Final Report

Godot Cable Modeling

Los Alamos National Laboratory
David Mascarenas

Team Member Names:
John Mo, Mustafa Tekin, Jonathan Zhao, Thomas Holt, Jordan Daryanani

CSCE 482 – Senior Capstone Design
Spring 2025

Texas A&M University

Department of Computer Science and Engineering

Contents

1. Abstract

Cable management in nuclear waste tank cleanups is a critical challenge due to the complexity of the expensive and flexible cables interacting with the robot, rough terrain, and beams. Cable management has become a significant concern when utilizing these robots. Researchers at Los Alamos are interested in software solutions to better understand and predict cable behavior during hypothetical situations. Still, they do not fully understand the algorithms used to understand this behavior. We are making a cable modeling simulation utilizing the Godot game engine. This simulation explores the basic algorithms behind cable modeling. Our simulation uses primarily static physics calculations for tension and environmental force interactions. We aim to extend this and compare it to an explicit solution that is faster and better under different conditions. Assessments of our project include unit, integration, and user acceptance testing from the engineers who aim to benefit from our insights. We expect correct responses to user input and observability of all hidden and critical-to-understand features. Results should confirm realistic cable behaviour, with potential further refinements, and assess our ability to provide insights into the purpose of the algorithms explored. This project helps create a foundation of understanding for our sponsor and their peers. Our approach advises future efforts to automate cable management in complex and unpredictable environments.

2. Introduction

2.1 General Scope and Problem Background

2.1.1 Motivation

Currently, there are hundreds of tanks within the US that have nuclear waste in them. These tanks that store the nuclear waste need to be cleared and the waste has to be processed. With the dry retrieval technique the goal is to attempt to remove the nuclear sludge from the tanks with robots. By creating simulations of dry retrieval in game engines, researchers are attempting to understand and get ahead of possible pitfalls that may occur while doing the actual dry retrieval process. The problem at hand is the need for simulation that can accurately model and show physics behaviors, so that issues regarding the terrain, robot, and cables can be better understood. The current simulation lacks cables incorporated into it which limits the realism of the simulation within the tanks. This issue has not been addressed as the simulation is a work in progress. The issue of cable physics and behaviors need to be addressed as it directly correlates with accuracy and success of the simulation. The overall goal is to create the most realistic simulation to get ahead of possible issues once the actual dry retrieval process starts.

2.1.2 Domain Context

Our project is categorized within the robotic simulation and 3D modeling with applied physics, within the dry retrieval work process. It concerns an industry of industrial automation, hazardous material handling, robotics, and game engines. Our work is based in 2D, and future works may bring it to the 3D space.

Domain References

All of the existing simulations can create accurate mechanics to rigid-body mechanics, however, things like rope and cable mechanics are harder to implement with accuracy. Finite Element Methods and direct stiffness could be applied to create accurately responsive cable mechanics.

Domain Statistics

The industrial robotics industry is a 17 billion dollar industry, meaning that there are a lot of use cases and benefits of the industry. Over 50% of AR and VR simulations utilize game engines for industrial applications for real-time rendering and physics behaviors.

Domain Examples

Industries regarding robotic automation, hazardous waste retrieval, and space applications all can be related domains of examples relating to this project.

This project employs realistic cable physics in real-time within a game engine, while reacting to the movement/motion of the robot model within the tank during dry retrieval.

2.2 Goals and Objectives

The scope of this project is to expand on the existing dry retrieval system with a new robot model and 3D cable models that have the appropriate physics incorporated within them. The environment and foundation of the simulation is ready. Our work is based in the 2D space and developing complex physics simulations of cable properties. A detailed breakdown of the objectives regarding the project can be found in the Annex.

2.3 Design Constraints

While our project does not have many constraints, the main constraint is creating models that are too complex or detailed as it can affect the overall performance of the simulation within the game engine. Our model designs and physics implementations should be accurate, but at the same time simple enough to work efficiently with the existing simulation. Time is also a significant constraint and so we are focusing on the 2D section, aiming to develop complex physics calculations and simulate the cables in the 2D space.

2.4 Solution

Our proposed solution is to expand an existing simulation with the Godot game engine, by adding a new robot model as well as 3D cable models with realistic physics behaviors to ensure realism and accuracy to what may happen in the real world. Our solution is well designed as existing simulations already have the environment/foundation of a mock dry retrieval process. Our main priority is focusing on developing accurate and realistic 2D models of cable physics and properties through complex equations without the use of libraries. Our solution should be

able to give researchers and engineers an idea of how the cables move and animate and shall, in the future, expose possible issues that may occur within the tank.

2.5 Evaluation Plan

In order to evaluate our resulting solution, we plan on directly testing our cable models within the Godot game engine. Through the results of the simulation we should be able to understand if our solution has realistic physics behaviors. Our 2D cable models can also be compared to existing simulations with accurate cable physics implementations. The rest of the report must be about how we got to our resulting solution and aims to explain what we did in detail.

3. Related Work

Novelty: Our project distinguishes itself by bridging the gap between theoretical simulation models and practical integration in the Godot engine. As far as we know, no existing Godot plugin offers a fully documented and extensible implementation of both FEM and mass-spring simulations for cable dynamics. Users can modify parameters, enable or disable functions, choose conditions, or turn on and off simulation types with ease.

Justification: Our project aims for a middle ground between accuracy and usability. Godot, being open-source and highly modular, benefits from an extensible plugin that addresses both the visual and functional aspects of cable simulation. Our work provides an original contribution to this space, particularly through Godot-specific design choices, and is freely available to the open-source community.

Primary Types of Work:

- **Domain References:**
    - Realistic cable simulation has been a longstanding challenge in interactive and real-time applications, particularly robotics. The need for accurate, performant, and visually convincing cable physics is crucial. Cable dynamics typically involve modeling elasticity, damping, collision, and response to external forces. Many simulations rely on mass-spring systems or constraints-based methods, which while accurate, are computationally expensive. In game development, the balance between realism and performance often skews toward the latter, especially when targeting a broad range of hardware.
    - Cable dynamics simulation is foundational in computer graphics, robotics, and game development. Traditional approaches model cables using either the Finite Element Method (FEM), mass-spring systems, or rigid-body chains with joint constraints. Each technique presents trade-offs in terms of realism, computational load, and stability. Mass-spring systems are commonly used in video games due to their simplicity and efficiency. These systems are intuitive and lend themselves well to visual explanations, but suffer from numerical instability at low damping or stiff spring constants, especially under large deformations. On the other hand, the Finite Element Method offers a more rigorous treatment of elasticity and deformation. While accurate, FEM often requires more computation, making

real-time applications challenging without GPU acceleration or simplification strategies.

- **Direct References:**
    - A few existing Godot community plugins implement flexible ropes or cables, such as "Rope2D" and "SoftBody2D". These tools typically approximate cables using chains of rigid bodies connected by joints, or by applying mass-spring systems using built-in physics nodes. While functional for basic use cases, they lack extensibility and are rarely documented or optimized for simulation fidelity. For example, the “Rope2D” plugin achieves visual flexibility by linking rigid bodies with pin joints, but it becomes unstable at small segment sizes or under complex force configurations. It also abstracts away much of the physical configuration, limiting customization.
    - Another relevant tool is Unity's Obi Rope, a commercial asset that supports highly detailed rope physics using position-based dynamics. While Obi Rope is mature and performant, it is tightly coupled to Unity’s engine and closed-source, limiting its adaptability and transparency. Our system, in contrast, is open-source and fully integrated into Godot, designed to be lightweight and modular to encourage experimentation.
- **Peripheral References:**
    - In designing our mass-spring and FEM solvers, we’ve looked into other implementations from physics simulation libraries such as Box2D and PhysX. While our implementation does not directly use these libraries, their architecture and design patterns informed our approach to time integration, damping, and constraint enforcement.
    - We drew inspiration from video tutorials, such as the Non-linear Finite Element Analysis of 2D Catenary & Cable Structures using Python on engineeringskills.com, which provided the mathematical background for our simulation. The python implementation of these videos was used by us as a baseline for our implementation.

4. Engineering Standard

4.1 Environmental and Health/Safety Concerns

4.1.1 Environmental Concerns

The only considerations are power usage for development and running the simulations.

4.1.2 Health/Safety Concerns

There are no negative health and safety concerns. Users must avoid using the simulations for extended time periods for eye strain.

4.2 Social, Political, and Ethical Concerns

4.2.1 Social Concerns

We have been informed that there have been controversies surrounding the allocation of resources in the Hanford Dry Retrieval endeavor. We were advised to restrict our language when discussing the use of robotics such that we don't imply this is a strong or guaranteed solution.

4.2.2 Political Concerns

As stated above, the cleanup effort has proven to be far more expensive than originally anticipated. Since our project is peripheral to this cause, we probably don't need to worry about political ramifications very much. Wasting resources is never good, but the political sphere would probably remain untouched.

4.2.3 Ethical Concerns

This project is mostly theoretical, but it aims to demonstrate a way to change how dangerous waste cleanup is executed. It may be the case that using heavy machinery would slow down the cleanup, increase the leakage rate, or dramatically increase spending. Failing to acknowledge the uncertainty in these areas would be a failure on our part.

4.3 Manufacturability, Sustainability, and Economics

4.3.1 Manufacturability

For the Dry Retrieval Godot Project, there are many manufacturability considerations. The first one would be scalability to ensure that the simulations could be expanded and done at a larger scale faster with more robots and cables to test realism and physics. The parameterization of the cable properties would allow for variability and testing of different cable parameters within the simulation, getting a better understanding of limitations regarding certain cable properties.

Increasing the performance making the simulation run more optimally would also ensure that results and analysis are made quicker with less downtime. The modular design of the simulation and 3D models would allow for easier modifications and simulation trials.

4.3.2 Sustainability

There are three sustainability considerations for the Dry Retrieval Godot project which are: economic, environmental, and social.

For economic sustainability this project should be usable across a wide variety of platforms and computers, requiring little resources. It should also incorporate cable physics and robot movement in a realistic manner to ensure an accurate understanding of the Dry Retrieval process before attempting to pursue it in real life.

For environmental sustainability, this project should be built with efficiency and accuracy regarding the Dry Retrieval process with the robots and the cables. If the project does not deliver a good understanding of the Dry Retrieval process before attempting it, then nuclear waste could possibly result in hazardous conditions for the environment and site operators.

Regarding social sustainability, this project could lead to further simulation incorporation within the early stages of other projects, especially ones that may be difficult to execute and too complex to understand.

4.3.3 Economics

A successful implementation of this project could have great economic impacts, such as understanding shortcomings and pitfalls within the simulation before they could possibly occur in a real-life Dry Retrieval process. This would allow for a more efficient allocation of resources minimizing wasting of resources during the actual Dry Retrieval process. This project is essentially cost-free making it a great opportunity to understand issues that may be encountered in the tanks without ever having to interact with them.

5. Requirements
5.1 Overview

The Godot Cable Modelling Project aims to develop a real-time, semi-realistic cable simulation within the Godot Game Engine to assist in cable management planning for nuclear waste cleanup operations. Our solution seeks to model cable behavior, predict cable tension and load, prevent cable obstruction, and offer a 2D representation of cable physics properties. The user experience focuses on real-time and intuitive visuals that allow the engineers to analyze cable tension, identify risks, and make informed decisions on strategies. The minimum viable product features a basic Godot 2D cable simulation and a basic user interface that allows the user to adjust the cable start and end points, along with length, weight, and other variables. In the future, we would like to improve the MVP with better physics and tension calculations, and move it into the 3D space.

5.2 User Stories and Definition of Success

1. As an engineer,
   I want the cables to be modular
   So that I can simulate different cables
   - Acceptance Criteria: Cables must have variables which can be adjusted.
   - Definition of Done: Cables variables are editable and affect the behaviour.
2. As an engineer,
   I want the cables to have realistic physical properties
   So that I can test the impact of the cables on maneuverability:
   - Acceptance Criteria: Cables must exhibit realistic flexibility, weight, and resistance during movement.
   - Definition of Done: Simulations and physical tests confirm accurate cable behavior.
3. As an engineer,
   I want to be able to dynamically change physical properties of the cables
   So that I can test the different physical implications of materials:
   - Acceptance Criteria: Users can adjust cable properties such as stiffness, weight, and elasticity in real-time.
   - Definition of Done: The system allows configurable cable properties and reflects changes accurately in simulations.
4. As an engineer,
   I want the cables to have collision detection
   So that I can decide whether or not the cables are realistic in the real-world:
   - Acceptance Criteria: Cables must detect and respond to collisions with objects, surfaces, and the robot.
   - Definition of Done: Collision detection is implemented, tested, and accurately reflected in simulations.

5. As an engineer,
   I want the cable to collide with itself
   So that I can see whether or not entanglements may occur:
   - Acceptance Criteria: The cable must detect and respond to self-collisions realistically.
   - Definition of Done: Self-collision detection is implemented, tested, and accurately represented in simulations.
6. As an engineer,
   I want to control the robot's movement manually,
   so that I can test specific movements and their effects on cable behavior:
   - Acceptance Criteria: Users can manually control the robot's movements in real time.
   - Definition of Done: Manual control is implemented, tested, and accurately influences cable behavior.
7. As an engineer,
   I want to have a realistically sized robot model,
   So that I can test the size constraints of the robot in different environments:
   - Acceptance Criteria: The robot model must match real-world dimensions and proportions.
   - Definition of Done: The model's size is implemented, verified, and tested in various environments.
8. As an engineer,
   I want to have a mode where I can slow down or pause the simulation,
   So that I can examine the simulation more clearly:
   - Acceptance Criteria: The simulation can be slowed down or paused at any point during the test.
   - Definition of Done: The slow down and pause functionality is implemented and works smoothly during simulations.
9. As an engineer,
   I want to be able to load different maps
   So that I can easily test different environments for the robot:
   - Acceptance Criteria: Users can load and switch between different environmental maps during simulations.
   - Definition of Done: Map loading functionality is implemented and tested with various environment types.
10. As an engineer,
    I want real-time data on the stress and strain of the system
    So that I can understand and assess points of failure:
    - Acceptance Criteria: Real-time data on stress and strain is available and updated continuously during the simulation.
    - Definition of Done: Stress and strain data is implemented, displayed, and accurately reflects the system's conditions during tests.
11. As an engineer,
    I want to be able to save cable data
    So that I can analyze the data further over a longer period of time:

    - Acceptance Criteria: Users can save cable data for future analysis in a specified file format.
    - Definition of Done: Data saving functionality is implemented and successfully stores cable data for later retrieval.
12. As an engineer,
    I want the ability to generate graphs of tension over time
    So that I can analyze when tension was greatest:
    - Acceptance Criteria: The system can generate and display graphs showing tension data over time.
    - Definition of Done: Graph generation is implemented and accurately reflects tension data during the simulation.
13. As an engineer,
    I want the simulation to provide warnings of unsafe tension levels
    So that I can avoid potential breakages:
    - Acceptance Criteria: The system provides alerts or warnings when tension exceeds predefined safe limits.
    - Definition of Done: Warning system is implemented and correctly triggers when unsafe tension levels are detected.
14. As an engineer,
    I want a feature that allows the robot to run automatically
    So that I can see different cable performances under the same circumstances:
    - Acceptance Criteria: The robot can be set to run autonomously, performing predefined tasks or movements.
    - Definition of Done: The automatic mode is implemented and successfully runs the robot under various scenarios.
15. As an engineer
    I want the cables to have tension-dependent snapping behavior
    So that I can identify failure points and ensure realistic material constraints:
    - Acceptance Criteria: The cables snap or break based on tension thresholds, simulating real-world material behavior.
    - Definition of Done: Tension-dependent snapping is implemented and accurately reflects the expected material limits during simulations.
16. As an engineer,
    I want the ability to compare simulation results from multiple test runs,
    So that I can evaluate how different cable setups impact overall performance:
    - Acceptance Criteria: Users can view and compare results from multiple simulations side-by-side.
    - Definition of Done: Comparison functionality is implemented and allows for clear analysis of different test results.
17. As an engineer,
    I want the robot to respond to cable resistance,
    So that I can evaluate how cable tension affects the robot’s movement:
    - Acceptance Criteria: The robot’s movement must be affected by the cable’s resistance and tension in real time.
    - Definition of Done: Cable resistance is integrated into the robot's movement system and tested to reflect realistic behavior.

18. As an engineer,
    I want a user-friendly interface for modifying cable properties,
    So that I can easily test different materials and configurations:
    - Acceptance Criteria: The interface allows easy modification of cable properties, such as material, length, and tension limits.
    - Definition of Done: The user interface is implemented, intuitive to use, and successfully modifies cable properties in real-time.
19. As an engineer,
    I want the cables to experience friction against surfaces,
    So that I can accurately model resistance when the cables are dragged:
    - Acceptance Criteria: The cables experience friction when in contact with surfaces, affecting their movement and tension.
    - Definition of Done: Friction is implemented and accurately affects the cables' behavior during simulations.
20. As an engineer,
    I want to visualize stress points on the cables in color-coded overlays,
    So that I can quickly assess areas at high risk of failure:
    - Acceptance Criteria: Stress points on the cables are displayed with color-coded overlays that indicate varying levels of stress.
    - Definition of Done: Stress visualization is implemented and accurately updates during simulations, clearly identifying high-risk areas.

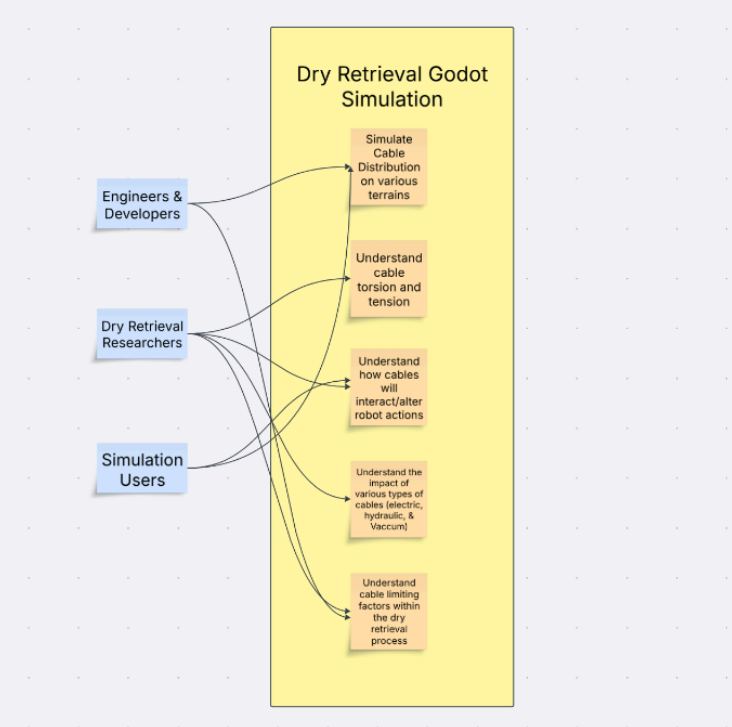
For an entire sprint, success is measured by:

- All user stories meeting their acceptance criteria and definition of done.
- All code passing unit tests and integration tests.
- No critical bugs or regressions introduced.
- Project documentation updated to reflect new features.
- Stakeholder approval for the completed work.

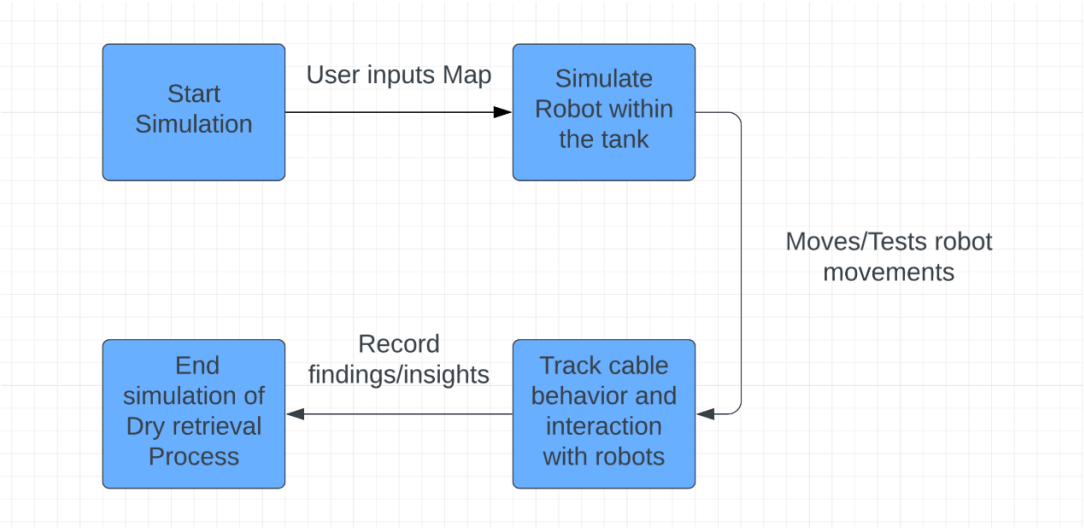
By maintaining clear definitions of success at both the individual user story and sprint levels, the project ensures consistent progress and high-quality implementation.

5.3 Requirements Models

Function Model

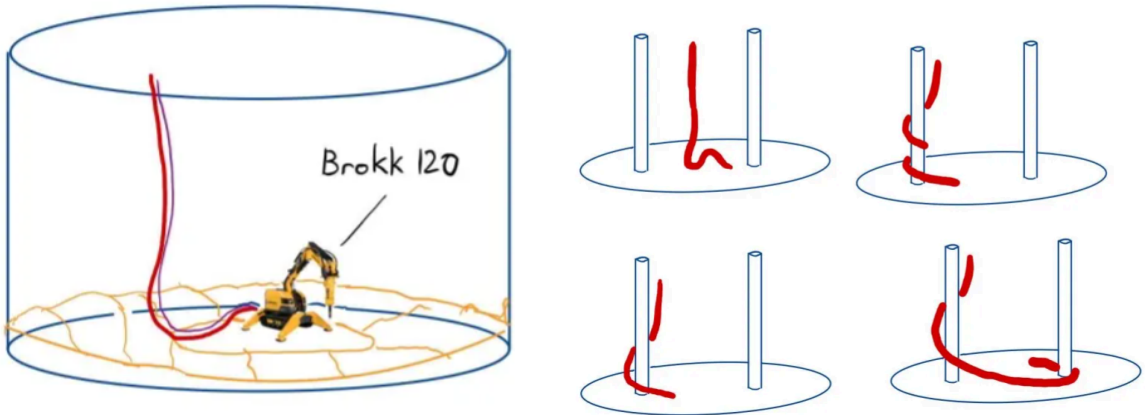


Behavior Model

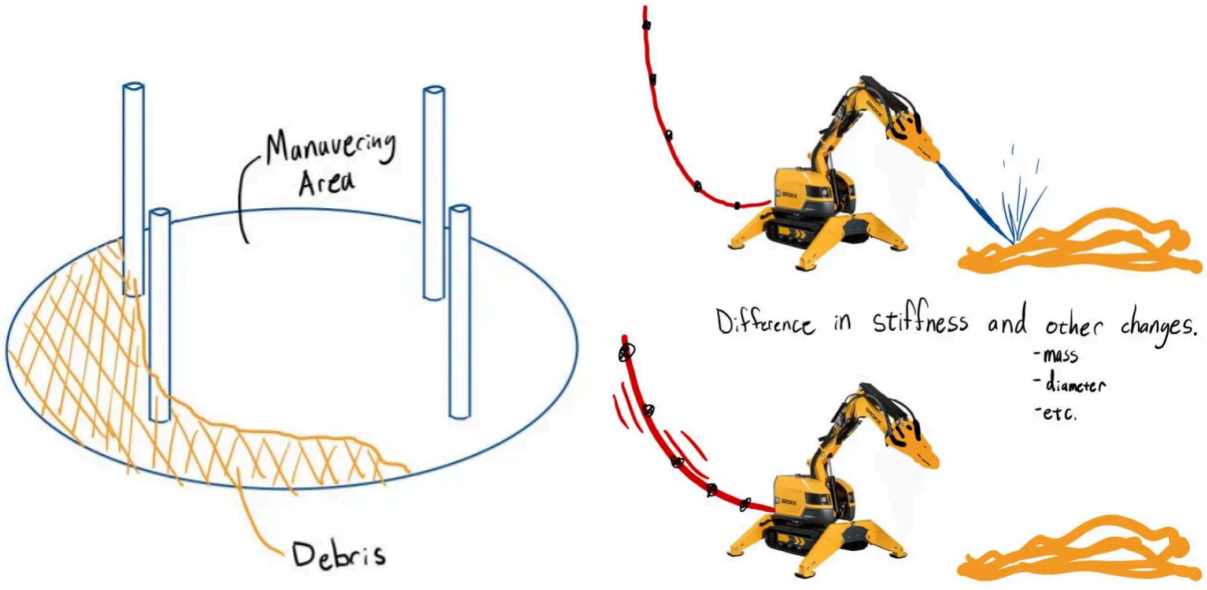


5.4 Prototypes

This project places a great importance on appeal and ease of access. We want this project to be good-looking and easy to use. To achieve this, we must create prototypes to demonstrate what our provided solution could look like. These prototypes aim to help us better understand what we can do and what our sponsor desires.

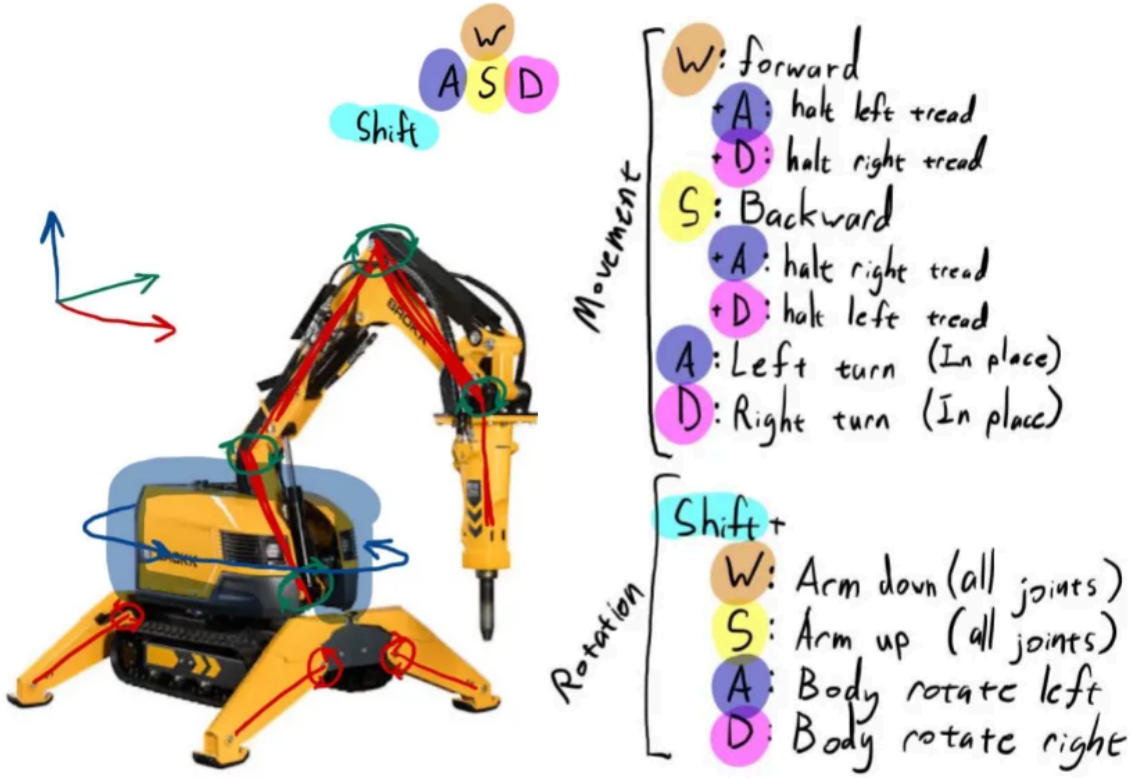


The image on the left demonstrates what we were requested to implement in the broadest terms possible. An updated model of the robot to be more in-line with practical, existing hardware plus a realistic cable that attaches to the back of the machine. The cable simulation should be robust enough to deal with situations as depicted above.

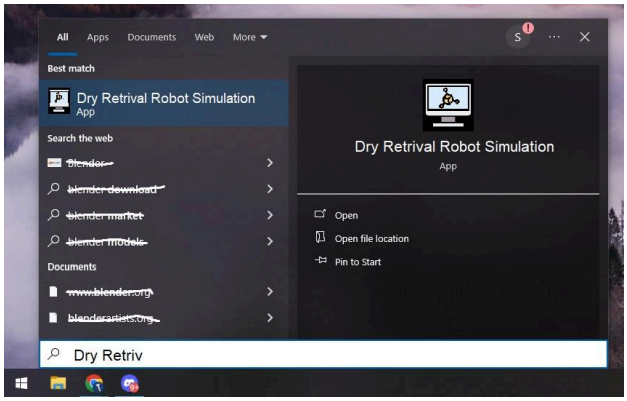


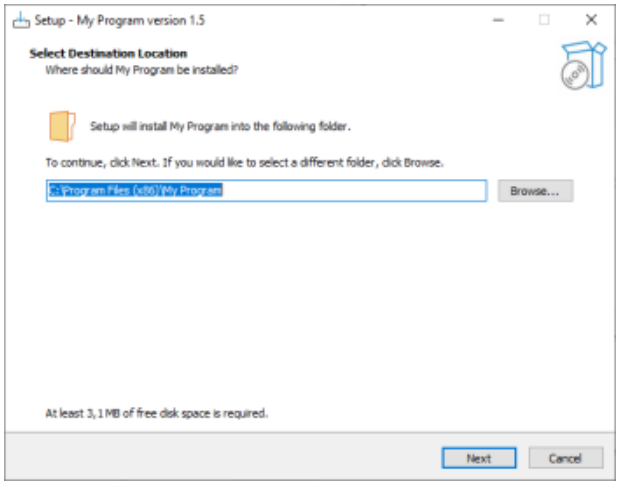
To properly test our implementation of a new simulation, we need more friendly environments for our robot to maneuver in. The left image depicts such a potential testing ground. The image

on the right shows a potential stretch goal where the properties of the cable can alter as the user operates the machine.



This example of a fully rigged Brokk robot. This model would require more advanced user controls. In the current version of the software, the controls are notably unintuitive. The control scheme proposed above handles the complexity of the machine in a way that is based on a First-Person-Shooter control scheme.





The two images above showcase what would be possible if we developed an installer for Windows. This is a common piece of software and should prove easy to develop if desirable. The setup wizard shown above was created with Inno Setup.

6. Design

6.1 Overview

Our design for the cable modelling simulation uses the Godot game engine to simulate realistic cable behaviour and physics. We hope to implement a dynamic cable model utilising a mass-spring system as a starting point, then switch to more complex models when necessary. The core of our approach involves integrating real-time (hopefully faster than real-time) physics calculations of tension, collision, and other properties to visualise a realistic cable.

Key Strategies:

- Cable Physics: Mass-spring model for basic behaviour with tension calculations, gravity, and frictional forces. More complex models like the Finite Element Analysis may be explored later on.
- Real-time simulation: Real-time data, hopefully, faster than real-time, to allow users to interact and view effects immediately.

Tools and Technologies:

- Godot Game Engine: Engine to render and simulate the physics.
- Python: Simulate physics outside Godot in a more neutral environment to compare execution time and give a baseline.

Equipment and Supplies:

- Computers with Godot : To develop and run the simulations.
- Test Environment: A simple 2d plot with grid lines.

6.2 Comparison of Potential Solutions

One of the biggest design problems our team must face is implementing the software to test and develop our cable model. This software must be useful for this purpose and probably allow us to edit model parameters on the fly. We must also develop models tested and evaluated for correctness. When considering what we were developing, we refer to the entire software as a whole, including models developed and tested within our cable environment.

Selection Criteria and Weightings

Speed, accuracy of prediction, and maintainability/readability are all important criteria to consider as we develop this project. The accuracy of our prediction is of the highest importance, hence the need for real-world evaluation techniques. Our project's maintainability is the final selection criteria.

	Speed	Accuracy	Maintainability	Weights
Speed	1	1/2	1/3	0.17
Accuracy	2	1	2	0.48
Maintainability	3	1/2	1	0.35

Solutions

One solution we have considered is using the Godot game engine as a physics environment. This solution would imply we could use its plentiful libraries, tools, and materials to develop dynamic rope models quickly and efficiently. Godot introduces tools that make user interface and user interactions trivial, which greatly increases the speed of development.

As an alternative to using Godot, we could implement our physics objects from a library, such as Bullet Physics or ProjectChrono, and render them without the aid of a game engine. This would give us control and intuition about exactly how the physics simulation is running. This granular control would probably be much more difficult, but we may see more accurate results. A dedicated physics library would probably prove to have more accuracy-oriented features compared with the Godot Physics engine. Using an alternative engine could give us FEA features that we can experiment with.

When considering what models of cable we should use, a simple mass-spring system could be our starting point, and then we can move on to more complex models later. This approach would allow us to try developing multiple models of cable in a compact and seamless environment. The cable models developed could be pursued and compared quickly.

Alternatively, a unique program could be developed for each model of cable or as our understanding evolves. This would allow us to develop quickly and explore ideas without worrying about generalization or integration. This would come at the cost of maintainability, where each iteration of the testing program would be different and inconsistent with the past implementations. Users may not be able to cycle between the simulations quickly with this approach or review the code effectively.

Decision

This matrix serves as a mathematical approach to deciding what methods should be used to solve our problem.

		Game Engine	Physics Engine	One program	Multiple programs
Speed	.17	.45	.55	.50	.50

Accuracy	.48	.40	.60	.55	.45
Maintainability	.35	.70	.30	.65	.35
Score		.51	.48	.57	.43

Based on these calculations, the best approach would be to use a game engine to develop a single program that houses iterations of cable models that can be tested and reviewed.

6.3 Data Design

Data Content:

- Cable Strain
- Cable Stress
- Time to complete
- Environmental Factors (Friction, Gravity)

Data Objects:

- A cable with n segments
- Time for compilation / display

Data Attributes:

- Cable data is saved externally into a file, with time, stress, and strain at each “t” time.
- Cable data is used to create a graph of stress and strain over time. Engineers can look over this data to determine when the cable is experiencing most strain and stress.
- Time data is saved / displayed in Godot / CSV

Data Flow:

1. Use of Godot and Implementation of Cables is used with different inputs (mass, length, etc.)
2. While these cables are being simulated, we display and save data on the cable and the time to compile/display.
3. This data is used to show the compile/display time difference between different cable models.

Data Repository

- Data is stored in CSV files, titled based on the physical properties of the cables when the program is run.

6.4 Functional: Structural Design

Level-0:

At the highest level, our system consists of the **Dry Retrieval Godot Cable Model**, which encapsulates all core functionalities, including cable physics, environmental interactions, user controls, and visualization. The system interacts with external user inputs and environmental parameters, driving the behavior of the simulated cable.

Level-1:

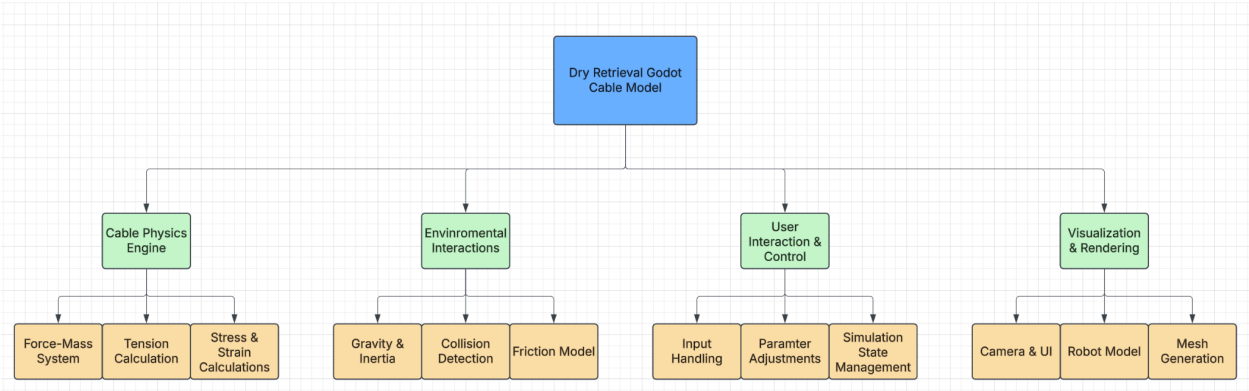
Expanding from Level-0, the primary system components for the Dry Retrieval Godot Cable Model decompose into distinct modules:

1. **Cable Physics Engine:** Implements the mathematical model governing cable movement, tension, and elasticity.
2. **Environmental Interactions:** Simulates external forces such as friction, gravity, and collision with objects.
3. **User Interaction & Control:** Allows users to manipulate cable properties and observe effects in real-time.
4. **Visualization & Rendering:** Manages 2D rendering of cable

Level-2:

Each Level-1 module further decomposes into its subcomponents:

1. Cable Physics Engine
    - **Force-Mass System:** Models cable as interconnected mass nodes with various forces applied.
    - **Tension Calculation:** Ensures proper stretching and compression dynamics.
    - **Stress & Strain Calculation:** Computes the internal forces within the cable by measuring stress and strain. This helps determine stretching limits, breaking points, and realistic elasticity.
2. Environmental Interactions
    - **Gravity & Inertia:** Applies realistic weight and motion.
    - **Collision Detection:** Prevents cable from passing through solid objects.
    - **Friction Model:** Implements resistance based on cable surface and environment.
3. User Interaction & Control
    - **Input Handling:** Detects user commands via keyboard/mouse.
    - **Parameter Adjustments:** Allows real-time tuning of cable properties.
    - **Simulation State Management:** Supports play, pause, and reset functions. Also includes the ability to run multiple simulations in a short amount of time.
4. Visualization & Rendering
    - **Mesh Generation:** Dynamically creates cable geometry.
    - **Robot Model:** Ensures accuracy with the real robot
    - **Camera & UI:** Provides intuitive perspectives and overlays.

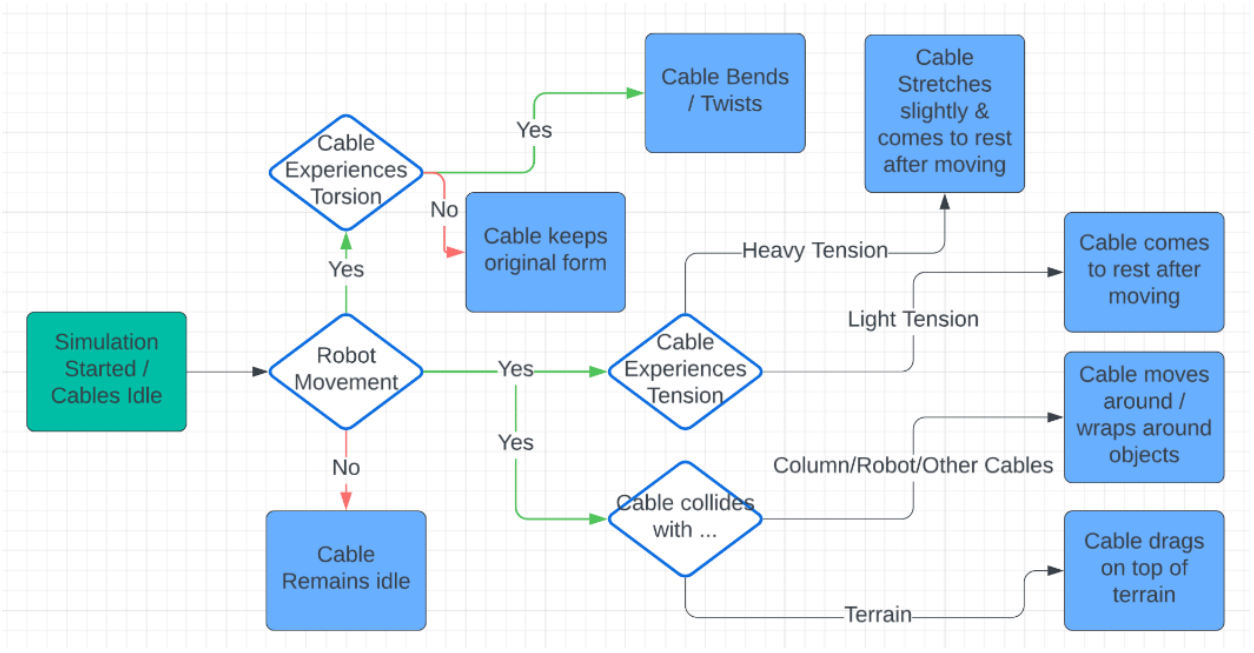


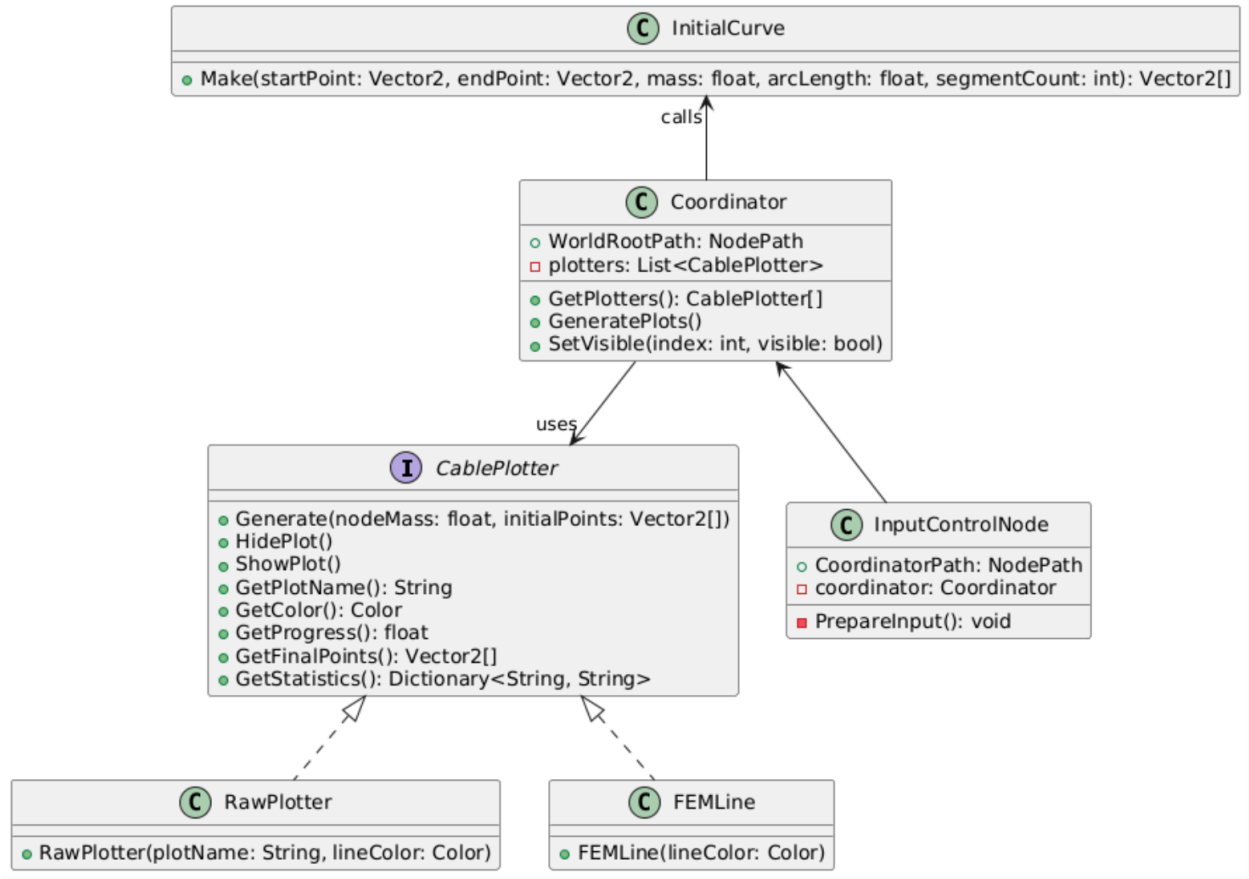
6.4.1 External Interfaces

Our simulation utilizes the following external APIs:

- Godot Physics Engine API: Provides built-in physics calculations for collision detection and rigid body dynamics.
- User Input API (Godot Input Class): Handles keyboard, mouse, and game controller inputs for interacting with the simulation.

6.5 Functional: Dynamic Design




InitialCurve
Make(startPoint: Vector2, endPoint: Vector2, mass: float, arcLength: float, segmentCount: int): Vector2[]
calls
Coordinator
WorldRootPath: NodePath
plotters: List<CablePlotter>
GetPlotters(): CablePlotter[]
GeneratePlots()
SetVisible(index: int, visible: bool)
uses
CablePlotter
Generate(nodeMass: float, initialPoints: Vector2[])
HidePlot()
ShowPlot()
GetPlotName(): String
GetColor(): Color
GetProgress(): float
GetFinalPoints(): Vector2[]
GetStatistics(): Dictionary<String, String>
InputControlNode
CoordinatorPath: NodePath
coordinator: Coordinator
PrepareInput(): void
RawPlotter
RawPlotter(plotName: String, lineColor: Color)
FEMLine
FEMLine(lineColor: Color)

7. Evaluation
7.1 Overall Evaluation Plan

Evaluation was completed throughout our system's development lifecycle, and it is detailed in this section. Testing, reviews, and real-life validation played a part in validating this system.

We must first consider our definition of success. Our users anticipated a software product that obeys a certain level of realism and reliability. This is our acceptance criteria and doneness definition. We need to ensure that our users are satisfied with our estimation of deformation, displacement, and tension.

We plan to achieve an estimate of tension and deformation that real engineers confirm are satisfactory. This needed to be evaluated using a survey. We presented the simulation results including images and tension data.

- Tests Performed
    - Real-world comparisons
    - Tension calculations
    - Complex environment demonstrations
- Metrics did you use to determine that your system is adequately tested
    - Likert scale evaluations by engineers.
- Strategies to automate testing
    - None

- **Interviews, Observations, and Surveys:** This involves gathering data after a user acceptance test to determine if your solution addressed the problem.

    - We surveyed engineers about the quality of our product.

    - They gave responses that can help us better understand.

7.1.1 Quality Plan

We now describe our quality goals and how our evaluation process ensures quality goals are met.

1. Quality Goals

- 0 errors under normal use
- 1 small bug per 200 lines of code.
- Realistic physics behavior

2. Quality Policies

These policies are rules that team members followed to meet the quality goals above.

Example policies for the quality goal

2.1. Coding standard shall be used (C#)

2.2. Code shall be peer-reviewed (On Github)

2.3. Bug-tracking on GitHub

2.4 Shared theoretical understanding

2.5 Zero escaped errors to the customer

3. Processes (with Artifacts)

Processes for the quality goal “0 errors under normal use” and “Realistic physics behavior”

3.1 Requirements Review; Artifacts: Review Process & Checklist

3.2 Design Review; Artifacts: Review Process & Checklist

3.3 Code Review; Artifacts: Review Checklist, Coding standard

3.4 User Acceptance Test; Artifact: UAT Result Document, Surveys

Requirements Review Process

1. After the requirements review has been completed and peer-reviewed, a review with the entire team was to be scheduled.

2. The reviewers are given at least 3 days to review the document.

3. At the review, the team identified strengths and areas for improvement, which the author incorporated into the document.

4. A team member is assigned to review that the changes have been made as discussed at the review, after which the requirements were to be baselined.

Analysis and Reporting Process

1. Group update captures how the team is performing on quality goals. For example, if the quality goal is "0 escaped errors," the actual # of escaped errors are gathered and analyzed

2. If the shortcoming is concerning enough, we meet and discuss a remedy.

3. These concerns and discussions are shared with our customer.

7.2 Internal Evaluation Plan

7.2.1 Testing Setup Procedure

In order to internally test our system the tester must load the individual cable simulation component into Godot game engine. Within the game engine there should be the 2D cable and properties that are changeable to visualise the physics. The evaluation should last approximately between 10 to 20 minutes. The tester through their laptop should be able to control the individual cable simulation within Godot for functional interaction and reliability of cable physics simulations.

Within the internal testing environment we should be able to test individual components such as cable sag point, length, weight, and other variables. After individual component testing the tester could move onto integration testing. Initially within our integration testing our cable was static. Next, the cable's movements can be adjusted in multiple ways through the exposed variables. The cable physics properties are tested with the adjustments. Cable tension data increases with pulling forces and the cable positions are to be compared to real life or other cable positioning in response to pulling forces. The cable individually could be placed under tension to visualise the cable tension data and ensure that it is accurate.

The 2D cables size and length could be altered to simulate different cables and scenarios

Overall, cable movement and reaction should replicate accurate real life behaviors that do not show any signs of abnormalities like cable jitter, excessive movement, and unexpected cable positioning or collision behaviors.

7.3 User Acceptance Test Plan

7.3.1 Recruitment of Users

For our project, we attained verbal approval from our sponsor indicating that he and a few others would be our users for the user acceptance tests. This project is best described as a proof of concept, so David and his selected group were our users for the user acceptance testing.

David Mascarenas - Main point of contact

- The purpose of David Mascarenas testing our project was to ensure that our simulation was running at a good speed and up to his standard of realism
    - We used a survey after demonstrating and giving him the project. This provided quantitative data on his acceptance of the project's realism and if it is up to standard.

Los Alamos National Laboratory Engineers

- David Mascarenas gave our project to a few other engineers at Los Alamos National Laboratory. These Engineers took our survey, and we needed to ensure that the project is up to standard for these engineers as well.
- Piper, Jacob Martin jpiper@lanl.gov
- Singh, Upendra upendra@lanl.gov
- Green, Andre Walter andre_green@lanl.gov
- Thanaravisara, Thanatat thanatat@lanl.gov
- Ramon, Rodrigo rodrigo.ramon@srnl.doe.gov
- Tomlin, Michael Michael.Tomlin@srnl.doe.gov
- Gunnard, Anthony aglor@lanl.gov

Anyone not part of David Mascarenas group and Los Alamos National Laboratory was excluded from our User Acceptance Testing and recruitment, as this project is simply a proof of concept for these select people.

7.3.2 User Acceptance Test Artifacts

7.3.2.1 Protocol

Our user testing involves a small group of five participants from our target demographic - engineers or technically proficient users familiar with cable systems. Each participant interacts with our simulation individually in a freeform session.

The testing session was self conducted where the user uses the app on their own. They were asked to define start and end points, adjust the cable length and segment count, and set a mass value. They would then explore enabling/disabling plots using the visibility checkboxes and interacting with the UI (e.g., panning, zooming, and interpreting outputs). Participants then completed a short survey afterward.

The survey uses a 5-point Likert scale and open-ended prompts to measure satisfaction, realism, and ease of use across six categories: Easy to Configure, Realistic Displacement, Camera Control, Useful Results, Engineering Utility, and Parameter Count.

During testing, we were interested in user confusion and frustration, and considerations on time to complete testing. Users' names and emails are collected. Feedback is stored and used to inform future improvements.

7.3.2.2 Survey

A 10-15 question user functionality survey conducted using Google Forms. Questions include:

- **Realism of Implementation Questions (On a scale of 1 to 5):**
    - These are going to be 5+ questions where we asked the user to compare a certain scenario simulated in the system to the real-life scenario equivalent. *(1 = Unrealistic, 5 = Very Realistic)*
- **General Likert Scale Questions (On a scale of 1 to 5):**
    - How easy was it to navigate the system? *(1 = Very Difficult, 5 = Very Easy)*
    - How intuitive was the user interface? *(1 = Not Intuitive, 5 = Very Intuitive)*
    - Did you experience any delays or performance issues while using the system? *(1 = Always, 5 = Never)*
- **Open-Ended Questions:**
    - What challenges or difficulties did you encounter while using the system?
    - Were there any missing features or functionalities that you expected? If so, please describe.
    - How would you improve the system for better usability?
    - Do you have any additional comments or feedback?

8. Implementation

We implemented this project through the use of Godot's C#. By comparing the computation speed between the cable implementation in Godot's C# output, and a baseline FEM model written in Python which was used as a template, we can deduce which model strikes a good balance between accurate and realistic physics, as well as an overall time effective solution.

Architecture Overview

Coordinator: This central coordinator object was responsible for the routing of input through the program's different components, primarily from the UI to the world space where cables are plotted.

CablePlotter Interface: This interface describes the shared traits of each line we aim to plot, visualize, and update..

World Viewport: Where the object responsible for drawing the cable paths is placed.

Camera Controller: The camera was added so that a user could view a cable regardless of its size.

UI and User Controls Panel: A complex container and object structure was erected to contain and define the basic user interface for this project. User controls are programmed to function correctly by a single object that is responsible for preparing them. The control panel contains the following programming:

- Start and Endpoints
- Segments
- Mass
- Length
- Plots Visibility Checkboxes
- Function buttons

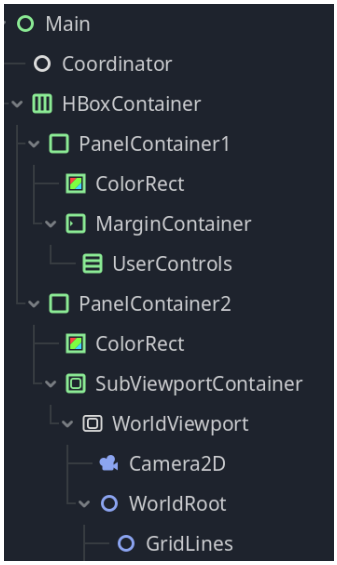


Figure 1: Screenshot of actual architecture used in the Godot visual interface.

Tools Used

Godot

- Godot is used to visualize the cables easily and gives us a nice interface that allows us to code directly in C#.

C#

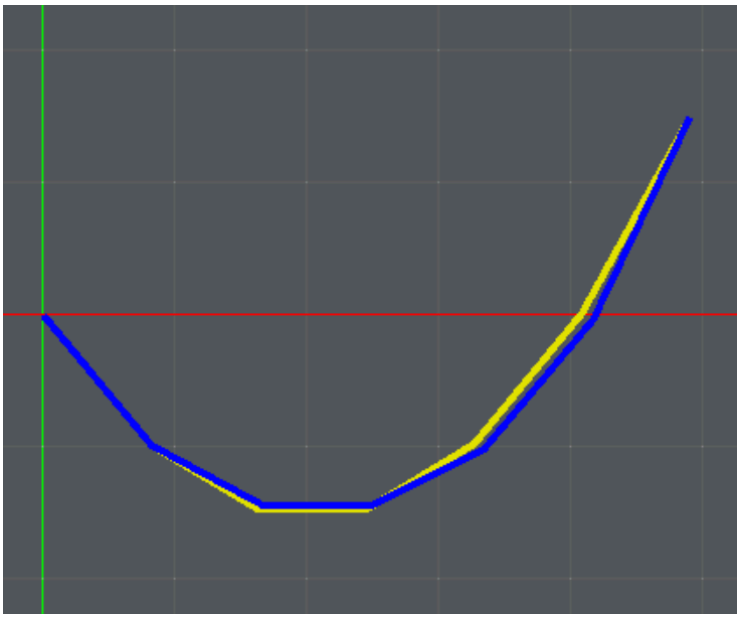
- C# is one coding language of Godot. We are able to code directly in Godot using C#, and our ideal outcome is a physics model that runs quickly in this language.

Python

- Python is used as our baseline coding language that we compared our C# output and time to render/display to. Python was used to code an initial FEM model during development.

Data Representation

All cable geometry and space are measured in metric units. After all calculations are complete, final cable positions are rescaled into world-space according to the coordinator's static functions for determining this. This organization ensures consistent visuals for users. Initial distance units are meters, and mass units are kilograms.



Visual Interpretability

By implementing a controllable camera, users get a pleasant and helpful way of viewing the final data positions. Axes are drawn to give some spatial background to the information presented. The user interface's values match that which is drawn.

Project Management

Team coordination was primarily conducted via GitHub and Discord. The project was developed in parallel, with one member focused on architecture, one on final FEM physics, and one on alternative model rendering, and one thinking about group documentation. Responsibility sharing became uneven during the project; as a result, the team pivoted to a structure where ownership over modules was more clearly defined. Work done on modules was only done with the

permission of the module owner. Development done in these sections was overseen by the section owner. Version control and management.

FEMLine: Mustafa Tekin
Coordinator, UI, CablePlotter: Thomas Holt
MassSpringLine: Jonathan Zhao

9. Results

Baseline and Motivation

There is a lack of cable simulation that balances realism and computational efficiency. Our goal is to develop a proof of concept simulation that maintains real-time performance while ensuring realistic cable dynamics. Our goal was to implement a system that was visually intuitive to engineers and helps develop an understanding of the algorithms behind predicting cable behavior. We wanted engineers to be able to reflect on the underlying physics of the system.

Evaluation Plan and Implementation

The evaluation of our project consisted of internal testing and user acceptance testing (UAT). Internal testing focused on verifying controllability and observability through unit and integration tests, ensuring that the cable physics behaved as expected under different conditions. The UAT was conducted with engineers at Los Alamos National Laboratory (LANL), led by David Mascarenas. Each participant interacted with the simulation and provided feedback via a survey. The survey collected Likert scale responses and open-ended feedback questions on realism, usability, and performance.

Results of Internal Testing

- **Controllability:** Users can directly set start and end points, cable length, mass, and resolution (segment count). Input changes are reflected in the visual output, confirming end-to-end data flow.
- **Observability:** Users could see the cable and the physics properties it demonstrates. Some features like time analysis and precise position are missing.
- **Performance:** Some staggering when loading FEM plot. No critical slowdowns or freezes. 60 Frames per second is able to be maintained throughout execution.
- **Physical Correctness**: While not being benchmarked against a commercial solver or real-world measurements, this solver matches intuitive expectations. This is a critical component of future work.
- **Stiffness assembly:** stiffness matrix assembly process yields values matching that of our given

Metric	Mass-Spring Cable	FEM Cable	Notes
Nodes	20	20	Consistent node count for fair comparison

Total Runtime	3.21 ms	97.48 ms	FEM significantly slower due to matrix ops
Simulation Threads	1 (main thread)	1 (isolated thread)	Both implementations single-threaded for consistency
Thread-Safe Progress Updates	Yes	Yes	Verified via locks
Max Displacement	0.083 m	0.081 m	Minor deviation from tension/weight integration
Final Position Deviation	—	0.004 m (avg)	Relative to mass-spring, in world space
Total Internal Force	0 N	0 N	Slightly higher in FEM due to elastic stiffness model
Converged Increments (FEM)	—	200	Within threshold in <15 increments
Statistics Written to File	yes	yes	csv

Results of User Acceptance Testing:

Survey responses from five engineers at LANL provided insights into the effectiveness of the simulation. Here is a quantitative analysis of what users said about our project.

Question	Scores (by participant)
Useful Results	2, 4, 4, 5, 5
Camera Control	3, 5, 5, 5, 5
Engineering Utility	3, 3, 4, 4, 5
Realistic Displacement	2, 4, 4, 5, 5
Easy to Configure	2, 4, 4, 5, 5
Parameter Count	3, 4, 4, 5, 5

Table 1: Individual responses from users.

While user satisfaction seems glowing in the Likert scale analysis, we also should consider the notes left under each Likert prompt. Some of these comments affect real-world usability.

The chart on the right (Figure 2) contains the summary of the responses given for each question topic. The actual questions are available in the Annex 7 test case documentation. Table 1 details the individual question response values.

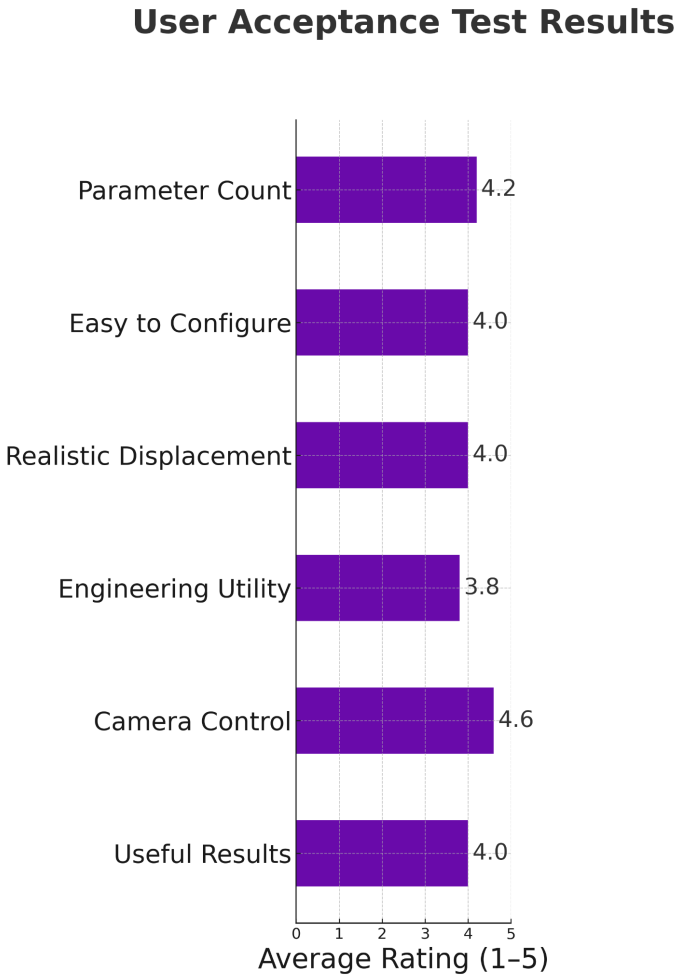


Figure 2: Likert scale responses.

Analysis of Feedback

According to the reviews and feedback left by our reviewers:

Strengths:

- Users reported that camera controls felt natural and intuitive.
- Clear visual distinction between undeformed and deformed cable plots.
- Interface allowed exploration of key parameters like mass, length, and segments.
- Even reviewers intending to integrate this into their own workflows acknowledged the potential value and time-saving nature of the simulation.

Areas for Improvement:

- **Error Handling:** When invalid parameters are entered (e.g., a cable length shorter than the straight-line distance between points), the simulation can silently fail or crash. Users requested clear pop-up warnings or prevention.
- **Visual Feedback:** Lack of unit labeling and axis indicators was frequently cited. Several users mentioned that basic labeling would move the program from "educational" to "engineer-usable."
- **Interface Stability:** Several testers encountered bugs when adjusting parameters dynamically, especially with mass and length. Some suggested the system needs stronger protections against invalid input combinations.
- **Performance:** Simulations with higher segment counts ($n > 20$) slowed down considerably, although some users noted this might partly be a hardware issue.
- **Feature Suggestions:**
    - Allow multiple waypoints (not just two anchors).
    - Visual warnings for self-intersections, kinks, or overstretched cables.
    - Provide direct output of results like cable tension and sag into the UI (instead of only printing to console).
    - Add mouse-based setting of start/end points for better UX.

Outcome and Achieved Success:
Results demonstrate that our project successfully achieved its definition of success as a proof of concept.

Outcome & Definition of Success

Our goal was to demonstrate a proof-of-concept cable simulation tool that supports both visual and computational inspection of cable behavior. We defined success as:

- Fast interactivity
- Multiple plotters with physics-informed output
- Parameter control via GUI
- Clear visuals and simulation feedback

Despite the issues outlined above, our project clearly succeeded as a proof of concept. Users could observe realistic cable deformation, apply custom parameters, and extract useful information from the simulation. In particular, the software provided good intuition about the relationship between cable mass, length, and sag — fulfilling our primary goal.

However, there is no denying that the system is fragile. If this project were to move toward production or daily engineering use, the focus would need to shift to hardening the input validation, expanding output clarity, and eliminating failure cases through better error reporting and user guidance.

Conditions & Reproducibility

Anyone can replicate the results of this project by:

- Using the publicly available codebase in Godot 4 on GitHub
- Running the simulation and adjusting parameters demonstrate the deterministic system we have implemented.
- Observing the visual and simulated output as parameters change

All code was written from scratch, and our core FEM algorithms are implemented and traceable for reproducibility. While it isn't the cleanest, it is labelled and commented for clarity.

10. Discussion

The results of this project were far outside of what our sponsor imagined initially. We underwent many iterations of design and scope. Our methodology consisted of many stages of learning and revising.

Methodology

While our simulation produced clear and visually intuitive results, some portions of the system would have benefited from team coordination. Our finite element method converges on a solution in a reasonable amount of time under regular conditions. Still, we did not rigorously test stability or scalability beyond 2D, which was a core goal.

Evaluation Gaps

We evaluated most visual and usability outcomes but did not measure physical accuracy against ground truth data. If repeated, we would build a stronger testing harness and include more physics benchmarking, including comparisons to commercial and known reliable solvers.

Safety and Failure Modes

This project should not be considered as a practical tool for people who need engineering analysis. It is too simple and inextensible. There are hidden model parameters that we failed to understand or integrate into our work. If this work is used for anything other than a discussion on the complexity of the algorithms involved, it may not prove to be a safe tool.

Team Reflection

When reflecting on our team, we were able to experience collaborative and productive development. While we did suffer from scope creep and coordination, we have all found this to be a valuable learning experience. In hindsight, we should have taken those early stages of planning and development more rigorously to gain a better understanding of the long-term plans

of our project. We should've taken our project schedule more seriously. As it stands, we fell behind on some tasks that should've been in the forefront of our consciousness.

Ethical Considerations

If adopted broadly, visual tools like this could be misunderstood as being “engineer-approved.” Care must be taken to label limitations, especially when used in high-risk fields like construction or nuclear cleanup. This tool is best referred to as an academic exercise. The licensing of Godot and .NET tools was respected, and we built all simulation logic in-house. No intellectual property rights were violated in the making (or potential using) of this project.

11. Future Work

The successful simulation of cable physics opens up several promising opportunities, both for improving the current system and for expanding the simulation to cover additional aspects. Some of the immediate opportunities include:

Real-Time Physics Adjustments:

Our current project serves as a foundation for real-time physics analysis, but there is potential for refining the system to account for more nuanced factors. For instance, cable elasticity, material properties, and environmental variables like wind or temperature could be incorporated for more accurate and dynamic simulations.

Interactive Visualization:

This simulation could serve as an even more interactive tool for engineers involved in the dry retrieval process. By adding a user interface that allows for real-time interaction with the cable physics, users could experiment with different scenarios quickly and gain a deeper understanding of the complexities of the physics and simulation.

Despite the successes in the project, several challenges remain that could not be fully addressed within the project timeline:

Greater Cable Complexities:

The interactions between the cable and the physics, especially when under specific parameters, were simplified in the current simulation. The general scope of the project had to be shifted from 3D to 2D, as that added complexity exemplified the many problems. This is an area where more advanced algorithms, possibly involving more intricate systems or more efficient algorithms, would be required to model realistic cable dynamics.

Limited Environmental Modeling:

While the simulation accounts for basic environmental forces, more complex environmental factors were not included. These factors would be essential for creating a more comprehensive and realistic model, particularly for real-world applications where such forces can significantly affect cable behavior.

Performance Optimization:

The simulation's performance could be optimized for larger-scale systems with more complex interactions. As the complexity of the physics increases, the computational resources required for the simulation might also increase, posing challenges in terms of processing speed and memory usage.

12. Conclusion

In conclusion, the Godot cable project addresses the importance of cable simulation efforts to enhance the understanding of possible issues beforehand to operational execution like the Dry Retrieval process, which is closely related to the efforts of this project. Issues like cable management and limitation factors of a robot attached to a cable within hazardous environments can create unforeseen issues and roadblocks on the road to accomplishing operational success in the real world. By utilizing the Godot game engine for incorporating both mass-spring systems and the Finite Element Method, this project aimed to create a scalable, accurate, and computationally efficient cable simulation.

The solution this project has worked on is improvements to cable model behavior from a realistic and programmatic sense. This application helps researchers/engineers understand how the realism and computation of the cable model physics can vary with some pros and cons for different implementations. The results from our evaluation—based on the accuracy of physics, responsiveness, and observability—inform future adjustments and potentially contribute to broader applications in robotics, hazardous material handling, and industrial automation. Overall, this project hopes to play a valuable role in improving the reliability and safety of nuclear waste retrieval efforts through robust future simulation tools and other cable-related software challenges.

13. References

Berry, J., Patel, V., & Vasudevan, K. (2011). The Hanford waste feed delivery operational research model. Proceedings of the 2011 Winter Simulation Conference (WSC), 2143–2153. https://doi.org/10.1109/WSC.2011.6147927

Boeing, A., & Bräunl, T. (2007). Evaluation of real-time physics simulation systems. Proceedings of the 5th International Conference on Computer Graphics and Interactive Techniques in Australia and Southeast Asia - GRAPHITE '07. https://doi.org/10.1145/1321261.1321312

Dai, C., Wu, Y., Zhang, Y., Wu, K., & Xu, A. (2020). Cabling simulation for Nb3Sn Rutherford cable. 2020 IEEE International Conference on Applied Superconductivity and Electromagnetic Devices (ASEMD), 1–2. https://doi.org/10.1109/ASEMD49065.2020.9276081

Eppel, T., & von Winterfeldt, D. (2008). Value-of-Information Analysis for Nuclear Waste Storage Tanks. Decision Analysis, 5(3), 157–167. https://doi.org/10.1287/deca.1080.0121

Gallucci, M. (2021, October 8). A glass nightmare: Cleaning up the Cold War's nuclear legacy at Hanford. IEEE Spectrum. https://spectrum.ieee.org/hanford-nuclear-site

Gephart, R. E., & Lundgren, R. E. (1995). Hanford Tank Clean up: A Guide to Understanding the Technical Issues. https://doi.org/10.2172/195769

Hileman, B. (1982). Nuclear waste disposal. Environmental Science & Technology, 16(5), 271A–275A. https://doi.org/10.1021/es00099a721

Hu, W., Qu, Z., & Zhang, X. (2012). A new approach of mechanics simulation based on game engine. Proceedings of the 2012 Fifth International Joint Conference on Computational Sciences and Optimization, 619–622. https://doi.org/10.1109/CSO.2012.141

Jomartov, A., Tuleshov, A., Kamal, A., & Abduraimov, A. (2023). Simulation of suspended cable-driven parallel robot on SimulationX. International Journal of Advanced Robotic Systems, 20(2), 172988062311614. https://doi.org/10.1177/17298806231161463

Kim, H., & Kim, J. (2023). Prediction of cable behavior using finite element analysis results for flexible cables. Sensors, 23(12). https://doi.org/10.3390/s23125707

Liu, T., Bargteil, A. W., O'Brien, J. F., & Kavan, L. (2013). Fast simulation of mass-spring systems. ACM Transactions on Graphics, 32(6), 1–7. https://doi.org/10.1145/2508363.2508406

Lu, H., Yijin, W., & Hu, Y. (2012). Design and implementation of three-dimensional game engine. Proceedings of the World Automation Congress 2012, 1–4. https://ieeexplore.ieee.org/document/6274802

Lv, N., Liu, J., Xia, H., Ma, J., & Yang, X. (2020). A review of techniques for modeling flexible cables. Computer-Aided Design, 122, 102826. https://doi.org/10.1016/j.cad.2020.102826

Mou, F., Wang, B., & Wu, D. (2022, April 11). Learning-based cable coupling effect modeling for robotic manipulation of heavy industrial cables. Nature News. https://www.nature.com/articles/s41598-022-09643-6

Phuoc Tho, T., & Truong Thinh, N. (2023). Evaluating cable tension distributions of CDPR for Virtual Reality Motion Simulator. Mechanics Based Design of Structures and Machines, 52(8), 5817–5835. https://doi.org/10.1080/15397734.2023.2265452

Salmela, T. (2022). Game development using the open-source Godot Game Engine. Theseus. https://www.theseus.fi/handle/10024/746943

Schroeder, D. (2022). Physics Simulations in Python: A Lab Manual. https://physics.weber.edu/schroeder/scicomp/PythonManual.pdf

Subbaiyan, P. B., Nizampatnam, B., Redkar, D., Sathusundarsingh, A., & Muniappan, B. (2023). Mechanical control cable modeling and simulation to predict the load loss and deformation. SAE Technical Paper Series. https://doi.org/10.4271/2023-28-0168

Wang, J., Lewis, M., & Gennari, J. (2003). A game engine-based simulation of the NIST urban search and rescue arenas. Proceedings of the 2003 Winter Simulation Conference, 1, 1039–1045. https://doi.org/10.1109/WSC.2003.1261528

Wang, Q., Palta, E., & Fang, H. (2024). Numerical modeling and simulation of cable barriers under vehicular impacts on a sloped median. International Journal of Protective Structures, 15(4), 891–916. https://doi.org/10.1177/20414196241226725

Warnitchai, P., Fujino, Y., & Susumpow, T. (1995). A non-linear dynamic model for cables and its application to a cable-structure system. Journal of Sound and Vibration, 187(4), 695–712. https://doi.org/10.1006/jsvi.1995.0553

Fried, I. (2001). A bendable finite element for the analysis of flexible cable structures. Computational Mechanics, 28(6), 496–505. https://doi.org/10.1007/s004660000206

Sammarchi, E. (2018). Dynamic modelling and simulation of a cable-driven parallel robot (Master's thesis, University of Bologna). https://amslaurea.unibo.it/id/eprint/17526/1/sammarchi_enrico_tesi.pdf

Mou, F., Wang, B., & Wu, D. (2022). Learning-based cable coupling effect modeling for robotic manipulation of heavy industrial cables. Scientific Reports, 12(1), 1–12. https://doi.org/10.1038/s41598-022-09643-6

Lv, N., Liu, J., Xia, H., Ma, J., & Yang, X. (2020). A review of techniques for modeling flexible cables. Computer-Aided Design, 122, 102826. https://doi.org/10.1016/j.cad.2020.102826

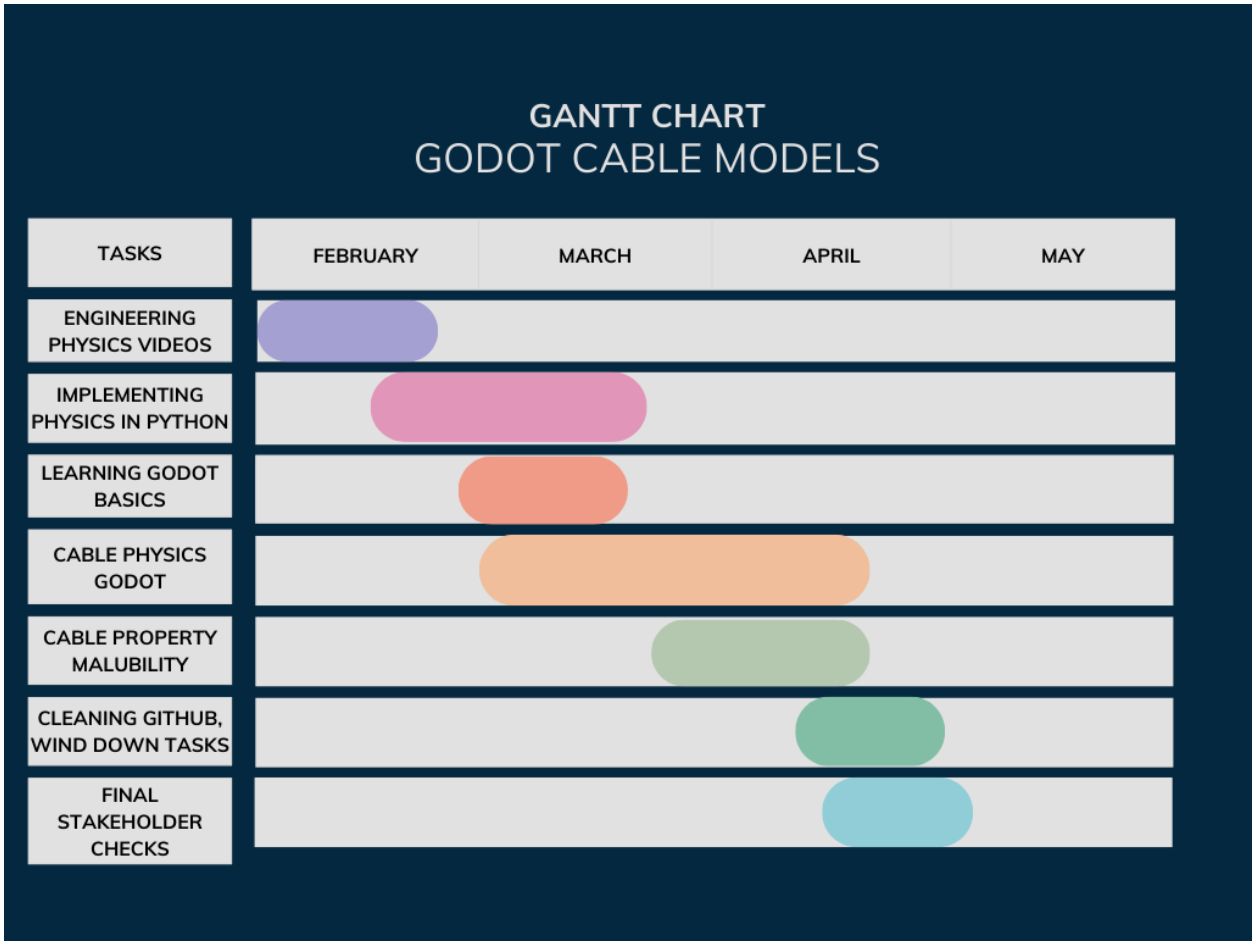
Annex 1: Project Plan

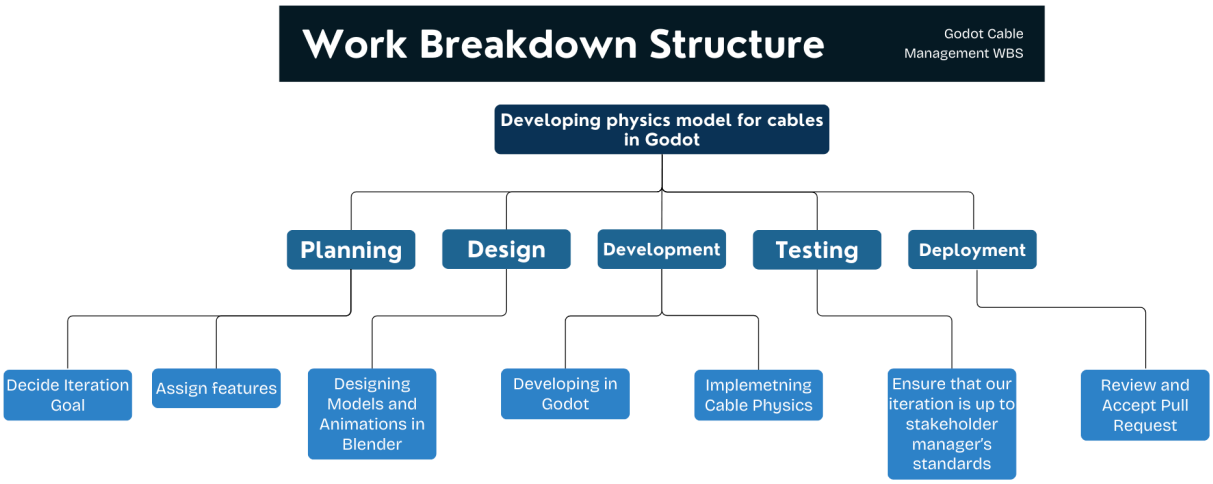
We plan on using the Spiral model. It uses repeated stages that are revisited as the project nears completion. It should prove to be flexible at the beginning of the project and grow more rigid as the project requirements are defined.

A1.1 Implementation Schedule

Because we are using a spiral model, we don't have sprints, but rather "checkpoints" or "iterations." This will lead to an iterative process that will reach regular checkpoints as we work. We will need to re-evaluate our work at these checkpoints.

We will begin the first 2-3 weeks by going through the requested education videos: The Direct Stiffness Method for Truss Analysis with Python and Non-linear Finite Element Analysis of 2D Catenary & Cable Structures using Python.





A1.2 Division of Labor and Responsibilities

Deliverables:

Deliverable	**Assigned to**	**Due Date**
Watch and learn from required physics videos	All	February 20th
Implement Catenary model	Thomas Holt and Mustafa Tekin	Feb 13th
Add Basic Starting Rope	Thomas Holt	Feb 20th
Implement UI and 2D workspace	Mustafa Tekin	Feb 20th
Integrate Starting components through UI	Thomas Holt	Feb 27
Node forces application	Jordan Daryanani	Feb 27
Implement code timers	Jordan Daryanani and	March 13

	John Mo	
Gridlines	Jordan Daryanani	March 13
Statistics panel with stats from the execution.	Jonathan Zhao	April 15th
Add model particular parameter exposure.	John Mo	April 15th
Apply node forces in UI	John Mo	April 20th
Add working code to GitHub Repo	Thomas Holt	April 27th
Poster and slide presentations	All	April 10th
Standalone Godot App	Thomas Holt	End of April
Final Report	All	End of April

For this project, we need to get access to the engineering physics videos online. For this, we created a shared Gmail account and paid $125 for full access to the engineering videos. We will go through both videos individually.

For the deliverables of this project, we will need Godot and Discord. Both are free. Godot will be used to implement the physics simulation and Discord will be used as our main form of communication between team members.

Role	Name	Email
Scope Coordinator	Jonathan Zhao	jonz64@tamu.edu
Schedule Coordinator	Jordan Daryanani	jordandary@tamu.edu
Stakeholder Management / Communication Coordinator	John Mo	johnmo@tamu.edu
Risk Coordinator	Mustafa Tekin	mt37863@tamu.edu
Quality Coordinator	Thomas Holt	tmholt02@tamu.edu

While all team members are considered developers, we do plan on using specific roles for the management of some high-level ideas. We are using the recommended roles enumerated above. These roles will allow us to stay on top of our project and its progress.

Quality Plan:

1. Weekly quality discussion
2. Address accessibility concerns

A1.3 Budget Costs

A1.3.1 In Class Capstone Project Budget

Laptop - Used to develop in Godot - 5 Units * $1000 each = $5000

Engineering Skills Subscription - Watch the required videos to learn how to implement cable physics into Godot - 1 Unit * $125 = $125

Godot - Create an interactable world with physics - Free, 5 Units

Discord - Communication between team members - Free, 5 Units

Zoom - Professional communication with stakeholders - Free, 5 Units

Hours/Wage: Student hours are free - N/A

Maintenance Cost: N/A

Total Cost: $5125 total

A1.3.2 Outside Company Budget/Cost

Laptop - Used to develop in Godot - 5 Units * $1000 = $5000

Engineering Skills Subscription - Watch the required videos to learn how to implement cable physics into Godot - 5 Units * $125 = $625

Godot - Create an interactable world with physics - Free, 5 Units

Discord - Communication between team members - Free, 5 Units

Zoom - Professional communication with stakeholders - Free, 5 Units

Hours/Wage: [100hrs] * [$50/hr] = $5,000

Maintenance Cost: N/A

Total Cost: $10,625

Annex 2: Project Management Artifacts

Annex 2.1: Stakeholder Management Plan

Stakeholder management and communication plan							
Stakeholder Name	**Category**	**Levels of power and interest (according to stakeholder grid)**	**Strategies for gaining support / reducing obstacles**	**Information needed / Document Name**	**Document Format / Medium (e.g., hardcopy, email, chat, etc.)**	**How Often / When Due**	**Status (e.g., stakeholder issues, status - happy or not, etc.)**
David Mascarenas	Customer	High Power and High Interest	Weekly updates on progress	Update Slides	Slides via Email	Weekly	Happy
Pauline Wade	Instructor	Low Power and High Interest	Showing up to class and interacting				Happy
Yamini Kamisetty	TA	Low Power and High Interest	Saying hi when she shows up for our weekly, saying hi in class				Happy
	Project Team	High Power and High Interest	Team meetings and being friendly. Frequent check ins on discord and group work sessions.				Happy

	Other high interest/high power						

Annex 2.2: Risk Management Plan

Risk	Impact	Likelihood	Mitigation Plan	Monitoring Plan	Management Plan
Not producing a fully working and tested system as expected	High	Low	Break the 2D transition into smaller milestones; continue testing each 2D component for scalability	Weekly progress reviews and Group check in.	Focus final sprint on integration/testing; If needed shift scope to strongest deliverables
Delay in transitioning from Python to C# modeling	High	Medium	Get started on a C# solution referencing an existing Python implementation	Weekly progress reviews and Group check in.	If delays persist, prioritize the cable geometry as a catenary, over FEM
Integration issues into Godot C# code	Medium	Low	Early review of the existing Godot files and documentation; Watch videos and understand how Godot works	Test Godot C# code continuously to ensure it works well.	Refer to other existing Godot code to understand structure and features within the engine

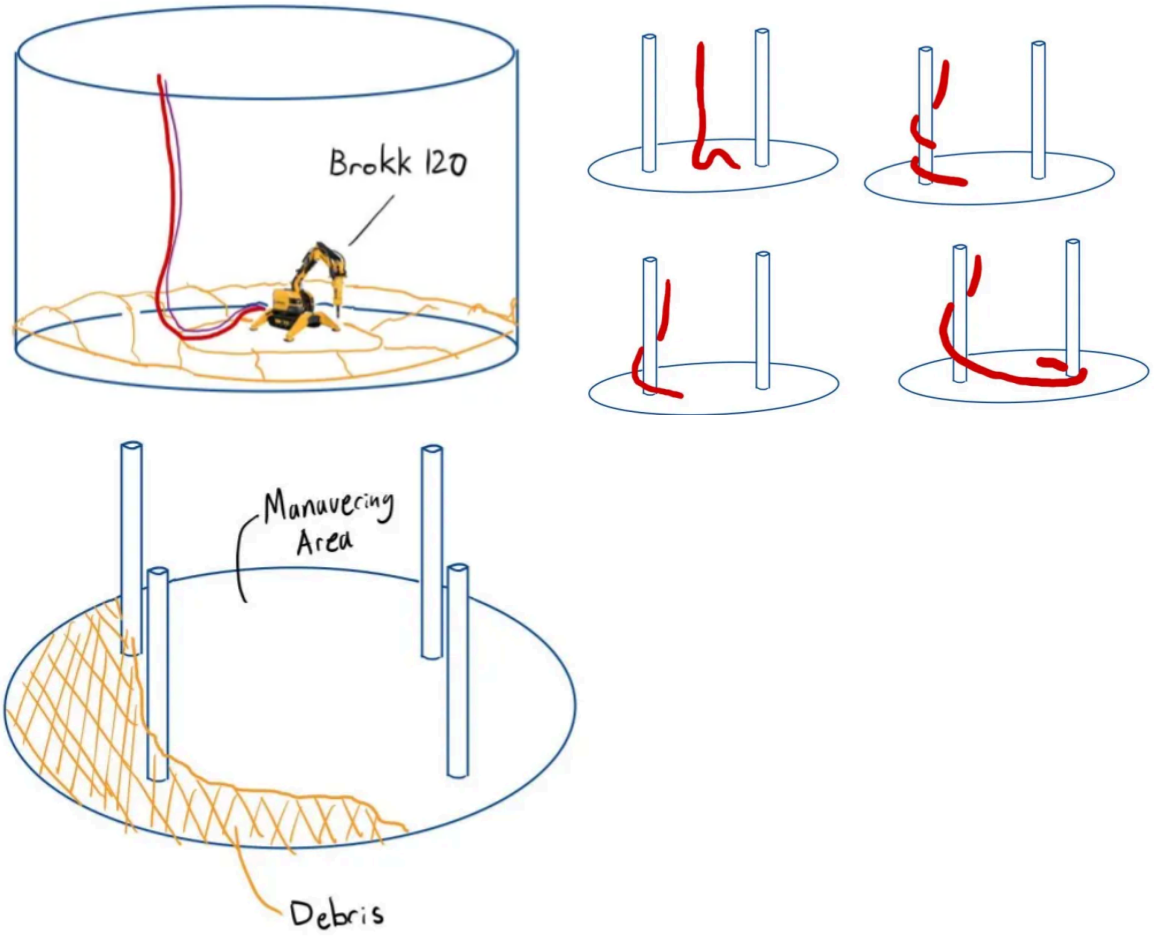
Annex 3: User Stories / Usage Scenarios

1. As an engineer,
   I want the cables to be modular
   So that I can simulate different cables
2. As an engineer,
   I want the cables to have realistic physical properties
   So that I can test the impact of the cables on maneuverability
3. As an engineer,
   I want to be able to dynamically change physical properties of the cables
   So that I can test the different physical implications of materials
4. As an engineer,
   I want the cables to have collision detection
   So that I can decide whether or not the cables will be realistic in a real-world
5. As an engineer,
   I want the cable to collide with itself
   So that I can see whether or not entanglements will occur
6. As an engineer,
   I want to control the robot's movement manually,
   so that I can test specific movements and their effects on cable behavior
7. As an engineer,
   I want to have a realistically sized robot model,
   So that I can test the size constraints of the robot in different environments
8. As an engineer,
   I want to have a mode where I can slow down or pause the simulation,
   So that I can examine the simulation more clearly
9. As an engineer,
   I want to be able to load different maps
   So that I can easily test different environments for the robot
10. As an engineer,
    I want real-time data on the stress and strain of the system
    So that I can understand and assess points of failure
11. As an engineer,
    I want to be able to save cable data
    So that I can analyze the data further over a longer period of time
12. As an engineer,
    I want the ability to generate graphs of tension over time
    So that I can analyze when tension was greatest
13. As an engineer,
    I want the simulation to provide warnings of unsafe tension levels
    So that I can avoid potential breakages
14. As an engineer,
    I want a feature that allows the robot to run automatically
    So that I can see different cable performances under the same circumstances

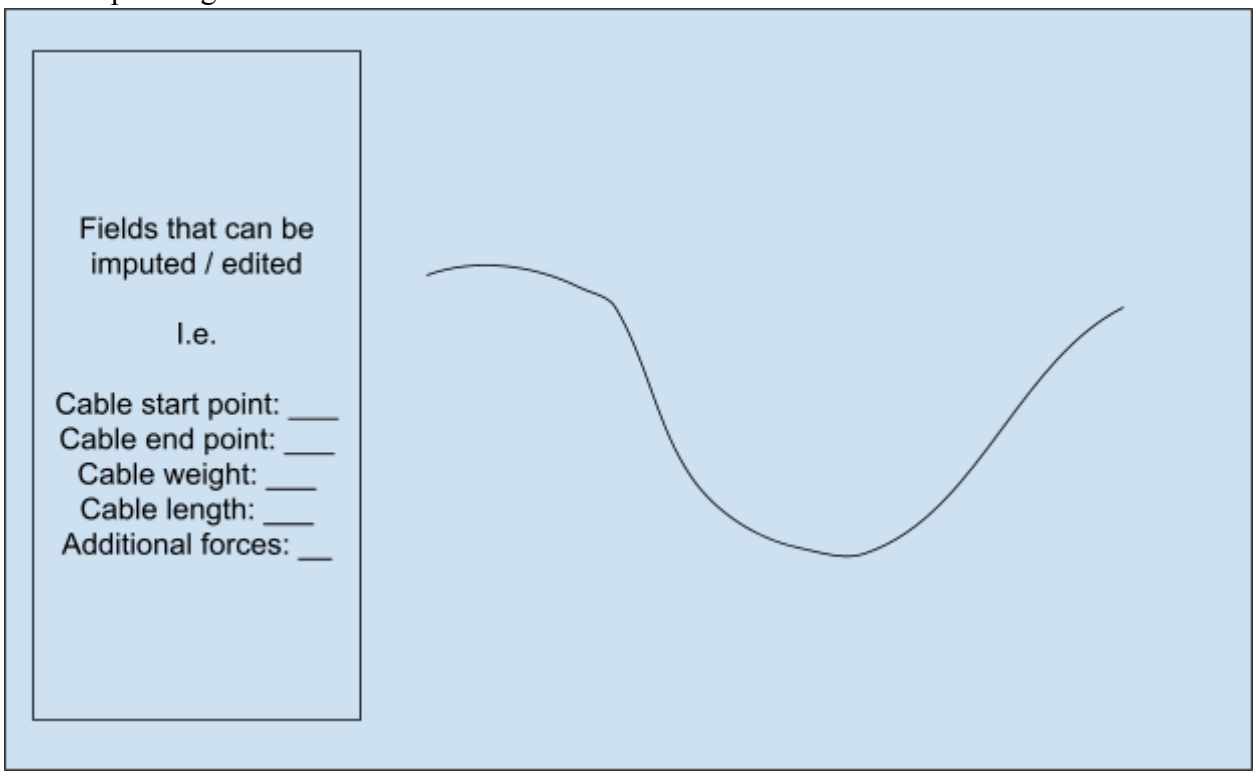
15. As an engineer
    I want the cables to have tension-dependent snapping behavior
    So that I can identify failure points and ensure realistic material constraints
16. As an engineer,
    I want the ability to compare simulation results from multiple test runs,
    So that I can evaluate how different cable setups impact overall performance
17. As an engineer,
    I want the robot to respond to cable resistance,
    So that I can evaluate how cable tension affects the robot's movement
18. As an engineer,
    I want a user-friendly interface for modifying cable properties,
    So that I can easily test different materials and configurations
19. As an engineer,
    I want the cables to experience friction against surfaces,
    So that I can accurately model resistance when the cables are dragged
20. As an engineer,
    I want to visualize stress points on the cables in color-coded overlays,
    So that I can quickly assess areas at high risk of failure

Annex 4: Prototype

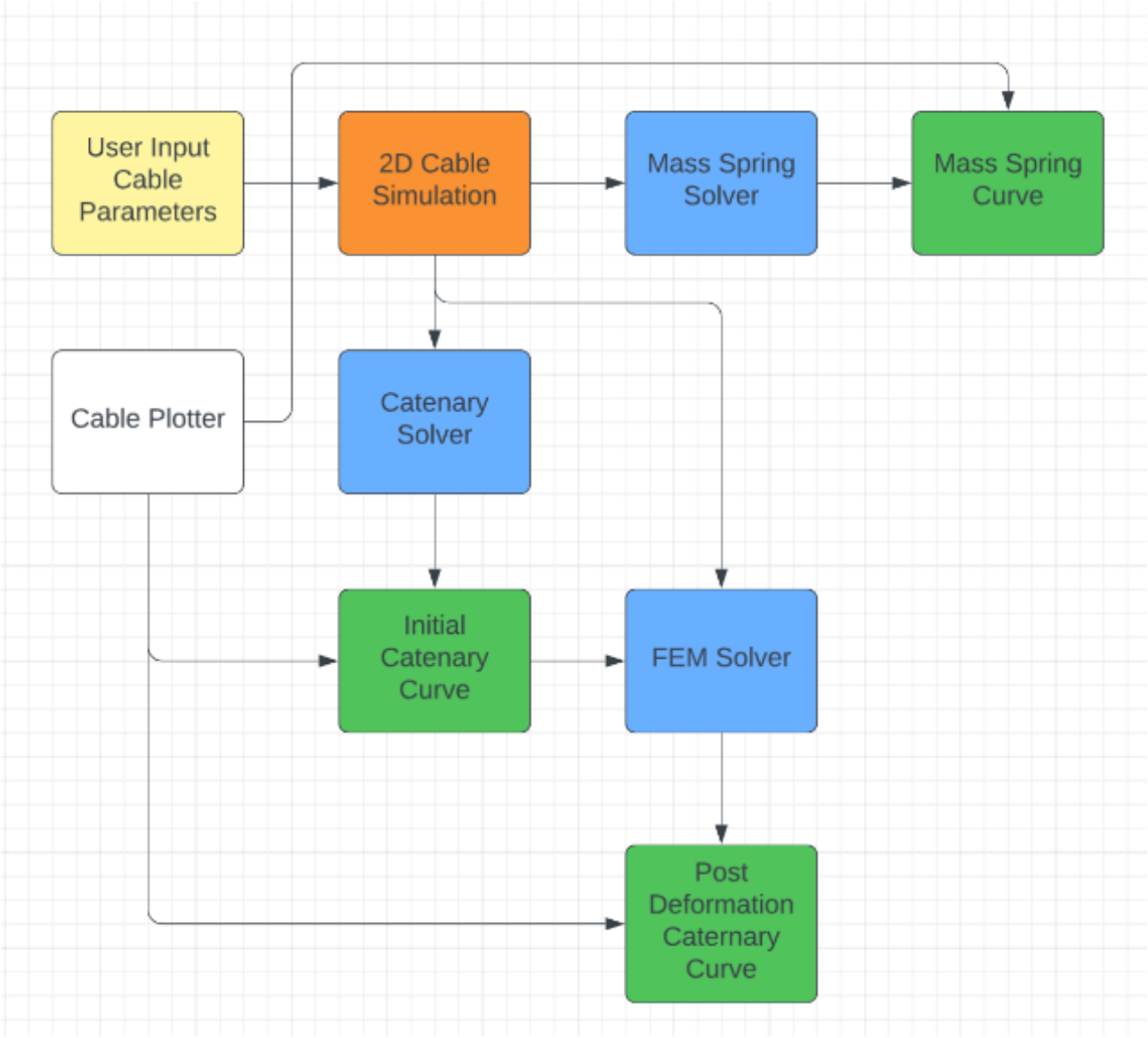
Original designs:



New scope designs:



Annex 5: Design Models
Data Model (Additional)



Data Attributes:

- float massPerUnit // γ (kg/m)
- float length // L_cable (m)
- float horizontalSpan // L (m)
- float heightDiff // h (m)
- float youngsModulus // E (Pa)
  int numSegments
- Vector2[] forceVectors // External point loads

Annex 6: Test Cases

Test Case: Simulation Startup Experience

Objective: Validate that a new user can launch and understand the simulation interface easily.

Sunny Day:

- New user opens the application.
- Starts simulation without needing detailed instruction.
- Understands interface and camera controls intuitively.

Rainy Day:

- User loads the program mid-simulation state with corrupt or partial save data.
- Simulation must detect and handle incomplete state, prompt user to recover or reset.
- UI should remain functional even if state data is invalid.

Test Case: Cable Parameter Adjustment UI

Objective: Ensure users can successfully input and adjust cable parameters.

Sunny Day:

- User enters custom values for cable length, stiffness, and density.
- Simulation updates the cable model accordingly and displays changes.

Rainy Day:

- Users enter combinations of values that result in unstable or paradoxical conditions (e.g., extremely long, low-mass, high-tension cable).
- System must detect physically invalid configurations and either auto-correct, warn the user, or fail gracefully.

Test Case: Load Application via Slider

Objective: Allow users to incrementally apply load to cable and view physical response.

Sunny Day:

- User drags the load slider.
- Simulation visibly updates force diagrams and cable behavior in real time.

Rainy Day:

- User uses an automation script or faulty input device to spam the slider rapidly.
- Simulation receives hundreds of rapid load changes per second.
- System must throttle input rate and prevent physics instability.

Test Case: Camera Control and Perspective Adjustment

Objective: Ensure users can freely move the camera to inspect the cable from multiple angles.

Sunny Day:

- User orbits, pans, and zooms the camera.
- No stutter or interface conflict occurs.

Rainy Day:

- Camera must still behave predictably and not jitter or lock when moved at speed.
- Input priority must be resolved clearly in documentation and code.

Test Case: Subjective Evaluation (Likert Scale)

Objective: Gather user feedback on simulation usability, output clarity, and data utility.

Sunny Day (Likert Questionnaire): Users will be asked to rate the following statements on a scale of 1 (Strongly Disagree) to 5 (Strongly Agree):

1. I was able to generate useful or informative results from this simulation.
2. I was able to control the camera to observe the final outcome clearly.
3. The simulation provided enough data to be useful for engineering purposes.
4. The cable's movement appeared realistic under simulated conditions.
5. The interface made it easy to interact with and configure the simulation.

Rainy Day:

- Test is conducted with minimal instruction and unfamiliar users under time constraint.
- Users rate low scores (1 or 2) on any item.

- Test team must capture feedback on specific points of friction or confusion for targeted improvement.

Test Case: FEM Deformation Accuracy and Stability

Objective: Validate that the FEM solver produces accurate and stable results under different loading conditions.

Sunny Day:

- User sets up a uniformly discretized cable with appropriate material properties (e.g., Young's Modulus, area, density).
- Applies a static point load or distributed load.
- Simulation converges and displays expected nodal displacements and internal forces.
- Output is consistent with analytical or benchmark solutions.

Rainy Day:

- User provides extremely stiff or soft material values, or very high loads.
- Simulation attempts to run but detects numerical instability or excessive deformation.
- System must:
    - Warn the user if the condition number is too high or the solver diverges.
    - Prevent display of misleading results.
    - Offer diagnostic output (e.g., max residual, node index with max displacement).

Test Case: FEM Time-Step Handling

Objective: Ensure proper integration of time into quasi-static or dynamic FEM simulations.

Sunny Day:

- User specifies a reasonable Δt.
- Simulation iteratively solves displacement until convergence at each time step.
- Cable deforms smoothly in visualization; energy trends are stable.

Rainy Day:

- Δt is unrealistically large or set to 0.
- System detects the issue and:
    - Provides informative feedback.
    - Prevents execution with unstable parameters.

Test Case: Mass-Spring Simulation – Real-Time Responsiveness

Objective: Validate that the mass-spring simulation responds accurately to force inputs.

Sunny Day:

- User initializes nodes and springs with reasonable mass, stiffness, and damping.
- Gravity and damping applied.
- Cable oscillates and stabilizes.
- Displacement, velocity, and acceleration values are updated smoothly in real time.

Rainy Day:

- User removes all damping or applies a massive force impulse.
- System must prevent numeric explosion:
    - Clamp velocities.
    - Apply emergency damping or auto-pause.
    - Output warning if kinetic energy exceeds threshold.

Test Case: Mass-Spring Time Integration (Semi-Implicit Euler)

Objective: Ensure semi-implicit Euler updates are consistent and stable across varying frame rates.

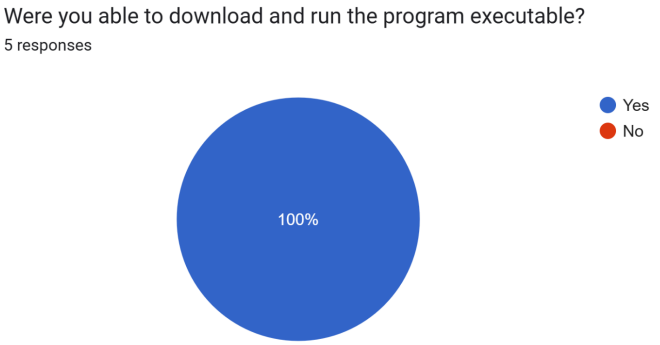
Sunny Day:

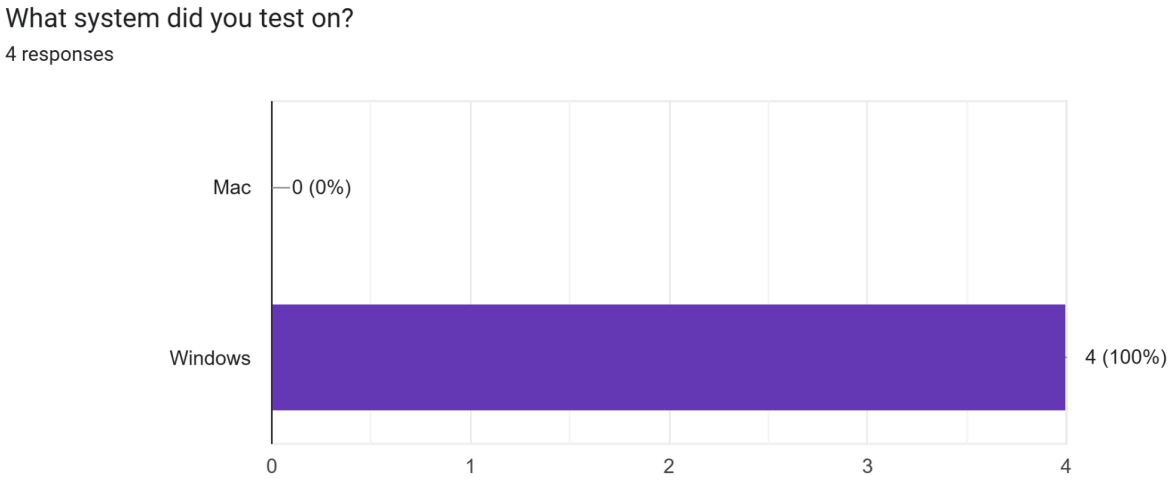
- Simulation runs at 60 FPS with dt locked to frame time.
- Node positions and velocities remain stable.
- Damping prevents long-term oscillation.

Rainy Day:

- User unlocks framerate or manually sets erratic dt values (e.g., alternating 0.01s and 0.5s).
- Simulation must:
    - Clamp maximum timestep or issue warning.
    - Interpolate or resample input forces across variable dt.

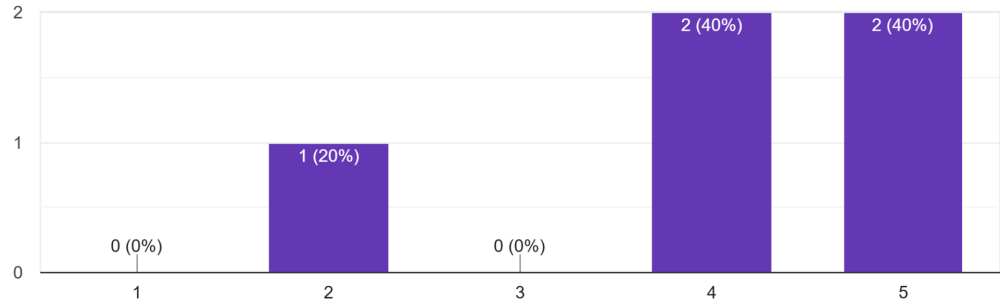
Annex 7: User Acceptance Test Artifacts





I was able to generate useful or informative results from this simulation.

5 responses



How can we improve in this area? (3 responses)

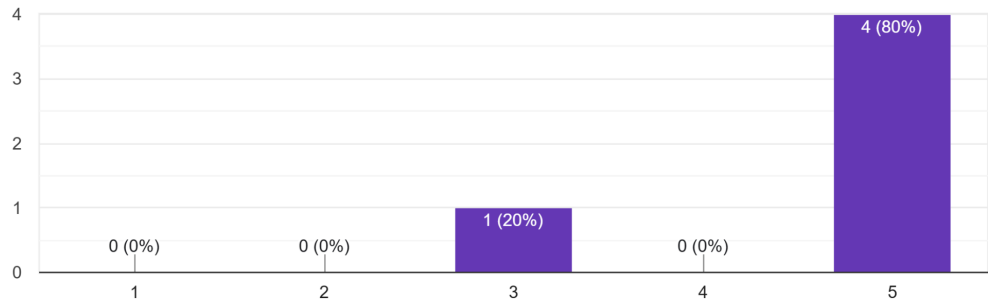
Adding units to the input (e.g. kg, m) and axes to the output; as a standalone application it works for getting qualitative intuition/feel for how the mass changes the cable sag but quantitative information would be useful. It looks like you're printing out the sag and the tension in the console; it would be nice if this were displayed in the GUI too.

Run time with extra segments (n>20) was a little long but that might be a hardware issue on my end. Simulating multiple stops, say going to the end of the tank and after that simulating the return and how that would bend the "cable" would be very useful even if there is only one additional waypoint. Add a grid or at a minimum numbered ticks on the X and Y axis. Also label the start point and end point X and Y. - Windows version

The software seems to be very buggy in that it does not reset well. Also, you need to do something to check that the length value is valid given the start and end points. It seems the system goes into failure if the length is too short given the start and end points. Length should never be allowed to be shorter than the straight line distance between the start and end points.

I was able to control the camera to observe the final outcome clearly and in sufficient detail.

5 responses



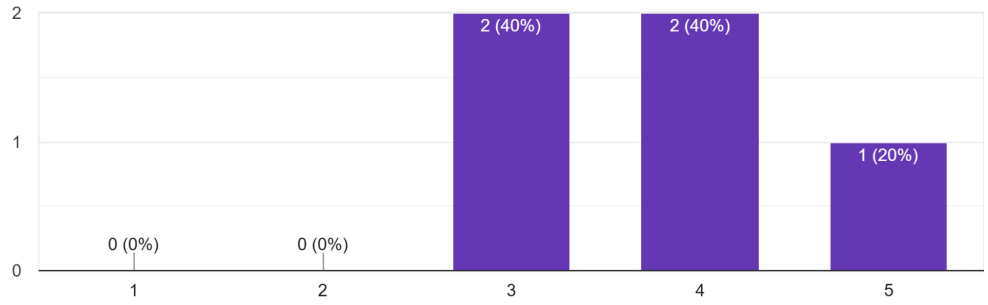
How can we improve in this area? (2 responses)

No improvements needed, thank you for a recenter button. - Windows version

The camera works

The simulation provided enough data to be useful for engineering purposes.

5 responses



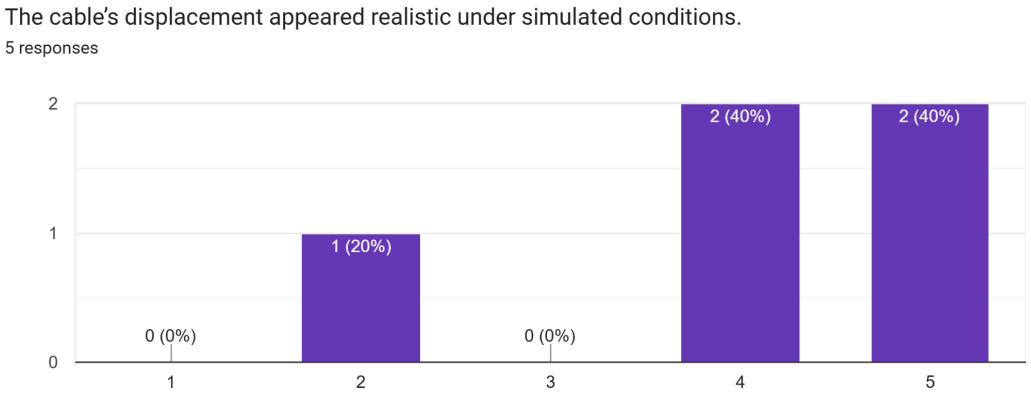
How can we improve in this area? (3 responses)

Again, units/axes would be an improvement if this is a standalone application. As a module I integrate with my own code this looks useful (and would likely save me a lot of time vs. having to write it myself).

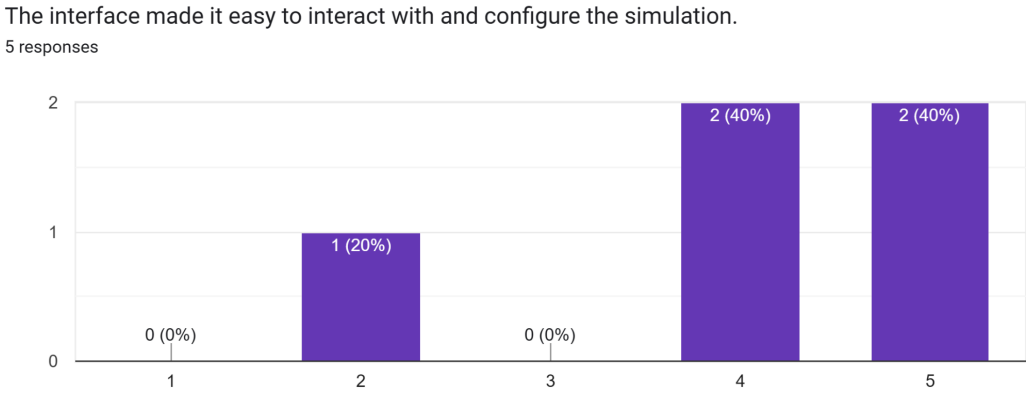
This is a good start, some useful features would include a second waypoint as I mentioned above. A stretch goal would be to add "curl/kink/knot" warnings if the cable pathway re-curves over itself. I

was de-tangling some wired earphones and that would be impossible to do in the tank. - Windows version

It would be better if the bugs associated with updating the simulation were fixed.



No Qualitative Results/Responses.



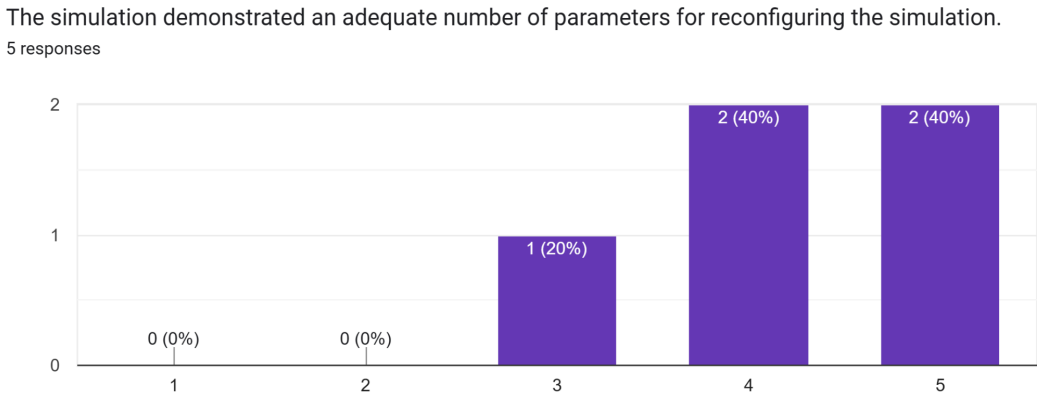
How can we improve in this area? (4 responses)

more labels

It would be good to communicate when the model parameters are invalid (e.g. when there is no solution). See 'feature preferences' question.

The features present were easy to configure. - Windows Version

The interface has alot of bugs. It would be nice if the start and end points could be selected with the mouse, but it is more important to fix the bugs that make the current interface fail first.



How can we improve in this area? (4 responses)

the external force function might need label for x and y direction

Seems fine to me; I do not have a mechanical background so there may be other forces/material parameters that would be of interest I'm not aware of. It might be nice to offer a density option (e.g. kg/m) for the cable (perhaps with some presets for various material types).

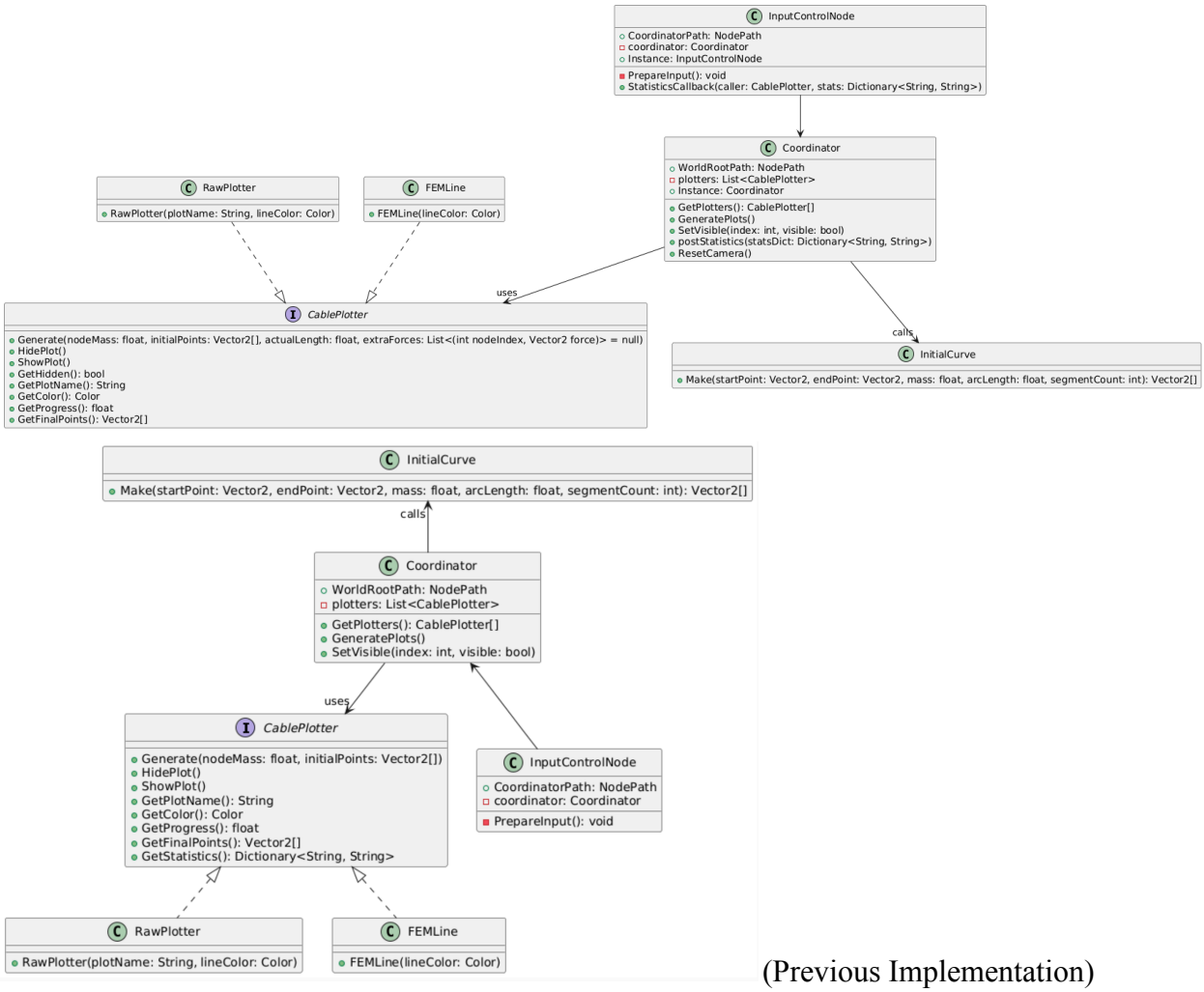
I said this above but adding another waypoint would be very useful. - Windows Version

It would be nice if the start and end points could be selected with the mouse, but it is more important to fix the bugs that make the current interface fail first.

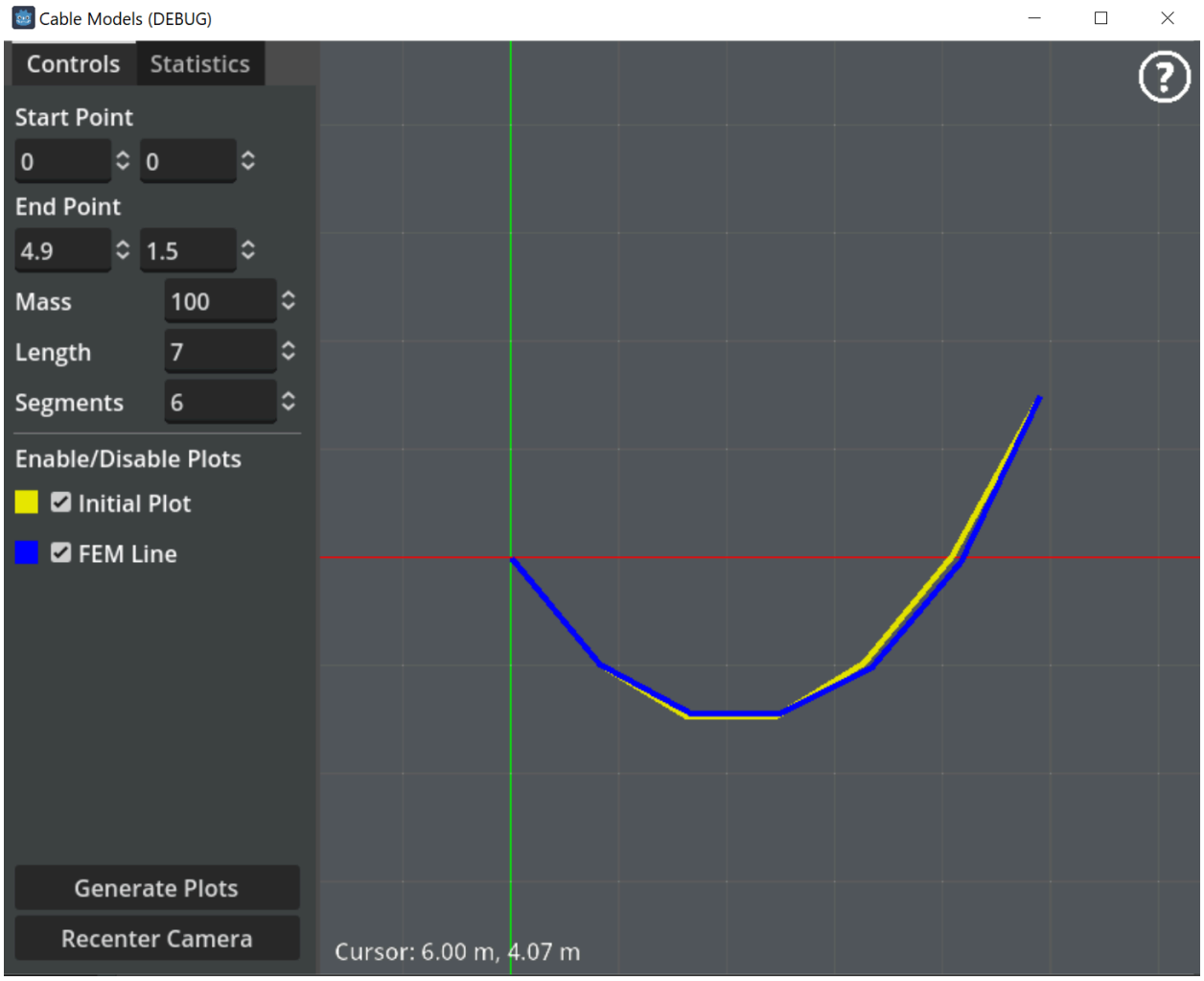
User Acceptance Testing Form (Responses)

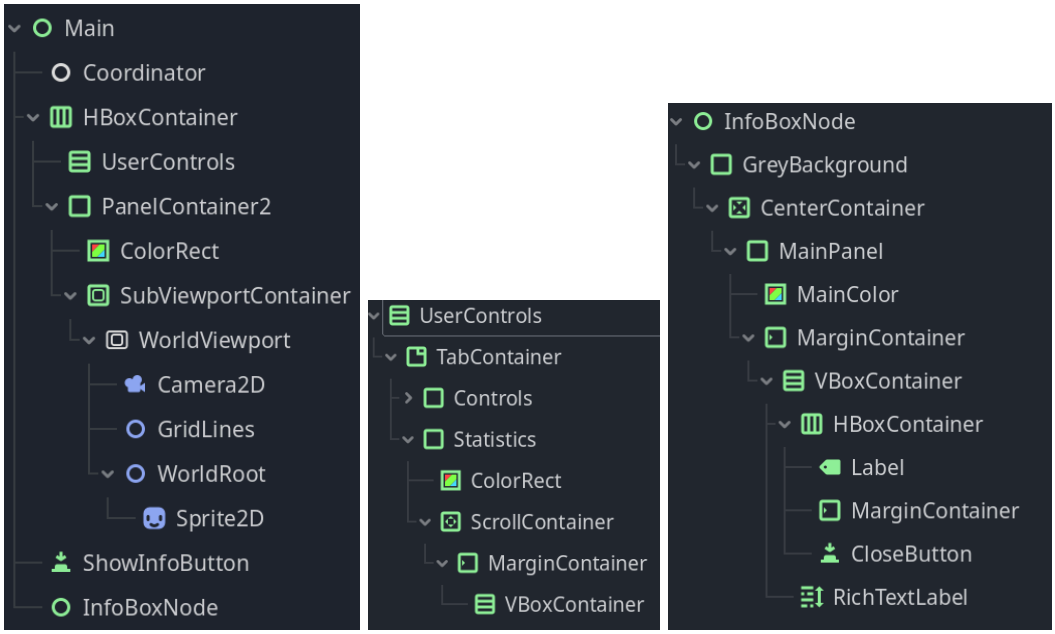
This linked spreadsheet contains individual responses to our questions including qualitative data.

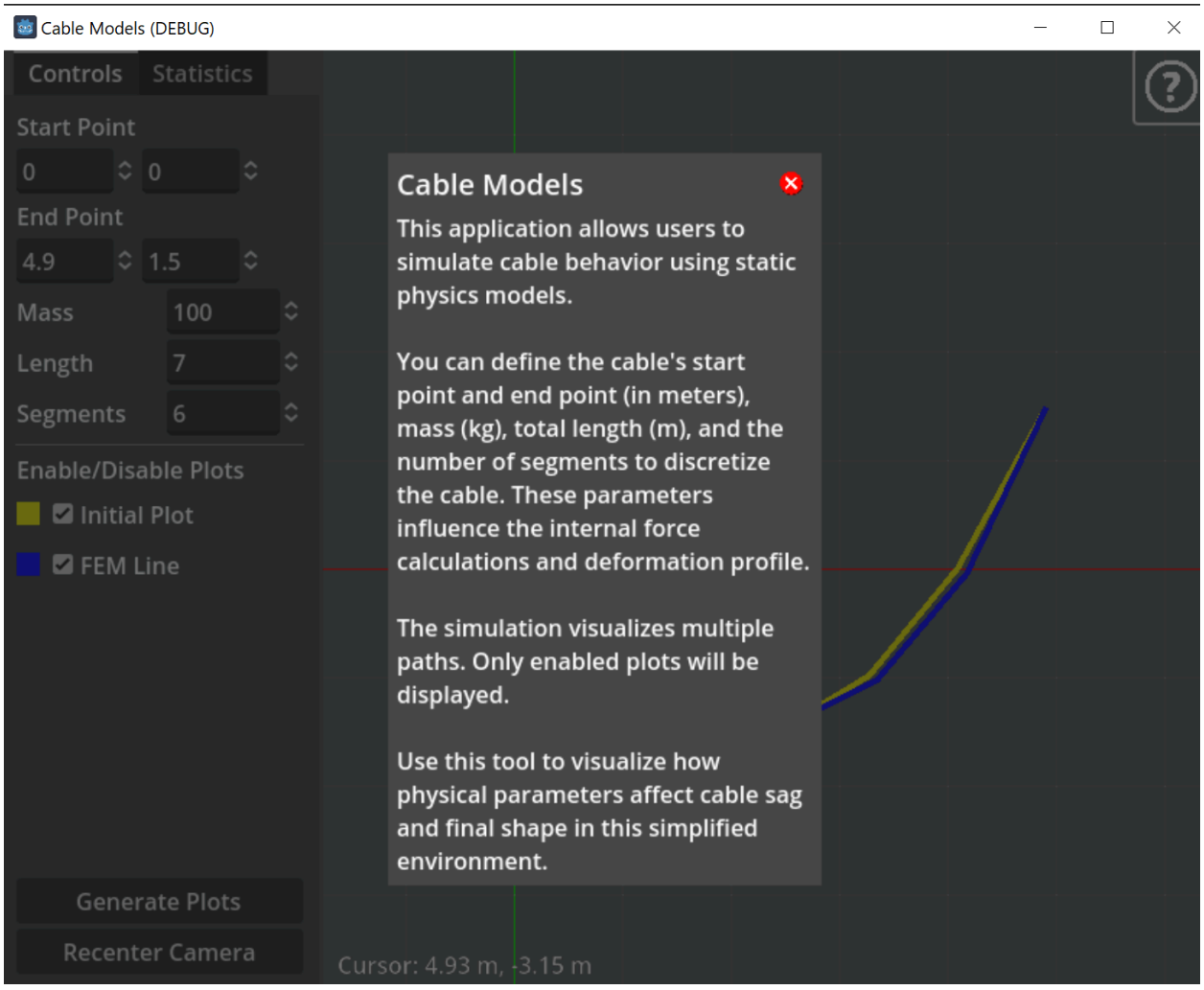
Annex 8: Implementation



(Previous Implementation)


Cable Models (DEBUG)
Controls
Statistics
Start Point
0
0
End Point
4.9
1.5
Mass
100
Length
7
Segments
6
Enable/Disable Plots
Initial Plot
FEM Line
Generate Plots
Recenter Camera
Cursor: 6.00 m, 4.07 m


Main
Coordinator
HBoxContainer
UserControls
PanelContainer2
ColorRect
SubViewportContainer
WorldViewport
Camera2D
GridLines
WorldRoot
Sprite2D
ShowInfoButton
InfoBoxNode
UserControls
TabContainer
Controls
Statistics
ColorRect
ScrollContainer
MarginContainer
VBoxContainer
InfoBoxNode
GreyBackground
CenterContainer
MainPanel
MainColor
MarginContainer
VBoxContainer
HBoxContainer
Label
MarginContainer
CloseButton
RichTextLabel


Cable Models (DEBUG)
Controls
Statistics
Start Point
0
0
End Point
4.9
1.5
Mass
100
Length
7
Segments
6
Enable/Disable Plots
Initial Plot
FEM Line
Generate Plots
Recenter Camera
Cable Models
This application allows users to simulate cable behavior using static physics models.
You can define the cable's start point and end point (in meters), mass (kg), total length (m), and the number of segments to discretize the cable. These parameters influence the internal force calculations and deformation profile.
The simulation visualizes multiple paths. Only enabled plots will be displayed.
Use this tool to visualize how physical parameters affect cable sag and final shape in this simplified environment.
Cursor: 4.93 m, -3.15 m

Annex 9: User's Manual

Installation

Method 1: Using Godot 4 .NET for Development

1. Ensure you have .NET enabled Godot 4 installed on your system. You can download it from the official Godot website.

2. Open Godot and navigate to the project directory.

3. Import the project by selecting the `project.godot` file.

4. Run the project within Godot to start the simulation.

Method 2: Downloading the Release from GitHub

1. Go to the GitHub releases page.

2. Download the appropriate release for your operating system:

   - For Windows: `CableModelsWindows.zip`

   - For Mac: `CableModelsMac.zip`

3. Extract the downloaded zip file to a desired location on your computer.

4. Run the executable file to start the simulation.

Information

This application allows users to simulate cable behavior using static physics models.

You can define the cable's start point and end point (in meters), mass (kg/m), total length (m), and the number of segments to discretize the cable. These parameters influence the internal force calculations and deformation profile. Press the "Generate" button to rerun the simulation.

The statistics tab contains details on simulation outputs.

The simulation visualizes multiple paths. Only enabled plots will be displayed.

Use this tool to visualize how physical parameters affect cable sag and final shape in this simplified environment.

Where is the Documentation?

Documentation on this project and code can be found in the permanent public repository for this project. There is also a readme available in the distributable with necessary information.