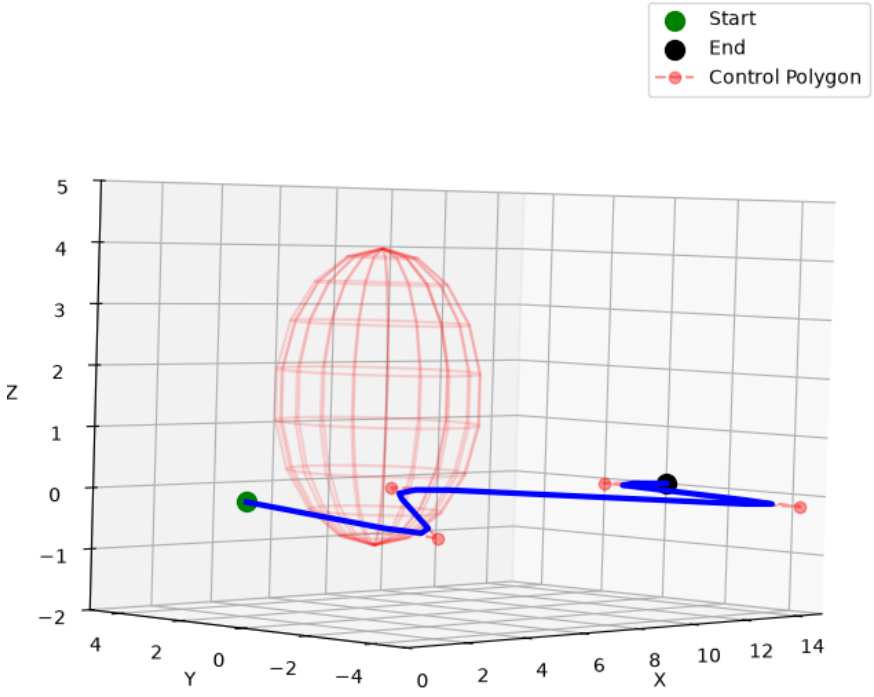


Figure 17: The optimized pipe that became 2D flat.

As for the reason why this happens, our educated guess is that the optimizer flattens the pipe because the testing vibration is a very simple **1D displacement** in the Z-axis $\sin(\omega t)$; see Fig. 12.

We believe that this is also likely the reason why the graph of the maximum von Mises stress of the optimized pipe shape (blue graph) in Fig. 16 does not show the spike at any resonance frequency ω.

To address the second caveat that we noticed, we invite you to take a look at the three graphs below.

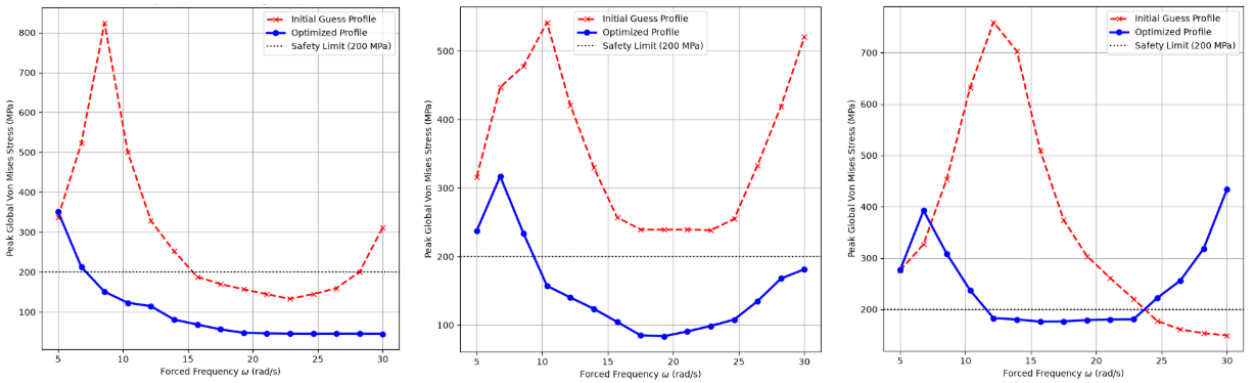


Figure 18: The comparison between frequency testing graphs under different initial guesses.

The first and the second figures behave quite nicely: the blue graph of the stress of optimized pipe decreases below the safety threshold fast and stays below that threshold. Meanwhile, in the third figure, we observe that although initially the optimized pipe behaves the way we want, around $\omega \approx 24$ rad/s the stress starts to increase and continues to go up until $\omega = 30$ rad/s. This is a major caveat of the current simulation.

The reason why this happens is because our KS function does not depend on the frequency ω. Recall that our optimization function is:

$$\tilde{M}_a[v, \iota] = \frac{1}{a} \log \left(\frac{1}{L(t_1 - t_0)} \int_{t_0}^{t_1} \int_0^1 \exp\left(a\sigma_{VM}^2\right) ||F|| d\xi dt \right), \quad L = \int_0^1 ||F|| d\xi$$

As the KS function does not depend on the frequency ω, we can optimize only over one frequency, or over a selection of frequencies at best. This is because the frequency ω is not a parameter in our optimization function and, therefore, it has to be fixed when performing the optimization.

As a result, the optimization function occasionally optimizes the pipe shape to merely shift the resonance point away from the chosen frequency rather than minimizing the maximum stress over all frequencies.

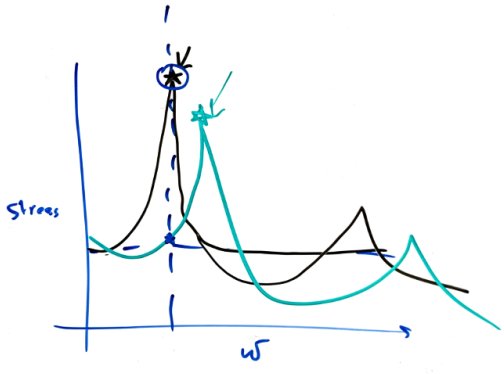


Figure 19: Discussion regarding how optimization function optimizes the stress at the chosen frequency rather than the peak stress over all frequencies.

We have attempted to resolve this by introducing the set of frequencies to test on instead of just one frequency. Unfortunately, the issue persists: the optimizer may optimize the max stress at given frequencies, but lead to the spike up of stress in the frequency in between. Meanwhile, we do believe that with bigger set up frequencies under a shorter distance between them we can make this problem less frequent and the spike will potentially be bounded by the distance between the chosen tested frequencies. It is also possible that re-framing the problem to be in the frequency domain would allow for better control over the stress experienced at individual frequencies, and help to mitigate the above issue.

7 Conclusion

7.1 Status of Deliverables and Limitations

The mathematical background for the minimum viable product has been completed. This includes the necessary physics and weak formulation, the corresponding adjoint equations to be solved, and the necessary formulas to calculate the gradient update. Currently the physics are limited to the 1D Timoshenko beam theory that was discussed prior, and do not represent a full continuum treatment of the problem. The optimization step currently only focuses on varying the control points of the splines, and does not change any cross sectional properties of the pipe. Due to issues in implementing the adjoint method derived prior, finite differences are used to calculate the necessary gradient at each step. A working optimization update step has been successfully constructed with this less efficient approach. The pipe routing algorithm has reached a working state, modified with suggestions from members of the Daikin team. Currently, it has not been implemented into the optimization step, as the team would like to refine that code before integrating the two algorithms.

7.2 Summary of Contributions

The mathematical background for the project, and the verification of Daikin's pre-thesis was completed by Davis Funk. The numerical simulation and optimization programs were created and tested by Anastasiia Rudenko. Itsuki Imai developed the route optimization algorithm for the team. Shota Nakagawa assisted the team with communications in Japanese, managing deadlines, and staying on track with the goals.

8 Future Work

8.1 Short-term Tasks / Remaining Implementations

A critical short term goal for the mathematical formulation would be to introduce cross sectional properties as another variable to optimize with respect to. Work on this will not commence until the team is happy with the state of optimization using the current, simpler formulation. For the numerical simulation, a number of short term tasks need to be addressed. As the current implementation is done through finite differences, as mentioned in the earlier section, one of the avenues for future work include fully implementing the adjoint method, as this is the preferred method. Another avenue is to apply testing over more complicated vibration function. A simpler initial task is to attempt to change the current vibration force to a more complicated model, built as the sum of multiple sine functions of varying frequencies. Other avenues include smoothing out the shape of the optimized pipe to avoid sharp angles and, as mentioned earlier, potentially moving away from KS function. Our pipe routing algorithm also has several short-term tasks. First, the code set has some improvement points from the perspective of UI and UX because it is not fully orchestrated. Second, it could be installed to receive feedback as gradient format from the evaluation function during perturbation. The feedback gives a clue for more accurate direction of perturbation with CMA-ME. Note that this task completely depends on a mathematical issue, how to obtain gradients. Third, some config parameters are left tentative so that users need to adjust them while running the algorithm in the actual process. Once the route refinement and optimization algorithms are to the groups' satisfaction, they must be integrated together into a complete package.

8.2 Long-term Research Directions

The main long term mathematical goal is to further refine the previous continuum model derived by Daikin in the pre-thesis Takiwaki [unpublished]. Building upon Daikin's previous work, the group has removed the assumption of $\det F = 1$. Removing this assumption did not fundamentally alter many of the equations, as we still universally integrate over the reference space. As a result, the loss of conservation of volume and orientation did not affect these integrals, since the reference space is unchanged even as restrictions on F change. Currently, the mathematics only focus on the reduction of stress. Of near equal importance to stress reduction is the reduction of noise. Future mathematical work will need to focus on introducing acoustic energy to the current formulation of the problem, and develop equations to help with the minimization of noise through a minimization of this acoustic energy. Acoustic structural simulation often incorporate stochastic methods in their development, which would require additional research by the team. A major long term goal for both the mathematics and numerical simulation is to reformulate the problem in the frequency domain. This will require work to rewrite the physics and optimization steps of the problem, and a new simulation the incorporates techniques of Fourier analysis into our FEM simulation. As the group members are not well versed in this topic, additional literature review will be required to achieve this goal. The group also believes a problem formulated in the frequency domain would allow for better control over the resonant frequencies of the structure through modification of the pipe shape. In terms of algorithm of simulation, this kind of algorithm is unprecedented, so it does not consists of one methodology, but a sets of existing theories. This means that the mathematical characterization of the algorithm could be a significant and tough problem.

Acknowledgments

The team would like to thank its academic mentor, Yuji Ando, for guidance and support during the project. We would also like to thank the industry mentors for their insightful advice and feedback and clear communication of Daikin's wishes. We would like to thank Daikin for providing us with the opportunity to tour and work at the Technology and Innovation Center in Osaka during the site visit. Lastly, we would like to thank IPAM and Tohoku University for facilitating the G-RIPS program, and providing us with this amazing opportunity to conduct research and experience Japan.

This work is an updated and extended version of the approach proposed in Takiwaki [unpublished].

Appendix

A Pseudo-code of the route refinement algorithm

Step 1-1: Generate a large batch of candidate route shapes
for each combination of (turn count, spread of turns) **do**
    Generate several random routes for this combination
    Keep the ones that stay in bounds, avoid obstacles, and bend realistically
    Add them to *candidates*
end for

Step 1-2: Narrow down to a diverse subset of those shapes
diverseRoutes ← a subset of *candidates* that covers a good spread of shapes

Step 1-3: Add optional U-turn detour variants to the diverse subset
for each route in *diverseRoutes* **do**
    Add the route itself to *pool*
    Also generate a few versions with a detour added, and add those too
end for
Score every route in *pool*

Step 1-4: File all resulting routes into cells and make an archive
Group *pool* into cells by shape similarity
archive ← for each cell, keep only its best-scoring route

Step 2-1, Step 2-2: Repeat perturbation, evaluation, and elimination many times
workers ← a handful of workers, each responsible for one cell
repeat
    for each worker **do**
        if worker has no cell, or its cell is gone, or it has been stuck for a while **then**
            Assign it a new cell (except filled cells) to work on
            Have it forget what it learned before
        end if
        Take the current best route in worker's cell
        Worker nudges that route's shape slightly, trying to lower its score
        Re-score the nudged route
        if it is better than before **then**
            Keep the improvement

```
        end if
        Let it compete for the best spot in whichever cell it now belongs to
    end for
    if it is time for a periodic check then
        for each route currently in the archive do
            Try adding one more detour to it
            if that is possible and the result is good enough then
                Keep it
            end if
        end for
    end if
until many rounds have been completed

Step 3: Stop. Sort the final archive by score
finalRoutes ← archive's routes, sorted best to worst
return the first N of finalRoutes
```

References

Lars Anderson and Soren Nielson. Elastic beams in three dimensions. DCE Lecture Notes No. 23, Aalborg University, Department of Civil Engineering, 2008. ISSN 1901-7268.

Francesco Ballarin, Andrea Manzoni, Gianluigi Rozza, and Sandro Salsa. Shape optimization by free-form deformation: Existence results and numerical solution for stokes flows. *Journal of Scientific Computing*, 60(3):537–563, 2014. ISSN 1573-7691. doi: 10.1007/s10915-013-9807-8. URL https://doi.org/10.1007/s10915-013-9807-8.

Andrew Bradley. Pde-constrained optimization and the adjoint method. DCE Lecture Notes No. 23, Aalborg University, Department of Civil Engineering, 2024.

Haung S. Choi and Edward J. Haug. Shape design sensitivity analysis of elastic structures. *Journal of Structural Mechanics*, 11(2):231–269, 1983. ISSN 0022-0396. doi: 10.2514/3.10093.

P. Duysinx and M. P. Bendsøe. Topology optimization of continuum structures with local stress constraints. *International Journal for Numerical Methods in Engineering*, 43(8):1453–1478, 1998. doi: https://doi.org/10.1002/(SICI)1097-0207(19981230)43:8<1453::AID-NME480>3.0.CO;2-2. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/%28SICI%291097-0207%2819981230%2943%3A8%3C1453%3A%3AAID-NME480%3E3.0.CO%3B2-2.

Thomas Eiter and Heikki Mannila. Distance measures for point sets and their computational complexity. In *Technical Report CD-TR 96/107, Christian Doppler Laboratory for Expert Systems, TU Vienna*, 1997.

Victor Eremeyev and Leonid Lebedev. Existence of weak solutions in elasticity. *Mathematics and Mechanics of Solids*, 18(2), 02 2013. doi: 10.1177/1081286512462187.

Chakib et al. Existence analysis of an optimal shape design problem with non coercive state equation. *Nonlinear Analysis: Real World Applications*, 28:171–183, 2016. ISSN 1468-1218. doi: https://doi.org/10.1016/j.nonrwa.2015.09.009. URL https://www.sciencedirect.com/science/article/pii/S1468121815001212.

Matthew C. Fontaine, Julian Togelius, Stefanos Nikolaidis, and Amy K. Hoover. Covariance matrix adaptation for the rapid illumination of behavior space. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference (GECCO '20)*, pages 94–102, 2020. doi: 10.1145/3377930.3390232.

Teofilo F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985.

Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001. doi: 10.1162/106365601750190398.

Claes Johnson. *Numerical solution of partial differential equations by the finite element method*. Courier Corporation, 2009.

Jean-Baptiste Mouret and Jeff Clune. Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909*, 2015.

Hans Lewy Richard Courant, Kurt Friedrichs. On the partial difference equations of mathematical physics. *Mathematische Annalen*, 100:32–74, 1928. doi: 10.1007/BF01448839.

Jan Sokołowski and Jean-Paul Zolésio. *Introduction to Shape Optimization: Shape Sensitivity Analysis*, volume 16 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, Berlin, Heidelberg, 1992. ISBN 3540541772, 9783540541776.

Jean Sokołowski. Shape and topology optimization for elasticity system using shape gradients and topological derivatives. *To appear / arXiv or conference proceedings if applicable*, 2026. Implementation available at: https://github.com/Tyiyiyi/Code-for-Shape-and-Topology-Optimization-for-Elasticity-System.

Michiaki Takiwaki. unpublished.

Srđan Trifunović and Ya-Guang Wang. Existence of a weak solution to the fluid-structure interaction problem in 3d. *Journal of Differential Equations*, 268(4):1495–1531, 2020. ISSN 0022-0396. doi: https://doi.org/10.1016/j.jde.2019.09.002. URL https://www.sciencedirect.com/science/article/pii/S0022039619303961.

Vassilis Vassiliades, Konstantinos Chatzilygeroudis, and Jean-Baptiste Mouret. Using centroidal voronoi tessellations to scale up the multi-dimensional archive of phenotypic elites algorithm. *IEEE Transactions on Evolutionary Computation*, 2018. Preprint first appeared 2016. https://arxiv.org/abs/1610.05729.