Ann and H.J. Smead
Aerospace Engineering Sciences
UNIVERSITY OF COLORADO **BOULDER**

Collision Avoidance System Testbed

Project Final Report

University of Colorado Boulder

AUTHORS

Conner Martin, Isaac Goldner, Sam Hartman, Cameron Turman, Trace Valade, Adam Holdridge, Angel Hoffman, Roland Bailey, Reade Warner, Hugo Stetz, Griffin Van Anne, Jason Balke

CUSTOMER

John Reed (United Launch Alliance)

ADVISOR

John Mah

May 3, 2021

Contents

List of Figures

List of Tables

Acronyms

CAST Collision Avoidance System Testbed. 4, 5, 8–12, 18–26, 29, 31–33, 49, 50, 53–55, 57–63, 69–72, 77, 78, 81, 83

EKF Extended Kalman Filter. 16, 18, 19, 33, 54, 72, 73

FOV Field of View. 80

LEO Low Earth Orbit. 77
LiDAR Light Detection and Ranging. 4, 5, 11, 16, 17, 20, 22, 23, 25, 41, 43, 45, 53, 54, 56, 63, 67, 70, 72, 78–80, 85

NEES Normalized Estimation Error Squared. 73
NIS Normalized Innovation Squared. 73

PDF Probability Density Function. 19

ULA United Launch Alliance. 8, 9, 12

WBS Work Breakdown Structure. 65

Nomenclature

$\dot{\omega}$	Stepper motor angular acceleration
η_P	Power efficiency
$\hat{x}_k$	State estimate at time step k
μ	Gravitational parameter of the earth
ω	Angular velocity
ρ	Density
σ	Variance
τ	Object detection-to-gantry maneuver process time constant
θ	Measurement bearing
ξ	Average complex representation of bearing measurements
a	Acceleration of orbital object
c_r	Coefficient of rolling resistance
d	Diameter of collision object
$e_{x,k}$	Error in the states
$e_{y,k}$	Error in the measurements
G	Gravitational constant
h	Height of ball release
I	Moment of inertia
k	Discrete time step
l	Length from the sensor to the end ot the testbed where object is beign launched
M	Mass of Earth
m	Mass of rolling ball
N	Number of Monte Carlo simulations
n	Number of states
p	Number of measurement elements

P_k	Estimation error covariance at time step k
P_{in}	Power draw of component
Q_{gen}	Heat generated
Q_k	Process noise covariance used to propagate state estimator
R	Sensor measurement noise covariance used to propagate state estimator
r	Measurement range
r	Orbit radius; range in sensor range-bearing measurement
t_d	Maximum allowable object detection-to-gantry maneuver process delay time
v	Linear speed; velocity of orbital object
X	Test bed long axis
x_k^+	Corrected states
Y	Test bed short axis
y_k^-	Predicted measurements
y_k	Received sensor measurement at time step k
Z	Test bed axis out of collision plane

I. Project Purpose

A. Project Overview & Motivation

Authors: Adam

Outer space is cluttered! Currently there are over 500,000 objects in orbit the size of a marble or larger, and new objects are being added with every launch [1]. At orbital velocities, these objects pose a serious threat to spacecraft, and collisions can be disastrous, potentially adding more debris. According to NASA, by 2005 the amount of debris in LEO had grown to the point that even if no additional objects were launched into orbit, collisions would continue to occur, compounding the instability of the debris environment and increasing operational risk to spacecraft [1]. Although ground station debris tracking is capable of predicting potential collisions, it can allow for errors of up to tens of kilometers [2]. Therefore, if an incoming object could be detected with a limited time frame until collision, a decision could possibly be made between ending the mission and performing a reaction maneuver. It is clear that it would be advantageous for active satellites to be capable of detecting and avoiding collisions on orbit.

ULA tasked the team to develop and test a detect and react algorithm for exploring the feasibility of on board avoidance routines. Team CAST has developed a test bed capable of conducting a live collision scenario to test an avoidance algorithm. State estimation and motion prediction are used to inform a custom control law to determine a required reaction maneuver to avoid collision. Due to the vast distance scale and velocities of actual orbital collisions, the CAST prototype does not attempt to maintain scale of the physical demonstration. Rather, the acceleration profile of the test bed avoidance maneuver replicates that of a typical satellite on orbit. After successful verification of the unscaled collision scenario, the simulation software used to develop the unscaled control law is used to implement the same control law in a scaled collision scenario representative of true orbital motion. The project therefore consists of three main phases: unscaled simulation, unscaled physical demonstration, and finally scaled simulation. These phases allow the results gleaned from the physical demonstration to be extended to real satellite applications.

Through this project, valuable insight is gained into the viability of a real-time spacecraft avoidance system. Further improvements are made based on test and simulation results of the system. Benefits and drawbacks of the collision avoidance package are identified and serve as a basis for future developments with the ultimate goal of creating a package fit for use on spacecraft.

B. Previous Work

Authors: Angel

International space agencies and the scientific community are in agreement that prevention of further space debris, although important, are not sufficient to stabilize the orbital debris environment [3]. Many efforts have been made to attempt to find a solution to clean up orbital debris. ClearSpace-1 was selected to conduct the first space mission to remove an item from orbit [4]. Collision avoidance of spacecraft is not a novel technology. In fact, NASA's International Space Station has conducted approximately 27 collision avoidance maneuvers since 1999 [3]. The novelty of the collision avoidance testbed is the study of state estimation and sensing on board the spacecraft itself without ground tracking of orbital debris. If the technology developed in this project was used in congruence with the ground tracking capabilities, collision avoidance that may not have been feasible before may become so.

II. Project Objectives and Functional Requirements

A. Project Objectives

Authors: Conner

CAST is focused on implementing a physical 2D demonstration of a detect, decide, and react algorithm in order to assess the feasibility of on-orbit collision avoidance systems. Throughout the course of the project, the encountered problems and decisions made will help ULA assess the potential use of a collision detection and avoidance technology in their products. The specific objectives for this project are outlined below:

- Implement a physical 2D demonstration that utilizes a detect and react algorithm.
- Detect foreign incoming object in the detection space of the testing environment.
- Perform state estimation and motion prediction of foreign object.
- Predict whether a collision is likely.
- Develop a control law that determines reaction maneuver in relative frame while mimicking thruster motion.

- Prove control law via implementation in various collision scenarios.
- Scale up control law in simulation to a 1300 km cross track collision scenario

Overall, team CAST provided ULA with a testbed consisting of a maneuvering system, launching mechanism and environment, electronics components box, and distance sensor, along with a state estimation and avoidance algorithm capable of following a specific acceleration profile. The critical project elements for success are centered around the guiding functional requirements and include the physical testbed, detection, determination, avoidance, and simulation.

B. Levels of Success

Authors: Conner

The levels of success for this project are given in table 1.

Table 1 CAST Levels of Success.

Project Element	Level 1	Level 2	Level 3	Level 4
Test Environment	Testbed is capable of creating a 1D collision trajectory (no miss scenario)	Testbed is capable of 1D collision with variations in approach speed	Testbed is capable of 2D collision scenario with variations in approach speed and heading	N/A
Detection	Able to filter out sensor measurements outside bounding region of testing environment and detect stationary object	Able to detect moving object (>50mm sphere) at speeds up to 0.5 m/s	Able to detect moving object (>50mm sphere) at speeds up to 1 m/s	Able to detect moving object (>50mm sphere) at speeds up to 2 m/s
State Estimation	Able to return estimation of state at current time and predict forward to point of collision	2 sigma prediction covariance driven to within an avoidable region	70% confidence dynamic consistency chi-squared hypothesis testing passes	95% confidence dynamic consistency chi-squared hypothesis testing passes
Avoidance	System can avoid a collision (without tracking acceleration profile input)	Avoidance maneuver follows acceleration profile with <15% error	Avoidance maneuver follows acceleration profile with <10% error	Avoidance maneuver follows acceleration profile with <5% error
Testbed Simulation	Control law simulated for 1D collision profile represented on testing environment	Control law simulated for any 2D collision profile capable of being represented on testing environment	N/A	N/A
Application Simulation	N/A	N/A	Control law scaled up to a single full-scale orbital cross-track scenario	Control law performance improved upon using results from full-scale orbital maneuver scenario results

C. Concept of Operations

Authors: Angel

Figure 1 below illustrates the four different phases that the Collision Avoidance System Testbed operates with. The first phase is to sense the incoming object with the on board sensor and perform state estimation which estimates where the incoming object is and with what degree of certainty. The second phase then determines how likely a collision is to occur and whether or not it is feasible to actually perform a maneuver based on the probability of a collision and the estimated time until the collision could occur. If it is determined that a maneuver should be performed, phase three is implemented where path planning and avoidance maneuvering takes place. Lastly, the positional data for both the sensor and the object is fed back to the computer in order to analyze the avoidance performance.

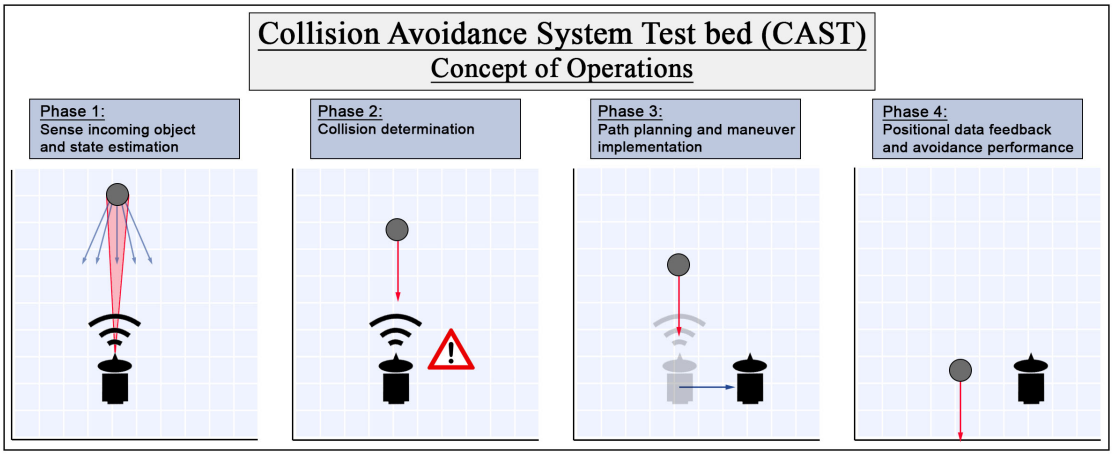

Fig. 1 CAST Concept of Operations

D. Project Deliverables

Author: Adam

The course specific project deliverables are given below along with a brief description:

- **Project Definition Document:** The project definition document (PDD) formed the technical foundation for the CAST project and was developed in cooperation with the project customer. This document presents an understanding of the project focus, objectives, functional requirements, and overall scope, and set a basis for the roles of each team member.
- **Conceptual Design Document:** The conceptual design document (CDD) detailed the top-level design activities for the project leading to the selection of a baseline design. This document includes a project description, design requirements, considered design options, a trade study process and results, and finally a selection of a baseline design.
- **Preliminary Design Review:** The preliminary design review (PDR) provided evidence that the project objectives were feasible and could be accomplished within the constraints of technology and time.
- **Critical Design Review:** The critical design review (CDR) represented the final phase in the design process of this project. CDR presented the overall design, described how it met the project and customer requirements, key manufacturing decisions, how the system would be integrated and tested, and the necessary resources to carry out all necessary tasks.
- **Fall Final Report:** The fall final report (FFR) was a comprehensive document detailing the design synthesis portion of this project.
- **Manufacturing Status Review:** The manufacturing status review (MSR) served as a formal interim review on the status of the project and outlined the schedule, budget, and a detailed manufacturing plan.
- **Test Readiness Review:** The test readiness review (TRR) served as a formal interim review on the status of the project during the testing phase.
- **AIAA Student Regional Conference Paper:** The AIAA paper was representative of a conference paper used in the aerospace industry. The CAST AIAA paper included a brief design overview as a verification, but focused mainly on the sensing and software components of the project.

- **Senior Project Symposium Video:** The senior project symposium video was a short, 5 minute presentation used to describe the project components, baseline results, and provide a background to industry professionals viewing the project.
- **Senior Project Symposium Presentation:** The senior project symposium presentation was a 15 minute presentation given to industry professionals to describe the overall motivation for the project, key project elements, and high-level results of the system testing.
- **Spring Final Review:** The spring final review (SFR) was the final presentation of this project, which outlined how and to what extent the project goals were accomplished. This specifically focused on the verification and validation of the project.
- **Project Final Report:** The project final report (PFR) is a comprehensive document that details the design practicum portion of this project. Its objectives include providing an explicit and complete picture of the project and how the project objectives and functional requirements were met and validated.

The customer did not request any additional deliverables. However, their guidance and feedback was used to guide the team in deciding what information should be contained within this final report.

E. Functional Block Diagram

Authors: Isaac

The functional block diagram shown in Fig. 2 demonstrates the interaction of the physical and software components of CAST. Overall, the system is comprised of both the CAST prototype and the primary computer. Within the CAST prototype is the two power supplies which feed power to the scanning LiDAR sensor and the two stepper drivers as well as the two gantry limit switches. The stepper drivers power the stepper motors which interact with the x and y axis encoders. The encoders then feed back position data into the stepper drivers. An Arduino Mega sends commands to the two stepper drivers on both axes in order to control the motors and is fed information from the two gantry limit switches as well as information about the x and y positions from the encoders. The Arduino Mega is powered at 5V DC provided from the laptop. Within the primary computer is the guidance, navigation, and control subsystem which consists of a state estimator, maneuver determination, and position plotting which is fed information from the state estimate as well as x and y axis position data from the Arduino Mega stationed on the CAST prototype. The Arduino Mega on the maneuvering subsystem is provided x and y axis commands from the maneuver determination on board the primary computer. The CAST prototype and primary computer work together through these connections to provide the user with a collision avoidance maneuver system that moves in the x and y directions and a user-friendly position plot to analyze the avoidance performance. The testbed environment and state estimation and avoidance algorithm is designed by team CAST, while the maneuvering system is acquired from IGUS and discussed in greater detail in the following sections.

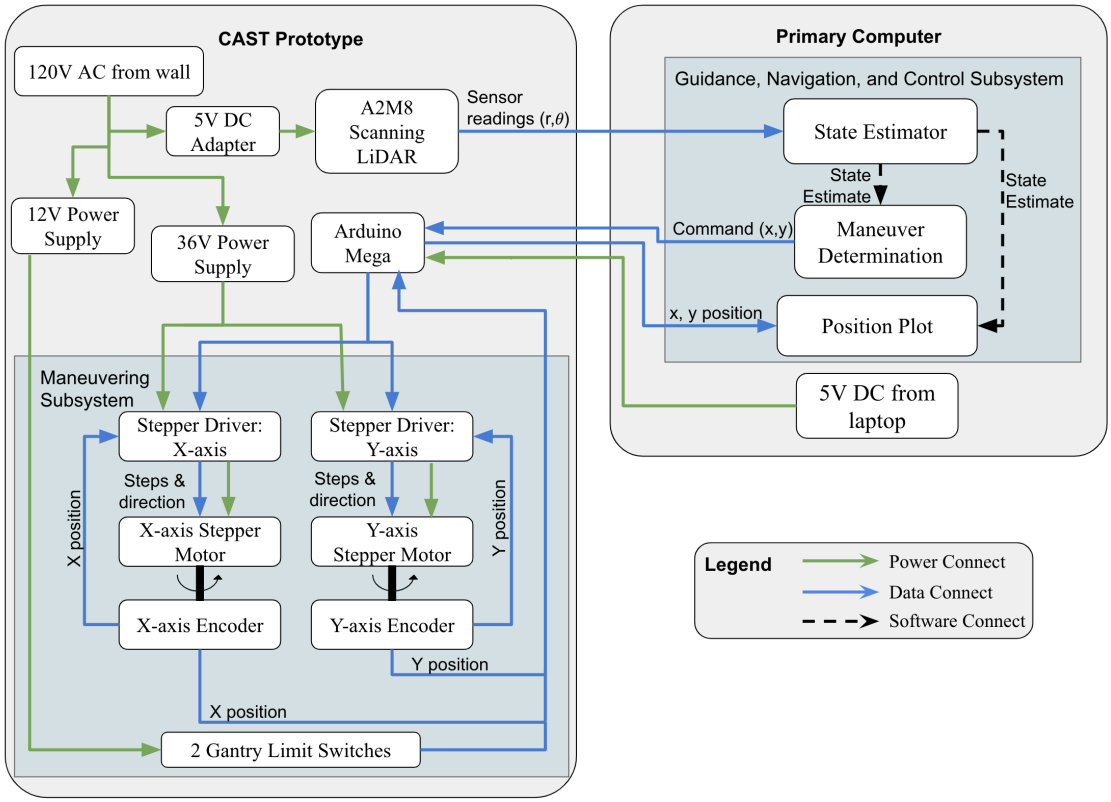

Fig. 2 CAST Functional Block Diagram

F. Functional Requirements

Authors: Isaac

FR1: The test system shall consist of a physical testbed capable of creating relative motion between two objects.

Motivation: A physical testbed must be created for repeated use in order to test sensing, determination, and maneuvering capabilities. To do so, the testbed must be capable of creating relative motion between two objects with a degree of certainty. Developing a physical testbed, rather than strictly a software simulation, for avoidance maneuvering is also based on the ULA customer-provided requirements.

FR2: The test system shall be capable of detecting a live, incoming object.

Motivation: The first step of an avoidance algorithm is to detect that an object is within a certain distance of the maneuvering hardware. Assuming that no external positional information is provided to the avoidance algorithm, the algorithm must rely solely on data provided via an onboard sensor to detect that an object is located within the testing environment.

FR3: The test system shall be capable of determining if a collision will occur.

Motivation: Following detection of an object within the testing environment (described in FR2), the system must use this information to determine if a collision will occur. Since the onboard sensor will not be capable of positional prediction with 100% certainty and the object within the testing environment will likely be moving, a probabilistic approach will be necessary for collision prediction. Therefore, the avoidance algorithm must use the sensor data and known situational uncertainties to determine if a collision will occur. Existing situational uncertainties exist for the predicted state estimation of the incoming ball as well as uncertainty for the point of collision.

FR4: The test system shall be capable of avoiding a physical collision using motion characteristic of a thruster response in orbit.

Motivation: If a collision is determined to occur within a specified probability (described in FR3), the test system must plan and implement an avoidance maneuver. With a number of satellites holding a monopropellant blowdown thruster onboard, the available thrust may decrease over time. Therefore, to better replicate the orbital collision scenario the avoidance maneuver shall follow a specified acceleration profile characteristic of a full-scale thruster response.

III. Final Design

A. Requirements Flow Down

Authors: Isaac

1. Requirements Derived from Functional Requirement 1

FR1: The test system shall consist of a physical testbed capable of creating relative motion between two objects

DR 1.1: *The test system shall be capable of creating various trajectories of the incoming object to create a collision or miss scenario.*

Motivation: To prove the robustness of the avoidance algorithm and maneuvering hardware, multiple incoming object trajectories must be tested. The ability for the test system to produce both collision and miss scenarios is critical in verifying the effectiveness of the detect and react system to ensure the algorithm produces a maneuver only when necessary.

DR 1.2: *The incoming object trajectory shall be within the plane of collision. In other words, the trajectory of the avoidance maneuver and the incoming object will reside within the same 2D domain.*

Motivation: The restriction of relative motion to a 2D plane is an important design constraint on the relative motion between the incoming object and avoiding object. This design requirement restricts the domain of the test environment which simplifies and focuses the scope of the testbed. This is justified based on the planar nature of a full scale orbital collision. See Appendix section X.B for planar justification.

DR 1.3: *The test system shall be fully functional after repeated detect and react procedures, where full functionality is defined as the ability to sense position and velocity data for an incoming object, integrate this data into the avoidance algorithm software, and perform an avoidance maneuver.*

Motivation: The test system required by FR1 is further required to remain full functionality over the course of multiple tests. Sensing equipment, maneuvering hardware, launching mechanisms, and all associated hardware and software must remain intact and fully operational after the testing is performed. This ensures that the testbed delivered to the customer remains as a useful product which can be repeatably used for testing and iteratively designing a product.

DR 1.4: *The total cost of the test bed system shall be less than $5000.*

Motivation: This requirement is based on the course related budget and limits the scale of the test system required by FR1.

DR 1.5: *The incoming object shall maintain constant velocity to within 5% of its initial velocity upon launch.*

Motivation: The relative velocity between two objects in orbit, assuming a 2D collision scenario, remains constant throughout the time scale of interest for the last-minute collision scenario avoidance. This assumption is validated in section X.B. Therefore, the testbed relative velocity shall also remain constant (to within 5%) during testing.

2. Requirements Derived from Functional Requirement 2

FR2: The test system shall be capable of detecting a live, incoming object

DR 2.1: *The sensor shall be capable of detecting one incoming object.*

Motivation: The time scale of interest is of a short duration due to the last minute nature of this collision scenario as described in the Project Objectives (Section II). While there are likely to be other space objects/debris within a relatively short distance of the avoidance satellite, this project is only focused on detecting the primary space debris of concern for collision. Therefore, the testbed must be capable of detecting a single object, while other objects comprising the test environment shall not prohibit sensing of the primary object.

DR 2.1.2: *The sensor shall be capable of sensing an incoming sphere of 50mm or greater diameter.*

Motivation: Designing a sensor solution requires knowledge of the object to be detected and the capable range of detecting. This design requirement focuses on the size of object to be detected, and is therefore coupled with DR2.3, which specifies the domain of detection capability. A 50mm diameter object is chosen to enable object detection with off-the-shelf sensors, eliminating the need to develop a custom sensor. Therefore, this design requirement allows the project team to focus on development of an integrated detect and react algorithm.

DR 2.2: *The sensor shall be capable of returning distance and bearing measurements of the incoming object.*

Motivation: In order to develop a reaction maneuver to avoid a collision, the testbed system must be capable of propagating the current state of the object forward in time. This procedure requires knowledge of the current object distance and bearing, or heading, angle. With these data points at distinct time steps, the velocity of the object can be computed over multiple timesteps to propagate the position forward in time.

DR 2.3: *The sensor shall be capable of sensing within the entire testbed domain.*

Motivation: As discussed in DR2.1.2, the capable range of detection is a critical parameter is sensor selection. This design requirement specifies the operating domain to place a limit on detection distance to enable sensor selection. This design requirement will also influence the design of the testbed environment itself, where the overall size is motivated by the ability to test a lower incoming velocity along as well as relative ease of transport.

DR 2.4: *The sensor field of view shall be at least 30 degrees.*

Motivation: This design requirement is motivated by the ability to test a range of collision scenarios. By having the ability to vary the incoming object angle by 30 degrees and being able able to sense within this heading range, a variety of collision scenarios can be modeled.

DR 2.5: *The sensor shall be capable of detecting an object while the maneuvering system is operating.*

Motivation: To enable a continual improvement in the state estimation error the sensor must be able to take measurements as the incoming object approaches. If the collision avoidance algorithm determines that a maneuver is necessary, then the system may start to maneuver while the sensor is still taking measurements.

DR 2.6: *The detection sensor sampling rate shall be high enough to drive the 2σ covariance ellipse into an avoidable region, where the avoidable region is defined as the 2D domain of the max distance the maneuvering system can operate in at any given time until collision.*

Motivation: This design requirement is fundamentally stating that the covariance of the incoming object must be small enough in a certain direction to enable the maneuvering system to travel outside of the ellipse. Therefore, this design requirement is coupled with the avoidance requirements as specified under Functional Requirement 4 (FR4).

DR 2.7: *A reorientation maneuver shall not be required for the test system to sense an incoming object.*

Motivation: This project assumes that no external data is provided to the avoidance algorithm, meaning that all information regarding detection is obtained solely from the onboard sensor. Therefore, the sensor must be capable of sensing an object approaching from any direction as specified by the dimensions of the test environment (and field of view).

3. Requirements Derived from Functional Requirement 3

FR3: The test system shall be capable of determining if a collision will occur

DR 3.1: *The test system avoidance algorithm shall be capable of estimating the state of an incoming object from sensor data, with state estimation error less than the 2σ estimate bound.*

Motivation: With the ultimate goal of enabling a collision avoidance maneuver, it is necessary to have a certain degree of confidence in the position estimate obtained through sensor readings. Therefore, a 95% confidence (2σ) is deemed appropriate for this project requiring a high degree of confidence in position estimate, without requiring custom sensor suite solutions.

DR 3.2: *The test system avoidance algorithm shall be capable of predicting collision probability from state estimation data.*

Motivation: While the first step in the detect and react procedure is to identify an object and perform state estimation, the testbed must then use this data to identify whether a collision will occur. This design requirement focuses largely on the software component of this project that requires intelligent decision making on whether a collision will occur.

DR 3.3: *The test system avoidance algorithm, maneuvering hardware, and sensor shall be capable communicating data between subsystems.*

Motivation: As discussed in DR 3.2, collision determination will occur following sensor readings. This collision probability determination must then be transferred to a physical maneuvering reaction. This design requirement specifies the integration of detection, determination, and maneuvering aspects of the project testbed so as to prevent an arbitrary maneuver.

4. Requirements Derived from Functional Requirement 4

FR4: The test system shall be capable of avoiding a physical collision using motion characteristic of a thruster response in orbit

DR 4.1: *The test system shall be capable of receiving and acting upon maneuvering commands based on received sensor data as an object is incoming (a live scenario).*

Motivation: The live scenario aspect of this project will be critical to the design solution. With a live scenario, decisions through the avoidance algorithm must be made in a limited amount of time. This design requirement is specifying that our product must be capable of both creating and executing maneuvers as the incoming object is approaching without receiving any pre-defined sensor data prior to the test.

DR 4.2: *The test system shall generate sufficient force to avoid a collision with the covariance ellipse of the incoming object state estimation for a subset of the possible relative velocities (0m/s to 2m/s).*

Motivation: As discussed under Functional Requirement 2 (FR2), the sensor sampling rate effects the size that the collision covariance ellipse can be driven to. If this covariance ellipse is not driven to a size that the maneuvering system is capable of moving outside of for a given time until collision, then a collision will occur. Thus, the overall motivation for this requirement is specifying that avoidance of an incoming object must be possible for at least the subset of the possible 0-2m/s velocities that the testbed will be capable of creating.

DR 4.3: *The test system shall produce a maneuver that does not deviate more than 5% in acceleration from a chosen scaled orbital response acceleration curve*

Motivation: While a direct scaling of an orbital collision scenario is not the focus of this project, the maneuvering system must be capable of maneuvering with a specified acceleration profile to mimick that of a full-scale orbital satellite. A baseline full-scale thruster acceleration profile has been modelled in Appendix section X.C. This requirement places a bound for which the maneuvering system must be capable of representing this acceleration profile, independent of the maneuvering solution.

B. Design Solution Overview

The following sections outline the various components of the design and describe how and why each is implemented into the full system. These include the software and programs used and written, the necessary electronics components, the LiDAR sensor, and all the various mechanical components such as the launch ramp, the sensor mount and protector and the gantry system. The decisions that were made in choosing each component is located in Appendix XI in the form of trade studies and scoring sheets.

C. Design Solution: Sensor

1. Sensor Capabilities

Authors: Sam

The selected sensor was the RPLiDAR A2M8 360° Laser scanner. Table 2 shows the capabilities of the selected sensor. This sensor was selected due to its effectiveness in the scale of the testbed environment as well as its scan rate and angular resolution. This allows the sensor to accurately detect the incoming object quickly to meet the associated requirements. When the trade studies were performed for off the shelf sensors, the closest competitor to LiDAR was radar. However, off the shelf radar sensors did not have range capabilities as good as the available off the shelf LiDAR sensors and range was the highest weighted criteria. For this reason, LiDAR was chosen for this project. If the sensor were to be scaled up, a different type of sensor or a sensor suite may be chosen.

Table 2 LiDAR Capabilities

Distance Range	0.15 - 12m
Scan Rate	5 - 15 Hz
Sample Frequency	2000 - 8000 Hz
Angular Resolution	0.45 - 1.35°

2. Measurement Clustering

Author: Conner

Typical Kalman filter variants assume a single measurement per object per sensor. However, the combination of the angular resolution and scan rate of the A2M8 sensor allow for multiple object detections through a full 360° scan. In this case, we elect to cluster sensor data to provide the filter with a single measurement of the foreign object per scan. A simple clustering algorithm was developed for range-bearing measurements. This clustering is performed after the bounding box has removed unnecessary detections in the testing environment. The range component of the measurement can be a simple mean, but averaging the bearing component imposes further difficulties when the data points are clustered around the $\theta = 0/2\pi$ boundary. In order to overcome this we impose the following algorithm:

$$\xi = \frac{1}{N}\sum_{i=1}^{N} e^{i\theta} \tag{1}$$

$$\theta_{\text{mean}} = \arctan\left(\frac{\text{Im}(\xi)}{\text{Re}(\xi)}\right) \tag{2}$$

After averaging the measurements over a full 360°, the single measurement is then ready to be passed on to the EKF for state estimation.

3. Bounding Box

Authors: Angel, Conner

A bounding box was created to filter out known sensor measurements of the incoming object from unwanted measurements of the environment. A preliminary scan and the known gantry dimensions are utilized to create this box. The sensor sends range and bearing measurements, and the range, r, was multiplied by the cosine and sine of the bearing to obtain the x and y positions corresponding to the measurements, respectively, as expressed in the following equations:

$$x_{\text{meas}} = r\cos(\theta) \qquad y_{\text{meas}} = r\sin(\theta)$$

The resulting x and y data is relative to the sensor, which is considered the origin of the system. In order to correctly define where the sensor is in space, the geometry of the test setup was utilized. At the beginning of a test, the gantry runs through a homing procedure which moves the sensor to the same center location at the start of every test. Using the knowledge of where the sensor is located at the beginning of every test and the geometry of the testbed itself, borders are created that define a rectangle within the testbed with corners at (-450, -425) mm, (-450, 425) mm, (1300,-425) mm, and (1300, 425) mm. Any data points that lie within the bounds of this rectangle can be saved as object data, and any data that lies outside these bounds can be thrown out as superfluous. Figure 3 provides a graphical representation of how sensor data interacts with the previously-described bounds present. In this plot, measurements are color-coded based on what they describe and the implemented bounding box is illustrated. Note that in this example the sensor is not located at the homing position.

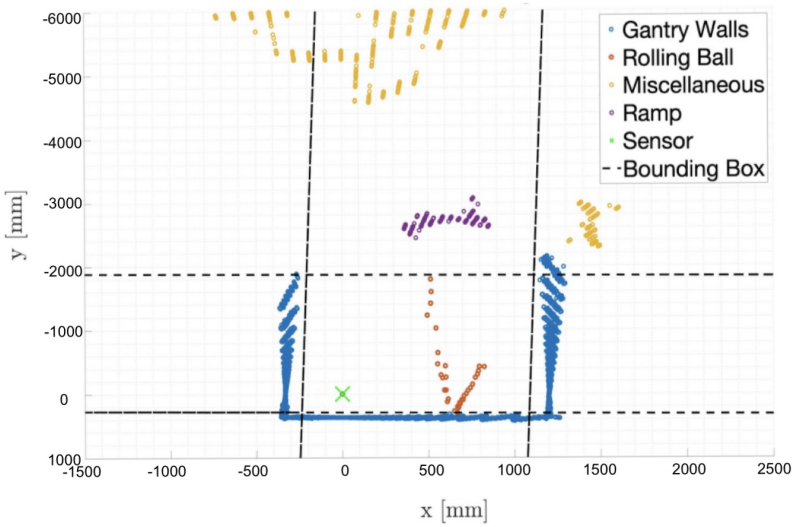

Fig. 3 Plot of LiDAR Sensor Surroundings.

4. Sensor Model

Authors: Angel

In order to scale the sensor up for space applications, a simulated sensor was used. To generate the modeled sensor data, the MATLAB sensor fusion toolbox was used with the target vehicle set as the object and the ego vehicle set as the sensor. Within the sensor fusion toolbox, sensor parameters can be passed in to define the simulated sensor characteristics. In order to simulate the LiDAR sensor that was used on the testbed, the performance parameters of the sensor model were set to be the same as the performance parameters of the physical sensor used in actual testing. In particular, the scan rate was set to 10 Hz and the sampling rate was set to 4 kHz. The incoming object was simulated as a 50 mm sphere mesh that followed a constant-velocity linear trajectory. When live collision scenarios were performed, measurements of the height that the ball was dropped from, the relative X and Y offset of the ramp to the sensor, and the angle that the ramp was at were recorded; these measurements defined the ball's trajectory and provided all of the information needed to simulate the incoming trajectory of the ball in the simulated sensor scenario.

D. Design Solution: Software

1. High Level Architecture

Author: Trace

The implementation of a detect and react algorithm requires several interconnected software components. Development of a large software system requires that each component be designed to fit into a larger system architecture. To

facilitate this, the principle components were identified and the high level architecture in figure 4 was designed. The system starts in an initialization phase, where the serial connections to the sensor and micro-controller are set up, the measurement bounding box is built, the state estimation and maneuver generation systems are configured, lookup tables are loaded into memory, a global logging system is opened, and the main loop is configured with data storage containers and a user exit condition. The main loop proceeds by first reading both serial ports, which must be done explicitly because MATLAB does not provide any software interrupts. Measurements are then filtered through the bounding box and clustered. Clustered measurements are then sent to the state estimation and prediction systems for processing. Collision predictions from this system are sent to the guidance system to determine the probability of collision and build a maneuver system if that probability eclipses the tune-able threshold. Maneuvers are then reduced to a computationally expedient form and sent to the micro-controller for execution. The main loop ends by saving measurements, state estimates, collision estimates, and maneuvers to their respective storage containers and checking if the user has decided to exit the loop. On exit the storage containers are saved to the file system, the sensor motor is stopped, and the serial connections are closed. Each of these components will be described in detail below.

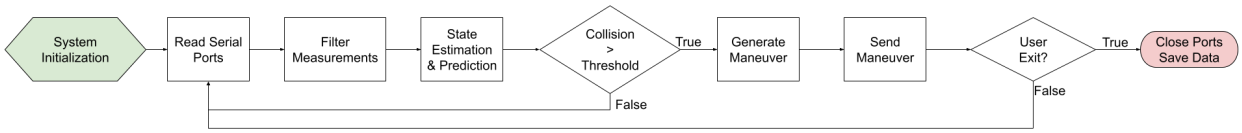

Fig. 4 CAST Software Architecture

2. State Estimation

Authors: Jason, Conner

To track the motion of the ball, CAST used an Extended Kalman Filter state estimation algorithm. The use of an EKF was necessary because the measurements provided by the sensor are given in range-bearing format, introducing non-linearity into the measurement equation through sines and cosines. It is also advantageous to use an EKF because this algorithm allows for sequential processing of the states, as opposed to relying on stored data.

The EKF implemented by CAST uses the typical predictor-corrector method. The prediction step predicts a mean state using a state transition matrix. The predicted covariance is calculated using the previous covariance, the state transition matrix, and process noise, Q_k. The process noise represents the uncertainty of the state of the collision object. The corrector step utilizes the predicted covariance, measurement jacobian, and sensor noise matrix, R, to calculate the Kalman gain. This gain is then used to calculate the new state based off the innovation and the corrected covariance. The measurement noise matrix is defined by the uncertainty of the sensor. The equations used in the EKF are shown in (3)-(9).

$$\tilde{F}_k|_{\hat{x}_k^+} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

$$\tilde{H}_{k+1} = \begin{bmatrix} -\frac{y-y_s}{(x-x_s)^2(\frac{(y-y_s)^2}{(x-x_s)^2}+1)} & \frac{1}{(x-x_s)(\frac{(y-y_s)^2}{(x-x_s)^2}+1)} & 0 & 0 \\ \frac{2(x-x_s)}{2\sqrt{(x-x_s)^2+(y-y_s)^2}} & \frac{2(y-y_s)}{2\sqrt{(x-x_s)^2+(y-y_s)^2}} & 0 & 0 \end{bmatrix} \tag{4}$$

Prediction Step

$$\hat{x}_{k+1}^- = \tilde{F}_k \hat{x}_k^+ \tag{5}$$

$$P_{k+1}^- = \tilde{F}_k P_k^+ \tilde{F}_k^T + Q_k \tag{6}$$

Correction Step

$$\tilde{K}_{k+1} = P_{k+1}^- \tilde{H}_{k+1}^T [\tilde{H}_{k+1} P_{k+1}^- \tilde{H}_{k+1}^T + R_{k+1}]^{-1} \tag{7}$$

$$\hat{x}_{k+1}^+ = \hat{x}_{k+1}^- + \tilde{K}_{k+1}(y_{k+1} - h[\hat{x}_{k+1}^-]) \tag{8}$$

$$P_{k+1}^+ = (I - \tilde{K}_{k+1}\tilde{H}_{k+1}) P_{k+1}^- (I - \tilde{K}_{k+1}\tilde{H}_{k+1})^T + \tilde{K}_{k+1} R_{k+1} \tilde{K}_{k+1} \tag{9}$$

The filter used has a few inherent issues. Due to time constraints, CAST was not able to modify the scan rate of the sensor during testing; this led to sub-optimal performance of the filter, as will be discussed further in section V.D.5. Additionally, the filter assumes that the ball is a point object, which introduces a bias into the system. Because measurements are taken on the front half of the ball, it is possible for measurements to introduce up to 2.54 cm of error from the center of the ball. This can be improved using extended object tracking as discussed in Section VIII.A.1 below.

Using the EKF, CAST can determine where the ball is at any point in time. To determine where the ball will be at the point of collision, the most recent state estimate is used to predict when the ball will cross the gantry Y axis. Using one prediction step of the EKF, expressed mathematically in (5) and (6), a state and covariance could be predicted.

3. Probability Calculation

Authors: Jason, Conner

After the state estimation software provides a prediction of the state and uncertainty covariance at the time of closest approach, the probability of collision could be calculated. This was accomplished using MATLAB's built in function **mvncdf**. By using the provided mean and covariance of the state, the function integrates over a rectangular area to determine the probability of the collision object being in the that area. The area of integration used is the length and width of the sensor mount plus the radius of the collision object. Figure 5 shows a contour map of the PDF with the outline of the integration area. After determining the probability of collision, the probability is compared to the probability threshold to determine if a maneuver is necessary. The probability threshold is set at 50%, as simulations have shown that this threshold provides enough time for the spacecraft to maneuver outside the 2σ uncertainty ellipse.

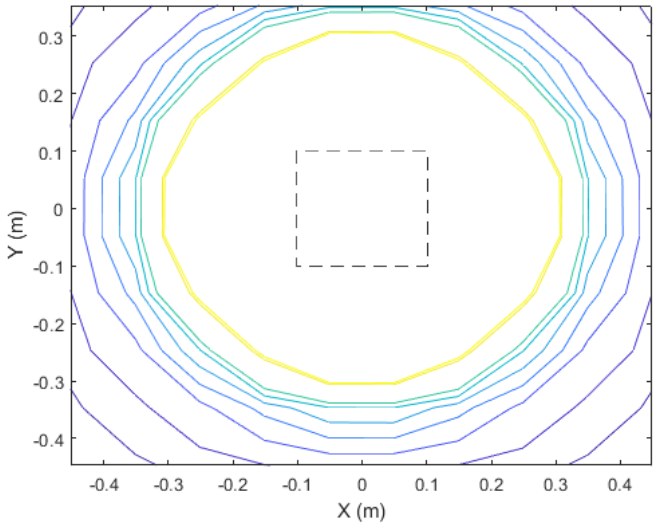

Fig. 5 Collision PDF with Integration Area

4. Maneuver Planning

Authors: Jason

Once it was determined that a collision would occur, CAST calculated a maneuver to avoid the collision. The first step of this process was to determine the direction of the maneuver. By using MATLAB to numerically calculate the gradient of the PDF, the value of the gradient at the location of the sensor could be determined. The value of the gradient was used as the maneuver direction. After selecting the maneuver direction the software would then calculate the burn time. To do this, a full-scale satellite collision scenario was simulated at an orbit of 1300 km, with a time of 30 seconds until collision. A 2σ ellipse for the local position of the ball was calculated and then scaled up and projected onto the collision plane of the satellite scenario. The software would then iterate over burn times ranging between 0 and 30 seconds to determine minimum burn time to avoid the collision. This process of scaling up from the test-bed scale to the full orbital scale was done so that the maneuver determination process in the testbed software retained fidelity to the process which would be implemented on an orbital avoidance system. Once the minimum burn time was determined,

the calculated trajectory was then scaled back down to testbed dimensions and timescales and subsequently enacted.

5. Latency Considerations

Authors: Trace

Since the CAST algorithm must run in real time, latency was a critical consideration. One significant issue was the status of MATLAB as an interpreted language and the lack of viable interrupt routines. This restriction was unavoidable once MATLAB was selected as the primary language for development. Additionally, the CAST algorithm contains a large amount of numerical calculations at every step of the process. Calculation speed was increased by vectorizing computations at every available point. The primary speed issue occurs in the maneuver generation step. In its naive implementation the maneuver step requires integrating two-body gravitational dynamics to determine if the proposed maneuver will successfully avoid the collision. This integration is extremely costly and cannot be performed in real time. To mitigate this, a lookup table of maneuvers on a one degree and one second directional and burn-time discretization was pre-computed and loaded into memory from the primary computer's file system. The final latency concern exists in transferring the maneuver solution to the micro controller for execution. To combat excessive transfer times the maneuver is reduced to a first degree polynomial and transferred as a pair of single precision floating point numbers. A first degree polynomial was chosen because it is the lowest degree polynomial that can replicate the desired motion within the required error. A number of additional improvements are still available; the polynomial regression can be sub-sampled from the maneuver to reduce the number of calculations required. The lookup tables can be converted into the body-centered frame from the inertial frame to avoid real-time coordinate transformations. The algorithm can be rewritten in a compiled language, and a real-time operating system can be built to avoid OS overhead.

6. Acceleration Tracking

Authors: Trace

Acceleration tracking was originally determined to be feasible by calculating the desired step delay from the dynamics of a stepper motor. However, in the final implementation this was determined to be overly complicated and unnecessary. Instead, the desired acceleration curve was integrated and a desired velocity profile was passed to the micro-controller. Upon being received by the microcontroller, a variable frequency pulse-width-modulated signal is generated to produce movement in the x and y axis of the gantry. The pulse width of this signal is inversely related to the velocity of the gantry. Instead of updating the pulse-width at every driver step, the frequency of the signal is only updated to match the desired maneuver velocity at regular intervals. Between update steps, the velocity remains constant. With a high update rate, the resulting movement profile approximates the smooth acceleration curve well.

7. Simulation Scaling

Authors: Trace

The final stage of software development was to execute the CAST algorithm on a simulated orbital collision. To do this, a pair of conjunctive orbits were generated under two-body dynamics. From these orbits a measurement set was simulated using a tracking mono-static LiDAR from MATLAB's sensor fusion toolbox. The CAST algorithm was then executed on the simulated measurements. The primary differences are that the measurements do not need to be scaled up to orbit for the maneuver determination step, nor does the maneuver need to be scaled back down to test-bed scales or transmitted to a micro-controller. Instead the new maneuver is directly integrated under two-body dynamics and the three orbits are saved for later analysis. Critically, this does not need to happen in real time and a lookup table is not necessary for the execution of the simulation.

E. Design Solution: Electronics

1. Component Selection

Controller *Authors:* Angel, Hugo

In order to control the motors on the gantry the group considered several controller options. When the team was considering different controllers the biggest decision factors were the group's experience with the language, the controller in question, and the controller's ability to interface with the other electrical components. Some other factors considered were cost, power, and timing. The languages that the group was most familiar with include MATLAB and

C++. Most team members have used or are familiar with Arduino, and this was heavily considered when looking into controllers. Interfacing directly with the computer would not be practical for the application needed because of the electrical components involved. A micro controller such as a Raspberry Pi or Arduino would be able to interface with the electronics easily but the processing power is not as high. Since both the Arduino and the Raspberry Pi were capable of performing the necessary tasks but the group was more familiar with the Arduino, the Arduino was ultimately chosen. Many connections were needed because of the large amount of components in the project, including connections from the computer to the stepper drivers and to all four gantry limit switches.* This quantity of necessary connections led to the group choosing the Arduino Mega 2560 for the controller, as this Arduino model is inexpensive and easy to work with but also has a large enough number of ports to fit the desired CAST implementation.

Gantry Control *Authors:* Hugo, Isaac

Control of the sensor package was executed using the maneuvering sub system which will be described below. The maneuvering sub system overall consists of a 2-axis linear rail gantry and stepper motors. Because the maneuvering sub system needed to be manipulated using appropriate electronic components, the team had to determine what electronic setup should be used for controlling the gantry.

The thought process for selection of an Arduino Mega 2560 as the maneuvering system controller is described above. Because this micro controller could not speak directly to the stepper motors used in the Igus design, drivers needed to be selected in order to mediate communication between the Arduino and the stepper motors. The principle decision to be made here was whether the CAST design should use motor drivers developed by Igus (the same company that was contracted to supply the gantry and motors for the design) or a third-party driver. The Igus driver that was considered was the dryve®D1 motor controller described in [5]. This controller offers numerous advantages: since it is made by the same company, it offers easy implementation with the Igus motors; they are intended to be highly user-friendly, and can thus be operated using computer software made specifically for the purpose of operating them; and they offer closed-loop implementation. In total, the Igus D1 controllers are remarkable for their ease-of-implementation. However, their exuberant cost ($450 per controller, and two would be needed—a conspicuously high price relative to the $40-$80 price range for a third-party controller listed in [6]) is not justified by the controllers' advantages; additionally, these controllers use a CAN interface, and the team was emphatically discouraged from using because of the difficulties inherent to implementing it. For these reasons, it was decided that a third-party motor driver should be used.

With the decision to utilize a third-party motor driver, selection of a motor driver focused specifically on closed-loop stepper drivers. A closed-loop stepper driver was deemed necessary to meet DR 4.3, which states that the maneuvering system must follow a specified acceleration profile with less than 5% error. If the NEMA 23 stepper motors were to miss a step for any reason, a closed-loop implementation would recognize this mistake and perform the necessary adjustments for the motor to catch back up to the expected acceleration profile. The other factor in stepper motor driver selection was the ability to interface with encoders. These encoders would enable team CAST to log the positional data over time. Finally, the stepper motor driver needed to operate at 36 V and at least 4.2 A, since this is the nominal voltage and current of the NEMA 23 stepper motors. Upon further research of closed-loop stepper motor drivers that met the above requirements the CL57T was determined to be the best option as it has an operating current from 0-8 A, operating voltage of 36 V, the ability to power 5 V encoders, and the ability to take in two encoder data channels.

A final, supplementary aspect of the electronics portion of the gantry control solution implemented in CAST was the use of limit switches to safeguard against excessive gantry motion. Two limit switches were used in the CAST design, and these limit switches were bundled with the motors that were provided to CAST as part of the gantry maneuvering system. The bundling of these limit switches with the acquired maneuvering system allowed for easy integration of the limit switches into the overall electronics system.

2. Wiring and Communications

Authors: Hugo, Isaac

Figures 6 and 7 present the electronics wiring schematic that was developed for the CAST electronics subsystem in the fall semester and later updated as clarifications and corrections arose amidst the team's work with the physical electronic components. The wiring layouts and communications protocols used in the CAST design were determined by

*It was later determined that only two limit switches would be implemented in the gantry control system, but at the time when the controller for the electronics system was being chosen it was believed that four would be needed.

the components that were selected for the CAST design, and thus extra work of deciding upon and implementing wiring schemes communications protocols was not necessary.

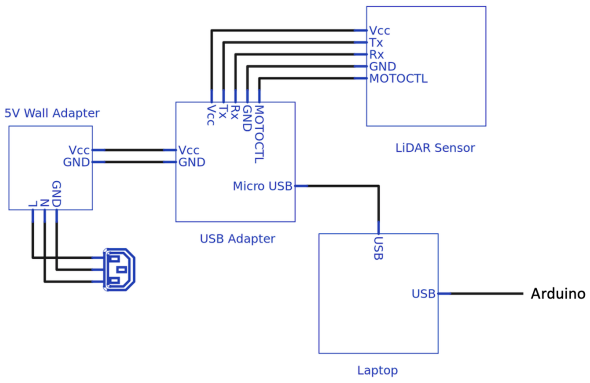

Fig. 6 Sensor Wiring Schematic

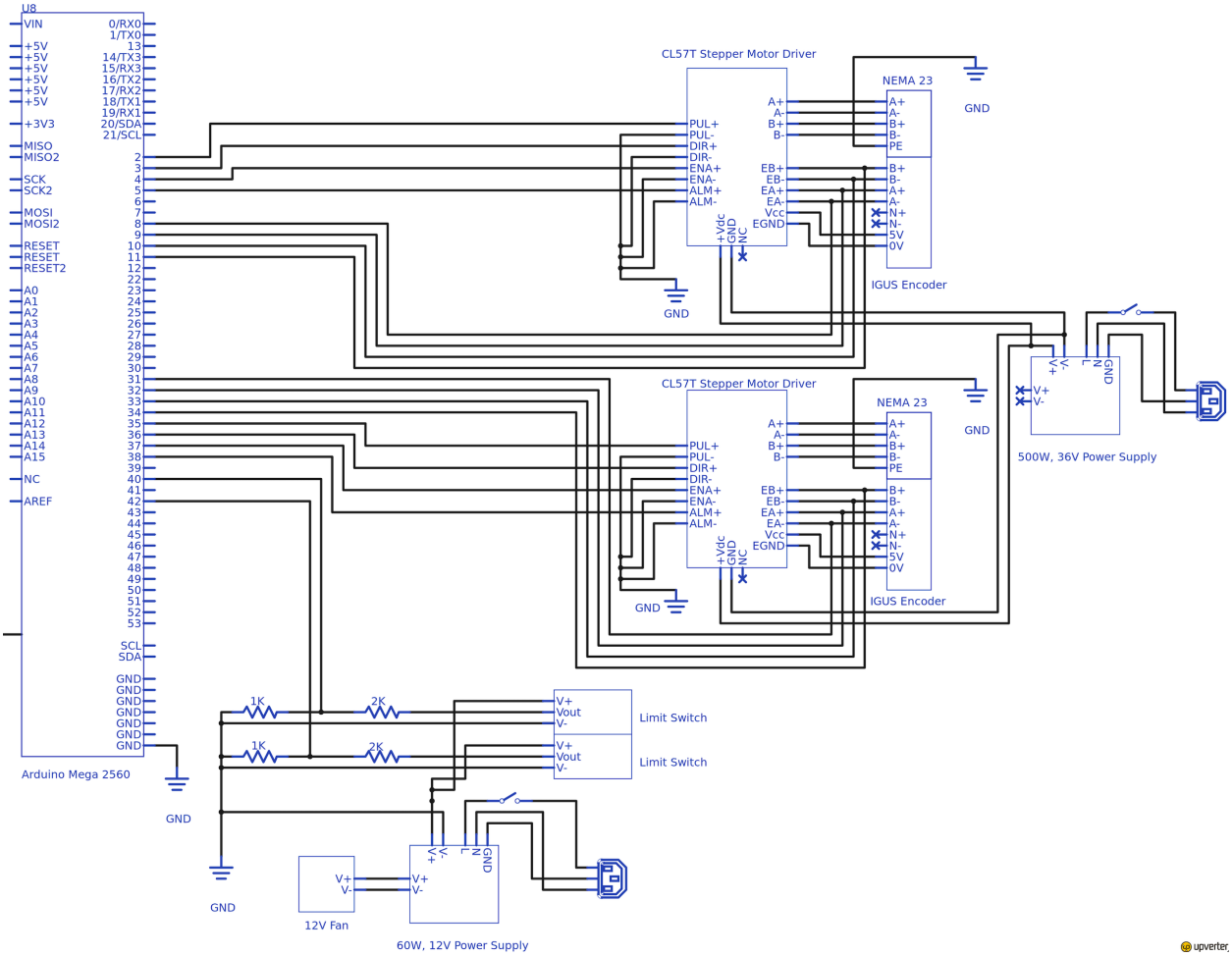

Fig. 7 Stepper Motors and Limit Switches Wiring Schematic

Figure 6 includes the components and wiring layouts necessary for the operation of the LiDAR sensor used in the

CAST design. The LiDAR sensor used by the team was specifically designed to be user-friendly, and thus the sensor wiring solution was straightforward. A USB adapter (provided with the LiDAR) connected directly to the primary laptop via USB 3.0 connection to allow for sensor communication. A 5V wall adapter connected to this USB adaptor for power (as further described in the following subsubsection), and the LiDAR sensor communicated with the primary computer (laptop) through this USB adapter using a UART communication protocol at a 115 kbps baud.

The bottom section of figure 7 presents the limit switch circuit that was used in the CAST design. Here, 12V power is directed to the two limit switches used in the gantry control system (as explained in the previous section), and these limit switches are in turn wired in such a way that they provide a zero voltage to the Arduino Mega when they are passive and a nonzero voltage when they are activated (their proximity switch is triggered). Correct wiring of the limit switches was ensured by referencing Igus' data sheet for them, [7].

The right side of figure 7 presents the wiring of the gantry control system. The two closed-loop stepper drivers used in the gantry control system are powered by a 36V power supply. The closed-loop stepper drivers are then in turn connected to the gantry motors/encoders and the Arduino Mega. Via these connections, the stepper drivers are able to operate the motors, receive data from the encoders, and send data to and receive commands from the Arduino Mega. Correct wiring of the gantry control system was ensured by referencing the data sheets for the CL57T Stepper Motor Drivers and the NEMA 23 motor-encoder combo, [8] and [9].

Commands were sent to and data was received from the Arduino Mega using a simple USB 2.0, 480 Mbps baud rate connection with the primary computer. The wiring and communications solution described here provides for the satisfaction of DR 3.3, in that the various subsystems of the CAST are able to communicate with each other.

3. Power

Authors: Isaac

In order to satisfy DR3.3, which states that the avoidance algorithm, maneuvering hardware, and sensor must be capable of data inter-communication, it was necessary to perform an electronics power analysis to ensure proper connections between each component and proper supplies of power (particularly in keeping with component voltage and current needs) to each component. Each electronic component and its associated operating voltage/current is shown in Fig. 8. To start, the closed-loop stepper motor drivers are powered by an AC/DC 36 V power supply. The pulse commands for stepper motor steps and direction are communicated via 5 V Arduino digital I/O pins. The stepper motor driver powers the stepper motor at 36 V as well as the encoder at 5 V. The encoders located on the stepper motors are incremental quadrature encoders, which include two channels: A and B. These encoder signals are inputs to the Arduino. A total of two limit switches are operated with a 12 V power supply. The 12 V output of these limit switches is lowered to 5 V with a voltage divider. The details of each connection can be seen in the wiring diagram shown in figure 7. For a description of capable and operating voltages and currents for each component see Appendix section X.A in Figures 50 through 52.

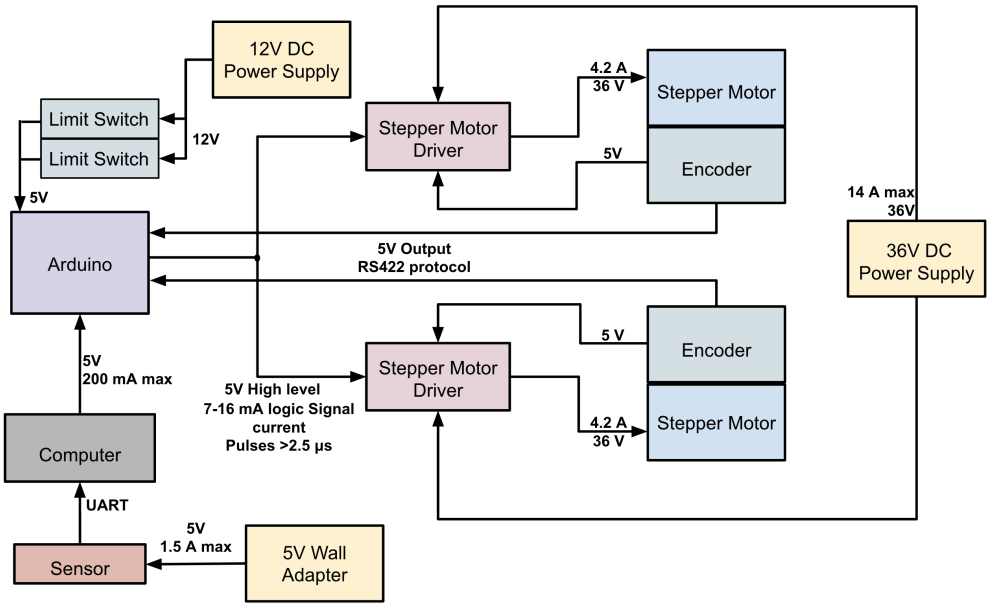

Fig. 8 Power and Current Diagram

4. Timing Delays

Authors: Trace, Hugo

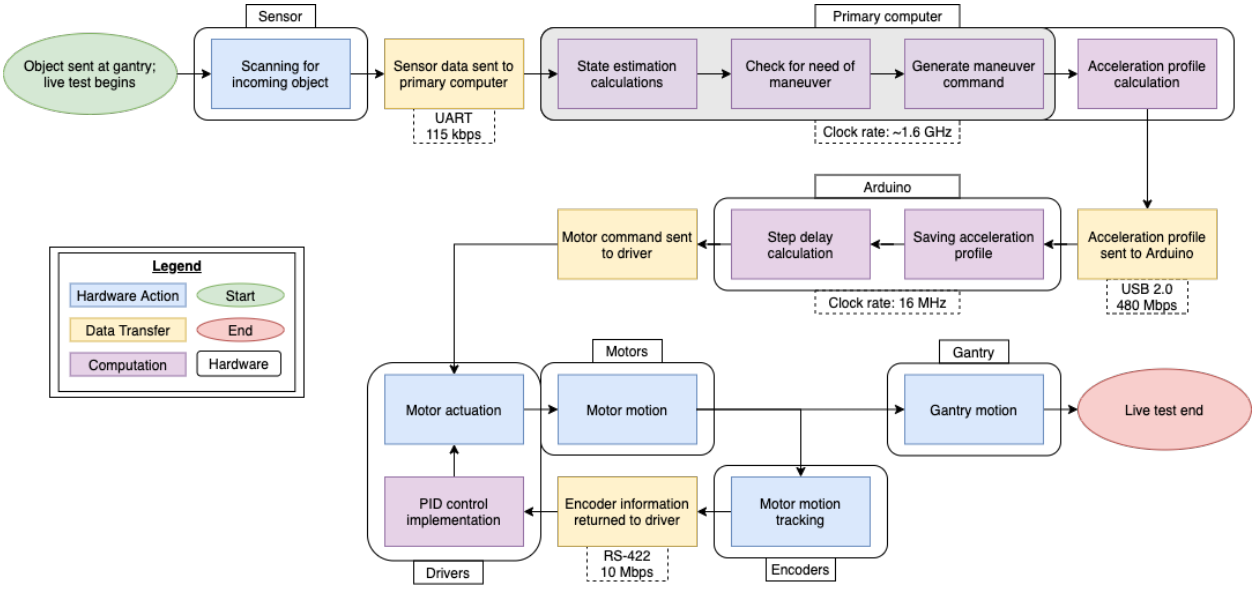

Fig. 9 A Diagram Representing the Timing Delays Inherent to the CAST Design.

The CAST system involves a live test, therefore it was imperative that timing delays in the final design be kept at a minimum so that a collision could be avoided by the physical system and the collision avoidance software thus properly demonstrated. The three kinds of timing delays present in the system are those due to internal component latency, those due to data and signal transfer times, and those due to software computation times. Figure 9 provides a visual representation of the sequence of time steps that occurs in a live test, with the three kinds of timing delays appropriately delineated using different colored blocks. What follows is a discussion of each step present in the diagram and what

timing delay it was expected to contribute to the system. Note that this discussion specifically relates to the theoretical time delays that were expected in the system based on preliminary software tests and component characteristics and performance parameters; the actual latency of the CAST system (determined by running latency tests) is discussed in the Verification and Validation section of the report.

Scanning for Incoming Object: The scanning LiDAR transmits range, heading, and a checksum at a rate of 4000 samples per second. In order to meet this specification the LiDAR must take no longer than 0.25 ms to process a data point.

Sensor Data Sent to Primary Computer: The speed of data transfer between components is determined by the connection baud rate and the amount of data to be transferred. [10] lists CAST's LiDAR sensor as using the UART communication protocol, with a baud rate of 115200 bits per second. The sensor transmits 84 byte packets representing 32 measurements, accoring to [11]. This represents a total transfer of 21 bits per measurement, requiring 0.18 ms per measurement to transfer.

State Estimation Calculations; Check for Need of Maneuver; Generate Maneuver Command: Empirical timing measurements of the state estimation and maneuver generation are provided in the latency testing section. Initial estimates placed the total time required for these steps at 4 ms.

Acceleration Profile Calculation: Acceleration profiles are pre-computed and stored in a lookup table. The memory access for these lookup tables takes roughly 3 ms.

Acceleration Profile Sent to Arduino: The same information is needed in order to determine the speed of data transfer from the primary computer to the Arduino as was needed to consider the speed of data transfer from the sensor to the primary computer. The Arduino Mega 2560 used in the CAST design uses USB 2.0 for its computer connection, and this connection, according to [12], has a baud rate of 480 mbps. It is expected that 36 bytes (288 bits) of data will be sent from the primary computer to the Arduino (using a data packet format consisting of 8 float values and 4 bytes of additional overhead such as heading information).

Our calculation for the speed of our data transfer will thus be as follows:

$$\text{(Connection baud rate)} = \frac{\text{(Amount of data to transfer)}}{\text{(Data transfer time)}} \tag{10}$$

$$\Rightarrow \quad \frac{480\ \text{Mb}}{1\ \text{s}} = \frac{36\ \text{B}}{t} \quad \Rightarrow \quad t = \frac{288\ \text{b}}{480 \cdot 10^6\ \text{b}} \quad \Rightarrow \quad t = 0.6\ \text{ns} \tag{11}$$

Saving Acceleration Profile: After receiving acceleration profile from the primary computer Arduino Mega 2560 used in the CAST design can write acceleration profile values to its memory at its full clock rate of 16 MHz for a total write time of 0.018 ms.

Step Delay Calculation: The stepper motors are driven based on a velocity profile directly via pulse-width-modulation. As such the step delay is a direct ratio of the desired velocity, and the calculation time is negligible.

Motor Command Sent to Driver; Motor Actuation; Encoder Information Returned to Driver: The motor drivers used in the CAST design take in simple voltages which alternate between high and low. As the drivers react directly to the PWM signal the transfer time between these two components is negligible.

[13] informs us that "the function of motor drivers is to take a low-current control signal and then turn it into a higher-current signal that can drive a motor." Thus, since the closed-loop stepper motor drivers in the CAST design act essentially as amplifiers, the delay between them receiving a signal from the Arduino and performing motor actuation is determined by the speed of electrical amplification, which is negligible.

The encoders output simple electrical pulses. Thus, the speed of transfer between the encoders and the closed-loop motor drivers is dependent on the speed of electricity, which provides for a negligible time delay.

Motor Motion; Gantry Motion; Motor Motion Tracking: Once the stepper motors in the CAST design receive

an electrical signal, their motion is electrical-mechanical and does not require any additional processing or computation. Thus, there is no delay for motor motion following successful motor actuation.

The gantry is driven by stepper motors in the CAST design in a linked physical dependence. Thus, there is no delay between motor motion and gantry motion.

The motor encoders in the CAST design are physically linked with the actual stepper motor, and so motor motion directly translates to the motion of the encoder's parts. Thus, there is no delay between motor motion and the encoder's tracking of that motor motion.

Final Tally and Theoretical Validation: Table 3 provides a tabulation and summation of the time delays elaborated on in the above subsections. Negligible and zero time delays are omitted, leaving only time delays which may add significantly to the total time delay of the system.

Table 3 Major Time Delays Inherent to the CAST Design

Time Step	Location	Time Delay (ms)
Scanning for incoming object	Sensor	0.25
Sensor Data Transfer	Sensor-Computer Connection	0.25
State estimation; maneuver check and generation	Primary computer	2
Acceleration profile calculation	Primary computer	2
Acceleration profile transfer	Primary computer-Arduino connection	0.0006
Saving acceleration profile	Arduino	0.018
	Total Time Delay:	<4.3

We thus have a final maximal expected time delay for the system. It represents the time necessary to detect an object, send detection data to the primary computer, perform state estimation and acceleration profile calculations, send an acceleration profile to the Arduino and save it, perform a step delay calculation, and send a motor command to the closed-loop stepper drivers for motor actuation followed by motor actuation and motor and gantry motion. This full sequence of steps will only need to be performed once. Prior to establishing a need for a maneuver, the only relevant time steps will be those from "scanning for incoming object" to "check for need of maneuver" in Fig. 9. Following the generation and transfer of an acceleration profile to the Arduino, only the steps including and following "step delay calculation" are relevant until the completion of the desired maneuver.

The maximum distance that can be traveled by the gantry in a maneuver is directly from the center of the 0.5x0.5 m gantry to one of its corners, $0.5 \sin(45\,\text{deg}) = 0.5\sqrt{2}$ m. The maximum speed that can be attained by the gantry is that which would be attained by executing a maximal velocity of 5 m/s in both directions at once, $5 \sin(45\,\text{deg}) = 5\sqrt{2}$ m/s. The process time constant for this most extreme possible maneuver is:

$$\tau = \left(1 - e^{-1}\right) \frac{(\text{Maximum distance traveled})}{(\text{Maximum speed})} = \left(1 - e^{-1}\right) \frac{0.5\sqrt{2}}{5\sqrt{2}} \quad \Rightarrow \quad \tau = 63 \text{ ms} \tag{12}$$

Utilizing a 10% sampling rule provides a maximal delay time of:

$$t_d = 0.1\tau = 0.1(63) \quad \Rightarrow \quad t_d = 6.3 \text{ ms} \tag{13}$$

Since this maximal delay time is 33% higher than our worst case estimated time delay, real time control requirements of DR 4.1 were expected to be satisfied upon the completion of construction and implementation of the CAST system.

F. Design Solution: Mechanical Design

1. Launching Ramp

Authors: Roland

The launching ramp subsystem, as determined from the trade studies, was built out of acrylic. The key design parameters for the ramp were the overall height of the ramp, the interface between the ramp and the MDF, and the curvature of the ramp. The height and curvature of the ramp were both important in ensuring that the ramp would meet

testbed velocity requirements. Since the testbed needed to be able to create a ball velocity of 0-2 m/s the ramp height had to be able to produce velocities greater than 2 m/s. These ramp heights were determined by applying rigid body motion and conservation of energy to the ball system. The derivation for ramp height is as follows.

$$mgh_0 = \frac{1}{2}m{v_f}^2 + \frac{1}{2}I{\omega_f}^2$$

$$mgh_0 = \frac{1}{2}m{v_f}^2 + \frac{1}{2}\frac{2}{5}mr^2\frac{{v_f}^2}{r}$$

$$gh_0 = \frac{1}{2}{v_f}^2 + \frac{1}{5}{v_f}^2$$

$$h = \frac{\frac{7}{10}v^2}{g}$$

This produced a necessary ramp height of 0.285 meters. Ultimately the ramp height was set at 0.381 meters (15 inches) which produced a maximum velocity of 2.3 m/s at the end of the ramp. Additionally, since the structure was a ramp and the ball needed to be travelling parallel to MDF surface at the end of it a curve was needed to change the direction of the ball. If this curve was too small the ball could see adverse effects, so the design was aimed at having a gentle curve (10 inch radius). Finally, the ball needed to not bounce upon leaving the ramp as this would affect the sensing requirements. To ensure the ball did not bounce the height of the ramp arms at the end of the ramp had to be just tall enough that the bottom contact of the ball would be placed exactly at MDF level. This was achieved by placing a leg of the 80/20 aluminum frame under the ramp to support it to the ideal height. The total design is shown in Fig 10.

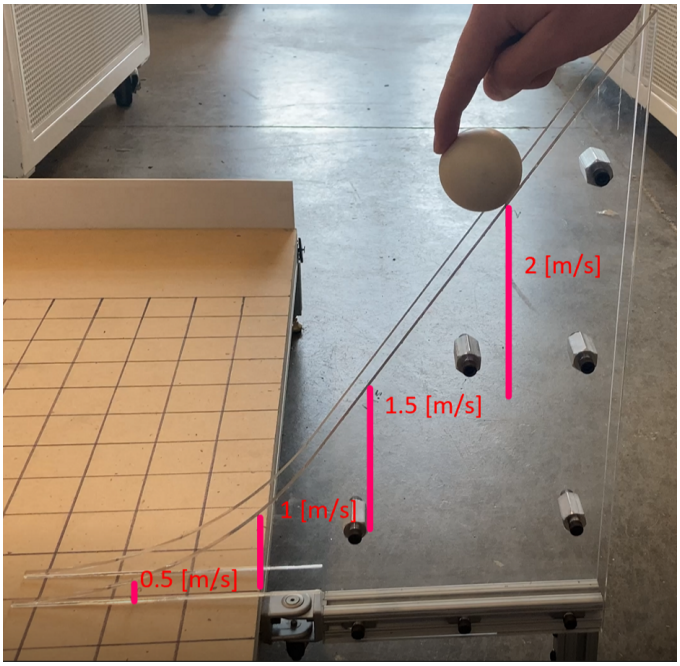

Fig. 10 Ramp Design

2. Sensor Mount and Protector

Authors: Adam

For the sensor to detect an incoming object, its field of view needed to be aligned with the center of the incoming object. To achieve this, a sensor mount and protector were designed and fabricated to interface the gantry and the sensor in a way that places the sensor in the correct location to provide adequate data. The sensor mount and protector both have a 0.25" thickness, which aligned the sensor vertically with the radius of the ball so that the middle of the sensor is in the same 2-D plane as the width of the ball. This additional step allows for the greatest view of the ball by the sensor which

ultimately increases the chance of the sensor detecting the incoming object. The sensor protector was implemented into the system as the sensor needed protection from collisions with the incoming object. The cylindrical shaped protector was designed for the sole purpose of protecting all sides of the sensor from a 2" diameter ball. The sensor mount and protector setup are displayed in Figures 11 and 12. Figure 11 shows a front view of the sensor-to-mount setup. As shown, the majority of the sensor is protected, leaving just its field of view parallel with the testbed's surface. Figure 12 displays the rear side of the set up. As shown, there is a cut in the protector to allow passage for wiring. A mount extension was designed to allow for top mounting. This was necessary as a gantry limit needed to be placed directly under the carriage.

Another key requirement for both the sensor mount and protector was to allow the sensor to detect while moving. The greatest obstacle to sensing while moving is vibration. Therefore, the mount and protector were designed to be rigid enough to further reduce amplification of the gantry's vibration. Both components were printed with Polyethylene Terephthalate Glycol (PETG), which is a semi-rigid material with good impact resistance. These components, in conjunction with the sensors relatively low mass, provides enough rigidity to not exacerbate the vibration resonance.

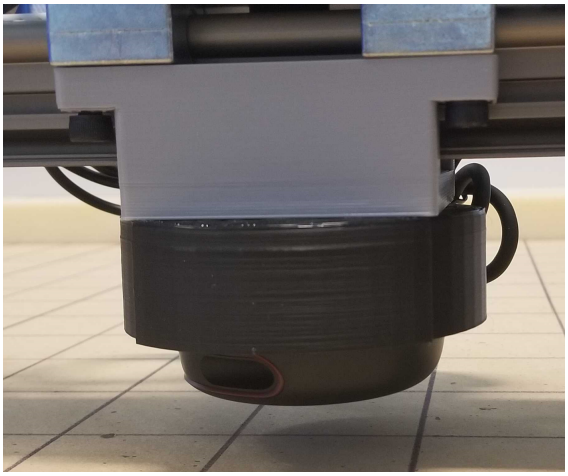

Fig. 11 Sensor Front View

Fig. 12 Sensor Rear View

3. Igus Gantry

Authors: Cameron

The Igus Gantry is the centerpiece of the physical system and consists of an x-axis with two parallel linear rails and a y-axis linear rail that is mounted perpendicularly on top of the x-axis rails. The bottom axis is bolted into the MDF platform, and the entire gantry was ensured to be square through meticulous assembly processes. Figure 13 shows a top down perspective of this crucial element of the the physical testbed.

Fig. 13 Top View of Igus Gantry

Igus produced several application reviews for various slight modifications of this gantry to ensure that it was capable of meeting the high speeds and accelerations required for this project. Further, they also ran some minor tests before the gantry was shipped to confirm that the gantry is capable of producing the type of tracking behavior that we were looking for.

Although the gantry is mostly unaltered, there were a few minor modifications that Team CAST made in order to properly adapt the gantry to our testbed environment. Chiefly, we added a 3D printed black standoff between the x and y axes on both bottom axis rails to ensure the height of our mounted sensor corresponded exactly with the middle of the collision object when it is rolling on the MDF. These standoffs are pictured in Figure 14. Along with that, the team also had to rout a 3/4 inch depression in the MDF for the lower axis motor to sit in so that the gantry would sit flat.

Fig. 14 Close Up of 3D Printed Gantry Standoff

In addition to some slight modifications, Team CAST also created a cable management system for the gantry that allowed for unimpeded movement of the gantry during operation. Figure 15 displays the two cable chain management system that passes several of the top axis wires through both to give the total 2 dimensional freedom desired in the system.

Fig. 15 Cable Management Through Cable Chains

IV. Manufacturing

A. Manufacturing Scope

Author: Sam

In the development of the testing environment, there were multiple components that were custom-manufactured. These components complemented the purchased and commercial off the shelf items. The team developed and 3D printed stand-offs to raise the upper axis of the gantry. This allowed the sensor to sense at the same level of the incoming object. These stand-offs were implemented within the custom gantry. The sensor additionally had a custom 3D printed sensor guard to protect the sensor from the incoming object as well as a 3D printed sensor mount to attach it to the top axis of the gantry. Cable chains were used to ensure that the cables to the sensor and limit switches would not become tangled upon operation of the system. These cable chains were an addition to the gantry and mounted via 3D printed mounts directly to the gantry. The electronics box was also 3D printed and housed the electronics such as the stepper motor drivers, power supply, and Arduino under the MDF of the testbed. This box was designed to ensure thermal regulation.

B. Mechanical

Author: Adam

The main mechanical component to this project is the linear gantry system. Manufacturing of this system was not necessary as the team decided to purchase a fully integrated gantry versus designing and manufacturing one. The gantry was assembled in the lab and consisted of primarily joining the two axes. In order to integrate it into the test environment, mounting brackets were utilized which fastened the gantry X-axis rails to the MDF board. Many custom components, such as the gantry standoffs, the sensor mount and protector, and the ramp required 3-D printing, laser cutting and assembly. These were all conducted in the AERO pilot machine shop. These components were designed and manufactured by the team as these parts are custom to our system.

1. MDF Alterations

To create a 1 meter by 8 meter testbed, two 4’ by 8’ segments of MDF were sized to properly create the space. The seam the MDF created rested upon a supporting segment of 80/20 framing. This seam did not affect the trajectory of the incoming object and the requirement of ball linearity was still achieved. The MDF not only needed to be properly cut, but it also had to be routed out in one area. One of the stepper motors of the gantry was not perfectly level with the two bottom rails. To allow for the rails to lay flat on the MDF, 0.3 inches of the MDF was routed to allow the motor to sit in. Finally, to attach the electronics box and mount the gantry rails to the MDF, multiple holes were drilled. Besides

the mounting holes, there was also a large hole drilled to allow for the power cables and data cable to go between the electronics box underneath the MDF to the required location on top of the MDF.

2. Testbed Framing

The testbed framing was mainly composed of 80/20 aluminum extrusion. This allowed for precise and simply assembly. With a few hex-keys the frame can be assembled or disassembled. The framing was supported in each corner with reinforced brackets and each corner was also supported with a leg. The framing had two long axes of 8' each and three short crossbeams of 5' each. The middle crossbeam supported the seam created by each segment of the MDF. Additionally, a leg was placed under the very middle of the MDF to allow for better leveling of the testbed. The 80/20 framing was used to also support the walls of the testing environment. The walls were 1/4 inch thick PVC and fit in the groove in the 80/20. With the addition of vertical braces, these walls were securing without additionally fastening hardware. Finally, to protect the back axle of the gantry from the incoming object, 3D printed backstop were added to the MDF. These backstops prevented the incoming object from repeatedly impacting the axle and bending it. These backstops did not absorb much energy from the incoming object and would send it back towards the sensor. To prevent this, a layer of foam was added to absorb the energy of the incoming object to significantly slow its movement.

C. Electronics

The manufacturing for the electronics included the printed circuit board designed by the team members on CAST as well as an electronics enclosure box to neatly stow away the electronics underneath the testbed.

1. Printed Circuit Board

Authors: Angel

The printed circuit board was designed using the program Upverter and printed with the company OshPark. The printed circuit board followed the wiring diagram outlined in section III.E.2. Figure 16 below shows the final revision of the printed circuit board. The two largest 2x7 rectangles are the SAMTEC IPL1-107-01-L-D-K parts, the 2x1 rectangle is the SAMTEC IPL1-102-01-L-S-K. Each of these connectors are added to enable easier plug-in of all necessary Arduino connections. The largest resistors are the 1k ohm and 2k ohm resistors. The smaller resistors and the LEDs on the printed circuit board were intended to be used for debugging but the printed board also had holes directly above the LEDs on the Arduino which was sufficient for the group's purposes. The 2x7 connectors were responsible for connecting the 5V power supply to the Arduino ground and connecting the encoder wires to the Arduino pins. The 2x1 connector was responsible for connecting to the voltage divider for the limit switches. The final printed circuit board is shown in figure 17 below.

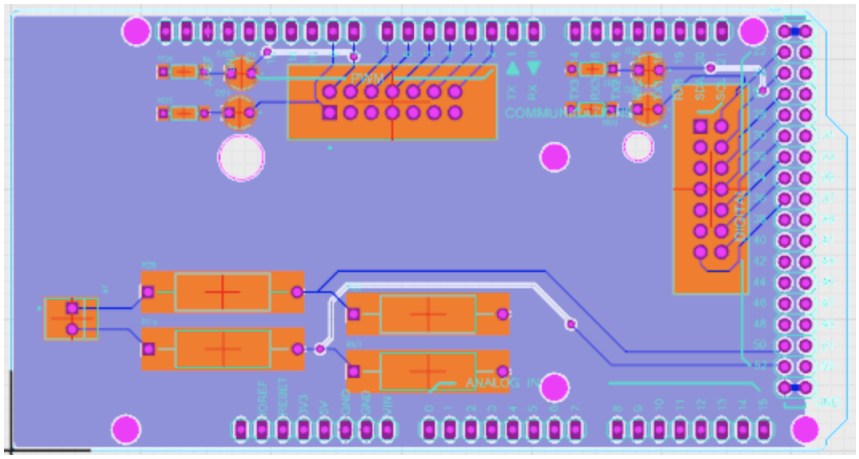

Fig. 16 Upverter PCB Design

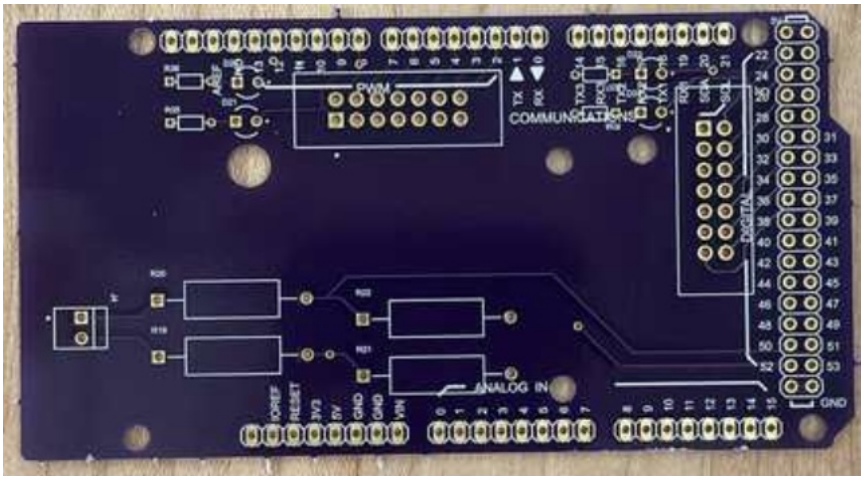

Fig. 17 Printed PCB

2. Electronics Box

Authors: Adam

An enclosure was necessary to house all the electronics components aside from the computer used to run the main algorithm. These components include 36 V and 12 V power supplies, two stepper drivers, an Arduino Mega 2360 and the printed circuit board. This enclosure was designed to perfectly fit each one of these components. To print this enclosure, its base surface area was constrained due to the LulzBot TAZ 6 3-D printers testbed bounds, which is 11 by 11 inches. The main design concern for the electronics box was the thermal requirements, as the components within have a maximum allowable temperature of 50 degrees Celsius. The material used to print the electronics box was Polyethylene Terephthalate Glycol (PETG) as it has a lower glass transition temperature than ABS or PLA, the two other considered materials. Furthermore, to compensate for the heat emitted from each of the electronics components and wiring, two of the walls act as openings to the outside environment through their cut hexagonal mesh pattern. This allows for air to flow through the box which regulates the heat. Figures 18 and 19 display the CAD model and the 3-D printed-and-fully-integrated electronics box respectively.

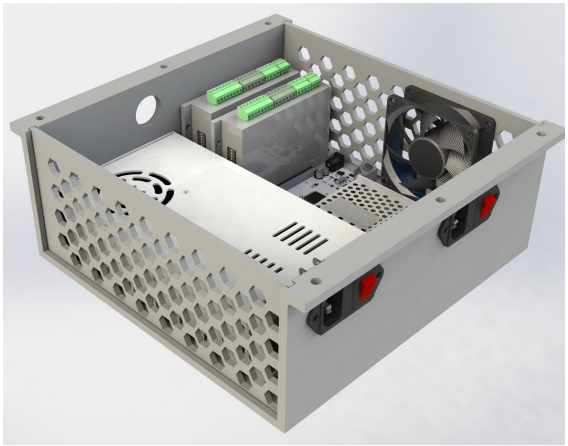

Fig. 18 Electronics Box CAD

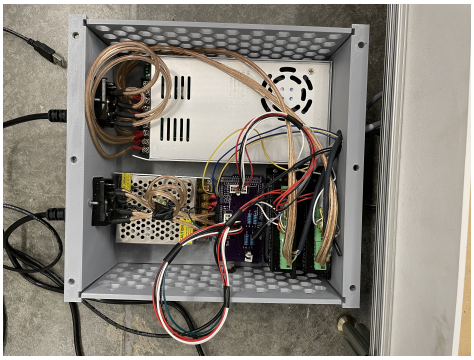

Fig. 19 Integrated Electronics Box

D. Software

Software manufacturing required the development of the sensor model, sensor communication software, state estimation, maneuver planning, and Arduino communications protocols using MATLAB, as well as development of the gantry control software for the Arduino using the Arduino IDE. Development was done by the members of Team CAST,

with the assistance of some MATLAB packages and peer reviewed algorithms. The sensor model was developed using MATLAB's sensor fusion model, which has the capability to simulate various sensors when given a sensor type and parameters. The EKF algorithm was developed based off an EKF algorithm for range bearing measurements presented in [14]. Maneuver planning was developed based on a model proposed by [15]. Arduino communications and motor control software was developed in its entirety by members of CAST. During the project, team CAST used several methods to better organize the development process. Before beginning the project, the team developed coding and naming standards to minimize confusion and allow for greater readability. To allow for version control, the team used a Github repository with branches for each development project. When changes needed to be merged into the main branch, they were reviewed by the team software lead before merging.

E. Manufacturing Challenges

Authors: Roland

Along the road to full system development there were many manufacturing challenges. Many of the initial challenges revolved around navigating the COVID-19 environment. Since a majority of the manufacturing had to be accomplished through a job shop model, the team needed to have very clear plans of how things fit together early on in the spring semester. Many of these details were not covered in the design that was developed over fall semester. This led to a massive upfront workload as the team worked to submit work requests as quickly as possible. This led to more problems in rushing the work needed. However, none of these challenges posted a major hindrance to the accomplishment of the project objectives or the final success of the project.

The first major manufacturing challenge that the team faced was in the MDF base. The team had initially planned on using a single 4'x8' piece of MDF for the base, but the final gantry received from IGUS was larger than anticipated. This meant that the testbed had to be scaled up to a 5 foot by 8 foot, a size that is not readily available in MDF. To remedy this issue, the team bought two 4'x8' sections and cut them to make a total 5'x8' section. In addition to this, the base 80/20 frame was adjusted so that the cut sections of MDF would align with the middle crossbeam of the base structure. The team had also not fully planned out what the testbed walls would be made out of; the final solution on this front was to cut pieces of PVC sheet to the required lengths and widths. To expedite these fairly basic manufacturing tasks, the team did most of this manufacturing outside of the Aerospace building using home equipment. This method allowed the team to easily manufacture things in a timely manner.

Everything else manufacturing-wise on the base structure went relatively smoothly. The team did have additional issues come up, but these were all easily solved using the rapid prototyping shop. One of these problems was the need for a sensor guard; after presenting the CAST MSR, the team saw a need for a sensor guard in order to protect the valuable sensor from possible collision with the collision object during testing. For something like a sensor guard, 3D printing with ABS was quickly seen as the best option.

An additional issue was that the team had never accounted for the issue of cable management in the fall semester CAD models. Thus, no solution was in place to ensure that cables did not tangle or impede testing. In this case the team once again used rapid development to create cable chain mounting hardware that interfaced with the IGUS gantry, and then procured cable chains at a price that was within budget. The cable chains worked well, but some challenges were met in properly managing some of the cables which had larger connections. In these situations the team spliced the cables and then reconnected them once they had been successfully inserted into the cable chains.

The final, and arguably the largest, challenge was the electronics box. Throughout the semester this was a constant nagging task that was done, redone, and done again. The difficulty was primarily in managing the many moving parts that were involved in mounting the electronics. As the team began developing software and interfacing with the gantry, the electronics needed to be changed, and thus the electronics box had to go through several iterations before coming to a point where it was functional and effective.

An additional obstacle with the electronics box was the fact that the team had to work within a hard limit on print size and cost. Developing the electronics box solely through the department would have been expensive, and so the team bought their own filament to save costs. To navigate the print size constraints, the box was printed in several parts and then joined together with fasteners. Finding all of the correct mounting hardware for all of the electronic components inside the box also proved tedious. Despite all of the complications that arose in implementing the electronics box, however, this task was accomplished and all obstacles were overcome.

F. Integration

Authors: Cameron

Once each of the subsystems had progressed to a mostly completed state, the combination of these subsystems into one fully functioning project was fairly straightforward and easy to execute. Figure 20 portrays the final product with the various important components labeled. The software was brought into the system via the main computer, which could be easily hooked up to the electronics box near the back corner of the MDF platform. The conveniently undermounted electronics box routed all necessary wiring through an opening in the MDF platform where it could then travel straight to certain motionless components, or be fed through the cable chains and attached to the moving stages on the gantry. The sensor mount allowed for easy attachment of the sensor and sensor guard to the center stage on the gantry. The ramp was bolted to a swivelling mechanism at the other end of the MDF platform. The termination of the testing area was created by screwing 3D printed barriers to the MDF and placing a thick foam mattress liner along the length of these barriers.

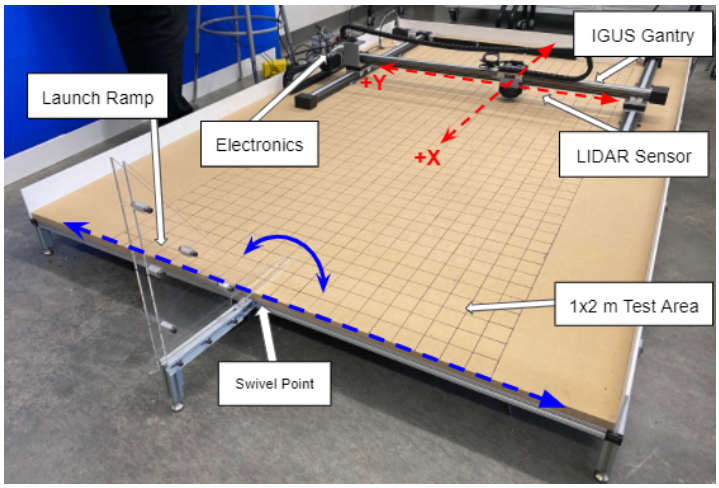

Fig. 20 Full System with Integrated Components

V. Verification and Validation

A. Testing Scope and Overview

Authors: Jason

The overall scope of testing included use of the physical testbed as well as software testing such as scaling up and error testing. Testing was organized into component level, subsystem level, and system level testing. The testing was critical to ensuring that the developed algorithm functioned as designed. Each test was performed at the Aerospace Building at CU Boulder. The testbed itself needed testing to ensure that it could properly test the algorithm created. For example, the linearity test was used to ensure that the testbed could deliver an incoming object that fit within requirements. Many tests used forms of video analysis to act as another step of verification of meeting requirements and validation of the system itself.

B. Component Level Testing

Authors: Roland

1. Ball Motion Testing

The purpose of the ball motion test was to ensure that the launching mechanism was able to create designed velocities. The test also aimed to determine the accuracy of the ramp to ensure further testing scenarios for near miss and hit would be accurate.

Setup: For this test, a camera was set up on a tripod in a position roughly 6 feet above the testbed. The testbed was then marked with a grid that was also used in later testing. This grid covered the entire testbed and had 5 cm grid

spacing. The ramp was set up in a position centered on the short axis of the tesbed. The tesbed was also fully leveled before testing.

Facilities and Equipment: The test was performed using team members cell phones to record video at 240 frames per second. The fully assembled ramp and ball were also used.

Procedure: The procedure was run for 5 tests at two speeds, 1 m/s and 2 m/s. The procedure was to start the video and then release the ball from the ramp, stopping the video once the ball was out of frame. The video frame was set up to cover the entire 8 foot length of the testbed.

Measurements: To simplify and make the results more accurate, two speed zones were identified on the testbed. Each zone was 0.5 meters long. There was one such zone right after the ramp and one centered on the gantry (1.5 m from ramp). Since the data from the camera was in video format the videos had to be manually analyzed to determine the number of frames per zone to determine the speed.

Issues with Measurements: Since the frame of the camera was maximized to show the full testbed there was lens distortion at each end. This distortion meant that using computer programs to track the ball was less accurate than the counting method. The counting method allowed for the user to choose the correct frame and manually correct for distortion and viewpoint effects.

Results: From the frame analysis the team found a 2.17 ± 0.4% percent error in velocity at 2 m/s. At 1 m/s, the team found a 4.24 ± 0.39% percent error.

Verification against Model: The model for the expected velocity decrease was based on the energy of the ball deformation. The team used an experimentally determined average acceleration for rubber balls on wood surfaces. The model is compared to the data below in figure 21.

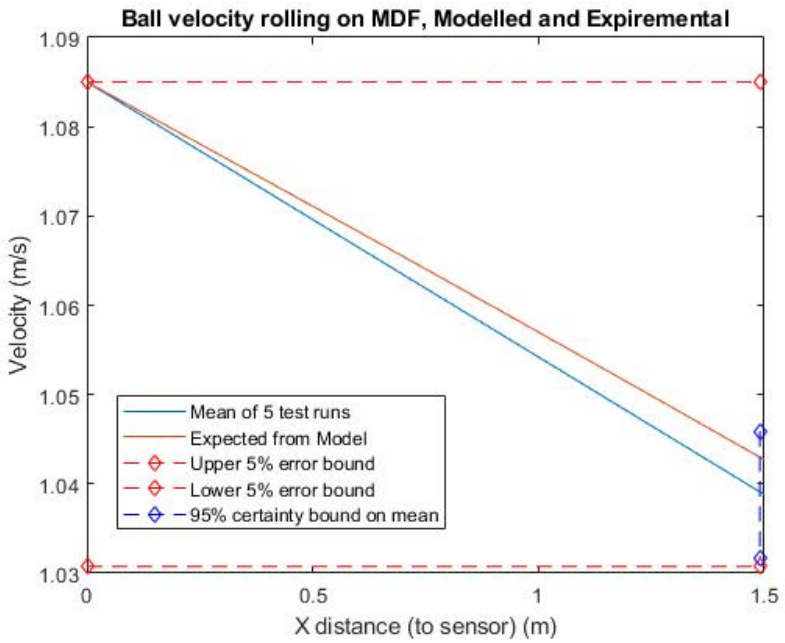

Fig. 21 Launching Mechanism

2. Lidar Sensor

The purpose of the Lidar Sensor test was to validate the received Slamtec Lidar sensor versus the manufacturer specifications. This was to ensure the sensor would meet all sensing requirements.

Setup: For this test the sensor was placed on the MDF base structure which was leveled. The sensor was plugged

into a computer that was running software which would record sensor measurements.

Facilities and Equipment: The test was accomplished at the project space in the Aerospace building using the MDF testbed without the gantry, the ramp, a computer with Matlab, and the sensor.

Procedure: For this test, the ball was placed near the ramp on the testbed and then the sensor was allowed to run for a few seconds to collect data. Then a second test was run where the ball was rolled at down the testbed. No bounding box was implemented for the test.

Measurements: The sensor provided range and bearing measurements for all detected objects (things the beam reflected off). The sensor also provided a data point for its location relative to all measurements. There were no issued in the measurements.

Results: In Fig 22 the data from the second sensor test is shown. This data shows the location of various objects on the testbed at the time of the test. From these data points the sensor was validated to be able to sense a moving object of 50 mm or less. A bounding box is shown, but again was not utilized, as multiple measurements outside of this bounding box are recieved.

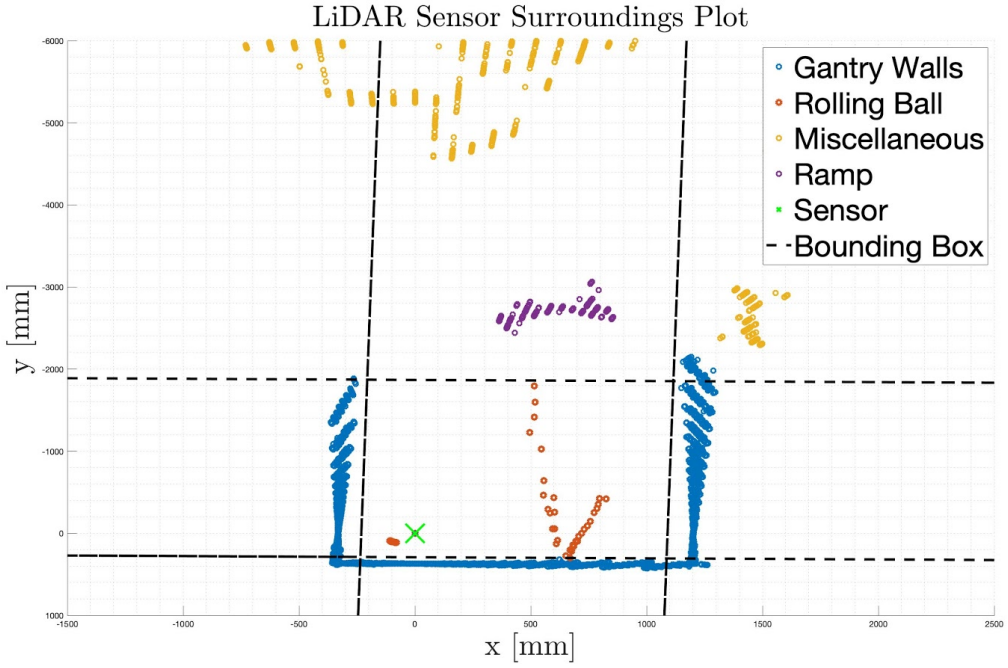

Fig. 22 Sensor Test Measurements

Verification against Model: Sensor capabilities matched those provided by the manufacturer meaning that the sensor was capable of meeting all stationary sensing requirements.

3. Software Unit Testing

The purpose of the software unit testing was to ensure that all software packages ran correctly. This was necessary so that when the software was used to control the physical hardware there were no bugs that would cause components to break. This also ensured that the speed of development of the physical testbed would be quicker as less time is needed to debug code.

Setup: For the software unit testing individual components of the software package needed to be tested. Before any testing can commence, baseline results had to be hand calculated so that there was something to compare the code against. These manual calculations were performed for known reasonable inputs into the software and provided a good

truth value for the software.

Facilities and Equipment: All software testing was able to be done on a single computer, ideally this should have been the laptop used to run the system but it did not really matter.

Procedure: For every major function within the software package known inputs are provided and then the results are compared with expected values. For something like the state estimation this is more straightforward because simulated sensor data can be passed in. For something like the motor control software there was no good way to test the software because it is overly complex to simulate the interaction with the software and the motor drivers.

Issues with Test: As stated earlier, the test should have ideally been ran on the loaned laptop that was used for the bulk of testing. This would have allowed the team to ensure the code works on the specific hardware. Additionally, not all parts of the software could be tested accurately which is why the motor integration process was more labored.

Results: Ultimately for every part of the software that a meaningful software unit test could be done, one was done. This saved the team a good deal of time on the integration part of the project since the team was confident that parts of the code like state estimation and maneuver planning worked before hand.

4. Latency

The latency testing was arguably the most important of the component tests performed. Latency testing was necessary to ensure that the assumptions that the team had made about being able to use Matlab for the state estimation and maneuver planning were accurate. This testing was ultimately a major stepping stone in the development because had it failed the entire software package would have had to potentially be redeveloped in another language or even in a real time operating system. Both of those option likely would have taken too much time to implement and could have lowered the level of success of the project. The reason latency is so important is because there is such a short time frame from the detection of the incoming object until the system has to maneuver to not get hit.

Setup: The setup for the latency was rather involved as each software component and the associated hardware had to be tested individually. The main parts that were tested were the sensor to the software, the main software loop, the software to the Arduino, and the Arduino execution. The software was set up with Matlab's built in timing libraries to quantify the system delays.

Facilities and Equipment: Following the setup the test required access to the Lidar sensor, the Arduino Mega, and a laptop. All testing was done in the Aerospace building project space as it occurred while systems were in integration.

Procedure: Each component interface mentioned in the set up was tested using the timing libraries. Each test was completed using only the necessary hardware so that testing was not interfered by other processes.

Measurements: From the timing libraries the team was able to asses a mean time for each process and an associated standard error of that mean. Both results were very helpful in ensuring that the processes were quick enough and that there were no large variations in each processes timing.

Issues with Measurements: Since each component interface was being tested individually, compound timing errors were not present. The team saw this when running the full real time testing and an unexpected sensor buffer came up. This sensor buffer meant that the data transfer from the sensor to the software was slower than expected.

Results against Model: The results from each component interface are tabulated in Fig 23. These values are also compared against the time allotment which was determined in the fall as a result of simulated latency tests. The maneuver generation and command transfer take significantly longer than modelled. This is due to growth in the computational requirements of the maneuver generation and overhead on the serial port access from both the operating system and the MATLAB procedure. However, these timing results were generated from successful maneuvers and the overall result was that the sensor to Arduino command execution timing was fast enough to enable a successful maneuver. Thus, DR 4.1 is satisfied, with maneuvering determination and implementation taking place quickly enough to allow for proper maneuvering of the sensor package.

Process	Latency Source	Estimated Time	Mean Result
Main Loop	Receive Sensor Data	0.1ms	1.0 ± 1.8e-2ms
	Estimation/Prediction Step	<2ms	0.24 ± 1.9e-2ms
	Total	2.1ms	1.24 ± 2.6e-2ms
Maneuver	Matlab Maneuver Generation	<2ms	35.5 ± 6.6ms
	Arduino Command Received and Stored	0.13ms	53.3 ± 16ms
	Arduino Step Delay Calculation	-	1.500±0.001ms
	Total	2.1ms	90.4 ± 17.3ms

Fig. 23 Latency Test Results

C. Subsystem Level Testing

Authors: Griffin

1. Command and Control Test

The purpose of the command and control subsystem tests was to show that the gantry could produce physical movement from the stepper motors. This test sought to show that both the x and y axis stepper motors could in fact move the gantry.

Setup: The setup for this test involves connecting a laptop to the stepper drivers via USB port. The laptop must have the gantry control software downloaded to run. The drivers are required to be powered.

Facilities and Equipment: This test was performed at the lab-space designated by the Aerospace Department at CU. The equipment required are a laptop, USB cable, gantry with stepper motors, motor drivers, and power supply for the motors.

Measurements: The results of this test were gathered by visually confirming that the gantry produced movement when commanded.

Results: Through visual verification, it was confirmed that the gantry did produce movement in both the x and y directions when commanded, thus the test was successful as expected.

2. Position Test

The purpose of the position test is to confirm that the gantry can move accurately based on position commands, and that the encoder feedback on the position is accurate. This test also confirms that the gantry can move over its entire range.

Setup: This test is performed by moving the gantry, then comparing the actual position of the gantry after the movement to the encoder measured position. After this is performed, the gantry is commanded to move its full expected range.

Facilities and Equipment: These tests were performed at the lab-space designated by the Aerospace Department. The equipment required is a laptop, USB cable, gantry with stepper motors, motor driver, motor encoders, and power supplies for the motor and encoders.

Measurements There were challenges initially determining which measurement technique to use to record the gantry's physical location. A measuring tape was eventually used as the method of choice for its convenience and availability.

Results: After several iterations of calibrating the encoder to distance conversion factor, the gantry's physical location was matching with the encoder feedback. After this was confirmed, the gantry was commanded to move over its full range, which it was successful in doing.

3. Velocity Test

The velocity tests were performed to confirm that the gantry can move at representative speeds. This is motivated by FR4, DR4.2, DR4.3.

Setup: This test is performed by commanding the gantry to move at its maximum operational velocity provided by the manufacturer documentation. This is performed on both axis with the motor RPM's versus time recorded on the gantry operation software.

Facilities and Equipment: These tests were performed at the designated lab-space by the Aerospace Department. The tests required a laptop, USB cable, gantry with stepper motors, motor drivers, and a power supply for the motors.

Measurements: The measurements were gathered from the stepper driver operating software. These measurements came in the form of motor RPM's and are automatically plotted with the commanded RPM profile.

Results: The results from the velocity tests showed that the stepper motors could reach their maximum expected speed of 560rpm. A plot of the results from one of these tests is shown below in figure 24.

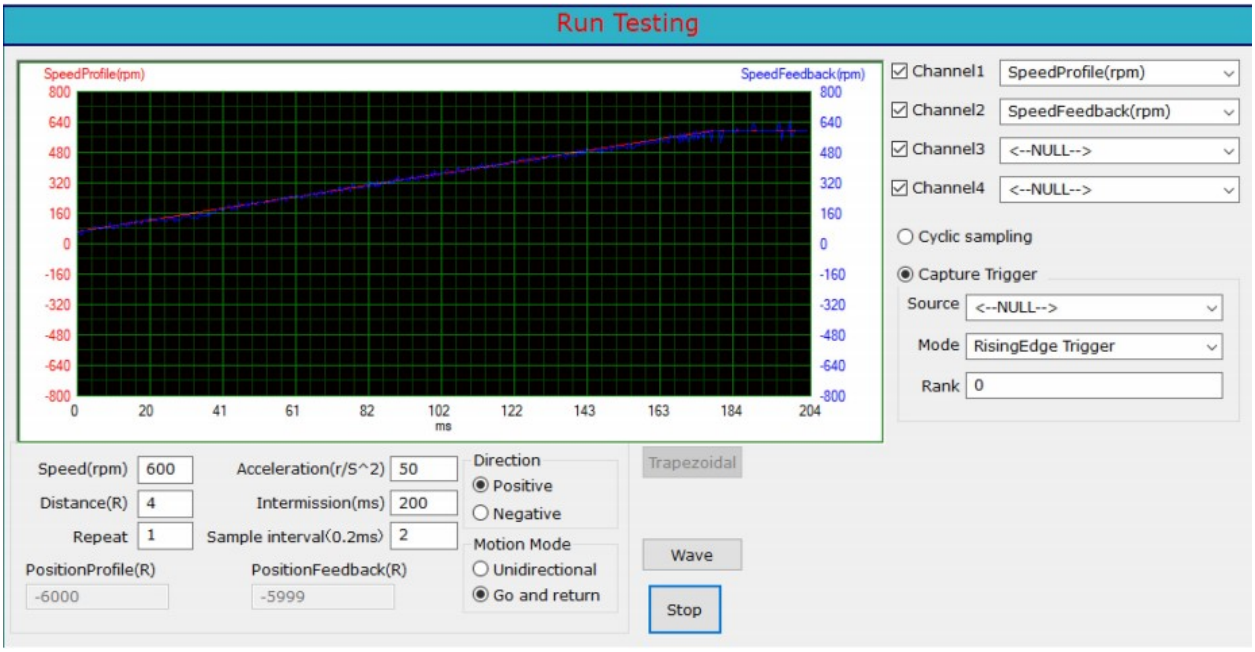

Fig. 24 Test Results from Gantry Max Velocity Test

Here it can be seen that the gantry speed feedback reaches a maximum value of 600rpm, which results in a passing velocity test.

Verification Against Model There was no model produced by the team for this test as the maximum expected speed was manufacturer provided.

4. Acceleration Test

The acceleration tests were performed to verify that the gantry can accelerate at representative rates. This test is motivated by FR4, DR4.2, and DR4.3.

Setup: This test is performed by recording the stepper motor speeds over time while commanding the gantry to move at a specified acceleration. The speed profile is compared to the commanded profile, and if the motors are able to replicate the commanded profile then the gantry's ability to move at that acceleration is confirmed.

Facilities and Equipment: These tests were performed at the lab-space designated by the Aerospace Department. The equipment required is a laptop, USB cable, gantry with stepper motors, motor driver, motor encoders, and power supplies for the motor and encoders.

Measurements: The measurements were gathered from the stepper driver operating software. The feedback from the encoders come in the form of motor rpm's which are automatically plotted against the commanded rpm profile.

Results: The figure below shows the results from one of these acceleration tests.

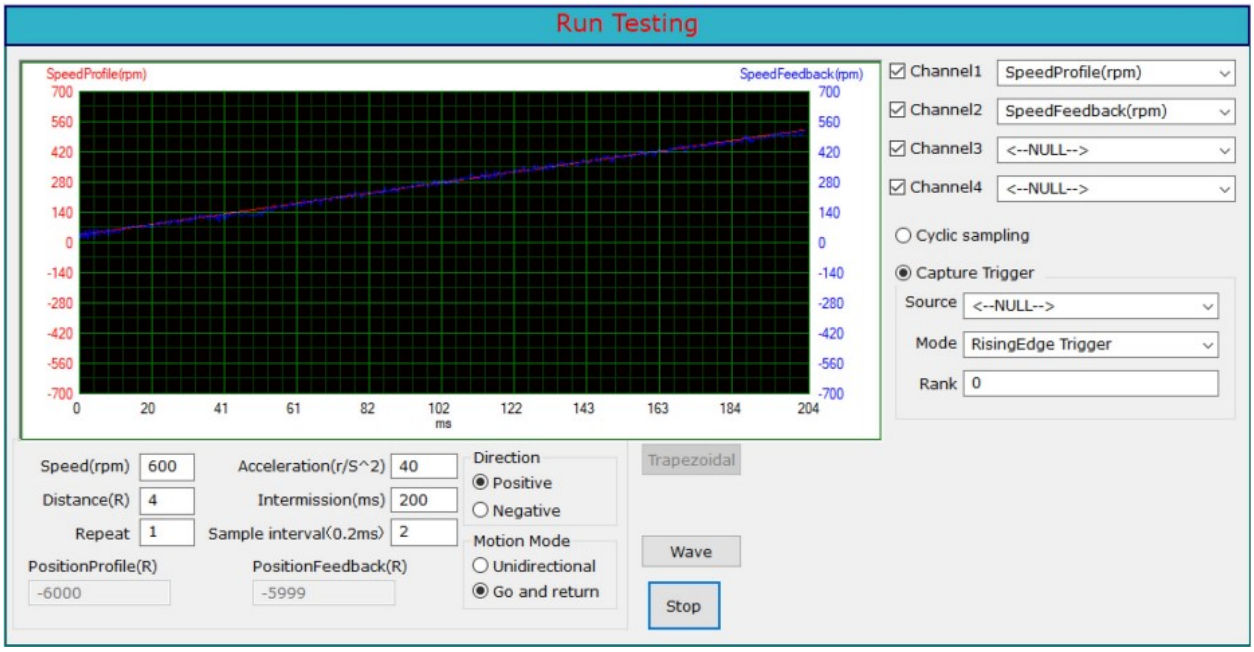

Fig. 25 Test Results from Gantry Max Acceleration Test

Here it can be seen that for a constant, $40\frac{r}{s^2}$ acceleration, that the stepper motor can closely match the commanded profile.

Verification Against Model There was no model produced by the team, as expected max gantry acceleration was received from the manufacturer. The gantry was able to successfully accelerate up to $100\frac{r}{s^2}$. This is much higher than the provided maximum acceleration by the manufacturer of $9.6\frac{r}{s^2}$. This can likely be attributed to the lack of any significant mass on the gantry during testing, as the manufacturer likely added significant safety margins to their provided number.

5. Thrust Curve Matching Test

The purpose of these tests is to confirm that representative acceleration curves, which mimic the thrust response of a spacecraft, can be produced on the gantry. This is motivated by FR4, DR4.2, and DR4.3.

Setup: This test is performed by first creating a test maneuver time polynomial for the gantry to execute. This maneuver is sent to the Arduino via a serial port connection, which handles and sends signals to the drivers to produce the movement. The encoder feedback is recorded and sent back to the laptop, again, using the serial connection.

Facilities and Equipment: These tests were performed at the designated lab-space by the Aerospace Department. The tests required a laptop, USB cable, gantry with stepper motors, motor drivers, an Arduino, and a power supply for the motors.

Measurements: Measurements are gathered from the stepper motor encoders. The encoder pulses are counted locally on the Arduino, then sent over to the laptop to be processed. The frequency of the data feedback was critical in this test, as the resolution of the recorded acceleration curve is dependent on how many data points are recorded.

Results: Over a range of test maneuvers, the average error in acceleration was 2.81%. This satisfies the 5% acceleration error requirement from DR 4.3. The results from one of these tests is shown below, where the accuracy between gantry recorded positions and the commanded curve is clear.

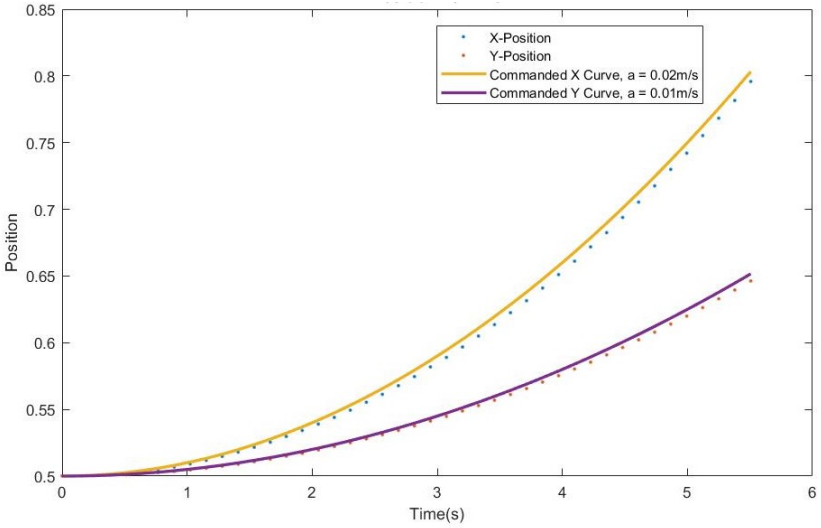

Fig. 26 Commanded Thrust Curve with Recorded X and Y Gantry Positions

Verification Against Model A preliminary model was created to model the gantry tracking algorithm's ability to replicate a commanded curve. The results of this showed that it would be possible to do this while keeping the acceleration error less than 5%, however testing was required to ensure that the stepper motors could physically perform the maneuvers required. Since the actual tests saw deviations within this 5% threshold, the model's validity was verified. In simulations, a slightly under-approximated acceleration was expected due to the curve tracking algorithm. This slight under-approximation was consistently seen throughout gantry testing, further verifying the validity of the model.

6. Vibration Analysis

The vibration analysis tests were performed to confirm that the sensor can sense an object while the gantry is moving. This test is motivated by FR 2 and DR 2.5.

Setup: This test is set up by placing a stationary ball at a distance from the sensor-mounted gantry. The gantry is then commanded to move at a velocity anticipated to be experienced during the full system maneuver, with the sensor measurements recorded on the operating laptop.

Facilities and Equipment: These tests were performed at the lab-space designated by the Aerospace Department. The equipment required is a laptop, USB cable, gantry with stepper motors, motor driver, motor encoders, a ball to be measured, and power supplies for the motor and encoders.

Measurements: The critical measurements to be taken are the ball locations as measured from the LiDAR sensor. Since the ball location during testing is stationary, only these LiDAR measurements are needed to assess the performance of the sensor during gantry movement.

Results: The recorded measurements from a vibration test, along with the actual ball location, are shown below:

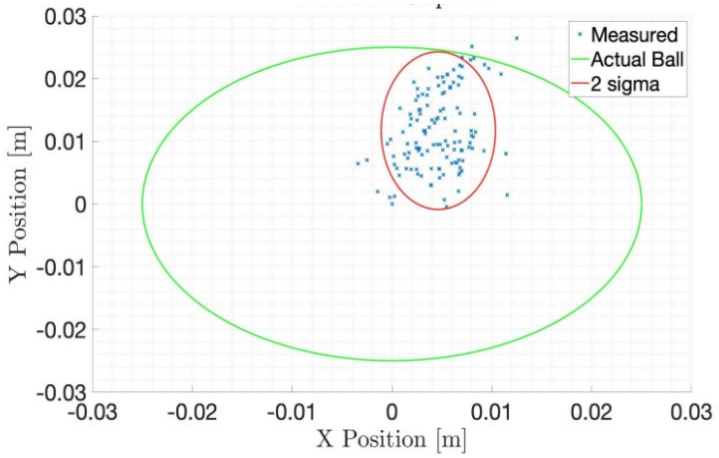

Fig. 27 Measured Ball Locations During Moving Gantry Vibration Tests

Over the course of these tests, 98.44% of the gantry measurements fall within the radius of the actual ball location, with 100% of measurements within twice the radius of the ball. These results satisfy DR 2.5 which states that the sensor must be capable of sensing an object while moving.

Verification Against Model The simulated sensor measurements of the ball with a vibrating gantry are shown below:

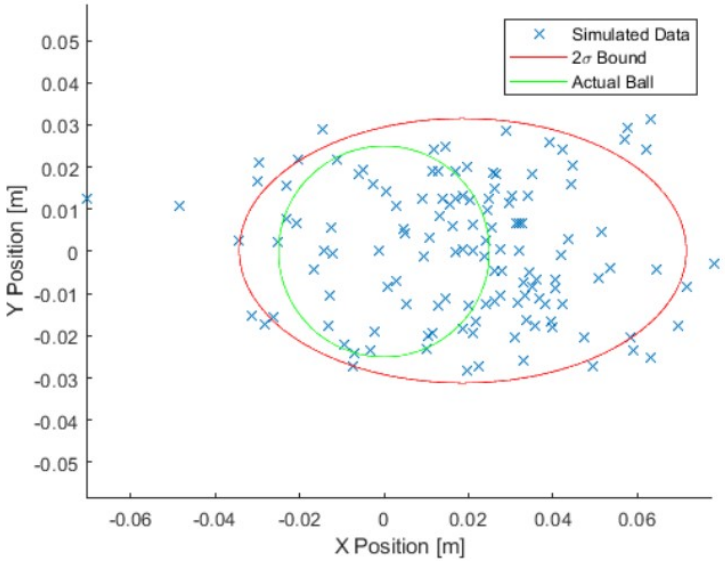

Fig. 28 Simulated Ball Locations During Moving Gantry Vibration Tests

In this simulation, 42% of the data falls within the radius of the ball. Compared to the actual results, where over 98% of the data falls within the ball radius, the model underperforms. This is due to an overly conservative gantry vibration model, which causes simulated sensor data to be more erratic.

7. Sensor/Software Integration

The sensor/software integration test is performed to verify that an incoming object can be sensed and that a real-time prediction of its trajectory can be generated. This is motivated by FR 2, FR 3, DR 2.1, DR 2.2, DR 2.3, DR 2.4, DR 2.6, DR 3.1 and DR 3.2.

Setup: The test begins by launching a ball down the ramp, towards the gantry. The sensor measures the incoming

ball's position which the software uses to estimate its state and predict the trajectory. This prediction is compared to slow motion tracking data of the incoming object obtained from video recording.

Facilities and Equipment: These tests were performed at the lab-space designated by the Aerospace Department. The equipment required is a laptop, USB cable, gantry, motor driver, motor encoders, a ball to be measured, LiDAR sensor, and launching ramp.

Measurements: The key measurements during testing are the ball positions at each video frame. This is obtained by observing the ball locations at each frame relative to the testbed grid. These locations are recorded and are used to compare to the predicted trajectory produced from the state estimation software. The process for manually recording the ball location at each frame was found to be very labor intensive, therefore only a handful of tests were performed.

Results: Shown below is the estimation error between the actual location, and estimated location for the incoming ball during a head-on test.

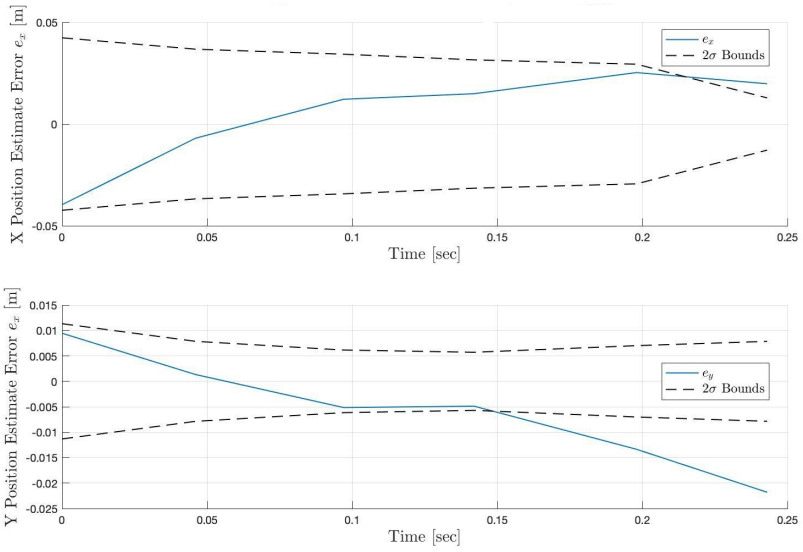

Fig. 29 Head-On Position Estimate Error vs Time

The estimation error in this test remains within the 2σ bounds for much of the time, however at the end of the trajectory the error eventually exceeds the bounds. This error is still within the radius of the incoming ball however so the results were still deemed successful.

The figure below shows the estimation error for a clear-miss test, where the ball is rolled towards the sensor at an offset y-distance.

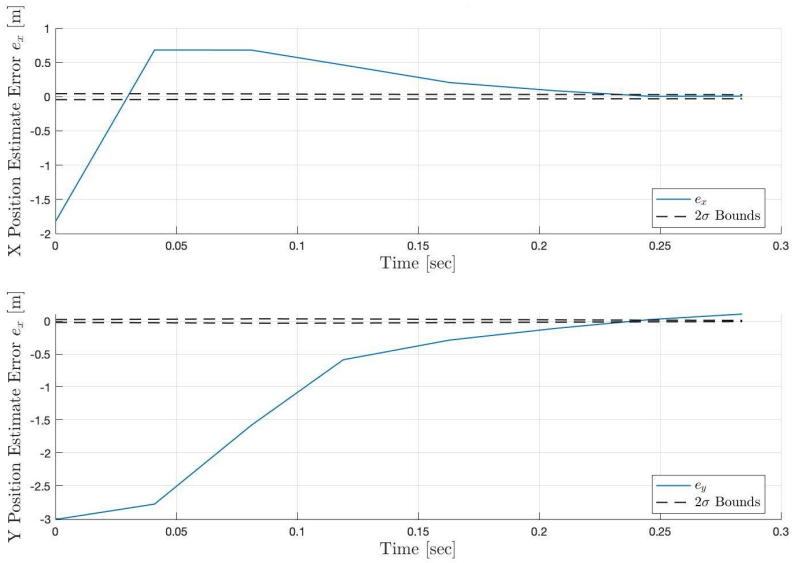

Fig. 30 Clear-miss Position Estimate Error vs Time

The state estimator is initially seeded with an object guess for a head on collision, which is why error is so high at the beginning of the test. The filter quickly recovers however, and drives the error within the 2σ bounds.

To improve the results for a clear-miss test, the state estimator was seeded an initial guess closer to the ball's true location. This yields the resulting figure shown below, where the error remains within the bounds for much of the test.

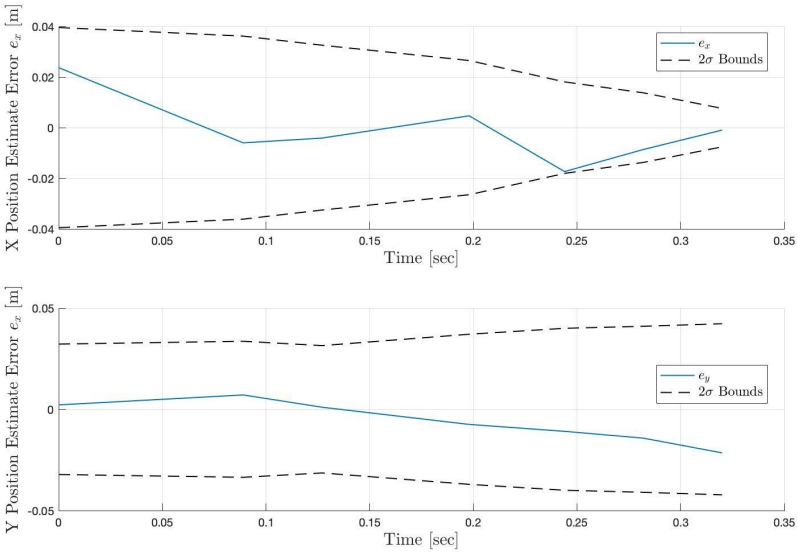

Fig. 31 Clear-miss Position Estimate w/improved Initial Guess Error vs Time

Verification Against Model Using generated sensor measurements of an incoming head-on ball passed through the Kalman filter, a simulated position error plot was created which is shown below:

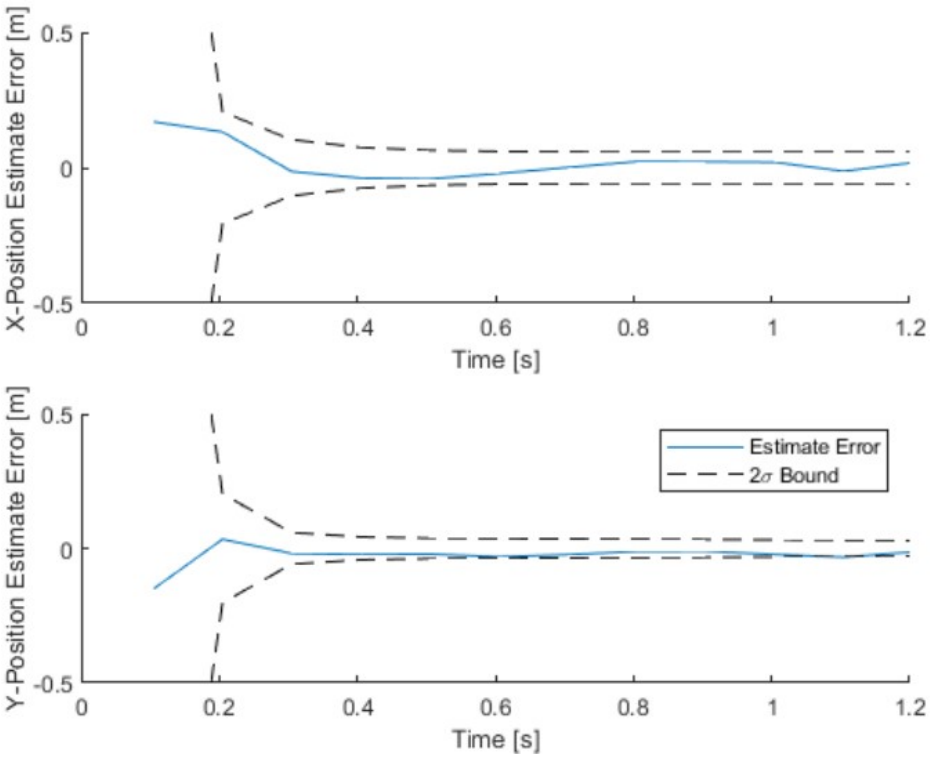

Fig. 32 Simulated Head-On Position Estimate Error vs Time

This simulation shows that the error should remain within the 2σ bounds for the entire duration of the test, which is what is also expected through intuition. During physical testing however, it was found that this was not the case. This can be attributed to the treatment of the ball as a point mass in the estimation software. As the ball approaches closer to the sensor, multiple measurements are gathered over the surface of the ball and clustered together. This introduces additional error into the estimator and is likely the cause of the measurement errors exceeding the bounds. Still, the errors remain within the radius of the ball and therefore do not largely impact the effectiveness of the avoidance system.

8. Sensor Model

Authors: Angel

The sensor model test is designed in order to have the ability to complete levels three and four of the Application Simulation from the levels of success in which a model will be needed for the sensor to scale up the to an orbital scenario.

Setup: In order to complete the test, all three considered scenarios: near miss, clear miss, and head-on are tested with the rolling ball beginning at roughly the same initial conditions for height dropped, the offset in the x direction, and the ramp angle. Using the measured parameters from the actual scenario, the trajectory of the ball is simulated and the sensor simulation is ran to observe the response from the model.

Facilities and Equipment: These tests were performed in the Aerospace Lab space that was provided to the team. The equipment that was required was a laptop, LiDAR sensor, all necessary cables to connect with the sensor, gantry, rolling ball, and the launching ramp.

Measurements: The key measurements for testing include the x and y ball distances measured by the LiDAR sensor, the timestamps for each measurement, the height the ball was dropped from, the X and Y offsets of the ramp to the sensor, the angle of the ramp, and which case is being tested. The velocity that the ball moves at is then calculated from the height that the ball was dropped from and the trajectory of the ball is determined from the initial parameters assuming a linear trajectory.

Results:

Table 4 below shows the measured data from three scenarios that were passed into the sensor model to validate the sensor model against.

Table 4 Parameters for Simulating Live Tests

Test Case	Initial Speed	Relative Ramp X Offset	Relative Ramp Y Offset	Ramp Angle
Head-on	1.88 m/s	1.5 m	0.00 m	0.0°
Near Miss	1.88 m/s	1.5 m	0.05 m	0.97°
Clear Miss	1.88 m/s	1.5 m	0.27 m	0.0°

The generated data is then passed into both the bounding box and clustering functions in order to ensure the simulated data is properly compared to the actual data.

Figure 33 below shows the overlaid plot for the head-on scenario.

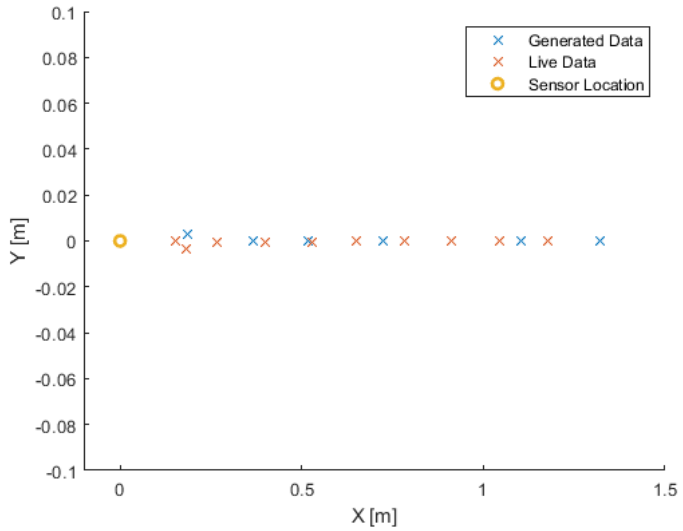

Fig. 33 Head-on Collision Scenario

The general trajectory does match very closely but the location that the points are at do not match as well. This will be discussed in the verification against model section.

Figure 34 below shows the plot of the near miss scenario sensor and simulated data.

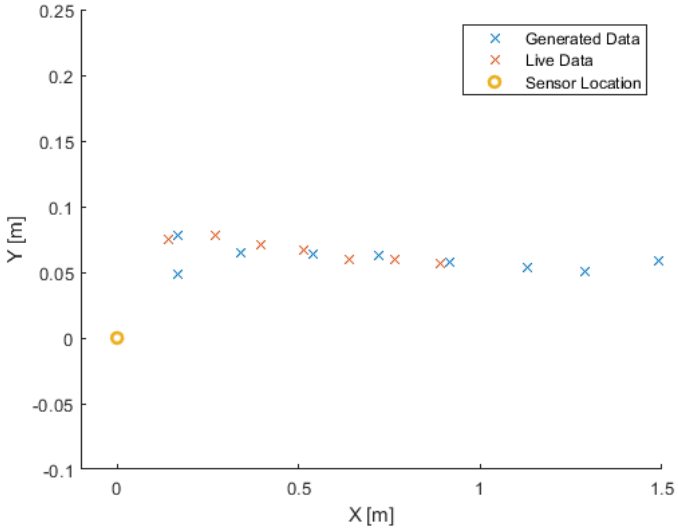

Fig. 34 Near Miss Collision Scenario

As seen in the figure, the simulated data doesn't match the live data as well as it did in the head on scenario but it still follows the general path of the actual data.

Lastly, the clear miss scenario plot overlay is shown in Fig. 35 below.

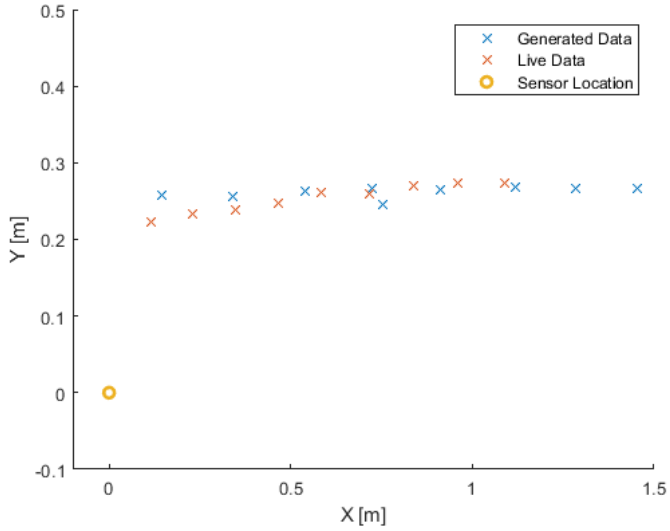

Fig. 35 Clear Miss Collision Scenario

As seen in the plot, the generated and live data again follow the same path but there is variation in the x position.

Verification Against Model:

To compare the modeled sensor data against the measured sensor data, tables for each of the collision scenarios outlining the average of the relative distances between two x data points, the average of the relative distances between two y data points, the average of the relative distances between two time values of the data points, and the standard deviations of all of these values were created.

For the head-on collision, looking at table 5 below the average of the relative distances between two x data points for both sets of data is very close to one another. The average y data for both sets also match the expected y value of zero

for the head on scenario. The standard deviations of the data show that there is slightly more variation in the live test data then in the simulated test data. This does not pose an issue because the difference in the data is small and the overall projection of the object is still very similar. The standard deviation of the time between samples is very close as well but the average time between samples is slightly off. Since there are multiple data points, although the timing between the simulated and actual test data varies, there is enough information from both cases to properly conduct the state estimation.

Table 5 Statistics for Head-on Scenario

Source of Data	Mean ΔX	SD ΔX	Mean Y	SD Y	Mean Δt	SD Δt
Generated Test Data	-0.189 m	0.023 m	0.00 m	0.0010 m	0.1000 s	0.195 s
Live Test Data	-0.111 m	0.052 m	0.00 m	0.0011 m	0.0856 s	0.237 s

Table 6 below shows the statistics for the near miss scenario. As expected from the plot, there is more variation in the simulated data then the live data. The average y data is a value close to zero (as expected for the near miss scenario) and the average delta x is very close in both the simulated and live test data. However, once again, the mean Δt values are off from one another.

Table 6 Statistics for Near Miss Scenario

Source of Data	Mean ΔX	SD ΔX	Mean Y	SD Y	Mean Δt	SD Δt
Generated Test Data	-0.165 m	0.069 m	0.060 m	0.0089 m	0.0887 s	0.215 s
Live Test Data	-0.124 m	0.002 m	0.067 m	0.0085 m	0.0667 s	0.144 s

Table 7 below shows the statistics for the clear miss scenario. As expected from the plot above, the variation in the x data is larger in the simulated data but is still small in magnitude.

Table 7 Statistics for Clear Miss Scenario

Source of Data	Mean ΔX	SD ΔX	Mean Y	SD Y	Mean Δt	SD Δt
Generated Test Data	-0.164 m	0.057 m	0.262 m	0.0076 m	0.0900 s	0.227 s
Live Test Data	-0.122 m	0.006 m	0.253 m	0.0185 m	0.0654 s	0.180 s

Overall, we see that the generated data tracks the y-position of the actual results closely but that the spacing on the x-points are generally higher for the generated data. This means that the velocity used for generating the test data was higher than the velocity at which the live tests were performed. This can be corrected by lowering the velocity of the generated data in order to better fit the actual test results or by increasing the initial height at which the ball is placed on the ramp. Additionally, the average Δt is close to the expected 0.1 seconds for the 10 Hz scanning rate of the generated data, since there is only one averaged measurement per 360° scan. However, the average Δt for the live data is lower than expected and would correspond to a scan rate of roughly 15 Hz. This frequency is within the physical bounds of the sensor, however, it was configured to run at the requested 10 Hz. This indicates that the sensor could actually be operating at the faster 15 Hz scan rate even though the configuration was verified to be set at a 10 Hz scan rate.

D. Full System Level Testing

Authors: Sam, Isaac

For the full system level testing, there were numerous tests conducted used to validate the designed system. These tests include the head-on collision test, the near-miss test, the clear-miss test, the control law scaling test, and the NEES/NIS test. For each of the described tests, we will discuss the setup, facilities and equipment used, the procedure, the measurements taken, any issues with the measurements, the results from these tests, and the verification of the test against the model.

1. Head-On Collision Test

The head-on collision testing was performed to verify avoidance of a collision was feasible, and that the collision covariance was reduced to an acceptable size for this maneuver.

Setup: The setup for this test includes setting the ramp to be directly on the centerline of the sensor and perpendicular to the MDF. This will allow for a proper collision scenario. Additionally, all of the electronics must be powered on, and the software must be sent to run this test. The homing process of the sensor will occur as the system powers on and is connected with the software to properly position the sensor for testing.

Facilities and Equipment: The testing occurred at the Aerospace Building on the testbed. To move the ramp laterally or change its heading, a hex key must be used. Additionally, a 2" diameter ball must be used to perform the test.

Procedure: First, the testbed must be prepared as described in the setup section, including properly setting the ramp positioning and conducting a homing sequence. Then, as the code runs, the sensor will begin spinning and shortly after will begin gathering data. This is when the ball can be released from the ramp. For the most accurate ball trajectory, releasing with one finger must occur. After the ball is released and a maneuver is performed, the software must be closed and the data examined. Five tests at each ramp speed must be conducted [0.5,1,1.5,2] m/s.

Measurements: The measurements gathered during testing includes the video footage of the gantry movement as well as the sensor data, state estimation data, and gantry position data. An image of example gantry footage can be seen in figure 36.

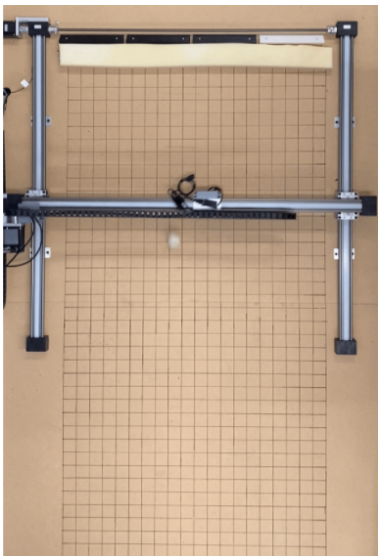

Fig. 36 Photo of Testing

Additionally, the full system test data from a head-on scenario at a specific timestep can be seen in figure 37. From this set of data including state estimated incoming object positon, gantry position and radius, and the collision covariance fixed at the start of maneuver, CAST is able to identify whether a collision has been avoided. A total of 10 full system tests in the head-on configuration were conducted to better characterize system performance.

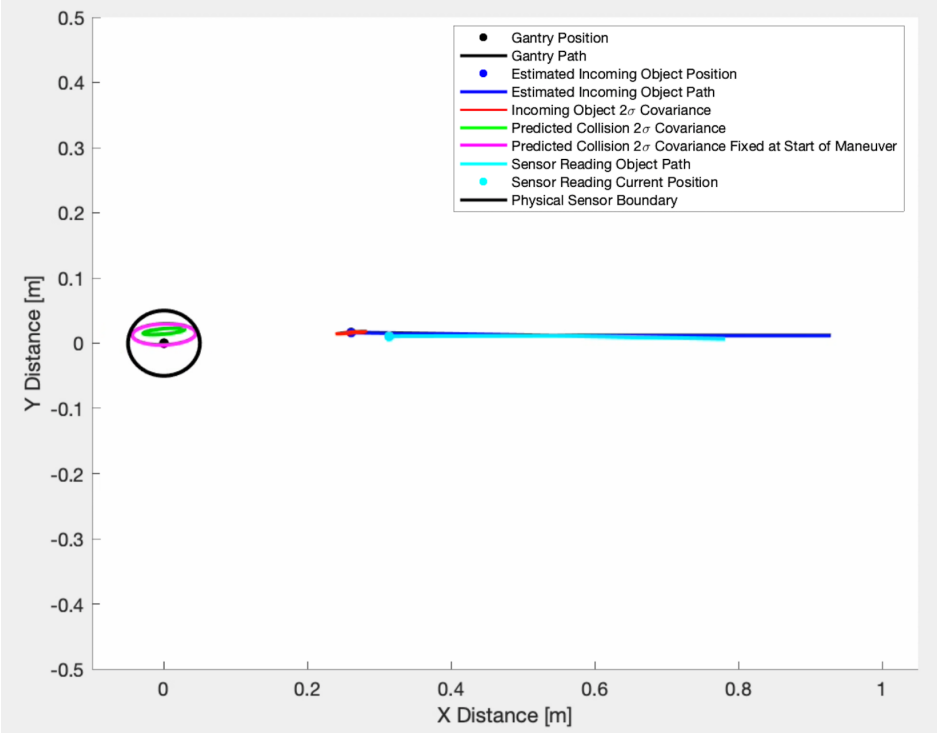

Fig. 37 Head-on Collision Full System Test Data

Issues with Measurements: To assist with testing, a white ball was used as the incoming object. This assisted the sensor with object detection as it did not absorb as much energy from the sensor. Based on the 10 head-on scenario tests performed, an issue was observed with the state estimation data. When the state estimation data was plotted in a live animation with the gantry movement, it became obvious that the state estimation data went astray during gantry movement. This led team CAST to believe that there was an issue with encoder data being passed back incorrectly to the state estimation algorithm. To further test this hypothesis a high and low acceleration maneuver test was run for the head-on scenario. In other words, when a maneuver is generated the specified acceleration profile was varied to observe the effect. Figure 38 shows the results of a high acceleration maneuver (approximately 3 m/s^2). It is clear that as the gantry maneuvers the state estimation data does not follow a linear path as one would expect. Figure 39 shows the results of a maneuver generated with a lower acceleration profile magnitude (approximately 1.5 m/s^2). Based on this data, it is clear that the state estimation data follows a more linear path, as one would expect even with gantry movement. This led to CAST to confirm the hypothesis that there is an issue with encoder data not being handled properly.

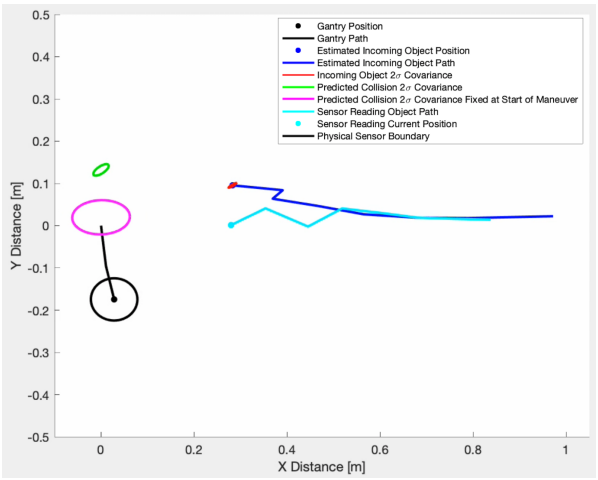

Fig. 38 High Acceleration Head-on Test

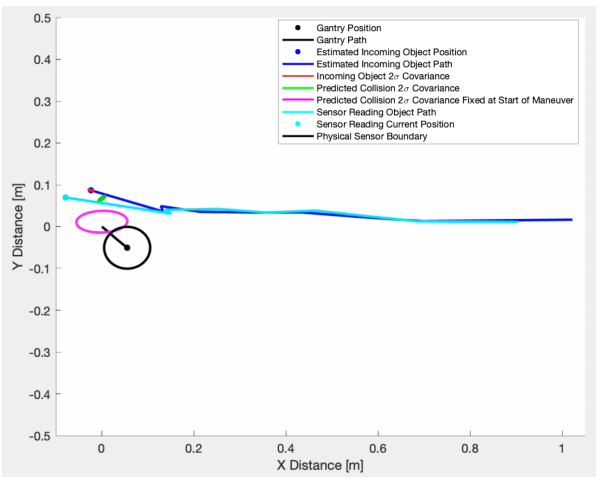

Fig. 39 Low Acceleration Head-on Test

Results: Overall, for each of the 10 tests conducted at 2 m/s, the gantry performed an accurate maneuver from the collision scenario. The gantry performed a maneuver and did not collide with the incoming object during these tests and the gantry also avoided the 2σ collision covariance, indicating full success even with the previously discussed encoder data challenges.

Verification against Model: The results of the head-on full system testing align with the predicted model. For a head-on scenario the collision covariance will overlap the sensor and the incoming object will physically collide with the gantry. Therefore, the model predicts a maneuver to occur, which does in fact happen for each of the 10 tests performed.

2. Near-Miss Test

This test is performed to test the system's ability to react on border-line maneuver or non-maneuver situations and tests the limits of the covariance. It is important to note that a near-miss test may or may not result in a maneuver. While a physical collision will not occur, it is possible that the collision covariance will overlap the sensor, resulting in a collision probability above a tunable threshold. If this threshold is met, a maneuver will be generated.

Setup: The setup for this test includes setting the ramp to be 10 cm off of the centerline of the sensor and perpendicular to the MDF. This can be off centerline in either direction. This will allow for a proper near-miss scenario. Additionally, all of the electronics must be powered on, and the software must be sent to run this test. The homing process of the sensor will occur as the system powers on and is connected with the software to properly position the sensor for testing.

Facilities and Equipment: The testing occurred at the aerospace building on the testbed. To move the ramp laterally or change its heading, a hex key must be used. Additionally, a 2" diameter ball must be used to perform the test.

Procedure: First, the testbed must be prepared as described in the setup section, including properly setting the ramp positioning and conducting a homing sequence. Then, as the code runs, the sensor will begin spinning and shortly after will begin gathering data. This is when the ball can be released from the ramp. For the most accurate ball trajectory, releasing with one finger must occur. After the ball is released and a maneuver may be performed, the software must be closed and the data examined. Five tests at each ramp speed must be conducted [0.5,1,1.5,2] m/s.

Measurements: The measurements gathered for the near-miss scenario show the collision covariance intersecting with the physical sensor position. A maneuver for this scenerio is performed if the 50% threshold is met by the incoming object. Depending on different trials of the test, the gantry would or would not perform a maneuver based on whether the threshold is met. Figure 40 shows the same data plotted as in the head-on scenario above. Based on this dataset, it is clear that a maneuver is not generated as the collision covariance does not overlap the physical sensor, which means the threshold is not met to generate a maneuver.

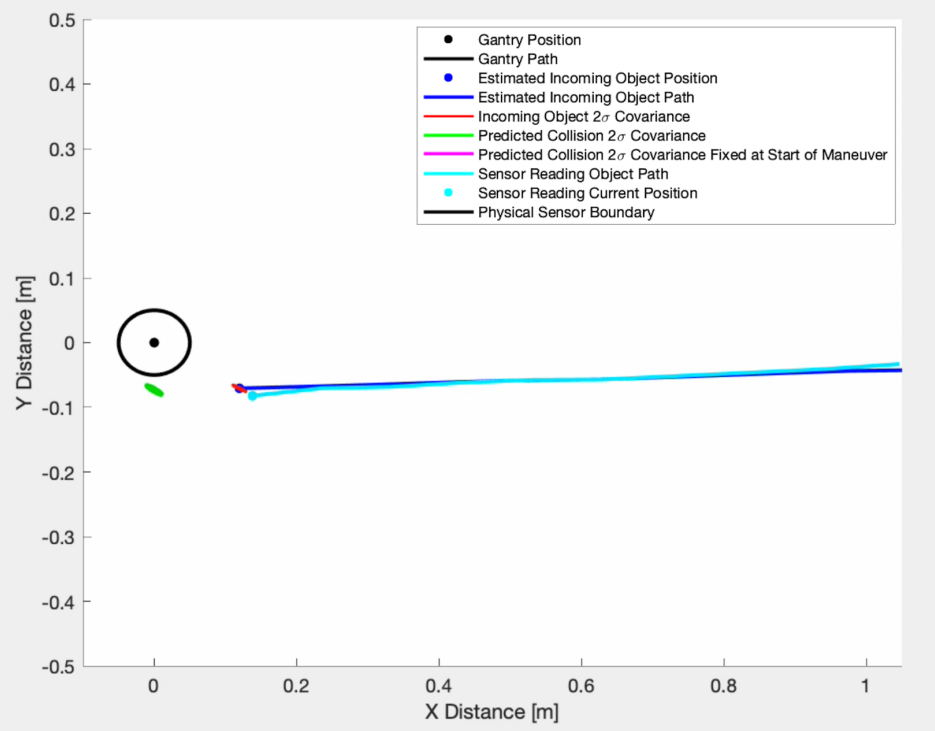

Fig. 40 Near Miss Collision Scenario Data

Results: For a total of 10 tests at 2 m/s relative velocity, the system performed a maneuver 7 times. This means that the 50% threshold of the integrated area of the collision covariance and the physical sensor was met in 70% of the cases.

Verification against Model: The results of the near-miss testing are directly in line with the model prediction. Since a collision is not always necessary for the near-miss cases, it is perfectly acceptable to see the 70% maneuver generate rate out of the 10 tests performed.

3. Clear-Miss Test

The clear-miss test is performed to verify that the system will not make a maneuver sequence in all scenarios. The system will be able to detect the incoming object, but will not perform a maneuver as the probability of collision is too low.

Setup: The setup for this test includes setting the ramp to be 40 cm off of the centerline of the sensor and perpendicular to the MDF. This can be off centerline in either direction. This will allow for a proper clear-miss scenario. Additionally, all of the electronics must be powered on, and the software must be sent to run this test. The homing process of the sensor will occur as the system powers on and is connected with the software to properly position the sensor for testing.

Facilities and Equipment: The testing occurred at the aerospace building on the testbed. To move the ramp laterally or change its heading, a hex key must be used. Additionally, a 2" diameter ball must be used to perform the test.

Procedure: First, the testbed must be prepared as described in the setup section, including properly setting the ramp positioning and conducting a homing sequence. Then, as the code runs, the sensor will begin spinning and shortly after will begin gathering data. This is when the ball can be released from the ramp. For the most accurate ball trajectory, releasing with one finger must occur. After the ball is released and a maneuver is performed, the software must be closed and the data examined. Five tests at each ramp speed must be conducted [0.5,1,1.5,2] m/s.

Measurements: The sensor collects data with the clear-miss case, however, the collision covariance never intersects with the incoming object, so maneuvers are not performed. This can be seen in figure 41 where again the collision covariance is outside of the physical sensor boundary, indicating that a maneuver is not required.

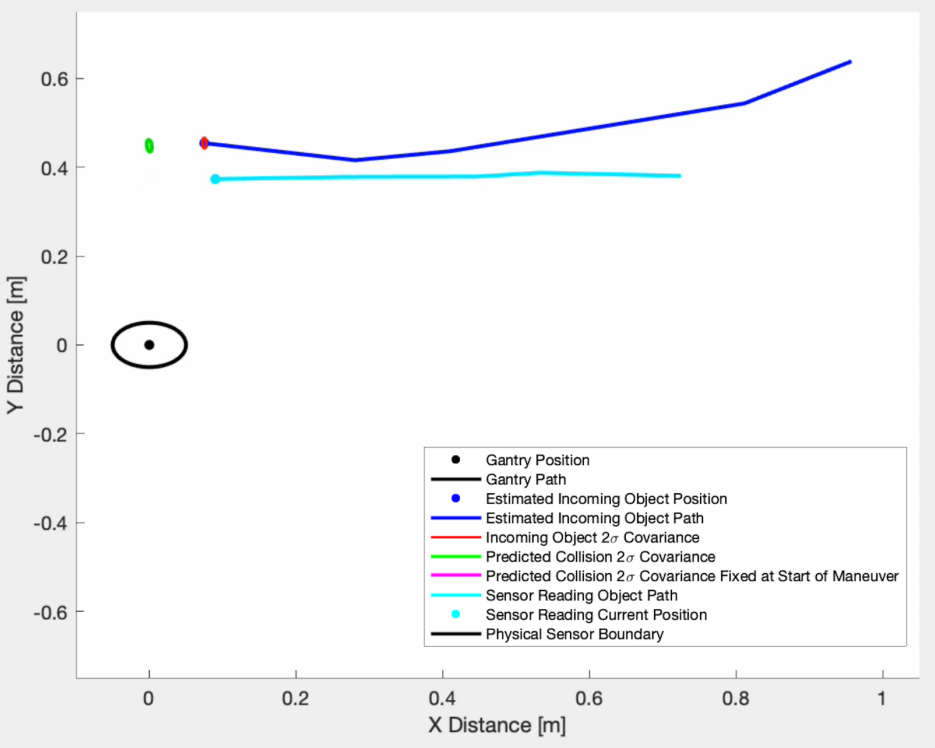

Fig. 41 Clear Miss Collision Scenario Data

Results: For each test conducted, the gantry did not performed a maneuver. The incoming object never crossed the 50% threshold of the covariance as the probability of collision was too low. There were 2 trials of this test out of 10 where a maneuver was performed by the gantry as the incoming object would bounce off of the backstop of the testbed and re-approach the sensor. Due to the 360° spinning LiDAR sensor on the testbed, the ball would then cross the covariance uncertainty threshold and a maneuver was made. This is further evidence the maneuver is not performed until the incoming object is much closer to the sensor.

Verification against Model: The results of the clear-miss testing are directly in line with the model prediction. Since the collision covariance never intersects the physical sensor, a maneuver is generated 0% of the time (this is ignoring the maneuvers generated after the ball has reached the backstop).

4. Control Law Scaling Test

The control law scaling testing is performed to verify the guidance, navigation, and control law in a scaled 2-body problem.

Setup: The setup for this test involves interfacing the CAST algorithm with MATLAB's sensor fusion toolbox. First a pair of conjunctive orbits is generated by integrating two-body dynamics backwards from the collision point. Then a set of measurements is generated from a simulated mono-static LiDAR implemented in the sensor fusion toolbox. The CAST algorithm is then run on the simulated measurements. The test can be performed by running the run_sim.m script on the scaling branch in the CAST git repository.

Facilities and Equipment: This test can be run on any computer with MATLAB and the CAST git repository downloaded.

Procedure: Run the run_sim.m file on the scaling branch in the CAST git repository.

Measurements Taken: Each test returns the initial chief orbit, the collision object orbit, the post-maneuver chief orbit, the maneuver direction and duration, all LiDAR measurements, all state estimations, and all collision predictions

generated over the course of the simulation.

Issues with Measurements: Measurements are based on a two-body dynamics simulations and simulated LiDAR measurements. Any data generated from these simulations are subject to modeling errors, floating point precision errors, and integration errors inherent in adaptive Runge-Kutta 4-5 initial value problem solutions.

Results: Presented in figure 42 is the final approach of one of the simulated scenarios. In some cases the maneuver places the chief in the path of the incoming object. This is due to a limitation of the algorithm as implemented in that it does not consider the velocity of the object when planning the maneuver.

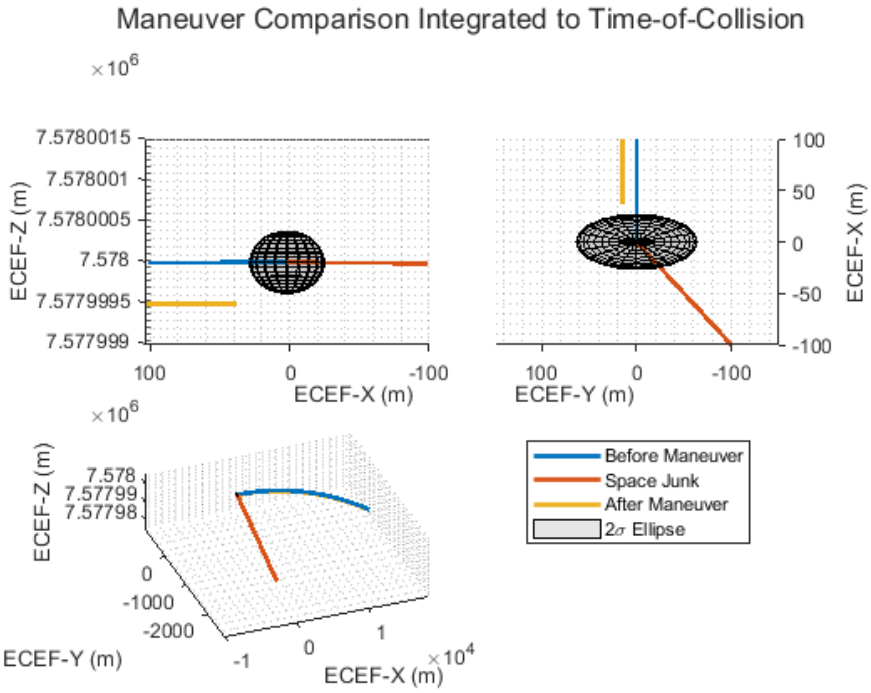

Fig. 42 Simulated Collision Avoidance

Verification against Model: Successful simulated collision avoidance matches the behavior of the algorithm on the test bed.

5. NEES/NIS Test

NEES and NIS testing is performed to verify the performance of Kalman Filters. It normalizes the filter error and innovations with their covariances in order to obtain an unbiased distribution of the errors. These errors are then compared to the χ^2 distribution to ensure that they are unbiased.

Setup: To perform NEES/NIS testing, a simplified main loop script was developed that ran only the state estimation functions. 50 collision simulations was created using the sensor model.

Facilities and Equipment: The NEES/NIS tests are software tests that can be performed on any computer with MATLAB and access to the Github repository. These tests were performed using the personal laptops of CAST team members in their homes.

Procedure The simulated sensor readings were then run through EKF testing loop. At each time the filter performed a state estimation, the normalized estimation errors and normalized innovations, which are defined in (14)-(18) were plotted. (14)-(18).

$$e_{x,k} = x_k - \hat{x}_k^+ \tag{14}$$

$$e_{y,k} = y_k - \hat{y}_k^- \tag{15}$$

$$\epsilon_{x,k} = e_{x,k}(P_k^+)^{-1}e_{x,k} \tag{16}$$

$$S_k = \tilde{H}_k P_k^- \tilde{H}_k^T + R \tag{17}$$

$$\epsilon_{y,k} = e_{y,k}(S_k)^{-1}e_{y,k} \tag{18}$$

After plotting the errors, they were compared to the bounds of a χ^2 distribution, which was calculated using the MATLAB function `chi2inv.m`.

Data Collected The data collected for this test was the normalized errors $\epsilon_{x,k}$ and $\epsilon_{y,k}$.

Results When the normalized errors are plotted in relation to the expected χ^2 distribution, as shown in figure 43, the results are lower than expected. This indicates that the estimate covariance is too large. This is primarily due to the timestep used by the filter, which is reliant on the sensor scan rate, being too large. When the prediction step of the filter occurs, the calculation of the prediction covariance is primarily dependent on the timestep used by the filter. When the timestep is too large the prediction step introduces uncertainty that cannot be accounted for in the correction step. To improve upon these results, the use of a shorter timestep, and consequentially, a faster scan rate, is recommended.

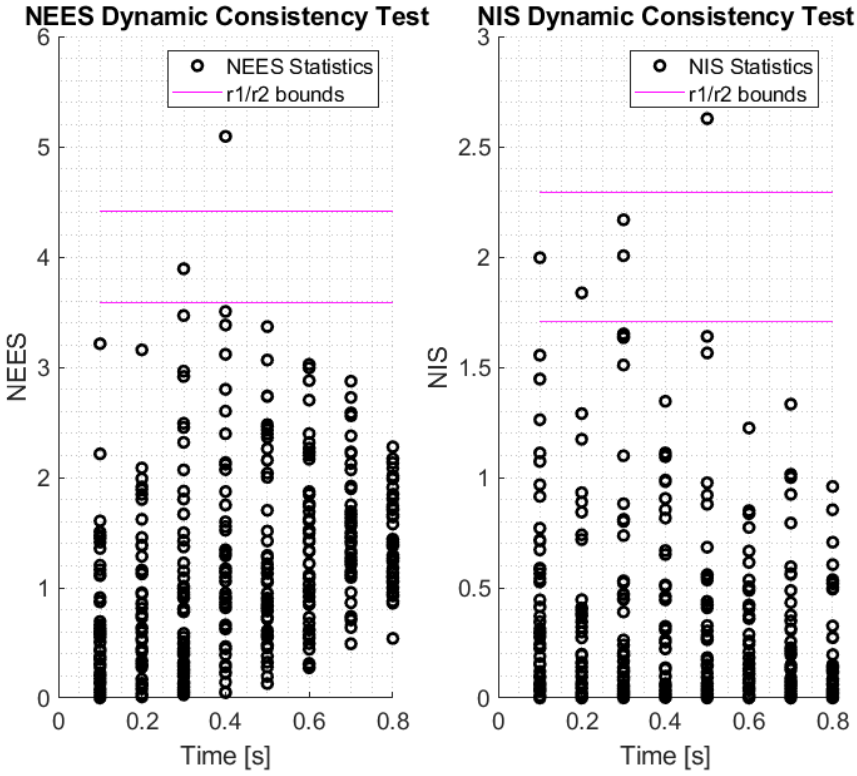

Fig. 43 NEES/NIS Plots

In addition to performance issues introduced by the timestep a number of other issues plagued the process of NEES/NIS testing during the year. Obtaining an accurate ground truth model – with appropriate process noise and sensor measurements – required debugging of the truth model generation as well as the filter. This debugging process was found to be extremely time consuming. The bias introduced into the system by recording multiple measurements was also detrimental to the performance of the filter. These improvements are discussed further in VIII.A.3 and VIII.A.1.

E. Validation and Evaluation of Functional Requirements and Success Criteria

The test plan for team CAST was designed to be able to specifically validate the project functional and design requirements as well as evaluate the specific project success criteria outlined in the levels of success (Table 1).

1. Validation of Functional Requirements

FR1: The test system shall consist of a physical testbed capable of creating relative motion between two objects.

Validation: Validation of this requirement requires the ability to launch the ball along its designed various trajectories and also requires the ability to move the gantry. These criteria were very simply validated through the ramp/ball tests as well as the gantry velocity and acceleration tests. The independent verification of both of these verifies that the system is capable of creating relative motion between two objects and *functional requirement 1 was satisfied*.

FR2: The test system shall be capable of detecting a live, incoming object.

Validation: This functional requirement was validated by tests relating to the sensor. First, the LiDAR sensor test verified that the system was able to detect a moving object within the domain of the testbed. The vibration test was used as further verification that detection would not be impeded by the movement of the gantry. These tests verified that *functional requirement 2 was satisfied*.

FR3: The test system shall be capable of determining if a collision will occur.

Validation: Validation of the third functional requirement called for the validation of the state estimator. The full system testing of the 3 different collision scenarios was successful, showing that the system is capable of determining if a collision will occur. However, the first test designed to verify the state estimator (the sensor/software integration test) was not passed as discussed earlier. Similarly, NEES and NIS testing was performed in order to further validate the state estimator. Again, these tests were not passed. This shows that the system is capable of determining if a collision will occur, but not to the accuracy, precision, and consistency that was required. This is largely due to the conditional dependency of the state estimation process on a good initial guess. State estimation only works well on the testbed scale when seeded with a good initial guess. For on-orbit application, this could mean seeding the initial guess with typical ground station tracking data or using a combination of sensors and estimators. On the testbed, transitioning to a real-time operating system would allow for proper sensor and encoder integration requiring less processing overhead. This would allow for a more consistent estimator overall. Overall, *functional requirement 3 was not satisfied*.

FR4: The test system shall be capable of avoiding a physical collision using motion characteristic of a thruster response in orbit.

Validation: The final functional requirement was verified mainly through full system testing and the thrust curve matching test. Initially, the latency test and velocity and acceleration tests verified that the system would be capable of collision avoidance. When the system was fully integrated, the scenarios of the full system test verified that the system is "capable of avoiding a physical collision". The thrust curve matching test verified that this was done "using motion characteristic of a thruster response in orbit." These tests verified that *functional requirement 4 was satisfied.*

2. Evaluation of Success Criteria
See all criteria in Table 1.

Test Environment:

Evaluation: All three levels of success under this project element were evaluated with the ramp and ball rolling test. This test found that the testbed is capable of supporting 2D object trajectories with variation in heading, angle, and velocity up to 2.3 m/s. Satisfaction of this top level ensures satisfaction of the lower levels under this project element. *Test Environment levels 1-3 were satisfied* and the Test Environment project element is therefore a success.

Detection:

Evaluation: Similarly to the previous project element, satisfaction of the highest level of success under the detection element also ensures satisfaction of the lower level requirements. The level 4 requirement was evaluated through the LiDAR sensor test. This test showed that the sensor could detect a moving object of the specified size. The speed and heading component of this requirement was validated through the

sensor/software test and full system tests. Therefore, *Detection levels 1-4 were satisfied* and the Detection project element is therefore a success.

State Estimation:

Evaluation: The first level of success under state estimation was validated through the sensor/software test. This test verified that the system is simply able to return estimation of state at current time and predict forward to point of collision. As this level does not specify any accuracy or consistency measures, this level was met. This test also verified that the 2σ prediction covariance driven to within an avoidable region (2nd level of success). The full system testing confirms this. The levels 3 and 4 require certain satisfaction of the NEES and NIS tests that were not achieved. Therefore, *State Estimation levels 1-2 were satisfied* and the State Estimation project element can only be considered a partial success.

Avoidance:

Evaluation: The first level of success under this project element was initially proven feasible through the sensor/software test and gantry velocity and acceleration tests. Full system testing of the head on and near miss scenarios showed that the system is in fact capable of avoiding a collision, meeting level one. Levels 2-4 depend on the capability and accuracy of the thrust curve matching. Level 4 was met through the full system testing and the thrust curve matching test, so *Avoidance levels 1-4 were satisfied* and the Avoidance project element is therefore a success.

Testbed Simulation:

Evaluation: The two levels under this project element were simply evaluated with the full system testing. The full system tests demonstrated the control law for multiple 1D and 2D collision profiles. Therefore, *Testbed Simulation levels 1-2 were satisfied* and the Testbed Simulation project element is therefore a success.

Application Simulation:

Evaluation: The final project element in the levels of success was evaluated through the control law scaling test. This test was performed purely under simulation. This simulation scaled up the control law to a single full scale orbital cross-track scenario. These results were then used iteratively in order to improve on the performance of the testbed control law. It was found through simulation that the first iteration of the maneuver implemented on the testbed had incorrect coordinate frame transformations. After implementing this correction, avoidance performance on the testbed greatly improved. This test verifies that the *Application Simulation levels 3-4 were satisfied* and the Application Simulation project element is therefore a success.

VI. Risk Assessment and Mitigation

A. Risk Identification

Authors: Cameron

CAST's approach to risk assessment was to identify those risks that we felt were relevant to the success of the project, estimate the severity and probability of those risks, use models/clarifications/different solutions to mitigate that risk, and then reevaluate where the state of that risk was after these implementations. Following this identification, which was executed primarily through brainstorming. We executed procedures while utilizing our identified mitigations, and addressed unaccounted for risks as they occurred. Table 8 provides a listing of the risks that were considered in the CAST design, including the letter that was used to mark said risks on risk assessment matrices. These risks have been organized according to whether they contribute to technical, logistical, financial, or safety risk.

Table 9 provides a legend for the color scheme used in the risk assessment tables to follow. Risks have been identified as key risks if their severity/probability combination could result in serious issues for the project, and if it was determined that those issues could be lessened. These key risks have been bolded and underlined in the risk assessment tables.

Table 10 shows the team's assessment of risk prior to mitigation, and Table 11 displays the same after mitigation.

Table 8 Risks Addressed by the CAST Team.

Risk	Type	Letter Code
Structural Failure	Technical	A
COVID Shutdown	Logistical	B
Shipping Delays	Logistical	C
Shorting Electronics	Technical	D
Shorting Motors	Technical	E
Weather Interference	Logistical	F
Medical or Personal Emergency	Logistical	G
Communication Losses	Technical	H
Insufficient Data Rate	Technical	I
Gantry System Malfunction	Logistical	J
Failure to Interface	Technical	K
Excessive Noise Invalidates Sensor Readings	Technical	L
Budget Violation	Financial	M
Limit Switch Failure	Technical	N
Improper Gantry Operation	Technical	O
Computation Time	Technical	P
Collision Object Damages System	Safety	Q
Bodily Injury	Safety	R
Overheating Electronics	Technical	S

Table 9 Color Scheme for Risk Assessment Matrices.

High Risk	Moderate Risk	Low Risk	Negligible Risk

Table 10 CAST Risk Assessment Matrix (Pre-Mitigation).

Severity	Probability				
	Frequent	Likely	Occasional	Seldom	Unlikely
Catastrophic		**P**	B	**C**, **J**, **K**	
Critical			**I**	**M**	A, G
Moderate			F	D	E, L, N, O
Negligible	Q		S	H, R	

Table 11 CAST Risk Assessment Matrix (Post-Mitigation)

Severity	Probability				
	Frequent	Likely	Occasional	Seldom	Unlikely
Catastrophic			B		
Critical					A, **C**, **I**, **J**, **K**, **M**
Moderate			**P**		G
Negligible	Q		F		D, E, H, L, N, O, R S

B. Risk Tracking and Mitigation

Authors: Cameron

1. Technical Risks

What follows is an explanation of the four attributes that CAST used to examine each key technical risk, concluding with a summary of the remaining, minor technical risks.

Key Risks

Insufficient Data Rate (I)

Description of Risk: The data rate is not fast enough to produce a desired response within the required timeframe, resulting in the CAST system being unable to properly demonstrate implementation of collision avoidance software.

Explanation of Severity/Probability: Given the numerous transfer points present between components of the CAST design, it is not unlikely that the data rate for the system may be too slow to allow for proper system implementation. A sufficient data rate is critical to the success of the project, since the CAST team would need to significantly reduce the scope of the project if it could not perform in accordance with timing requirements; this would in turn decrease the usefulness of the final product provided to the customer.

Mitigation: The CAST team has performed a significant amount of research into transfer rates; the results of this research and analysis are presented in subsubsection III.E.4. Through its efforts, the team has been able to quantify the timing delays expected in the CAST design, and by verifying that the expected total time delay is smaller than what has been established as the maximum allowable time delay the team has established that the CAST system can be expected to perform according to timing requirements.

New Severity/Probability: The results of the team's investigations into timing delays and latency show that the CAST system is unlikely to have a data rate insufficient for its proper performance. Thus, this risk is effectively mitigated despite its potential to have a critical adverse effect on the success of the project.

Failure to Interface (K)

Description of Risk: Interface does not work as intended and/or communication between different components requires technological expertise outside of the scope of the team's current level skill.

Explanation of Severity/Probability: If the CAST team cannot effectively coordinate communication between the various subsystems to produce a final product that can interface properly, it can not complete the project to any extent. Considering the limited interfacing experience of many CAST group members, there is a possibility that an important aspect of interfacing could go unaddressed which could cause total interfacing failure.

Mitigation: As with the "Shorting Electronics" and "Shorting Motors" risks detailed below, the team, having already developed wiring and power connection diagrams for the CAST design (Figure 7), as well as having performed significant research into communication interfaces for the purpose of quantifying time delays inherent to data transfer, can be confident that all necessary communication interfaces are properly accounted for and can be executed without issue during the CAST system assembly. Additionally, the CAST team intentionally steered away from complicated communication protocols as it was engineering its baseline design in favor of technology which the team can confidently work with and which has significant documentation available for it.

New Severity/Probability: With the current level of preparation, the CAST team is unlikely to come across any issues with communication or power interfaces. If a problem does arise in this area, the team's use of simple and well-documented communication protocols should remove any obstacles to fixing it, and thus such a problem should not lead to catastrophic consequences.

Computation Time (P)

Description of Risk: The algorithm takes too long to run, and the gantry can not respond to the collision object in time to avoid it.

Explanation of Severity/Probability: This is one of the primary challenges of this engineering solution, because trajectory prediction typically takes a significant amount of computational energy and time. Factoring in the additional risk of communication delays, it is fairly likely that CAST's algorithm could be too slow to avoid a fast travelling

collision object in real time. One of the key objectives of this project is to produce an algorithm that can avoid this issue, so it is catastrophic to success if this risk of slow computational time occurs.

Mitigation: To mitigate this risk, the CAST team has researched and implemented software development strategies designed to reduce computation times,[†] and has also run preliminary computational time estimates with partial code solutions. The timing delays of the entire system have also been investigated in subsubsection III.E.4 and summarized in Fig. 9 and Table 3, and it has been confirmed through this research and analysis that the expected software delays will not be enough to slow the system down beyond acceptable limits.

New Severity/Probability: The team's endeavors to streamline and optimize the CAST system's software, as well as the work it has put into quantifying software timing delays, lessens the likelihood of a computational time failure slightly, and ensure that CAST has supplemental optimizations and additions in mind which can ensure that a time failure will only moderately impact project success if it is unavoidable.

Minor Risks

Structural Failure (A): A structural failure of either the base structure or the gantry could result in breakage of system components, which could be particularly problematic if said components are expensive or difficult to repair/replace. This is fairly unlikely to occur given the sturdiness of the gantry and that of the MDF board and its metal structural supports. However, to better quantify this risk and mitigate it through proper research and analysis the team has performed a structural analysis on the base structure which showed that there will be a very high factor of safety across the entire the base structure. Additionally, the team has confirmed that the gantry's structural strength will meet the needs of the CAST design: the gantry's mass rating of 8.2 kg (taken from [16]) is well above the sensor package's mass of 0.190 kg (taken from [10]). With these analyses in place, the team can be confident that structural failure in the system will almost certainly not occur.

Shorting Electronics (D): Surging or improperly connecting electronic components could potentially destroy or damage them. This is fairly unlikely to happen given the fairly straightforward nature of the connections present in the system. However, damage to electronics would not have catastrophic consequences, as the CAST team has built margin into the project budget (as shown in Fig. 48) which can be used to replace any damaged parts. Additionally, the CAST team has spent a significant amount of time examining the specifications sheets of both the electronics that will be used in the CAST design and the components that they will be connected to, culminating in the wiring diagram presented in Fig. 7 and the PCB design presented in Fig. 17, which will be invaluable in guiding the team to make proper connections. Also, the team members will be consulting with subject matter experts in the area of electronics and communications prior to any verification and validation tests of the system.

Shorting Motors (E): Surging or incorrectly connecting motors could destroy or damage them beyond repair. Even before mitigation, this is fairly unlikely to occur, as the motor connections are simple and will be made using a cable package from Igus designed to be user friendly and easily manageable. Additionally, a damaged motor would not cause significant consequences to the project in terms of scheduling delays or budget shortages, as the nema 23 stepper motors used in the system design can be readily and cheaply replaced. Additional mitigation strategies include those which have been covered in the discussion on "shorting electronics" above: referencing well-researched wiring and power diagrams and consulting with subject matter experts before making any component connections.

Communication Losses (H): Loss of transferred packets or transfer of unexpectedly large amounts of data can adversely affect software runtime and consequently prevent the successful performance of the system. Such communication losses could occur from improper wiring of cables, but this risk has been effectively mitigated by the CAST team's research into system wiring (culminating in the wiring diagram presented in Fig. 7). Such communication losses could also occur due to cable defects, but to mitigate this risk the team has purchased a high-quality cable and wiring solution from Igus to ensure crisp and robust communications between the Igus gantry and the Arduino (which is the most important line of communication of the system) and will be checking the quality of other third-party cables before implementing them in the CAST design.

Excessive Noise Invalidates Sensor Readings (L): Since the CAST software depends on accurate sensor readings, if excessive noise within or around the testbed compromises the accuracy of sensor readings then the effectiveness of the system can be compromised. It is not likely for this to occur, as very little noise is expected to affect the system. To account for this in a robust and quantifiable way, however, the team has performed a vibration analysis on the system and has analyzed how vibrations may affect sensor readings, determining that the error introduced in sensor readings by

[†]One such software development strategy has been to pre-calculate collision avoidance maneuver profiles and save them in a data file that can later be accessed during a live test, thus eliminating the need for a live calculation of a maneuver response for collision avoidance. The team has also considered the effects of parallel processing, general mathematical optimizations, and slowing down the collision object.

system vibration is negligible. Additionally, the team implemented sensor internal noise values taken from [10] in the development of its collision avoidance software. Thus, this risk is fully accounted for and mitigated.

Limit Switch Failure (N): The gantry limit switches help to prevent improper gantry or motor operation; if they malfunction, the motors could be subject to minor harm or (in rare cases) total breakage. The most effective mitigation for this possibility is that the CAST team is already fully prepared to connect the switches properly, having created full wiring (Fig. 7) as well as a PCB design (Fig. 17). Additionally, in its Nov 30 purchasing meeting with Igus the team was informed the fact that the limit switch is built into the cable connection provided by Igus, operating automatically in conjunction with the motor gantry motors. With this information in mind along with the team's preparations, it is nearly impossible for a malfunction of the limit switches to occur unless gross negligence or recklessness occurs on the part of any team members.

Improper Operation of Gantry (O): Damage to the gantry can occur if the gantry system is connected improperly or its limitations are exceeded in a test. This is unlikely to happen, as the system is specially designed to be user-friendly. Additionally, in its 30 Nov meeting with Igus the CAST team was informed that incorrect connections or improper operation will simply cause the gantry to terminate operation rather than creating a situation where the gantry could damage itself or the attached load, thus further mitigating this risk.

Overheating Electronics (S): The overheating of electronic systems due to external temperatures or overuse may occur occasionally during system testing. However, this occurrence is not likely to lead to severe consequences, as the solution will generally be to simply give electronic components time to cool down before continuing their use. In any case, this risk is effectively mitigated by the design of the electronics box that will be used in the CAST design, which uses mesh walls and fans to guard against excessive temperatures. Additionally, the team has performed a preliminary thermal analysis of the electronics box, which adds further confidence that no overheating will occur.

2. Logistical Risks

What follows is an explanation of the four attributes that CAST used to examine each key logistical risk, concluding with a summary of the remaining, minor logistical risks.

Key Risks

Shipping Delays (C)

Description of Risk: Shipment of the gantry from Igus is delayed or critically hindered due to COVID-19 conditions or other unforeseen circumstances. This issue is exacerbated by the fact that the Igus gantry has a significant lead time, and can become further problematic if the shipment must come from Cologne, Germany (the location of Igus' main headquarters).

Explanation of Severity/Probability: Seeing as how the gantry system is a primary component of the CAST design, if the CAST team is unable to obtain the gantry system within a reasonable timeframe then the project will be catastrophically delayed; much of the work that can be done without the physical system has already been completed, and further progress will be halted until the testbed is built, verified, and validated. Additionally, unforeseen workload surrounding assembly of the system to a fully operational point will be undetermined until the gantry arrives.

Mitigation: The purchase process for the gantry was initiated atypically early—in the week of Nov 30, 2020—and research has been conducted into Igus' shipping practices, including a purchasing meeting between Igus representatives and the CAST team on Nov 30. Igus will be shipping the gantry from their main hub of United States operations in Rhode Island (which mitigates some of the risks inherent to international shipping in a time of quarantine). With an expected lead time of 4-6 weeks after receipt of the order, the gantry should arrive in mid- to late-January via USPS (a very reliable shipping source).

New Severity/Probability: Following the purchasing meeting and clarification of shipping terms, the severity of a shipping delay was determined to be less than catastrophic. Early order has also lessened the probability of a delay that effects success in any critical way.

Gantry Malfunction (J)

Description of Risk: The gantry system is delivered improperly calibrated or damaged to the point where it does not work correctly and the solution to this issue cannot be found.

Explanation of Severity/Probability: Taking into account general shipping and vendor practices, it is not entirely unlikely that the gantry system is delivered in a non-functional state, or damaged during delivery to a point where it is no longer operational. This would be a catastrophic occurrence, as the gantry constitutes the central component of the CAST design and could not be replaced (any backups or alternatives that have been considered will violate the remaining budget following the purchase of this gantry).

Mitigation: During the final purchasing meeting on Nov 30 2020 between Igus representatives and the CAST team, the team was assured that Igus performs a thorough full systems test prior to shipping any system and that malfunctioning or damaged deliveries are a rare occurence. Additionally, Igus ships via USPS, which has a no damage guarantee, and Igus also provides warranty services in the case that the gantry must be repaired or replaced.

New Severity/Probability: As a result of research into Igus' shipping practices, it is unlikely that the gantry system will arrive in a malfunctioning state, and if it does then Igus' full warranty service offers an option to reship a duplicate system. Therefore, a catastrophic system malfunction will actually only be critical, and is extremely unlikely.

Minor Risks

COVID Shutdown (B): A rise or spike in COVID-19 cases can lead to a partial or complete shutdown of project operations by preventing in-person meetings or access to necessary facilities. Such a shutdown could interrupt project progress and, if severe, could lead to the premature termination of the entire endeavor. While this risk is significant and not "minor," it is fully out of the team's control; besides practicing social distancing and other COVID-19 mitigation practices as well as including time margin in the project schedule (as detailed in the next minor risk description), the team has no way to mitigate the risk of a COVID shutdown.

Weather Interference (F): Adverse weather conditions (fairly common in Boulder) could prevent access to facilities for meetings, assembly, or testing. Weather conditions are out of the team's control, but by including significant time margins in the project schedule—as illustrated in Fig. 47—the team can confidently say that any weather-induced delays will not be catastrophic to the project's progress.

Medical or Personal Emergency (G): If a team member becomes ill or suffers from some other personal emergency, they could be prohibited from fully participating in the CAST design process. This is unlikely to occur, but would be problematic if it did happen for the fact that each team member possesses critical knowledge and skills necessary for the completion of the project. The primary mitigation for this risk is redundancy in the CAST team organizational structure and in the distribution of responsibilities: with contributors helping each of the project subsystems (as illustrated in Fig. 44), individuals will be able to step up and take charge of any part of the project where that additional support is needed.

3. Financial Risk

What follows is an explanation of the four attributes that CAST used to examine the singular, key financial risk for the project.

Key Risks

Budget Violation (M)

Description of Risk: The project's $5000 budget is violated due to replacement of purchased items or other unforeseen circumstances.

Explanation of Severity/Probability: CAST has had the budget in mind throughout the process of engineering the CAST design, and thus it is seldom that the team would be put in a situation where the budget would be violated. However, an overshooting of the budget could prevent the team from purchasing necessary parts or software, which would lead to a very problematic inability to complete the CAST design.

Mitigation: To mitigate budget concerns, the CAST team has done extensive research into the most cost-effective solutions for the collision avoidance problem, asked for and obtained discounts, and clarified warranty services that can be utilized in the event of equipment breakage. The CAST team has also given itself close to a $1000 margin in the budget should any unforeseen expenses arise, as is made clear in Fig. 48.

New Severity/Probability: With the team's effective budget planning and research taken into account, it is unlikely that the CAST team will violate the budget.

4. Safety Risks

What follows is a brief discussion of the safety risks that the CAST team investigated; these were all classified as minor risks.

Minor Risks

Collision Object Damages System (Q): It is possible for the collision object to collide with either the gantry or another component with enough force to cause damage. However, this risk is unlikely to be problematic given the stability of the system and the slow speed of the collision object. In any case, this risk is effectively mitigated by the team's choice of a light and soft collision object, and can be further mitigated by implementing foam padding or other protective features to protect system components.

Bodily Injury (R): Bodily injury can be inflicted on a team member or observer due to motion of the gantry or collision object. It's not probable for such an injury to be severe, and it is also unlikely for such an injury to occur if anyone operating or observing system tests simply follows common sense and due regard for safety. That said, this risk will be further mitigated by safety protocols which will be written up by the safety lead and followed by anyone operating the CAST system.

C. Risk Impact

Authors: Cameron

Through the vigorous risk assessment process that we performed, CAST was able to complete the semester without any unexpected risks impacting the project. However, some of the risks that we identified caused significant impediment to system completion and testing at different points over the Spring Semester. Moreover, a few of the mitigations that we implemented were not sufficient to fully reduce the risk that we experienced, so we had to execute additional mitigation procedures to stay on track with the project deliverables timeline. Table 12 identifies those risks that became most relevant during the manufacturing, testing, and finalizing stages of this project.

Table 12 Risks with Significant Impact on CAST Team Operations.

Risk	Type	Letter Code
Shipping Delays	Logistical	C
Medical or Personal Emergency	Logistical	G
Insufficient Data Rate	Technical	I
Failure to Interface	Technical	K
Budget Violation	Financial	M
Computation Time	Technical	P

Shipping Delays (C): Shipping delays constituted some of the most significant departures from the expectations that we set for risk at the end of the Fall Semester. Despite accounting for lead times and potential late arrivals, CAST ran into an issue where several products were either damaged on arrival, or needed to be sent back and warrantied almost immediately after being received. In particular, the LiDAR sensor had to be ordered 4 times, all of which impacted the manufacturing and testing timelines significantly. Still, with the massive margin that we planned for, we were able to perform all desired operations once we finally received a working sensor for the system.

Medical or Personal Emergency (G): We had one instance where a member of our team contracted COVID. Due to the fact that they were immunocompromised, they required significant recovery time to fight the illness and reach a fully healthy state again. Luckily, this member experienced a swift recovery, and the other team members in their subsystem were able to pick up some of their responsibilities while they were unable to work on the project.

Insufficient Data Rate (I): One of the partial inadequacies of the system that Team CAST produced for this project's satisfaction is that it far exceeds the initial latency estimates anticipated from the modeling phase. Specifically, both the MatLab maneuver generation and the reception/storage of the Arduino command have associated latencies that are an order of magnitude larger than we expected. Fortunately, the specifications of the system were more than capable of accounting for this additional time delay and still avoid the object. When the team discovered these latency results, we did implement some optimizations, but in order to improve the system response time any more there would need to be pronounced changes in design (development of an independent operating system and/or use of much better

processing equipment). Therefore, if this latency effect had turned out to be more of an issue, it is likely that we would have been unable to address it within the delineated time constraints.

Failure to Interface (K): There was a brief two week period where the team failed to establish a successful interface with the closed loop stepper drivers of the system. The result was that the Igus Gantry was unoperational for this segment of time. To address the issue, the team ordered open loop stepper drivers that were considered to be easier to interface with. As the open loop stepper drivers were still shipping, we were able to troubleshoot the closed loop stepper driver communication issue, so these additional mitigation procedures ended up being unnecessary.

Budget Violation (M): Although we were never very close to exceeding our budget from an overall sense, the timeline of the returns and warranties put us in a situation where we were less than $50 away from $5000 at one point. This resulted in one of our team members having to personally purchase a replacement sensor while we waited for refunds and returns to reimburse him several weeks later.

Computation Time (P): As mentioned in the Insufficient Data Rate explanation above, some of the more computationally expensive processes within our maneuver planning code ended up taking longer to run than we expected. The speeds and accelerations that our system was capable of producing were sufficient enough to make this issue irrelevant for the collision object speeds produced in our system, but a faster projectile speed would eventually make this problem much more significant.

VII. Project Planning

To accomplish the goals of this project, careful attention had to be paid to the schedule and planning of resources. This process involved the creation of an organizational chart, work breakdown structure, work plan, cost plan, and test plan. This section will cover these items as well as discuss how they were used throughout project development.

A. Organizational Chart

At the beginning of the year, the team was organized into the structure presented in the organizational chart in figure 44. These positions are outlined briefly below:

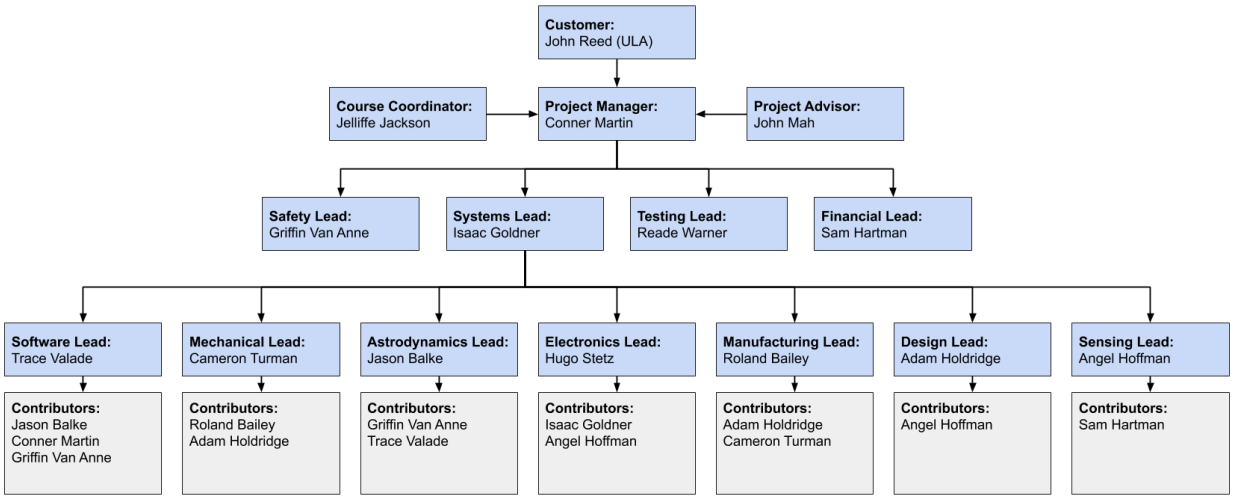

Fig. 44 Organizational Chart

B. Work Breakdown Structure

Through this organization structure, all teams members got the opportunity to fulfill a leadership role and contribute to a technical aspect of the project. The products for this project were split according to subsystem and class deliverables. Figure 45 gives the work breakdown structure. A key thing to note is the exclusion of manufacturing, design, sensing, and astrodynamics leads from the main categories. This is due to the work falling under a separate category. For example, the software development category has the maneuver planning task, which is primarily based on astrodynamics

work. However, because it is realized in software, it is better represented by this category. This allows the WBS to remain concise.

The work products themselves were based off key content required for class deliverables, integrating hardware to present a working prototype, and areas of improvement to the design that the team has identified throughout previous work.

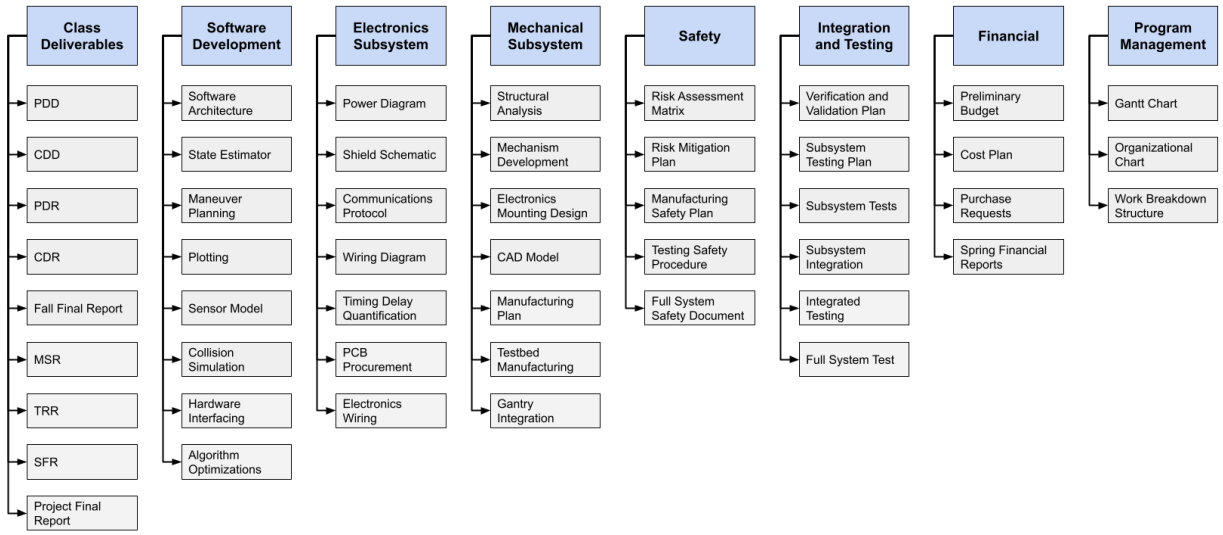

Fig. 45 Work Breakdown Structure

C. Work Plan

In order to complete the tasks outlined in the WBS, a schedule in the form of a Gantt chart was created and is shown in figure 46 for the fall semester and figure 47 for the spring semester. The critical path, outlined in red, follows the path of receiving, testing, integrating, and improving upon the gantry maneuvering subsystem. The maneuvering subsystem is the main driver of this critical path due to it's long lead and integration time. Outside of this system, tasks were organized in order to line up with moving between phases of testing. For example, the electronics are scheduled to be completed before the integrated testing occurs, while software can be continuously be worked on throughout the entire development period of the prototype.

Margins were allocated according to the uncertainty in time estimates of the work to be produced. Additionally, the gantt charts presented are in their final form, meaning that tasks have been adjusted to fall in line with when they were actually completed, however margin is still shown on the tasks that strongly needed the allocated margin. Developing these margins involved budgeting additional time for particularly difficult or risky tasks. Some tasks, such as software development, posed the risk of taking much more time due to bug fixing and unforeseen complications. Other tasks, such as PCB procurement, are given little margin due to either already having a working model, or a very clear idea of what work needs to be done to accomplish the task.

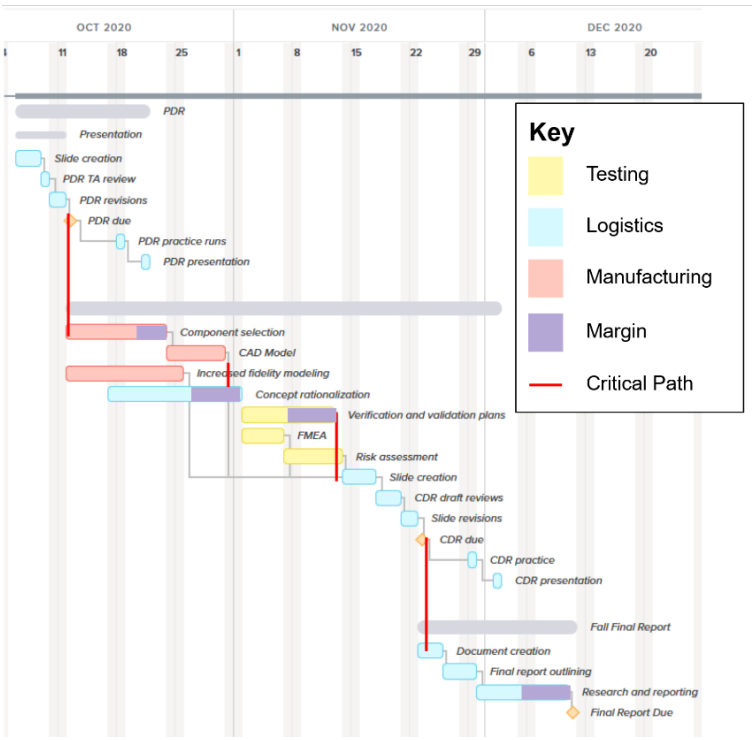

Fig. 46 Work Plan/Gantt Chart - Fall

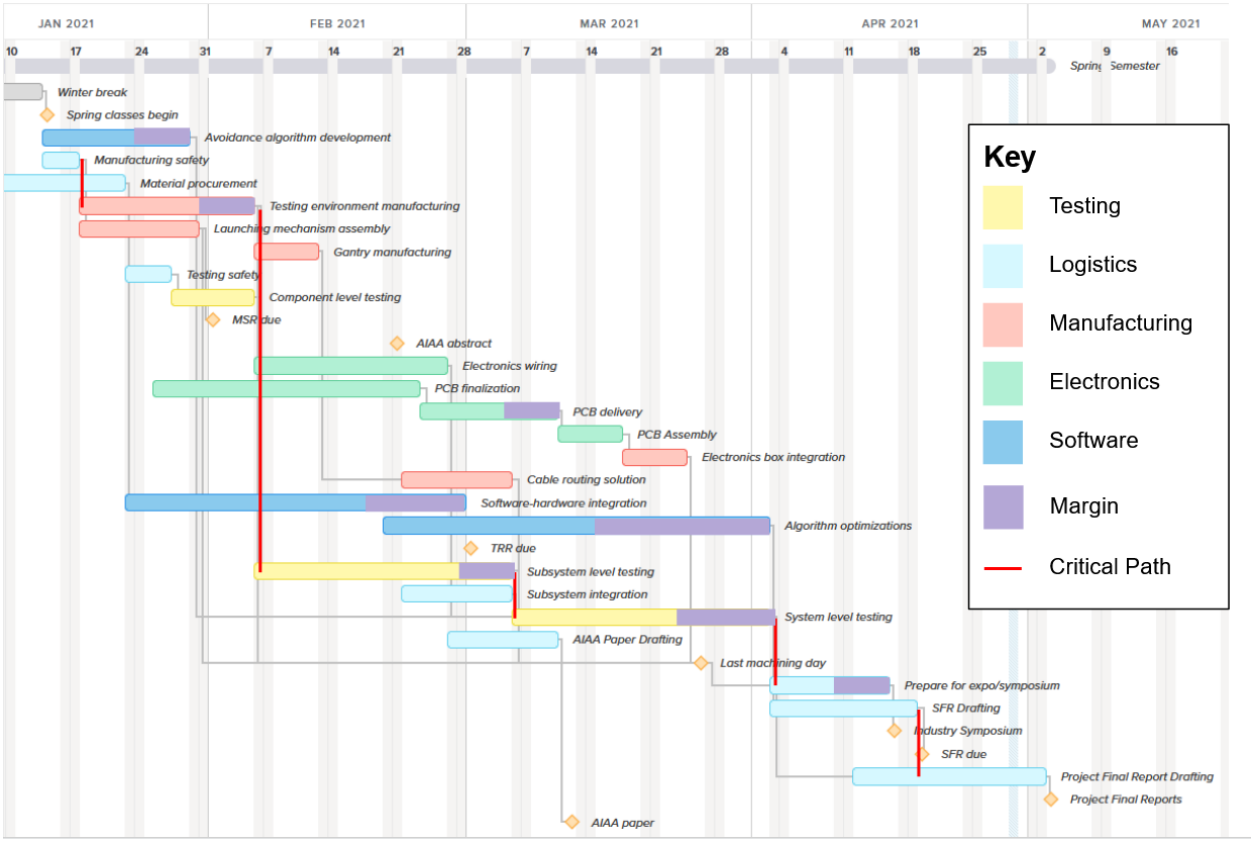

Fig. 47 Work Plan/Gantt Chart - Fall

D. Cost Plan

All major components were organized into a budget shown in figure 48. The largest purchase in this project was the gantry order from Igus ®. This custom gantry cost $3,000 and was the main purchase for the maneuvering element. Additionally, a significant purchase for the testbed was the 80/20 aluminum extrusion and associated components. These components, along with the MDF board and walls contributed to the testing environment element of the project. The electronics element consisted of components such as the Arduino, PCB, stepper drivers, and power supplies. The sensing element of the cost plan accounts for the $319 LiDAR sensor as well as accompanying accessories. The shipping costs are included into the cost plan presented. The team made some returns over the course of the semester which helped allow for the cost plan after manufacturing to be less than the budget presented at CDR. This allowed the team to complete the project under the $5,000 initial limit and pass the associated requirement.

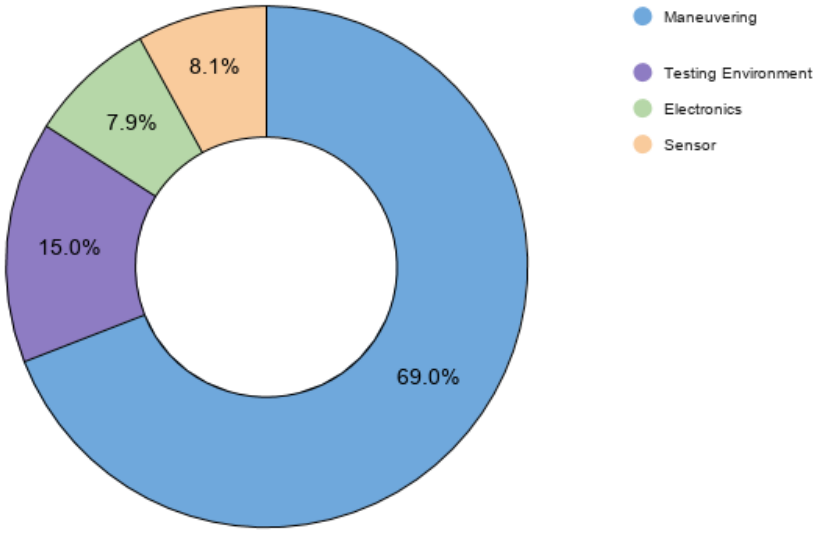

Fig. 48 Spring Cost Plan

Table 13 shows each element of the cost plan compared to the proposed budget at CDR. There are added margins to further compare to the CDR budget. The notes include reasoning for why there are changes between the CDR budget and final project cost plan.

Table 13 Cost Plan by Project Element

	CDR ($)	**PFR ($)**	**Margin ($)**	**Notes**
Maneuvering	3100	3000	+ 100	-Student Discount
Testing Environment	500	652	- 152	-Cable Chains -Shipping
Electronics	350	344	+ 6	
Sensor	330	350	- 20	-Tax
Total	4430 (+150 Shipping)	**4342**	+ 88	
Remaining	570	658	+ 88	

Figure 49 shows an itemized list of the major purchases by the project. Some of the purchases are via Amazon® and grouped into which element of the cost plan they fit under. This shows in a more detailed manner where the funds where allocated throughout the course of the project. This Bill of Materials also shows the final budget of the project as $4,342 which provides a 13% margin between the total budget amount of $5,000.

Direct Costs	Dec 20	Jan 21	Feb 21	Mar 21	Apr 21	Total
Gantry	$ 2,995.64					
Sensor (RobotShop)		$ 321.80				
Stepper Motor Drivers		$ 113.67				
MDF Boards		$ 69.82				
McMaster T-Frame Order		$ 451.48				
Acrylic Sheet		$ 26.58				
McGuckin's Electronics		$ 25.87				
McMaster Ramp Stand-Offs		$ 14.45				
McMaster Bolt/Ball Order		$ 44.31				
Amazon Order - Jan 14 (Electronics)		$ 55.97				
Amazon Order - Jan 20 (Electronics)		$ 52.98				
Amazon Order - Jan 31 (Electronics)		$ 9.99				
Amazon Order - Feb 09 (Electronics)			$ 24.12			
Amazon Order - Feb 11 (Electronics)			$ 55.98			
McGuckin's Bolts			$ 10.01			
Amazon Return - Open Loop Stepper Drivers			$ (55.98)			
McMaster Return - Bolts, Nuts, Brackets			$ (49.30)			
McGuckins' Bolts			$ 8.60			
PCB			$ 47.80			
Sensor Replacement (Amazon)			$ 319.00			
ABS 3D Printing				$ 19.99		
Sensor Return			$ (319.00)			
Sensor Return RobotShop				$ (319.00)		
Sensor Purchase (Not P-Card + Tax)				$ 347.22		
Sensor Replacement (Second Amazon)				$ 319.00		
Cable Extenstion USB				$ 13.58		
Cable Management Harnesses				$ 26.48		
Sensor Reimbursement (Conner Check)						
Sensor Return (Second Amazon)				$ (319.00)		
PLA 3D Printing Ticket					$ 30.00	
MonthlyTotal Expenses	$ 2,995.64	$ 1,186.92	$ 41.23	$ 88.27	$ 30.00	**$4,342.06**

Fig. 49 Itemized Bill of Materials

E. Test Plan

Table 14 outlines the test plan for team CAST during the 2021 spring semester. It shows each planned test, its completion date, and also highlights some of the necessary specialized equipment. The tests were split up into three main phases: component level testing, subsystem level testing, and full system level testing. Component level testing (tests highlighted in blue) was completed first, by the end of of February. Subsystem level testing (tests highlighted in yellow) was next to be completed. Finally, full system level testing (highlighted in red) was completed last, by April 18th.

The initial tentative schedule was aggressive and had margin built in. This promoted early completion of tests where possible and allowed for testing delays due to issues such as broken hardware. It can be seen that tests were finished before SFR due to the ambitious initial schedule.

Table 14 Test Plan

Test	Specialized Equipment	Date Completed
Launching Mechanism and Table Leveling	Frame by frame tracking software	02/18
LiDAR Sensor	Frame by frame tracking software	02/08
Software Unit Testing	N/A	02/15
Latency Testing	N/A	02/27
Gantry Command and Control	N/A	02/17
Gantry Positional Control	N/A	02/11
Gantry Velocity Control	N/A	02/26
Gantry Acceleration Control	N/A	02/26
Thrust Curve Matching	N/A	03/05
Vibration Characterization	N/A	04/02
Sensor/Software Integration	Frame by frame tracking software	04/18
Clear Miss Scenario	Frame by frame tracking software	04/18
Near Miss Scenario	Frame by frame tracking software	04/18
Collision Scenario	Frame by frame tracking software	04/18
Control Law Scaling	N/A	04/18
NEES/NIS Testing	N/A	04/16

VIII. Lessons Learned

Authors: Griffin, Conner

A. Project Specific Lessons

There are a handful of significant areas of improvement for this project that should be implemented if it is to be continued. These areas include: extended object tracking, real time operating system, filter tuning, and electronics improvements.

1. Extended Object Tracking

There is an inherent issue with the detection system on-board CAST. The scanning rate and sampling rate of the LiDAR system are high enough that multiple measurements of the incoming object are detected per scan. To reduce complexity, our team implemented a clustering system which averages the filtered measurements in a full 360° scan. However, this process means a loss of critical information about the shape and size of the incoming object. To better align with a situation in which there is no prior information about an incoming object, extended object tracking should be implemented. This involves using the measurements to figure out spatial information about the incoming object and then use this information within the state estimation process. This boils down to being able to track a rigid body rather than a point mass.

2. Real Time Operating System

CAST was implemented in MATLAB largely due to the ease of use of the Sensor Fusion and Tracking Toolbox. However, CAST ran into serious issues with serial overhead in MATLAB. To eliminate these issues, the next project iteration should be implemented using a real-time operating system. This has the benefit of also transitioning the software base to a compiled language, which will offer further speed and performance increases.

3. Filter Tuning

Filter tuning proved to be a source of much trouble throughout the year. In order make the process less troublesome, careful attention should be paid to the ground truth model. Sensor measurement noise can often be read off a datasheet, but process noise is something that needs to be intuitively tweaked. By having confidence in the ground truth model, NEES consistency testing can focus on the state estimator, rather than having to attempt to debug both the ground truth and the state estimator. For NIS consistency testing, the sensor, sensor model, and datasheet for the sensor measurement noise should all be verified against each other to ensure that output is uniform across the board.

Finally, having accurate data on the ground truth for live tests is important for testing the consistency of the live estimation process. Team CAST attempted to improve the accuracy of our video data by adding a grid to the testbed, however this was still not accurate enough. This could be further improved by using a high speed camera, or through an additional sensor outside the testbed whose sole purpose is to report the ground truth of the incoming object.

4. Electronics Improvements

Team CAST severely overestimated the computational burden that directly converting encoder signals to position measurements had on the software for the Arduino. To workaround this, in many of the tests the pulses sent to the drivers were directly counted rather than the encoder pulses. This gives a good short-term approximation of where the gantry is due to the use of closed-loop drivers, however error between the commanded stepper position and actual position on the gantry could accumulate over time. Thus it would have very beneficial to make use of an encoder buffer chip, such as the LS366R, to handle the encoder pulse counting. The use of such a chip would allow for encoder feedback to still be used, however it would minimize both the computational demands that processing the signal using software introduces, as well as minimizes the chances of skipping pulses in the counting process.

B. General Lesson

Throughout the year, the members of team CAST have learned many valuable lessons about the process of working with a project throughout its entire life-cycle. From the early project scoping phase to the production of the final report, the team realized that communication is just as critical as any technical aspect of the project. Due to the work-from-home nature of this year, effective and frequent communication amongst the whole team was even more important than in more ordinary circumstances. Along these lines, the team also learned that identifying issues early on can prevent roadblocks that may affect team progress in the future. If a member is stuck on a problem, communicating this with leadership allows for more resources and time to be allocated to this task, which ultimately benefits the success of the whole project. Effective communication in conjunction with well-made documentation also reduces the chances of work being repeated, which the team encountered several times throughout testing.

Over the course of the semester, countless tasks were delegated to the members of CAST. It was inevitable that at times, every member of the team would be working on something outside of their comfort zone. This made the simple act of starting tasks often times very difficult. To mitigate this, the team found that when experienced members walk through the beginning phases of an assignment with their less familiar teammate, the rate at which that task is completed is greatly increased. Further, it would have greatly benefited the team if every member was familiarized with the critical technical aspects of the project, such as the software.

C. Advice to Future Seniors

Senior design offers students a unique opportunity in their academic career to apply their classroom knowledge to real-world challenges and get a taste of what their future careers might hold. This was a great opportunity to learn many valuable lessons, and team CAST urges future students to treat it first and foremost as a *learning experience*. It is important to not let the challenges you encounter in this project consume all other aspects of your life. You are expected to face difficulties and fail at times in this class, and you should focus on the lessons you can learn from those shortcomings.

The multiple reviews and reports you encounter throughout the semester are sometimes stressful to navigate with all of the other courses you might be taking. The team members on CAST urge future teams to make organization and documentation of utmost importance throughout the entire year. Having designs, results, etc., on hand when it comes time to create reports and presentations makes the process much quicker and painless.

Acknowledgments

The team would like to thank John Mah for advising the team and getting us through the various direction changes. The project would not be where it is without his advice and continuous support.

The team would like to thank the customer John Reed for supplying us with this intriguing project. It has been an exciting concept to explore and a wonderful learning experience.

The team would like to thank Dr. Nisar Ahmed and his ASEN 5044 course for providing the required state estimation knowledge to make this project feasible.

The team would like to thank Captain Austin Sellers for providing advice and resource materials on the methods used in satellite conjunction assessment.

The team would like to thank Trudy Schwartz and Bobby Hodgkinson for their guidance to the team regarding electronics, including communications protocols, component assessment and selection, and discussions on component capabilities and configurations. The multiple planning and discussion meetings held between the CAST team and these technical experts enabled the team to create a viable plan for electronics and communications for the testbed.

IX. Individual Report Contributions

A. Adam Holdridge

1. Design Work

Concept of Operations, Full System CAD Model (Testbed, gantry system and standoffs, sensor mount and protector, electronics enclosure, launching mechanism)

2. Writing

Project overview and motivation, project deliverables, design solutions overview, mechanical design - sensor mount, and manufacturing scope - mechanical and electronics.

B. Angel Hoffman

1. Design Work

PCB design and revisions, sensor/software integration, sensor component level testing, vibration testing, bounding box calculations, sensor serial communication, LiDAR compatibility, Full system testing

2. Writing

Previous Work, CONOPs, sensor bounding box, sensor model design solutions, sensor model verification and validation, and manufacturing-electronics sections of the report.

C. Cameron Turman

1. Design Work

Manufacturing of MDF, Securing Gantry, Aluminum extrusions, gantry mounting, electronics box mounting, ramp testing, full system testing, cable chain mounting and mounting design.

2. Writing

Igus Gantry, Integration, Risk Identification, Risk Tracking and Mitigation, Risk Impact, Appendix Material.

D. Conner Martin

1. Design Work

State estimation, forward prediction, thermal analysis, measurement clustering, sensor model, org chart, WBS, Gantt chart, CAD, EKF implmentation, vibration testing, subsystem integration, full system testing documentation and

operation.

2. Writing

Measurement clustering, probability calculation, organizational chart, work breakdown structure, work plan, lessons learned, project objectives, levels of success

E. Griffin Van Anne

1. Design Work

Arduino to Matlab interface, Arduino gantry control code, serial communications, subsystem integration and testing, planar deviation analysis.

2. Writing

Sub-system level testing, Lessons Learned, Advice to Future Seniors, planar deviation (appendix)

F. Hugo Stetz

1. Design Work

Electronic component research and selection, electronics wiring, encoder testing, and limit switch operation and integration.

2. Writing

General formatting and nomenclature; Design Solution: Electronics (overall organization, Component Selection, Wiring and Communications, and Timing Delays); Risk Assessment and Mitigation (Risk Identification and Risk Tracking and Mitigation).

G. Isaac Goldner

1. Design Work

Systems engineering requirements, levels of success, electronics wiring, electronics power design, electronics stepper driver selection and operation, subsystem integration, thruster blowdown model, full system test analysis and plotting

2. Writing

Functional Block Diagram, high-level functional requirements, requirements flow-down, electronics wiring and power, full system testing results, thruster blowdown model (appendix)

H. Jason Balke

1. Design Work

Development of maneuver planning, probability calculation, EKF, NIS/NEES testing, preliminary development of motor control

2. Writing

Design solution state estimation, probability calculation, maneuver planning, NIS/NEES testing

I. Reade Warner

1. Design Work

Development of functional and design requirements, development of levels of success, some work on problem scaling, overall project scoping, gantry assembly, created entire test plan, performed tests, analyzed test data.

2. Writing

Levels of Success Table, Validation against Functional Requirements, Test Plan.

J. Roland Bailey

1. Design Work

Ramp performance design, manufacturing of MDF, Aluminum extrusions, gantry mounting, electronics box mounting, ramp testing, cable chain mounting pieces.

2. Writing

Ramp mechanical design, component level testing section, manufacturing challenges.

K. Sam Hartman

1. Design Work

Levels of success, thruster blowdown model, testbed assembly, electronics fastening and electronics box mounting

2. Writing

Manufacturing scope, Testing overview, Sensor capabilities, Full system testing, Cost Plan

L. Trace Valade

1. Design Work

Software architecture, version control management, scaled simulation, rplidar packet decoders, logging infrastructure, development of software requirements, software timing and testing

2. Writing

High level architecture (software), latency considerations, acceleration tracking, simulation scaling

References

[1] Garcia, M., "Space Debris and Human Spacecraft," 2016. URL https://www.nasa.gov/mission_pages/station/news/orbital_debris.html.

[2] Racelis, D., "Two-Line Element Error Analysis for MEO and GEO Satellites," University of Arizona, 2018.

[3] Aeronautics, N., and Administration, S., "NASA's Efforts to Mitigate the Risks Posed by Orbital Debris," *NASA Office of Inspector General Office of Audits*, 2021. URL https://oig.nasa.gov/docs/IG-21-011.pdf.

[4] "ClearSpace One - A Mission to make space sustainable," , 2020. URL https://clearspace.today/.

[5] *dryve® D1 Motor Controller*, igus, 2020. URL https://www.igus.com/info/dryve-motor-controller.

[6] *Closed Loop Stepper Driver*, STEPPERONLINE, 2020. URL https://www.omc-stepperonline.com/closed-loop-stepper-driver/.

[7] *Accessories*, igus, 2020. URL https://www.igus.com/contentData/wpck/pdf/global/drylin_E_accessories_EN.pdf.

[8] *User Manual: CL57T Closed Loop Stepper Driver*, STEPPERONLINE, 2020. URL https://www.omc-stepperonline.com/download/CL57T.pdf.

[9] *drylin® step motor NEMA23*, igus, 2020. URL https://www.igus.com/info/drive-technology-nema-23-ca.

[10] *RPLIDAR A2 Introduction and Datasheet*, SLAMTEC, March 2018. URL https://www.robotshop.com/media/files/pdf2/ld208_slamtec_rplidar_datasheet_a2m8_v1.1_en_2_.pdf.

[11] *RPLIDAR Interface Protocol and Application Notes*, SLAMTEC, April 2016. URL https://www.robotshop.com/media/files/pdf2/rpk-02-communication-protocol.pdf.

[12] *What are the USB data transfer rates and specifications?*, Sony, September 2020. URL https://www.sony.com/electronics/support/articles/00024571.

[13] Mathur, R., "Choosing the right motor-driver," 2017. URL https://sproboticworks.com/blog/choosing-the-right-motor-driver.

[14] Fagerlund, S., "Target Tracking Based on Bearing Only Measurements," 1980.

[15] Patera, R., and Peterson, G., "Space Vehicle Maneuver Method to Lower Collision Risk to an Acceptable Level," *Journal of Guidance, Control, and Dynamics*, Vol. 26, 2003, pp. 233–237. https://doi.org/10.2514/2.5063.

[16] *drylin® XY gantries & cartesian robots*, igus, 2020. URL https://www.igus.com/info/xy-gantry.

[17] Lohani, B., Chacko, S., Ghosh, S., and Sasidharan, S., "Surveillance system based on Flash LiDAR," *Indian Cartographer*, 2013.

X. Appendices

A. Electronics

	Current	Voltage
Wall Adapter: *Output*	0-2 A	5 V
Sensor: *Input*	0-1.5 A	5 V
Computer: *Output*	0-500 mA	5V
Arduino: *Input*	0-200 mA	5V

(a) Arduino and Distance Sensor Operating Power

	Voltage
Encoder: *Output*	5 V
Stepper Motor Driver: *Input*	5 V
Arduino Digital: *Input*	5V
Stepper Motor Driver: *Output*	5 V
Encoder: *Input*	5 V

(b) Encoder Operating Voltage

Fig. 50 Supplemental Electronics Components Data

	Current	Voltage
DC Power Supply: *Output*	0-4.5 A	12 V
Limit Switch: *Input*	0-300 mA	12 V
Limit Switch: *Output*	5 mA	5 V (voltage divider)
Arduino: *Input*	20 mA	5 V

(a) Limit Switch Operating Power

	Current	Voltage
Arduino Digital: *Output*	0-20 mA	5 V
Stepper Motor Driver Control Signal: *Input*	7-16 mA	4.5-24 V
DC Power Supply: *Output*	0-11 A	36 V
Stepper Motor Driver Power: *Input*	0-8 A	24-48 V

(b) Stepper Motor Driver Operating Power

Fig. 51 Supplemental Electronics Components Data

	Current	Voltage
Stepper Motor Driver: *Output*	0-8 A	36V
Stepper Motor: *Input*	4.2 A	24-48 V

Fig. 52 Stepper Motor Operating Power

B. Planar Deviation and Linearity

Assuming circular LEO orbits with no perturbations, the equations of motion for the objects are shown below:

$$|v| = \sqrt{\frac{GM}{r}}, \vec{a} = -\frac{v^2}{|r|} \tag{19}$$

Using the equations above and MATLAB's Runge Kutta method function, *ode45*, circular orbits for orbits are propagated. By adjusting initial velocity directions, multiple collision scenarios were tested. It was found that over a 100 second time interval before collision, the maximum deviation of the incoming object from the observation plane of the sensing object is only 4°. This small planar deviation means that the CAST can be restricted to only 2D relative motion while maintaining reasonable fidelity to a full-scale orbital collision geometry.

Using the same equations listed in 19, the relative trajectory of an incoming orbital object was determined for a range of collision angles. These results are plotted below in figure 53.

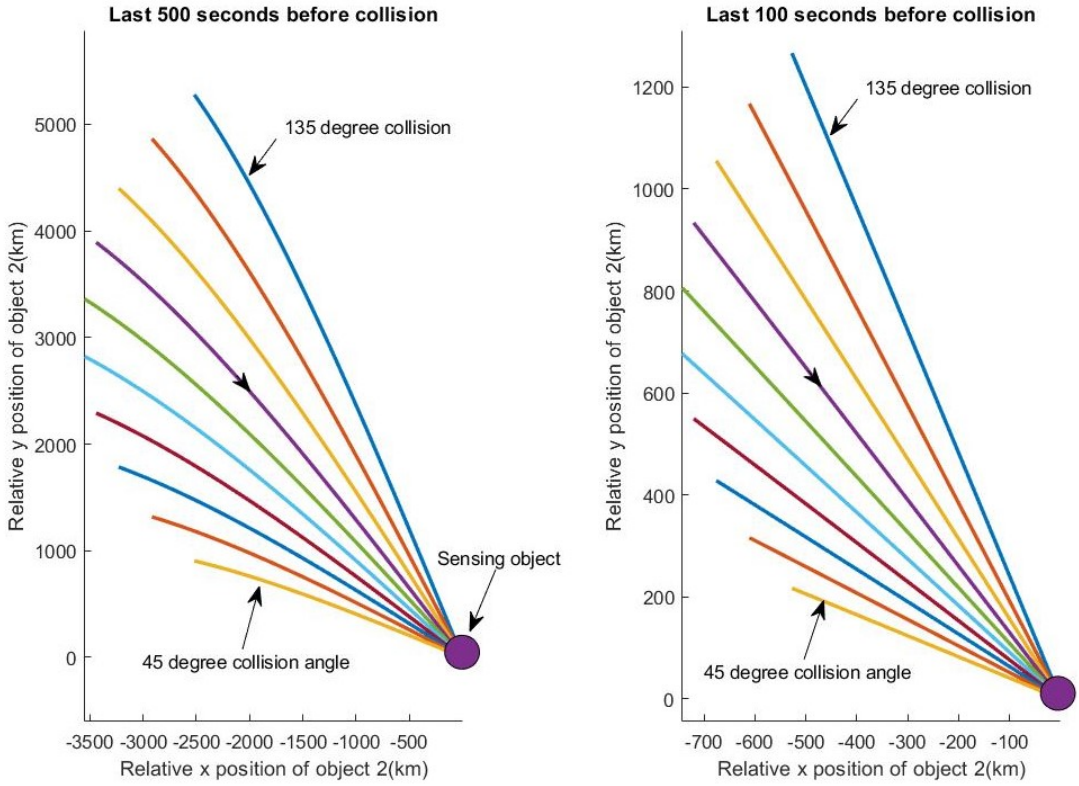

Fig. 53 Relative Position of Incoming Object with Respect to 2D Observation Plane of Sensing Object

For the shorter time interval of 100 seconds before collision no non-linearity is visibly present in the motion between the two objects. The coefficient of determination of a linear fit of the trajectory over this interval is 0.999, indicating linear motion is very representative of the actual motion. It is reasonable to simplify the collision scenario to have constant relative velocity between the two objects due to the linear motion seen in the full-scale scenario. Since the deviation from linearity in the relative motion is so small, the testbed collision will need to maintain a high degree of linearity and constant relative velocity for these aspects of the orbital collision geometry to be preserved.

C. Thruster Blowdown Model

In order to satisfy Functional Requirement 4, the maneuvering system must follow the acceleration profile of a thruster characteristic of a satellite on orbit. Therefore, a thrust-time profile has been developed for a hydrazine monopropellant blowdown thruster. Specifically, the MR-107s thruster is modelled by solving the ordinary differential equation for the rate of change of propellant volume over time. This thruster is chosen as a baseline due to its current use in industry and available thrust to avoid a last-minute collision scenario. The key assumptions included in this model are a discharge coefficient (C_D) equal to 0.8, the density of hydrazine is constant, the pressurized gas in the propellant tank is N_2, and the specific impulse is constant from beginning of life to end of life. The resulting thrust-time profile is shown in Figure 54. Based on the timescale of these results, for a short-duration burn, such as the one being performed in the collision avoidance system testbed, the acceleration can be well-approximated by a linear decrease in thrust over time.

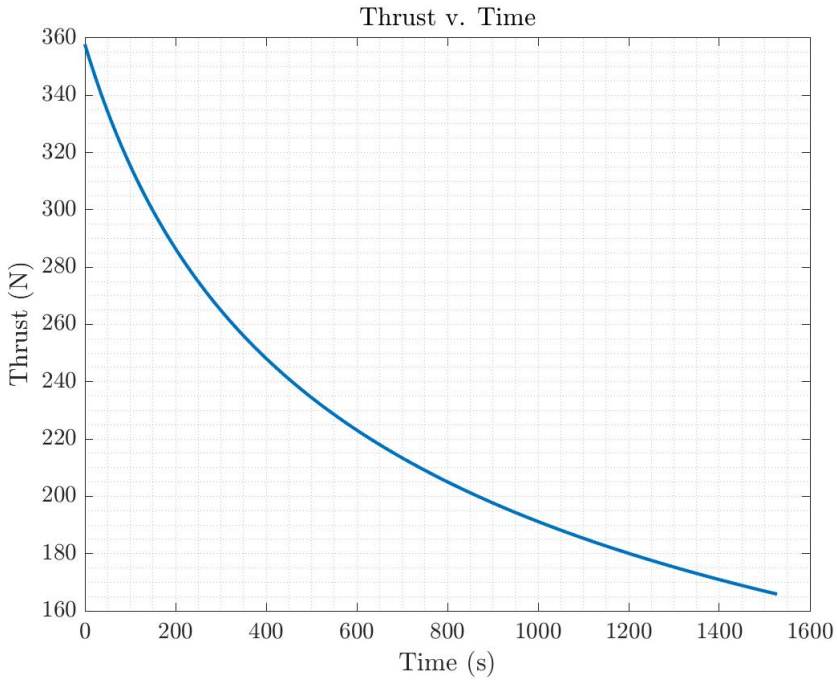

Fig. 54 Thrust Blowdown Curve Over Time

XI. Design Selection

A. Sensor

Authors: Conner, Sam, Angel

1. Options Considered

The ability to sense a potential colliding object is necessary for CAST to be able to demonstrate that collision avoidance is possible. Several types of sensor technologies were considered including laser rangefinders, optical, LiDAR, and radar. Laser rangefinders are used for long-distance sensing and work via a time-of-flight method. This method

measures the time a laser takes to travel to an object, reflect off it, and return to the sensor. This is then calculated to determine it distance. A LiDAR system is similar and can either scan or use lenses to increase the field of view. Scanning LiDAR uses mechanical machinery to physically rotate the laser beam and scan the environment. Flash LiDAR operates by spreading the beam through the use of lenses or other means [17]. Figure 55 gives a quick overview of the differences between the LiDAR technologies. Optical systems can use the visible light or infrared spectrum to collect positional data. Finally, radar works by emitting radio waves to collect positional data of an object. Each sensor technology has their pros and cons which can be seen in greater detail in table 15.

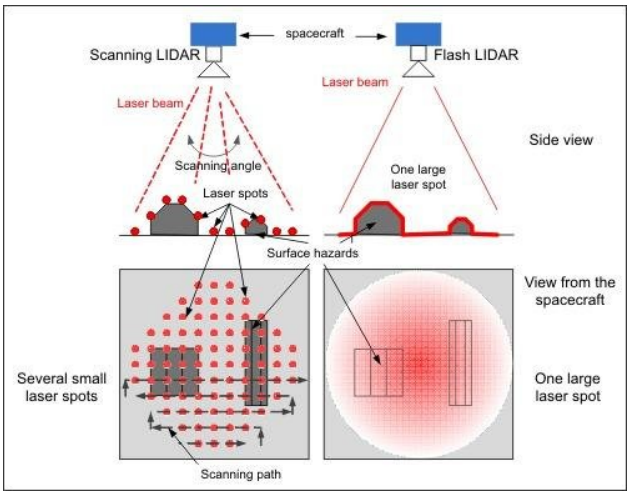

Fig. 55 Comparison between scanning and flash LiDAR

This technology is able to maintain much of the accuracy and distance benefits of the laser rangefinder, while increasing the field of view. However, these benefits come at a cost. In order to provide 2-D data, the source laser must be spread over a greater area. This is usually done by either scanning LiDAR or flash LiDAR. Scanning LiDAR uses mechanical machinery to physically rotate the laser beam and scan the environment. Additionally, since the laser cannot be simultaneously pointed in two directions at once, this limits the sampling rate for a single measurement position. Flash LiDAR operates by spreading the beam through the use of lenses or other means [17].

A list of the pros and cons for each of the sensor options considered can be found in table 15.

Sensor	Pros	Cons
Laser Rangefinder	Extremely fast readings High accuracy	Single beam Need precise aiming
Optical	Can collect whole image at once Wide FOV	Requires light source Need multiple measurements to get depth/distance measurements
LiDAR	Can detect smaller objects High accuracy Small package size No geometry distortions	Shorter operating distance Accuracy depends on material properties
Radar	Large range of available frequencies Possible wide beam width	Decreases usefulness as distance increases Package size depends on antenna requirements Object shape changes range of detection

Table 15 Pros and Cons for Each Sensor Option Considered

2. Rationale

This trade study will investigate the capabilities and specifications of different available sensor technologies that come in commercial off the shelf packages. The optimal sensor has a long range, high sample rate, large field of view (FOV), and low cost.

3. Criteria and Weighting

Table 16 Sensors: Weighting

Criteria	Weight	Driving Reqs.	Description
Range	0.45	DR 2.3	The sensor needs to be able to detect an object at a far enough distance so a successful maneuver can occur.
Cost	0.1	DR 1.4	It is required that the project remains within the project budget, the sensors contribute to the cost.
Sample Rate	0.3	DR 2.6	The sample rate dictates whether or not there will be enough measurements in a sufficient amount of time.
FOV	0.15	DR 2.4	The field of view determines the angle that is observable by the sensor. The object must be within this angle for the sensor to pick it up.

4. Trade Study

The scoring table is presented in table 17.

Table 17 Sensors: Scoring

	Scoring				
Criteria	**1**	**2**	**3**	**4**	**5**
Range	<25 m	>25 m	>50 m	>100 m	>150 m
Cost	>$2000	>$1000	>$500	>$100	<$100
Sample Rate	<10 Hz	<100 Hz	<500 Hz	<1 kHz	>1 kHz
FOV	0-5 deg	5 - 15 deg	15 - 25 deg	25 - 35 deg	>35 deg

The trade study for the sensor technology is given in table 18, the scoring description for this trade study can be found in table 17.

Table 18 Sensors: Trade Study

	Weight	Laser Rangefinder	Optical/Visual IR	LiDAR	Radar
Range	0.45	4	1	5	4
Cost	0.1	4	3	3	4
Sample Rate	0.3	4	3	2	2
FOV	0.15	1	5	3	3
Total	1	3.55	2.4	3.6	3.25

5. Evaluation of Results

The highest scoring sensor technology was LiDAR with a score of 3.6/5. Laser Rangefinder and Radar were close runner-ups with an overall score of 3.55/5 and 3.25/5 respectively. These sensor technologies operate based off similar principles so the similar scoring makes sense. LiDAR is capable of sensing objects at long ranges that makes it valuable for the testbed environment. LiDAR was lacking the most in the sample rate. The field of view and cost for LiDAR make it a promising sensing technology.

B. Maneuvering Subsystem

Authors: Roland, Cameron, Conner

1. Options Considered

During the initial development and definition of CAST, many different maneuvering systems were considered. The ones most seriously considered were: drones, an air table used in conjunction with a cold gas thruster module, a High Density Polyethylene (HDPE) sheet used in conjunction with a cold gas thruster module, a DIY gantry on a hard surface, and a commercial gantry on a hard surface. This gantry works via two stepper motors driving the x and y linear rails. If one of the two gantry-based options was selected then an additional trade study was to be conducted to determine the best surface.

The concept of operation for the drones was to have one or two quad copter style commercial drones which could either have an object launched at them or be set on a collision course with each other. This option is very intriguing as it allows for a wide array of tests, but is very complex and would be hard to relate to the space environment. The two Cold Gas Thuster Module (CGTM) options would rely on placing a CGTM on top of a low friction surface (air table or HDPE) and then launching objects at that CGTM. The final two options would rely on a gantry robot which is a combination of linear rails in multiple axes that can produce motion in any of those axes simultaneously. The best way to implement one of these options would be to mount a 2-D gantry onto a hard surface and then use that surface to launch an object at the gantry. This gantry can either be prebuilt or manufactured by the team. A list of the pros and cons for each of the options is provided in Table 19.

Maneuvering Subsystem	**Pros**	**Cons**
Drones	High-speed capable Easy construction & command and control	Expensive Potential difficulty in automating control responses High baseline technological competency Only usable in open air environment
CGT on Air Table	Similar design to a real thruster One table currently available for free	Difficult to accurately generate a specific force Potential to damage test items Difficult to upscale Varying frictional effects
CGT on HDPE	Similar design to a real thruster Constant frictional force	Difficult to accurately generate a specific force Potential to damage test items
DIY Gantry	Flexible testbed material options Medium-speed capable Fully customizable acceleration/force abilities	Increased complexity/design time Manufacturing required
Commercial Gantry	Flexible testbed material options Medium-speed capable Limited manufacturing required	Communication with external company required Increased expense

Table 19 Maneuvering subsystem List of Pros and Cons for Each Option Considered.

2. Criteria and Weighting

Five criteria were selected for consideration in the trade study on maneuvering hardware. These criteria and assigned weightings are summarized in table 20.

Table 20 Maneuvering Hardware Weighting

Criteria	Weight	Driving Reqs.	Description
Cost	0.2	DR 1.4	How much the method costs to implement.
Complexity	0.3	FR 1	The greater the complexity of the solution, the worse the score.
Acceleration	0.3	FR 4	Needs to be able to at least generate enough force to model a smallsat spacecraft thruster.
Range of Motion	0.1	DR 1.2	The number of degrees of freedom available when with the considered hardware.
Repeatability	0.1	DR 1.3	How much preparatorp work must be done between each test.

3. Trade Study

A full description of how the maneuvering hardware trade study was scored is provided in Table 21. The result of the trade study is displayed in Table 22. In this latter table, the weights of all the criteria are shown as well as scores for each maneuvering method (total, as well as for each criteria). The maneuvering method with the highest score was chosen for the final design.

Table 21 Maneuvering Hardware Scoring

	Scoring				
Criteria	**1**	**2**	**3**	**4**	**5**
Cost	>$1000	>$500	>$250	>$100	<$100
Complexity	Needs to be developed from scratch/no known working examples	Requires significant design considerations and machining, but has been accomplished before	Multiple working examples to base design off of. Some machining required	Requires minor assembly from readily available solutions	Commercially available
Acceleration	< 0.125 m/s^2	0.125 - 0.25 m/s^2	0.25 - 0.5 m/s^2	0.5 - 1 m/s^2	> 1 m/s^2
Range of Motion	Device can only move in one direction	Device can move in two directions	Device can move in three directions	Device can move in all directions in 2D plane	Device can move in all directions in 3D plane
Repeatability	Can only perform the test once without extensive component maintenance, alterations or replacements	Can perform the test once with significant required maintenance after each set	Can perform the test with minimal required maintenance after each set	Can perform the test more than once, with minimal amounts of maintenance/ alterations required	Can perform the test multiple times without the need for part adjustments or replacements

	Weight	Drones	CGT on Air Table	CGT on HDPE	DIY Gantry	Commerical Gantry
Cost	0.2	2	2	2	1	1
Complexity	0.3	1	2	2	2	5
Acceleration	0.3	3	3	2	4	4
Range of Motion	0.1	5	4	4	4	4
Repeatability	0.1	2	2	2	5	5
Total	**1.0**	**1.49**	**2.5**	**2.2**	**2.9**	**3.8**

Table 22 Maneuvering Hardware Trade Study

4. Evaluation of Results

The highest scoring option was the commercial gantry at 3.8/5. This is due largely in part to its reduced complexity and ease of repeatable operation. Since the design software is expected to take the largest effort and development time, being able save on time and effort by purchasing a ready-made gantry provides a valuable opportunity. For this team CAST will contract out to Igus®the manufacturing of a completed gantry system so that team CAST can spend more time and effort on the detect and react component of the project. Next, a trade study will be conducted to determine what surface the gantry will be mounted to.

C. Base Structure/Material

Authors: Cameron, Isaac, Conner, Adam

1. Options Considered

In choosing a material for the base structure of the testbed design, CAST examined five potential options: Plywood, Aluminum, Medium Density Fiberboard (MDF), High Density Polyethelene (HDPE), or Available Ground. The following tables and explanations detail the general pros and cons and a provide a brief description of each material. Aside from the available ground option, the potential options will all consist of sheets of material (approximately 2x1m) mounted on short metal legs a few inches above the ground with a launching mechanism of some sort mounted to one end. A collision object will be rolled along this platform toward a gantry that is mounted on the opposite end of the platform from the launching mechanism. A list of the pros and cons of various base structure materials is shown in table 23.

Base Structure/Material	Pros	Cons
Plywood	Available Relatively Durable Inexpensive Easy to Manufacture	Somewhat Rough Surface Lacking Professional Look Easily Warped
Aluminum	Extremely Smooth Professional Looking Very Durable	Expensive Thin Does Not Maintain Shape Well Hard to Manufacture
MDF	Fairly Smooth Fairly Professional Looking Easy to Manufacture Hard to Warp Inexpensive	Screws Not Great to Use Not as Durable Size Control Limited
HDPE	Extremely Smooth Somewhat Professional Looking Hard to Warp	Difficult to Manufacture Somewhat Expensive Not as Durable
Ground	Free No Extra Work	No Manufacturing Capability Variable Smoothness Fixed Location for Same Ground

Table 23 Base Structure/Material Pros and Cons List

2. Criteria and Weighting

In order to determine the best choice for the material of the base platform, weighting categories were determined based on what CAST values in a base material according to driving requirements (Table 24).

Table 24 Platform Material: Weighting

Criteria	Weight	Driving Reqs.	Description
Smoothness	0.2	DR 1.5	The surface should be smooth enough to ensure that it does not significantly effect the trajectory of the collision object
Manufacturability	0.3	DR 1.1	The material should be as easy as possible to modify, mount, and adapt to avoid frustration and time wasted on a less important aspect of the overall test system
Expense	0.25	DR 1.4	The material should be as cheap as possible to avoid straining the budget
Durability	0.15	DR 1.3	The material should be available in multiple thicknesses and be durable enough to support the required loads of the system.
Portability	0.1	N/A	The system should as portable as possible to give flexibility to where tests and demonstrations can be performed

3. Trade Study

Next, these weighting categories were given weights and associated scoring criteria on a 1-5 scale (Table 25. This mixture of objective and subjective scoring criteria was then applied to each of the material options in a trade study and the selection was the winner of this trade study (Table 26).

Table 25 Platform Material: Scoring

	Scoring				
Criteria	**1**	**2**	**3**	**4**	**5**
Smoothness	5th	4th	3rd	2nd	1st
Manufacturability	Subjective	Subjective	Subjective	Subjective	Subjective
Expense	$560.55-$448.44	$448.44-$336.33	$336.33-$224.22	$224.22-$112.11	$112.11-$0
Durability	0-18 MPa	18-36 MPa	36-54 MPa	54-72 MPa	72-90 MPa
Portability	80+ lbs	60-80 lbs	40-60 lbs	20-40 lbs	0-20 lbs

Table 26 Platform Material: Trade Study

	Weight	Plywood	Aluminum	MDF	HDPE	Available Ground
Smoothness	0.20	3	5	4	4	1
Manufacturability	0.3	4	2	4	3	0
Expense	0.25	4.42	0	4.6	4.21	5
Durability	0.15	1.72	5	0.56	1.83	5
Portability	0.10	1.19	3.58	1.76	0	0
Total	1	3.28	2.7	3.41	3.03	2.2

4. Evaluation of Results

The resultant selection for the material of the base platform is MDF board with a score of 3.41/5. This score is primarily due to it's cheap cost, smoothness, and ease of manufacturing. MDF was found to perform poorly in durability, however this can be counteracted by containing the MDF within the testing environment such that it is securely placed and taking care when manufacturing. Particularly, the team needs to be careful when drilling holes to make sure the MDF does not "blow out" or disintegrate. This risk is easily mitigated with proper manufacturing strategies.

D. Launching Mechanism

Authors: Roland, Conner, Cameron, Adam

1. Options Considered

Several options were considered in approaching the problem of launching a detectable object for the gantry to react to. In this consideration the detect-ability and predictability of the launched object were weighed the most important. For the detect-ability aspect, the choice of a LiDAR sensor necessitated the use of a highly visible object. This means that any launched object chosen needed to be white in color so that it is reflective enough to ensure proper sensing. With that consideration in mind there were a few options for the actual delivery method of the launched object. The ideas discussed involved two distinct types of delivery, a linear rail system and a rolling system. The first option, the linear rail system, essentially would use a linear rail similar to those used on the gantry solution to bring a detectable object towards the sensor. This option would be highly predictable but was quickly deemed overly complex and as a result was thrown out. The final three options all relied on the same idea of rolling a spherical object towards the sensor. The rolling motion was determined to be useful as it necessitated generally less complexity and allowed for a greater range of launcher selections. The rolling motion also maintains a constant enough velocity to meet the design requirements and will be shown in the Detailed Design section.

The three options considered that utilized rolling motion were a pneumatic launcher, a spring loaded launcher, and a ramp. The pneumatic launcher was quickly thrown out because it would be complex, costly, and would be hard to predict the motion of the ball if spin was induced not in the rolling direction. The spring launcher and the ramp were the two main options considered. A spring launcher would have essentially been a device that would depress a spring a certain amount and then the ball could be loaded up to a plunger on the spring. When the spring is released the spring potential energy would convert to kinetic energy for the ball rolling. Concerns for this method were raised in how well the ball would stay up against the plunger and if it did not the system would become an inelastic collision which would make it much less predictable. For this reason, and the fact that the ramp does not have many downsides, the ramp was chosen. The ramp operates on the general principle of converting gravitational potential energy into kinetic energy, as show in figure 56.

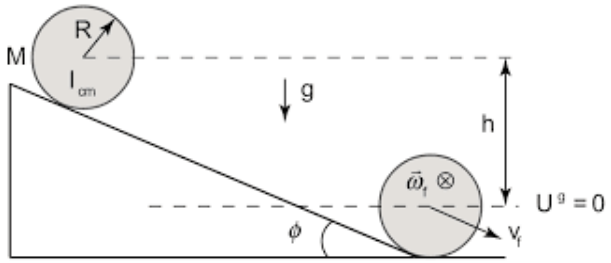

Fig. 56 Rigid Body Dynamics for Ramp Design

Using rigid body dynamics the ball's velocity can be calculated based on the height at which it was released on the ramp. This ramp can also be moved along the test bed's Y axis and pivoted about the z axis to provide a whole range of launching angles for the gantry to react to. This increase in launch angles provides more utility in matching the test results to the space environment. A list of the pros and cons for each option considered can be found in table 27.

Launching Mechanism	Pros	Cons
Pneumatic	Highly adjustable design Accurate force generation	Complex Costly Hard to predict spinning motion
Spring-loaded	Cheap Can find off the shelf solution	Hard to adjust velocity
Ramp	Easily adjust speed Easily adjust angle of collision	Requires manufacturing

Table 27 Launching Mechanism List of Pros and Cons for Each Option Considered

2. Criteria and Weighting

Six criteria were chosen to evaluate the trade study on the launching mechanism. These criteria and assigned weightings are given below and summarized in table 28.

Table 28 Launching Mechanism Weighting

Criteria	Weight	Driving Reqs.	Description
Speed	0.3	DR 1.1, 1.3, 1.5	The collision object needs to be able to produce varying velocities and replicate these various velocities fairly accurately. It also needs to be launched in a manner that maintains nearly constant velocity following launch.
Complexity	0.2	DR 1.3	The lower the complexity of the launching mechanism, the more time can be dedicated to other key areas of the project. Simple solutions are also more likely to be robust for multiple trials.
Repeatability	0.2	DR 1.3	The collision needs to occur consistently and produce nearly identical cases for repeated testing launches.
Cost	0.2	DR 1.4	In order to stay under budget we need to be conscious of budget and produce a solution that does not overly strain the current projected cost plan.
Launch Angle	0.1	1.1, 1.2	The launching mechanism should be capable of varying launch angle reliably and accurately.

Table 29 Launching Mechanism Scoring

	Scoring				
Criteria	**1**	**2**	**3**	**4**	**5**
Speed	0 - 0.25 m/s	0.25 - 0.5 m/s	0.5 - 1 m/s	1 - 2 m/s	>2 m/s
Complexity	Needs to be developed from scratch/no known working examples	Requires significant design considerations and machining, but has been accomplished before	Multiple working examples to base design off of. Some machining required	Requires minor assembly from readily available solutions	Commercially available
Repeatability	Can perform test once before needing to replace major components or fabricate new ones	Requires significant time to reset test environment to original state. Likely to put testing schedule on hold between runs	Can repeat tests with some time between each test devoted to resetting the test environment	Can repeat tests multiple times in a row. Small portion of time required between sets of tests	Can repeat tests with no alterations to the testbed between runs
Cost	>$1000	>$500	>$250	>$100	<$100
Launch Angle Range	0 Degrees	0 - 10 Degrees	10 - 20 Degrees	20 - 30 Degrees	30 + degrees

3. Trade Study

The trade study for the launching mechanism is given in table 30, the scoring description for this trade study can be found in table 29.

	Weight	Pneumatic	Spring	Ramp
Speed	0.3	5	4	5
Complexity	0.2	1	3	3
Repeatability	0.2	1	4	5
Cost	0.2	3	4	5
Launch Angle	0.1	4	5	5
Total	**1**	**2.9**	**3.9**	**4.6**

Table 30 Launching Hardware Trade Study

4. Evaluation of Results

The ramp launching mechanism was selected with a high score of 4.6/5. The reasoning for this score lies primarily in its adaptibility to multiple collision scenarios. This can be achieved by simply starting the incoming object at a larger height on the ramp, with exact heights calculated via rigid body dynamics. Additionally, this ramp can be placed on a simple hinge that allows it to rotate with respect to the testing environment and provide multiple launch angles.