Final Report

ME 321 Mechanical Engineering Analysis for Design

Duke University

Spring 2025

Group 15

Ethan Fernandez

Pierre Gathy

Connor Mitchell

May 1, 2026

Contents

1 Introduction

Following recent flood events, PumpCo has commissioned our team to design a structure to support a pump system. The pump will be mounted on a rigid, adjustable platform connected to an existing frame via clevis pins. This structure must withstand the torsional forces from a belt-driven pulley and remain stable under operation.

The system will be operating near bodies of water, requiring materials and construction suited for wet environments. Our design approach includes symbolic and finite element analysis and iterative optimization to ensure compliance with all functional and structural requirements.

- Support a 12.5 lb centrifugal pump operating at 150 GPM and 60 ft head
- Pump driven at 1800 RPM with 200N belt tension
- Must use at least two support arms and clevis pin connections to frame
- Deflection at platform tip must be ≤ 1 mm
- Maintain a factor of safety ≥ 1.5 for yield and buckling
- Entire structure must fit within a 24" height limit
- Raw material cost must not exceed $150
- Must resist corrosion (no wood or degradable plastics)

The sections that follow outline our analysis, design process, simulation validation, and prototype evaluation.

2 Equilibrium Calculations (Symbolic)

2.1 Free Body Diagrams

Shown are the free body diagrams for the scope-basic version of the design. Because this is mainly meant to consider force distribution, the actual geometry uses a base concept and does not match the final design. FBDs are in Figs. 1, 2, and 3.

2.2 Power and Torque Calculations

Define all constants used for the Fluid Power as follows:

$$\text{Water Density, } \rho = 1000\,\mathrm{kg\,m^{-3}}$$
$$\text{Volume Flow Rate, } Q = \frac{150 \times 0.00379}{60}\,\mathrm{m^3\,s^{-1}}$$
$$\text{Gravity, } g = 9.81\,\mathrm{m\,s^{-2}}$$
$$\text{Pump Height, } h = 60 \times 0.3048\,\mathrm{m}$$

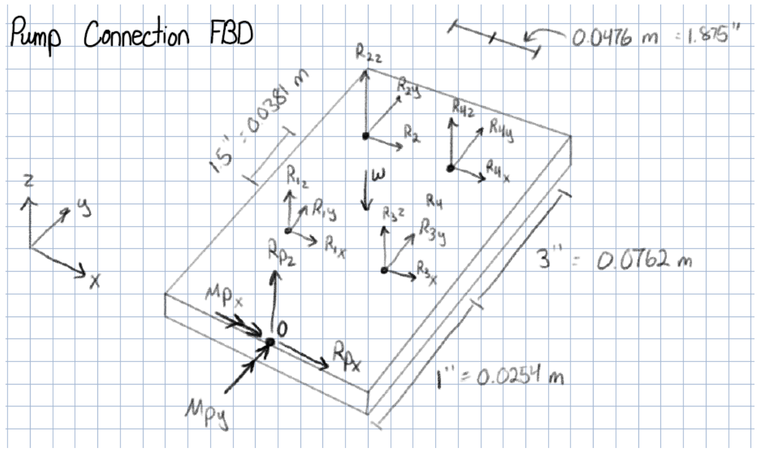

Figure 1: Pump Connection Platform Free Body Diagram

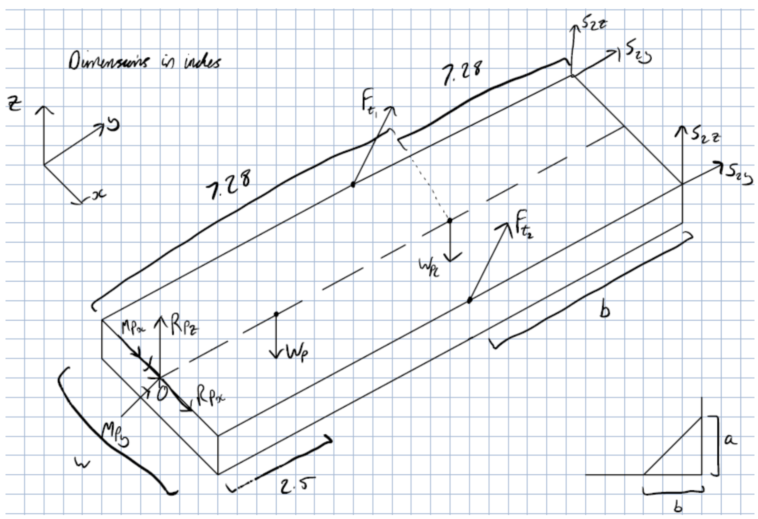

Figure 2: Platform Free Body Diagram

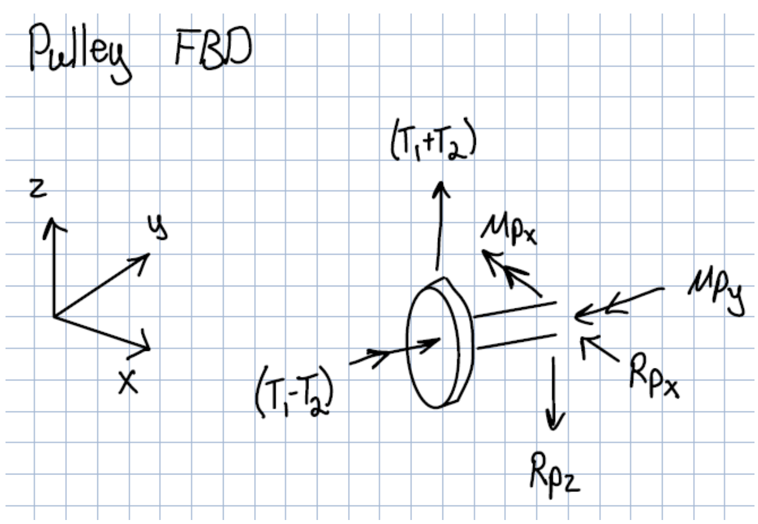

Figure 3: Pulley Free Body Diagram

The Fluid Power equation can be applied

$$P_f = \rho \cdot Q \cdot g \cdot h$$

and substituted

$$P_f = 1000 \cdot \left(\frac{150 \cdot 0.00379}{60} \right) \cdot 9.81 \cdot (60 \cdot 0.3048)$$

To determine the final result:

$$P_f \approx 1697.92\,\mathrm{W} \approx 2.276\,\mathrm{HP}$$

Accounting for efficiency, the required mechanical power can be determined:

Given

$$\text{Efficiency, } \eta = 0.65$$

$$P_{\text{mech}} = \frac{P_f}{\eta}$$

$$P_{\text{mech}} = \frac{1697.92}{0.65}$$

$$P_{\text{mech}} \approx 2612.19\,\mathrm{W} \approx 3.5\,\mathrm{HP}$$

Next, the pulley and reaction forces must be derived using the given conditions:

$$T_2 = 200\,\mathrm{N}$$
$$d_{\text{pulley}} = 0.254\,\mathrm{m}$$
$$\omega = 188.5\,\mathrm{rad\,s^{-1}}$$
$$P_{\text{mech}} = 2612.19\,\mathrm{W}$$
$$W_{\text{pump}} = 55.6\,\mathrm{N}$$

$$\tau = \frac{P_{\text{mech}}}{\omega} = \frac{2612.19}{188.5} \approx 13.85\,\mathrm{N\,m}$$

Deriving the tension on the tight side of the belt:

$$T_1 = T_2 + \frac{2 \cdot \tau}{d_{\text{pulley}}} = 200 + \frac{2 \cdot 13.85}{0.254} \approx 309.1\,\mathrm{N}$$

Deriving reaction forces at the pulley center:

$$R_{pz} = T_1 + T_2 = 309.1 + 200 = 509.1\,\mathrm{N}$$
$$R_{px} = 0\,\mathrm{N}$$

Deriving moments at the pulley center:

$$M_{px} = T_1 \cdot 0.102 + T_2 \cdot 0.102 = 309.1 \cdot 0.102 + 200 \cdot 0.102 \approx 52.35\,\mathrm{N\,m}$$
$$M_{py} = -\tau = -13.85\,\mathrm{N\,m}$$

2.3 Equilibrium Equations

Analysis code provided in the Appendix. Let the reaction forces at each bolt be defined as:

$$\begin{aligned}
\text{Bolt 1:}\quad & R_{1x}, R_{1y}, R_{1z} \\
\text{Bolt 2:}\quad & R_{2x}, R_{2y}, R_{2z} \\
\text{Bolt 3:}\quad & R_{3x}, R_{3y}, R_{3z} \\
\text{Bolt 4:}\quad & R_{4x}, R_{4y}, R_{4z}
\end{aligned}$$

Assumptions

$$\begin{aligned}
R_{ix} &= 0 \quad \text{for } i = 1,2,3,4 \\
R_{iy} &= 0 \quad \text{for } i = 1,2,3,4 \\
P_f &= 1697.92\,\text{W} \\
P_{\text{mech}} &= 2612.19\,\text{W} \\
\omega &= 188.5\,\text{rad}\,\text{s}^{-1} \\
\tau &= 13.85\,\text{N}\,\text{m} \\
T_1 &= 309.1\,\text{N} \\
T_2 &= 200\,\text{N} \\
R_{px} &= 0\,\text{N} \\
R_{pz} &= 509.1\,\text{N} \\
M_{px} &= 52.35\,\text{N}\,\text{m} \\
M_{py} &= -13.85\,\text{N}\,\text{m} \\
W &= 55.6\,\text{N}
\end{aligned}$$

Applying these constraints, a set of equilibrium equations and conditions can be applied. Z-Direction Force Balance

$$R_{pz} + R_{1z} + R_{2z} + R_{3z} + R_{4z} - W = 0$$

Moment Balance about X-axis

$$M_{px} + R_{1z}(0.0254) + R_{3z}(0.0254) + R_{2z}(0.1016) + R_{4z}(0.1016) - W(0.064) = 0$$

Moment Balance about Y-axis:

$$M_{py} + R_{1z}(0.048) + R_{2z}(0.048) - R_{3z}(0.048) - R_{4z}(0.048) = 0$$

Torsional Symmetry:

$$R_{2z} = R_{4z}$$

The results of this system are solved as shown later.

Now to begin determining the support reactions and forces. Recall the geometric parameters:

$$\begin{aligned}
a &= 0.3048\,\mathrm{m} && \text{(horizontal support length)}\\
b &= 0.2032\,\mathrm{m} && \text{(vertical support length)}\\
w &= 0.1524\,\mathrm{m} && \text{(platform width)}\\
l &= 0.37\,\mathrm{m} && \text{(platform length)}\\
t &= 0.00635\,\mathrm{m} && \text{(platform thickness)}\\
\rho_{\mathrm{Al}} &= 2710\,\mathrm{kg\,m^{-3}}\\
g &= 9.81\,\mathrm{m\,s^{-2}}
\end{aligned}$$

Calculated Platform Weight:

$$W_{\mathrm{plat}} = w \cdot l \cdot t \cdot \rho_{\mathrm{Al}} \cdot g \approx 9.498\,\mathrm{N}$$

Support Direction Cosines:

$$\text{Support z-component: } \frac{a}{\sqrt{a^2+b^2}} \approx 0.8321, \quad \text{Support y-component: } \frac{b}{\sqrt{a^2+b^2}} \approx 0.5546$$

Platform Free-Body Equations
Force Balance in Z-Direction

$$R_{pz} - W - W_{\mathrm{plat}} + S_{1z} + S_{2z} + (F_{t1} + F_{t2}) \cdot \frac{a}{\sqrt{a^2+b^2}} = 0$$

Force Balance in Y-Direction

$$S_{1y} + S_{2y} + (F_{t1} + F_{t2}) \cdot \frac{b}{\sqrt{a^2+b^2}} = 0$$

Moment Balance about X-Axis

$$-M_{px} - W \cdot 0.0635 - W_{\mathrm{plat}} \cdot \frac{l}{2} + (F_{t1} + F_{t2}) \cdot \frac{a}{\sqrt{a^2+b^2}} \cdot (l-b) + (S_{1z} + S_{2z}) \cdot l = 0$$

Moment Balance about Y-Axis

$$M_{py} + (F_{t1} - F_{t2}) \cdot \frac{a}{\sqrt{a^2+b^2}} \cdot \frac{w}{2} + (S_{1z} - S_{2z}) \cdot \frac{w}{2} = 0$$

Torsion Neutralization Conditions

$$S_{1z} = S_{2z}, \quad S_{1y} = S_{2y}$$

The solution to this system is discussed later.

2.4 Shear Force and Moment Diagrams

Shear Force and Moment Diagrams with respect to x were developed based on sampling from the python assembly simulations as shown in Fig. 4.

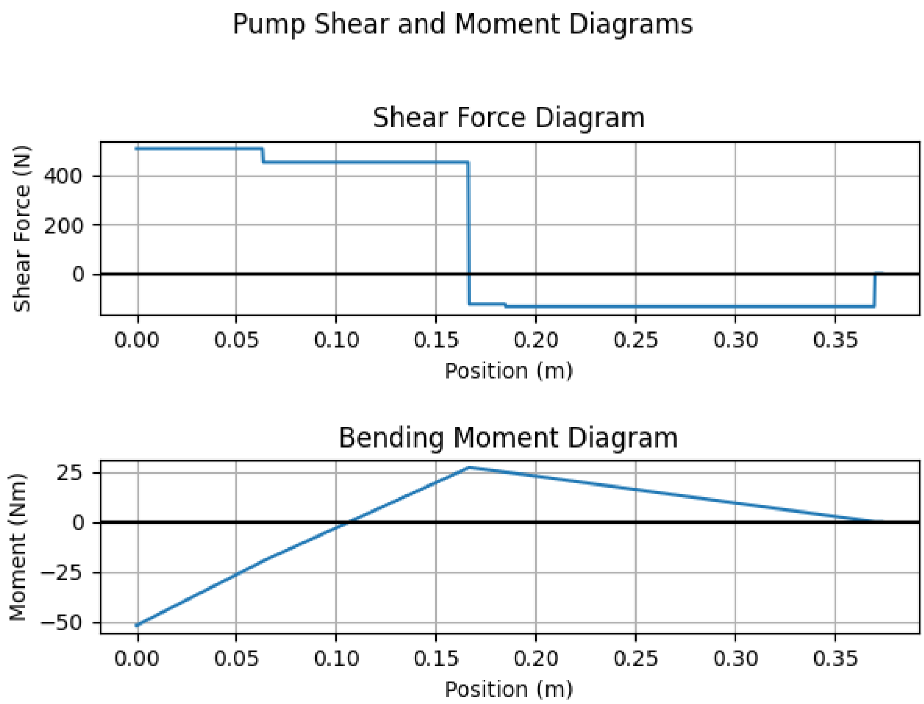

Figure 4: Platform VM Diagrams

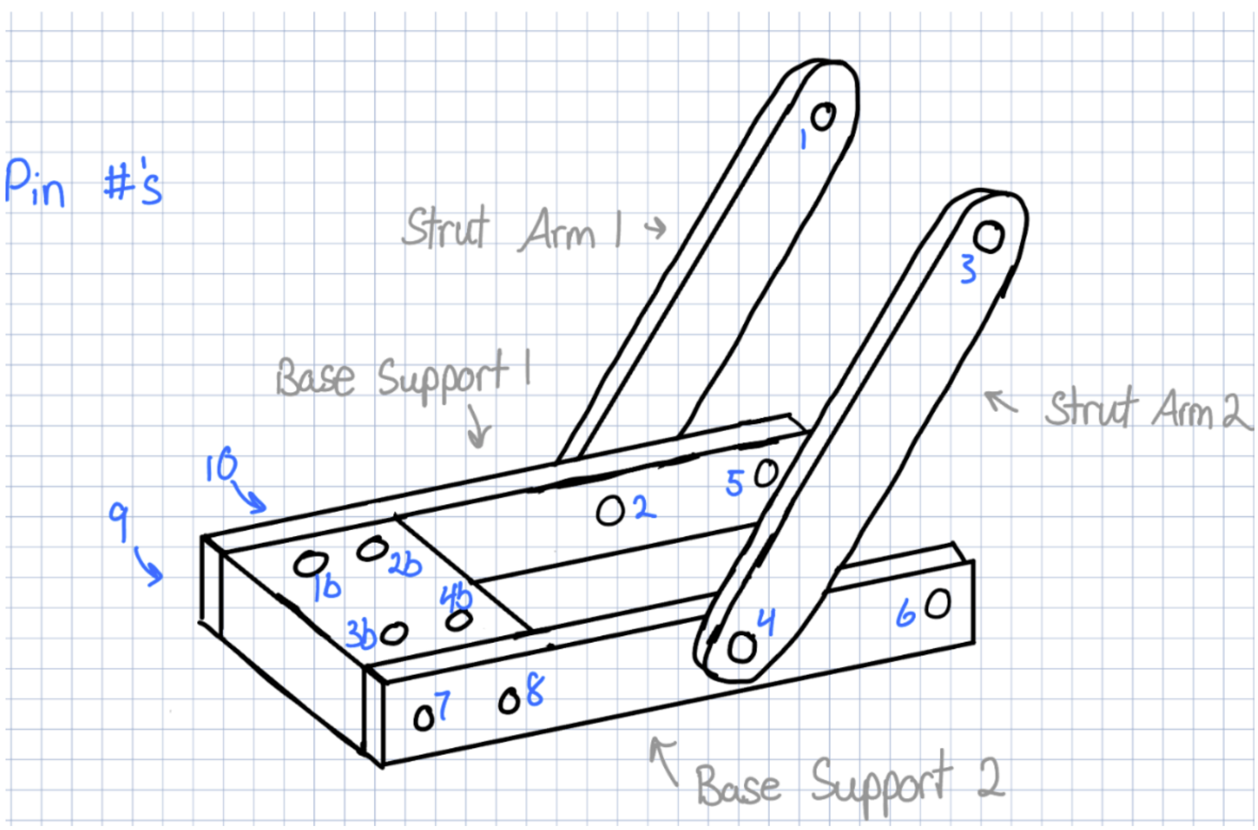

Figure 5: Full labelled assembly

3 Stress Analysis (Symbolic)

3.1 For each component:

3.1.1 Strut Arm 1, 2

- **Stresses Experienced**
  - Compression
- **Stress Concentrations**
  - Plate loaded in compression by a pin through a hole
- **Expected Maximum Stress Locations**
  - The greatest stress can be found at the pin holes. The through holes create a weak spot in the arm around the pins where there is less solid material, decreasing the normal area and hence the nominal stress. This is also where the stress concentration will occur.

3.1.2 Pins 1–10 (Treating bolts as pins for the purposes of analysis)

- **Stresses Experienced**
  - Direct shear
- **Stress Concentrations**
  - No major stress concentrations
- **Expected Maximum Stress Locations**
  - The greatest stress can be found at the meeting point of connected components, because this is where the direct shear occurs. Assuming ideal construction, the direct shear is the only acting stress on the pins, so this must be the point of greatest stress.

3.1.3 Base Support Arm 1, 2

- **Stresses Experienced**
  - Bending, torsion, axial compression (axial compression only between wall connection and strut arm connection)
- **Stress Concentrations**
  - Rectangular bar with transverse hole under bending, torsion, and axial compression
- **Expected Maximum Stress Locations**
  - At the four pin holes, where reduced cross-section increases stress and concentration.

3.1.4 Pump Platform (Plate)

- **Stresses Experienced**
  - Bending, torsion, tension, transverse shear
- **Stress Concentrations**
  - Negligible (hole diameter small relative to plate dimensions)
- **Expected Maximum Stress Locations**
  - At the bolt hole interfacing with the pump, where reaction force is highest.

3.2 For each unique component:

3.2.1 Strut Arm 2

a. Nominal stress (two-force member):

$$\sigma_{\text{nom}} = \frac{F}{(w-d)\,t}$$

b. Stress concentration (transverse hole in compression):

$$\sigma_{\max} = K_t \frac{F}{(w-d)\,t}$$

c. Principal stresses (uniaxial compression):

$$\begin{aligned} \sigma_1 &= \sigma_{\max}, \\ \sigma_2 &= 0, \\ \tau &= 0.5\,\sigma_{\max}. \end{aligned}$$

3.2.2 Pin 4

a. Nominal shear stress:

$$\tau_{\text{nom}} = \frac{4F}{\pi d^2}$$

b. Stress concentration: none

c. Direct shear (pin):

$$\begin{aligned} \sigma_1 &= \tau_{\max}, \\ \sigma_2 &= -\tau_{\max}, \\ \tau &= \tau_{\max}. \end{aligned}$$

3.2.3 Base Support Arm 2

a. Nominal Stresses (Using $\alpha = \frac{1}{3} * (1 - 0.63 * \frac{t}{b} + 0.052 * \frac{t}{b}^5)$)

i. Location 7:
   i. $\sigma_b = \frac{M_7 y}{I}$
   ii. $\tau_{tor} = \frac{T_7}{\alpha \cdot b \cdot t^2}$

ii. Location 8:
   i. $\sigma_b = \frac{M_8 y}{I}$
   ii. $\tau_{tor} = \frac{T_8}{\alpha \cdot b \cdot t^2}$

iii. Location 4:
   i. $\sigma_b = \frac{M_4 y}{I}$
   ii. $\sigma_{axial} = \frac{F}{A}$
   iii. $\tau_{tor} = \frac{T_4}{\alpha \cdot b \cdot t^2}$

iv. Location 6:
   i. $\sigma_b = \frac{M_6 y}{I}$
   ii. $\sigma_{axial} = \frac{F}{A}$
   iii. $\tau_{tor} = \frac{T_6}{\alpha \cdot b \cdot t^2}$

b. Stress Concentration

i. Location 7: Rectangular bar with transverse hole in bending and torsion (h/w = 0.5)
   i. $\sigma_{b,\max} = K_t \frac{M_7 y}{I}$
   ii. $\sigma_{axial,\max} = 0$
   iii. $\sigma_{\max} = \sigma_{b,\max} + \sigma_{axial,\max}$
   iv. $\tau_{tor,\max} = K_t \frac{T_7}{\alpha \cdot b \cdot t^2}$

ii. Location 8: Rectangular bar with transverse hole in bending and torsion (h/w $\geq$ 1)
   i. $\sigma_{b,\max} = K_t \frac{M_8 y}{I}$
   ii. $\sigma_{axial,\max} = 0$
   iii. $\sigma_{\max} = \sigma_{b,\max} + \sigma_{axial,\max}$
   iv. $\tau_{tor,\max} = K_t \frac{T_8}{\alpha \cdot b \cdot t^2}$

iii. Location 4: Rectangular bar with transverse hole in bending, torsion, and compression (h/w $\geq$ 1.0)
   i. $\sigma_{b,\max} = K_t \frac{M_4 y}{I}$
   ii. $\sigma_{axial,\max} = K_t \frac{F}{t \cdot (w-d)}$
   iii. $\sigma_{\max} = \sigma_{b,\max} + \sigma_{axial,\max}$
   iv. $\tau_{tor,\max} = K_t \frac{T_4}{\alpha \cdot b \cdot t^2}$

iv. Location 6: Rectangular bar with transverse hole in bending and torsion (h/w = 0.5)

   i. $\sigma_{b,\max} = K_t \frac{M_6 y}{I}$
   ii. $\sigma_{axial,\max} = K_t \frac{F}{t \cdot (w-d)}$
   iii. $\sigma_{\max} = \sigma_{b,\max} + \sigma_{axial,\max}$
   iv. $\tau_{tor,\max} = K_t \frac{T_6}{\alpha \cdot b \cdot t^2}$

c. Bending, torsion, and axial compression are the only non-negligible acting forces, max shear and principal

   i. Location 7:
      i. $\sigma_1 = \frac{\sigma_{\max}}{2} + \sqrt{(\frac{\sigma_{\max}}{2})^2 + (\tau_{tor,\max})^2}$
      ii. $\sigma_2 = \frac{\sigma_{\max}}{2} - \sqrt{(\frac{\sigma_{\max}}{2})^2 + (\tau_{tor,\max})^2}$
      iii. $\tau_{\max} = \sqrt{(\frac{\sigma_{\max}}{2})^2 + (\tau_{tor,\max})^2}$
   ii. Location 8:
      i. $\sigma_1 = \frac{\sigma_{\max}}{2} + \sqrt{(\frac{\sigma_{\max}}{2})^2 + (\tau_{tor,\max})^2}$
      ii. $\sigma_2 = \frac{\sigma_{\max}}{2} - \sqrt{(\frac{\sigma_{\max}}{2})^2 + (\tau_{tor,\max})^2}$
      iii. $\tau_{\max} = \sqrt{(\frac{\sigma_{\max}}{2})^2 + (\tau_{tor,\max})^2}$
   iii. Location 4:
      i. $\sigma_1 = \frac{\sigma_{\max}}{2} + \sqrt{(\frac{\sigma_{\max}}{2})^2 + (\tau_{tor,\max})^2}$
      ii. $\sigma_2 = \frac{\sigma_{\max}}{2} - \sqrt{(\frac{\sigma_{\max}}{2})^2 + (\tau_{tor,\max})^2}$
      iii. $\tau_{\max} = \sqrt{(\frac{\sigma_{\max}}{2})^2 + (\tau_{tor,\max})^2}$
   iv. Location 6:
      i. $\sigma_1 = \frac{\sigma_{\max}}{2} + \sqrt{(\frac{\sigma_{\max}}{2})^2 + (\tau_{tor,\max})^2}$
      ii. $\sigma_2 = \frac{\sigma_{\max}}{2} - \sqrt{(\frac{\sigma_{\max}}{2})^2 + (\tau_{tor,\max})^2}$
      iii. $\tau_{\max} = \sqrt{(\frac{\sigma_{\max}}{2})^2 + (\tau_{tor,\max})^2}$

3.2.4 Pump Platform

a. Nominal Stresses

   i. Location 7: bending (single axis) and shear
      i. $\sigma_b = \frac{M_7 y}{I_x}$
      ii. $\tau = \frac{3\tau_{dir}}{2A}$
   ii. Location 8: bending (single axis) and shear
      i. $\sigma_b = \frac{M_8 y}{I_x}$
      ii. $\tau = \frac{3\tau_{dir}}{2A}$

iii. Location 9: bending (single axis) and shear
   i. $\sigma_b = \frac{M_9 y}{I_x}$
   ii. $\tau = \frac{3\tau_{dir}}{2A}$
iv. Location 10: bending (single axis) and shear
   i. $\sigma_b = \frac{M_{10} y}{I_x}$
   ii. $\tau = \frac{3\tau_{dir}}{2A}$
v. Location 1b: Bending (about x and y)
   i. $\sigma_{b,x} = \frac{M_{1bx} y}{I_x}$
   ii. $\sigma_{b,y} = \frac{M_{1by} y}{I_y}$
vi. Location 2b: Bending (about x and y)
   i. $\sigma_{b,x} = \frac{M_{2bx} y}{I_x}$
   ii. $\sigma_{b,y} = \frac{M_{2by} y}{I_y}$
vii. Location 3b: Bending (about x and y)
   i. $\sigma_{b,x} = \frac{M_{3bx} y}{I_x}$
   ii. $\sigma_{b,y} = \frac{M_{3by} y}{I_y}$
viii. Location 4b: Bending (about x and y)
   i. $\sigma_{b,x} = \frac{M_{4bx} y}{I_x}$
   ii. $\sigma_{b,y} = \frac{M_{4by} y}{I_y}$

b. Stress Concentrations

i. Location 7: Rectangular bar with transverse hole in compression (h/w = 0.5)
   i. $\sigma_{b,\max} = K_t \frac{M_7 y}{I}$
   ii. $\tau = \frac{3\tau_{dir}}{2A}$ (No substantial stress concentration)
ii. Location 8: Rectangular bar with transverse hole in compression (h/w = 0.5)
   i. $\sigma_{b,\max} = K_t \frac{M_8 y}{I}$
   ii. $\tau = \frac{3\tau_{dir}}{2A}$ (No substantial stress concentration)
iii. Location 9: Rectangular bar with transverse hole in compression (h/w = 0.5)
   i. $\sigma_{b,\max} = K_t \frac{M_9 y}{I}$
   ii. $\tau = \frac{3\tau_{dir}}{2A}$ (No substantial stress concentration)
iv. Location 10: Rectangular bar with transverse hole in compression (h/w = 0.5)
   i. $\sigma_{b,\max} = K_t \frac{M_{10} y}{I}$
   ii. $\tau = \frac{3\tau_{dir}}{2A}$ (No substantial stress concentration)
v. Location 1b: Rectangular bar with a transverse hole in bending

i. $\sigma_{b,x} = K_t \frac{M_{1bx}y}{I_x}$
ii. $\sigma_{b,y} = K_t \frac{M_{1by}y}{I_y}$

vi. Location 2b: Rectangular bar with a transverse hole in bending
i. $\sigma_{b,x} = K_t \frac{M_{2bx}y}{I_x}$
ii. $\sigma_{b,y} = K_t \frac{M_{2by}y}{I_y}$

vii. Location 3b: Rectangular bar with a transverse hole in bending
i. $\sigma_{b,x} = K_t \frac{M_{3bx}y}{I_x}$
ii. $\sigma_{b,y} = K_t \frac{M_{3by}y}{I_y}$

viii. Location 4b: Rectangular bar with a transverse hole in bending
i. $\sigma_{b,x} = K_t \frac{M_{4bx}y}{I_x}$
ii. $\sigma_{b,y} = K_t \frac{M_{4by}y}{I_y}$

c. Max Shear and Principal Stresses

i. Location 7:
i. $\sigma_1 = \frac{\sigma_{b,\max}}{2} + \sqrt{(\frac{\sigma_{b,\max}}{2})^2 + (\tau_{tor,\max})^2}$
ii. $\sigma_2 = \frac{\sigma_{b,\max}}{2} - \sqrt{(\frac{\sigma_{b,\max}}{2})^2 + (\tau_{tor,\max})^2}$
iii. $\tau_{\max} = \sqrt{(\frac{\sigma_{b,\max}}{2})^2 + (\tau_{tor,\max})^2}$

ii. Location 8:
i. $\sigma_1 = \frac{\sigma_{b,\max}}{2} + \sqrt{(\frac{\sigma_{b,\max}}{2})^2 + (\tau_{tor,\max})^2}$
ii. $\sigma_2 = \frac{\sigma_{b,\max}}{2} - \sqrt{(\frac{\sigma_{b,\max}}{2})^2 + (\tau_{tor,\max})^2}$
iii. $\tau_{\max} = \sqrt{(\frac{\sigma_{b,\max}}{2})^2 + (\tau_{tor,\max})^2}$

iii. Location 4:
i. $\sigma_1 = \frac{\sigma_{b,\max}}{2} + \sqrt{(\frac{\sigma_{b,\max}}{2})^2 + (\tau_{tor,\max})^2}$
ii. $\sigma_2 = \frac{\sigma_{b,\max}}{2} - \sqrt{(\frac{\sigma_{b,\max}}{2})^2 + (\tau_{tor,\max})^2}$
iii. $\tau_{\max} = \sqrt{(\frac{\sigma_{b,\max}}{2})^2 + (\tau_{tor,\max})^2}$

iv. Location 6:
i. $\sigma_1 = \frac{\sigma_{b,\max}}{2} + \sqrt{(\frac{\sigma_{b,\max}}{2})^2 + (\tau_{tor,\max})^2}$
ii. $\sigma_2 = \frac{\sigma_{b,\max}}{2} - \sqrt{(\frac{\sigma_{b,\max}}{2})^2 + (\tau_{tor,\max})^2}$
iii. $\tau_{\max} = \sqrt{(\frac{\sigma_{b,\max}}{2})^2 + (\tau_{tor,\max})^2}$

v. Location 1b:

i. $\sigma_1 = \sigma_{b,x} + \sigma_{b,y}$
ii. $\sigma_2 = 0$
iii. $\tau_{\max} = \frac{\sigma_{b,x}+\sigma_{b,y}}{2}$

vi. Location 2b:

i. $\sigma_1 = \sigma_{b,x} + \sigma_{b,y}$
ii. $\sigma_2 = 0$
iii. $\tau_{\max} = \frac{\sigma_{b,x}+\sigma_{b,y}}{2}$

vii. Location 3b:

i. $\sigma_1 = \sigma_{b,x} + \sigma_{b,y}$
ii. $\sigma_2 = 0$
iii. $\tau_{\max} = \frac{\sigma_{b,x}+\sigma_{b,y}}{2}$

viii. Location 4b:

i. $\sigma_1 = \sigma_{b,x} + \sigma_{b,y}$
ii. $\sigma_2 = 0$
iii. $\tau_{\max} = \frac{\sigma_{b,x}+\sigma_{b,y}}{2}$

4 Failure Determination (Symbolic)

4.1 Arm 2

4.1.1 Factor of Safety

$$n_{\text{ductile}} = \frac{S_y}{\sqrt{\sigma_{\max}^2}},$$
$$n_{\text{brittle}} = \frac{S_{uc}}{\sigma_{\max}}.$$

4.1.2 Euler buckling load

$$P_{\text{cr}} = \frac{\pi^2 EI}{L^2}$$

4.2 Pin 4

4.2.1 Factor of Safety

$$n_{\text{ductile}} = \frac{S_y}{\sqrt{3\,\tau_{\max}^2}},$$
$$n_{\text{brittle}} = \frac{\tau_u}{\tau_{\max}}.$$

4.3 Base Support Arm 2

4.3.1 Factor of Safety

i. Location 4,6,7,8:

(a) $n_{ductile} = \frac{S_y}{\sqrt{\sigma_1^2 - \sigma_1\sigma_2 + \sigma_2^2}}$ [DE Theory for accuracy]

(b) $n_{brittle} = \frac{S_u}{\sigma_1}$ [Maximum normal stress theory]

4.4 Base Platform

4.4.1 Factor of Safety

i. Locations 4,6,7,8:

(a) $n_{ductile} = \frac{S_y}{\sqrt{\sigma_1^2 - \sigma_1\sigma_2 + \sigma_2^2}}$ [DE Theory for accuracy]

(b) $n_{brittle} = \frac{S_u}{\sigma_1}$ [Maximum normal stress theory]

ii. Locations 1b, 2b, 3b, 4b:

(a) $n_{ductile} = \frac{S_y}{\sqrt{\sigma_1^2}}$ [DE Theory for accuracy]

(b) $n_{brittle} = \frac{S_{uc}}{\sigma_1}$ [Maximum normal stress theory]

4.4.2 Maximum Deflection

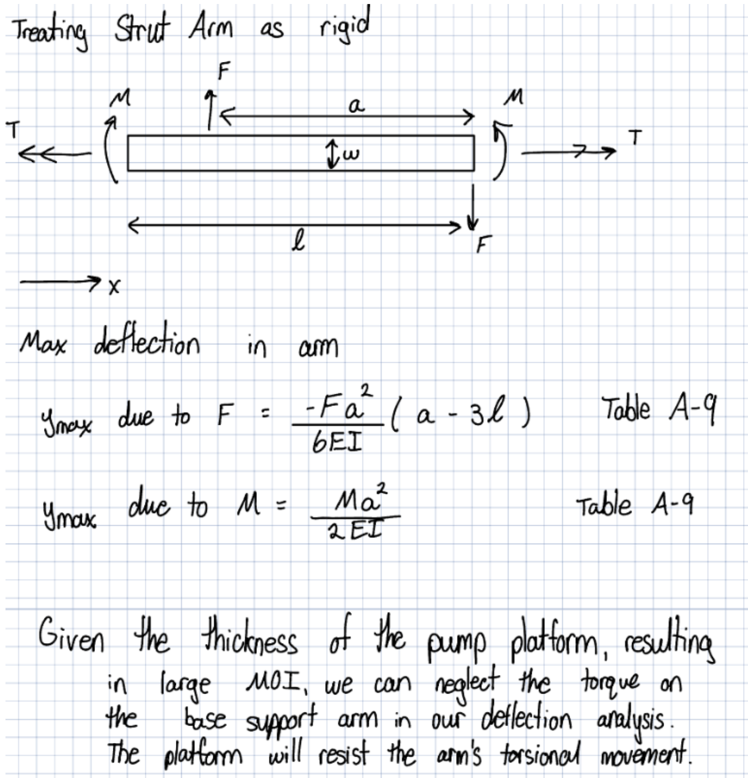

Figure 6: Maximum deflection calculation

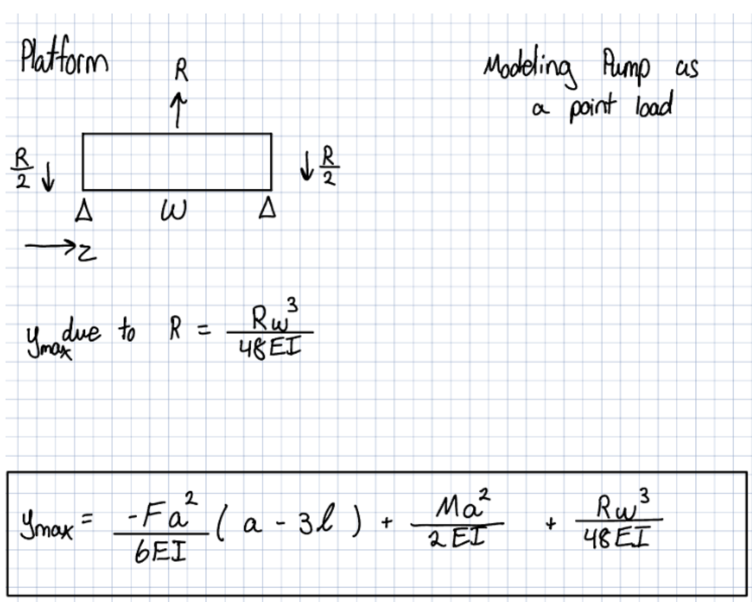

Figure 7: Maximum deflection calculation cont.

5 Design Iteration (Not Symbolic)

5.1 Initial Dimensions

- Platform length (l_p) = 5”
- Platform width (w_p) = 6.938”
- Base support arm thickness (t_{ba}) = 0.25”
- Base support arm width (w_{ba}) = 1”
- Base support arm length (active) (l_{ba}) = 13.497”
- Strut arm thickness (t_s) = 0.25”
- Strut arm width (w_s) = 1”
- Strut arm length (l_s) = 13”
- Strut arm joint distance from wall (b) = 5”

5.2 First Iteration Results

5.2.1 Calculate stress concentration factors (K) for all unique components

- Strut Arm 2 (h/w = 0.5, d/w = 0.25)
  - K_t = 4.8 (Table A-15-12)
- Base Support Arm 2
  - Locations 4,6,7,8: (d/h = 1, d/w = 0.25)
    - K_t (bending) = 1.8 (Table A-15-2)
    - K_t (torsion) = 2.7 (Table A-15-10)
  - Location 4 (d/w = 0.25)
    - K_t (axial) = 2.4 (Table A-15-1)
  - Location 6 (d/w = 0.25, h/w = 0.5)
    - K_t = 4.8 (Table A-15-12)
- Pump Platform Stress Concentrations
  - Locations 7,8,9,10 (d/w = 0.275)
    - K_t = 2.4 (Table A-15-1)
  - Locations 1b,2b,3b,4b (d/h = 0.375, d/w = 0.05)
    - K_t = 2.45 (Table A-15-2)

**Tables in Appendix A*

5.2.2 Using initial dimensions, derive expressions for the following

- Factors of Safety n:
  - Base Arm
    - Location 7 Hole, Ductile: 1.99
    - Location 8 Hole, Ductile: 1.81
    - Location 4 Hole, Ductile: 4.04
    - Location 6 Hole, Ductile: 25.25
    - Location 7 Hole, Brittle: 3.97
    - Location 8 Hole, Brittle: 3.20
    - Location 4 Hole, Brittle: 5.56
    - Location 6 Hole, Brittle: 34.74
  - Strut Arm
    - Location 3 Hole, Ductile: 1609.03
    - Location 4 Hole, Ductile: 1609.03
    - Location 3 Hole, Brittle: 1432.56
    - Location 4 Hole, Brittle: 1432.56
  - Platform
    - Overall, Ductile: 55.53
    - Overall, Brittle: 49.44
  - Pin
    - Overall, Shear: 25.88
- Critical Buckling Load in Support Arm:
  - Strut Arm Critical Buckling Load: 3434.13 N
  - Strut Arm Buckling Factor of Safety: 4.49
- Maximum Deflection of Support Platform: 0.067 mm (Meets design requirements)

**Python code in appendix B*

5.2.3 ANSYS Material Plots

- Global Requirements:
  - Machining speed > 150 spm
  - Water (fresh) resistance: good + excellent
- Base Arm Derived Requirements:
  - $S_y \geq 228$ MPa

- Strut Arm Derived Requirements:
  - Young's Modulus $\geq$ 11.1 GPa
- Base Arm Derived Deflection Requirements:
  - Young's Modulus $\geq$ 4.28 GPa [higher value of 11.1 GPa used for filtering]

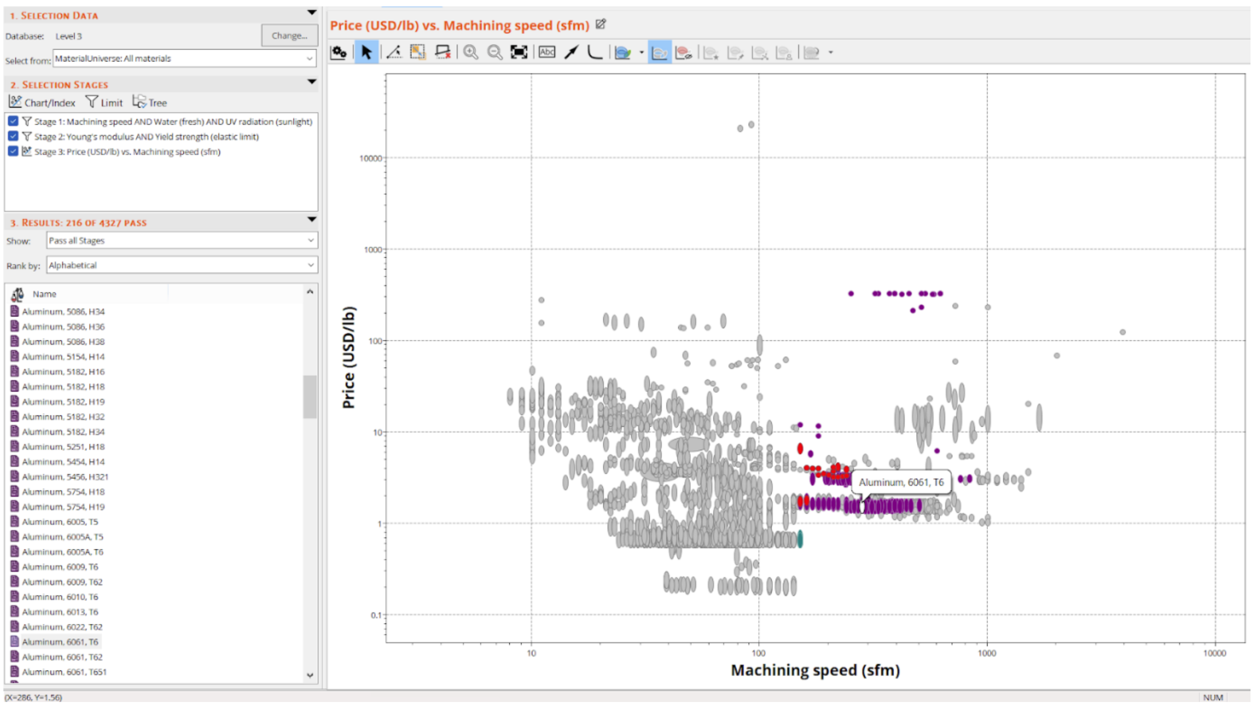

Figure 8: ANSYS Granta EduPack (Level 3) Plot

The material selected was Aluminium 6061 due to it fitting all requirements and wide availability. The material was chosen to fit the failure requirements of the components that would fail first, and thus will be used for the strut arms, base support arms, and platform for simplicity of ordering.

5.2.4 Bill of Materials

Description	Part Number	# Parts Needed	Unit Cost	Total Cost
Base Support Arm 3/16" x 1.25" 3 ft. Aluminum ...	8975K587	1.00	8.69	8.69
Strut Arm 1/4" x 1" 3 ft. Aluminum (6061)	8975K596	1.00	10.84	10.84
Platform 1" x 5" x 1 ft. Aluminum (6061)	8975K352	1.00	60.10	60.10
Bolts 1/4-20 0.375" Zinc-Plated Alloy Steel	90128A241	10.00	0.23	5.68
Bolts 3/8"-16 1.5" Zinc-Flake-Coated Alloy Steel	91274A316	4.00	0.73	2.92
Total				**88.23**

*All parts were sourced from McMaster-Carr

6 Finite Element Validation

6.1 FEA Setup Description

The FEA analysis was conducted by removing the pump and replacing with the resultant vertical forces at the bolt locations. These were calculated previously in our equilibrium calculations. Additionally, the connections of the arms with the back plate were modeled as hinges as they would be in the experiment.

6.2 CAD Screenshots and Setup Explanation

Figure 6 and 7 are the CAD assemblies of our design both with and without the mounted pump, respectively. For the purpose of analysis, the pump was removed and replaced by forces at the bolted connections that would result in a vertical force and a torque on the platform. The platform and strut arms are bolted to the base arms and the connections of the arms at the backplate are hinges.

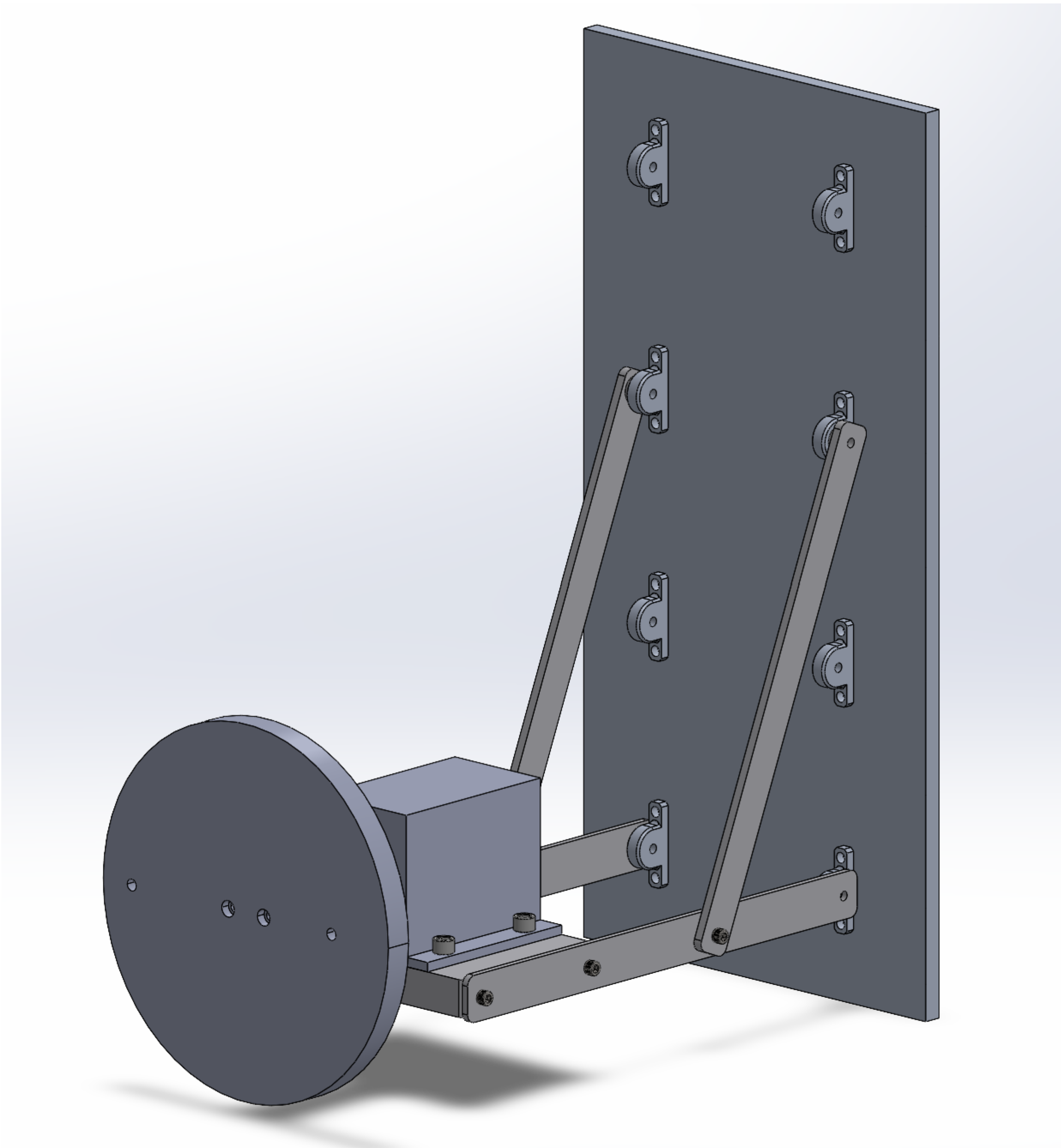

Figure 9: Total Setup Assembly

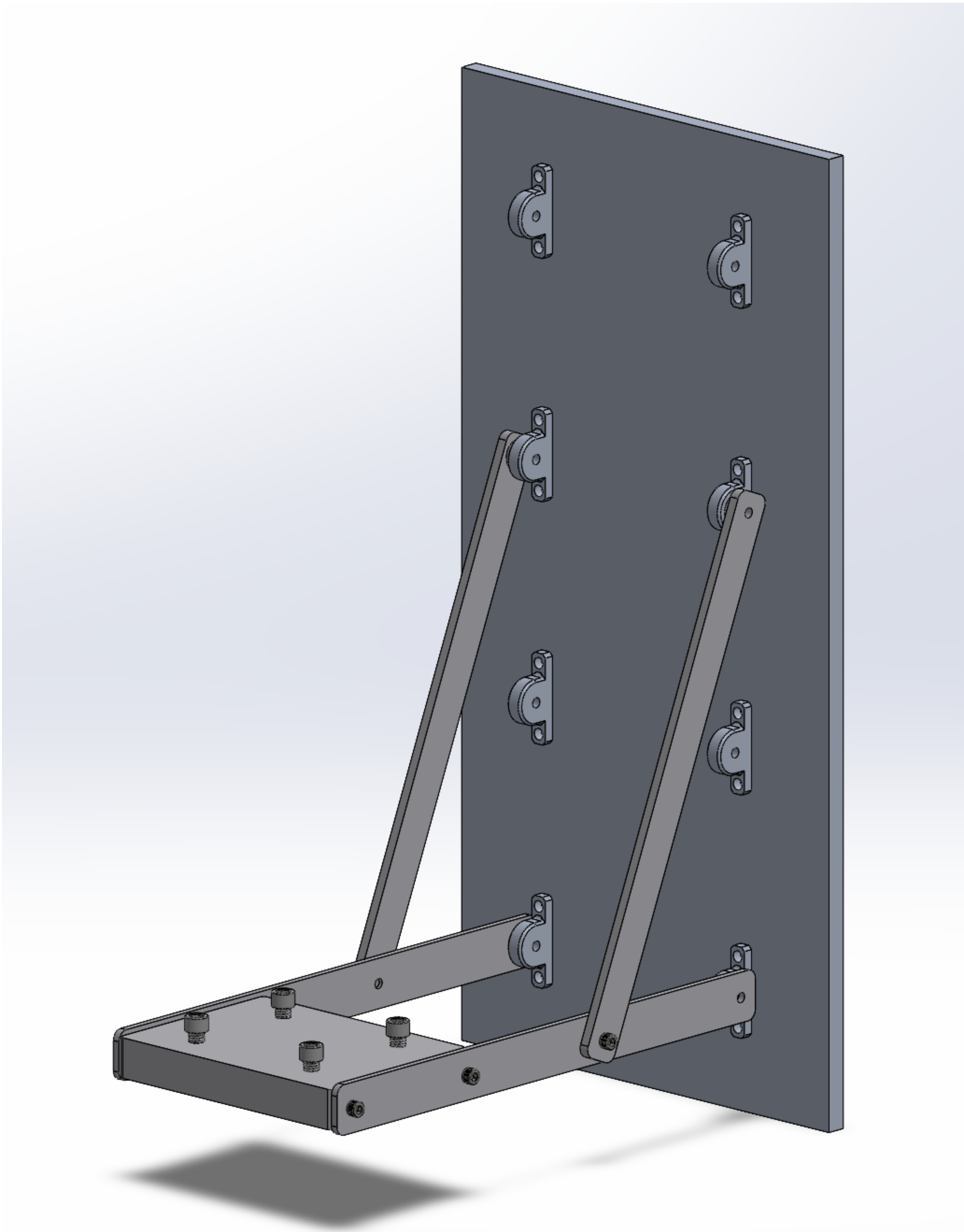

Figure 10: Total Setup Assembly without Pump

6.3 FEA Result Screenshots

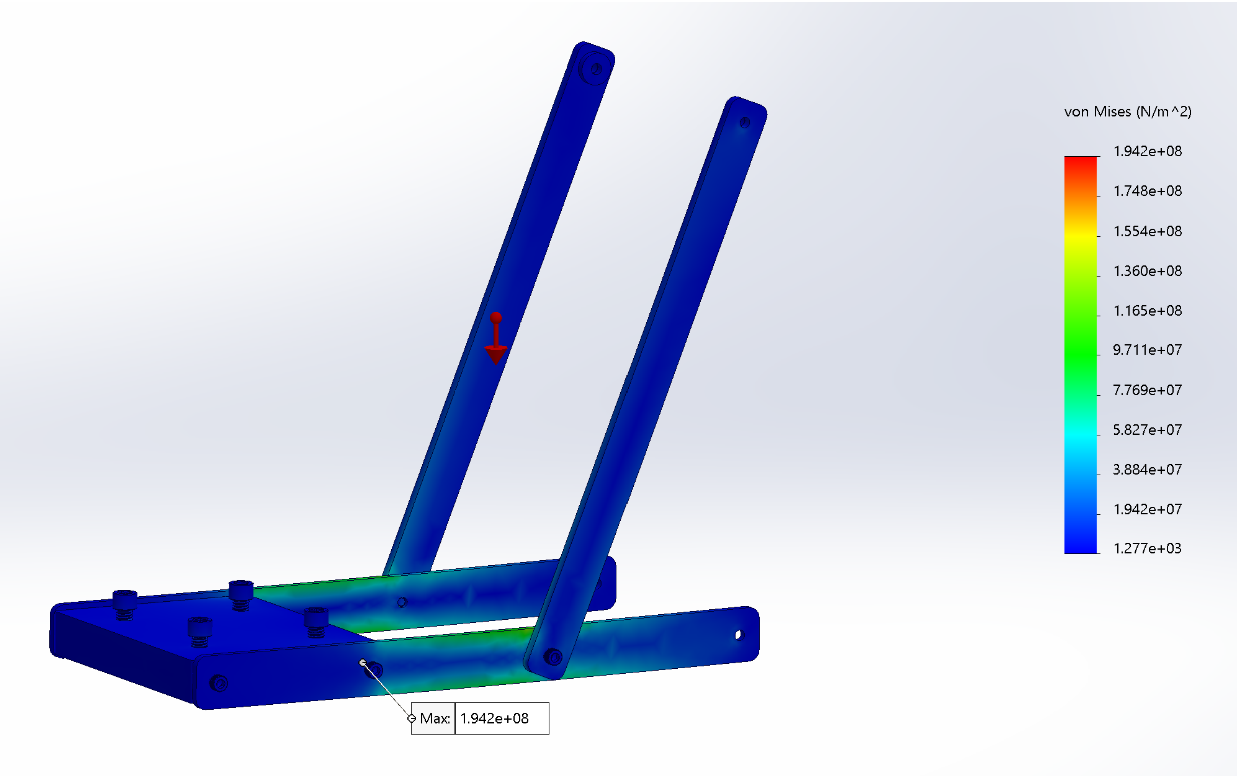

Figure 11: FEA Stress Analysis (Full Assembly)

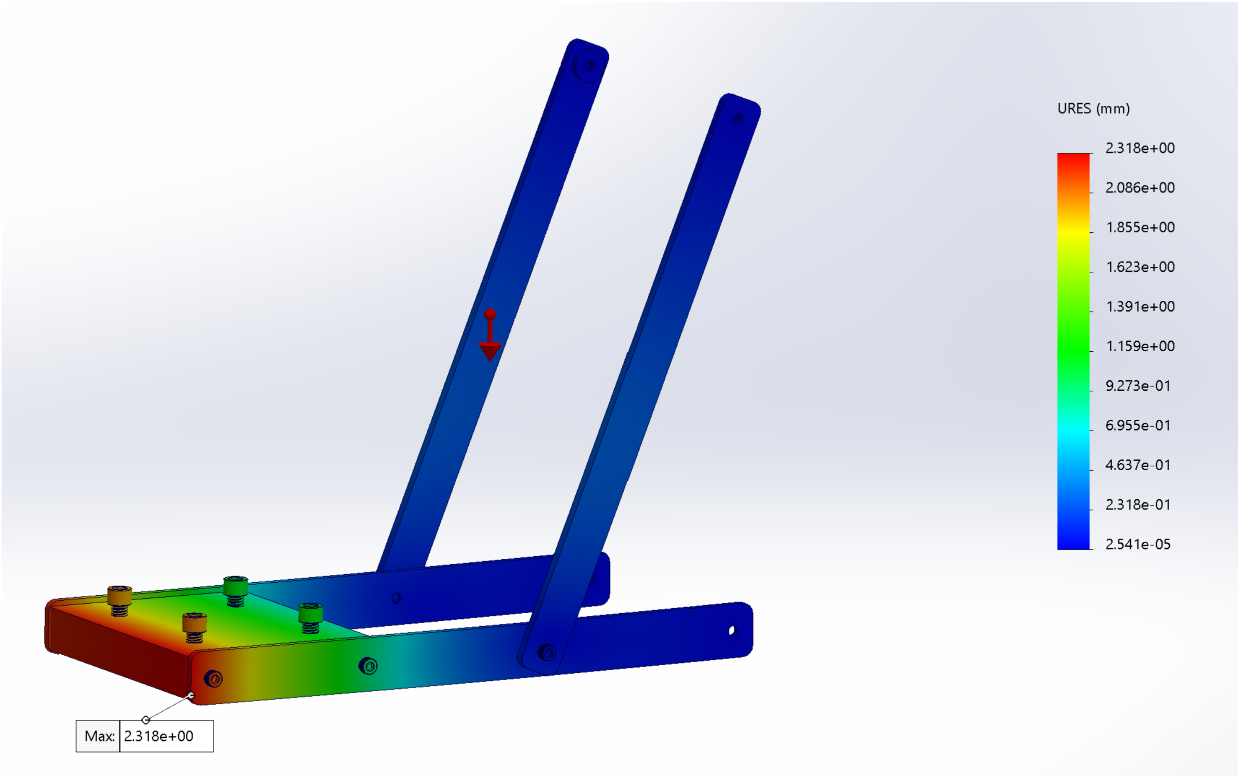

Figure 12: FEA Displacement Analysis (Full Assembly)

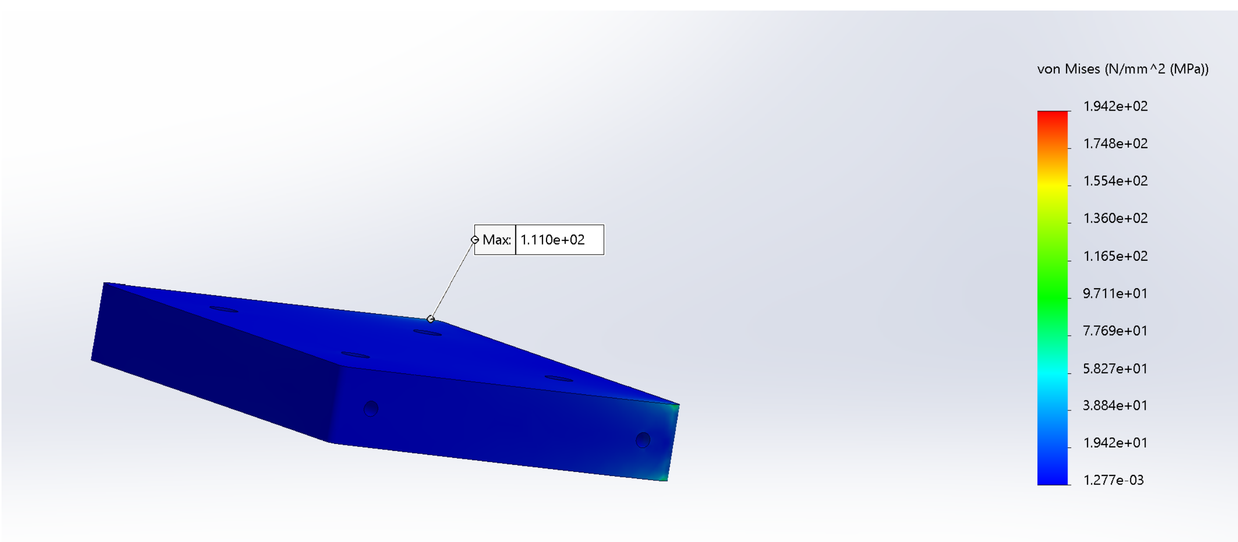

Figure 13: FEA Stress Analysis (Platform)

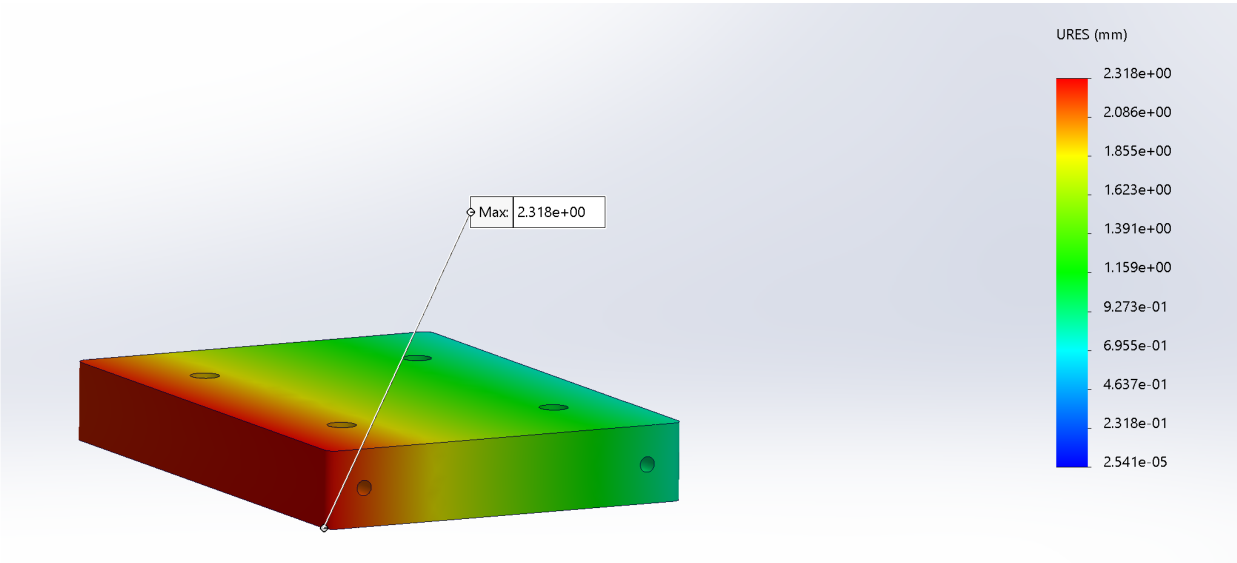

Figure 14: FEA Displacement Analysis (Platform)

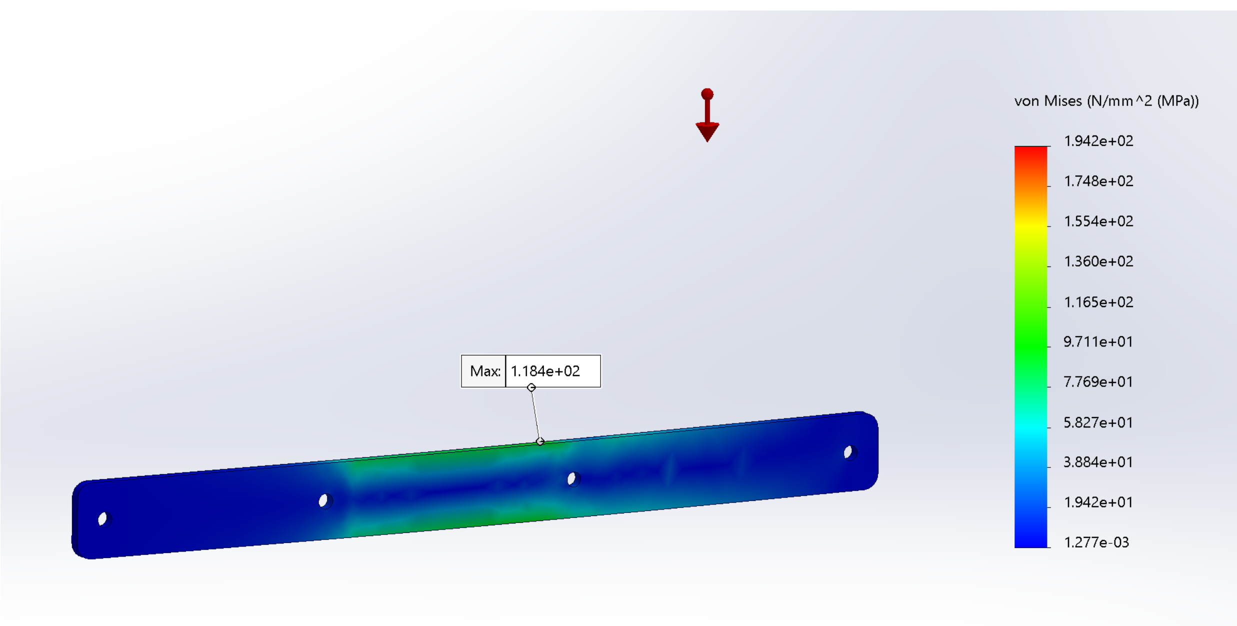

Figure 15: FEA Stress Analysis (Base Arm 1)

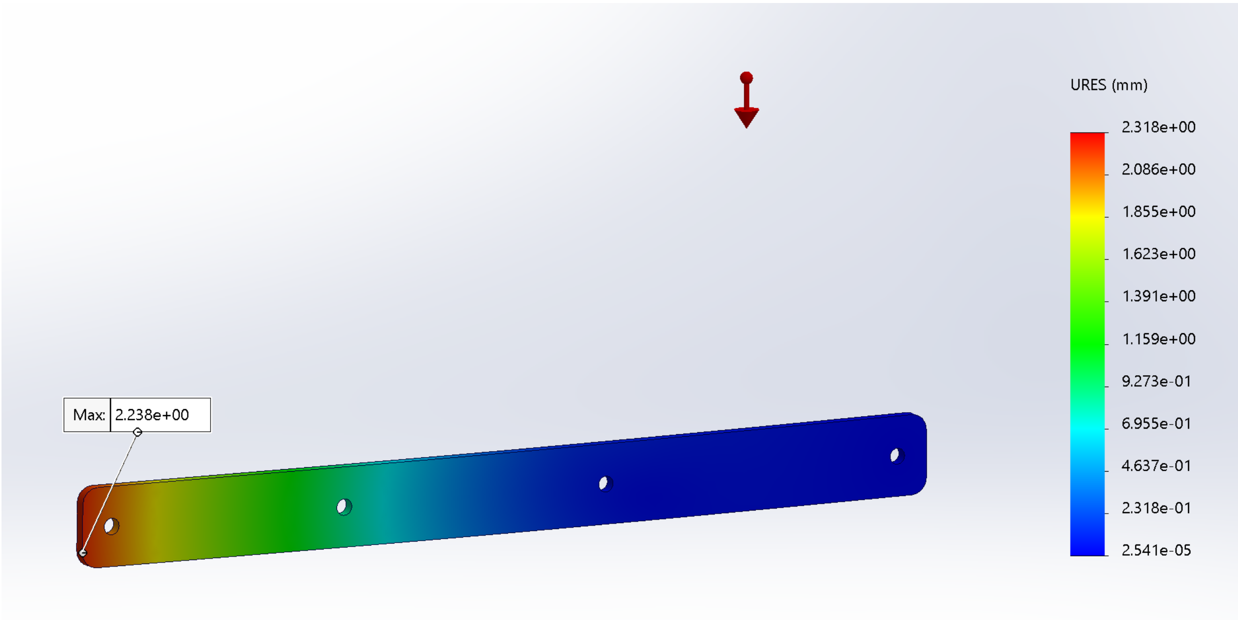

Figure 16: FEA Displacement Analysis (Base Arm 1)

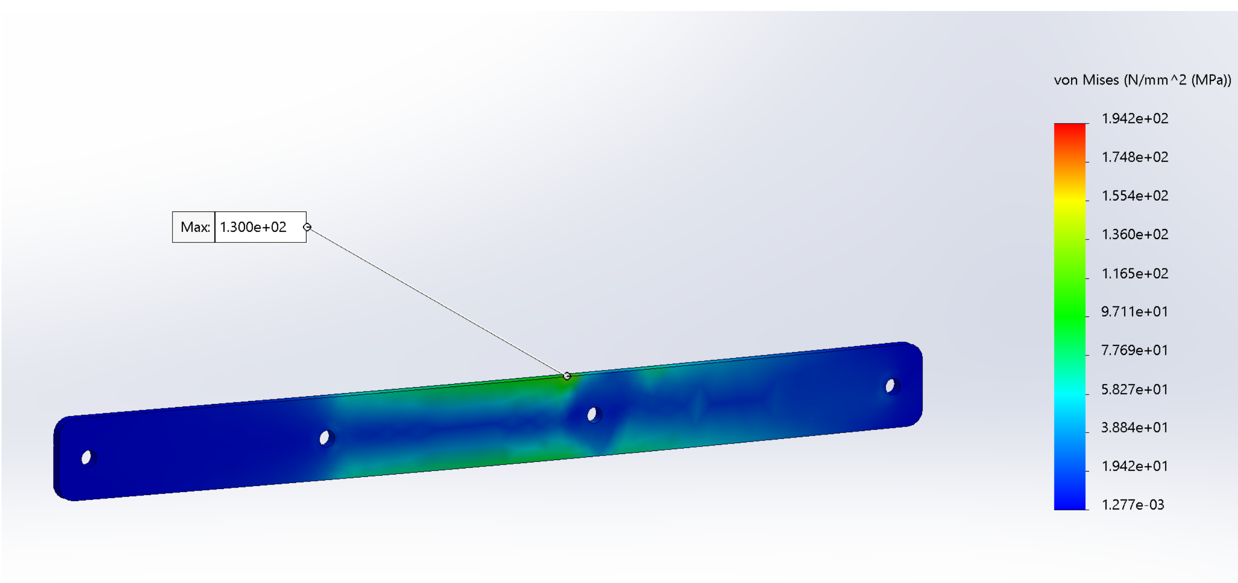

Figure 17: FEA Stress Analysis (Base Arm 2)

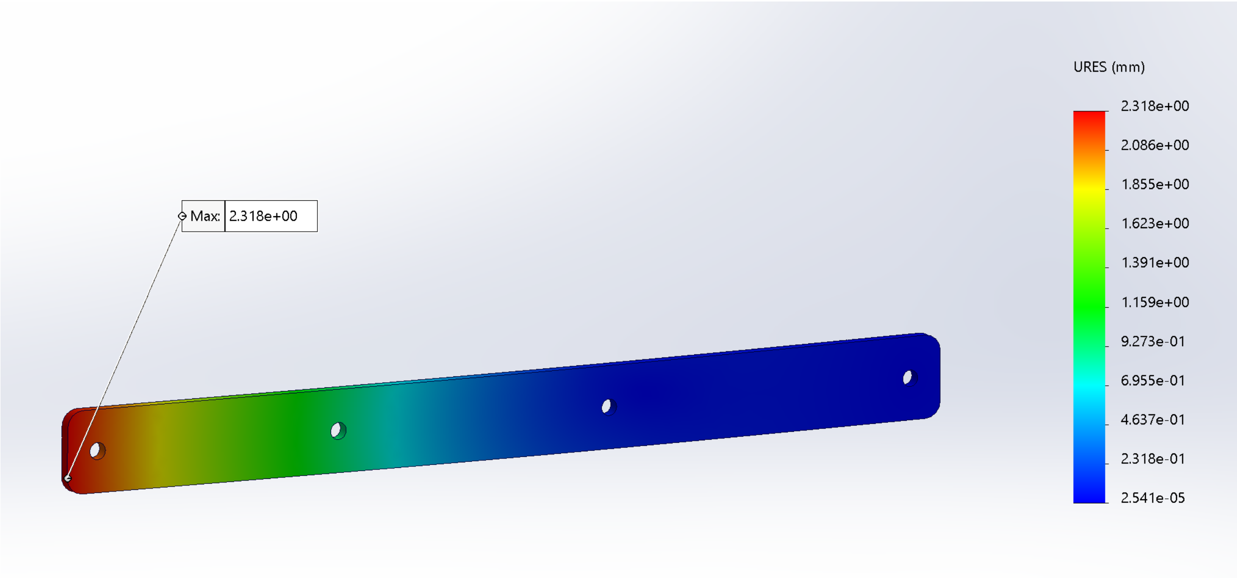

Figure 18: FEA Displacement Analysis (Base Arm 2)

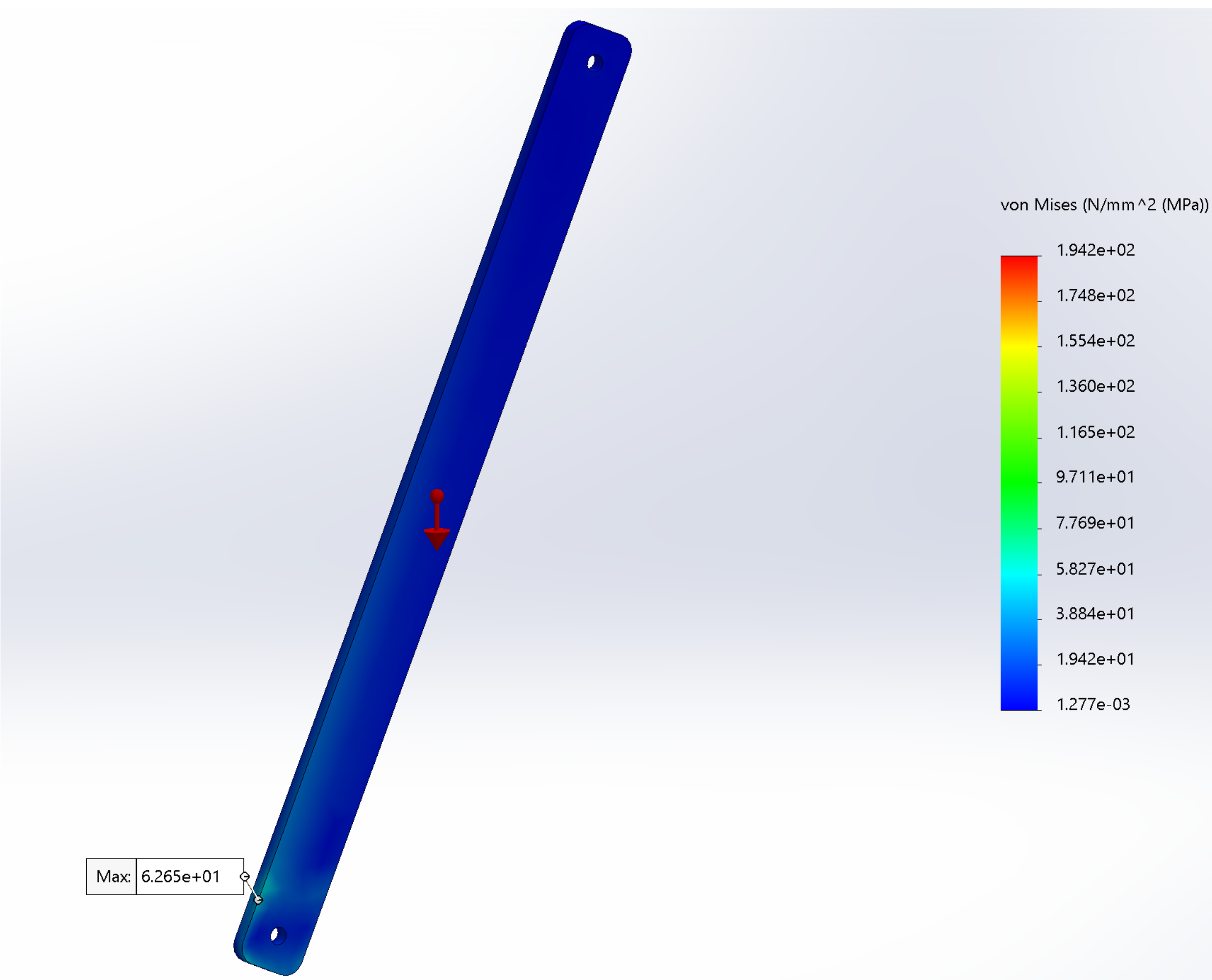

Figure 19: FEA Stress Analysis (Strut Arm 1)

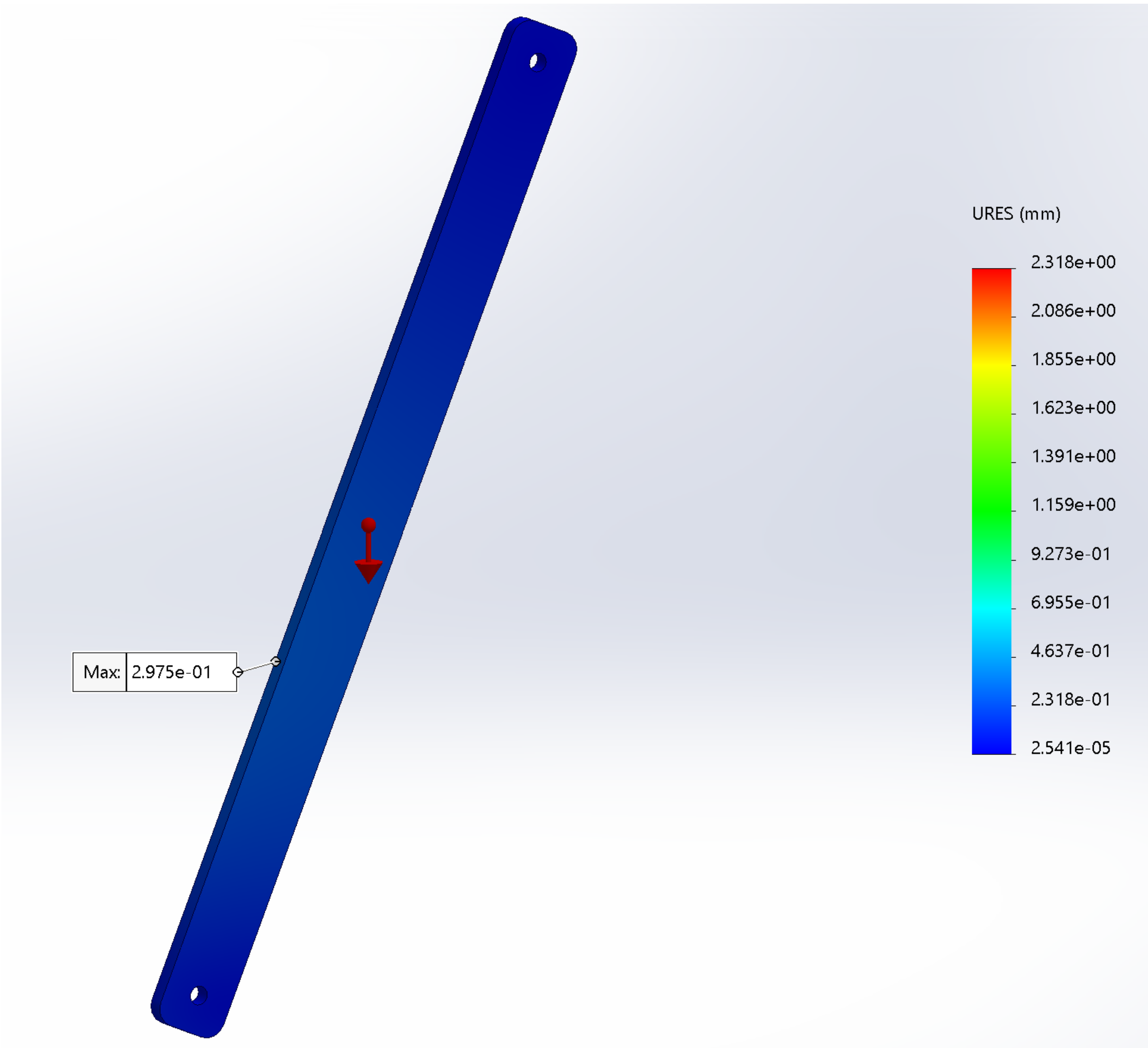

Figure 20: FEA Displacement Analysis (Strut Arm 1)

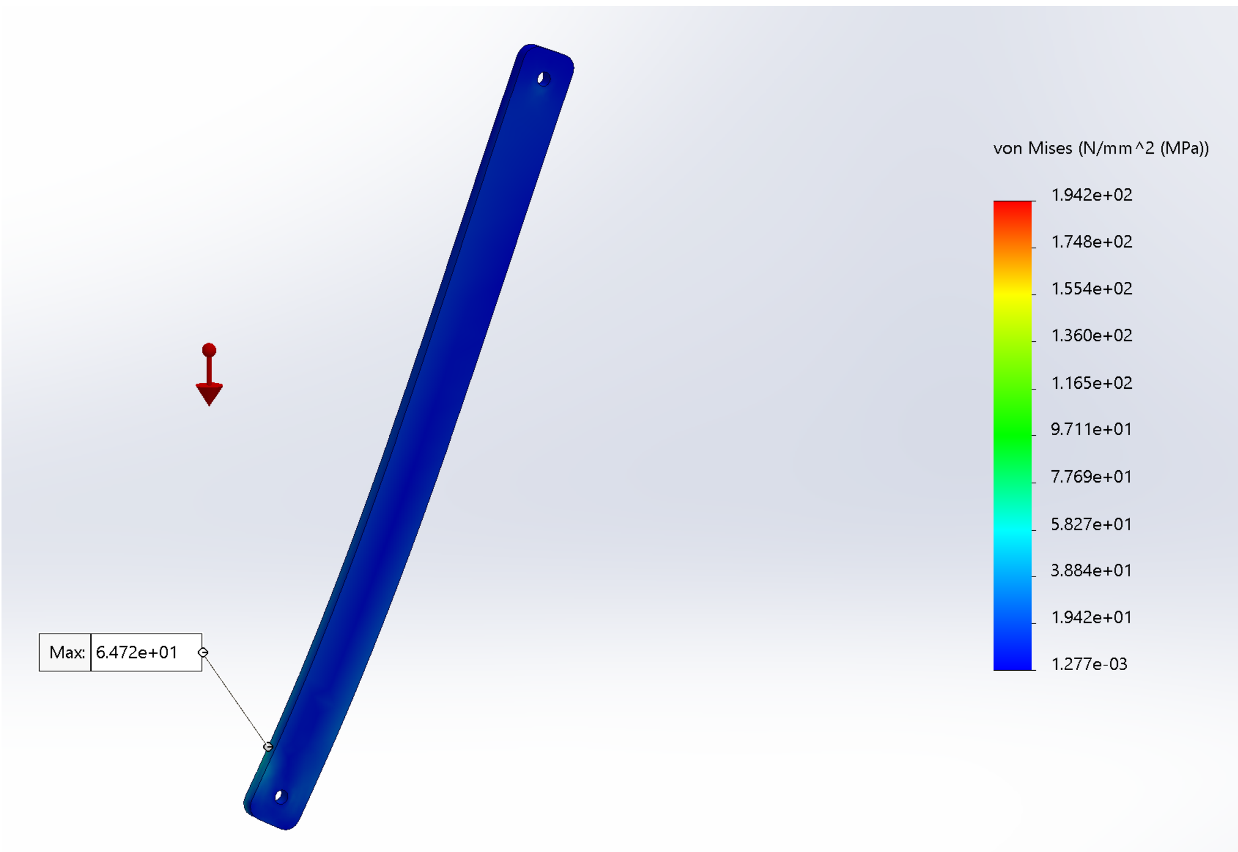

Figure 21: FEA Stress Analysis (Strut Arm 2)

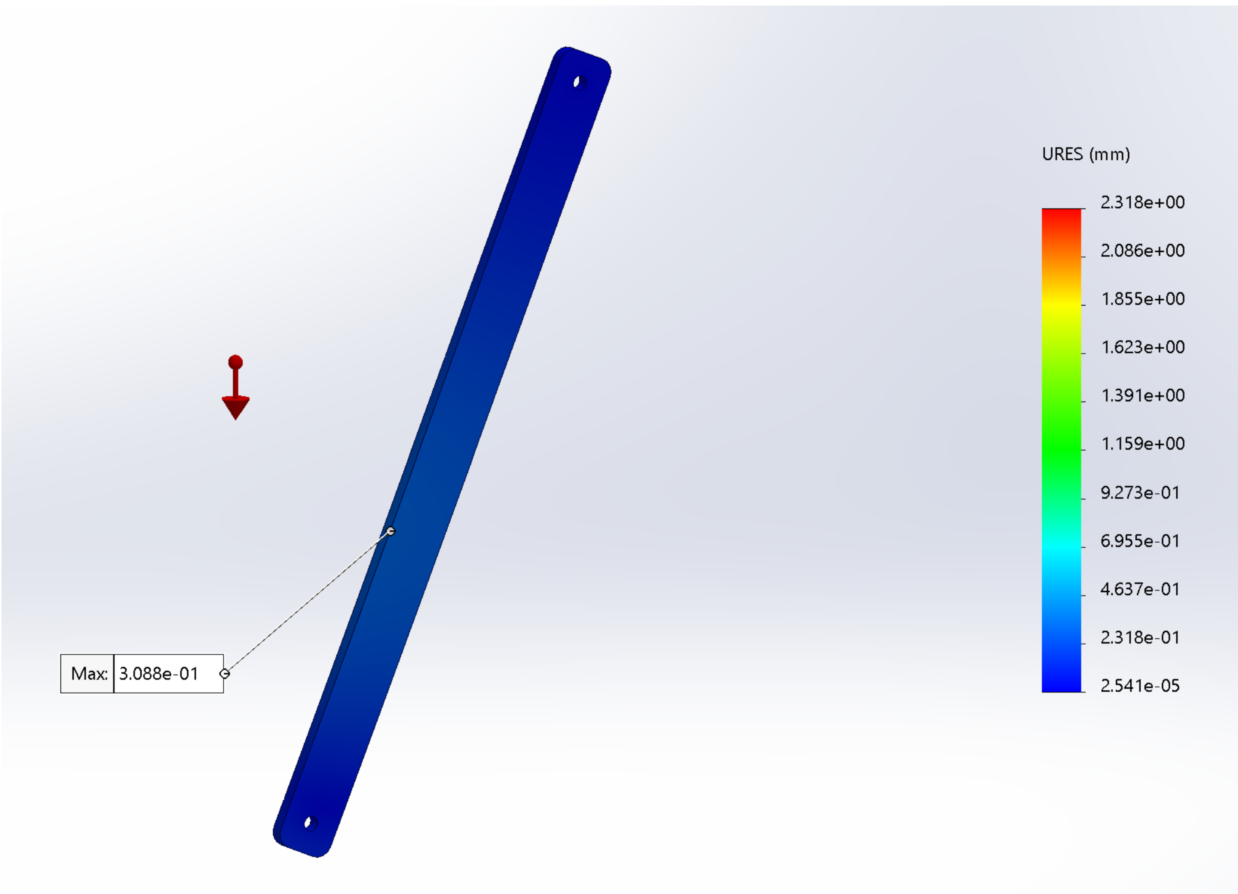

Figure 22: FEA Displacement Analysis (Strut Arm 2)

6.4 Convergence Plot and Discussion

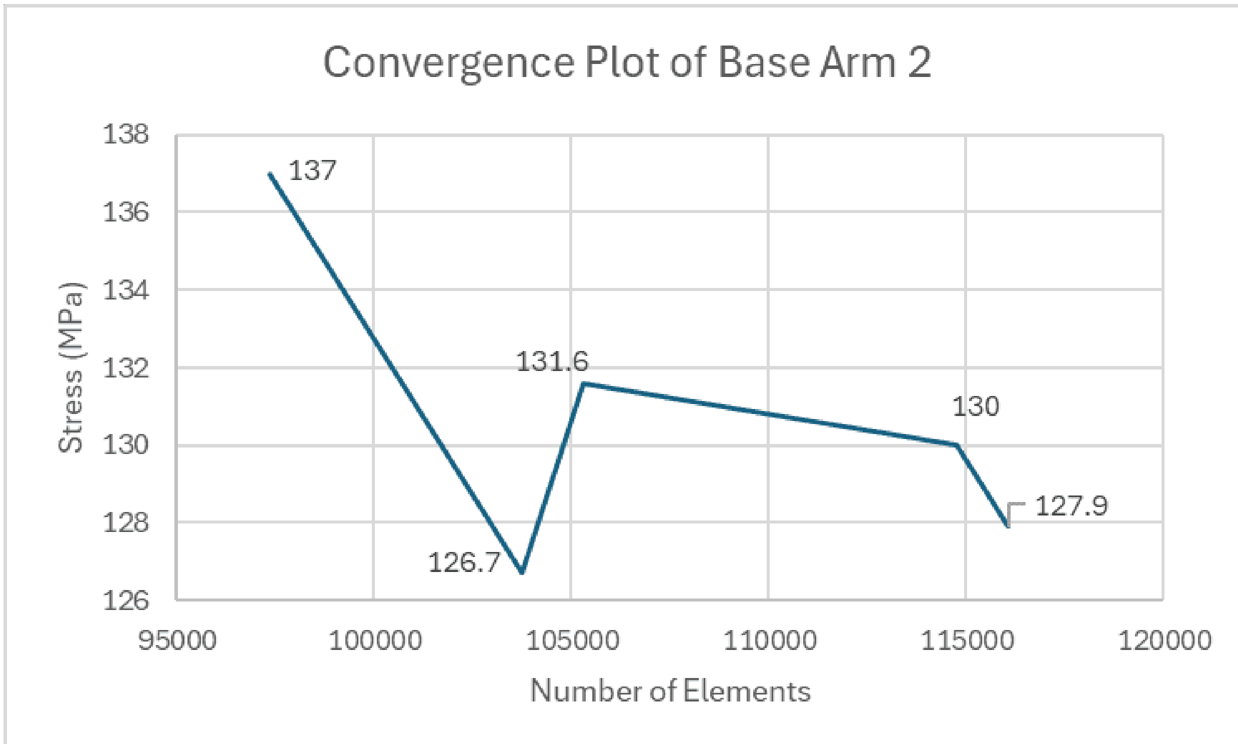

Figure 23: Convergence Plot of Base Arm 2

Figure 23 displays the results of the convergence analysis performed on base arm 2. It was created by comparing the number of elements in the created mesh in our FEA analysis with the calculated maximum stress. This plot is created to ensure that the number of elements in the mesh does not have a significant impact on the maximum calculated stress. To do this, the number of elements is slowly increased, and the stress is recorded. This is done until the next recorded stress is within 5% of the previous.

6.5 GA3 Questions Answered

Max Stress Locations

Pictured below are the locations of the maximum stress experienced by each component. For the arms, we expected that the maximum stress would be near the stress concentrations. For the platform, we predicted the maximum stress would be at the bolt holes where the pump attached to the platform. This is not what we see in the FEA analysis, which predicts the maximum stress location to be the connection with the base arm

Failure Location

Based on our FEA analysis, we would expect the base arms to fail first. In our GA2, we found that the base arm had the lowest factor of safety, meaning it was the closest to failure. The location of the maximum stress on the base arm is consistent with our prediction of where the arm would fail.

Calculated Deflection

The calculated deflection in our FEA analysis was 2.3 mm at the end of the pump platform. This differs from our GA 2 results where we calculated a deflection of 0.067 mm.

Design Changes

While the general design wouldn't be changed, various aspects of it could. For example, the factors of safety in the strut arms are all greater than 1400, meaning there would be room to narrow these pieces in order to cut down costs.

7 Design Changes

No substantial design changes were implemented post-GA2. Based on the simulations which were run during this stage of analysis, the design was finalized and further findings (FEA, etc.) were consistent with previous results.

8 CAD and Fabrication Drawings

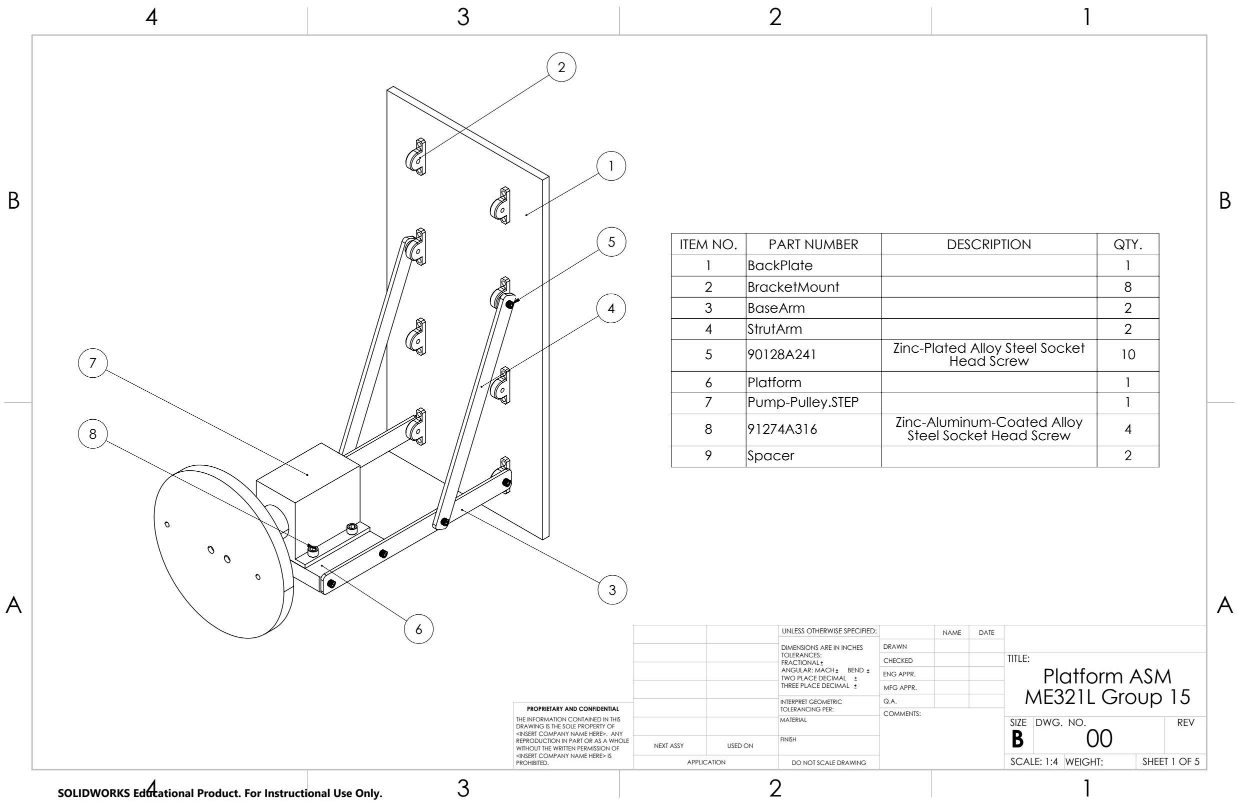

Figure 24: Assembly Drawing and Bill of Materials

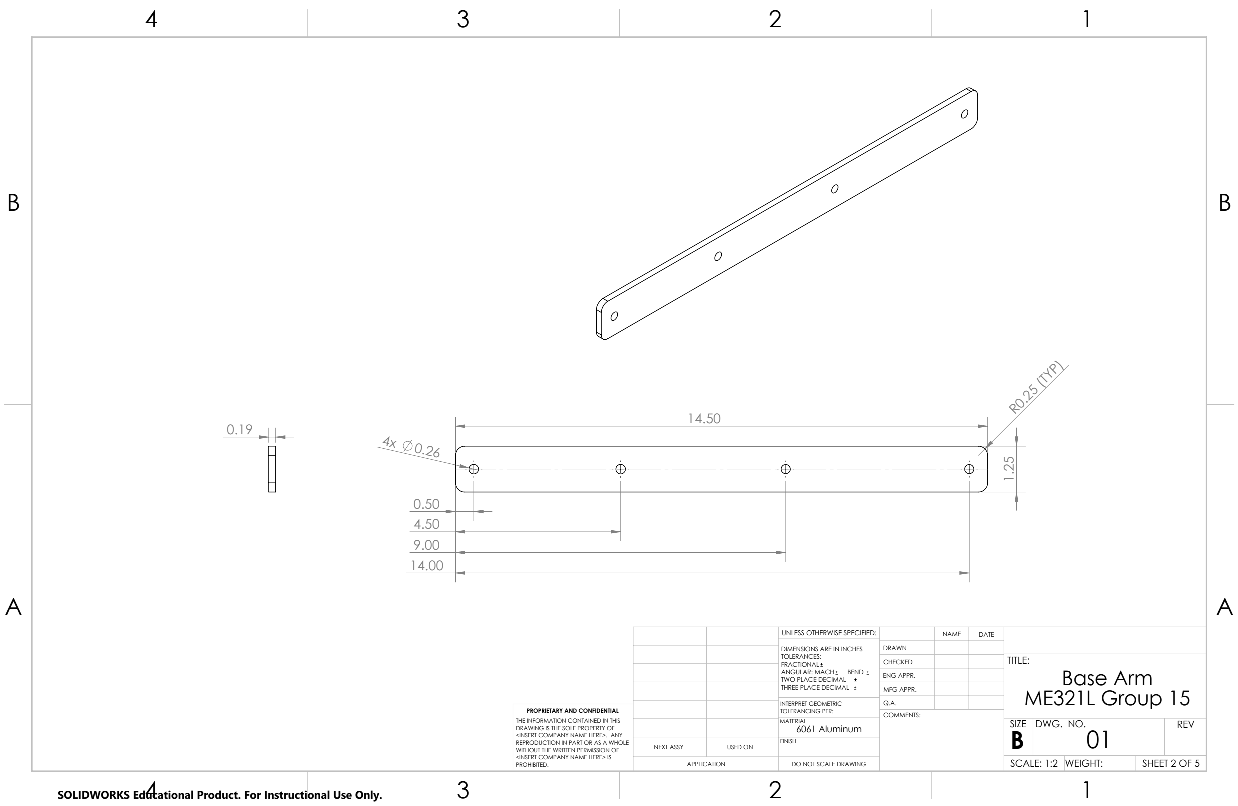

Figure 25: Base Arm Drawing

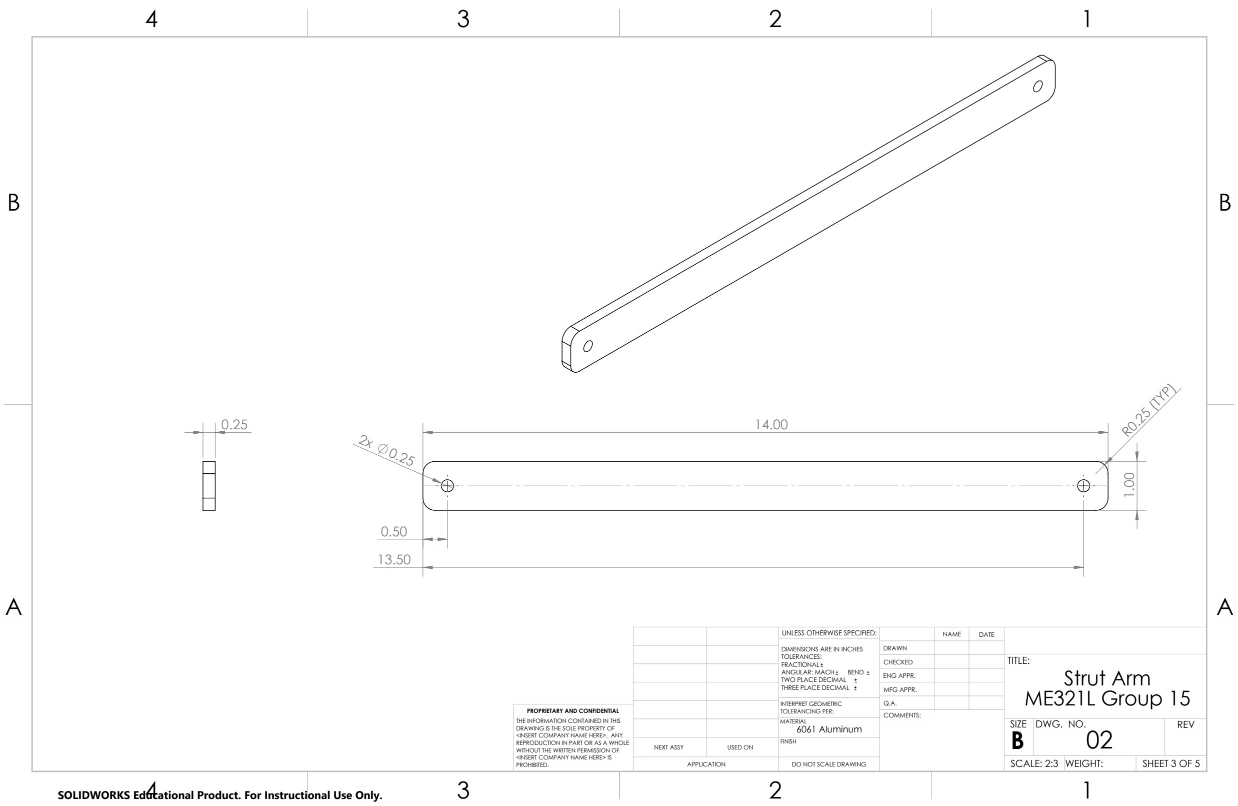

Figure 26: Strut Arm Drawing

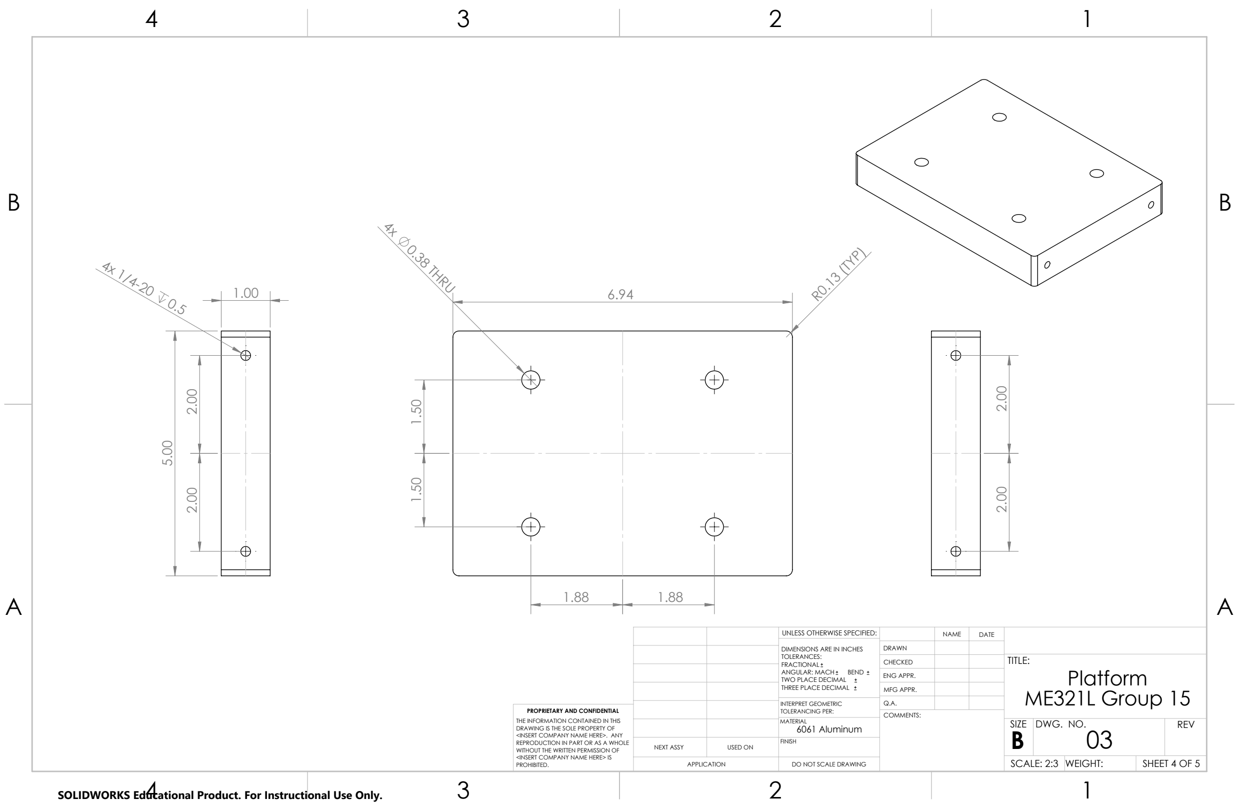

Figure 27: Platform Drawing

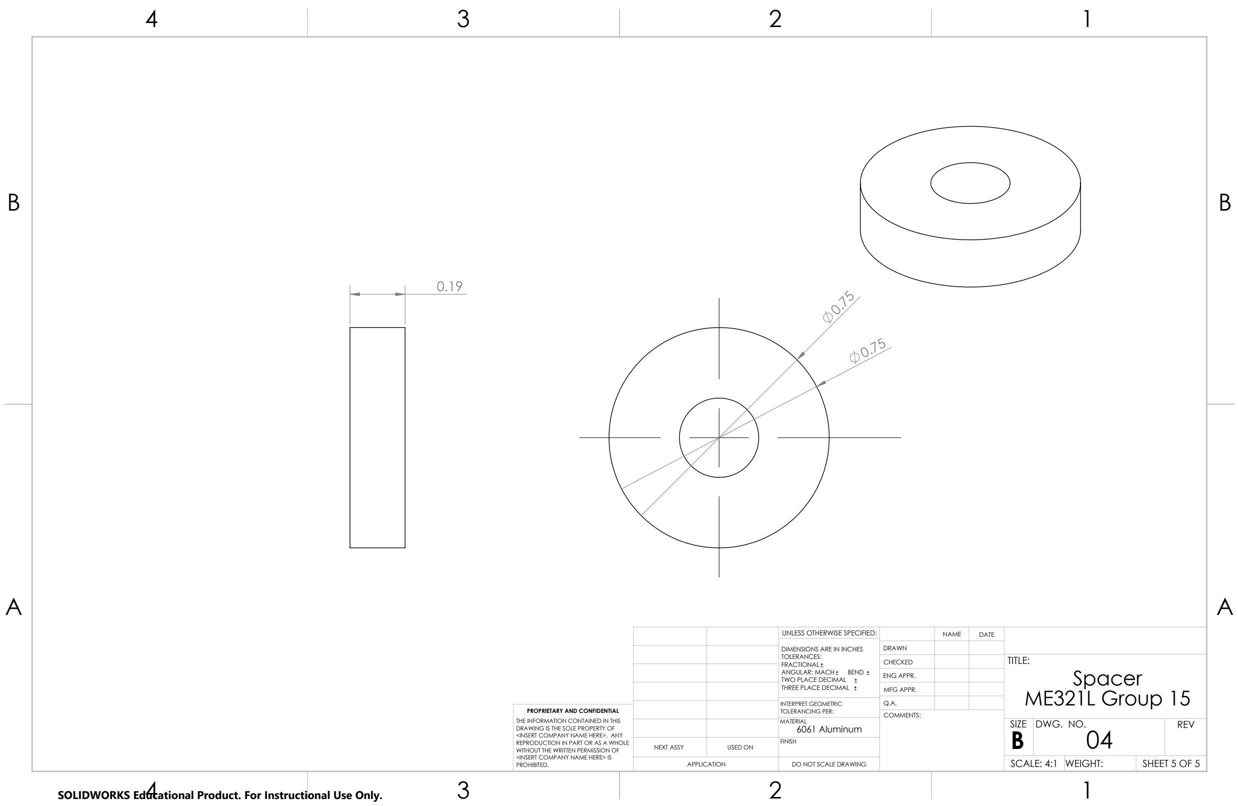

Figure 28: Spacer Drawing

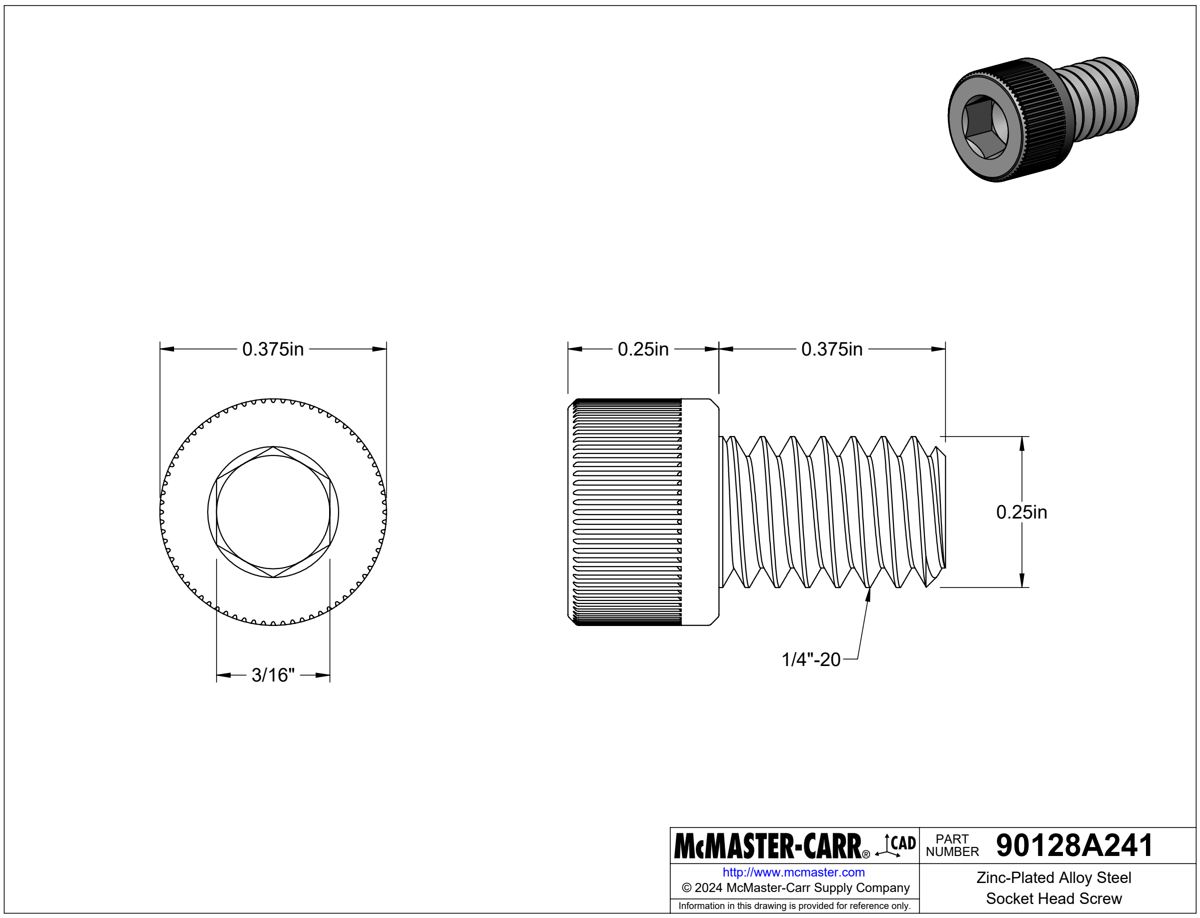

Figure 29: Platform-Arm Screws Drawing

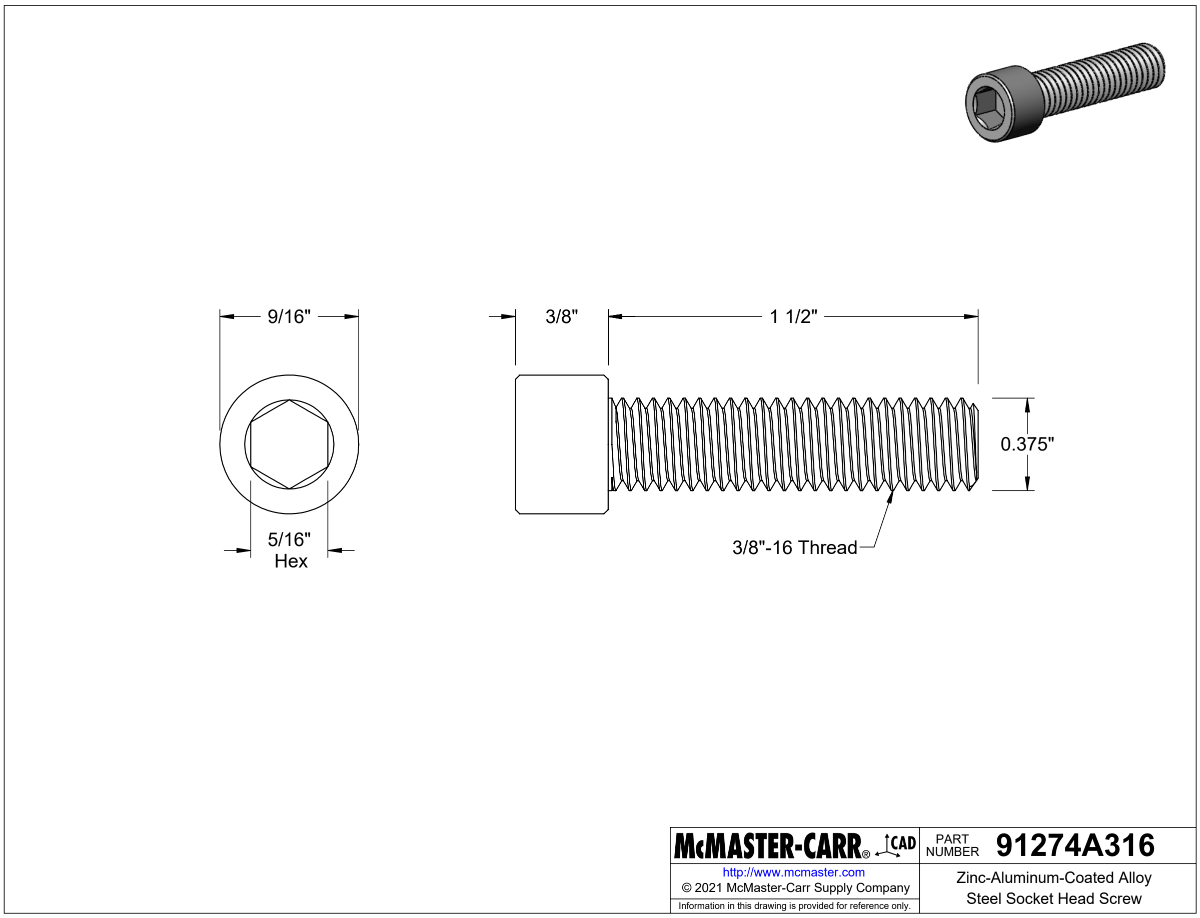

Figure 30: Pump-Platform Screws Drawing

9 Physical Prototype Performance Results

Figure 31: Final Design

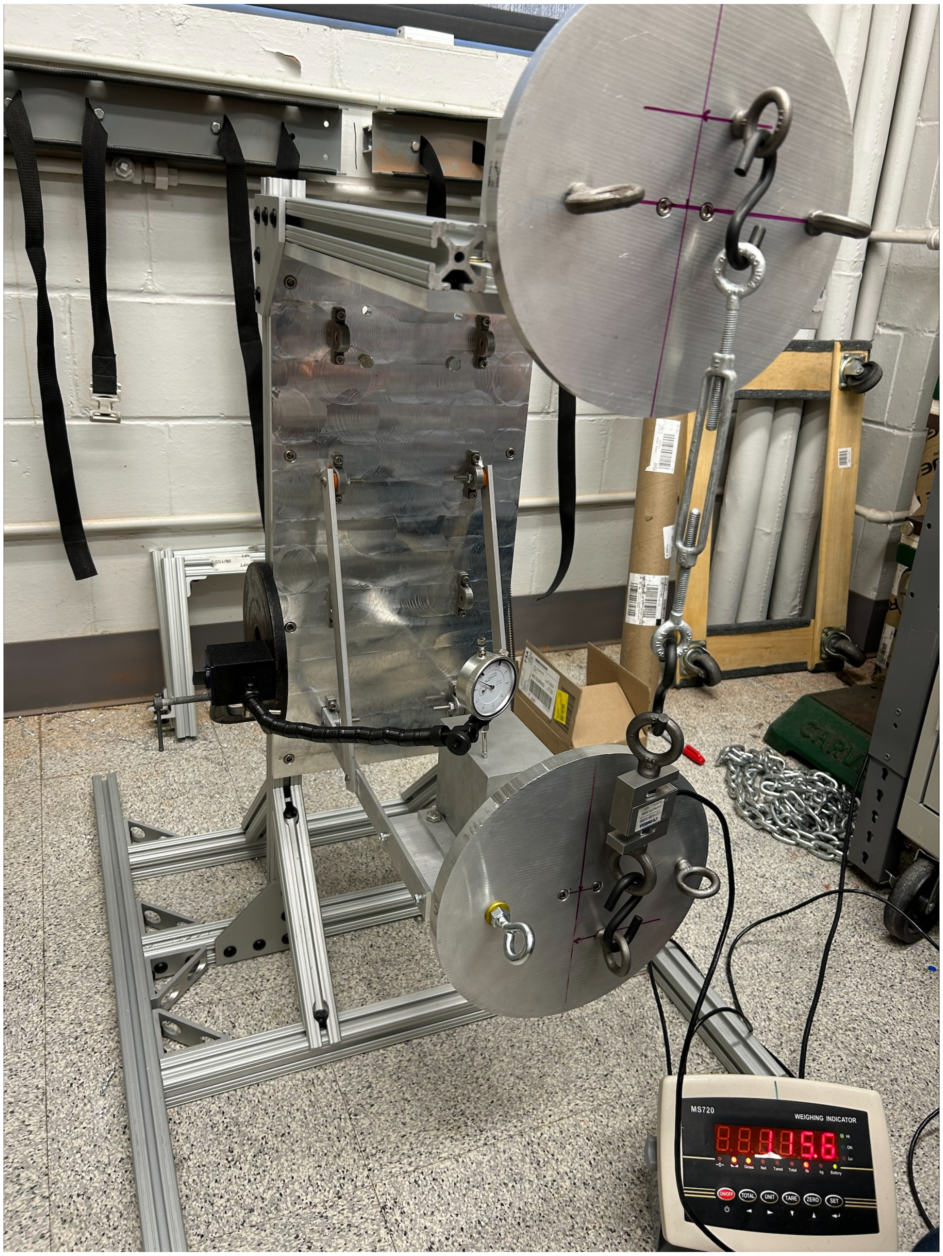

Figure 32: Final Design in Test Frame

Performance Metrics

Weight: **5 lbs 2.2 ounces**

Connected to Frame: **Yes**

Height: **13.5 inches**

Maximum Load: **115 lbs** *(tested up to 200 lbs)*

Maximum Deflection: **0.185 inches**

10 Discussion

i. Did your design successfully interface with the testing setup? If not, how could you modify the design to make it fit?

   (a) We 3D printed spacers to ensure proper interfacing with the test setup.

ii. Did your design withstand the maximum applied load based on the specifications? And did your design pass the deflection criteria?

   (a) Our design withstood the maximum load of 115 lbs. However, the deflection of our design was just under 5 millimeters which did not the pass the required 3 millimeters of allowed deflection.

iii. How did your calculated factor of safety, FEA results, and physical performance compare? Provide explanation for any discrepancies that exist.

   (a) The calculated factors of safety were high in the strut arms (over 1400), and the lowest values in the base arms were above 1.5. However, FEA showed a maximum deflection of 2.3mm, far exceeding the calculated 0.067mm. Physical testing confirmed the actual value around 5mm. This discrepancy likely comes from simplifications in the analytical model (e.g., assuming ideal supports or neglecting minor compliance in materials and fasteners) that FEA and testing captured.

iv. What unexpected challenges did you face and how did you overcome them?

   (a) The main challenge was our design not interfacing with the test frame properly. This was fixed by 3D printing custom spacers, allowing secure attachment and alignment.

11 Conclusion

Over the course of this semester, we successfully designed, analysed, and prototyped a pump support structure to meet the strict specifications provided by PumpCo. Our work involved applying key mechanical engineering principles, such as static equilibria, stress analysis, buckling theory, and beam deflection.

We used a factor of safety of at least 1.5 as a design constraint. We accounted for stress concentrations, selected corrosion-resistant materials suitable for wet environments, and incorporated tolerances. Material selection and geometry optimisation were carried out using both analytical methods and finite element analysis.

We learned the value of designing for manufacturability, anticipating assembly errors, and testing early and often to catch unforeseen issues. Ultimately, this project improved our ability to connect analytical work with physical design under real constraints.

Appendix

A Stress concentration tables

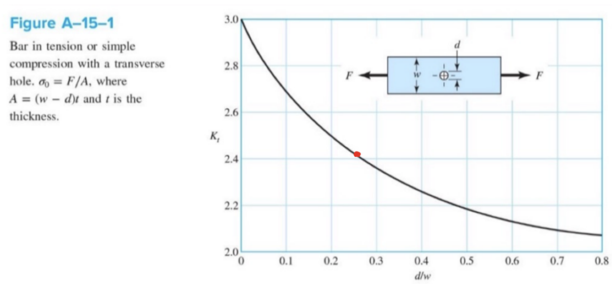

Figure 33: Table A15-1

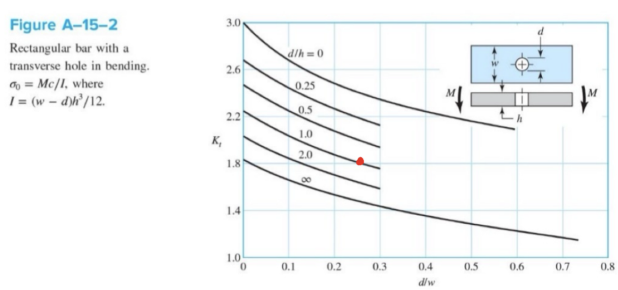

Figure 34: Table A15-2

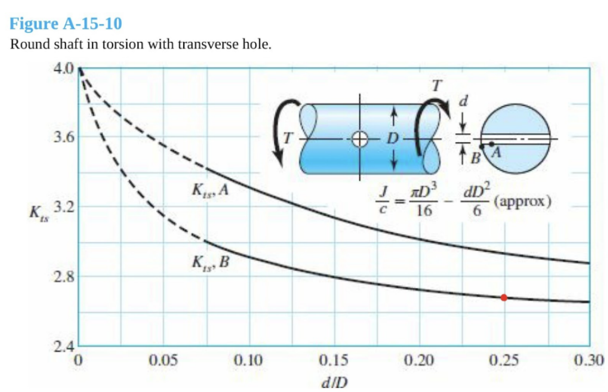

Figure 35: Table A15-10

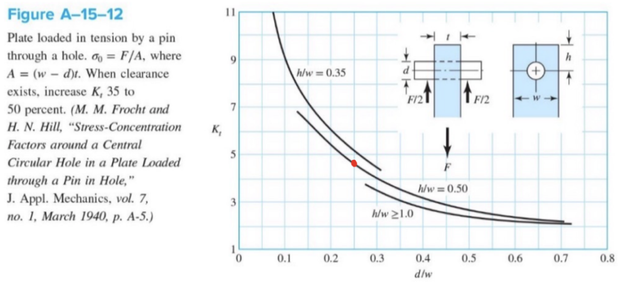

Figure 36: Table A15-12

B Python Code

```
    import numpy as np
import math
import sympy as sp
from matplotlib import pyplot as plt
from sympy import symbols, simplify, solve

def inch_to_m(len):
    return len * 0.0254

def w_to_hp(watts):
    return watts / 745.7

# ### FLUID POWER ###
P_f = symbols('P_f')

# Fluid Values
water_dens_val = 1000 # kg / m^3
volume_flow_rate_val = 150 * 0.00379 / 60 # m^3 / s
g_val = 9.81 # m / s^2
pump_height_val = 60 * 0.3048 # m
water_density, volume_flow_rate, g, pump_height = symbols('water_density volume_flow_rat
fluid_pwr_vals = {water_density: water_dens_val, volume_flow_rate: volume_flow_rate_val,

# Fluid Power Equation
fluid_pwr_eqn = sp.Eq(P_f, water_density * volume_flow_rate * g * pump_height)
fluid_pwr_eqn = fluid_pwr_eqn.subs(fluid_pwr_vals)
fluid_pwr_soln = solve(fluid_pwr_eqn, P_f)
print("\n(1) Fluid Power Solution:")
print("\tFluid Power: ", fluid_pwr_soln[0].evalf(), "W ~= ", w_to_hp(fluid_pwr_soln[0].e

# ### MECHANICAL POWER ###
mech_pwr = symbols('mech_pwr')
```

```
efficiency_val = 0.65
efficiency = symbols('efficiency')
mech_pwr_vals = {efficiency: efficiency_val, P_f: fluid_pwr_soln[0]}

mech_pwr_eqn = sp.Eq(mech_pwr, P_f / efficiency)
mech_pwr_eqn = mech_pwr_eqn.subs(mech_pwr_vals)
mech_pwr_soln = solve(mech_pwr_eqn, mech_pwr)

print("\n(2) Mechanical Power Solution:")
print("\tMechanical Power: ", mech_pwr_soln[0].evalf(), "W ~= ", w_to_hp(mech_pwr_soln[0

# ### Pulley Reaction Force ###
# System set of pump/pulley forces acting at the end of the platform for single point as
T2_val = 200 # N
d_pulley = 0.254 # m
rot_freq = 188.5 # rad/sec
torque_req = mech_pwr_soln[0].evalf() / rot_freq
T1_val = T2_val + (2 * torque_req / d_pulley)
R_pz_val = T1_val + T2_val
R_px_val = 0
M_px_val = (T1_val * 0.102) + (T2_val * 0.102)
M_py_val = -torque_req
W_pump_val = 55.6 # N

print("\n(3) Pulley Reaction Forces:")
print("\tT1: ", T1_val, "N")
print("\tT2: ", T2_val, "N")
print("\tR_pz: ", R_pz_val, "N")
print("\tR_px: ", R_px_val, "N")
print("\tM_px: ", M_px_val, "Nm")
print("\tM_py: ", M_py_val, "Nm")
print("\tW_pump: ", W_pump_val, "N")


T1, T2 = symbols('T1, T2')
R_px, R_pz = symbols('R_px, R_pz')
M_px, M_py = symbols('M_px M_py')
W = symbols('W')

# ### SOLVING FOR BOLT CONNECTIONS ON PLATFORM ###
# Bolt Symbols
R_1x, R_1y, R_1z = symbols('R_1x R_1y R_1z')
R_2x, R_2y, R_2z = symbols('R_2x R_2y R_2z')
R_3x, R_3y, R_3z = symbols('R_3x R_3y R_3z')
```

```
R_4x, R_4y, R_4z = symbols('R_4x R_4y R_4z')

consts = {T1: T1_val, T2: T2_val, R_px: R_px_val, R_pz: R_pz_val, M_px: M_px_val, M_py:

# Z forces
eq_z = sp.Eq(R_pz + R_1z + R_2z + R_3z + R_4z - W, 0)

# Y Forces
y_zeros = {R_1y: 0, R_2y: 0, R_3y: 0, R_4y: 0}

# X Forces
x_zeros = {R_1x: 0, R_2x: 0, R_3x: 0, R_4x: 0}

# X Moment
eq_mx = sp.Eq(M_px + (R_1z * 0.0254) + (R_3z * 0.0254) + (R_2z * 0.1016) + (R_4z * 0.101

# Y Moment - 0.048 m distance from y for R components, no W
eq_my = sp.Eq(M_py + (R_1z * 0.048) + (R_2z * 0.048) - (R_3z * 0.048) - (R_4z * 0.048),

# Torsion neutralization
eq_torsion_neutr = sp.Eq(R_2z - R_4z, 0)

# Solve for unknowns
consts.update(y_zeros)
consts.update(x_zeros)
eq_z = eq_z.subs(consts)
eq_mx = eq_mx.subs(consts)
eq_my = eq_my.subs(consts)
eq_torsion_neutr = eq_torsion_neutr.subs(consts)

# Solve for bolt forces
sol_bolt = solve([eq_z, eq_mx, eq_my, eq_torsion_neutr], (R_1z, R_2z, R_3z, R_4z))

print("\n(3b) Bolt Reaction Forces:")
for key in sol_bolt:
    print(f'\t{key}: {sol_bolt[key].evalf()}')



# ### SOLVING FOR PLATFORM CONNECTIONS ###
# Platform Symbols
W_plat = symbols('W_plat')
S_1y, S_1z = symbols('S_1y S_1z')
S_2y, S_2z = symbols('S_2y S_2z')
F_t1, F_t2 = symbols('F_t1 F_t2')
```

```
w, l, a, b = symbols('w l a b')

# Parameters / Constants:
# w - rough based off geometry, l - rough based off geometry
# a, b - should do a sweep, temp for now
a_val = 0.3048
b_val = 0.2032
plat_thickness = 0.00635
plat_w_val = 0.1524
plat_l_val = 0.37
plat_dims = {w: plat_w_val, l: plat_l_val}
support_dims = {a: a_val, b: b_val}

# Plat temporary weight calcs
Aluminum_density = 2710 # kg / m3
g = 9.81 # m/s^2
w_plat_val = plat_w_val * plat_l_val * plat_thickness * Aluminum_density * g
plat_weight = {W_plat: w_plat_val}

# Support components
supp_z_comp = a_val / math.sqrt(a_val**2 + b_val**2)
supp_y_comp = b_val / math.sqrt(a_val**2 + b_val**2)

# Platform equations
plat_eq_z = sp.Eq(R_pz - W - W_plat + S_1z + S_2z + ((F_t1 + F_t2) * (supp_z_comp)), 0)
plat_eq_y = sp.Eq(S_1y + S_2y + ((F_t1 + F_t2) * (supp_y_comp)), 0)
plat_eq_mx = sp.Eq(-M_px - (W * 0.0635) - (W_plat * l / 2) + ((F_t1 + F_t2) * (supp_z_co
plat_eq_my = sp.Eq(M_py + ((F_t1 - F_t2) * (supp_z_comp) * (w / 2)) + ((S_1z - S_2z) * (
plat_eq_tor_neutr1 = sp.Eq(S_1z - S_2z, 0)
plat_eq_tor_neutr2 = sp.Eq(S_1y - S_2y, 0)

# Solve for platform connections
consts.update(plat_dims)
consts.update(support_dims)
consts.update(plat_weight)

plat_eq_z = plat_eq_z.subs(consts)
plat_eq_y = plat_eq_y.subs(consts)
plat_eq_mx = plat_eq_mx.subs(consts)
plat_eq_my = plat_eq_my.subs(consts)
plat_eq_tor_neutr1 = plat_eq_tor_neutr1.subs(consts)
plat_eq_tor_neutr2 = plat_eq_tor_neutr2.subs(consts)
solve_conn = solve([plat_eq_z, plat_eq_y, plat_eq_mx, plat_eq_my, plat_eq_tor_neutr1, pl

print("\n(4/5) Platform & Support Reaction Forces:")
```

```
for key in solve_conn:
    print(f'\t{key}: {solve_conn[key].evalf()}')

print("\nRelevant Constants:")
for key in consts:
    print(f'\t{key}: {consts[key]}')

F_t2_val = solve_conn[F_t2].evalf()
F_t1_val = solve_conn[F_t1].evalf()
S_1y_val = solve_conn[S_1y].evalf()
S_2y_val = solve_conn[S_2y].evalf()
S_1z_val = solve_conn[S_1z].evalf()
S_2z_val = solve_conn[S_2z].evalf()

R_1z_val = sol_bolt[R_1z].evalf()
R_2z_val = sol_bolt[R_2z].evalf()
R_3z_val = sol_bolt[R_3z].evalf()
R_4z_val = sol_bolt[R_4z].evalf()


# ### PLOTTING VM DIAGRAMS ###

# Platform Shear and Moment Diagrams
y = np.linspace(0, plat_l_val * 1.01, 1000)
shear = []
moment = []
for pos in y:
    v = consts[R_pz]
    m = consts[M_px]
    m += consts[R_pz] * pos
    if pos >= 0.0635:
        v -= consts[W]
        m -= consts[W] * (pos - 0.0635)

    if pos >= (plat_l_val / 2):
        v -= consts[W_plat]
        m -= consts[W_plat] * (pos - (plat_l_val / 2))

    if pos >= (plat_l_val - b_val):
        v += solve_conn[F_t1] * supp_z_comp
        v += solve_conn[F_t2] * supp_z_comp
        m += solve_conn[F_t1] * supp_z_comp * (pos - (plat_l_val - b_val))
        m += solve_conn[F_t2] * supp_z_comp * (pos - (plat_l_val - b_val))

    if pos >= (plat_l_val):
```

```
        v += solve_conn[S_1z]
        v += solve_conn[S_2z]
        m += solve_conn[S_1z] * (pos - plat_l_val)
        m += solve_conn[S_2z] * (pos - plat_l_val)
    shear.append(v)
    moment.append(m)

# Plot Shear and Moment
fig, axs = plt.subplots(2)
fig.suptitle('Pump Shear and Moment Diagrams')
axs[0].plot(y, shear)
axs[0].set_ylabel('Shear Force (N)')
axs[0].set_xlabel('Position (m)')
axs[0].set_title('Shear Force Diagram')
axs[1].plot(y, moment)
axs[1].set_ylabel('Moment (Nm)')
axs[1].set_xlabel('Position (m)')
axs[1].set_title('Bending Moment Diagram')
fig.tight_layout(pad=2.0)

for ax in axs:
    ax.grid(True)
    ax.axhline(y=0, color='k')

plt.show()


#### STRUCTURAL DESIGN ANALYSIS #####

# Base Arm Parameters
base_arm_pm1_y = inch_to_m(0.5) # m
base_arm_pm2_y = inch_to_m(4.5) # m
base_arm_sh_y = inch_to_m(8.997) # m
base_arm_wh_y = inch_to_m(13.997) # m
std_hole_diam = inch_to_m(0.25) # m

base_arm_pm1_Kt_b = 1.8
base_arm_pm2_Kt_b = 1.8
base_arm_sh_Kt_b = 1.8
base_arm_wh_Kt_b = 1.8

base_arm_pm1_Kt_axial = 2.4
base_arm_pm2_Kt_axial = 2.4
base_arm_sh_Kt_axial = 2.4
base_arm_wh_Kt_axial = 2.4
```

```
base_arm_pm1_Kt_tor = 2.7
base_arm_pm2_Kt_tor = 2.7
base_arm_sh_Kt_tor = 2.7
base_arm_wh_Kt_tor = 2.7



#### BASE ARM ANALYSIS - Base Arm 2 ####
base_arm_pm1_M, base_arm_pm2_M, base_arm_sh_M, base_arm_wh_M = symbols('base_arm_pm1_M b
base_arm_pm1_shear, base_arm_pm2_shear, base_arm_sh_shear, base_arm_wh_shear = symbols('

for i in range(len(y)):
    if y[i] <= base_arm_pm1_y:
        base_arm_pm1_M = moment[i] / 2
        base_arm_pm1_shear = shear[i] / 2
    if y[i] <= base_arm_pm2_y:
        base_arm_pm2_M = moment[i] / 2
        base_arm_pm2_shear = shear[i] / 2
    if y[i] <= base_arm_sh_y:
        base_arm_sh_M = moment[i] / 2
        base_arm_sh_shear = shear[i] / 2
    if y[i] <= base_arm_wh_y:
        base_arm_wh_M = moment[i] / 2
        base_arm_wh_shear = shear[i] / 2

base_arm_t = inch_to_m(0.375) # m
base_arm_w = inch_to_m(1.25) # m
base_arm_l = inch_to_m(14.497) # m

base_arm_I = (base_arm_t * base_arm_w**3) / 12 # m^4
base_arm_y = base_arm_w / 2 # m

base_arm_pm1_max_bending_nom = (base_arm_pm1_M * base_arm_y) / base_arm_I
base_arm_pm2_max_bending_nom = (base_arm_pm2_M * base_arm_y) / base_arm_I
base_arm_sh_max_bending_nom = (base_arm_sh_M * base_arm_y) / base_arm_I
base_arm_wh_max_bending_nom = (base_arm_wh_M * base_arm_y) / base_arm_I

base_arm_pm1_max_bending_conc = base_arm_pm1_max_bending_nom * base_arm_pm1_Kt_b
base_arm_pm2_max_bending_conc = base_arm_pm2_max_bending_nom * base_arm_pm2_Kt_b
base_arm_sh_max_bending_conc = base_arm_sh_max_bending_nom * base_arm_sh_Kt_b
base_arm_wh_max_bending_conc = base_arm_wh_max_bending_nom * base_arm_wh_Kt_b

print("\nBase Arm 2 Max Bending Stress (Nominal):")
print(f'\tPM1: {base_arm_pm1_max_bending_nom}')
```

```
print(f'\tPM2: {base_arm_pm2_max_bending_nom}')
print(f'\tSH: {base_arm_sh_max_bending_nom}')
print(f'\tWH: {base_arm_wh_max_bending_nom}')

print("\nBase Arm 2 Max Bending Stress (Concentrated):")
print(f'\tPM1: {base_arm_pm1_max_bending_conc}')
print(f'\tPM2: {base_arm_pm2_max_bending_conc}')
print(f'\tSH: {base_arm_sh_max_bending_conc}')
print(f'\tWH: {base_arm_wh_max_bending_conc}')

# Axial Forces
base_arm_pm1_axial_force = 0
base_arm_pm2_axial_force = 0
base_arm_sh_axial_force = abs(F_t2_val * b_val / math.sqrt(a_val**2 + b_val**2))
base_arm_wh_axial_force = abs(S_2y_val)

base_arm_pm1_axial_stress_nom = base_arm_pm1_axial_force / (base_arm_t * (base_arm_w - s
base_arm_pm2_axial_stress_nom = base_arm_pm2_axial_force / (base_arm_t * (base_arm_w - s
base_arm_sh_axial_stress_nom = base_arm_sh_axial_force / (base_arm_t * (base_arm_w - std
base_arm_wh_axial_stress_nom = base_arm_wh_axial_force / (base_arm_t * (base_arm_w - std

base_arm_pm1_axial_stress_conc = base_arm_pm1_axial_stress_nom * base_arm_pm1_Kt_axial
base_arm_pm2_axial_stress_conc = base_arm_pm2_axial_stress_nom * base_arm_pm2_Kt_axial
base_arm_sh_axial_stress_conc = base_arm_sh_axial_stress_nom * base_arm_sh_Kt_axial
base_arm_wh_axial_stress_conc = base_arm_wh_axial_stress_nom * base_arm_wh_Kt_axial

print("\nBase Arm Axial Stress (Nominal):")
print(f'\tPM1: {base_arm_pm1_axial_stress_nom}')
print(f'\tPM2: {base_arm_pm2_axial_stress_nom}')
print(f'\tSH: {base_arm_sh_axial_stress_nom}')
print(f'\tWH: {base_arm_wh_axial_stress_nom}')

print("\nBase Arm Axial Stress (Concentrated):")
print(f'\tPM1: {base_arm_pm1_axial_stress_conc}')
print(f'\tPM2: {base_arm_pm2_axial_stress_conc}')
print(f'\tSH: {base_arm_sh_axial_stress_conc}')
print(f'\tWH: {base_arm_wh_axial_stress_conc}')

# Combining axial and bending stresses
base_arm_pm1_max_stress = abs(base_arm_pm1_axial_stress_conc) + abs(base_arm_pm1_max_ben
base_arm_pm2_max_stress = abs(base_arm_pm2_axial_stress_conc) + abs(base_arm_pm2_max_ben
base_arm_sh_max_stress = abs(base_arm_sh_axial_stress_conc) + abs(base_arm_sh_max_bendin
base_arm_wh_max_stress = abs(base_arm_wh_axial_stress_conc) + abs(base_arm_wh_max_bendin

print("\nBase Arm Max Stress (Concentrated):")
```

```
print(f'\tPM1: {base_arm_pm1_max_stress}')
print(f'\tPM2: {base_arm_pm2_max_stress}')
print(f'\tSH: {base_arm_sh_max_stress}')
print(f'\tWH: {base_arm_wh_max_stress}')

# Torsion
base_arm_tor_alpha = (1 - (0.063 * base_arm_t / base_arm_w) + (0.052 * ((base_arm_t / ba
print(f'Base Arm Torsion Alpha: {base_arm_tor_alpha}')
base_arm_pm1_T = (-(S_1y_val + (F_t2_val * (a_val / math.sqrt(a_val**2 + b_val**2)))) *
base_arm_pm2_T = base_arm_pm1_T
base_arm_sh_T = 0
base_arm_wh_T = 0

base_arm_pm1_tor_nom = base_arm_pm1_T / (base_arm_tor_alpha * base_arm_w * (base_arm_t *
base_arm_pm2_tor_nom = base_arm_pm2_T / (base_arm_tor_alpha * base_arm_w * (base_arm_t *
base_arm_sh_tor_nom = base_arm_sh_T / (base_arm_tor_alpha * base_arm_w * (base_arm_t **
base_arm_wh_tor_nom = base_arm_wh_T / (base_arm_tor_alpha * base_arm_w * (base_arm_t **

base_arm_pm1_tor_conc = base_arm_pm1_tor_nom * base_arm_pm1_Kt_tor
base_arm_pm2_tor_conc = base_arm_pm2_tor_nom * base_arm_pm2_Kt_tor
base_arm_sh_tor_conc = base_arm_sh_tor_nom * base_arm_sh_Kt_tor
base_arm_wh_tor_conc = base_arm_wh_tor_nom * base_arm_wh_Kt_tor

print("\nBase Arm Torsion (Nominal):")
print(f'\tPM1: {base_arm_pm1_tor_nom}')
print(f'\tPM2: {base_arm_pm2_tor_nom}')
print(f'\tSH: {base_arm_sh_tor_nom}')
print(f'\tWH: {base_arm_wh_tor_nom}')

print("\nBase Arm Torsion (Concentrated):")
print(f'\tPM1: {base_arm_pm1_tor_conc}')
print(f'\tPM2: {base_arm_pm2_tor_conc}')
print(f'\tSH: {base_arm_sh_tor_conc}')
print(f'\tWH: {base_arm_wh_tor_conc}')

base_arm_pm1_max_tor = abs(base_arm_pm1_tor_conc)
base_arm_pm2_max_tor = abs(base_arm_pm2_tor_conc)
base_arm_sh_max_tor = abs(base_arm_sh_tor_conc)
base_arm_wh_max_tor = abs(base_arm_wh_tor_conc)

# Principal and Shear stresses
base_arm_pm1_principal_stress = [0, (base_arm_pm1_max_stress/2) + math.sqrt(((base_arm_p
base_arm_pm2_principal_stress = [0, (base_arm_pm2_max_stress/2) + math.sqrt(((base_arm_p
base_arm_sh_principal_stress = [0, (base_arm_sh_max_stress/2) + math.sqrt(((base_arm_sh_
base_arm_wh_principal_stress = [0, (base_arm_wh_max_stress/2) + math.sqrt(((base_arm_wh_
```

```
base_arm_pm1_principal_stress.sort()
base_arm_pm2_principal_stress.sort()
base_arm_sh_principal_stress.sort()
base_arm_wh_principal_stress.sort()

base_arm_pm1_max_shear = 0.5 * (base_arm_pm1_principal_stress[2] - base_arm_pm1_principa
base_arm_pm2_max_shear = 0.5 * (base_arm_pm2_principal_stress[2] - base_arm_pm2_principa
base_arm_sh_max_shear = 0.5 * (base_arm_sh_principal_stress[2] - base_arm_sh_principal_s
base_arm_wh_max_shear = 0.5 * (base_arm_wh_principal_stress[2] - base_arm_wh_principal_s

print("\nBase Arm Principal and Shear Stresses:")
print(f'\tPM1: {base_arm_pm1_principal_stress}')
print(f'\tPM2: {base_arm_pm2_principal_stress}')
print(f'\tSH: {base_arm_sh_principal_stress}')
print(f'\tWH: {base_arm_wh_principal_stress}')

print("\nBase Arm Max Shear:")
print(f'\tPM1: {base_arm_pm1_max_shear}')
print(f'\tPM2: {base_arm_pm2_max_shear}')
print(f'\tSH: {base_arm_sh_max_shear}')
print(f'\tWH: {base_arm_wh_max_shear}')

base_arm_mat_S_uc = 310 * 10**6 # tbd - for 6061 atm
base_arm_mat_S_ut = 310 * 10**6 # tbd - for 6061 atm
base_arm_mat_S_y = 276 * 10**6 # tbd - for 6061 atm
base_arm_mat_E = 70 * 10**9 # tbd - for 6061 atm
base_arm_mat_G = 28 * 10**9 # tbd - for 6061 atm

# ductile failure analysis
base_arm_pm1_duct_eq_stress = math.sqrt(((base_arm_pm1_principal_stress[2] - base_arm_pm
base_arm_pm2_duct_eq_stress = math.sqrt(((base_arm_pm2_principal_stress[2] - base_arm_pm
base_arm_sh_duct_eq_stress = math.sqrt(((base_arm_sh_principal_stress[2] - base_arm_sh_p
base_arm_wh_duct_eq_stress = math.sqrt(((base_arm_wh_principal_stress[2] - base_arm_wh_p

base_arm_pm1_duct_fos = base_arm_mat_S_y / base_arm_pm1_duct_eq_stress
base_arm_pm2_duct_fos = base_arm_mat_S_y / base_arm_pm2_duct_eq_stress
base_arm_sh_duct_fos = base_arm_mat_S_y / base_arm_sh_duct_eq_stress
base_arm_wh_duct_fos = base_arm_mat_S_y / base_arm_wh_duct_eq_stress
print("\nBase Arm Ductile Failure Factors of Safety:")
print(f'\tPM1: {base_arm_pm1_duct_fos}')
print(f'\tPM2: {base_arm_pm2_duct_fos}')
print(f'\tSH: {base_arm_sh_duct_fos}')
print(f'\tWH: {base_arm_wh_duct_fos}')
```

```
# brittle failure analysis
base_arm_pm1_brittle_fos_comp = base_arm_mat_S_uc / base_arm_pm1_principal_stress[2]
base_arm_pm2_brittle_fos_comp = base_arm_mat_S_uc / base_arm_pm2_principal_stress[2]
base_arm_sh_brittle_fos_comp = base_arm_mat_S_uc / base_arm_sh_principal_stress[2]
base_arm_wh_brittle_fos_comp = base_arm_mat_S_uc / base_arm_wh_principal_stress[2]

base_arm_pm1_brittle_fos_tens = base_arm_mat_S_ut / base_arm_pm1_principal_stress[2]
base_arm_pm2_brittle_fos_tens = base_arm_mat_S_ut / base_arm_pm2_principal_stress[2]
base_arm_sh_brittle_fos_tens = base_arm_mat_S_ut / base_arm_sh_principal_stress[2]
base_arm_wh_brittle_fos_tens = base_arm_mat_S_ut / base_arm_wh_principal_stress[2]

base_arm_pm1_brittle_fos = min(base_arm_pm1_brittle_fos_comp, base_arm_pm1_brittle_fos_t
base_arm_pm2_brittle_fos = min(base_arm_pm2_brittle_fos_comp, base_arm_pm2_brittle_fos_t
base_arm_sh_brittle_fos = min(base_arm_sh_brittle_fos_comp, base_arm_sh_brittle_fos_tens
base_arm_wh_brittle_fos = min(base_arm_wh_brittle_fos_comp, base_arm_wh_brittle_fos_tens

print("\nBase Arm Brittle Failure Factors of Safety:")
print(f'\tPM1: {base_arm_pm1_brittle_fos}')
print(f'\tPM2: {base_arm_pm2_brittle_fos}')
print(f'\tSH: {base_arm_sh_brittle_fos}')
print(f'\tWH: {base_arm_wh_brittle_fos}')

print("\nMaterial Selection Required Yield Strength:")
des_fos = 1.5
print("\t", des_fos * max(base_arm_pm1_duct_eq_stress, base_arm_pm2_duct_eq_stress, base



#### STRUT ARM ANALYSIS ####
strut_arm_comp_F = abs(F_t2_val) # N

strut_arm_comp_Kt = 4.8

strut_arm_w = inch_to_m(1) # m
strut_arm_t = inch_to_m(0.25) # m
strut_arm_l = inch_to_m(13) # m
strut_arm_mat_E = 70 * 10**9# tbd - for 6061 atm

strut_arm_mat_S_y = 310 * 10**6 # tbd - for 6061 atm
strut_arm_mat_S_ut = 310 * 10**6 # tbd - for 6061 atm
strut_arm_mat_S_uc = 276 * 10**6 # tbd - for 6061 atm

strut_arm_I_cs = strut_arm_w * (strut_arm_t**3) / 12 # m^4
strut_arm_buckling_load = math.pi**2 * strut_arm_mat_E * strut_arm_I_cs / (strut_arm_l**
strut_arm_fos_buckling = strut_arm_buckling_load / strut_arm_comp_F
```

```
print("\nStrut Arm Buckling Load: ", strut_arm_buckling_load, "N")
print("Strut Arm Buckling FOS: ", strut_arm_fos_buckling)

# Axial Force
strut_arm_axial_stress_nom = -strut_arm_comp_F / (strut_arm_w - std_hole_diam) # Pa
strut_arm_axial_stress_conc = strut_arm_axial_stress_nom * strut_arm_comp_Kt # Pa

strut_arm_principal_stress = [0, 0, strut_arm_axial_stress_conc] # Pa
strut_arm_principal_stress.sort()

strut_arm_max_shear = 0.5 * (strut_arm_principal_stress[2] - strut_arm_principal_stress[

print("\nStrut Arm Shear Stress: ", strut_arm_max_shear, "Pa")

# Ductile Failure Analysis
strut_arm_duct_fos = abs(strut_arm_mat_S_y / strut_arm_principal_stress[0])

print("\nStrut Arm Ductile Failure Factors of Safety:", strut_arm_duct_fos)

# Brittle Failure Analysis
strut_arm_brittle_fos_comp = abs(strut_arm_mat_S_uc / strut_arm_principal_stress[0])
print("\nStrut Arm Brittle Failure Factors of Safety:", strut_arm_brittle_fos_comp)

print("\nMaterial Selection Required Yield Strength:")
des_fos = 1.5
print("\t", des_fos * (strut_arm_duct_fos))

print("\nMaterial Selection Required Young's Modulus (>=):")
print("\t", des_fos * S_2y_val * strut_arm_l**2 / (math.pi**2 * strut_arm_I_cs))



#### PLATFORM ANALYSIS ####
plat_t_val = inch_to_m(1) # m

plat_mat_E = 70 * 10**9 # tbd - for 6061 atm
plat_mat_S_y = 310 * 10**6 # tbd - for 6061 atm
plat_mat_S_ut = 310 * 10**6 # tbd - for 6061 atm
plat_mat_S_uc = 276 * 10**6 # tbd - for 6061 atm

Kt_y = 2.45
Kt_x = 2.45

# Bending - y
plat_mom_y = 0.5 * plat_w_val * (R_1z_val + R_2z_val + R_3z_val + R_4z_val)
```

```
plat_I_y = plat_l_val * (plat_t_val**3) / 12 # m^4
plat_b_y_stress_nom = plat_mom_y * (plat_t_val / 2) / plat_I_y # Pa
plat_b_y_stress_conc = plat_b_y_stress_nom * Kt_y

# Bending - x
plat_mom_x = 0.5 * inch_to_m(4) * (R_1z_val + R_2z_val + R_3z_val + R_4z_val)
plat_I_x = plat_w_val * (plat_t_val**3) / 12 # m^4
plat_b_x_stress_nom = plat_mom_x * (plat_t_val / 2) / plat_I_x # Pa
plat_b_x_stress_conc = plat_b_x_stress_nom * Kt_x

# Axial
plat_axial_stress_nom = -(R_1z_val + R_2z_val + R_3z_val + R_4z_val) / (plat_w_val * pla

plat_max_stress = abs(plat_b_y_stress_conc) + abs(plat_b_x_stress_conc) + abs(plat_axial
print("\nPlatform Max Stress: ", plat_max_stress, "Pa", " (b_y: ", plat_b_y_stress_conc,

plat_principal_stress = [0, 0, plat_max_stress] # Pa
plat_principal_stress.sort()

plat_max_shear = 0.5 * (plat_principal_stress[2] - plat_principal_stress[0]) # Pa

print("\nPlatform Shear Stress: ", plat_max_shear, "Pa")


# Ductile Failure Analysis
plat_duct_fos = abs(plat_mat_S_y / plat_principal_stress[2])

print("\nPlatform Ductile Failure Factors of Safety:", plat_duct_fos)

# Brittle Failure Analysis
plat_brittle_fos_comp = abs(plat_mat_S_uc / plat_principal_stress[2])
print("\nPlatform Brittle Failure Factors of Safety:", plat_brittle_fos_comp)

print("\nMaterial Selection Required Yield Strength:")
des_fos = 1.5
print("\t", des_fos * (plat_duct_fos))



### Pin Analysis ###
pin4_mat_G = 264 * 10**6 # Carbon Steel
pin4_shear = base_arm_sh_shear
pin4_area = 0.25 * math.pi * (std_hole_diam**2) # m^2
pin4_shear_stress = pin4_shear / pin4_area # Pa
pin4_shear_fos = pin4_mat_G / pin4_shear_stress
```

```
print("\nPin 4 Shear Stress: ", pin4_shear_stress, "Pa")
print("Pin 4 Shear FOS: ", pin4_shear_fos)



### MAX Deflection Analysis ###
def_from_B1 = -0.5 * R_1z_val * ((plat_l_val - b_val - 0.0254)**2) * ((plat_l_val - b_va
def_from_B2 = -0.5 * R_2z_val * ((plat_l_val - b_val - 0.0762)**2) * ((plat_l_val - b_va
def_from_B3 = -0.5 * R_3z_val * ((plat_l_val - b_val - 0.0254)**2) * ((plat_l_val - b_va
def_from_B4 = -0.5 * R_4z_val * ((plat_l_val - b_val - 0.0762)**2) * ((plat_l_val - b_va
# def_from_tor_theta = -0.5 * plat_w_val * math.tan(8 * base_arm_pm1_T / (base_arm_mat_G
def_from_plat_bend = 5 * (R_1z_val + R_2z_val + R_3z_val + R_4z_val) * (plat_w_val**3) /

print("\nDeflection from B1: ", def_from_B1, "m")
print("Deflection from B2: ", def_from_B2, "m")
print("Deflection from B3: ", def_from_B3, "m")
print("Deflection from B4: ", def_from_B4, "m")
# print("Deflection from Torque: ", def_from_tor_theta, "m")
print("Deflection from Plat Bending: ", def_from_plat_bend, "m")

total_deflection = def_from_B1 + def_from_B2 + def_from_B3 + def_from_B4 + def_from_plat
print("\nTotal Deflection: ", total_deflection, "m")


#### NET BOLT LOAD ####
print("\nNet Bolt Load: ", (R_1z_val + R_2z_val + R_3z_val + R_4z_val), "N")
```