Engineering Portfolio

I Z A A K
M O N T O Y A

2021 - Current

NORTHWESTERN UNIVERSITY BS/MS ‘25
BIOMEDICAL ENGINEERING

Table of Contents

Profile

Name: Izaak Montoya
E-mail: izaakmontoya2025@u.northwestern.edu
LinkedIn: https://tinyurl.com/yzstuby

About this portfolio

This portfolio aims to supplement my resume by highlighting selected projects I've developed during my undergraduate studies. I believe it's important to participate in hands-on projects to apply coursework principles, and I've made sure to do that throughout my time at Northwestern. During my undergraduate studies in biomedical engineering, I got involved with many different projects that familiarized me with many different types of engineering. For each project, I provide a description, source code (when applicable), and images. If you have any questions about any project, feel free to get in touch with me by e-mail.

Respiratory 3D-EIT Phantom

09/2024- Current

Overview

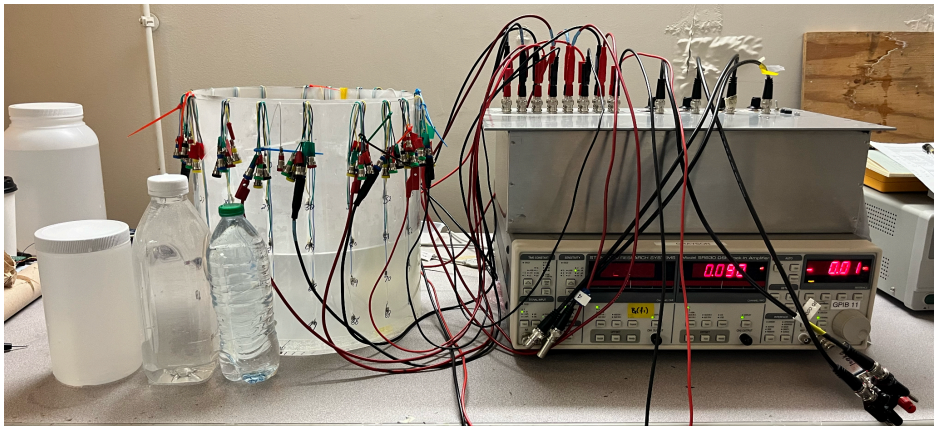
- A project currently being developed for Northwestern's Biomedical Engineering Capstone Design Sequence over two quarters (Fall 2024-Winter 2025).
- Consists of the design of a physical respiratory phantom alongside a surrounding 3D-EIT data collection setup.
    - This anatomically accurate respiratory phantom will have a 3D-printed thoracic shell that I will design in SolidWorks, which will house independently inflatable lungs and bony structures such as the rib cage, shoulder blades, trachea, etc.
    - The data collection setup has already been designed and created during our first quarter of working on the project.
- Utilized by and a collaboration with Northwestern professor Dr. Matthew Grayson for a novel form of 3D-EIT called *Data-Driven EIT*, which could hold the potential to push the entire field of medical imaging forward.

My contributions

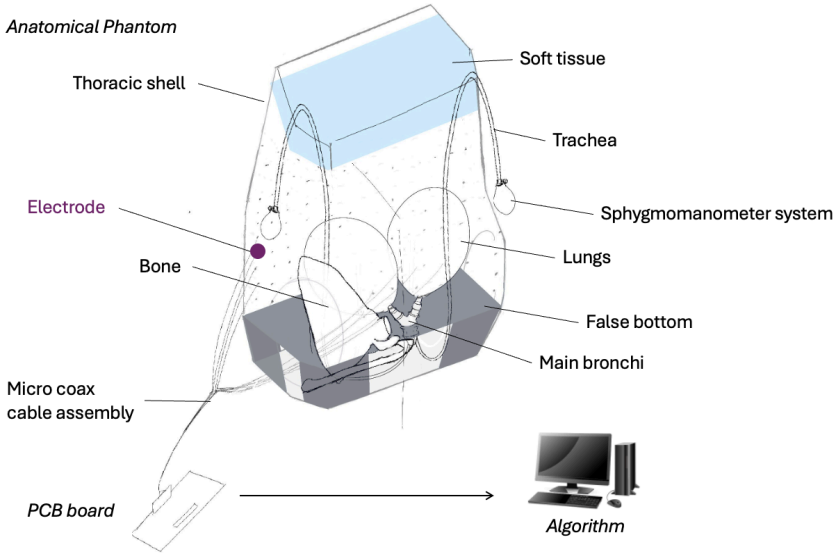
- Heading design of multiple prototypes, setting project schedules and goals.
- Lead contact concerning the technical aspects of the project (how Data-Driven EIT works, what design specifications are required, our client's current research on EIT, etc.). These topics concern electrical engineering and mathematical research that have not been covered in our curriculum. This has made it important to effectively mediate between my team members and clients so that we are on the same page concerning these complex research topics.

Results

- Have completed experimentation on a basic saltwater phantom setup as seen below, ensuring design feasibility and a proper data collection setup for the future more advanced phantom.
    - This means we have already designed and built a functioning 3D-EIT imaging setup.



- Have completed designs of the final anatomically correct phantom and received approval from our client and design coaches, as seen below.



Tamper Evident Bracket

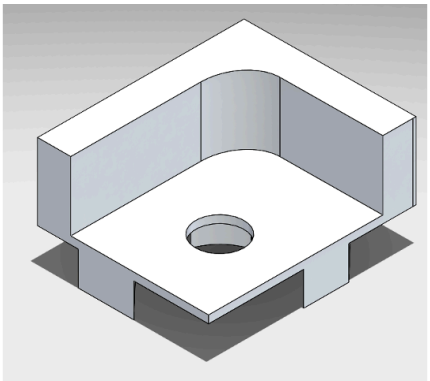
06/2024- 08/2024

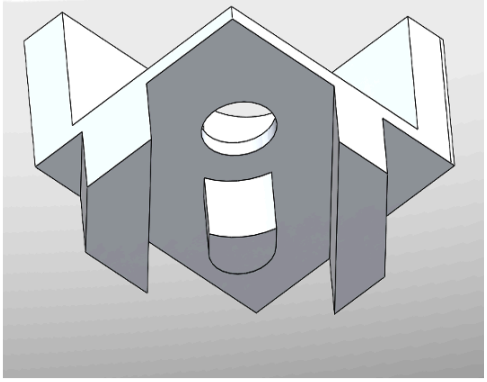
Overview

- Project from my internship at Takeda Pharmaceuticals during the summer of 2024 at a plasma distribution facility.
- A consistent bottleneck at the facility was the packaging speed of a specific high throughput packaging machine, where mechanical adjustments of the tamper-evident sticker applicator station accounted for 15% of all work orders the machine saw.
- This work order data led me to independently identify one of the root causes of the

machine’s inefficiency as redundant mechanical adjustments in the T/E station that moved over time due to both the dozens of employees that worked in the room and the machine's vibration.

- To solve this issue, I designed and 3D-printed a bracket to reduce applicator variability and redundant adjustments that affect ideal applicator positioning.



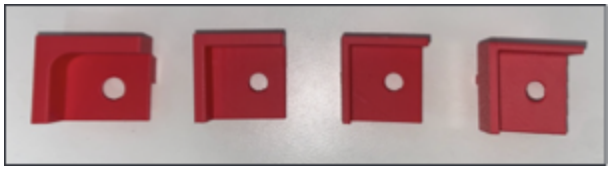


My contributions

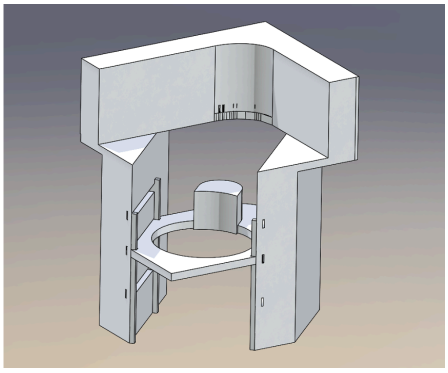
- Identified a root cause of T/E station inefficiencies and independently ideated, designed (in SolidWorks), and implemented a bracket solution to identify the issue.
- Presented project to upper management on multiple occasions to demonstrate cost savings and reduced downtime, securing approval for prototyping and implementation.

Results

- Achieved a final bracket design after a continuous improvement cycle that effectively locked down machine components on both sides of the machine in ideal positions.



- Reached this final design without ever requiring machine downtime by testing for fit with a prototype fitting bracket utilizing an interlocking assembly.



- Machine introduction effects were not seen, but projections added up to over $650,000

in savings and an extra 15,000 bags produced every year from several hours of labor hours reduced every month.

Ecuador Water Distribution System

09/2021- 08/2024

Overview

- Designed and implemented a sustainable water distribution system to improve reliable access for a community of 200+ people, in Paja Colorada, Ecuador.
- The proposed design featured several pumps and tanks to account for elevation change and to store water in various locations. Implementation of the design aimed to establish a functional and reliable water source, storage, and distribution system in order to provide consistent access to clean drinking water straight to around 50 households.
- The project ran through Northwestern's Engineers Without Borders chapter.

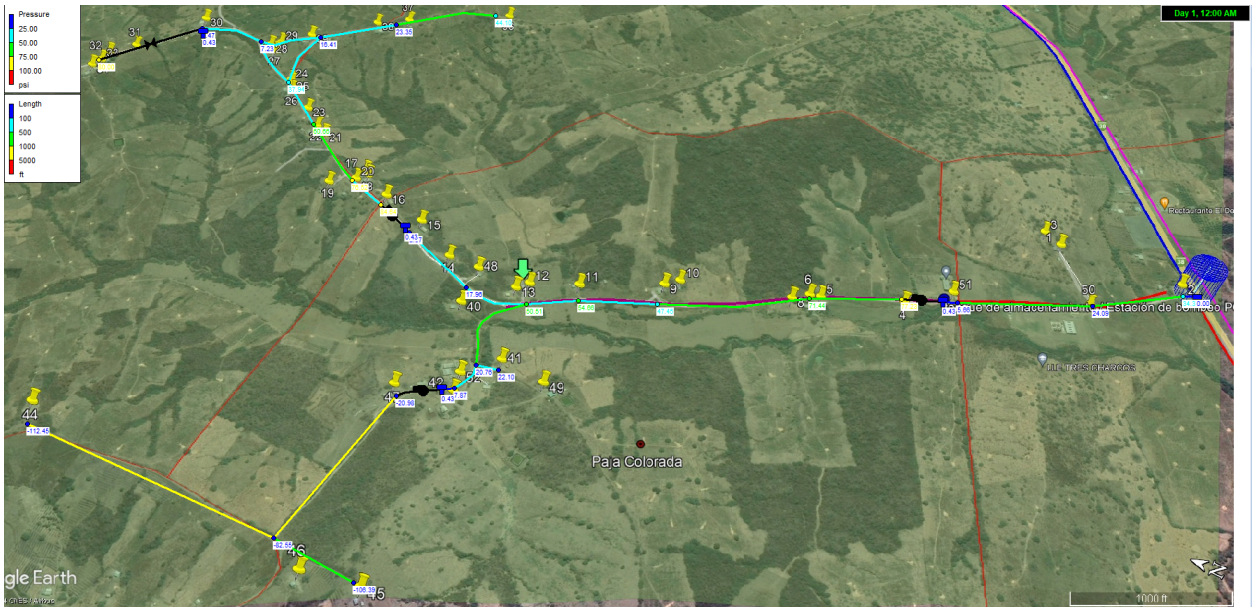
My contributions

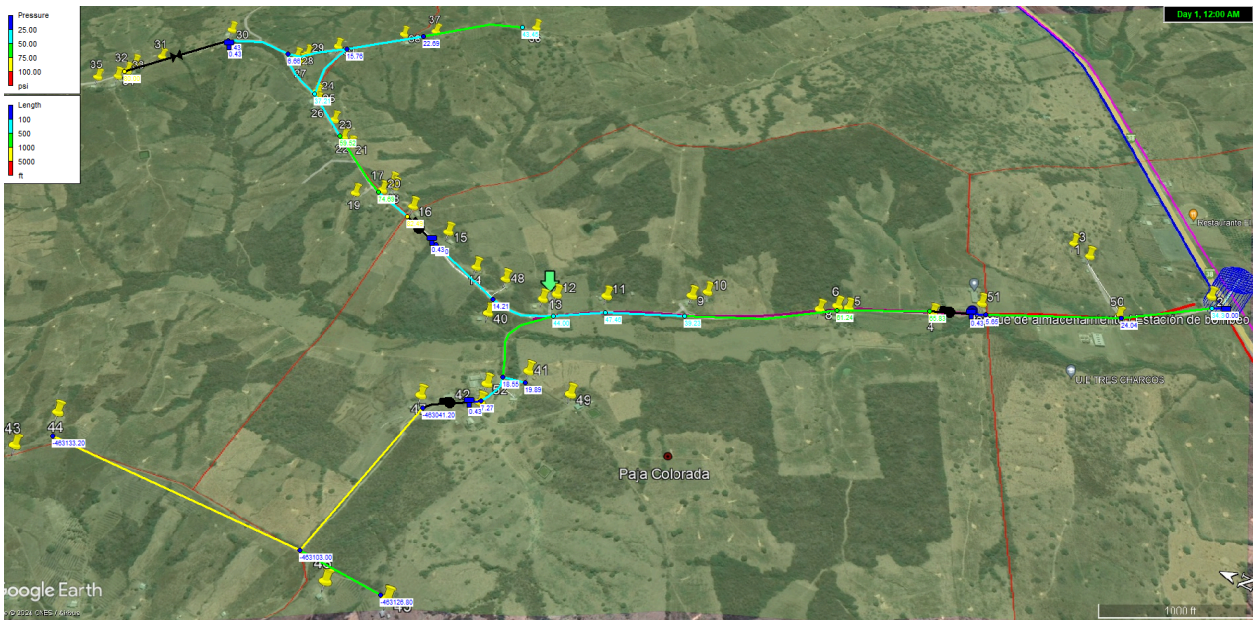
- Served as the President of the team working on this project, leading the project team through the later stages including actual design and closing of the project.
- Designed the proposed piping layout and reservoir connections using hydraulic modeling software, EPANET, to simulate flow behavior and pressure points.
- Led discussions with local partners and national EWB organizations to work through the design process and implementation.

Results

- Led team through a very rocky project closeout due to governing bodies in Ecuador withdrawing support for the project due to high crime. Due to this, the piping design was delivered to the community for potential future use and our funding was used to allow for a more robust piping system in the surrounding region along with water storage tanks throughout the community. The piping to deliver water to individual households was not implemented though, and this was a very valuable lesson for me since I felt some personal responsibility for the project implementation falling short despite it being completely out of our hands. I believe being content with shortcomings when it's out of one's control is an important skill to learn, and I'm still thankful for the people I worked with and the results we achieved.
- Enhanced quality of life for the community through more reliable access to potable water, fostering long-term community growth and health improvements.
- Learned a lot about a type of engineering I've never done before, as this was a more civil engineering-focused project.

- Incorporated low-maintenance materials and gravity-fed systems into the water distribution design to reduce long-term operational costs. This design involved utilizing pumps during the night to fill storage tanks, which were then emptied via gravity to feed water throughout the community. These night and daytime hydraulic modeling simulations can be seen below.





Blood Pressure + Breathing Rate Trackers

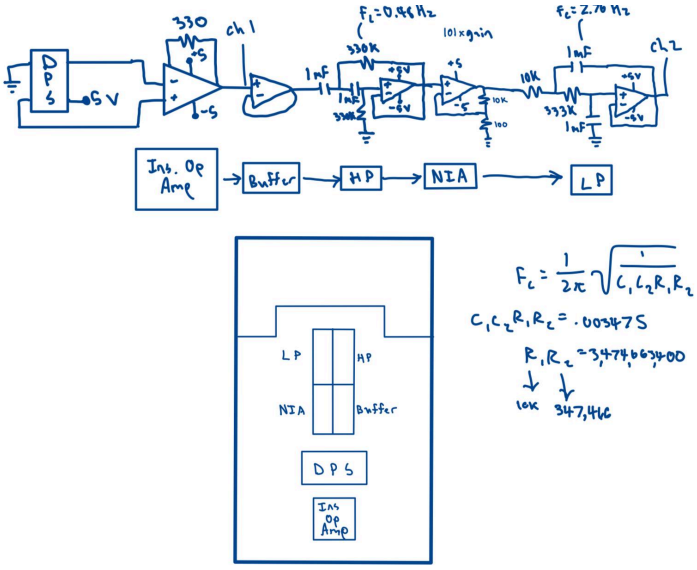
03/2024- 06/2024

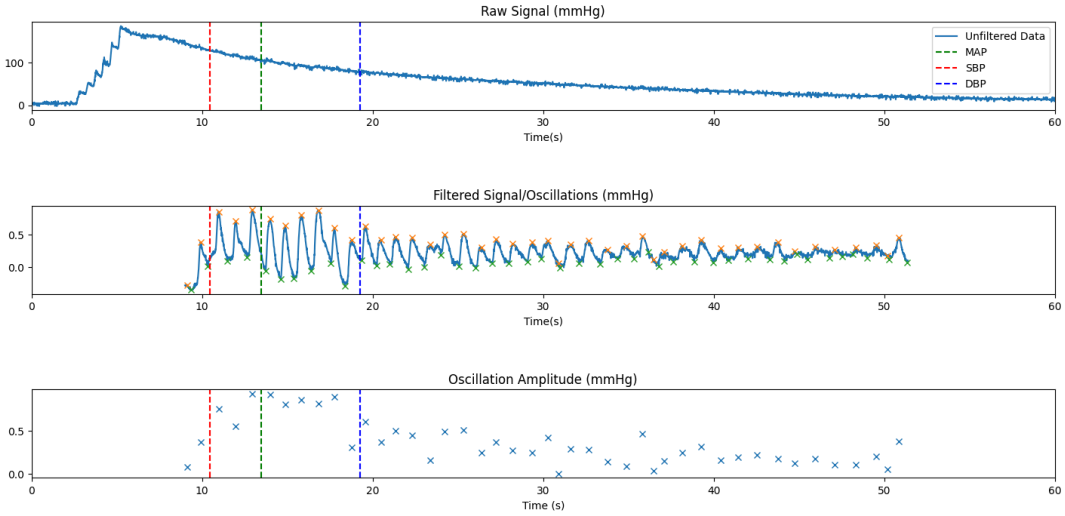
Overview

- Two closely related independently completed projects to design circuits with accompanying code that track a person's breathing rate and blood pressure.
    - The blood pressure sensor uses oscillometry by integrating a consumer cuff with a pressure transducer. This signal is fed into a circuit which filters the data before sending it to a Raspberry Pico. This filtering includes a high pass filter to get rid of noise, followed by data processing in Python to extract mean, systolic, and diastolic arterial pressure.
    - The breathing sensor utilizes a microphone to listen to a person's breathing. This signal is filtered using either 1 or 2 high pass filters on "Exhale" and "Inhale" channels on a Raspberry Pico. After running this data through some code that is included below, it can be detected whether someone is inhaling or exhaling depending on the frequencies present. Depending on how much time is passing between these inhalations/ exhalations, this code also finds a person's breathing rate.
- Done through Northwestern's Biomedical Engineering Circuits course sequence.

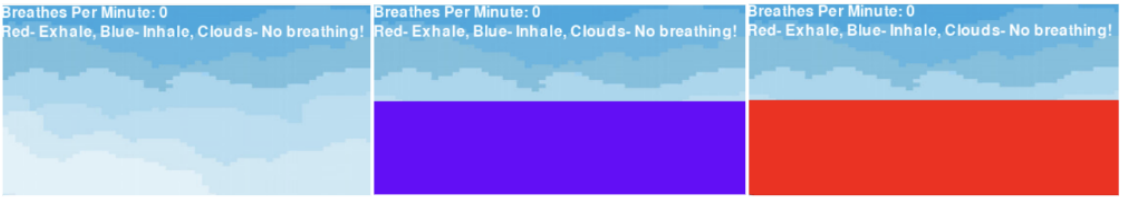
Results

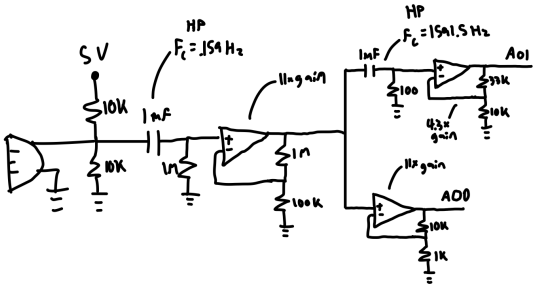
- Achieved anatomically realistic numbers for the blood pressure sensor with these results and the circuit design shown below, signifying an accurate final design.

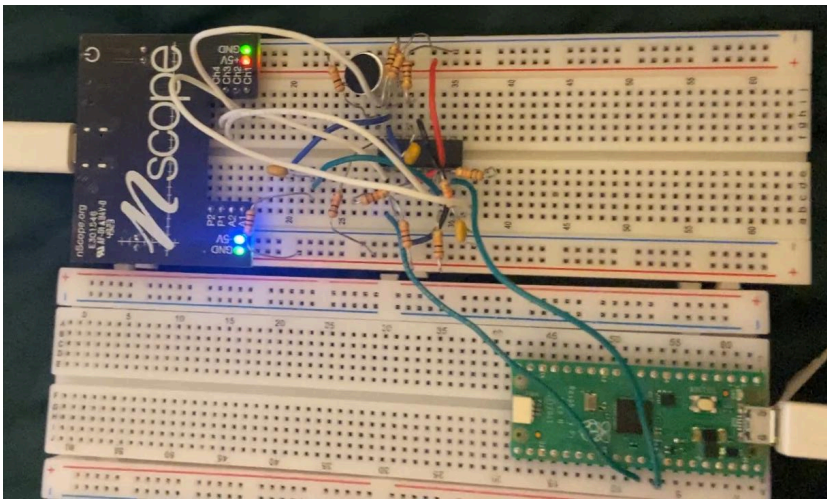




- Achieved anatomically realistic numbers for the breathing sensor with example results, the circuit itself, and the circuit diagram shown below, signifying an accurate final design.







Carbon Curing Frame

01/2022- 03/2022

Overview

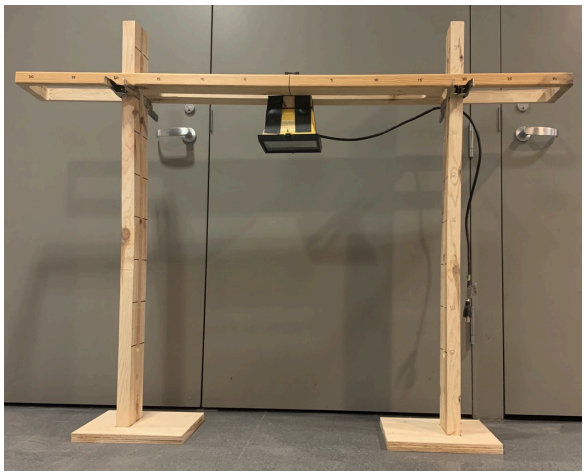
- Designed and prototyped a collapsible, adjustable carbon fiber curing rig for Northwestern Formula Racing to improve uniform heat distribution, storage, and transportability, addressing structural and operational inefficiencies in the team's current curing process.
- This design consisted of vertically sliding overhead boards secured by hand clamps for height adjustment and wide slots for customizable width. Halogen lamps were attached via custom vices along the frame, allowing continuous fine-tuning of heat distribution to accommodate carbon fiber parts of varying sizes.
- Developed the design through client interviews, aerodynamics team demonstrations, iterative physical mockups, and performance tests, incorporating feedback from peers, professors, and shop experts at Segal Design Institute.

My contributions

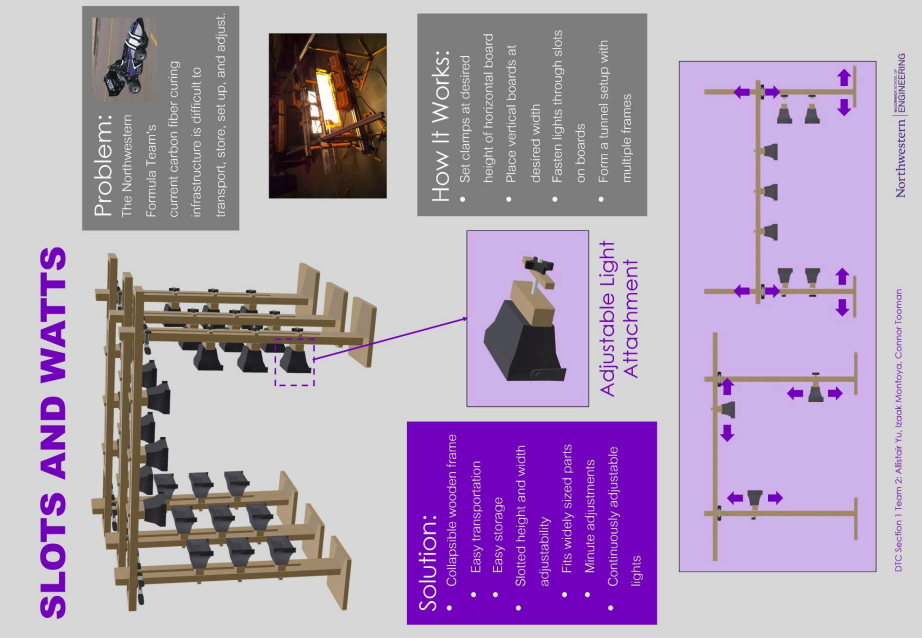
- 3D-modeled the design to allow for easy communication and demonstration of the design's functionality.
- Contributed to a highly functioning engineering team that split project tasks extremely evenly and productively, allowing for contributions to every aspect of the project for each person.

Results

- Achieved a fully functional final design that's shown below that was already ready for implementation by our client, who was happy with the design.



- Presented final design at a design symposium using the below design poster.



10-Way Air Distribution System

03/2022- 06/2022

Overview

- Designed and prototyped a portable, high-capacity air distribution system to efficiently inflate 10+ innertubes simultaneously for a non-profit's (Friends of the Chicago River) annual Summer Float Party, reducing wait times and improving event logistics.
- The design involved a manifold-based system that connects to any standard air compressor, featuring 10 recoil hoses with shut-off valves and universally compatible 3D-printed tips for individual tube inflation. The system's compact design ensures easy transport, storage, and user accessibility.
- Developed the design through client interviews, research, and iterative physical mockups, incorporating feedback from event staff, professors, and peers. Conducted performance tests to meet the goal of inflating four innertubes per minute, refining the prototype based on real-time user input and environmental factors.

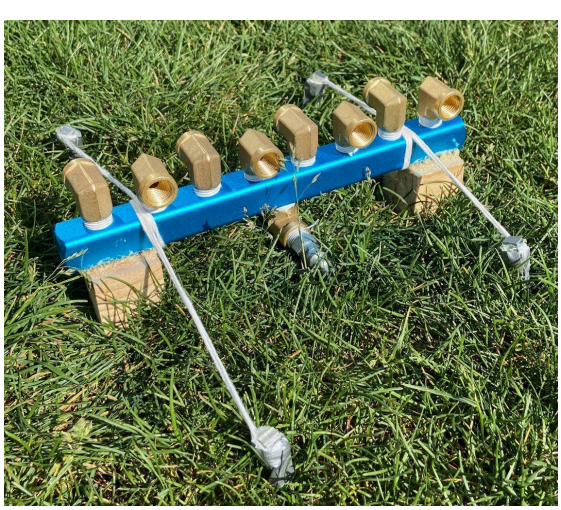
My contributions

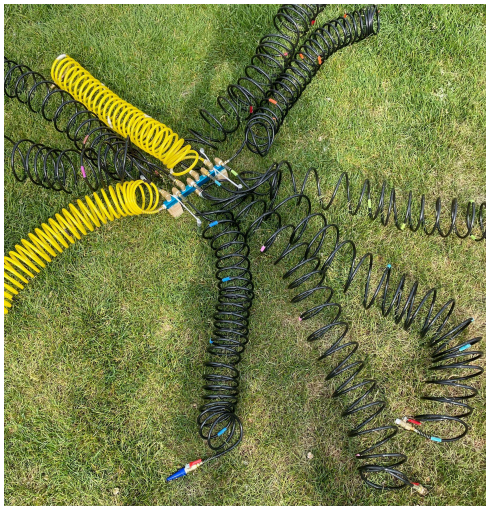
- Led the actual design process as principal engineer, selected correct sub-components, and did calculations on airflow/ pressure allowances.
- Designed hose tips in Blender and 3D-printed components.

- Worked within an effective engineering team for remaining tasks, ensuring optimal communication.

Results

- Achieved a fully functional 10-way air distribution system as seen below that was ready for implementation at the nonprofit event.
- Presented final design at a design symposium, effectively communicating design features and rationale to a larger audience.





Appendix A: Blood Pressure Tracker Code

use_nscope_api.py (used for collecting data from the circuit, initial plotting)

```
# if on a mac M, use arch -x86_64 /usr/bin/python3

import numpy as np # for time array
import matplotlib.pyplot as plt # for plotting
import scipy as scipy
import BMEfunctions as bme
import csv

def save_as_csv(filename, data1, data2):
    with open(filename, 'w', newline='') as csvfile:
        writer = csv.writer(csvfile)
        writer.writerow(['Column1', 'Column2'])  # Write header
        writer.writerows(zip(data1, data2))

# nScope can only be open by 1 program at a time
# make sure the nScope app is closed
try:
    import nscopeapi as nsapi
    ns = nsapi.nScope()
except Exception as e:
    print("Unable to communicate with nScope")
    print(e)
else:
    print("Successfully opened connection to nScope!")

    # turn on A1
```

```
    # ns.setAXWaveType(1,0) # channel (1 or 2), wave type (0=sine, 1=tri)
    # ns.setAXAmplitude(1,3) # channel (1 or 2), voltage
    # ns.setAXUnipolar(1,0) # channel (1 or 2), is uniplar (0=bipolar, 1=unipolar)
    # ns.setAXFrequencyInHz(1,10) # channel (1 or 2), frequency
    # ns.setAXOn(1,1) # channel, 0=off 1=on

    ns.setChannelsOn(True,True,False,False) # which Ch to read from
    sample_rate = 130
    data_points = 7800
    ns.setSampleRateInHz(sample_rate)
    ns.requestData(data_points)

    # wait for all the data to come in
    data = []
    data2 = []
    while ns.requestHasData():
        d = ns.readData(1) * 1000 / (152.5 * 0.0957) # raw voltage * 1000 mV/V / (152.5 gain * .0957 mV/mmHg) -->
Final answer in mmHg
        d2 = ns.readData(2) * 1000 / (101*152.5*0.0957)
        data.append(d) # move the data from Ch1 into the list
        data2.append(d2)

    # make a time array for plotting
    t = np.arange(0,data_points/sample_rate, 1/sample_rate)
    freq, pxx = scipy.signal.periodogram(data,sample_rate)
    freq2, pxx2 = scipy.signal.periodogram(data2,sample_rate)

    # plot the data
    plt.subplot(3,1,1)
    plt.plot(t,data)
    plt.xlabel('Time(s)')
    plt.ylabel('Cuff Pressure(mmHg)')

    plt.subplot(3,1,2)
    plt.plot(t, data2)
    plt.xlabel('Time(s)')
    plt.ylabel('Filtered Pressure(mmHg)')

    plt.subplot(3,1,3)
    plt.plot(freq, pxx, label='Unfiltered Data')
    plt.plot(freq2, pxx2, label='Filtered Data')
    plt.yscale('log')
    plt.xlabel('Frequency (Hz)')
    plt.ylabel('Amplitude (mmHg)')
    plt.xlim(0, 65)  # Set the x-axis limits from 0 to 65
    plt.legend()


    plt.tight_layout()
    plt.show()

    # save the data as a csv
    filename = 'HR_Unfiltered2.csv'
    save_as_csv(filename, t, data)
    filename2 = 'HR_Filtered2.csv'
    save_as_csv(filename2, t, data2)
    print(f'Data saved to {filename}')
    print(f'Data saved to {filename2}')
```

Processing_HR_Data.py (used for processing data from use_nscope_api.py including: trimming, peak and trough finding, blood pressure calculations, and plotting)

```
import numpy as np # for time array
import matplotlib.pyplot as plt # for plotting
import scipy as scipy
import pandas as pd

# Loading in csv data from data collection in use_nscope_api.py
data = pd.read_csv('HR_Unfiltered2.csv')
unfiltered_data = data.iloc[:,1]
data2 = pd.read_csv('HR_Filtered2.csv')
# Only selecting data from 9-51.4 seconds (in this dataset specifically), used for trimming
filtered_data = data2.iloc[1170:6682,1]
time = data2.iloc[1170:6682, 0]
timeog = data2.iloc[:, 0]
sample_rate = 130
# periodogram code for earlier on in the lab
# freq, pxx = scipy.signal.periodogram(unfiltered_data,sample_rate)
# freq2, pxx2 = scipy.signal.periodogram(filtered_data,sample_rate)


#peak and trough finding, with minimum distances of .7*60*sampling rate/HR between peaks/troughs
peaks, _ = scipy.signal.find_peaks(filtered_data, distance = .7*60*130/68)
inverted_signal = -filtered_data
troughs, _ = scipy.signal.find_peaks(inverted_signal, distance = .7*60*130/68)
amp = []

#finding amplitudes in the trimmed data
for i in range(len(peaks)):
    amp.append(filtered_data.iloc[peaks[i]] - filtered_data.iloc[troughs[i]])

x = np.arange(1, len(amp) + 1)

#adjust MAP, SBP, DBP indexing values based on specific dataset

# Peak for max amplitude occurs between 12.92 and 14 --> 13.46 seconds, which times 130 samples per second is 1750
MAP = unfiltered_data.iloc[1750]
print("MAP:" + str(MAP) + "mmHg")

# Max amplitude was 0.933 mmHg, 55% of this is .513. The second and third oscillations have amplitudes of .47 and
.622, respectively.
# So the average of the second and third oscillations will be used for SBP, which occur at 9.91 sec and 10.96 sec
SBP = (unfiltered_data.iloc[1288] + unfiltered_data.iloc[1425]) / 2
print("SBP:" + str(SBP) + "mmHg")

# Max amplitude was .933 mmHg, 70% of this is .653. The eleventh and twelth oscillations have amplitudes of .31 and
.61, respectively.
# The average of these oscillations will be used for DBP since this seems to be where the cutoff would occur, which
occur at 18.87 sec and 19.63 sec
DBP = (unfiltered_data.iloc[2453] + unfiltered_data.iloc[2552]) / 2
print("DBP:" + str(DBP) + "mmHg")


# plot the data in 3 subplots, final output desired from this lab
plt.subplot(3,1,1)
plt.plot(timeog,unfiltered_data, label='Unfiltered Data')
plt.axvline(x=13.46, color='g', linestyle='--', label='MAP')
plt.axvline(x=10.44, color='r', linestyle='--', label='SBP')
plt.axvline(x=19.25, color='b', linestyle='--', label='DBP')
plt.xlabel('Time(s)')
plt.title('Raw Signal (mmHg)')
plt.xlim(0, 60)
plt.legend()
```

```
plt.subplot(3,1,2)
plt.plot(time, filtered_data, label='Filtered Data')
plt.plot(time.iloc[peaks], filtered_data.iloc[peaks], "x")
plt.plot(time.iloc[troughs], filtered_data.iloc[troughs], "x")
plt.axvline(x=13.46, color='g', linestyle='--', label='MAP')
plt.axvline(x=10.44, color='r', linestyle='--', label='SBP')
plt.axvline(x=19.25, color='b', linestyle='--', label='DBP')
plt.xlabel('Time(s)')
plt.title('Filtered Signal/Oscillations (mmHg)')
plt.xlim(0, 60)

plt.subplot(3,1,3)
plt.plot(time.iloc[peaks], amp, 'x')
plt.axvline(x=13.46, color='g', linestyle='--', label='MAP')
plt.axvline(x=10.44, color='r', linestyle='--', label='SBP')
plt.axvline(x=19.25, color='b', linestyle='--', label='DBP')
plt.xlabel('Time (s)')
plt.title('Oscillation Amplitude (mmHg)')
plt.xlim(0, 60)

plt.tight_layout()
plt.show()

# for plotting periodogram earlier in the lab
# plt.plot(freq, pxx, label='Unfiltered Data')
# plt.plot(freq2, pxx2, label='Filtered Data')
# plt.legend()
# plt.yscale('log')
# plt.xlabel('Frequency (Hz)')
# plt.ylabel('Amplitude (mmHg)')
# plt.xlim(0, 65)  # Set the x-axis limits from 0 to 65
# plt.show()
```

Appendix B: Breathing Rate Tracker Code

```
#initializing and loading pygame
import serial
ser = serial.Serial('COM3')
print('Opening port: ' + str(ser.name))
import pgzrun, pygame
import time
WIDTH = 600
HEIGHT = 200
n1_int = 0
n2_int = 0
t_prev = 0
bpm_int = 0
# Load and sclae the background image
background_image = pygame.image.load("8bitclouds.jpg").convert()
background_image = pygame.transform.scale(background_image, (WIDTH, HEIGHT))
# Load, play, and set the volume for a music track
pygame.mixer.music.load("music.mp3")
pygame.mixer.music.play(loops=-1)
pygame.mixer.music.set_volume(0.3)
def update():
n_bytes = ser.readline() # read all the letters available
s = str(n_bytes) # turn them into a str
result1 = s[s.find('(')+1:s.find(',')] # find everything beween ( and ,
result2 = s[s.find(',')+1:s.find(')')] # find everything between , and )
global n1_int
n1_int = int(result1) # convert str to int
```

```
global n2_int
n2_int = int(result2)
global t_now
t_now = time.monotonic()
global t_prev
global bpm_int
#calculate breathes per minute by finding time between exhales
if (n2_int> 65535*1/5) and (t_now-t_prev > 1.5): #signal oscillates, filtering
out peaks that occur faster than what is reasonable
bpm_int = int(60/ (t_now-t_prev))
t_prev = t_now
time.sleep(.001)
def draw():
screen.blit(background_image, (0, 0))
global bpm_int
global t_now
screen.draw.text('Breathes Per Minute: ' + str(bpm_int),(0, 0))
screen.draw.text('Red- Exhale, Blue- Inhale, Clouds- No breathing!', (0,20))
#changing what is drawn to the screen depending on exhale vs inhale vs no
breathing
if (n2_int>65535*0/5) and (t_now-t_prev < 1.5): #exhale happening
x = 600
r = 255
b = 0
time.sleep(.001)
elif (n1_int>65535*1/5) and (t_now-t_prev > 1.5): #inhale happening
x = 600
r = 108
b = 255
else: #no breathing
x = 0
r = 0
b = 0
screen.draw.filled_rect(Rect((WIDTH-x,HEIGHT-100),(WIDTH,HEIGHT)),(r,0,b))
pgzrun.go()
```