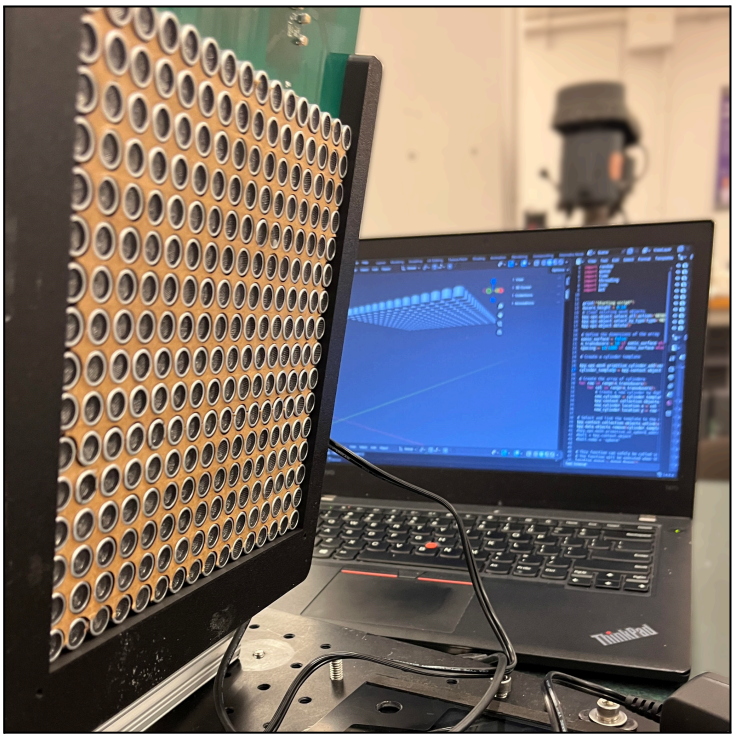
# Ultrasonic Levitation for Holographic Imaging



Paige Hall
Zavary Koehn
Casey Krombein
Howard Li

Project Sponsor:
Engineering Physics Project Lab

Project # 2505
ENPH 459
Engineering Physics Project Lab
The University of British Columbia
April 8th, 2025

# Executive Summary

This report presents the results of our investigation into the operation and behavior of the ultrasonic levitation system developed in Project # 2365 - Ultrasonic Holography. Our two primary goals were to understand the current system implementation and the physics behind it, as well as characterize the system by measuring key performance parameters. The current system consists of a 16x16 grid of transducers soldered to a custom PCB that uses two FPGAs to generate acoustic waves, as well as a phase solver algorithm implemented in Rust, position input scripts in Python, and a 3D simulation written in Blender. This implementation allows us to create and control a pressure field such that a small, light-weight particle can be levitated at a specific location, called a trap. Our investigation of this system involved performing performance limits with respect to key parameters such as the update rate, measuring the resulting speed and acceleration. We measured a maximum linear velocity of 67.87 cm/s and maximum acceleration of 3220 cm/s^2.

We also observed that the particle behaves similarly to a harmonic oscillator when entering a stationary trap with some velocity. This suggests that trap stability or stiffness can be assessed by measuring the effective *k* (effective spring constant) value of a trap. As trap stiffness is directly related to trap quality, measuring the *k* will provide a better understanding of the underlying pressure field under different conditions. These observations and results have informed our recommendations for future project work and improvements. Ultimately, we achieved our goal of understanding the system and the physics. Further testing and improvements are required to optimize the system and fully characterize its behavior, and we have outlined multiple recommendations for future work.

# Table of Contents

# Table of Figures

# Introduction

This project aims to develop the technology to create holograms. Holograms are 3D representations of an image that appear to float in mid-air. Holograms are often depicted in popular media as a means of communication, such as in Star Wars (Figure 1), but depending on the implementation, there are many other valuable applications.

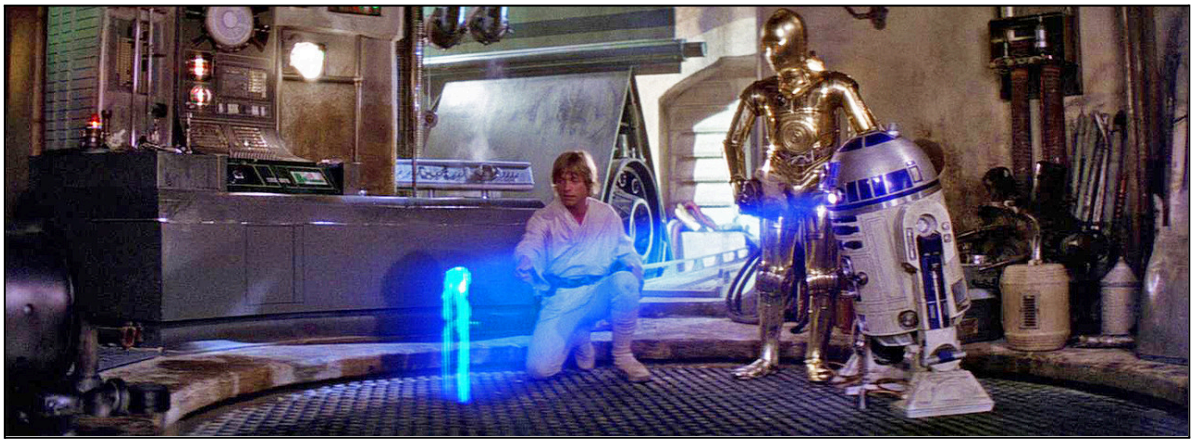


*Figure 1: Princess Leia's hologram from "Star Wars: A New Hope" [1].*

Our implementation relies on ultrasonic levitation, which involves levitating particles using ultrasonic (>20 kHz) acoustic waves. Ultrasonic levitation has applications in manufacturing and assembly [2], mid-air chemical mixing (contactless fluid manipulation) [3], contactless nanoscale manipulation for biological experiments [4] [5], and mid-air displays [6] [7].

Our project tackles mid-air displays. There are many ways to generate 3D images using ultrasonic levitation. Some examples are:

- Levitating a small rotating screen and project an image onto it (Figure 2a) [6],
- Levitating a collection of particles to form shapes, and (Figure 2b) [8],
- Levitating a single particle, with sufficient speed for the particle to form a blurred outline of its path (Figure 2c) [7] [9].

Our project aims to implement this last method at the undergraduate level, as opposed to our cited research, which was undertaken by graduate/professional researchers. Audio, lighting, and haptics can be implemented in conjunction with the 3D image to enhance the sensory experience [7].

When a small object moves at a high speed, motion blur causes the formation of blurred lines in the direction of motion. This is because the human brain can not process individual images faster than 10-12 frames per second (FPS), and is, for example, why the standard cinematic frame rate is 24 FPS [10]. If a particle traces a shape faster than 10-12 times per second, the object will begin to appear as a blurred line that follows the direction of motion and traces out the desired path.

This is known as the persistence of vision optical illusion, and it is the method used in Figure 2c. To produce this effect, a system needs the ability to levitate a small particle and move it

around very quickly. If we can achieve this, we can use ultrasonic levitation to produce holograms.

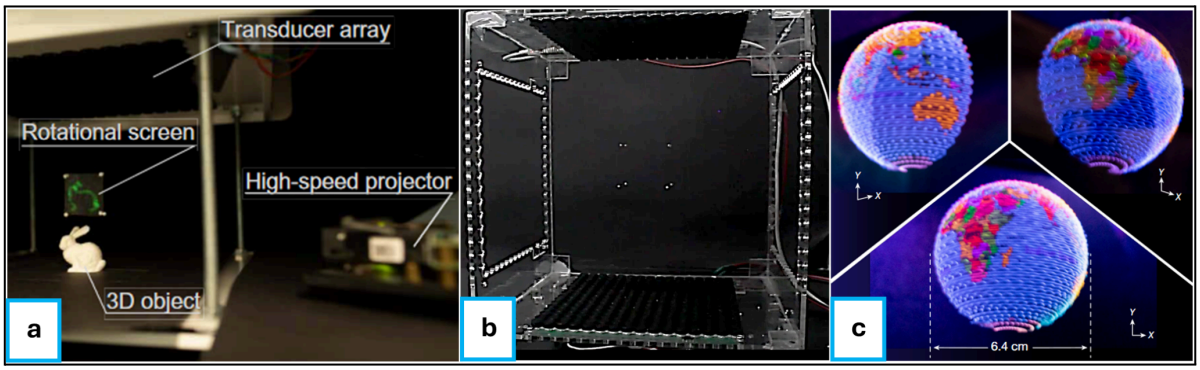



*Figure 2: Examples of existing research on ultrasonic levitation.* ***a)*** *A hologram levitating via a small rotating screen [6].* ***b)*** *A hologram levitating via a collection of particles [8].* ***c)*** *A hologram levitating via a fast moving particle (persistence of vision) [7].*

We are continuing the work of a previous team in this project. The members of this team were Andrew Chen, Quinn Ferbers, Kevin Lin, and James Seto. They designed and fabricated the current PCB (Figure 3) and also implemented various algorithms and simulations in code. Their work is accessible via GitHub and is summarized in their final report [11]. Their setup is able to levitate particles, but cannot produce holograms.

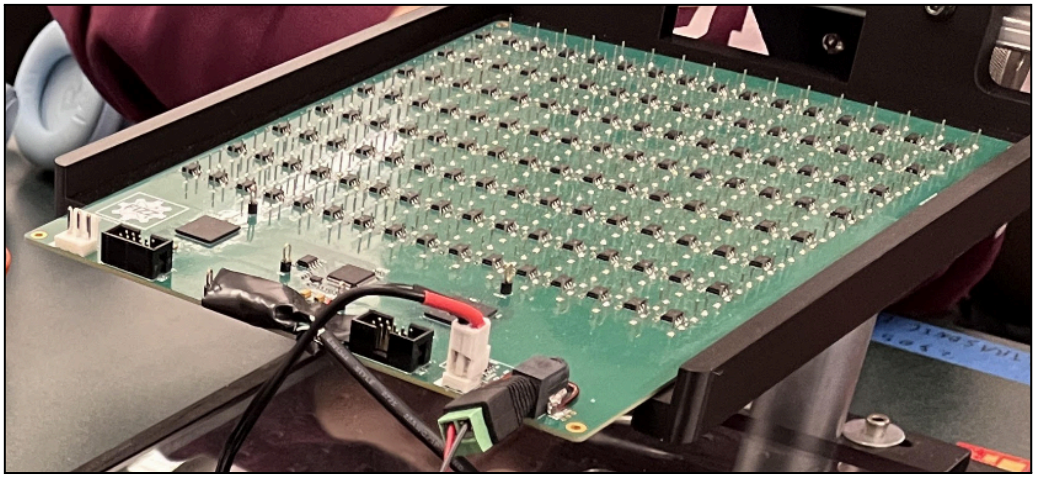


*Figure 3: System PCB, mounted on a 3D-printed stand. The PCB for our system was designed by the previous team. For more details concerning the stand, see Appendix A.*

The scope of this project involves three main categories of activity:

1. Understanding the existing system implementation.
2. Characterizing the system.
3. Identifying and implementing upgrades to improve system performance so that we can create holograms.

Understanding the current system involves learning the hardware, software, and underlying physics principles. Characterization involves testing the current system to quantify its

capabilities, identify its weak spots, and determine how we can improve its performance. Our understanding of the system and its behaviour will inform the system upgrades we choose to implement, which we discuss further in Recommendations.

Our goals for this semester were to tackle the first two categories of activity. The testing and results, as well as the fundamental theory behind ultrasonic levitation, are discussed below.

# Discussion

## Fundamental Physics

Ultrasonic levitation involves filling a region of space with sound waves at ultrasonic frequencies to induce pressure gradients capable of exerting forces on objects within said region. By modifying the relative phases of these sound waves, we can control the pressure gradients, and by extension, levitate particles and generate holograms.

Small speakers called transducers are used to produce the sound waves. Our transducers emit ultrasonic waves at 40kHz [11]. They are placed in a grid/array formation (Figure 4a) positioned above a reflective plate (Figure 4b), such that the emitted waves travel downwards and reflect upwards off the plate.

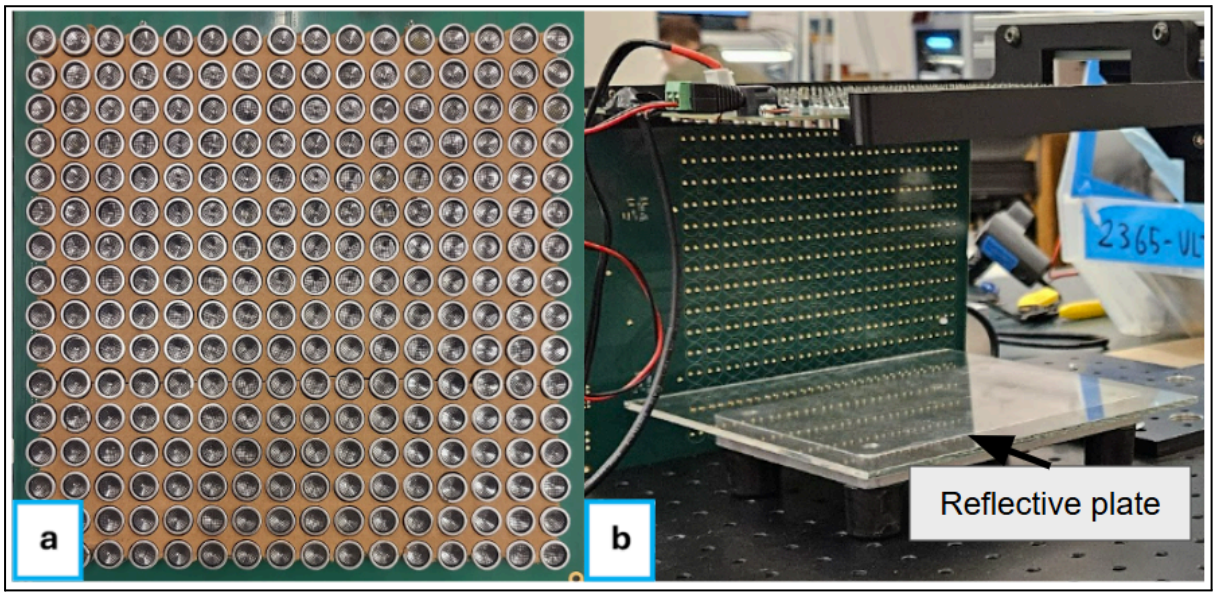



*Figure 4: Grid of transducers and reflective plate.* ***a)*** *The 16 x 16 grid of transducers which we use to generate a pressure field between the PCB and the reflective plate shown in* ***b)****.*

The interference from reflected waves creates a standing wave pattern in the space between the reflective plate and the transducer array. Note also that a flat grid is not the only possible configuration for acoustic levitation (Figure 5).

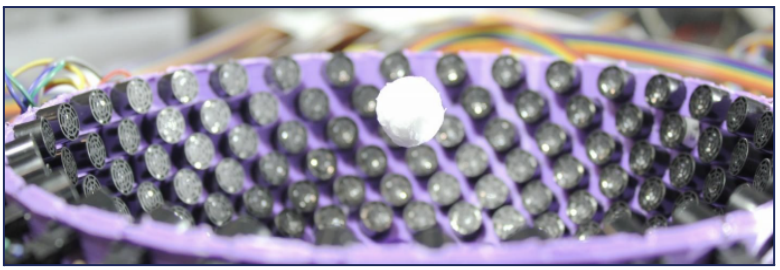


*Figure 5: Transducers arranged in a hemispherical configuration [12].*

The standing wave effect could also be created by replacing the reflective plate with an identical grid of transducers facing upwards at twice the distance between the transducer grid and the reflective plate (Figure 6).

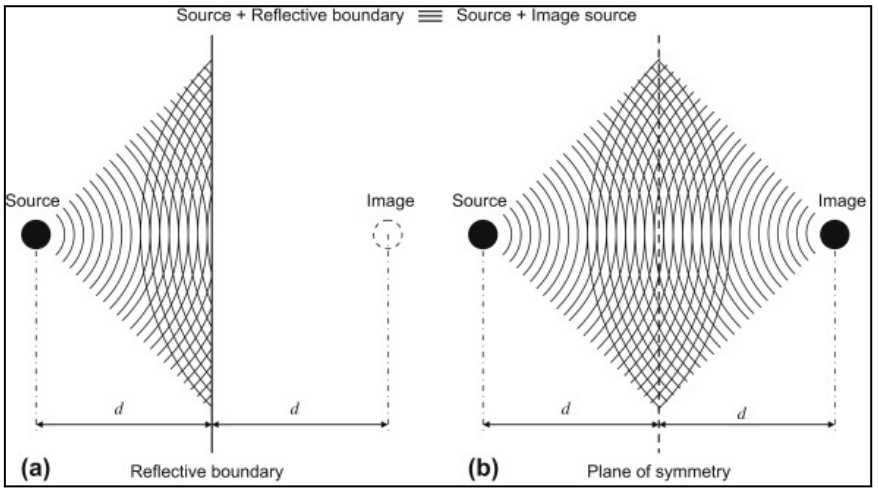



*Figure 6: Substituting a reflective boundary with a reflected source.* ***a)*** *A source and reflective plate at a separation d is equivalent to* ***b)*** *Two identical sources at a separation 2d. In our case, the source and image are transducer grids.*

We have retained the setup used by the previous team (Figure 4). With this configuration, the last step needed to levitate particles is a careful selection of the relative phases of each transducer's emitted wave. In order to levitate a particle at a particular point P, we need to ensure that there is a local minimum in the acoustic field's potential at that point. This potential is known as the Gor'kov potential, and it describes the pressure potential within an acoustic field [11]. This involves selecting the phase offset of each transducer such that all emitted sound waves interfere to create a pressure node at that location. Once this is achieved, point P is called a trap because it will trap an appropriately sized particle that is placed in the field nearby. Here, appropriately sized particles have a diameter on the order of the wavelength λ / 4 of the emitted waves (λ ≈ 8.575 mm for our transducers). There are several ways to determine the relative phases required to produce a trap at point P, which is the role of a transducer phase offset solver (henceforth called a *solver*). The details of our algorithm and why it was selected by the previous team are discussed in the transducer phase offset section. The underlying physics of trap formation will be outlined below.

When a sound wave contacts a particle, the particle experiences a force called the acoustic radiation force (ARF). The ARF is equal to the negative gradient of the Gor'kov potential and it pushes the particle towards a region of low potential. With acoustic standing waves, the low potential regions are located at the nodes. This occurs due to the non-linearity of the sinusoidal pressure oscillations, which results in varying force magnitudes at different positions in the wave.

When the particle is at position A, it experiences a force with magnitude proportional to the mean squared pressure amplitude (Figure 7) [11]. The particle is pushed in the direction opposite the pressure field gradient, as the force points from higher pressure to lower pressure, and ends up at point B. When the pressure wave is at the opposite side of its oscillation range, the ARF on the particle points the opposite way. The particle is now at a region with lower mean-squared pressure (point B) because the ARF pushed it there. Resultantly, the magnitude of the ARF is smaller and the particle has a net movement to the right.

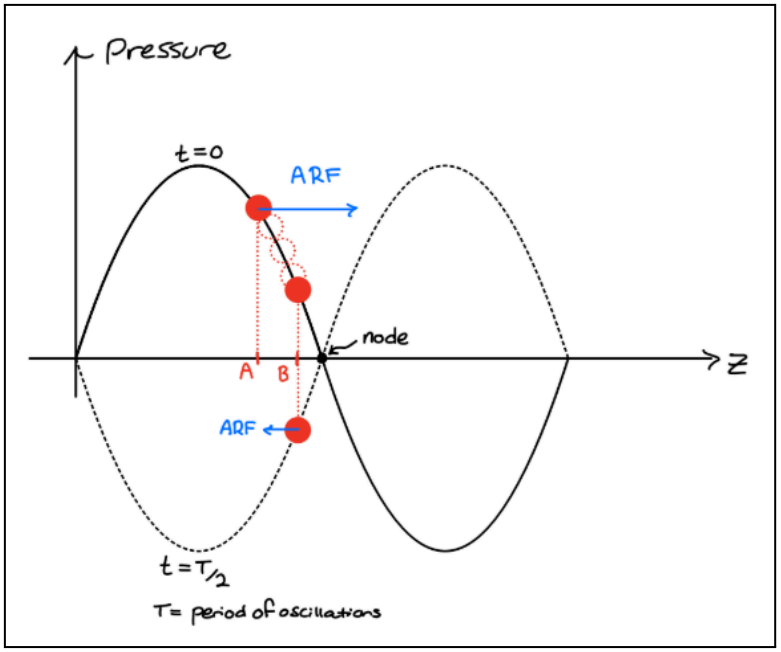



*Figure 7: Pressure vs position plot of a longitudinal wave propagating in the z-direction..*

Over time, this non-linearity of pressure magnitude over different positions along the standing wave will lead to a net movement of the particle towards the standing wave nodes where the net ARF is zero and the pressure oscillations are minimized, creating a stable point of equilibrium where the particle will remain unless subjected to strong external forces. Therefore, the nodes of the standing waves are potential minima known as Gor'kov potential minima. The ARF is the associated force to the Gor'kov potential field which pushes particles towards the Gor'kov potential minima.

## Key Design Parameters

When trying to move particles very fast, a key factor becomes how strongly a trap can hold a particle under disturbances. This factor is called stiffness. Stiffness is the spatial gradient of the ARF and is related to the steepness of the potential well. We need a high trap stiffness to ensure that large accelerations will not displace the particle from the trap.

Another important consideration in this system is ghost traps, also known as artifacts. Due to many factors, which will be discussed in detail below, producing a trap at location P can also inadvertently produce additional Gor'kov minima nearby. These minima are called ghost traps. Ghost traps are usually not as strong or as stable as the desired trap at point P, but can still trap a particle. It is important to be careful when placing particles in the field to ensure they are being placed in the correct traps.

There are several factors which affect the strength of traps and ghost traps, as well as stability. These interdependencies are summarized in Figure 8 and the non-trivial relations will be discussed below.

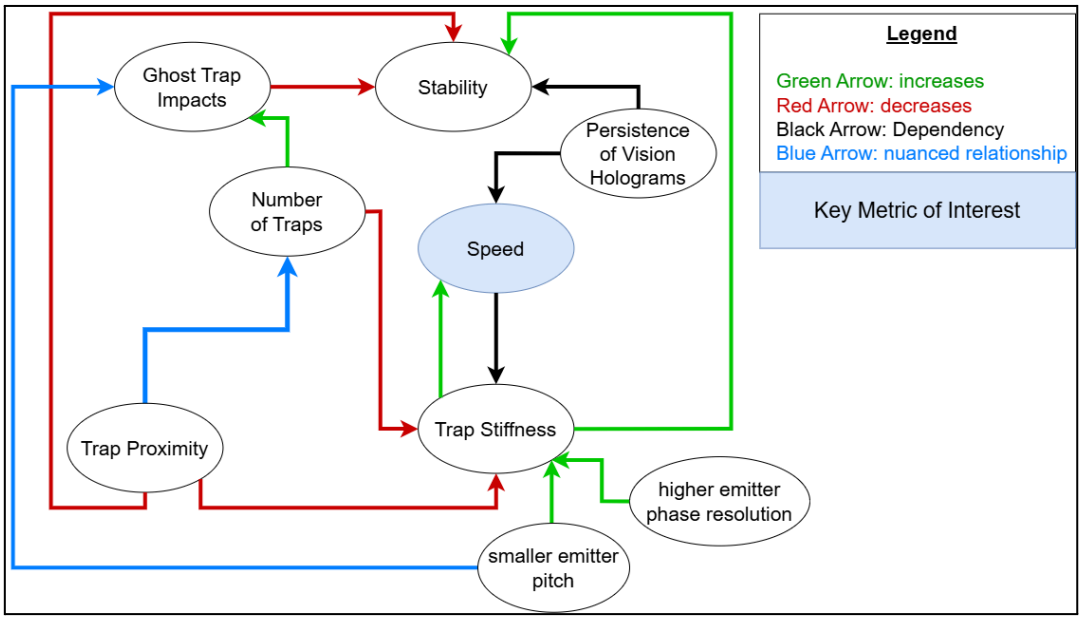



*Figure 8: Flowchart depicting interrelationships and dependencies of system parameters.*

Trap proximity has a nuanced effect on the possible number of traps. Trap proximity is governed by the Rayleigh Criterion, which dictates how close two traps can be placed together while still remaining distinct [8]. If two traps are placed too close together, they will merge (Figure 9). However, having higher trap proximity allows for more traps to be placed in the acoustic field, leading to a tradeoff between the number of traps and their proximity.

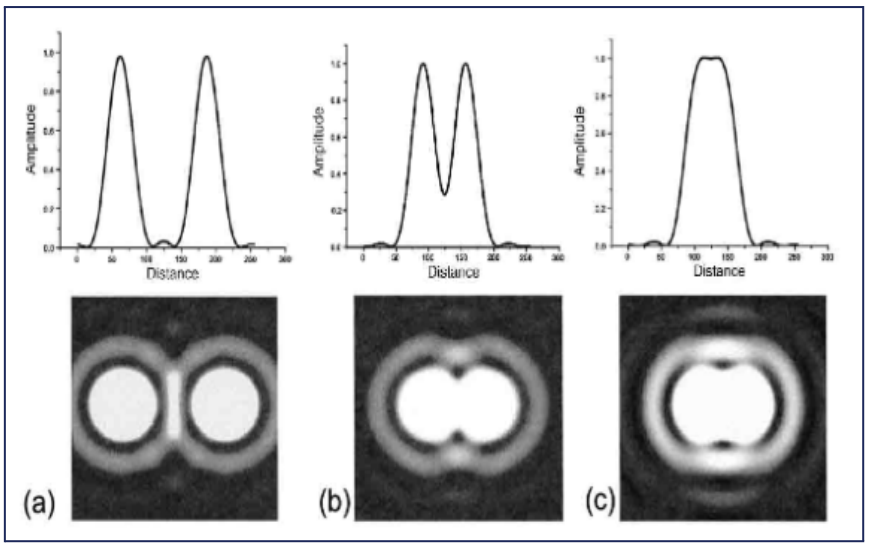



*Figure 9: Example images depicting the Rayleigh Criterion for two peaks [13]. **a)** The two peaks are distinct. **b)** The peaks begin to merge. **c)** The peaks are no longer distinct.*

Emitter phase resolution refers to how precisely the phase of each transducer can be controlled. It impacts stiffness because the stiffness is directly linked to how well we can control the interactions between waves, and this is done by modulating the relative phases (see Transducer Phase Offset Solver).

Emitter pitch refers to the center-to-center spacing of the transducers. If they are closer together, interference can occur over smaller spatial scales, leading to increased spatial resolution in the acoustic field.

Emitter pitch has a nuanced relationship with ghost traps. There are two causes of ghost traps. The first is when the periodicity of sound waves and nature of the solver lead to unintended local minima in the Gor'kov potential. The impact of these ghost traps can increase with smaller emitter pitch because the resolution of the acoustic field will increase and allow for more interference complexity in a smaller region of space. The second cause (grating lobes) is due to the interference of secondary beams emitted by a transducer (Figure 10) [14]. These grating lobes are due to the discrete nature of the transducer array and the symmetry in the geometry. Their strength and location are dependent on the emitter pitch relative to the wavelength [15]. The optimal emitter pitch that minimizes the effects of grating lobes is $\lambda / 2$, where $\lambda$ is the desired wavelength of the primary emitted beam.

The effects of the ghost traps become more pronounced as we generate more traps. For a low number of traps, the intended traps are much stronger than the ghost traps. However, generating more traps leads the strength of each intended trap to decrease, until they approach the strength of the ghost traps [8].

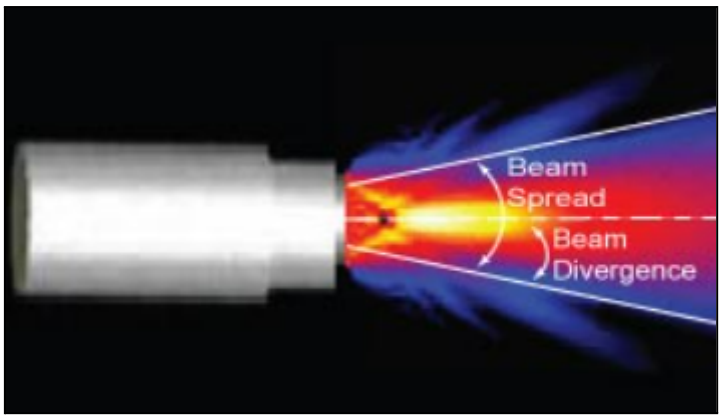



*Figure 10: Beam spread and divergence from a single transducer [16]. The secondary beams are in blue and are emitted at different angles than the main beam.*

# Implementation

## Transducer Phase Offset Solver

To solve for the phase offsets required for a specific trap location, a numerical solver is needed. Multiple implementations exist, but the previous team mainly considered the Gor'kov algorithm and the HAT algorithm as potential options. The Gor'kov algorithm uses gradient descent to solve for a minima in the Gor'kov potential [6]. The HAT algorithm instead uses the Gerchberg-Saxton method in order to create the desired traps [8]. In the end, the HAT algorithm was chosen due to faster iteration cycles, greater stability, and faster convergence [11]. The code for this solver was written in Rust, due to its high performance.

The HAT algorithm starts by initializing all phase offsets to zero. Then, at each desired bead[1] location, a complex pressure, containing both phase and amplitude information, is calculated by creating a superposition of the individual contributions of each transducer. These individual contributions are calculated using the far field piston source model (refer to the citation for an exact formula) [17]. The resulting pressure is normalized so it only contains phase data. Then, the control points are treated as if they themselves are transducers with that phase offset, and the resultant complex pressure is back-propagated to the actual transducers. The transducer phase is then set to the resulting phase, and the process is repeated until it converges towards areas of maximum pressure at the control points. Finally, a $\pi / 2$ phase shift is applied to each calculated phase to shift the region of maximum pressure vertically and leave a potential well at the desired location.

## Electrical Design, Digital Logic and Communication

The hardware components of the system are housed on a 10 layer custom PCB, which is mounted on a custom 3D-printed stand (for details about the stand, see Appendix A). The ten layers are required since each of the 256 transducers mounted on the board requires a separate trace, in addition to there also being multiple power planes required. Each pair of

[1] *The terms "bead" and "particle" are interchangeable. We use "particle" in the context of general principles and theory, and "bead" when discussing our system/implementation.*

transducers is driven by a MIC4127 gate driver IC, which provides the required high speed switching of 18-24V power to the transducers. The board contains two Cyclone 10 FPGAs which create the required waveform to drive the transducers. There is also an FTDI chip, along with an associated EEPROM bank, oscillator, and USB C port (now defunct, as explained below). The PCB, depicted in Figure 11, also contains input voltage regulators and other auxiliary ICs and components.

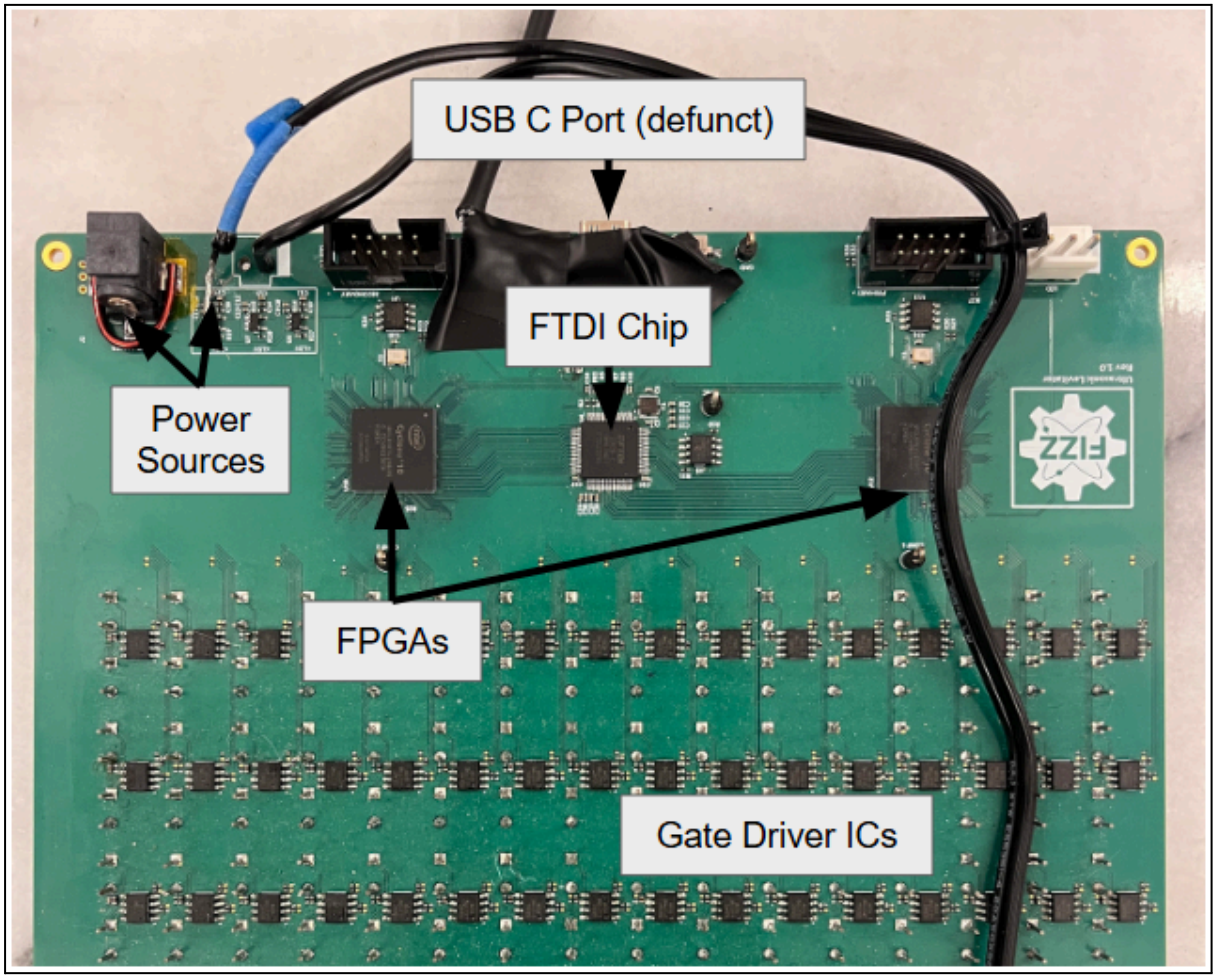



*Figure 11: Annotated diagram of the PCB designed by the previous team.*

The FTDI communication chip receives the phases which are computed by the solver via USB. The FTDI chip converts the USB data to FIFO (first in, first out) data, essential for interfacing with our FPGAs. Due to the need for multiple channels of communication, the previous team chose to use the asynchronous protocol with a bandwidth of 8Mb/s as opposed to the synchronous protocol with a bandwidth of 40 Mb/s [11].

The FIFO data is split evenly between the 2 FPGAs, which are responsible for generating a square wave for each transducer signal with the correct phase offset. These FPGAs are synchronised with a 40kHz squarewave and have a clock cycle driven at 24.576 MHz by the external oscillator [11].

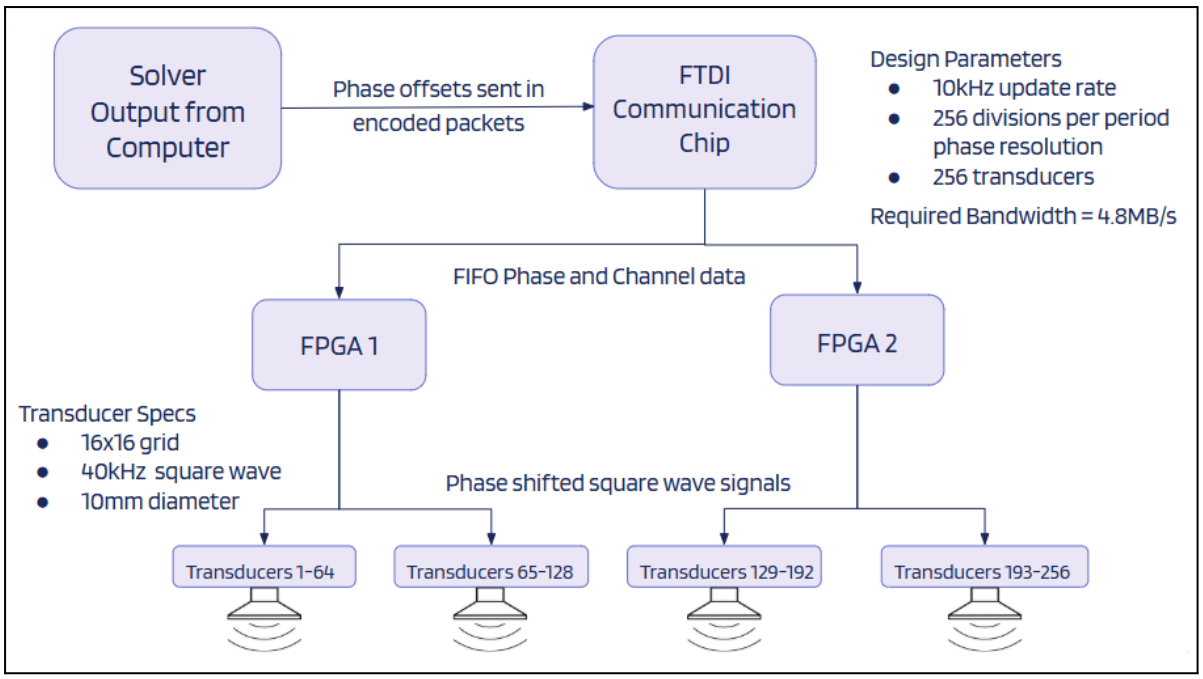



*Figure 12: Block diagram outlining the digital logic and communication pipeline. The solver output from the computer feeds into an FTDI communication chip, then into the FPGAs, then into the transducers.*

The board has a couple of key issues that should be addressed in a redesign. Firstly, the USB C port is missing pulldown resistors on key pins, meaning that computers won't recognize the connection. This issue was particularly nebulous and caused significant delays during setup. This was fixed by soldering a USB A cable directly to the board, but this is not a long-term solution. Secondly, the way that the FPGAs are mounted to the PCB makes them challenging to reprogram, and in future iterations we intend to make greater use of dev boards. Finally, a few connectors were reversed from how they should be, which was fixed through cut traces and jumper wires. This will be easy to fix in the future.

## Position Input

Position input refers to the software which allows the end-user to communicate the desired locations of traps to the solver, which takes in a list of (x, y, z) coordinates/trap locations as inputs. Our system allows the for the generation of trap locations by the following three methods, all implemented in Python scripts:

1. **Keyboard Input.** By pressing or holding down the arrow keys, the current location of the trap is updated and sent to the solver.

2. **Preprogrammed Lines.** When the arrow keys are held down, the rate at which new trap locations are published is limited by Python's PyQt graphical user interface (GUI) framework. By instead publishing a list of preprogrammed trap locations, we were able to improve the straight-line speed of the bead by over 50x.

3. **Path Following.** By representing the horizontal plane in which the bead moves as a flat canvas, the end-user can draw an arbitrary path by clicking and dragging the

mouse (Figure 13). If the path is drawn too quickly and/or contains many sharp turns, the bead may fail to successfully follow the entire path.

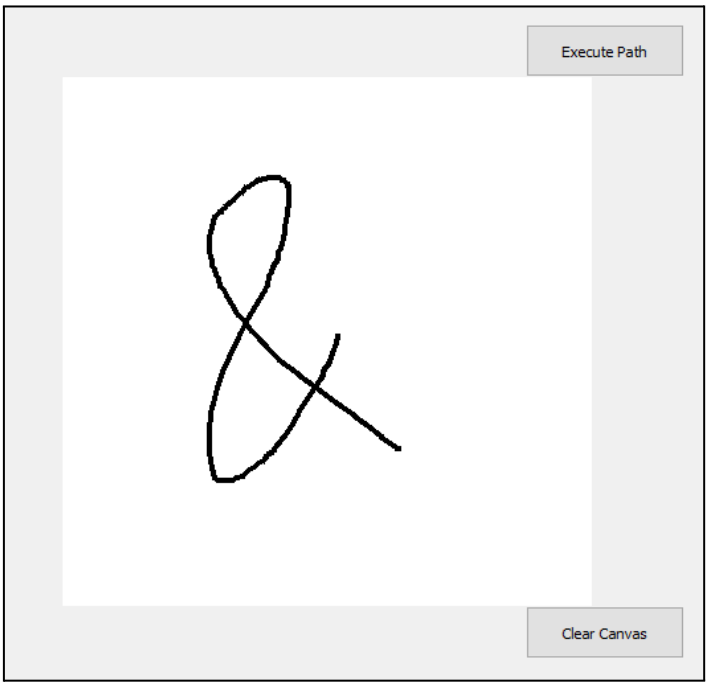



*Figure 13: GUI for path following position input script.*

These scripts only utilize horizontal movement, as we found that the system is unable to displace particles vertically. We also designed all three scripts to initially set the location of the trap to the center of the board. This is to ensure consistency when testing, and to help us initially place the bead into the trap.

An overview of the similarities and differences between the three position input methods is depicted below (Table 1).

| ***Aspect \ Method*** | **Keyboard Input** | **Preprogrammed Lines** | **Path Following** |
|---|---|---|---|
| *Real-time?* | Yes | No | Yes |
| *Movement Speed* | Slow | Fast | Variable |
| *Update Rate* | Slow | Fast | Fast |

*Table 1: Differences in behaviour between the three position input methods.*

When the scripts receive inputs from the end-user, the Redis publisher-subscriber (Pub-Sub) data communication framework transmits the received trap locations to the solver, as well as to a 3D simulation, which visualizes the expected location of the particle, relative to the grid of transducers (see Appendix B).

### System Level Diagram

Now that we have discussed how each individual component of the implementation operates, we return to a high-level overview of the entire system, which is depicted in the system level diagram below.

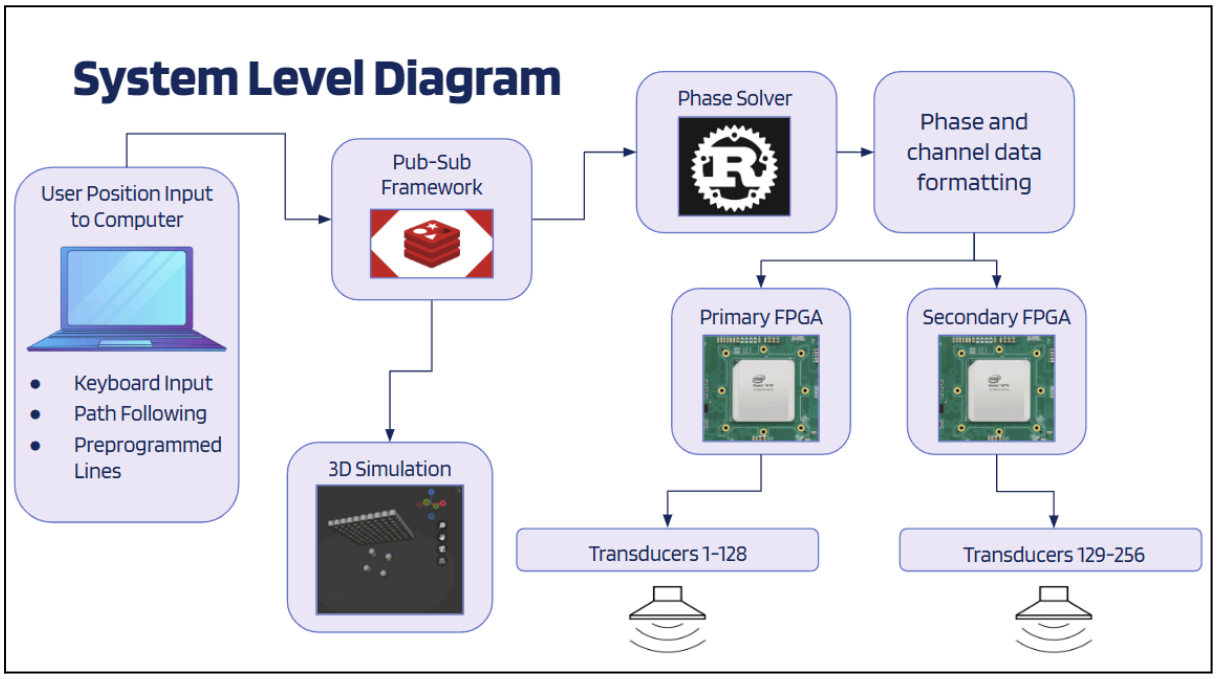



*Figure 14: System Level Diagram.*

We summarize the system operation as follows:

1. The end-user specifies the desired locations of traps via position input scripts.
2. The trap locations are published to the solver and a 3D simulation using the Redis pub-sub framework.
3. The solver converts trap locations to transducer phase offsets.
4. The phase offset data is serialized and delivered to a primary and secondary FPGA.
5. The FPGAs send signals to operate a 16x16 grid of transducers, creating a pressure field capable of levitating beads at the specified locations.

These steps all execute in real-time, leading to a system that is highly responsive to end-user input.

## Testing

In developing a testing procedure for the current system, we focused on obtaining accurate measurements for the maximum speed and acceleration at which we can move the beads. These are the 2 quantitative parameters that determine the quality of our system for displaying holograms. Qualitative parameters such as stability and accuracy were not the focus of our testing at this time are discussed in the Recommendations section.

We focused exclusively on measuring speed and acceleration for point-to-point motion. In short, the bead moves a certain distance left, waits a set delay time, then moves right the

same distance to return to the starting position. We found that this path is best set such that it is centred in the field, at the point (8.4 cm, 8.4 cm), where the grid of transducers measures 18.6 x 18.6 cm. As well, we chose to use a step profile for all speed changes (i.e. going from 0 cm/s to the set speed over one update cycle).

For ease of analysis, we structured our testing around performing ‘parameter sweeps’. We start all of the parameters well within the limiting behavior of our system, then gradually modify a certain parameter by a set amount. For example, performing a speed sweep involves setting the starting speed to 20 cm/s, having the bead perform a set path, then increasing the speed by 5 cm/s and repeating. By collecting data in these sweeps, it is both more efficient when collecting data, and easier to analyze the effect a specific parameter has on the output speed and acceleration.

This method also ensures that the controlled parameters are kept consistent, since it is challenging to achieve the same starting conditions each time if the bead is displaced from the trap. This semester, we performed speed sweeps, update rate sweeps, and delay time sweeps. The testing procedure contains two major stages, namely data collection and data processing. Our testing procedure, the description tests performed, as well as the excel data and video analysis code can be found in Appendix C.

## Data Analysis

Before performing any parameter sweeps, we chose to collect some baseline test data using relatively arbitrary parameter values. While this was originally not intended to be used for analysis, the data reveals there is a major rate limiting factor present in the software. This test data, as well as the data from our three parameter sweep tests is summarized below.

| ***Test Number*** | **Test 1** | **Test 2** | **Test 3** | **Test 4** |
|---|---|---|---|---|
| *Test Type* | Baseline Test | Update Rate Sweep | Set Speed Sweep | Delay Time Sweep |
| *Set Speed (cm/s)* | 100 | 100 | 20 - 60 | 100 |
| *Update Rate (Hz)* | 800 | 500-1500 | 800 | 800 |
| *Delay Time (s)* | 0.5 | 0.5 | 0.5 | 0.5-0 |
| *Max Speed (cm/s)* | 67.87 | 42.96 | 51.60 | 62.81 |
| *Swept Parameter at Max Speed* | N/A | 500 Hz | 45cm/s | 0.25 s |
| *Max Acceleration (cm/s^2)* | 3220 | 2711.34 | 2717.3 | 2487.7 |

| | | | | |
|---|---|---|---|---|
| *Swept Parameter at Max Acceleration* | N/A | 800 Hz | 40 cm/s | 0.2 s |
| *Parameter Value at Failure* | N/A | 1300 Hz | 45 cm/s | 0.2 s |

*Table 2: System parameters, as evaluated from various system tests. In the bottom row, "failure" is defined as when the bead falls out of the trap and thus ceases to levitate*.

When finding these maximum values, we discarded the data at the very beginning and very end of the test. This is because the beginning of the video captured the bead being placed in the field, and the end captured the failure behavior. While this data is valuable, it would not be accurate to describe these values as a result of the point-to-point motion, so it was not considered. This data cropping was performed on a case-by-case basis according to the video footage.

We also obtained position, speed, and acceleration graphs vs time. Below are the graphs for the baseline test (Figure 15), and graphs for the remaining tests can be found in Appendix C. There is an open question regarding the amount of smoothing required when graphing each data set. The image processing code applies a Gaussian blur to the data set before graphing. While it is clear that some blurring is required to eliminate small uncertainties between frames, it is unclear to what extent blurring affects our data. As it stands, we have been determining the strength of the Gaussian blur, σ, via trial and error.

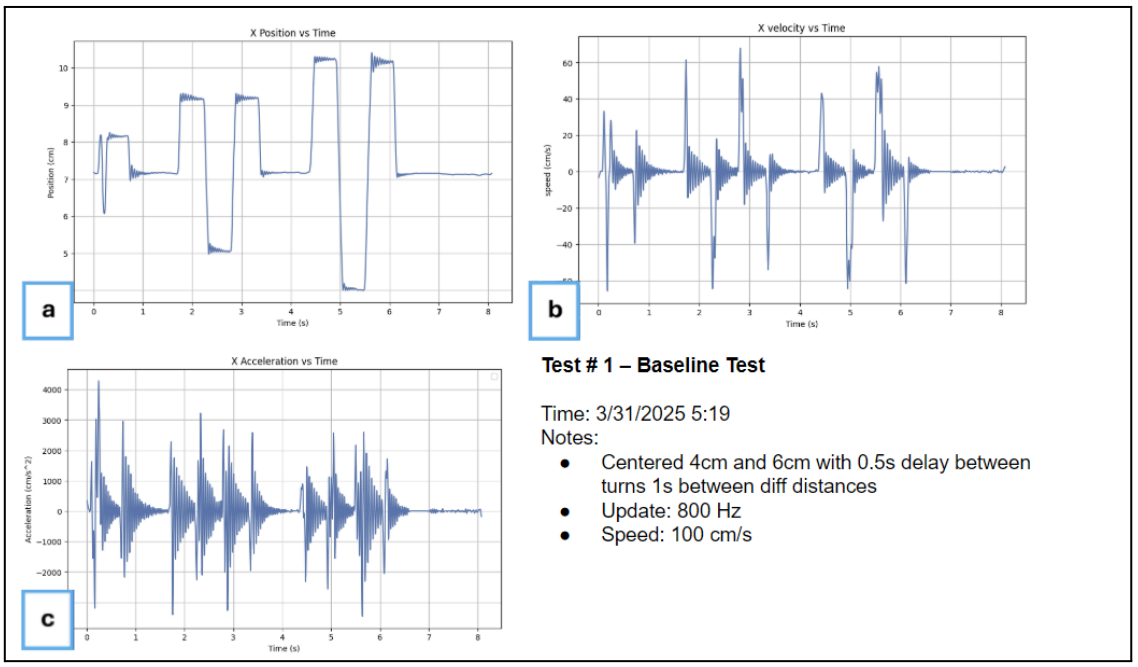



*Figure 15: Graphs for the baseline test. The **a)** position, **b)** velocity, **c)** and acceleration of the bead against time are depicted for the baseline test. The test parameters are outlined in the bottom-right. Refer to Appendix C for the test parameters.*

The accuracy of these numbers is uncertain as we only had time to take one set of measurements for each set of parameters. However, even with this uncertainty, we conclude that the max speed and acceleration are approximately 60 cm/s and 2700 cm/s^2 respectively when performing point-to-point motion.

One coincidence of using a step function for the speed profile is that the accelerations at the instant of speed change can be very large. So while it may appear as though the system is capable of extremely high accelerations, it most likely can only maintain these accelerations for durations on the order of 0.1s. This is because while the acceleration of the bead is proportional to the acoustic force, the resistance forces, such as drag, are proportional to velocity. At low speeds, it is possible to achieve very high accelerations because resistance forces are negligible, but this does not hold at higher speeds. It is unclear at this time what the maximum sustained acceleration the system can support.

As well, when graphing the position of the bead vs time, we observed behaviour similar to that of a harmonic oscillation when it ‘settles into a trap’. During the delay period in the point-to-point motion path, the bead displays what appear to be damped oscillations. This will be the main focus of our system quantification efforts in the immediate future: determining the k value for these oscillations in different areas of the field and for different values of these parameters.

# Conclusions

This year, our primary achievements have been in learning about the pre-existing system and the physics behind it. We spent a lot of time exploring the previous system, debugging various issues and understanding its behaviour. The learning process has been a success this year, and we are confident in moving forward to fully understanding the entire system and becoming comfortable in utilizing it.

The high-level goal of this project is to create holograms through use of levitating beads. In proving whether or not that is a viable concept, our evidence is inconclusive so far. With the current implementation, we are still limited by a lack of sufficient speed and control.

However, we have also found that there is still significant room to improve the capabilities of the system. With the planned improvements in hardware, solver code, and bead pathing, we expect to achieve significant progress in the coming year towards our goal of creating simple holograms.

# Recommendations

Our recommendations fall into 3 major categories: software, hardware, and testing.

## Hardware Recommendations

- Separate out the transducer array PCB and the FPGAs. All the components being on the same PCB adds unnecessary complexity and makes troubleshooting difficult.

- Use higher-quality transducers and decrease their spacing in the grid. By upgrading to more powerful transducers, we can increase the strength of the pressure field. Decreasing the spacing between transducers reduces the number of ghost traps (the current spacing between transducers is 11.26 mm).
- Improve the rigidity of the mechanical stand. Currently, the weight of the PCB exerts a torque which causes the board to slant, slightly destabilizing the traps.

### Software Recommendations

- Create a better system for deciding where to place traps by modelling bead velocity as well as position. Considering the dynamics of the bead will allow for faster movement than only evenly spacing the traps as they move.
- Improve solver convergence speed by caching previous results. Since the trap is not moving much from update to update, most transducer phases will not change much, so by caching previous results we can start closer and speed up convergence.
- Investigate pre-computing the required phases for a given path, allowing positions to be sent faster than the rust solver could compute them.
- Determine why the solver doesn't allow for vertical movement, and address the issue.

### Testing Recommendations

- Measure the stiffness *k* of traps. The bead displays harmonic motion when settling into a trap. This means that a trap has an effective *k* value that is large when the acoustic force is large and small when the acoustic field is small. Because *k* is then directly proportional to trap stiffness, by measuring *k* we can obtain a quantitative measure of trap quality and stability.
- Use different speed profiles. Currently, the point to point motion algorithm uses a square speed profile; by using a trapezoidal speed profile instead, it should be possible to increase the maximum speed.

# Deliverables

We submit the following items as project deliverables:

1. This report, which describes both the fundamentals and details of our project and represents the culmination of our work.
2. Our GitHub repository, containing old files created by the previous team, related to system design, and new files created by us.

# References


[1] G. Lucas, *Star Wars: A New Hope.* [DVD]. San Francisco, CA: Lucasfilm Ltd., 1977.

[2] J. Raffel *et al.*, “Ultrasonic Levitation as a Handling Tool for In-Space Manufacturing Processes,” *Journal of Manufacturing Science and Engineering*, pp. 1–7, Aug. 2024, doi: https://doi.org/10.1115/1.4066335.

[3] A. Watanabe, K. Hasegawa, and Y. Abe, “Contactless Fluid Manipulation in Air: Droplet Coalescence and Active Mixing by Acoustic Levitation,” *Scientific Reports*, vol. 8, no. 1, Jul. 2018, doi: https://doi.org/10.1038/s41598-018-28451-5.

[4] T. Vasileiou, D. Foresti, A. Bayram, D. Poulikakos, and A. Ferrari, “Toward Contactless Biology: Acoustophoretic DNA Transfection,” *Scientific Reports*, vol. 6, no. 1, Feb. 2016, doi: https://doi.org/10.1038/srep20023.

[5] X. Ding *et al*., “On-chip manipulation of single microparticles, cells, and organisms using surface acoustic waves,” *Proceedings of the National Academy of Sciences*, vol. 109, no. 28, pp. 11105–11109, Jun. 2012, doi: https://doi.org/10.1073/pnas.1209288109.

[6] R. Hirayama, G. Christopoulos, D. Martinez Plasencia, and S. Subramanian, “High-speed acoustic holography with arbitrary scattering objects,” *Science Advances*, vol. 8, no. 24, Jun. 2022, doi: https://doi.org/10.1126/sciadv.abn7614.

[7] R. Hirayama, D. Martinez Plasencia, N. Masuda, and S. Subramanian, “A volumetric display for visual, tactile and audio presentation using acoustic trapping,” *Nature*, vol. 575, no. 7782, pp. 320–323, Nov. 2019, doi: https://doi.org/10.1038/s41586-019-1739-5.

[8] A. Marzo and B. W. Drinkwater, “Holographic acoustic tweezers,” *Proceedings of the National Academy of Sciences*, vol. 116, no. 1, pp. 84–89, Dec. 2018, doi: https://doi.org/10.1073/pnas.1813047115.

[9] T. Fushimi, A. Marzo, B. W. Drinkwater, and T. L. Hill, “Acoustophoretic volumetric displays using a fast-moving levitated particle,” *Applied Physics Letters*, vol. 115, no. 6, p. 064101, Aug. 2019, doi: https://doi.org/10.1063/1.5113467.

[10] “Persistence of Vision (POV) Animation Guide | Adobe,” Adobe. https://www.adobe.com/uk/creativecloud/animation/discover/persistence-of-vision.html (accessed Apr. 08, 2025).

[11] A. Chen, Q. Ferbers, K. Lin, and J. Seto, “Pushing the Frontier of Open-Source Ultrasonic Phased Arrays for Multi-Modal Stimuli,” Apr. 2024.

[12] A. Marzo, M. Caleap, and B. W. Drinkwater, “Acoustic Virtual Vortices with Tunable Orbital Angular Momentum for Trapping of Mie Particles,” *Physical Review Letters*, vol. 120, no. 4, Jan. 2018, doi: https://doi.org/10.1103/physrevlett.120.044301.

[13] D. E. Wolf, “The Optics of Microscope Image Formation,” *Methods in cell biology*, pp. 11–42, Jan. 2007, doi: https://doi.org/10.1016/s0091-679x(06)81002-2.

[14] F. C. Laing and A. B. Kurtz, “The importance of ultrasonic side-lobe artifacts.,” *Radiology*, vol. 145, no. 3, pp. 763–768, Dec. 1982, doi: https://doi.org/10.1148/radiology.145.3.7146410.

[15] B. E. Anderson, W. J. Hughes, and S. A. Hambric, “Grating lobe reduction in transducer arrays through structural filtering of supercritical plates,” *The Journal of the Acoustical Society of America*, vol. 126, no. 2, pp. 612–619, Jul. 2009, doi: https://doi.org/10.1121/1.3159366.

[16] “Nondestructive Evaluation Techniques : Ultrasound,” *Center for Nondestructive Evaluation*. https://www.nde-ed.org/NDETechniques/Ultrasonics/EquipmentTrans/beamspread.xhtml.

[17] “Radiation from a Plane Circular Piston.” [Online]. Available: https://jontallen.ece.illinois.edu/uploads/473.F18/Lectures/Chapter_7b.pdf.

# Appendix A - Mechanical Stand Design

When we took over this project, the system was designed with a U-shaped stand intended to tightly constrain the PCB (board). The board slides into the U and is held in place via tension in the tolerances. This stand was very strong and very stiff, which presented a difficulty in setting up and dismantling the system. Due to repeatedly pulling at the arms when inserting the board, the previous stand snapped. When redesigning, we chose to keep the general shape but switched from using low tolerances to using screw holes and PCB mounts to constrain the board.

While designing this stand to be 3D printed, we were limited by the size of the Project Lab 3D printers and had to significantly reduce the size of the design. As a result, the stand has a large moment arm, causing the board to tilt relative to the reflecting plate. Any future redesign of the stand should aim to eliminate this tilt. As well, the next stand revision should allow a camera to be mounted a fixed distance away from the transducer array, which will improve our ability to take consistent measurements.

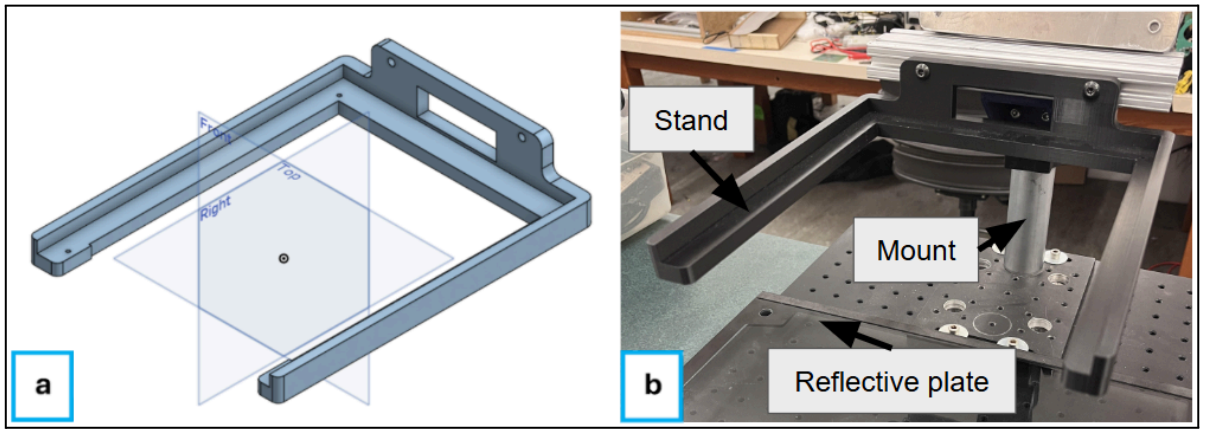



*Figure 16: Mechanical stand, in CAD and in real life.* ***a)*** *The stand, as seen in Onshape, and* ***b)*** *a photo of the printed stand.*

# Appendix B - 3D Simulation

As discussed in the Position Input section, the trap locations generated by the position input scripts proceed not only to the solver, but also to a 3D simulation. Written in the 3D computer graphics software Blender, the simulation generates a visualization of the expected bead location, as depicted in Figure 17.

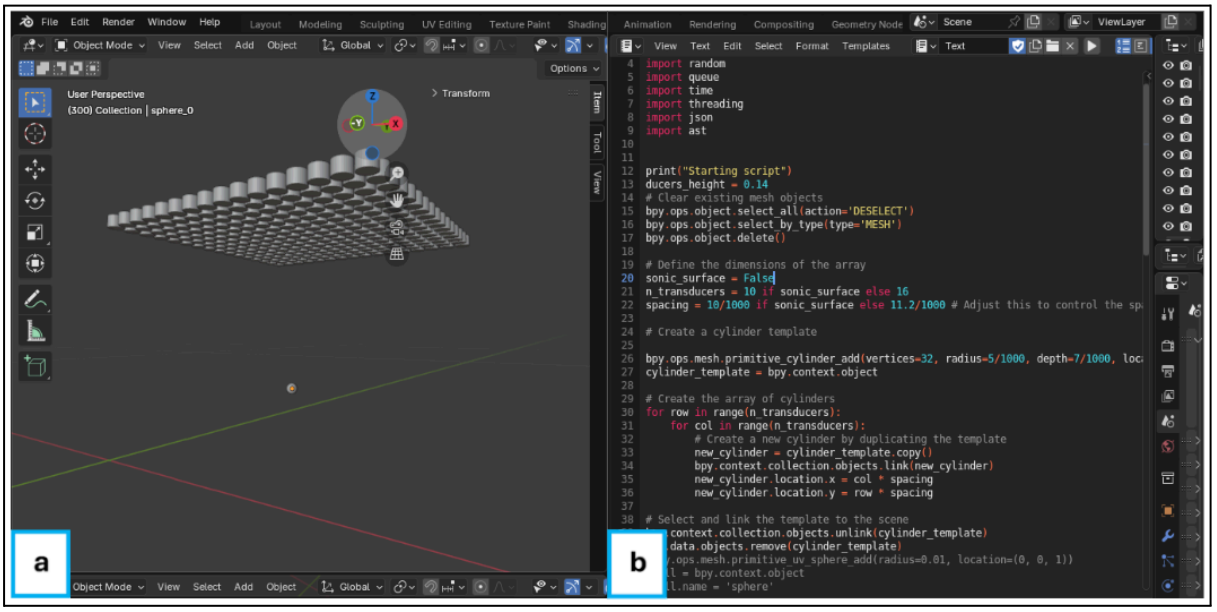



*Figure 17: 3D simulation, written in Blender. The 3D simulation allows the end-user to visualize the expected position of particles, relative to the grid of transducers. The Blender window consists of **a)** a visual projection of 3D space, and **b)** the script required to run the simulation in real-time.*

The simulation updates in real-time upon receiving messages from the Redis pub-sub framework. Blender also synergizes well with the position input scripts, as they are both written in the same language (Python).

# Appendix C - Test Procedures and Test Data

Our collected data and image processing code can be found here. Below is our data collection procedure, as well as graphs from all test data.

## Data Collection Procedure

1. Set up a tripod with a camera perpendicular to the direction of motion you intend to analyze.
2. Open the keyboard control position input script and modify the programmed path as desired by modifying the (x,y) positions, delay time, update rate, and speed.
3. Record the test number, the test parameters set in step 2, and the date and time.
4. Start recording, press “x” to execute the chosen path, and stop recording when the motion ends.
5. Repeat steps 3 and 4 with the same chosen test parameters 3 - 5 times.
6. Export the footage to Google Drive, and rename the videos to distinguish which video corresponds to which set of test parameters.
7. Repeat for each set of testing parameters which you wish to analyze.

## Data Processing Procedure

1. Open the “Speed Calc.ipynb” file in Google Colaboratory, modify the input and output file paths as required, and (if necessary) convert the video to .mp4 format.
2. Modify the location and size of both the red box and green line in cell 3 such that the bead remains in the red box for the entirety of the video and the green line aligns with the plate underneath the reflective plate.
    a. Note: you can use any reference object with a known length if the plate is not visible, so long as the “reference_object_length” variable is adjusted accordingly.
3. Run cell 4 to crop the video. This will ensure that only the portion in the red box is analyzed.
4. Adjust the masking filters in cell 5 such that a green box appears around only the bead in at least the first 10 frames. This will ensure that the software is tracking the bead and only the bead in every frame.
5. Run cells 6, 7, and 8 to obtain the values for position, velocity and acceleration.
6. Run cell 8 to export the relevant data and parameters to Excel for analysis.

# Test Data Graphs

*Please zoom in if the following figures are unclear. Times are listed in PM.*

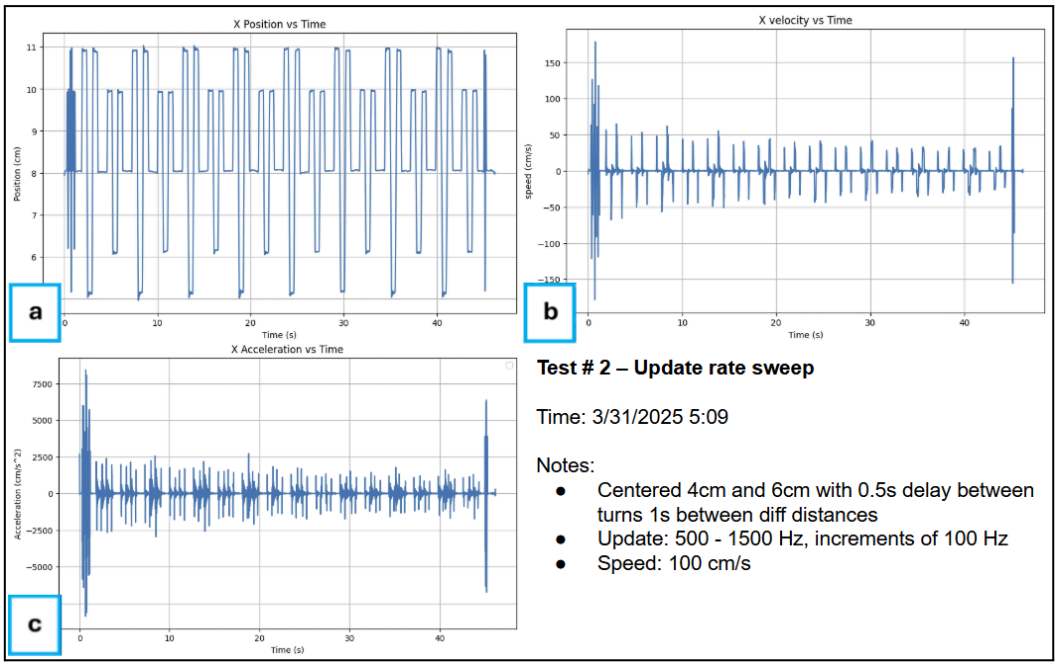



*Figure 18: Graphs for the update rate sweep test. The **a)** position, **b)** velocity, **c)** and acceleration of the bead against time are depicted for the update rate sweep test. The test parameters are outlined in the bottom-right.*

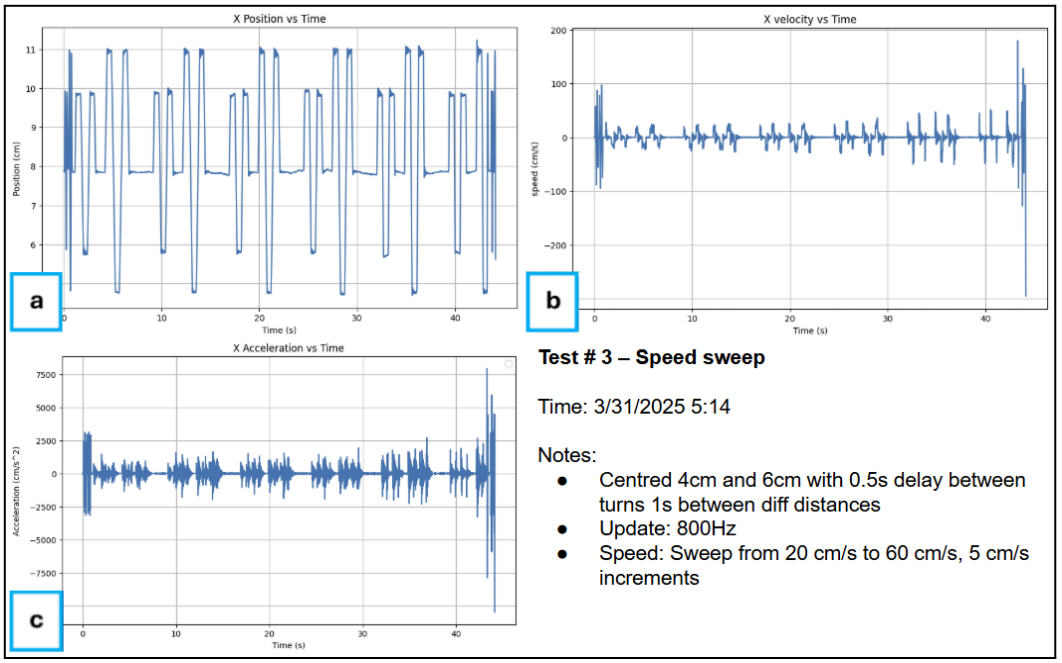



*Figure 19: Graphs for the speed sweep test. The **a)** position, **b)** velocity, **c)** and acceleration of the bead against time are depicted for the speed sweep test. The test parameters are outlined in the bottom-right.*

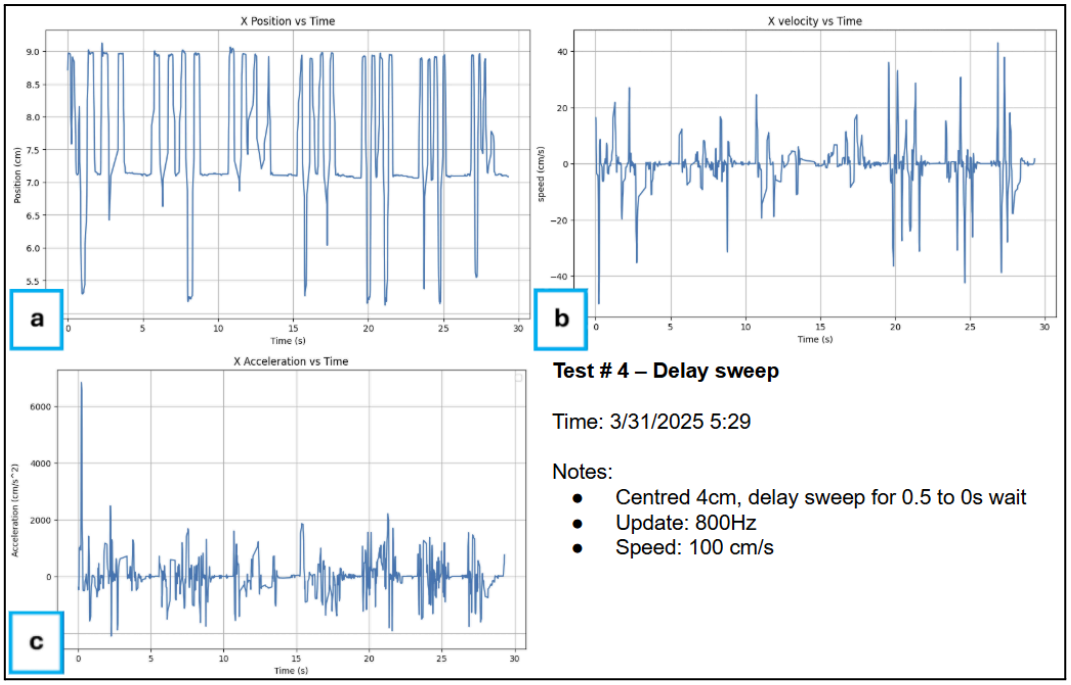



*Figure 20: Graphs for the delay sweep test. The **a)** position, **b)** velocity, **c)** and acceleration of the bead against time are depicted for the delay sweep test. The test parameters are outlined in the bottom-right.*