Northeastern University

Capstone II

Fall 2017

Hyperloop Control System
Final Report

Team Members
Matan Kaminski
James Massucco
Luke Merkl
Jared Miller
Justin Pietrocarlo
Younes Sherkat

Advisor
Dr. Bahram Shafai

Group Number
B1

December 17, 2017

Contents

1 Introduction **3**

2 Problem Formulation **4**

3 Analysis **5**
- 3.1 Dynamics Model . . . 5
- 3.2 Control Hardware . . . 5
  - 3.2.1 Control Hardware: Trains . . . 5
  - 3.2.2 Control Hardware: Cars . . . 6
  - 3.2.3 Control Hardware: Background on Networks . . . 6

4 Design **7**
- 4.1 Dynamics Model . . . 7
  - 4.1.1 Dynamics: Bag Model of Levitation . . . 8
  - 4.1.2 Dynamics: Disturbance Effects . . . 10
  - 4.1.3 Issues with the Model . . . 11
- 4.2 Control Scheme . . . 11
- 4.3 Control Hardware . . . 12
  - 4.3.1 Control Hardware: Signal Conversion Devices . . . 12
  - 4.3.2 Control Hardware: System Interconnection . . . 13

5 Parts and Implementation **14**
- 5.1 Avionics Node Board . . . 14
  - 5.1.1 Avionics Node: IC Selection . . . 14
  - 5.1.2 Avionics Node: Schematic . . . 16
  - 5.1.3 Avionics Node: Layout . . . 16
- 5.2 System Demonstration . . . 17
  - 5.2.1 Demo: Hardware . . . 17
  - 5.2.2 Demo: Software . . . 17
  - 5.2.3 Demo: Communication . . . 18

6 Cost Analysis **18**

7 Conclusion **18**

8 Appendices **19**
- 8.1 List of Tables . . . 19
- 8.2 List of Figures . . . 19
- 8.3 Parameters of Heave Dynamics Model . . . 20
- 8.4 Outputs of Heave Dynamics Model . . . 21
- 8.5 Avionics Node Board 3D Render . . . 29

9 References **30**

Abstract

The Hyperloop Control System addresses two key challenges of hyperloop technology: stability and intra-pod communication. For the question of stability, the air levitation system of Paradigm Hyperloop was examined in detail. This system was modeled based on extensive research on hovercrafts and validated against actual system operation. The model enabled simulating track unevenness at high speeds. Simulations suggested that stability could be guaranteed through a combination of mechanical design optimizations and active control of air flow. An avionics system was also developed to actively control air flow. The system employs custom avionics boards that communicate over an Ethernet network to a central controller. They utilize a unique protocol designed to guarantee real-time data transmission and simple processing of commands. The avionics boards serve as general sensor and actuator interface modules. They can be placed around the pod, connected to nearby sensors and actuators, and will enable real-time command and control of these devices. The dynamics model and avionics system go a long way towards helping Paradigm Hyperloop reach 600mph during their testing in Summer 2018.

1 Introduction

Advances in transportation technology tend to yield economic and social benefits. Whenever a new method of travel comes about and gains popularity, the geography of trade and transit shift, and the world becomes increasingly interconnected. When new links between locations get forged, a greater variety of goods and ideas can be traded which adds diversity and value. Speeding up these links means that goods can be delivered at a much greater volume and lower cost, and people can more frequently travel to a much larger set of locales. So far, there have been five waves of economic development which were spurred by new transportation philosophies: seaports, canals, railways, automobiles/roads, and airways/information [2]. Each of these introduced new logistical modes, and increased mobility of people, goods, and services. Hyperloop aims to be a sixth revolution in transportation, bringing the world closer together like never before.

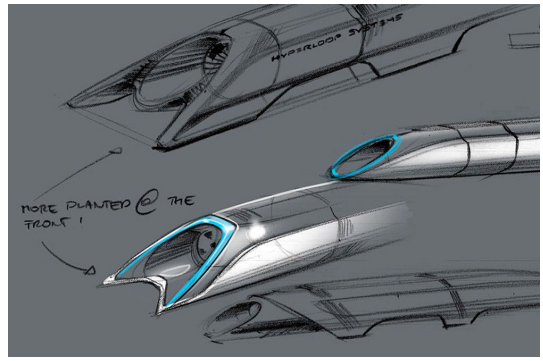

Figure 1: Hyperloop conceptual design [1]

The technology behind Hyperloop dates back long before Elon Musk's 2013 "invention." The concept of a vacuum-tube train (vactrain) has a history going back over 100 years. Robert Goddard formulated the details behind vactrain implementation in 1906, and his estate was granted a patent in 1950. The patent describes a maglev vehicle traveling in an evacuated tube [3]. Physicist Robert M. Salter published a paper in 1972 detailing the Very High Speed Transit system (VHST), where he claimed that with 1970s technology, pods could travel at speeds of 1400mph in tunnels across the US. This would result in an LA to New York travel time of 21 minutes [4].

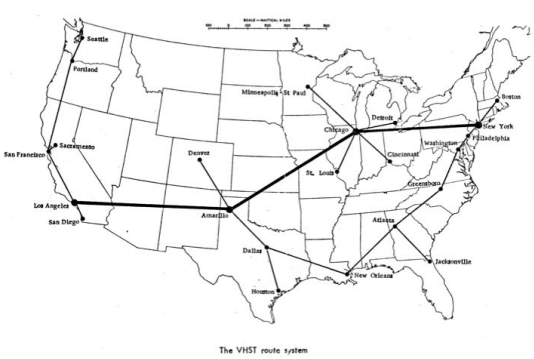

Figure 2: Proposed locations of stops and links of VHST [4]

After the bloated budget and disappointing results of the California High Speed Rail project of the 2000s, Elon Musk introduced the term "Hyperloop" in 2012. Its main inspiration is to traverse the 700 mile commute between San Francisco to Los Angeles in 35 minutes. In general, a single Hyperloop hop is designed to travel between any two destinations less than 900 miles apart. Assuming the technological issues are ironed out, building a Hyperloop would cost an estimated $6b as compared to high speed trains' $60b [1]. Musk immediately open sourced the idea, and now private companies such as Hyperloop One and HTT are working on R&D and physical implementation. Hyperloop has also become grounds for a contest between universities, and this crowdsourcing approach is enabling research innovations.

Paradigm Hyperloop, a team of students working at Northeastern University in partnership with several other schools, have been investigating hyperloop technology since 2015. The team has made significant advances in high speed air levitation. The technology they've developed has been termed "air skates." They used this technology to levitate on air at 60mph at the Hyperloop SpaceX Competition in August, winning 2nd place. To date, they are one of the leading investigators of this technology and they have plans to test it at up to 600mph next year.

The Hyperloop Control System focuses on continuing the work of Paradigm Hyperloop by tackling some of the key electrical and controls challenges of the project, and of the air levitation system in particular.

2 Problem Formulation

During early testing of Paradigm air skate technology, a prototype hyperloop pod was outfitted with four air skates and was pressurized with air. Upon activation of the skates, the pod exhibited significant oscillations. Although these were tuned out of the system during successive iterations of the technology, they were shown to still exist to a lesser extent during vacuum testing. This raised a significant concern about the stability of this levitation mechanism. The analysis of the pod's dynamics and stabilization of levitation are the primary problems addressed by the Hyperloop Control System.

Figure 3: Paradigm Hyperloop Pod, constructed by students from Northeastern and Memorial University, Newfoundland. Pictured outside Northeastern University

A control problem such as stabilization of pod levitation has a few key components. The first is an accurate model of the system dynamics. Accuracy of the model is determined by validation against the observed motion of the system. Once a model is formed, the region of stability must be determined. If the operation of the pod is determined to fall outside this region of stability during any point, then an active control scheme must be employed. Assuming that this is the case, there must then be an electronic system capable of implementing the active control scheme.

The goal of the Hyperloop Control System is to produce a pod dynamics model, a control policy to ensure stability of levitation, and an electronics system capable of implementing this control policy.

3 Analysis

This section details the foundational research and problem analysis that contributed to the Hyperloop Control System. This research falls into the categories of Dynamics Model and Control Hardware. Details on the Control Scheme can be found in Section 4.

3.1 Dynamics Model

The impetus for the Hyperloop Control System was up-and-down (heave) vibrations of the pod, and these vibrations were not being controlled due to the open-loop control scheme. No underlying model for these vibrations existed, and the only resources were numerical CFD models and tribal wisdom (i.e. increasing the volume of the skate before the bag decreases stability). A first approximation of pod dynamics was through a quadcopter model, in which the main similarity is that of rigid body motion with applied force on the bottom [5]. Quadcopter dynamics include a roughly linear angular rotor speed to lift force relation, while no such simple connection exists for the hyperloop pod. Unraveling of the flow-to-force conversion led to the development of a full dynamic model of air skates. These air skates involve air retained in a bag lifting up a mass, which draws similarities to existing models and design of air cushioned vehicles [6]. This exploration is at the heart of this project and is explored in great detail in Section 4.1.1.

3.2 Control Hardware

A control scheme dictates a functional relationship between a set of inputs and a set of outputs. In a simulation, these may take the form of numbers or waveforms, but in reality they are physical quantities. Inputs require a transducer or sensor in order to translate a physical quantity into a voltage or current. Outputs require some form of device, typically called an actuator, in order to execute some physical action. And the control scheme itself must execute on a computing device which connects to the sensors and actuators of the system.

In order to understand what control hardware might be appropriate for the Hyperloop Control System, control systems for trains and cars are analyzed and their applicable features pointed out.

3.2.1 Control Hardware: Trains

Trains have one main control decision: stop or go. Additionally, when they are going, their speed must be controlled. Because of their history, these decisions are traditionally made by human operators. However, as train infrastructure has grown more advanced, the need for additional control mechanisms has emerged. In the United States, all trains employ positive train control (PTC), a technique that requires an "all clear" sign for the train to be moving at any time. A diagram of a PTC system can be seen in Figure 4. PTC systems rely on sensors on the train as well as a network of sensors on the track

to measure train position and velocity. [8] In advanced systems, sensors also monitor track temperature, vibrations, and mechanical failures. [9] Each sensor node has local decision making abilities and is able to report failure conditions directly, rather than simply reporting sensor data. These sensors are linked together by wireless (mainly satellite) communication networks. Sensor data is transmitted to regional control hubs. Based on additional information such as schedules as well as control parameters, the control hubs send signals to each train under their control to allow movement. [7]

The connection of sensors over a network in PTC systems is highly advantageous. Adding new sensors is as simple as connecting them to a network and the network routes the data appropriately. Additionally, the wireless networks used on trains allows for the easy addition or removal of train cars without needing to do any wiring. Finally, the network allows for tremendous parallelism in communications.

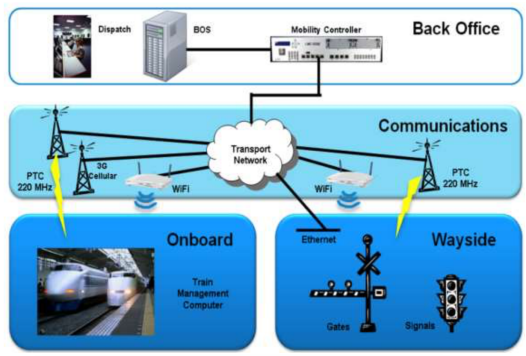

Figure 4: A positive train control (PTC) system consists of sensors connected to a central back office server through a multi-component network. [7]

3.2.2 Control Hardware: Cars

Cars have a wide variety of control systems ranging from suspension to engine timing to windows. Typically, each of these systems has a unique electronic control unit (ECU) comprised of a processor which takes in a sensor inputs and sends commands for actuators (e.g. window motor). [10] The ECUs communicate with sensors and actuators and with each other over a CAN bus. The CAN bus is a two-wire serial bus shared by many devices. [11] The CAN bus not only allows communication between sensors, ECUs, and actuators, but it also enables OBD-II diagnostic systems used by mechanics and savvy car owners to troubleshoot car problems.

The control architecture of a car is distributed, since each ECU handles a different component of the car's overall operation. ECUs can communicate with other ECUs to get information they need. This architecture is beneficial for cars mainly because it allows for reporting of many different types of failures through a single port. There is also a centralized component to the control of a car, since the major control decisions that determine car motion come from the driver. This type of system has the advantage of making control decisions easier, since one single "computer" knows the state of the system and from this computes the appropriate control output (e.g. car veering to right -> steer left).

3.2.3 Control Hardware: Background on Networks

Networked communication systems form the backbone of the control hardware for both trains and cars, so their basic operation is described in more detail here. One of the most popular models for understanding network operation is the Open Systems

Interconnection (OSI) model. The OSI model breaks the functionality of a network into successive layers of abstraction, as shown in Figure 5. This figure also shows how data flows between a user, a network, and back.

The lifecycle of a group of data, called a packet, within the OSI model starts at the Application Layer (also known as Layer 7). As transmission of this packet proceeds, it is encapsulated with additional wrappers, or headers, for each layer. These headers contain information about where the data is going and how to get there. As packets of data get transported from device to device they get unwrapped depending on which layer the device is acting on. For example, a Data Link Layer (Layer 2) switch will unpack the packet to get to the Data Link Layer headers in order to determine how to forward the packet to the next device. [12]

Generally speaking, fears of using networks for control systems stem from the speed at which a network can send packets, as well as congestion on a unregulated network.

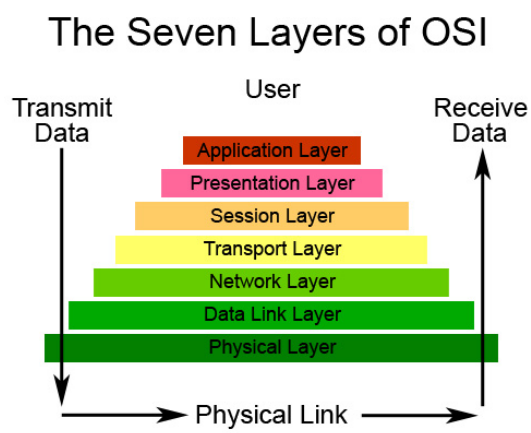

Figure 5: OSI Network model with data transmission and data receipt shown [13]

4 Design

This section details the creation of the Dynamics Model, the formulation of the Control Scheme, and the high-level architecture of the Control Hardware.

4.1 Dynamics Model

The original goal of this section of the project was to create a full dynamics model of the hyperloop pod. A universal controller would be used with a three stage pipeline: state estimation, controller policy, and desired actuation. Sensor data would be denoised and observable states would be inferred through a state estimator, and this cleaned up data would be passed to the controller. The controller would figure out what kind of force would be required, and processors on nodes near the actuators would translate desired force into air flow. As pleasant as this scheme would have been, the dynamics of the pod's motion are far too complicated for this work. The connection between force and flow is extremely nontrivial, and the disturbances of the system may be faster than the sampling rate of sensors and frequency responses of the actuators. A model of pod heave motion was created and validated, and stability of this system was investigated. Many other subsystems on the pod have internal control loops with communication facilitated by the avionics board. An example is the sensor fusion of an IMU with light strips through a Kalman filter to guess horizontal position and velocity, which can be controlled through bang-bang (minimum time) methods with

cold gas output or engine/wheels and brakes. The heave levitation subsystem deserves special consideration, and is the focus of this part of the paper.

4.1.1 Dynamics: Bag Model of Levitation

Paradigm Hyperloop pod 2 is kept aloft by four air skates. Each air skate is composed of three valves to provide air, a series of diffusion layers, and a perforated nylon fabric facing the outside. When air flows through the system, the bag inflates and the net pressure difference keeps the pod up. Throughout this process, the majority of the bag is touching the ground.

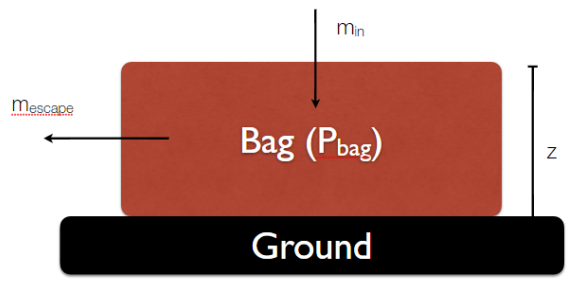

Figure 6: Bag Model, air flows through top and out sides

This is a nonlinear system with three states: the height of the skate z, the vertical speed of the skate $\dot{z}$, and the pressure inside the bag P_{bag}. The height of the bag changes through gravity and the gauge pressure of the bag. In reality, the equation $F = PA$ is invalid, but that is an acceptable approximation. This forms the first equation of motion:

$$M\ddot{z} = (P_{bag} - P_{atmo})A - Mg$$

Here, M is the mass of the pod, P_{atmo} is the outside pressure (101 kPa at atmosphere or 0.66 kPa at tube pressure) and A is the area of the skate. If there are multiple skates, then all quantities are treated as net ($A_{net} = N * A_{skate}$).

Equations of motion for the pressure inside the bag are much more involved, and are at the heart of this study. In 2013, Ahmed Sowayan presented equations of motion of the heave dynamics of air cushioned vehicles, focusing on skirted hovercraft hovering on a hard cushion of air [14]. The relevant equation governing the change in pressure underneath the craft is equation 10 in the 2013 paper:

$$\dot{p} = \frac{\gamma RT}{V}\left(\dot{m}_{in} - \dot{m}_{out} - \frac{pA\dot{z}}{RT}\right)$$

Where the volume $V = Az$ and the volume gradient $\dot{V} = A\dot{z}$. This equations states that the change in pressure is proportional to the net mass flow in the volume, dependent on applied mass flow $\dot{m}_{in} = u$, air escaping the bag $\dot{m}_{out}$, and the change in the mass of air due to changes in the bag volume $\frac{pA\dot{z}}{RT}$. This paper uses the compressible Bernoulli equation to find the mass outflow, and creates the complicated expression:

$$\dot{m}_{out} = \frac{c_0 pLz}{\sqrt{RT}}\left[\frac{2\gamma}{\gamma - 1}\left(\left(\frac{p_{atmo}}{p}\right)^{\frac{2}{\gamma}} - \left(\frac{p_{atmo}}{p}\right)^{\frac{\gamma+1}{\gamma}}\right)\right]^{\frac{1}{2}}$$

modified by an efficiency constant c_0. This expression is suitable for a skirted hovercraft, where there is no porous boundary layer between the pressurized air underneath

the craft and the atmosphere. Simulation of this open-skirted approach has shown that the resultant ride height under $\dot{m}_{in}^* = 0.16\frac{kg}{s}$ is 0.885 mm, and this equilibrium is unstable. The air skates' bags provide a means to retain pressurized air. A first approximation for flow across a porous boundary can be modeled through Darcy's law:

$$\dot{m}_{escape} \approx -\frac{\rho_{atmo}\kappa A_e}{\mu}\frac{\partial P}{\partial x}$$

Darcy's law is a linear relation between pressure gradient across the porous material and outgoing mass flow rate. κ is the porosity of bag, x is the bag thickness, and A_e is the effective area in which air can escape ($Per * z$). Using the Darcy approximation results in an extremely fast pole (-40,000 rad/s) and a very low ride height at equilibrium (0.68 mm) since air quickly escapes the bag. In reality, the holes in the bag result in choking effects (impingement), since air needs to travel through the throat diameter. This is better modeled by Forchheimer's equation, which adds a quadratic penalty to mass flow [15]. The higher order penalty means that less air will escape the bag at a given height, meaning that the equilibrium ride height is higher.

$$\rho_{atmo}\frac{\partial P}{\partial x} \approx \beta(\frac{\dot{m}_{escape}}{A_e})^2 + \frac{\mu}{\kappa}(\frac{\dot{m}_{escape}}{A_e})$$

β is an inertia flow parameter which penalizes mass flow rate, μ is the viscosity of air at the given temperature, and κ is the porosity of the bag. Non-darcy flow as expressed by Forchheimer better approximates true pod dynamics, sliding the pole from s = -40,000 to -14.8. The time constant of the system in pod liftoff is predicted to be 0.4 seconds, which matches exactly video of the pod.

The accumulated equations of motion that describe the heave dynamics of the pod are:

$$M\ddot{z} = (P_{bag} - P_{atmo})A - Mg \tag{1}$$

$$\dot{P}_{bag} = \frac{\gamma RT}{Az}(\dot{m}_{in} - \dot{m}_{escape} - \frac{P_{bag}A\dot{z}}{RT}) \tag{2}$$

$$\rho_{atmo}\frac{\partial P}{\partial x} = \beta(\frac{\dot{m}_{escape}}{A_e})^2 + \frac{\mu}{\kappa}(\frac{\dot{m}_{escape}}{A_e}) \tag{3}$$

$$A_e = Per * z \tag{4}$$

$$\frac{\partial P}{\partial x} = \frac{P_{bag} - P_{atmo}}{\ell} \tag{5}$$

The equilibrium ride height is a function of the nominal applied mass flow rate $\dot{m}_{in}^*$. The mass flow equation can be reformulated as:

$$\frac{\beta}{Per^2}\left(\frac{m_{in}^*}{z^*}\right) + \frac{\mu}{\kappa Per}\left(\frac{m_{in}^*}{z^*}\right) = \rho_{atmo}\frac{\partial P}{\partial x}$$

This equation can be solved for $\frac{m_{in}^*}{z^*}$. Given m_{in}^*, this equation can be used to find the resulting ride height z^*. Going the other way, a target z^* can be attained by supplying

mass flow m_{in}^*. The bag pressure is constant $P_{bag}^* = P_{atmo} + \frac{Mg}{NA}$, where N is the number of bags.

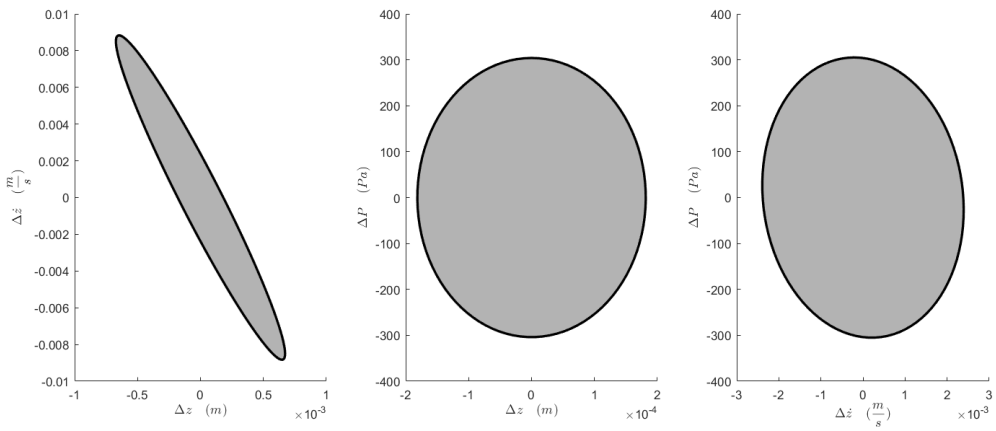

Figure 7: certified region of stability for pod in atmosphere (underestimate of true region of attraction)

4.1.2 Dynamics: Disturbance Effects

The dominant disturbance of the system is changes in the subtrack height. The track is made of a series of 12.5' plates, with a gap tolerance of 0.25". Between plates, there is a drift tolerance of ±0.004" with a maximum accumulated drift of ±0.4" away from nominal height [16]. This single disturbance hides two separate phenomena. When the skate is hovering over a gap in the track, there is a greater area of the bag exposed to air, so m_{escape} increases (w_{gap}). Drifts in the track height force a new equilibrium ride height, and pull down/push up the bottom of the bag (w_{height})). The full equations of motion including effects of disturbances are described below:

$$M\ddot{z} = (P_{bag} - P_{atmo})A - Mg \tag{6}$$

$$\dot{P}_{bag} = \frac{\gamma RT}{A(z + w_{height})}(\dot{m}_{in} - \dot{m}_{escape} - \frac{P_{bag}A\dot{z}}{RT}) \tag{7}$$

$$\rho_{atmo}\frac{\partial P}{\partial x} = \beta(\frac{\dot{m}_{escape}}{A_e})^2 + \frac{\mu}{\kappa}(\frac{\dot{m}_{escape}}{A_e}) \tag{8}$$

$$A_e = Per * (z + w_{height}) + W w_{gap} \tag{9}$$

$$\frac{\partial P}{\partial x} = \frac{P_{bag} - P_{atmo}}{\ell} \tag{10}$$

w_{gap} is the length of the plate gap underneath the skate, so the total additional area exposed to air is $W w_{gap}$. A greater exposed area raises the outgoing mass flow rate of air, which in turn reduces the equilibrium ride height z^*. w_{height} is the height

of each subtrack component, which does not effect the mass flow rate of air. Instead, it instantaneously changes the volume of the bag, and the bag responds accordingly. Given a track change w_{height}, the new equilibrium track height is $z^* \Leftarrow z^* + w_{height}$.

The two disturbances are each L_∞ bounded, based on the geometry of the track. If the underlying system is open loop stable, the influence of the disturbances are each Input to State Stable (ISS) [17]. The gap disturbance has extremely low energy, and has almost no effect on the pod's heave motion. The height disturbance is the major exogenous input of the system, and has the greatest effect on pod motion. All disturbances have the same poles and zeros as the original system when linearized, but with slightly different gains. The H_∞ norm of the height disturbance is 20 times greater than the original system, and rangefinder sensors can only tell the disturbed position of the subtrack.

Numerical simulations, linearizations, and analyses of the above model can be found in Appendix 8.4.

4.1.3 Issues with the Model

True dynamics of the hyperloop pod require a full solution of the Navier-Stokes equations. This model of motion ignores spatial dependence of the velocity field and pressure distributions, which give rise to interesting partial differential equations [18] (at steady state they appear like anti-helmholtz equations). Full spatial dependence will show that in a puck model (no bag), gauge pressure vanishes at the sides of the bottom face. In terms of the bag model, this involves fluid travelling between two plates, and air will probably more easily escape the center than the boundaries. When the pod is moving, Couette and Poiseuille flow will lower the outside pressure and change $\dot{m}_{escape}$ [19]. Couette flow also provides a drag force, not to neglect the drag coming from the shearing of the nylon bag on the ground. All simulations were performed with constant velocity, and acceleration/decceleration will assuredly change the response to disturbances. The quadratic Forchheimer flow rate is a rough guess of the true behavior, and testing needs to be done to emperically determine the κ and β parameters. The model treats z as the top of the bag, and ignores the bottom of the bag. In reality, the nylon bag has an elasticity that resists stretching (E = 2-4 GPa), which also means under a height disturbance that the bag may fall towards the new track height and slowly increase the bag volume. This model is an comparatively simple approximation to the true dynamics of air levitation, and it is remarkable that it performs as well as it does.

In competition 3, an air levitation scheme composed of planar air skates will be used instead of bags. At speeds of 400-600 mph, there is a clear and present danger of the nylon bags tearing and causing system failure. This capstone analysis has introduced a methodology of formulating and investigating dynamics instead of design through an ad-hoc approach in CFD.

4.2 Control Scheme

The underlying system of a hyperloop's heave dynamics is highly nonlinear and stiff. Standard linear controllers may result in finite escape time, even when the system is

open loop stable. One way to synthesize controllers is through region of attraction maximization through sum of squares techniques, but this requires a first guess on a Control Lyapunov Function (CLF) [20]. The bilinear algorithm to synthesize a controller through this scheme requires an initial guess on a Lyapunov function, which could not be attained within the timeline of this project. Further work will be done to investigate possible control schemes, and open loop control for the heave dynamics will remain in place. In the true pod implementation, there will be nonlinear observers and sensor fusion to process the input, and a nonlinear control scheme to stabilize the pod (discrete control of a continuous time system). As a primarily rigid-body system, a hyperloop pod lends itself to centralized control. This means that control decisions are made by a single computer that retains a knowledge of the holistic system state. [21] This is in contrast to a decentralized/distributed approach, such as controlling agents in a drone swarm. Although decentralized and redundant control have benefits in terms of safety and automation, it adds unnecessary complication to our pod with marginal benefit. [22]

4.3 Control Hardware

A centralized control system requires a method for translating between a control loop running on a digital processor and the analog inputs and outputs of an actual pod system. Past experience discourages simply running analog wires from all sensors and actuators to a central location, and instead encourages the use of local analog-to-digital and digital-to-analog conversion. This raises two design questions: what devices perform these conversions, and how do they connect to the central control device?

4.3.1 Control Hardware: Signal Conversion Devices

Converting signals between the analog and digital regimes is a very common problem, and so the solutions for transformations in both directions are well understood. Devices known as analog-to-digital converters (ADCs) allow for converting electrical signals from sensors or transducers into discrete-time digital data streams. The inverse operation is performed by digital-to-analog converters (DACs).

ADCs and DACs don't typically have much logic beyond the conversion operations. If it is desired to organize these signals and move them around a system, a general purpose processor or other device must be present to facilitate. This application requires a processor for this reason. Some tasks this processor might do are to control administrative functionality of the ADCs and DACs to keep them running. It might also pre-process data in order to extract useful parameters. Once the processor has organized the data, it must interface with the central controller. This is discussed in Section 4.3.2.

Given that long analog signal runs are to be avoided and there are many sensors and actuators in disparate locations, it seems clear that multiple groupings of ADCs and DACs with a processor will be present around the system. The simplest solution to this design is to create one signal board that is capable of being replicated and copies placed around the pod, each able to fill the needs of that specific area or subsystem

on the pod. This board will be called the avionics node board going forward. Since a general avionics node board is to be designed, all of the sensors and actuators and sensors in the system must be compatible with this board. It is thus relevant to make a brief inventory of the interfaces of each of the pod sensors and actuators.

Actuators are a relatively simple breed, and only come in two varieties in this application. The most common is a solenoid type valve, which simply has two wires: a supply and a return. To actuate the valve, the supply is either connected or disconnected. A few proportional valves are employed on the pod, requiring a more sophisticated interface. These actuators accept a supply and return as well as an analog signal proportional to the desired valve position (or other configurable parameter). These valves require DACs to generate these analog signals. For most valves, a supply of 24V is the standard. Based on previous experience, each avionics node board should support 8 solenoids and 2 proportional valves.

Sensors come in a larger variety. In general, most sensors accept a supply and a return and provide an analog signal proportional to some physical quantity. The supply is typically 5V but can sometimes be 24V. The analog signal is typically an analog voltage, but can sometimes be an analog current (this is more common for 24V-supplied sensors). In addition, there is an important class of sensors called a thermocouple which does not require and power and simply provides a very small differential voltage proportional to the temperature of the probe. Each of these types of sensors requires an ADC, and some require additional analog conditioning circuits in order to convert them to a voltage signal in the proper range of the ADC. Based on previous experience, each avionics node board should support 16 sensors. Of these, all should be standard inputs (0-5V) with some able to be reconfigured for alternate functions. In terms of alternate support there should be 6 thermocouple inputs, 4 current inputs, and 2 24V voltage inputs.

Finally, it bears mentioning that all of the hardware on these avionics nodes will need to be powered. The pod will be outfitted with a battery rail measuring a minimum of 27V and a maximum of 42V. It is reasonable to predict that the avionics nodes will require 24V and 5V rails based on the sensors and actuators mentioned above. It is also likely they will need 3.3V rails for digital devices like the processor.

4.3.2 Control Hardware: System Interconnection

Based on the use of networks for interconnection in trains and cars, this paradigm seems to also be appropriate for the Hyperloop Control System. When specifying a network technology, it is traditional to chose the hardware first followed by the software. Since the network may span reasonable distances, and since it is desirable to use a common network technology, IEEE 802.3 (Ethernet) is the most sensible hardware interconnection method. This standard defines the type of cabling and the final translation of digital information into analog transmissions on this cabling. [23]

In order for the avionics node boards to be compatible with an Ethernet network, they will network interfaces including both a physical port and a translation device to turn serial, digital data into the analog signals that travel down Ethernet cables.

Based on the choice of Ethernet for the physical transmission medium, a few other

design choices become sensible and are briefly mentioned. First, Ethernet networks are traditionally setup where nodes link to each other through one or more devices called switches. Switches enable the routing of data to the appropriate node or nodes based on the address data contained in the packet. A switch-based architecture linking the avionics node boards to the central controller device makes sense for this project because it utilizes common hardware and make sense from a cable routing perspective.

While hardware layers (typically Layers 1 and 2) of a network are bounded by the availability of materials due to the complexity of their design, there is significantly more flexibility for higher layers. The majority of the Hyperloop Control System interconnection design is focused on Layers 3 and above.

For the design of the communication scheme over the network, heavy consideration was given to packet size and congestion on the network. There was a requirement for the system to respond at sub-millisecond speeds.

To accomplish this, we designed our communication scheme around UDP as we are operating on a not-lossy network and as a result the overhead of establishing a TCP stream will have little to no benefits in this application. We set about establishing a "Talking Stick" protocol in which the controller on the network would tightly control which node was talking on the network at any given point in time.

With these basic controls in place, we added functionality to reduce the overhead of sending and receiving messages that rarely changed by giving the controller the ability to subscribe to certain sensor values and only receive an update if there is a change to the state.

Additionally the controller can broadcast messages to all nodes for critical information such as an emergency state transition where it is imperative that all the nodes start safing their systems in the appropriate order, as fast as possible.

5 Parts and Implementation

This section focuses on the implementation level details of the Hyperloop Control System. It deals only with hardware implementation details.

5.1 Avionics Node Board

The design process for the avionics boards first focused on selection of major integrated circuit (IC) components. The second step was circuit design and schematic capture. Lastly was the layout. Each had very different and unique challenges, and are discussed in the following sections.

5.1.1 Avionics Node: IC Selection

The primary components of the avionics node are the processor, sensor ADC, actuator DACs and solenoid drivers, network interface, and the power supply.

Processor choice needs to balance two competing design requirements: computational power to prevent bottlenecks and ease of use/deployability. The Arduino family are all very easy to program, so a processor from this family would be an ideal choice.

One of the most advanced processors in this family is the Atmel SAMD21, found on the Arduino M0. [24] This processor is a 32-bit ARM Cortex M0, providing great performance for embedded applications like this one while still enabling Arduino IDE-based development. The SAMD21 provides six serial peripheral managers, meaning that the chip can operate six serial interfaces. [25] For this application, SPI is chosen for all intra-board communication due to its high speed and ubiquitous support.

The sensor ADC needs to handle 16 sensor channels. The input range should be 0-5V to accommodate the most common sensor type. Based on past experience with the chip, the TI ADS7953 16-channel ADC is selected. This ADC has a 12-bit measurement resolution and a maximum single-channel sample rate of 1MSPS. It requires a 5V rail and a user-selectable SPI interface supply voltage, as well as a precision 2.5V reference. [26]

The actuator DAC needs to handle 2 proportional valves, which could accept either a 0-5V or a 4-20mA analog signal. Market availability pushes to split this into 2 single-channel DACs. A choice that fits these requirements is the TI DAC7760. This DAC provides a voltage output and a current output, which for this application can be tied together into a single signal (where only one can be allowed to be enabled at a time). It accepts a 24V supply as well as a SPI interface supply voltage, making it convenient to interface between the microcontroller and these DACs without isolation. [27]

The actuator solenoid controllers need to be able to hand 8 solenoids. Each solenoid may draw 500mA or more. Based on these requirements basic searching produced a 4-channel, 1A per channel solenoid driver called the TI DRV8806. This chip operates very simply, accepting just 4-bit control signals where each bit enables or disables a channel. The communication interface is SPI-like so it can be used just like the other SPI devices in the system. Two of these solenoid driver ICs will be used. [28]

The network interface should be readily compatible with the processor, so one used on an Arduino header should be chosen. It should also be capable of up to 100Mbps network speeds. Based on this, the WIZnet W5500 network interface is chosen. This chip implements Ethernet Layer 1 (PHY) and Layer 2 media access control (MAC). It connects between an Ethernet port (with integrated magnetics) and a processor via a SPI interface. [29] It is also used on a popular Arduino network add-on, which helps to speed development by providing a reference design and software.

The power supply consists of a 24V buck, followed by a 5.5V buck which serves to increase the efficiency of the 5V and 3.3V that both draw off its output. The 24V rail may need to supply up to 3A. For this application, the TI TPS54361 buck controller fits the role. It can supply up to 3.5A with efficiency over 90%. [30] For design simplicity, the same chip can be used for the 5.5V regulator. A similar process leads to the selection of the TI TPS795 for the 3.3V and 5V regulators. This chip is a linear regulator with a high power supply rejection ratio (PSRR), meaning that it prevents input noise from polluting the output. [31] The 5V and 3.3V rail will thus be clean, steady supplies for the sensitive analog and digital components in the system.

5.1.2 Avionics Node: Schematic

The schematic capture involved adaptation of reference designs for each of the previously selected ICs, as well as passive component selection. Because of the vastness of this process, only a few notable examples of this will be pointed out. The full schematic is attached to this package.

For the 5V and 3.3V power supplies, low noise is important. Thus, passive LC low-pass filters were added to each of their outputs. Additionally, most power rails were fitted with a 0Ω resistor footing in series, so that current could easily be measured for any of them by depopulating this resistor and inserting an ammeter.

For the network interface, a RJ45 jack was chosen to allow for interface with a standard copper Ethernet cable. The chosen jack was based on a recommendation from WIZnet. It has integrated magnetics which makes design of the output of the network interface relatively simple.

For the DACs, it is recommended to disable the 24V supply until the 3.3V supply is fully enabled. Since the 24V feeds the 3.3V supply, this is never possible. To work around this issue, a controllable FET is placed between the raw 24V supply and the 24V input to the DACs. This FET is enabled when the 3.3V rail is fully enabled, thus meeting the sequencing requirements of the chips.

5.1.3 Avionics Node: Layout

The most notable challenges of layout for the avionics node board were component placement and routing of SPI traces.

Component placement was mostly challenging because of the size of the connectors and because it required constant checking of the positioning of subsystems relative to the microcontroller, to make sure that routing would not become unduly complicated. The result of a lot of tweaking was the allocation of the left side of the board for actuator circuitry operating at 24V. From there, the rest of the circuitry radiated out from the bottom center of the board. The microcontroller was placed there, with the USB and Ethernet interfaces on the right and the sensor interfaces in the middle top. For a 3D rendering of the component placement, see Appendix Section 8.5

Far more challenging than placement was routing of SPI traces. Routing a high-speed signal like SPI requires careful consideration of ground reference. Although SPI is not particularly high speed (technically has a maximum clock frequency of 20MHz, although the WIZnet W5500 calls for 80MHz), the maximum signal frequency seen on the bus is actually a function of only the slew rate of the driver. This means that even a slow bus may have high frequency content if the device creating the digital signal switches between levels very quickly. For these high frequency signals, electromagnetics come into play. In particular, if the signal trace departs from the ground return path significantly, it starts to become an antenna for those high frequency components, and any signal coming close to this signal may get these high frequency components coupled onto it. This can cause interference with other signals. This antenna effect also created an impedance change on the signal in question, which can lead to reflections and oscillations. In general, it is important to keep high-speed traces referenced as tightly

to ground as possible. This is a challenge on a two-layer board, and it is ultimately what kept the project from completing the board layout on time.

5.2 System Demonstration

In order to demonstrate the real-time ability of the proposed system, we constructed a ball beam balance. With a ball beam balance, a beam is attached to a servo and a ball is placed on the beam. This system is naturally open-loop unstable as the ball will role to either side of the beam as opposed to a set point on the beam. The goal of the system is to adjust the tilt of the beam until the ball remains stationary at the set point. The open loop instability shows the real-time nature of the avionics board and the constant vigilance of the controller.

5.2.1 Demo: Hardware

For this demonstration we utilized two Arduino M0s with 2 Wiznet 5500s for the nodes. One of these nodes was attached to two HC-SR04 distance sensors to determine the position of the ball. The second node was attached to an HS-485HB Servo motor adjust the angle of the beam. The two nodes communicate with the master over a TP-Link network switch. The master is a Beagle Bone Black.

5.2.2 Demo: Software

For software a controller was synthesized in MATLAB and then ported over to C and compiled on the BBB. After initializing the node boards and the controller requests the latest sensor readings. In response to those readings the controller would process them set a new desired angle for the demo. The controller then sends this to the nodes with the new data. The model is a standard ball beam system with transfer function:

$$P(s) = \frac{-mg}{\left(\frac{J_{ball}}{R^2} + m\right) s^2}$$

This is a simplified model of a ball-beam system assuming instantaneous angular response and disregarding angular momentum of the beam. The simplified system has an order 2 pole at $s = 0$, which prevents H_∞ control from being used. The position of the ball is determined by sensor fusion of two ping sensors, and LQG control was employed with disturbances in each sensor channel to provide state estimation and control estimates.

The servo motor has an internal closed-loop proportional gain system; it's operation is governed by $\tau = K(u - \theta)$ where $u = \theta_{desired}$ is the user-supplied angle. The true proportional gain K is not published on the servo data sheets, and the motor's value of current angle is also not exposed to the outside. In the future, the full equations of motion for the ball-beam system in terms of $\{r, \dot{r}, \theta, \dot{\theta}\}$ [32] can be coupled with robust control and parameter estimation on the uncertain motor gain K to control the system. It would be interesting to see how the supplied $\theta_{desired}$ changes in terms of leading the servo motor for different values of K.

5.2.3 Demo: Communication

The communications scheme was implemented as described in Section 4 and we observed average round-trip time of 100μs with little optimizations to the network interfaces on these devices. A faster speed could theoretically be achieved with further software optimizations.

6 Cost Analysis

A survey was conducted on products on the market to investigate the cost effectiveness of the custom designed solution. An off the shelf product was found with similar capability for each of the avionic board's features. These products are not guaranteed to work with each other and require significantly more space than one board. The cost is shown in the table below.

Item	Manufacturer	Product	Channels	Cost
DAC	DGH Corp	DG500	2	$295.00
Solenoid Driver	Advantech	ADAM-4168	4	$170.00
ADC	Arduino	Arduino Mega	8	$38.50
Network Interface	Wiznet	5500 Shield	N/A	$23.00
Processor	Arduino	M0	N/A	$20.00
			Total	**$546.50**

Table 1: Breakdown of different expenses

The estimated bill of materials for a single avionics board costs approximately $160 and 5 PCBs are about $50. Our $210 solution is twice as cost effective and more space efficient than the off the shelf combination above.

7 Conclusion

As Hyperloop technology turns from fantasy into reality, it is important to focus on the engineering challenges that lay in the way of its commercial implementation. Air bearing levitation control for a Hyperloop pod can be tackled using a network-connected, real-time collection of a sensors and actuators combined with an advanced non-linear controller. The Hyperloop Control System Capstone project has done important work to produce a dynamics model of this levitation mechanism and a real-time hardware network to implement an eventual controller. Although the timeline of the class did not allow for full realization of a control scheme, this work will continue as Paradigm Hyperloop works to become the fastest hyperloop vehicle in the world. This Capstone project has pushed the work of that team, and of hyperloop as a whole, closer to reality.

8 Appendices

8.1 List of Tables

List of Tables

1 Budget 18
2 Heave Dynamics Model Parameters 20

8.2 List of Figures

List of Figures

1 Hyperloop Alpha Concept Image 3
2 Very High Speed Transit route map 3
3 Paradigm Hyperloop Pod 4
4 Positive Train Control Block Diagram 6
5 OSI Network Model 7
6 Bag Model 8
7 Open Loop Lyapunov Function in Atmosphere 10
8 Linearization of Heave Dynamics Model at Atmospheric Pressure . . . 21
9 Simulation of Pod Drop from 1cm 22
10 Simulation of Pod Liftoff 23
11 Linearization of Heave Dynamics Model at Tube Pressure 24
12 Simulation of Pod in Vacuum 25
13 Pole-Zero Map of System Transfer Function with Disturbances 26
14 Profile of Track at 60mph 26
15 Profile of Track at 400mph 27
16 Simulation of Pod Motion under Disturbances at 60 mph 27
17 Simulation of Pod Motion under Disturbances at 400 mph 28
18 Simulation of Pod Motion under Disturbances at 700 mph 28
19 Avionics Node Board 3D Render 29

8.3 Parameters of Heave Dynamics Model

Variable	Quantity	Value	Units
L	Skate Length	0.9144	m
W	Skate Width	0.3048	m
N	Number of Skates	4	1
A	Total Area	1.1148	m^2
Per	Total Perimeter	9.7536	m
M	Pod Mass	909	kg
g	Gravitational Acceleration	9.81	$\frac{m}{s^2}$
$\dot{m}^*_{in}$	Input Mass Flow	0.16	$\frac{kg}{s}$
z^*	Ride Height	8.28	mm
ℓ	Bag Thickness	0.002	m
κ	Bag Permeability	1e-5	m^2
β	Forchheimer Coefficient	1.2e6	m^{-1}
μ	Air Viscosity	1.85e-5	$Pa * s$
T	Air Temperature	300	K
R	Specific gas constant of air	287	$\frac{J}{kg*K}$
γ	Air specific heat ratio	1.4	1
P_{atmo}	Atmospheric Pressure	101.325	kPa
ΔP^*_{bag}	Gauge Pressure of Bag	8	kPa
P_{tube}	Tube Pressure	0.666	kPa
ρ_{atmo}	Atmospheric Density	1.1768	$\frac{kg}{m^3}$

Table 2: Environmental and System parameters of the Heave Dynamics Model

8.4 Outputs of Heave Dynamics Model

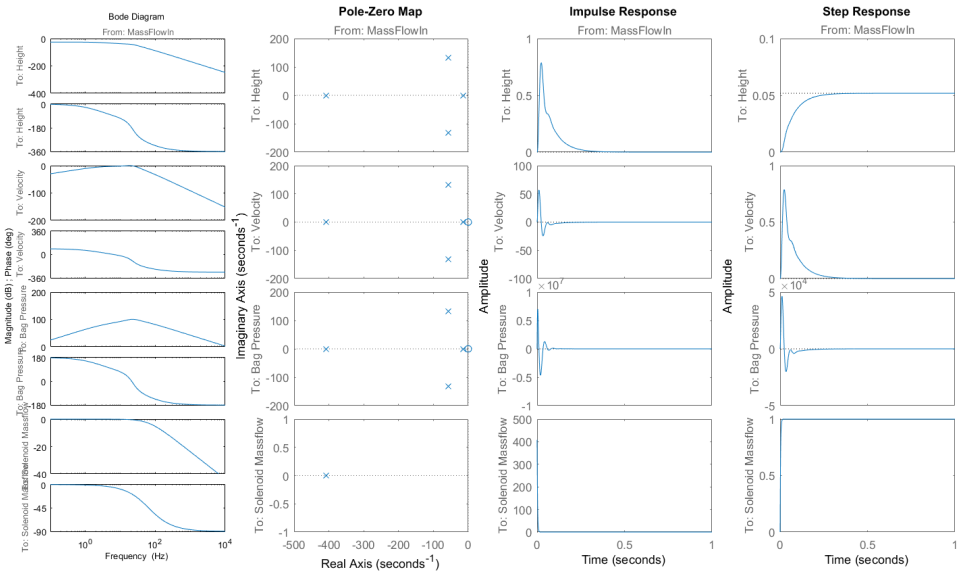

Figure 8: Linearization of heave dynamics in atmospheric pressure. Results in stable equilibrium. Pole at -14.8 is from non-Darcy flow, at -408 is from actuator critical frequency, and $-57.9 \pm 133j$. In linear Darcy flow or when the quadratic Forchheimer penalty is low, the non-Darcy flow slides to the left. When this pole $Re(s) \leq -57.9$, the oscillatory heave poles now dominate dynamics, and the pod's behavior is to have decaying oscillations at 21 Hz. States are height, change in height, bag pressure, and actual mass flow in.

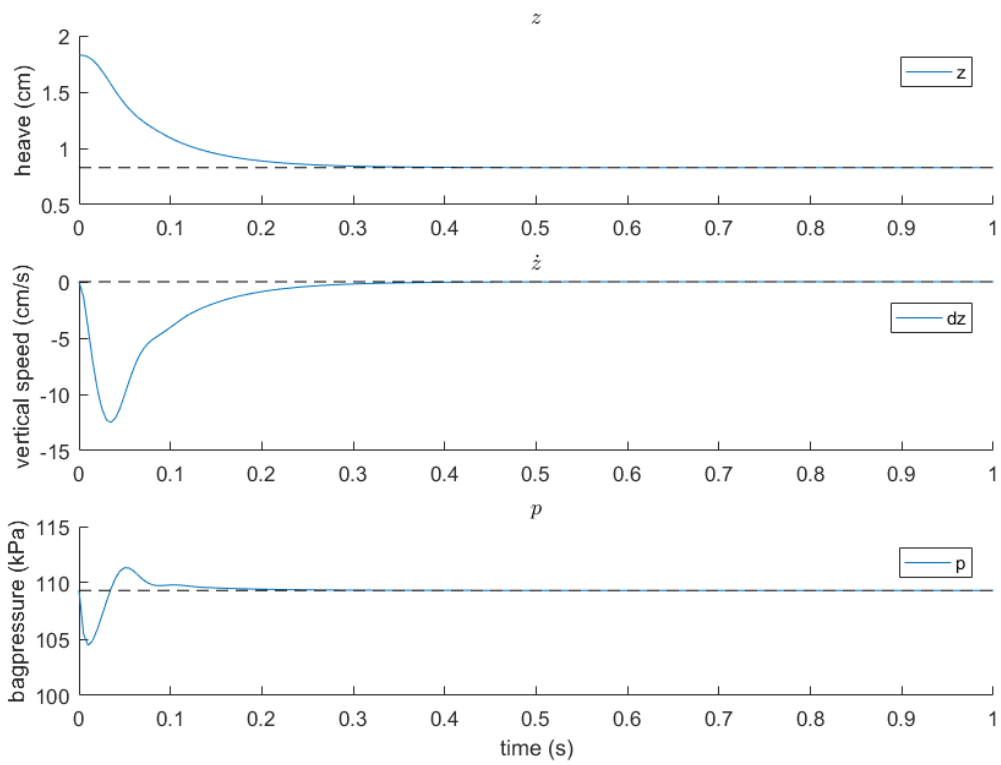

Figure 9: Pod dropping from 1cm above equilibrium point. Volume is originally higher than equilibrium, so the pressure drops. At a certain critical point, the pressure reaches a minimum and starts to rise, slowing down the descent of the pod.

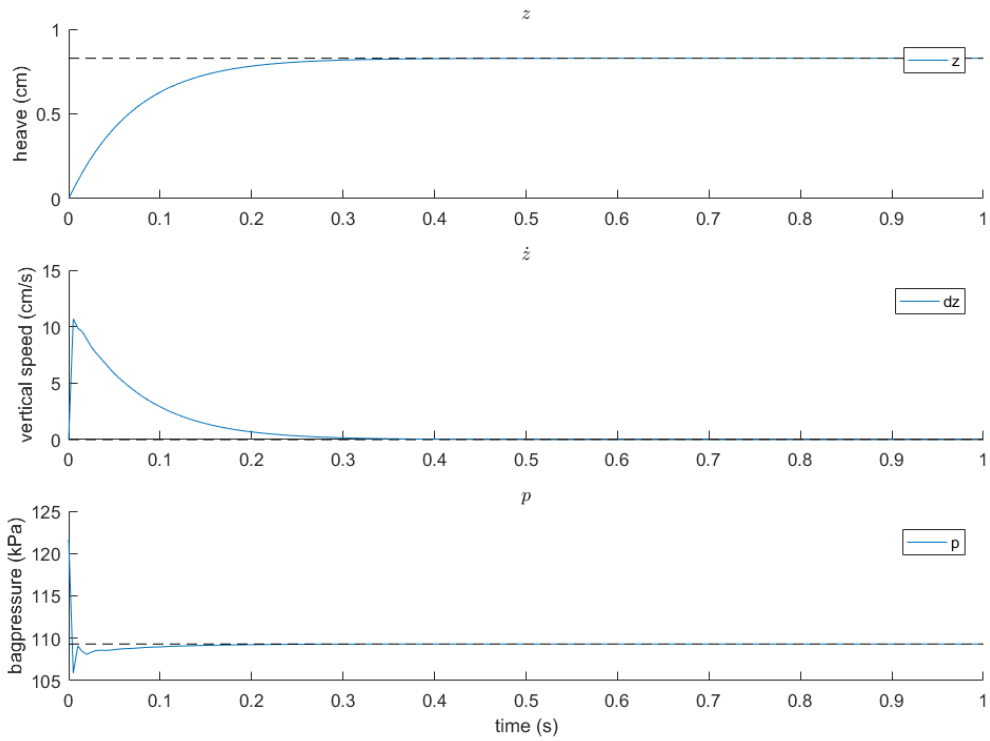

Figure 10: Pod lifting off from the ground (actually 1e-10 m). The extraordinarily small volume coupled with a constant incoming mass flow rate results in a massive pressure spike at the beginning, which pushes the pod up. The velocity then starts to decrease, and gently pushes the pod to equilibrium. The time constant and exponential behavior is validated with respect to video footage of the pod.

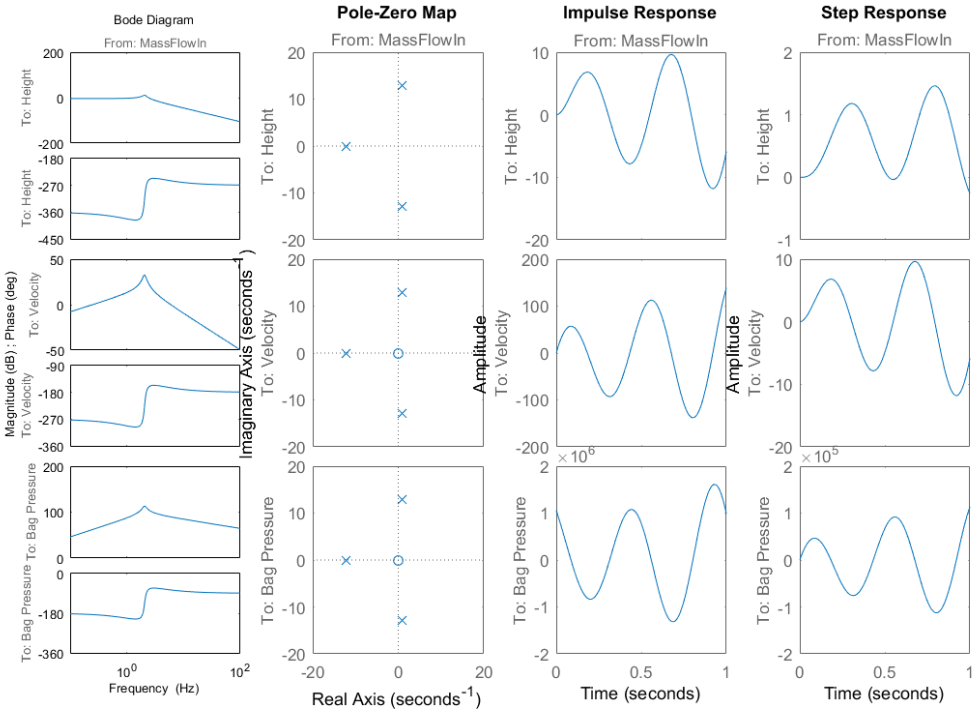

Figure 11: Linearization of the pod in tube pressure (1-5 torr) results in an unstable system, with the two heave oscillatory poles (complex conjugate) pushed beyond the $j\omega$-axis. The model predicts open loop instability in tube pressure. Vacuum testing of the pod has shown pronounced vibrations, which certify a tendency towards instability.

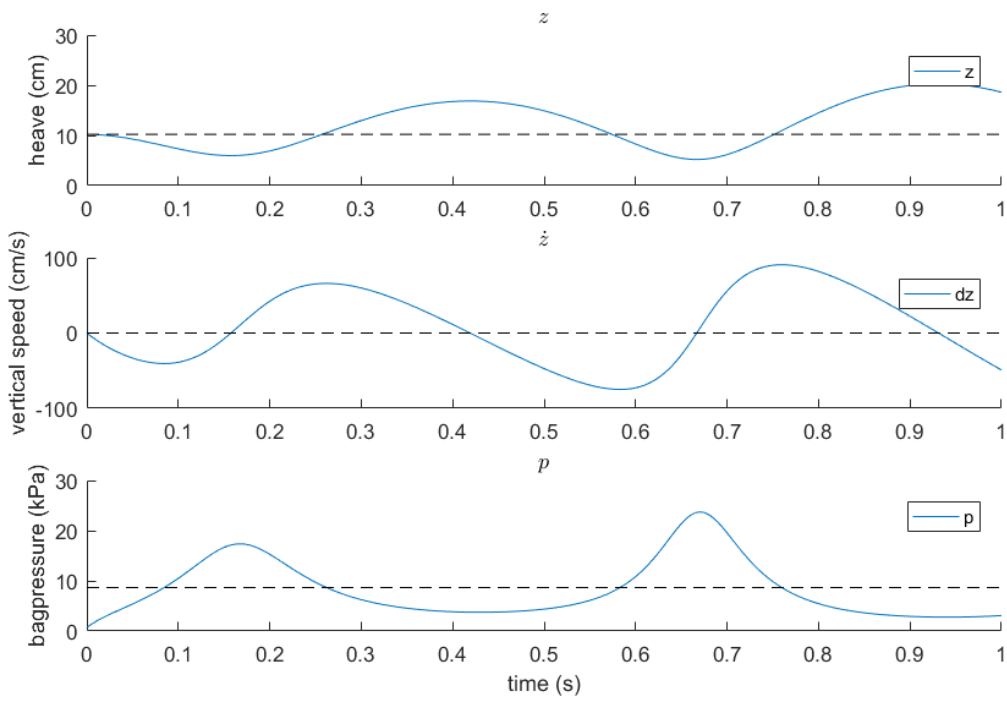

Figure 12: Simulation of the open-loop unstable heave dynamics of the pod at tube pressure (5 torr = 0.666 kPa).

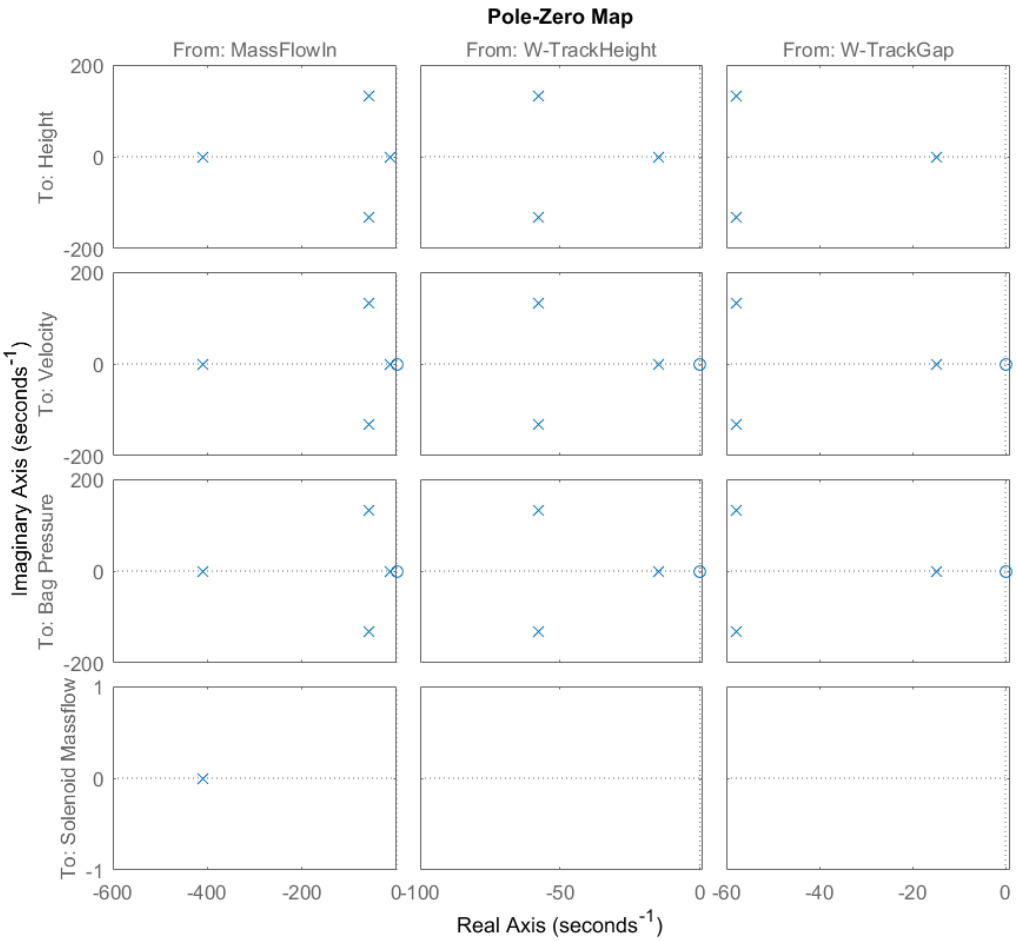

Figure 13: Pole zero map of linearization of the disturbance-affected heave behavior. All poles and zeros in the disturbed transfer function are identical to the applied mass flow, except for the $s = -408$ pole from the actuator's lowpass filter. This shows stability with respect to the disturbances. Unfortunately, the disturbances have a larger gain than the original system, and H_∞ norms can be in the range of 2e6.

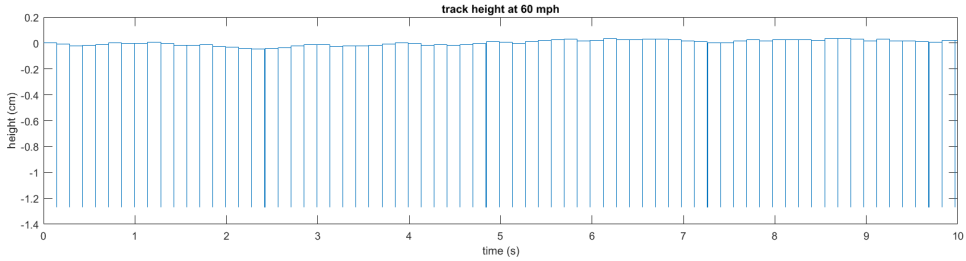

Figure 14: Example profile of the track when the pod is traveling at 60mph. Gaps and height drift are clearly shown.

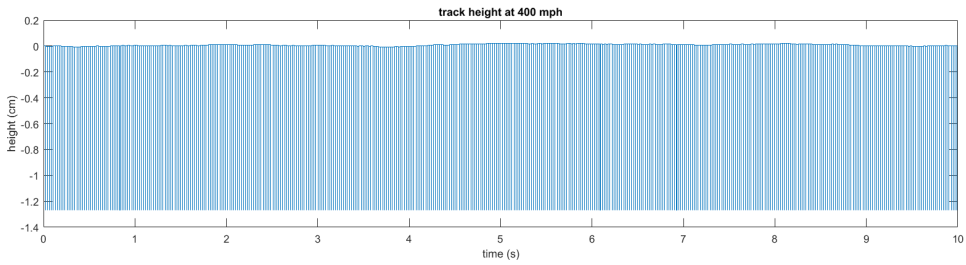

Figure 15: Example profile of the track when the pod is traveling at 400mph. Pod traverses more ground and encounters more height changes and gaps.

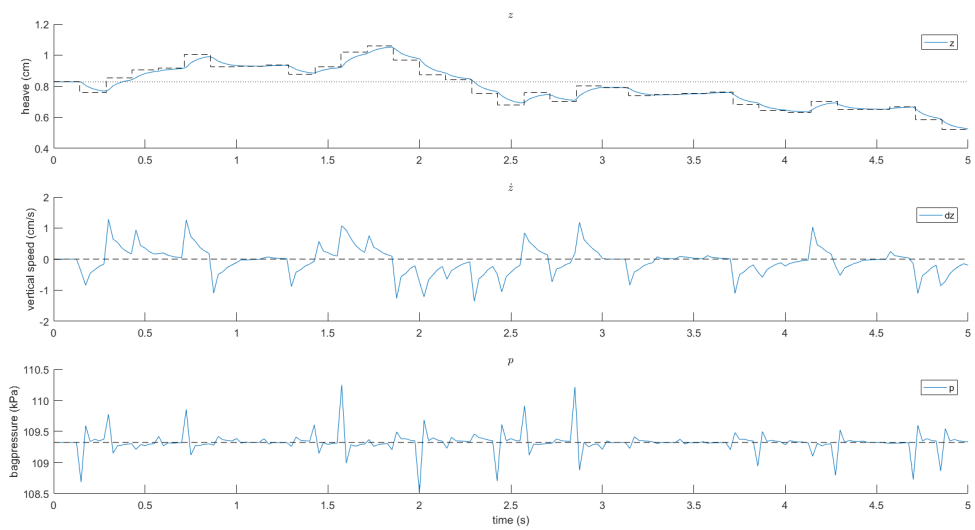

Figure 16: Simulation of heave dynamics when the pod is traveling at 60mph. Step response to w_{height} is fast enough that the skate catches each height change, resulting in a bumpy ride.

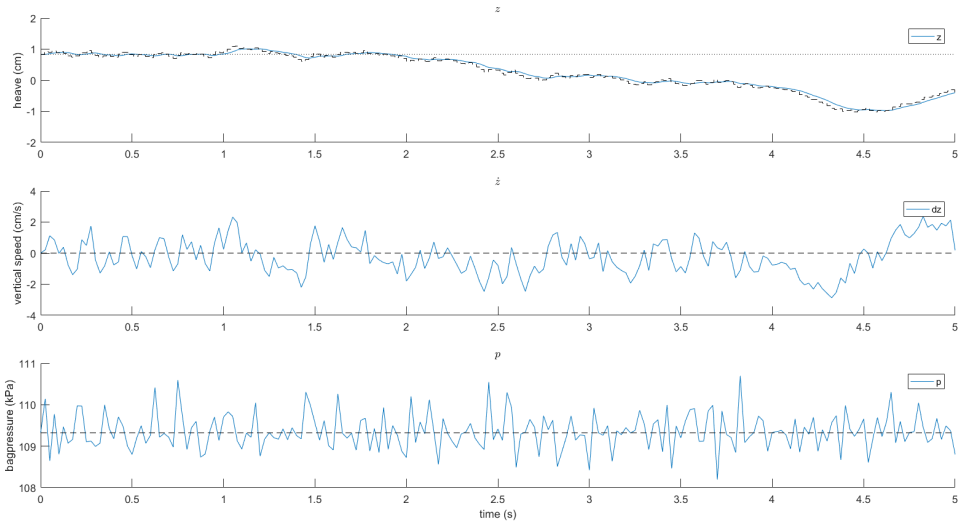

Figure 17: Simulation of heave dynamics when the pod is traveling at 400mph. System is smoothing out step height changes. Similar to driving a car quickly over a series of bumps leads to a gentle ride.

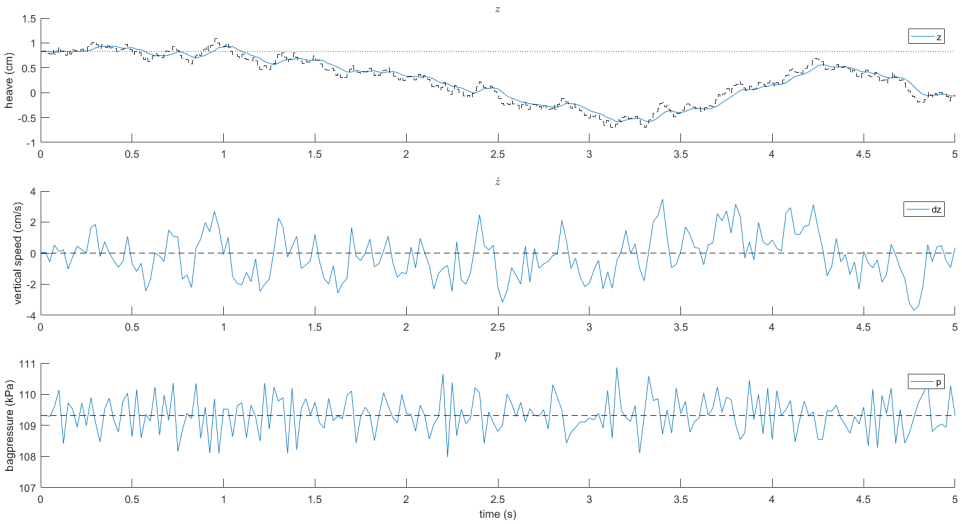

Figure 18: Simulation of heave dynamics when the pod is traveling at 700mph. There is a noticeable lag between step height changes and response. There is a lowpass filtering effect inherent in the system rejecting the raised frequency of step height changes. Gaps continue to not effect the pod in any substantive way.

8.5 Avionics Node Board 3D Render

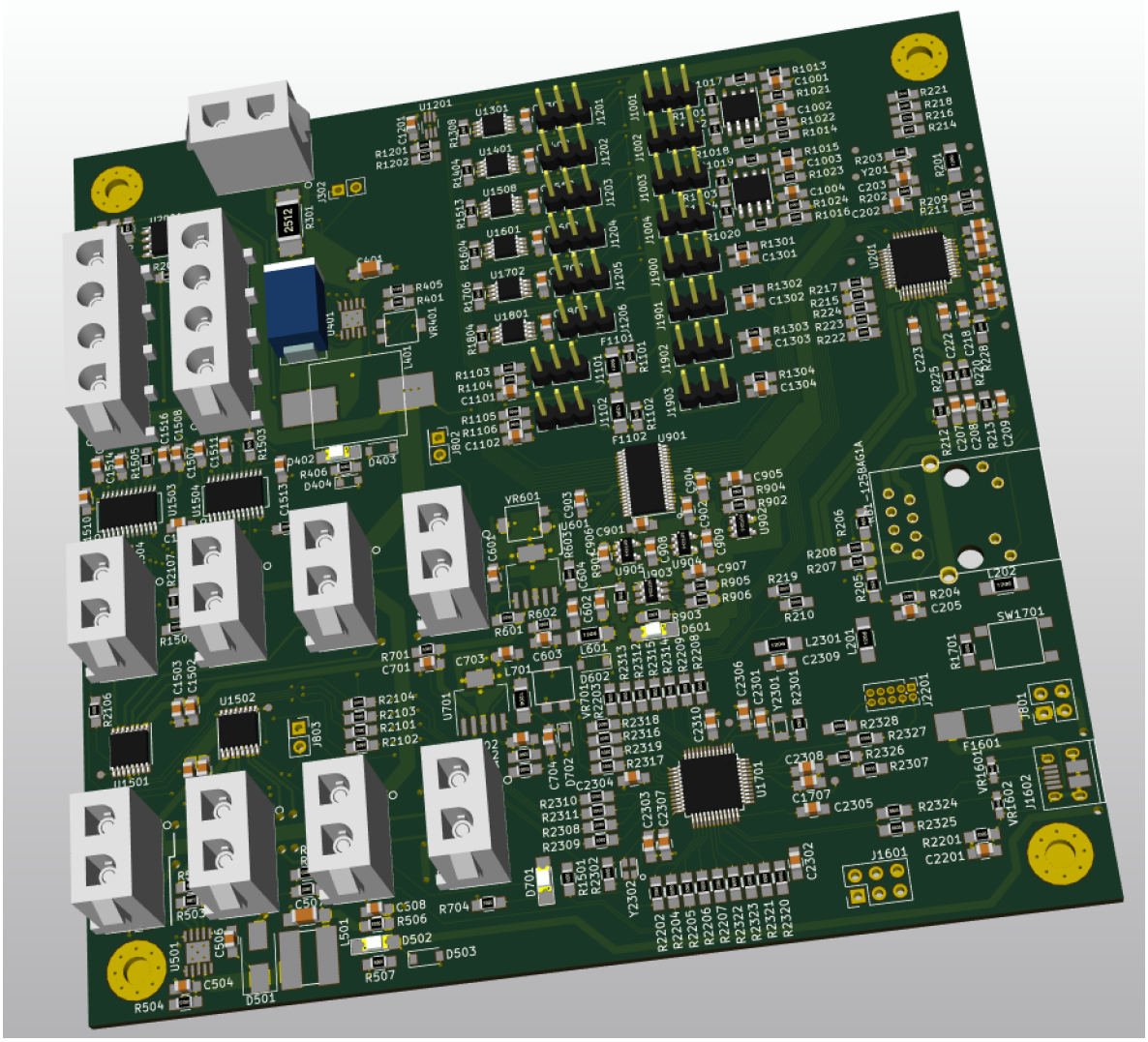

Figure 19: 3D rendering of the avionics node board. Some components are not included in the render, but their placement is clear.

9 References

[1] E. Musk, "Hyperloop alpha," Tesla Motors, Tech. Rep., 8 2013.

[2] J.-P. Rodrigue and T. Notteboom, "Transportation and economic development," 2017. [Online]. Available: https://people.hofstra.edu/geotrans/eng/ch7en/conc7en/ch7c1en.html

[3] E. Goddard, "Vacuum tube transportation system," Jun. 20 1950, uS Patent 2,511,979. [Online]. Available: https://www.google.com/patents/US2511979

[4] R. M. Salter, "The very high speed transit system," 1972. [Online]. Available: https://www.rand.org/content/dam/rand/pubs/papers/2008/P4874.pdf

[5] O. O. O. N. Gordon Ononiwu, Ondoma Onojo, "Quadcopter design for payload delivery," *Journal of Computer and Communications*, vol. 4, no. 10, pp. 1–12, 8 2016.

[6] L. Yun, *Theory & Design of Air Cushion Craft.* Burlington: Elsevier, 2000.

[7] L. Systems, "Scalable and reliable communications infrastructure for positive train control (ptc)." [Online]. Available: https://www.cisco.com/c/dam/en/us/solutions/collateral/industry-solutions/positive-train-so.pdf

[8] A. of American Railroads. (2017) Positive train control. [Online]. Available: https://www.aar.org/policy/positive-train-control

[9] L. Meyer, "Optical sensors improve railway safety," 2013. [Online]. Available: http://www.osa.org/en-us/about_osa/newsroom/news_releases/2013/optical_sensors_improve_railway_safety/

[10] A. Chawla, "Control systems in automobiles." [Online]. Available: http://web.iitd.ac.in/~achawla/public_html/736/18-Control%20Systems%20in%20Automobiles_v3.pdf

[11] R. B. GmbH, "Can specification version 2.0," Bosch, Tech. Rep., 1991. [Online]. Available: http://www.bosch-semiconductors.de/media/ubk_semiconductors/pdf_1/canliteratur/can2spec.pdf

[12] L. F. I, "What's the difference between the osi seven-layer network model and tcp/ip?" 2013. [Online]. Available: http://www.electronicdesign.com/what-s-difference-between/what-s-difference-between-osi-seven-layer-network-model-and-tcpip

[13] R. Hitchcock, "Osi reference model: Layer 1 hardware," 2008. [Online]. Available: http://techgenix.com/osi-reference-model-layer1-hardware/

[14] A. S. Sowayan and K. A. AlSaif, "Investigation of the heave dynamics of air cushion vehicles (acv): parametric and chaotic studies," *Latin American Journal of Solids and Structures*, vol. 10, no. 4, pp. 725–745, 2013.

[15] V. A. Jambhekar, "Forchheimer porous-media flow models-numerical investigation and comparison with experimental data."

[16] "Spacex hyperloop test-track specification," Aug 2016. [Online]. Available: https://cdn.atraining.ru/docs/TubeSpecs.pdf

[17] E. D. Sontag, "Input to state stability: Basic concepts and results," in *Nonlinear and optimal control theory*. Springer, 2008, pp. 163–220.

[18] T. Witelski, D. Schwendeman, and P. Evans, "Analysis of pressurized porous air bearings," 2005.

[19] G. Stachowiak and A. Batchelor, *Engineering Tribology*, ser. Tribology series. Elsevier Science, 1993. [Online]. Available: https://books.google.com/books?id=yzxaPE1dwQkC

[20] A. Majumdar, A. A. Ahmadi, and R. Tedrake, "Control design along trajectories with sums of squares programming," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 4054–4061.

[21] I. Sommerville, *Centralized Control*, 9th ed. Boston, MA: Pearson, 2008.

[22] N. Lomas, "Mit creates a control algorithm for drone swarms," 2016. [Online]. Available: https://techcrunch.com/2016/04/22/mit-creates-a-control-algorithm-for-drone-swarms/

[23] C. Spurgeon, *Ethernet : the definitive guide*. Cambridge, MA: O'Reilly, 2000.

[24] "Arduino m0." [Online]. Available: https://store.arduino.cc/usa/arduino-m0

[25] "Sam d21e / sam d21g /sam d21j: Smart arm-based microcontroller datasheet summary," 2016. [Online]. Available: http://www.atmel.com/Images/Atmel-42181-SAM-D21_Summary.pdf

[26] "Ads79xx 12/10/8-bit, 1 msps, 16/12/8/4-channel, single-ended, micropower, serial interface adcs," 2015. [Online]. Available: http://www.ti.com/lit/ds/symlink/ads7953.pdf

[27] "Dacx760single-channel, 12- and16-bit programmablecurrent andvoltageoutput digital-to-analog converters for 4-ma to 20-ma current loop applications," 2016. [Online]. Available: http://www.ti.com/lit/ds/symlink/dac7760.pdf

[28] "Drv8806 quad serial interface low-side driver ic," 2015. [Online]. Available: http://www.ti.com/lit/ds/symlink/drv8806.pdf

[29] "W5500 datasheet," 2013. [Online]. Available: http://wizwiki.net/wiki/lib/exe/fetch.php?media=products:w5500:w5500_ds_v106e_141230.pdf

[30] "Tps54361 4.5-v to 60-v input, 3.5-a, step-down dc-dc converter with soft-start and eco-mode," 2017. [Online]. Available: http://www.ti.com/lit/ds/symlink/tps54361.pdf

[31] "Tps795 ultralow-noise, high-psrr, fast, rf, 500-ma low-dropout linear regulators," 2015. [Online]. Available: http://www.ti.com/lit/ds/symlink/tps795.pdf

[32] C. G. Bolívar-Vincenty and G. Beauchamp-Báez, "Modelling the ball-and-beam system from newtonian mechanics and from lagrange methods."