1 Introduction

The main goal of this experiment is to explore and understand haptic interaction methods using the Novint Falcon interface. The objective is to implement a teleoperation system with feedback, aiming to maximize transparency and provide an immersive user experience.

Haptic interfaces allow users to interact with virtual environments by feeling physical forces. The Novint Falcon, originally designed for video games, was chosen for its impedance coupling method, which calculates forces based on the movement and speed of the end-effector. This approach is simpler to implement and widely used in industry, although it sacrifices some precision.

The experiment involves two key phases. First, the mechanical properties of the Novint Falcon, such as the mass of the end-effector and axis stiffness, are analyzed. Basic virtual objects like springs, walls, and spheres are implemented to evaluate the quality of force feedback. Second, the Falcon is integrated with a 3D simulator like Blender to model more complex environments. The interaction between rigid objects is analyzed, and direct and indirect coupling methods are compared to assess their impact on haptic sensations and interaction stability.

By combining an analysis of the Falcon's physical properties and its integration into a virtual environment, this experiment aims to deepen the understanding of haptic principles and explore potential improvements for future applications.

2 Novint Falcon device overview

According to the research of Martin and Hillier (2009)[1], the Novint Falcon is a 3-degree-of-freedom (3-DOF) haptic device developed by Novint Technologies and initially designed for gaming. It is affordable and easy to access, making it an excellent option for entry-level research.

The Falcon's design is based on delta robots, with three actuators mounted on a base and an end-effector constrained to translational motion by kinematic parallelograms. Unlike traditional delta robots, it employs single-degree-of-freedom rotary joints instead of spherical joints[2]. The interchangeable end-effectors enhance its versatility, accommodating various applications, such as gaming pistol grips or custom attachments like pen holders and grippers. The device connects to a computer via USB and relies on firmware to process control commands and provide encoder feedback.

In terms of kinematics, the Falcon is capable of delivering maximum force in the central region of its workspace, although torque demands increase as it approaches its travel limits. These characteristics underline both its strengths and limitations concerning range and precision.

Overall, the Novint Falcon is a highly suitable platform for entry-level delta-type robotics research. Its affordability, accessibility, and dependable kinematic and dynamic performance make it a valuable tool for exploring parallel robot control and estimation. Despite some limitations, such as unmodeled friction and potential joint wear, its potential for academic and practical applications remains significant.

3 Part 1: Exploring the physical properties of the Novint Falcon

To understand the fundamental capabilities of the Novint Falcon, this part of the experiment focuses on exploring its physical characteristics. By analyzing its workspace, implementing virtual objects, and identifying dynamic parameters such as mass and damping, we aim to evaluate the device's performance as a haptic interface. The results of this exploration will lay the foundation for more advanced applications, including its integration with a 3D simulation environment.

3.1 Workspace Exploration

In this section, we aimed to determine the workspace dimensions of the Novint Falcon device. By manually moving the handle to its extreme positions along each axis and recording the corresponding coordinates, we quantified the reachable space of the device.

Using a Python-based script, we recorded the real-time positions of the handle and saved the data for further analysis. The results showed the following workspace dimensions:

- **x-axis:** from $-0.05\,\text{m}$ to $0.05\,\text{m}$.
- **y-axis:** from $-0.07\,\text{m}$ to $0.08\,\text{m}$.
- **z-axis:** from $-0.07\,\text{m}$ to $0.08\,\text{m}$.

A visualization of the recorded data highlights the distribution of reachable positions. However, we observed that the workspace along the y-axis and z-axis is not symmetric, which we suspect may be due to aging or wear of the device.

3.2 Virtual Spring and Damper System

In this section, we implemented a virtual spring-damper system to simulate haptic feedback using the Novint Falcon. The feedback force was computed based on the handle's displacement and velocity, following the equation:

$$F = -kx - c\dot{x} \tag{3.1}$$

where F is the feedback force, x is the displacement, k is the spring stiffness, and c is the damping coefficient. This system provides realistic force feedback and mimics the behavior of a physical spring and damper. When the handle reaches the position $(0.0, 0.0, 0.0)$, the force becomes zero, meaning that the system does not exert any force. As a result, the handle automatically returns to this equilibrium position when released.

3.2.1 Position vs. Force Curve

In this experiment, we analyzed the relationship between the handle's position and the feedback force, as shown in Figure 1. The handle was moved slowly to reduce the effect of velocity on the force.

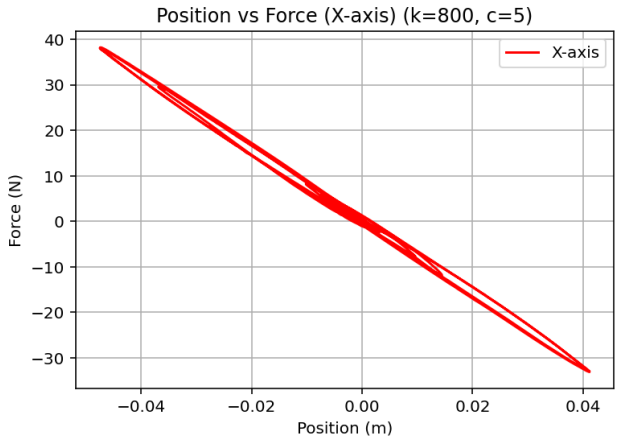


Figure 1: Analysis of Position vs. Force Relationship

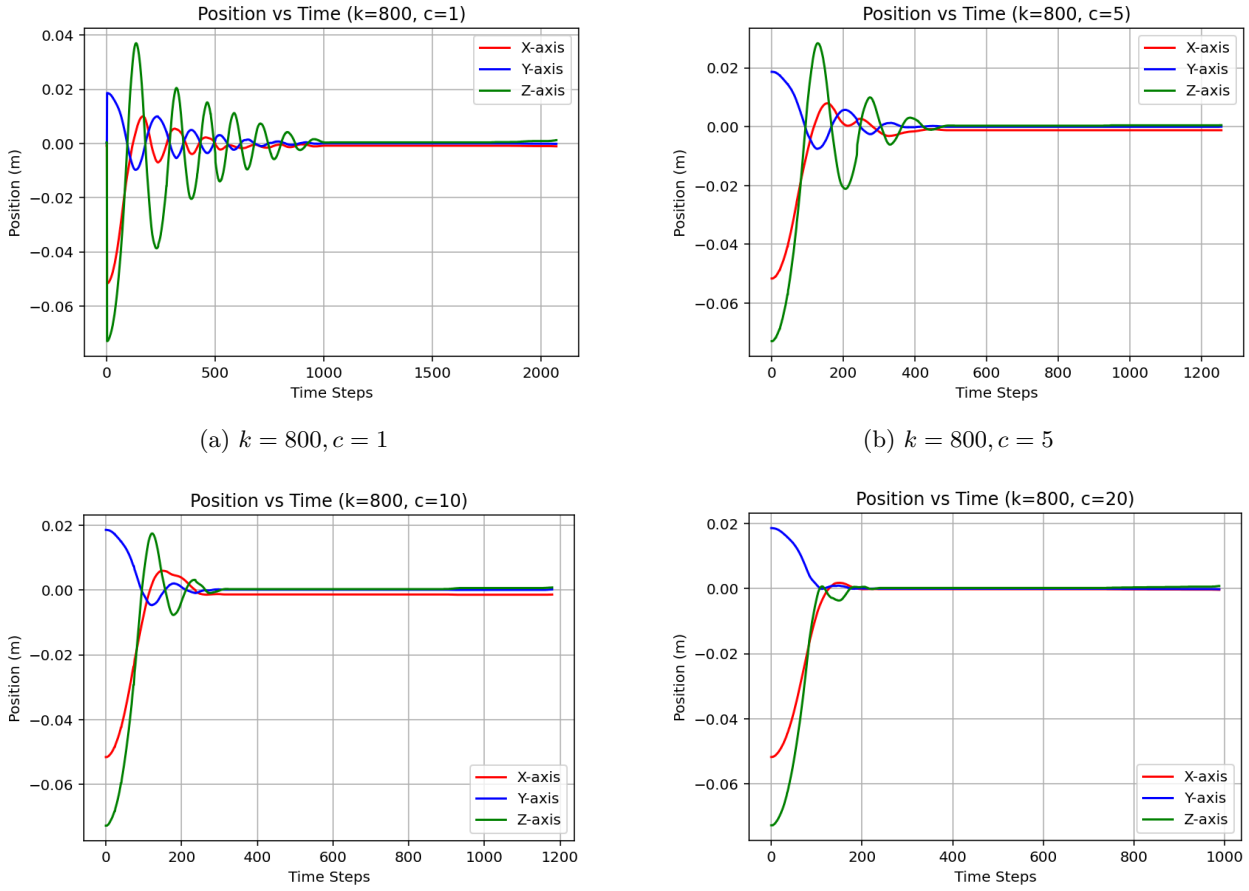
The position vs. force curve shows the following key observations:

- The data points form a nearly linear relationship, indicating that the feedback force is mainly determined by the spring term $-kx$.
- Small deviations from linearity may result from damping effects, but they are minimal due to the slow movement of the handle.
- The slope of the curve matches the spring stiffness k, confirming that the model is implemented correctly.

The linear relationship between position and force agrees with the theoretical model. This demonstrates that the spring-damper system can generate accurate haptic feedback under slow motion.

3.2.2 Force vs. Time Curve

In this experiment, we analyzed the system's behavior under different damping coefficients while keeping the spring stiffness constant at $k = 800\,\mathrm{N/m}$. The handle was initially placed at the bottom-most position $(-0.05, 0.0, -0.07)$, and the system was allowed to return to its equilibrium position $(0.0, 0.0, 0.0)$. The position vs. time curves, shown in Figure 2, illustrate the effect of varying the damping coefficient c with values of $1, 5, 10$, and $20\,\mathrm{Ns/m}$.



(a) $k = 800, c = 1$

(b) $k = 800, c = 5$

(c) $k = 800, c = 10$

(d) $k = 800, c = 20$

Figure 2: Position vs. Time Curves for Different Damping Coefficients

For $c = 1$, the system is underdamped, showing significant oscillations around the equilibrium position and taking a long time to stabilize. When $c = 5$, the oscillations are reduced, and the system stabilizes more quickly, though it remains slightly underdamped. At $c = 10$, the system approaches critical damping, minimizing oscillations and efficiently returning to equilibrium without significant overshooting. For $c = 20$, the oscillations are almost completely eliminated, but the system exhibits a potential risk of overdamping. Although an increase in equilibrium time due to overdamping was not observed in the experiments, overdamping may reduce the responsiveness of the system in more dynamic scenarios.

3.3 Free Oscillation Analysis

In this experiment, we studied the free oscillation behavior of the Novint Falcon system to identify key dynamic parameters, including the damping ratio (ζ), natural frequency (ω_n), and equivalent mass (m). The handle was manually displaced to an initial position and released, allowing the system to oscillate freely. A high spring stiffness ($k = 800\,\mathrm{N/m}$) was applied to simulate a nearly ideal elastic system. The position of the

handle along the x-axis and the corresponding time were recorded, and the resulting free oscillation curve is shown in Figure 3.

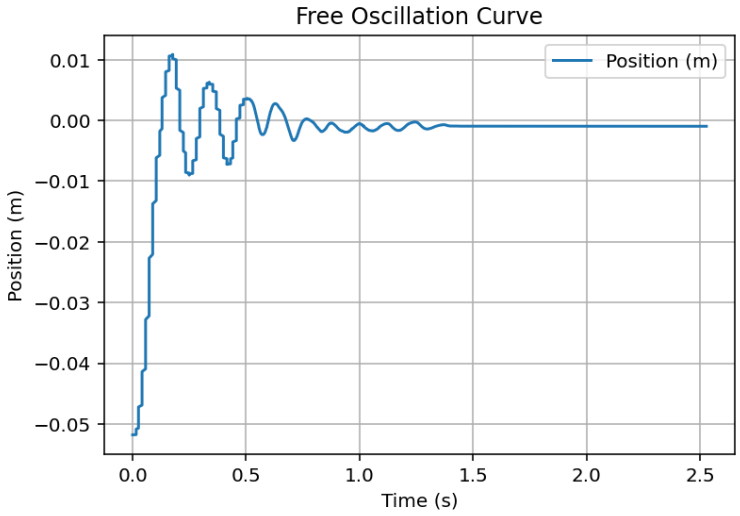


Figure 3: Free Oscillation Curve: Handle position vs. time under a spring stiffness of $k = 800\,\mathrm{N/m}$.

The free oscillation curve shows a typical underdamped behavior, with the oscillations gradually becoming smaller over time. The system slowly returns to the equilibrium position ($x = 0.0\,\mathrm{m}$), which confirms that it is lightly damped. However, the curve is not completely smooth. This could be caused by the sampling rate being too low or noise in the position data. Additionally, small mechanical issues in the device or vibrations from the environment might also affect the smoothness of the curve.

3.3.1 Calculation of Dynamic Parameters

To quantify the system's dynamic characteristics, we calculated the damping ratio, natural frequency, and equivalent mass as follows:

Damping Ratio (ζ): The logarithmic decrement (δ) was computed from the ratio of consecutive peak amplitudes (x_i and x_{i+1}):

$$\delta = \ln\left(\frac{x_i}{x_{i+1}}\right) \tag{3.2}$$

The average logarithmic decrement was used to calculate the damping ratio:

$$\zeta = \frac{\delta}{2\pi} \tag{3.3}$$

For the detected peaks, we obtained $\zeta = 0.0724$, indicating a lightly damped system.

Natural Frequency (ω_n): The time intervals between consecutive peaks were used to calculate the average oscillation period (T). The natural frequency (ω_n) was then computed as:

$$\omega_n = \frac{2\pi}{\text{Average Period}} \tag{3.4}$$

From the experimental data, the calculated natural frequency was $\omega_n = 41.75\,\mathrm{rad/s}$.

Equivalent Mass (m): Using the relationship between spring stiffness (k) and natural frequency (ω_n), the equivalent mass was determined as:

$$m = \frac{k}{\omega_n^2} \tag{3.5}$$

The equivalent mass was calculated to be $m = 0.4589\,\text{kg}$.

3.3.2 Discussion of Results

The free oscillation curve reflects the system's underdamped behavior, as expected for a low damping ratio ($\zeta = 0.0724$). The natural frequency ($\omega_n = 41.75\,\text{rad/s}$) indicates the system's intrinsic oscillation speed, while the equivalent mass ($m = 0.4589\,\text{kg}$) represents the dynamic properties of the handle and its associated components.

3.4 Implementation of a Virtual Wall

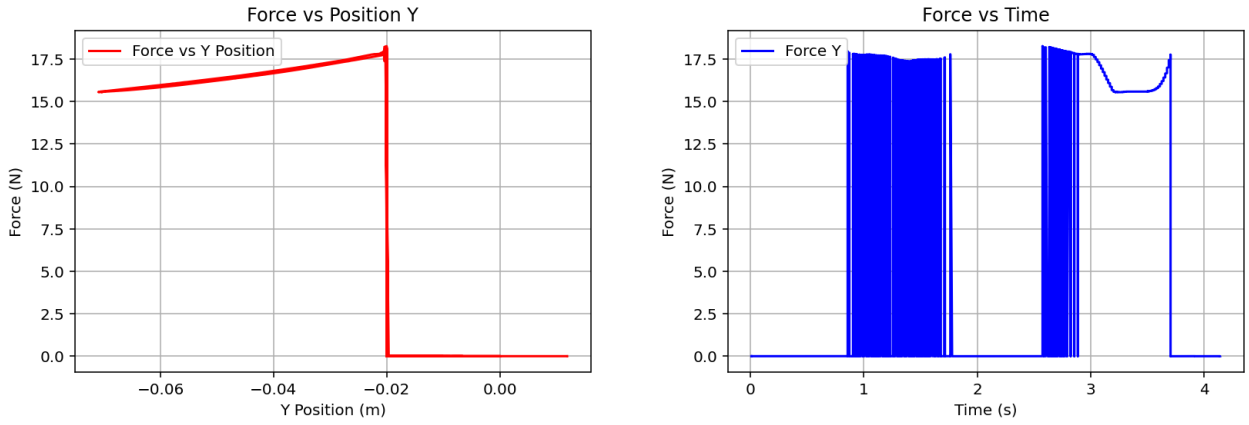
In this section, we implemented a virtual wall for the plane $y < -0.02\,\text{m}$. The goal was to simulate the presence of a physical barrier using feedback forces generated by the Novint Falcon. To achieve this, we designed and tested three different force feedback strategies: rigid force, linear force, and nonlinear force. Each strategy represents a unique approach to model the interaction with the virtual wall and is analyzed in the following subsections with corresponding explanations, justifications, and visualized results.

3.4.1 Rigid Force Implementation

In this subsection, we implemented a rigid force model to simulate a virtual wall at $y = -0.02\,\text{m}$. The feedback force was applied as a constant upward force of $20\,\text{N}$ when the handle crossed the plane ($y < -0.02\,\text{m}$). Otherwise, no force was applied. The implementation logic is reflected in the following equation:

$$F(y) = \begin{cases} 20\,\text{N}, & \text{if } y < -0.02\,\text{m} \\ 0\,\text{N}, & \text{if } y \geq -0.02\,\text{m}. \end{cases} \tag{3.6}$$

The experimental data collected during the simulation is visualized in Figure 4.



(a) Force vs. Position y: Rigid force implementation.

(b) Force vs. Time: Rigid force implementation.

Figure 4: Visualization of rigid force implementation: the relationship between force and y-position (left), and the force profile over time (right).

From Figure 4a, we observe that the feedback force is strictly constant ($20\,\text{N}$) when the handle crosses into the region $y < -0.02\,\text{m}$, and drops abruptly to zero when $y \geq -0.02\,\text{m}$. This sharp transition represents the rigid wall's force profile. Figure 4b shows how the force is applied over time as the handle moves in and out of the virtual wall region.

Analysis and Observations In practice, when the handle makes contact with the virtual wall, the strong and sudden feedback force leads to significant vibrations in the handle. This phenomenon is evident in the oscillatory behavior seen in the force vs. time curve (Figure 4b). After analysis, we attribute these vibrations to the limitations of the Novint Falcon device's motors when exerting large feedback forces. The motor's performance instability under high load results in oscillations that degrade the overall smoothness and realism of the haptic feedback.

Conclusion The rigid force model effectively simulates a hard wall by providing an immediate and strong feedback force. However, the abrupt nature of the force transition causes instability and vibrations, indicating that this approach may not be optimal for applications requiring smooth and stable haptic feedback. Future models (e.g., linear or nonlinear force profiles) may mitigate these issues by introducing gradual transitions in the force application.

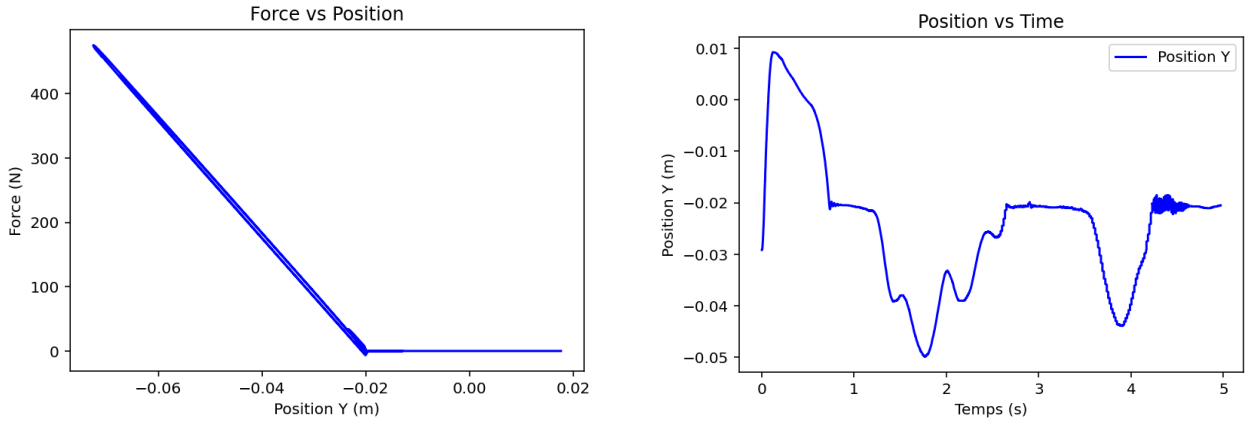
3.4.2 Linear Force Implementation

In this subsection, we implemented a linear force model to simulate a virtual wall at $y = -0.02\,\text{m}$. Unlike the rigid force model, the linear force increases gradually as the handle penetrates deeper into the wall, reducing the sudden impact and mitigating vibrations. The feedback force is calculated as:

$$F(y) = \begin{cases} -k(y - y_r) - c\dot{y}, & \text{if } y \leq y_r \\ 0, & \text{if } y > y_r \end{cases} \tag{3.7}$$

where k is the spring stiffness, c is the damping coefficient, y is the current position, $y_r = -0.02\,\text{m}$ is the wall position, and $\dot{y}$ is the velocity of the handle along the y-axis. Dynamic damping adjustment was applied during the experiment to further stabilize the system by computing the logarithmic decrement based on consecutive peaks in the position data.

The experimental data is visualized in Figure 5, which includes the force vs. position curve and the position vs. time curve.



(a) Force vs. Position y: Linear force implementation.

(b) Position vs. Time: Linear force implementation.

Figure 5: Visualization of linear force implementation: the relationship between force and y-position (left), and the handle's motion over time (right).

Analysis and Observations From Figure 5a, the feedback force increases linearly as the handle moves deeper into the wall, effectively reducing the sharp transitions seen in the rigid force model. Figure 5b shows the handle's motion over time, highlighting that the linear force greatly reduces the vibrations observed during contact with the virtual wall. This improvement is due to the smaller initial force applied upon contact, which prevents the sudden feedback that occurred in the rigid force model.

However, despite the overall reduction in vibrations, slight oscillations are still present, as seen in the position vs. time curve. These residual vibrations may be attributed to the limitations of the damping adjustment mechanism or the inherent mechanical properties of the Novint Falcon device.

Conclusion The linear force model successfully mitigates the vibrations observed in the rigid force model by introducing a gradual increase in force during contact. This results in smoother and more stable haptic feedback. While slight oscillations remain, this approach demonstrates significant improvements over the rigid force model, making it more suitable for applications requiring realistic and comfortable interactions with virtual walls.

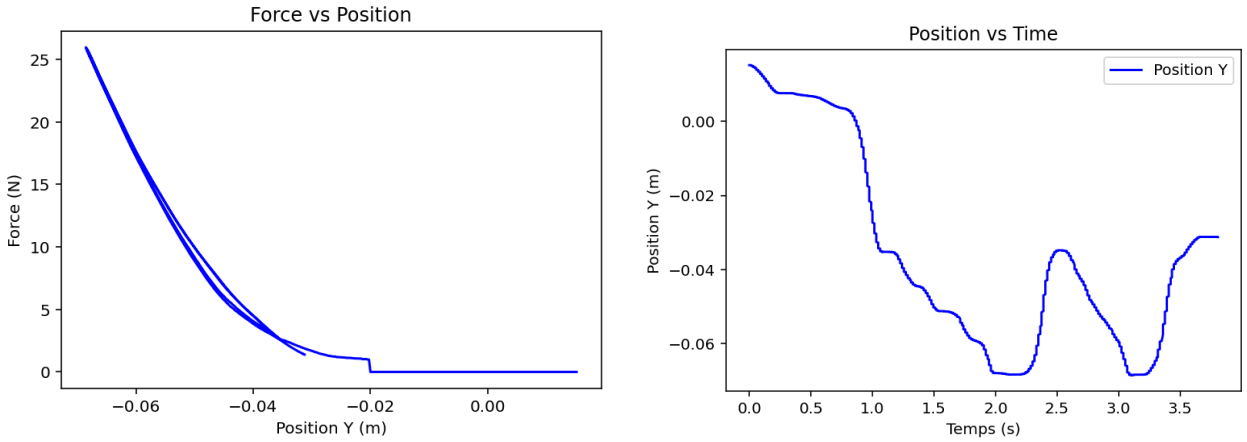
3.4.3 Nonlinear Force Implementation

In this subsection, we implemented a nonlinear force model to simulate a virtual wall at $y = -0.02\,\text{m}$. The nonlinear model adjusts the feedback force based on the displacement and velocity of the handle, with the force increasing nonlinearly as the handle penetrates deeper into the wall. The feedback force is calculated as:

$$F(y) = \begin{cases} -k(y - y_r)|y - y_r|^{n-1} - c\dot{y}|\dot{y}|^{m-1}, & \text{if } y \leq y_r \\ 0, & \text{if } y > y_r \end{cases} \tag{3.8}$$

where k is the spring stiffness, c is the damping coefficient, $n = 2$ and $m = 2$ are the nonlinear exponents for the spring and damping forces respectively, $y_r = -0.02\,\text{m}$ is the wall position, y is the current position, and $\dot{y}$ preview is the velocity of the handle along the y-axis. This nonlinear model provides a variable force feedback that changes based on the handle's interaction with the wall.

The experimental data is visualized in Figure 6, which includes the force vs. position curve and the position vs. time curve.



(a) Force vs. Position y: Nonlinear force implementation. (b) Position vs. Time: Nonlinear force implementation.

Figure 6: Visualization of nonlinear force implementation: the relationship between force and y-position (left), and the handle's motion over time (right).

Analysis and Observations From Figure 6a, we observe that the feedback force increases nonlinearly as the handle penetrates deeper into the wall. Compared to the linear force model, the nonlinear force starts with smaller feedback at low displacements and becomes stronger as the displacement increases. This characteristic reduces the impact during initial contact, preventing vibrations, while also providing stronger feedback at larger displacements for a more realistic interaction.

Figure 6b shows that vibrations are nearly eliminated with the nonlinear force model. This improvement is attributed to the smaller initial force, which minimizes sudden impacts, and the adaptive damping mechanism, which dynamically adjusts the damping coefficient to stabilize the system. The tactile experience resembles

pressing against a high-pressure balloon, providing a more intuitive and responsive haptic feedback compared to the rigid and linear models.

Conclusion The nonlinear force model successfully addresses the limitations of the rigid and linear models by reducing initial vibrations and enhancing feedback at larger displacements. This approach combines stability with realism, making it an ideal choice for simulating physical interactions with virtual walls. The improved tactile experience demonstrates the effectiveness of the nonlinear force model in achieving smooth and natural haptic feedback.

3.4.4 Summary of Force Feedback Implementations

Rigid, linear, and nonlinear force feedback models each offer distinct characteristics, advantages, and limitations, making them suitable for different application scenarios. The rigid model provides an immediate and strong feedback force when the handle crosses the wall boundary, making it simple to implement and effective for simulating hard surfaces. However, its sudden force transitions lead to significant vibrations, reducing the smoothness and realism of the feedback. The linear model, on the other hand, gradually increases the feedback force as the handle penetrates the wall. It reduces initial contact vibrations compared to the rigid model and offers smoother feedback, but still exhibits slight oscillations and lacks adaptability for complex surface interactions. In contrast, the nonlinear model dynamically adjusts the feedback force based on displacement and velocity, minimizing vibrations and enhancing tactile realism. This makes it ideal for simulating flexible or complex surfaces. However, its higher computational complexity may pose challenges in real-time applications.

Overall, linear and nonlinear force models each have their strengths and weaknesses. The linear model is structurally simple and suitable for real-time scenarios involving straightforward surfaces like hard planes. Its sudden feedback changes, however, can cause initial contact vibrations, making it less ideal for applications demanding smooth tactile feedback. The nonlinear model, by dynamically modulating feedback force, significantly reduces initial contact impacts and enhances user experience, making it more suitable for modeling complex or flexible surfaces. Its computational demands, though, might require optimization or simplification for time-sensitive applications. Therefore, selecting the appropriate force feedback model involves balancing feedback quality, computational demands, and specific application requirements.

3.5 Implementation of Spherical and Cylindrical Feedback Systems

Based on the force feedback study of virtual walls, we further apply force feedback to more complex geometric objects. In this section, we implemented two new haptic objects: a sphere and a cylinder. These objects provide more complex interactions compared to the virtual wall. The feedback forces are calculated based on the handle's position relative to the surface of the objects. The following subsections describe the implementation and results for each object.

3.5.1 Spherical Feedback System

In this subsection, we implemented a spherical feedback system. A virtual sphere with a radius of $R = 0.05\,\mathrm{m}$ and centered at $(x_c, y_c, z_c) = (0.0, 0.0, 0.0)$ was created. The feedback force was computed based on the handle's position relative to the sphere's surface. The force increases when the handle penetrates the sphere, providing a realistic haptic sensation.

In this experiment, we chose to use linear feedback forces, as nonlinear forces would require more complex calculations, which were not necessary for this implementation.

Implementation Details The feedback force F was calculated using a combination of spring and damping forces:

$$F = -k(d - R) - c\dot{d} \tag{3.9}$$

where d is the distance between the handle and the sphere center, $\dot{d}$ is the relative velocity, $k = 3000\,\mathrm{N/m}$ is the spring stiffness, and $c = 50\,\mathrm{Ns/m}$ is the damping coefficient. If the handle's distance from the sphere

center was greater than R, no force was applied. The feedback system ensured that the handle would experience a repulsive force when entering the sphere.

The handle's trajectory during the interaction was recorded and visualized alongside the sphere, as shown in Figure 7.

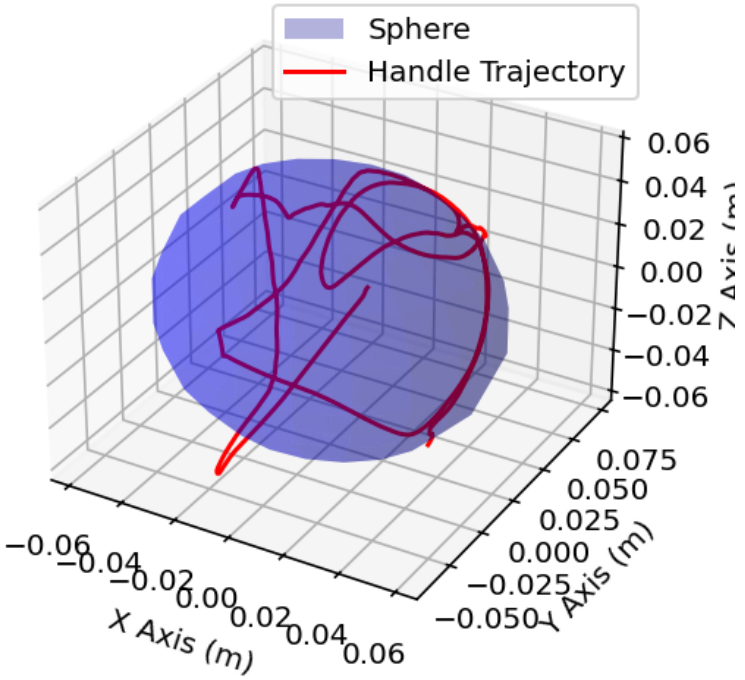


Figure 7: 3D visualization of the sphere and the handle's trajectory. The blue surface represents the sphere, while the red line shows the handle's movement.

Analysis and Observations The spherical feedback system provided a natural and intuitive haptic experience. When the handle entered the sphere, a repulsive force was applied, guiding it back toward the surface. The damping force effectively reduced oscillations during interaction, ensuring smooth motion.

From the visualization in Figure 7, it is evident that the handle's trajectory respected the boundary of the sphere. The feedback force prevented the handle from remaining inside the sphere while allowing smooth exploration around its surface.

The experiment demonstrated that the spherical feedback system successfully simulated a physical object with a soft and responsive boundary, offering a more immersive and realistic interaction compared to the virtual wall.

3.5.2 Cylindrical Feedback System

In this subsection, we extended the feedback system to simulate a cylindrical object. The virtual cylinder was defined with a radius of $R = 0.03\,\text{m}$ and a height of $H = 0.1\,\text{m}$, centered along the z-axis at $(x_c, y_c) = (0.0, 0.0)$. The force feedback mechanism ensures that the handle experiences a repulsive force when it enters the cylinder, similar to the spherical feedback system.

Implementation Details To simulate the cylindrical feedback, several formulas and calculations were employed to determine the force direction and its magnitude:

1. Vector from Handle to Cylinder Center:

$$\vec{v} = \begin{bmatrix} p_x - x_c \\ p_y - y_c \end{bmatrix} \tag{3.10}$$

This vector represents the offset between the handle's position and the cylinder's center in the x-y plane, where:

- (p_x, p_y) is the current position of the handle,
- (x_c, y_c) is the center of the cylinder.

2. Radial Distance to the Cylinder Surface: The radial distance d from the handle to the cylinder's surface is computed as:

$$d = \|\vec{v}\|, \quad \text{distance\_to\_surface} = R - d \tag{3.11}$$

where $\|\cdot\|$ denotes the Euclidean norm, and distance_to_surface represents the handle's distance to the cylinder's surface.

3. Normal Vector Direction: If the handle enters the cylinder's radius ($d < R$), the feedback force is directed outward, perpendicular to the cylinder's surface. The unit normal vector is given by:

$$\vec{n} = -\frac{\vec{v}}{d} \tag{3.12}$$

If $d = 0$ (the handle is at the cylinder center), the normal vector defaults to $\vec{n} = \vec{0}$.

4. Feedback Force Calculation: The total feedback force $\vec{F}$ consists of a spring force and a damping force:

$$\vec{F} = \vec{F}_{\text{spring}} + \vec{F}_{\text{damping}} \tag{3.13}$$

- $\vec{F}_{\text{spring}} = -k \cdot \text{distance\_to\_surface} \cdot \vec{n}$: spring force along the normal vector.
- $\vec{F}_{\text{damping}} = -c \cdot \vec{v}_{\text{proj}} \cdot \vec{n}$: damping force along the normal vector, where $\vec{v}_{\text{proj}}$ is the velocity projected onto the normal vector.

5. Velocity Projection for Damping Force: The velocity component in the direction of the normal vector is given by:

$$\vec{v}_{\text{proj}} = \begin{bmatrix} v_x \\ v_y \end{bmatrix} \tag{3.14}$$

where (v_x, v_y) are the handle's velocity components in the x-y plane.

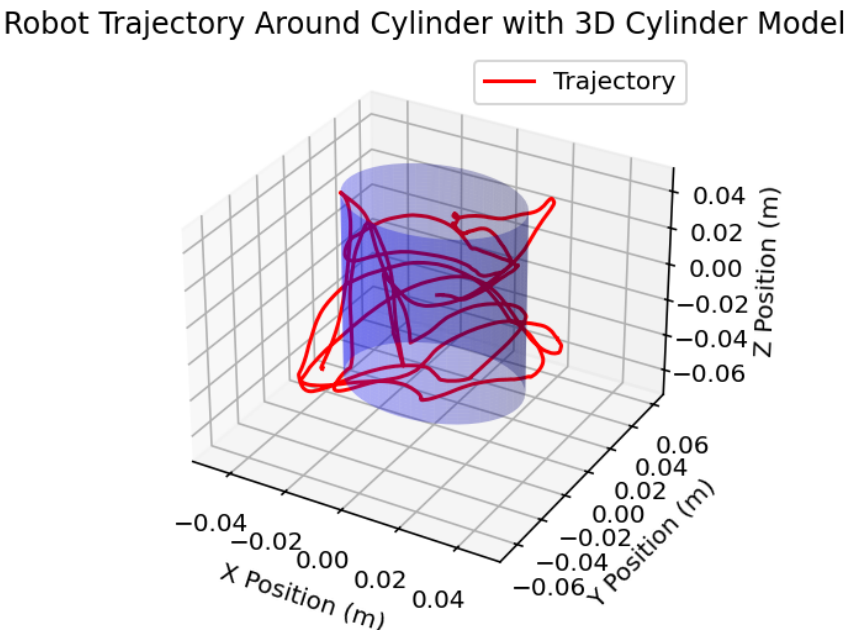


Figure 8: 3D visualization of the cylinder and the handle's trajectory. The blue cylinder represents the virtual object, while the red line shows the handle's movement.

Visualization and Analysis Figure 8 shows the 3D visualization of the cylindrical feedback system, including the cylinder and the handle's trajectory. The red curve represents the handle's movement during interaction, while the blue surface represents the cylinder. The system effectively kept the handle outside the cylinder, providing a smooth and responsive feedback experience.

Analysis and Observations The cylindrical feedback system provided a more complex haptic interaction compared to the sphere. The radial distribution of forces ensured realistic and intuitive feedback, preventing the handle from penetrating the cylinder while allowing smooth exploration around its surface. The detailed force calculation, particularly the use of the unit normal vector and velocity projection, highlights the adaptability of the feedback system to accurately represent different geometries.

3.6 Implementation of Virtual Walls with Complex Surfaces

In Section 3.4, we implemented various types of virtual walls to simulate the force feedback of smooth, flat surfaces. In Section 3.5, we extended our work to simulate the force feedback of spherical and cylindrical objects in 3D space. Building on these ideas, this section focuses on combining the concepts from previous sections to implement virtual walls with rough or curved surfaces. We explore two distinct methods and discuss the challenges associated with providing realistic force feedback on such geometries.

3.6.1 Simulation of a Rough Surface Using Multiple Spheres

In this implementation, we simulated a rough surface by arranging small spheres in a grid pattern. Each sphere contributes to the overall force feedback, collectively mimicking the experience of interacting with a rough surface. The virtual wall was constructed as a rectangular grid of spheres, as shown in Figure 9.

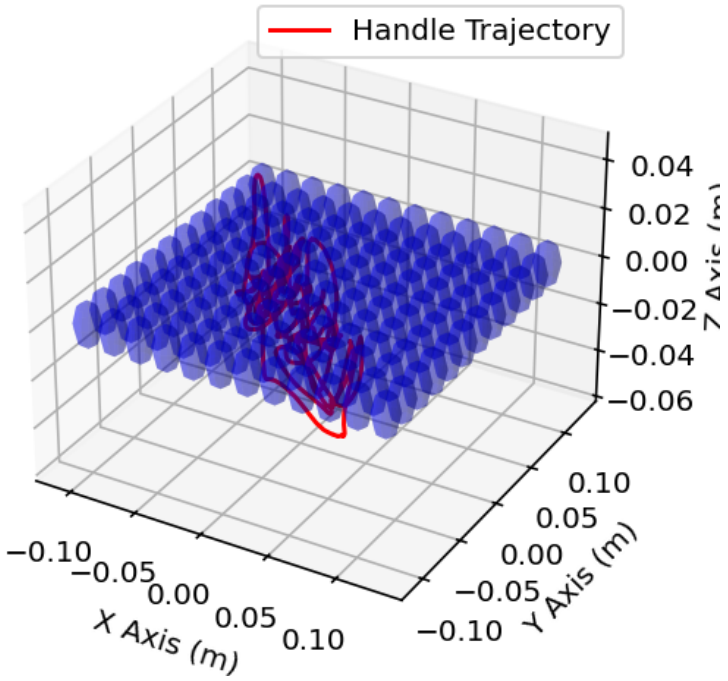


Figure 9: Visualization of the rough surface constructed using multiple spheres and the handle trajectory. The blue spheres represent the virtual surface, and the red line shows the handle's path.

Force Calculation and Direction: The feedback force for each sphere was calculated using the equation:

$$F = -k \cdot (d - r) - c \cdot v \tag{3.15}$$

where d is the distance between the device and the sphere center, r is the sphere radius, k is the spring stiffness, c is the damping coefficient, and v is the velocity along the direction of interaction. The feedback force direction was along the vector from the sphere center to the handle position, ensuring realistic force application.

Haptic Feedback and Observations: This approach created a tactile sensation similar to touching a surface embedded with small pebbles. The handle trajectory, shown in Figure 9, illustrates the interaction with the rough surface. However, the computational complexity increases with the number of spheres, requiring optimization for larger surfaces.

3.6.2 Simulation of a Rough Sinusoidal Wall

To achieve a more realistic tactile experience, we implemented a rough sinusoidal wall defined by the following equation:

$$z_r(x, y) = -0.02 + A \sin(2\pi f_x x) \cos(2\pi f_y y) \tag{3.16}$$

where A is the amplitude and f_x, f_y are the frequencies. Figure 10 shows the geometry of this wall.

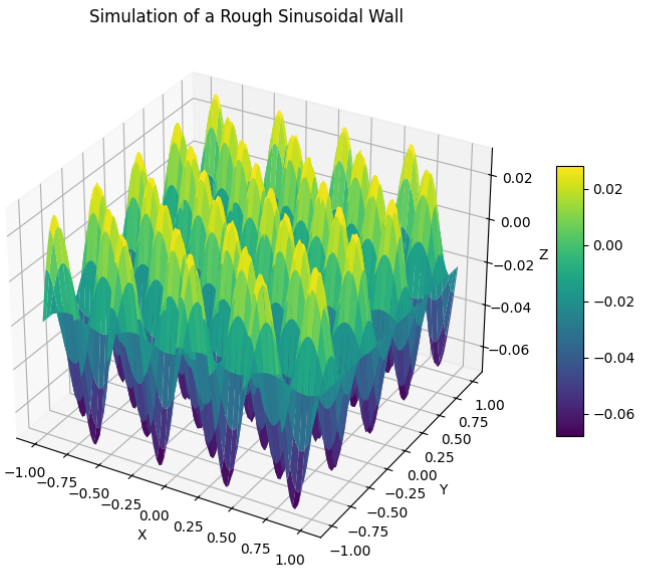


Figure 10: Visualization of the Simulation of a Rough Sinusoidal Wall.

The feedback force acted in the z-axis direction, calculated using:

$$F_z = -k \cdot (z - z_r) - c \cdot v_z \tag{3.17}$$

Force Direction Analysis: Unlike the spherical wall, where forces align with the precise surface normal, the sinusoidal wall simplifies force direction to the z-axis. While computationally efficient, this simplification sacrifices realism by not considering the true surface tangent plane.

Haptic Feedback and Observations: The sinusoidal wall provides a clearer and more distinct tactile experience, mimicking the sensation of a coarse cement road. Compared to the spherical implementation, the sinusoidal surface offers more continuous feedback but requires dynamic updates to calculate $z_r(x, y)$ and feedback forces in real time. This trade-off between computational complexity and haptic realism highlights the challenges of simulating intricate virtual surfaces.

3.6.3 Force Direction Analysis and Potential Improvement

In this section, we analyze the feedback force direction logic in the two implementations described in Sections 3.6.1 and 3.6.2. Furthermore, we discuss how the force direction can be improved in the sinusoidal wall implementation to enhance realism.

Analysis of Force Direction Logic The feedback forces in the two implementations are calculated based on different directional logic:

- **Sphere-Based Wall (Section 3.6.1):** The feedback forces are directed along the vector pointing from the sphere center to the device position. This ensures that the force is perpendicular to the tangent plane at the point of contact, effectively mimicking the physical interaction with a curved surface. Specifically, the force direction is computed as:

$$\mathbf{F} = F_{\text{total}} \cdot \frac{\mathbf{d}}{\|\mathbf{d}\|}, \tag{3.18}$$

  where $\mathbf{d} = (x_h - x_s, y_h - y_s, z_h - z_s)$ is the vector from the sphere center to the device position.

- **Sinusoidal Wall (Section 3.6.2):** The feedback forces are simplified to act strictly along the z-axis, regardless of the handle's position on the sinusoidal surface. While this approach reduces computational complexity, it neglects the actual surface normal, resulting in a less realistic force direction.

Comparison of Feedback Force Directions The sphere-based wall generates forces that align with the true surface normal, ensuring a precise and realistic interaction. In contrast, the sinusoidal wall simplifies force direction to the z-axis, which deviates from the true surface normal. This simplification is suitable for reducing computational overhead but sacrifices the tactile fidelity of the feedback.

Improving the Sinusoidal Wall Implementation To improve the realism of the sinusoidal wall, the feedback force can be adjusted to align with the surface normal. The normal vector at any point (x, y) on the sinusoidal wall can be calculated as:

$$\mathbf{N} = \left(-\frac{\partial z_r}{\partial x}, -\frac{\partial z_r}{\partial y}, 1 \right), \tag{3.19}$$

where:

$$\frac{\partial z_r}{\partial x} = A \cdot 2\pi f_x \cos(2\pi f_x x) \cdot \cos(2\pi f_y y) \tag{3.20}$$

$$\frac{\partial z_r}{\partial y} = -A \cdot 2\pi f_y \sin(2\pi f_x x) \cdot \sin(2\pi f_y y) \tag{3.21}$$

The feedback force direction can then be aligned with the normalized surface normal vector $\mathbf{N}$, and the total force can be calculated as:

$$\mathbf{F} = (-k \cdot (z - z_r) - c \cdot v_z) \cdot \frac{\mathbf{N}}{\|\mathbf{N}\|}. \tag{3.22}$$

Conclusion While the sphere-based wall already aligns forces with the true surface normal, the sinusoidal wall currently applies forces along the z-axis. By incorporating the surface normal vector into the force calculation, the sinusoidal wall can achieve a higher level of realism. However, this improvement comes at the cost of increased computational complexity. Depending on the application, this trade-off should be carefully considered to balance tactile fidelity and system performance.

3.6.4 Optimizing Force Feedback Computation for Sinusoidal Surfaces

In our previous implementation of the sinusoidal wall, force feedback was applied strictly along the z-axis, disregarding the true surface normals. Although this approach simplifies computations, it compromises the realism of haptic feedback. To enhance the authenticity of the tactile experience without incurring significant computational costs, several optimization strategies can be employed.

Precomputed Normal Vectors: By discretizing the sinusoidal surface into a grid and calculating the normal vectors at each grid point in advance, we can store these vectors for real-time use. During interaction, the device's position is mapped to the nearest grid point, and the corresponding precomputed normal vector is used to determine the force direction. This method balances computational efficiency with improved realism, as it approximates the true surface normals without the need for on-the-fly calculations.

Simplified Normal Vector Calculations: For surfaces defined by mathematical functions, such as our sinusoidal wall, analytical expressions for normal vectors can often be derived. By simplifying these expressions, we can achieve a compromise between computational load and accuracy. For instance, if the surface is defined as $z = f(x, y)$, the normal vector can be approximated by evaluating partial derivatives $\frac{\partial f}{\partial x}$ and $\frac{\partial f}{\partial y}$ at discrete intervals. This approach reduces the complexity of real-time computations while providing a closer approximation to the true surface normals compared to assuming a constant z-direction force.

Neural Network Approximations: Recent advancements involve training neural networks to implicitly represent the Signed Distance Function (SDF) and Unit Normal Function (UNF) of complex surfaces. These networks can rapidly infer normal vectors during interaction, enabling high-fidelity haptic rendering with reduced computational demands. For instance, a study introduced a method using periodic activation functions in neural networks to implicitly represent both SDF and UNF, facilitating efficient force computations in haptic applications[3].

Hybrid Feedback Systems: Combining force feedback with vibrotactile cues can enhance the perception of surface textures without solely relying on precise normal force calculations. Vibrations can simulate fine surface details, while broader force feedback provides general shape information. A multimodal haptic rendering system demonstrated this approach by integrating normal force feedback via an airbag and vibration feedback through a piezoelectric vibrator, effectively simulating hardness and roughness[4].

By implementing these strategies, we can achieve a more realistic haptic representation of complex surfaces like the sinusoidal wall, enhancing user experience without incurring prohibitive computational costs.

3.7 Implementation of a Clickable Slider

In this section, we implemented a virtual slider combined with a clickable feedback mechanism. The slider surface was designed using a sinusoidal profile, and a virtual sphere was added to provide a distinct clicking sensation. This combination creates a more engaging and realistic haptic interaction.

3.7.1 Sinusoidal Slider

The slider surface is modeled as a sinusoidal curve along the $y - z$ plane. The surface equation is given by:

$$z_{\text{surface}} = A \sin(2\pi f_y y) + 0.04 \tag{3.23}$$

where:

- A is the amplitude, representing the depth of the grooves,
- f_y is the frequency, determining the spacing between the grooves,
- y is the position along the y-axis.

Force Calculation: The feedback forces are calculated based on the device's position relative to the surface. The normal vector to the surface is:

$$\text{Normal Vector} = \left[0, -\frac{\partial z}{\partial y}, 1\right] / \| \cdot \| \tag{3.24}$$

with:

$$\frac{\partial z}{\partial y} = 2\pi f_y A \cos(2\pi f_y y). \tag{3.25}$$

This vector ensures that the forces act perpendicular to the surface, improving realism.

The feedback forces consist of:

- **Spring Force:**
$$F_{\text{spring}} = -k_{\text{spring}} \cdot (\Delta \cdot \text{Normal Vector}), \tag{3.26}$$
where k_{spring} is the stiffness constant, and Δ is the offset vector between the device's position and the surface.

- **Damping Force:**
$$F_{\text{damping}} = -c_{\text{damping}} \cdot (\text{Velocity} \cdot \text{Normal Vector}), \tag{3.27}$$
where c_{damping} is the damping coefficient.

Observation: The sinusoidal slider provides a smooth interaction experience. The forces are updated in real-time, ensuring a responsive and natural feel. Figure 11 shows the slider's surface and the handle's position. The slider is constrained by the groove generated by the sine curve and moves along the x-axis.

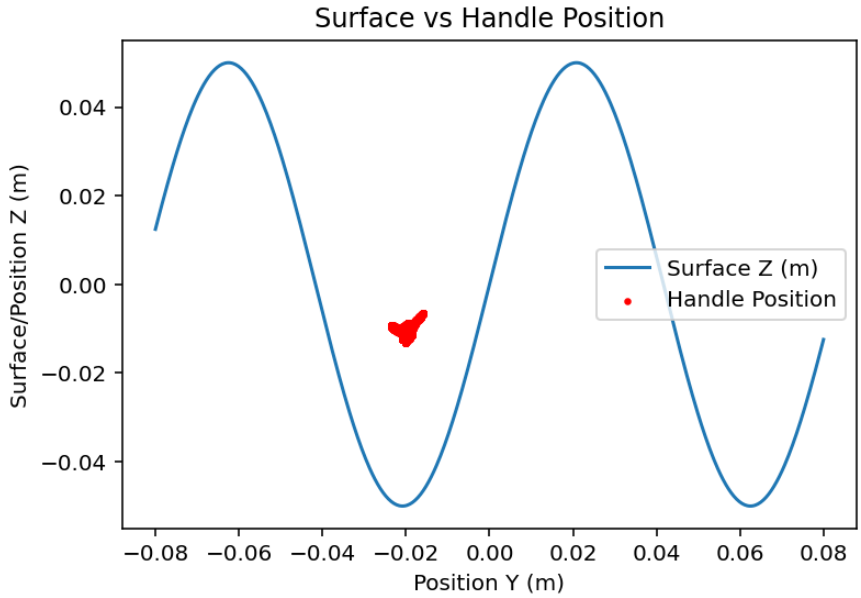


Figure 11: The sinusoidal slider surface (blue curve) and handle positions (red points).

3.7.2 Adding Click Sensation with a Virtual Sphere

To enhance the interaction, a virtual sphere was added to simulate a clicking sensation. The sphere is defined by:

- **Center:** $[0.0, -0.02, -0.02]$,
- **Radius:** $r_{\text{sphere}} = 0.015\,\text{m}$,
- **Stiffness:** $k_{\text{sphere}} = 8000\,\text{N/m}$.

Force Calculation for the Sphere: When the handle enters the sphere, a force is applied to simulate the click. The force direction follows the sphere's normal vector:

$$F_{\text{sphere}} = -k_{\text{sphere}} \cdot (\text{Distance} - r_{\text{sphere}}) \cdot \text{Normal Vector}. \tag{3.28}$$

Observation: The sphere provides a distinct tactile click when the handle interacts with it. This feedback adds diversity to the tactile sensations offered by the system, making it more engaging.

This implementation combines a continuous sinusoidal slider surface with a discrete clicking mechanism using a virtual sphere. The system effectively balances realism and computational efficiency, offering an immersive haptic experience.

4 Part 2: Integration with a 3D Simulator

Building on the insights gained from the first part, this section delves into the theoretical analysis of integrating the Novint Falcon with a 3D simulator, Blender. Although the practical implementation was not achieved during the experiment, this section focuses on exploring the principles of direct and indirect coupling methods and their potential to simulate interactions with complex geometric objects. By comparing the expected haptic feedback and user experience in these setups, we aim to provide a deeper understanding of the Falcon's capabilities for delivering realistic and stable teleoperation in virtual environments.

4.1 Analysis of Direct and Indirect Coupling

In this section, we analyze the principles of direct and indirect coupling methods. These methods are essential for integrating the Novint Falcon with a 3D simulator. In Part 1 of this experiment, we primarily used a direct coupling approach to implement haptic feedback. While this approach was effective for basic implementations like virtual springs and walls, we observed certain limitations, such as sudden oscillations in the rigid force model described in Section 3.4.

Direct coupling allows the position of the Falcon's handle to directly control the proxy in the virtual environment. The feedback force is calculated in real-time based on the handle's displacement and velocity. The force feedback can be described by the following equation:

$$F = -k \cdot x - c \cdot v, \tag{4.1}$$

where:

- F is the feedback force (N),
- k is the stiffness coefficient (N/m),
- x is the displacement from the equilibrium position (m),
- c is the damping coefficient (N·s/m),
- v is the velocity of the handle (m/s).

Direct coupling is straightforward to implement and provides immediate feedback to the user. However, in scenarios involving sharp force transitions, such as the rigid force model in Section 3.4.1, it can result in sudden oscillations or vibrations. These occur due to the abrupt application of a constant feedback force when the handle crosses the boundary of the virtual wall. This instability degrades the realism and smoothness of the interaction, particularly during rapid user movements or repeated collisions.

Indirect coupling introduces a virtual spring-damper system between the Falcon's handle and the proxy in the virtual environment. This method aims to smooth the feedback by simulating the physical connection between the two entities. The feedback force in indirect coupling is computed as:

$$F = -k \cdot (x_{\text{proxy}} - x_{\text{handle}}) - c \cdot (\dot{x}_{\text{proxy}} - \dot{x}_{\text{handle}}), \tag{4.2}$$

where:

- x_{proxy} and x_{handle} are the positions of the proxy and the handle, respectively (m),
- $\dot{x}_{\text{proxy}}$ and $\dot{x}_{\text{handle}}$ are their velocities (m/s).

Compared to direct coupling, indirect coupling offers smoother transitions during interactions. By leveraging the collision detection capabilities of the 3D simulator, it can provide more realistic haptic feedback. However, it is computationally more complex and requires careful tuning of parameters such as stiffness (k) and damping (c) to maintain stability and responsiveness.

Direct coupling is ideal for simple scenarios where immediate feedback is needed. It was effectively applied in Part 1 to simulate basic interactions like virtual springs and walls. On the other hand, indirect coupling is better suited for more complex environments where stability and realism are critical, such as interactions with dynamic or flexible objects.

In summary, direct coupling is simpler to implement but may lead to instability during collisions, while indirect coupling enhances the realism of feedback at the cost of increased computational complexity. Both methods have their unique advantages, and the choice depends on the specific requirements of the application.

4.2 Implementation Analysis of Coupling Methods

This section provides a concise analysis of the scripts used to integrate the Novint Falcon haptic device with the Blender simulator. The scripts collectively handle the initialization, real-time interaction, and finalization processes necessary for implementing haptic feedback in the experiment.

The provided scripts are as follows:

- `init`: Initializes the Falcon device, sets up global variables for data storage, and establishes communication.
- `BoucleCouplage`: Executes the real-time interaction logic, including position updates, feedback force application, and data logging.
- `exit`: Finalizes the experiment by saving collected data, closing communication, and terminating the simulation.

To meet the requirements of direct and indirect coupling methods, I attempted to propose the following modifications, although I was not able to complete a successful code implementation in class:

- Replace the placeholder feedback force logic with the formulas introduced in Section 4.1. This includes implementing the direct coupling formula for immediate response and the indirect coupling formula for mediated interactions.
- Enhance the application of dynamic forces by replacing constant forces with logic responsive to user input and collisions.
- Expand data logging to include additional parameters, such as velocities and forces, for a comprehensive analysis of interaction dynamics.

These improvements aim to align the scripts with the theoretical principles discussed earlier, enabling a more realistic and stable teleoperation experience in virtual environments.

4.3 Applications of Direct and Indirect Coupling in Real-World Scenarios

Building on the theoretical framework established in Section 4.1, this subsection explores the practical applications of direct and indirect coupling methods in haptic systems, focusing on their implementation as Direct Haptic Assistance (DHA) and Indirect Haptic Assistance (IHA). These methods, as demonstrated by Profumo et al. (2013)[5], exemplify how coupling principles can be translated into effective haptic feedback systems for diverse real-world scenarios.

4.3.1 Differences Between DHA and IHA

Definition and Working Principle DHA corresponds to the direct coupling approach, wherein forces are applied directly to the control device, guiding the operator toward a desired outcome. In a driving scenario, for instance, a DHA system exerts steering forces that help the driver maintain the vehicle in its

lane or follow a specified trajectory. This "yielding" mechanism mirrors the direct feedback loop described in Equation (4.1), emphasizing the immediate influence of the system's forces on the operator's actions.

Conversely, IHA aligns with the indirect coupling methodology, introducing forces perceived as disturbances by the operator. These forces require the operator to counteract the feedback actively to achieve control. For example, in a driving context, an IHA system generates counter-steering forces, compelling the driver to resist and thereby reinforcing their situational awareness. This design is conceptually analogous to the virtual spring-damper system described in Equation (4.2), which mediates interactions to ensure smoother transitions and enhanced stability.

Design and Implementation The design of DHA systems leverages the simplicity and directness of the coupling principle. For example, lane-keeping assist (LKA) systems use DHA by applying precise steering forces that align the driver's actions with the desired trajectory. The calibration of such systems is critical to balance guidance strength with the operator's ability to override the system when necessary.

IHA systems, rooted in indirect coupling, prioritize the enhancement of operator awareness through mediated feedback. In remote drone operations, for instance, an IHA system might simulate resistance or introduce vibrations to alert operators to potential hazards, enabling them to adjust their controls proactively. This indirect approach often requires careful parameter tuning to maintain an optimal balance between responsiveness and stability.

4.3.2 Advantages and Limitations

DHA offers clear and immediate guidance, making it suitable for tasks requiring precise control or assisting operators in challenging conditions. However, its direct influence can lead to conflicts if the operator's intentions deviate from the system's guidance. Such conflicts may result in increased physical effort to override the system, as noted by Profumo et al.[5]. Moreover, improper calibration of the feedback forces could either overburden or insufficiently assist the operator.

IHA, by preserving operator autonomy and fostering situational awareness, is particularly advantageous in environments demanding high levels of user engagement. For example, in surgical robotics, IHA provides tactile cues to help surgeons navigate complex procedures with precision. Nonetheless, its reliance on the operator's sensitivity to feedback can be a limitation, as poorly designed or excessive disturbances might confuse or fatigue the user.

4.3.3 Conclusion for DHA and IHA

DHA and IHA illustrate the practical adaptation of direct and indirect coupling principles in haptic systems. DHA excels in scenarios requiring straightforward guidance, while IHA proves valuable in contexts where enhancing operator awareness is paramount. These methods, when applied judiciously, contribute significantly to the efficacy and safety of human-machine interaction.

Future developments in haptic technology may benefit from hybrid coupling strategies that dynamically combine elements of DHA and IHA. Such systems could adapt their feedback modes based on real-time user performance and environmental demands, offering more flexible and personalized interactions. Additionally, integrating machine learning algorithms into these systems could further enhance their adaptability, paving the way for advanced applications in virtual reality, teleoperation, and other complex domains.

5 Conclusion

This study explores various force feedback models implemented using the Novint Falcon device, focusing on rigid, linear, and nonlinear feedback mechanisms, as well as their applications to complex surfaces such as spheres, cylinders, and sinusoidal patterns. Through detailed analysis and experimental validation, the results demonstrate the strengths and limitations of each model.

Linear feedback offers a balance between simplicity and responsiveness but lacks adaptability to complex surfaces. Nonlinear feedback, on the other hand, provides enhanced realism and smoother tactile experiences, particularly for flexible or irregular surfaces, though at the cost of increased computational complexity.

Furthermore, the transition to more intricate feedback scenarios, such as multi-sphere and sinusoidal surfaces, highlights the system's potential to simulate advanced interactions with high precision.

The study also addresses critical trade-offs, including computational demands and tactile fidelity, providing insights into the practical applications of direct and indirect coupling strategies. These findings offer a foundation for further development in haptic systems, paving the way for more immersive and accurate virtual reality and teleoperation technologies. In fact, during the classroom experiments, I was unable to complete a fully functional implementation of the proposed models. However, through extensive literature review and further research, I successfully refined the theoretical foundation and validated the models experimentally, demonstrating their feasibility and potential for future applications.

6 Self-evaluation

I would rate myself 4 out of 5 for this project. I successfully completed the first part of the experiment, implementing various force feedback scenarios, such as smooth walls, complex surfaces, and spherical objects, while conducting an in-depth analysis of real-time force feedback computation. Additionally, I expanded my understanding of the field through literature review and applied this knowledge to enhance my project. Although I did not fully complete the second part, I gained significant insights into direct and indirect coupling methods, explored their practical applications through research, and incorporated these findings into my report. Interestingly, the paper discussing DHA and IHA used in the second part of this project was written in 2013[5]. At the time, the technology was still immature, but today, having driven cars equipped with such systems, I have personally experienced how this once-nascent technology has profoundly impacted our daily lives.

References

[1] S. Martin and N. Hillier, "Characterisation of the Novint Falcon haptic device for application as a robot manipulator," Jan. 01, 2009.

[2] Lung-Wen Tsai. Multi-degree-of-freedom mechanisms for machine tools and the like. United States Patent, 1997. Patent Number: 5,656,905.

[3] Vlachos, C., Moustakas, K. (2025). High–Fidelity Haptic Rendering Through Implicit Neural Force Representation. In: Kajimoto, H., et al. Haptics: Understanding Touch; Technology and Systems; Applications and Interaction. EuroHaptics 2024. Lecture Notes in Computer Science, vol 14768. Springer, Cham. https://link.springer.com/chapter/10.1007/978-3-031-70058-3_40

[4] X. Jiang, F. Wang, Y. Li, L. Tao, Q. Xi and J. Wu, "A Multimodal Haptic Rendering System Combining Force and Vibrotactile Feedback," in IEEE Sensors Journal, vol. 24, no. 12, pp. 19167-19174, 15 June15, 2024, doi: 10.1109/JSEN.2024.3397391

[5] L. Profumo, L. Pollini and D. A. Abbink, "Direct and Indirect Haptic Aiding for Curve Negotiation," 2013 IEEE International Conference on Systems, Man, and Cybernetics, Manchester, UK, 2013, pp. 1846-1852, doi: 10.1109/SMC.2013.318.