Cornell Hyperloop

Magnetic Levitation Control Challenge

Author: Nikita Dolgopolov

Outline

- System Goals & Primary System Requirements
- Secondary System Requirements
- Test Cases
- Approach
- Current Design
- Current Results
- Next steps

Levitation Controls

System Goals

System Goals & Primary Requirements

- Provide pod stability against roll, pitch, and vertical displacement
- Achieve and maintain a reference levitation height
- Reject external disturbances on the vehicle
- Only source of data - readings of 4 distance sensors (4 levitation gaps)

Levitation Controls

Secondary System Requirements

Secondary System Requirements

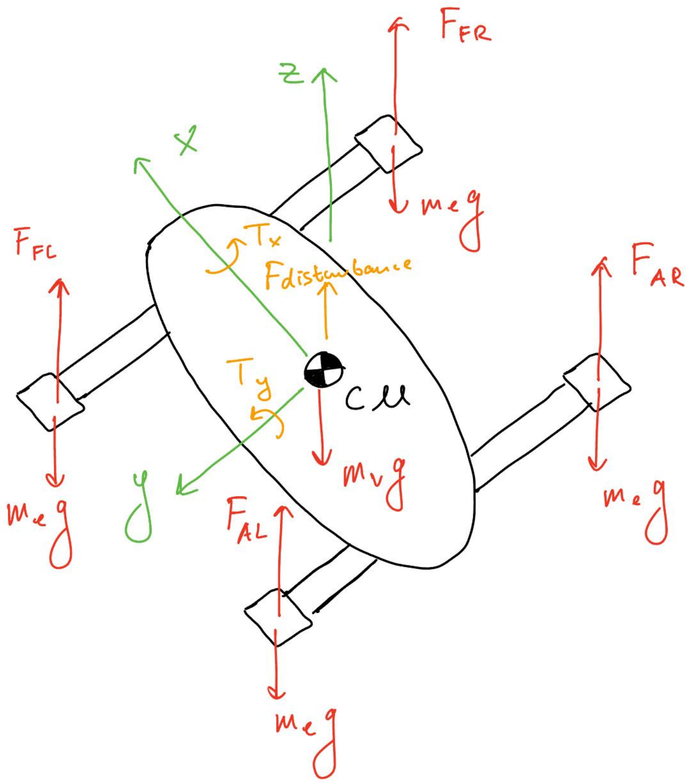
- Deal with vibration caused by bending of magnet support arms
- Deal with bending of the rail
- Account for control current range limitations
- Low signal response time (targeting 0.2ms for real-life limitations)

Levitation Controls

Test Cases

Test Cases

1. Maintaining levitation height
2. Transitioning between levitation heights
3. Constant force applied at CM in -z direction
4. Sinusoidal force applied at CM in -z direction
5. Constant torque applied at CM about x- or y-axis
6. Sinusoidal torque applied at CM about x- or y-axis
7. Sensor noise
8. Mismatch between actual vehicle mass and ideal (expected) vehicle mass
9. Combination of the load cases



Levitation Controls

Approach

Design Approach

1. Develop a 1-DOF magnet controls and test
2. Develop full-pod kinematics equations
3. Implement PID controls system without bending and disturbances
4. Implement PID controls with disturbances
5. Implement PID controls with disturbances and bending/vibrations

Each Stage is simulated with relevant load cases using MATLAB

The controls are tuned via iterative approach and use of Bode Plots

Levitation Controls

Current Design

Current Design

- Step from 0.02m to 0.01m
- Simulation time step: 0.2ms
- Simulated over 30s, 3min, 30min
- Determined that scaling proportional error via E_pow can boost performance
- Functionality for current limits implemented
- Assume no bending of the magnet arms

Kp = 50
Ki = 20
Kd = 2
E_pow = 0.9

```
FFR(i+1) = mg_eff*(1 + Kp*(g_ref(i)-gFR_t(i))^(e_pow)+Kd*(0-dgFR_dt(i)) + Ki*accFR(i+1));
```

Signal Flow Path Diagram - single-body system

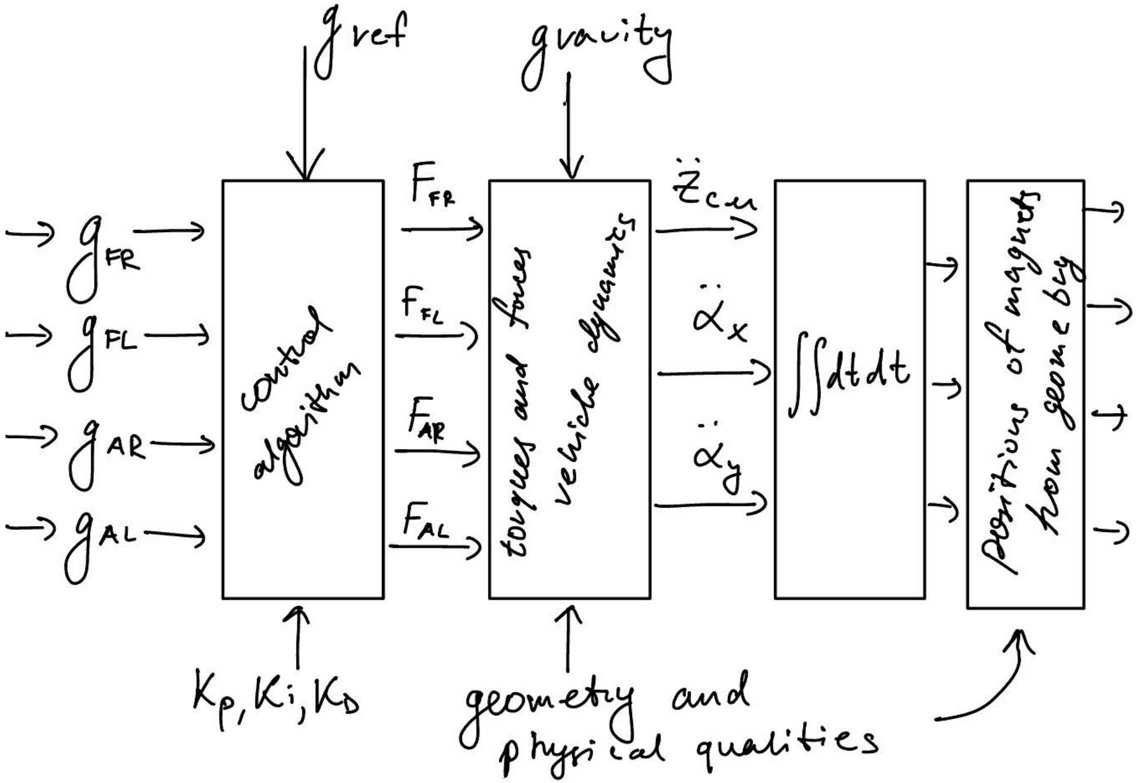
Control forces are based on gaps

Forces cause translation and rotation

Integrate to get current state

Magnet positions are calculated from the vehicle geometry

In multi-body system, the magnet forces are exerted on the magnets and the rail, while the beam bending forces are exerted on the vehicle itself



Electromagnet kinematic model

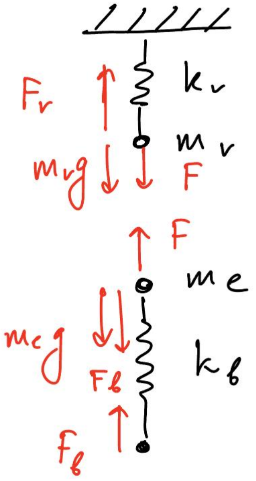
Mass-spring system on each corner

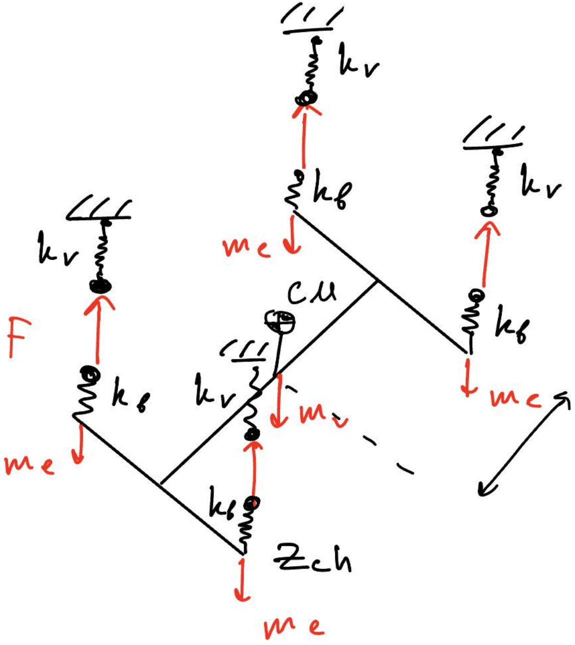
$$m_e \ddot{z}_e = -m_e g - F_b + F$$

$$m_v \ddot{z}_v = -m_v g - F$$

$$\ddot{g}_{FR} = \ddot{z}_v - \ddot{z}_e + \ddot{z}_{ci}$$

$$g_{FR} = g_{FR_0} + (z_r - z_e)$$



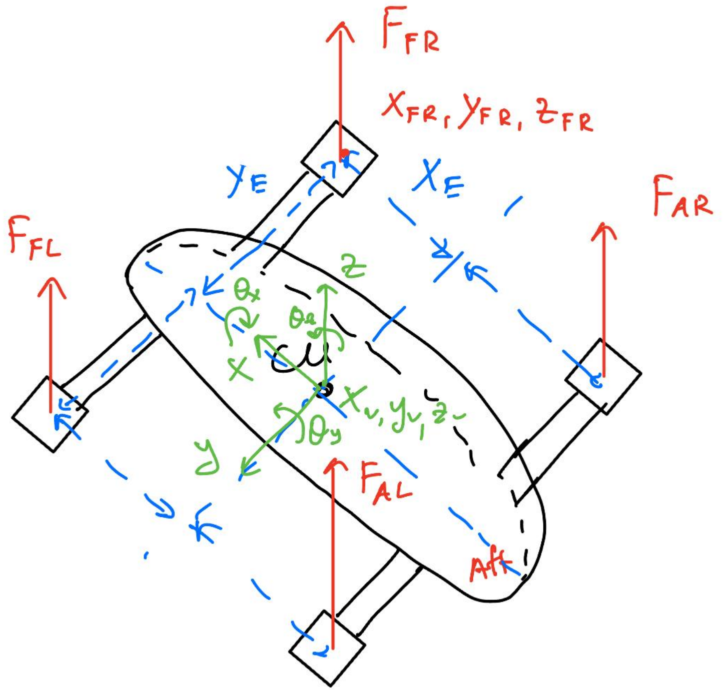


Magnet Kinematic Model

Center of Mass Kinematics

The forces and torques on the body are the result of spring forces

In the case of no springs, electromagnet forces are applied to the pod directly



Pod Kinematic Model

Center of Mass Kinematics

```
%%%  POD KINEMATICS  %%%

%CM kinematics rotation, only external forces matter

ddalphaX_ddt(i+1) = 1/I_vx *(-k_b*z_bFR_t(i)*y_E + k_b*z_bFL_t(i) * y_E - k_b*z_bAR_t(i) * y_E + k_b*z_bAL_t(i) * y_E);
dalphaX_dt(i+1) = dalphaX_dt(i) + ddalphaX_ddt (i+1) * dt;
alphaX_t(i+1) = alphaX_t(i) + dalphaX_dt (i+1) * dt;

ddalphaY_ddt(i+1) = 1/I_vy *(-k_b*z_bFR_t(i)*x_E - k_b*z_bFL_t(i) * x_E + k_b*z_bAR_t(i) * x_E + k_b*z_bAL_t(i) * x_E);
dalphaY_dt(i+1) = dalphaY_dt(i) + ddalphaY_ddt (i+1) * dt;
alphaY_t(i+1) = alphaY_t(i) + dalphaY_dt (i+1) * dt;

%CM kinematics translation, only external forces matter

ddzCM_ddt(i+1) = (-g*m_v + k_b*z_bFR_t(i) + k_b*z_bFL_t(i) + k_b*z_bAR_t(i) + k_b*z_bAL_t(i))/(m_v);
dzCM_dt(i+1) = dzCM_dt(i) + ddzCM_ddt(i+1) * dt;
zCM_t(i+1) = zCM_t(i) + dzCM_dt(i+1) * dt;
```

Assume vertical forces

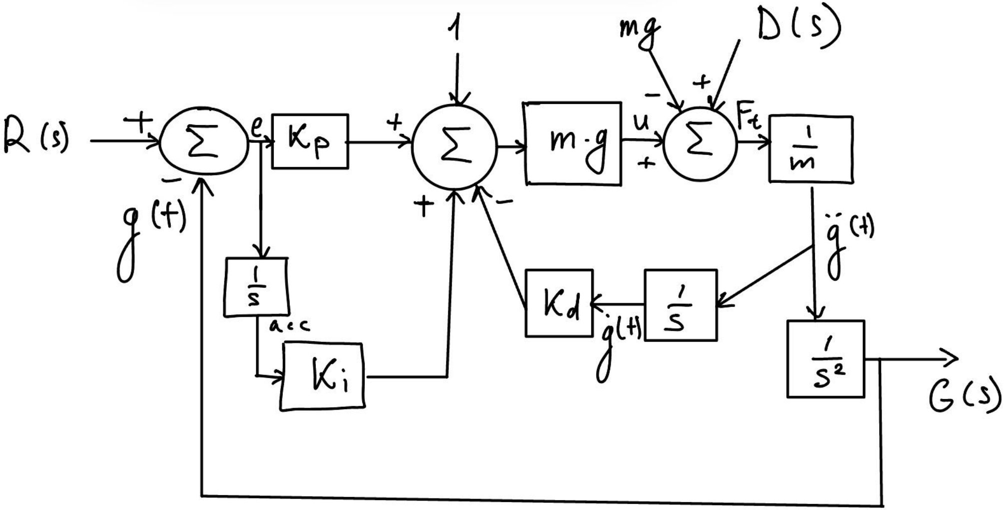
Ignore yaw, horizontal translation

Non-deflected Beam Positions

```
%%% CHASSIS KINEMATICS %%%

%rotation matrices to get position directly
Rx = [1 0 0; 0 cos(alphaX_t(i+1)) -sin(alphaX_t(i+1)); 0 sin(alphaX_t(i+1)) cos(alphaY_t(i+1))];
Ry = [cos(alphaY_t(i+1)) 0 sin(alphaY_t(i+1)); 0 1 0; -sin(alphaY_t(i+1)) 0 cos(alphaY_t(i+1))];
Rz = [cos(alphaZ_t(i+1)) -sin(alphaZ_t(i+1)) 0; sin(alphaZ_t(i+1)) cos(alphaZ_t(i+1)) 0; 0 0 1];

c_curr_pos = Rx*Ry*Rz*[x_E x_E -x_E -x_E; -y_E y_E -y_E y_E; -z_offset -z_offset -z_offset -z_offset];

z_cFR_t = c_curr_pos(3, 1);
z_cFL_t = c_curr_pos(3, 2);
z_cAR_t = c_curr_pos(3, 3);
z_cAL_t = c_curr_pos(3, 4);
```

Using rotation matrices to obtain
non-deflected beam positions

1-DOF Controls Algorithm



This model does not include pitch, roll, z of full system

$$\left(\left(\left(R(s) - G(s)\right)\left(K_p + \frac{K_i}{s}\right) + 1 - G(s)\cdot s \cdot K_d\right)\cdot mg + D(s) - mg\right)\cdot\frac{1}{m} = G(s)\cdot s^2$$

$$D(s) = m s^2 G(s) + s K_d G(s) mg - \left(K_p + \frac{K_i}{s}\right)\left(R(s) - G(s)\right) mg$$

set to 0

$$\frac{G(s)}{D(s)} = \frac{1}{ms^2 + mg\left(K_p + sK_d + \frac{K_i}{s}\right)}$$

Levitation Controls

Current Results

Current Results

Single-body system (electromagnets and beams are fixed rigidly to body)

- Passing all individual test cases
- Passing combined test cases, but not exhaustive testing so far

Multi-body system (electromagnets and are connected by springs)

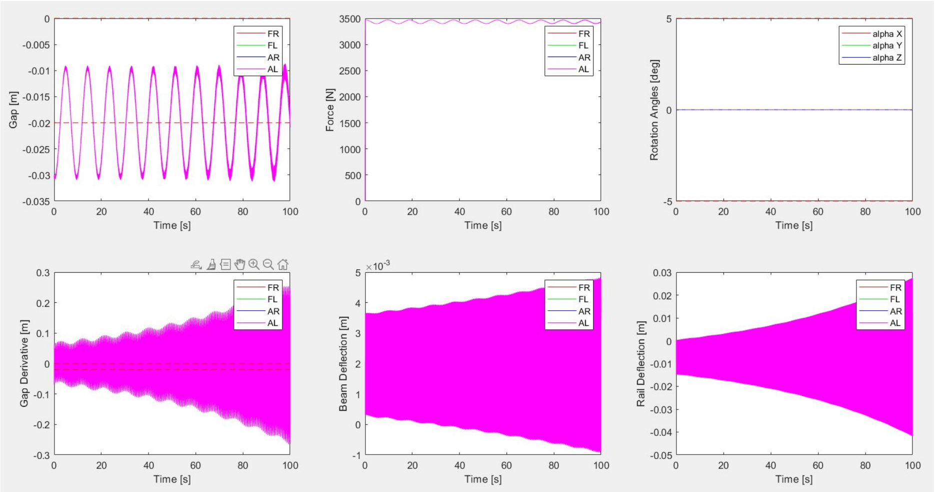
- Failing test cases
- Current task is to confirm kinematic model
- Explore various control methods like lead compensation

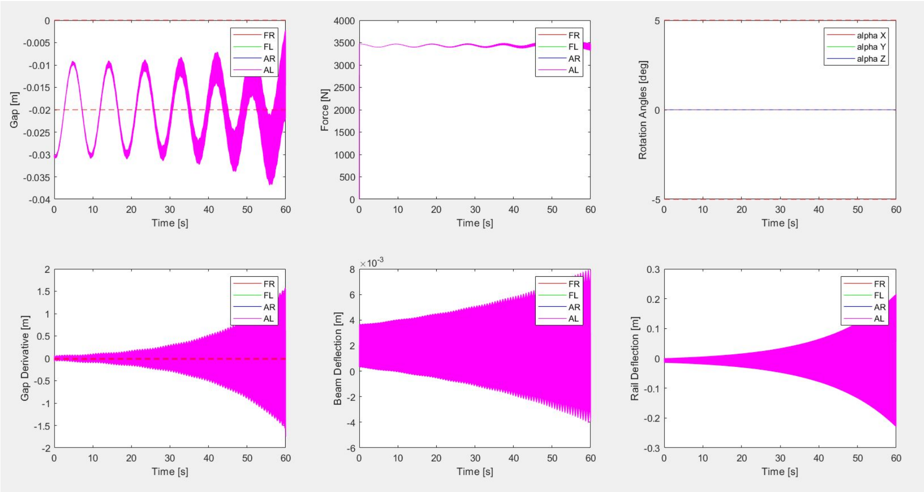
Failing System - multi-body

Vibration rises steadily

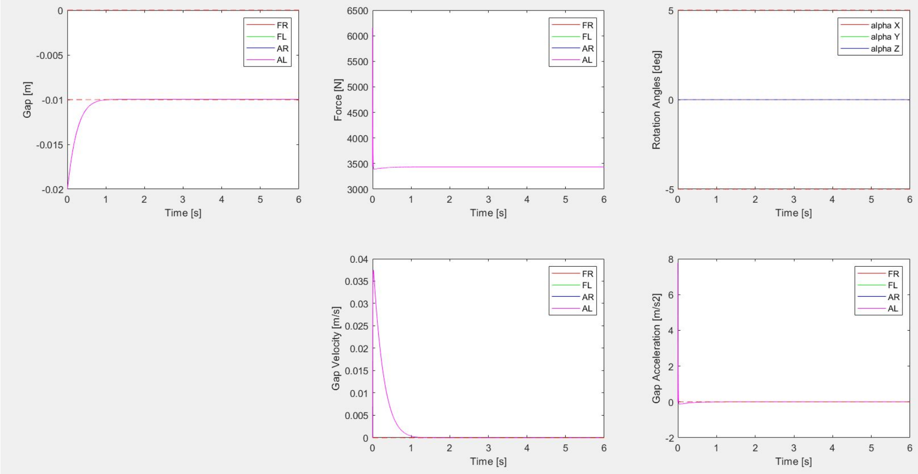
Derivative feedback does not help - could be dumping energy in to the system?

Incorrect kinematics?



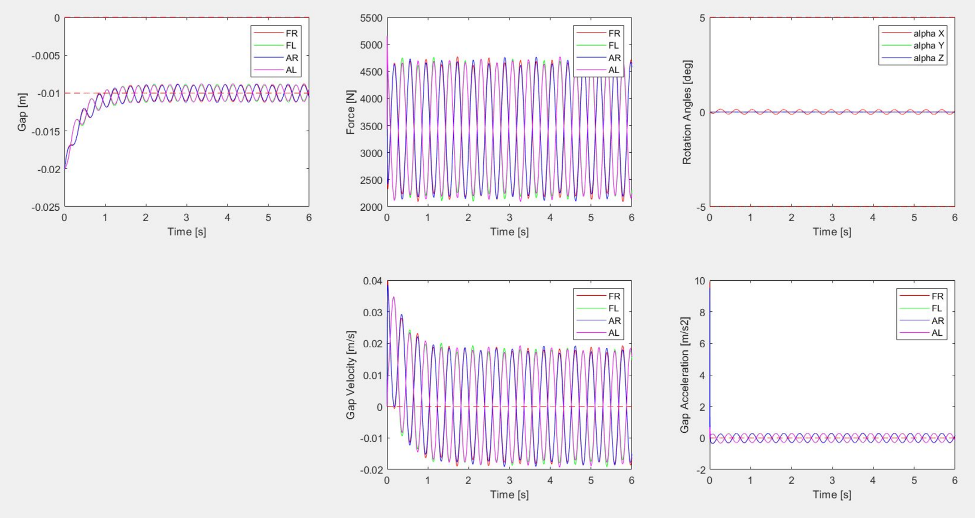


Working System - single-body



Response to a step

Response to a mixed load case:



Levitation Controls

Next Steps

Next Steps

0. Currently have a working 4-DOF single-body model that works

1. Confirm kinematics of the multibody model and explore more control methods
2. Ensure the stability of the final model in basic test cases
3. Consider switching back to a 1-DOF system for faster iteration
4. After 1-DOF system works, will implement it in 4-DOF

Questions?