8/20/2024 (Tuesday, 11:10 AM, EABA Room 121)
First day of class. Team formation and initial brainstorming session.
Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah Kilpatrick

(Team formed)

Action items:
1. Each team member to research one project idea further
2. Prepare pros and cons for each idea for next meeting

8/22/2024 (Thursday, 11:10 AM, EABA Room 121)
Team meeting to select final project.
Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah Kilpatrick

We came up with four project ideas:

1. Machine Intelligence Mimicking Industrial Choreography (MIMIC)
   - Problem: Inefficient factory environments
   - Solution: Design a tool to teach robotic arm hand movements
   - Next steps: Research modifiable robotic arms

2. Car To Car Collision Avoidance (C2C)
   - Problem: ~3,000 US deaths annually due to distracted driving
   - Solution: Car sensors providing alerts through GUIs
   - Challenges: Response time concerns
   - Approach: Start with proof of concept, then scale

3. Guidance Utility for Impaired Daily Experiences (GUIDE)
   - Problem: Over 340,000 Americans have no vision
   - Solution: Device to sense nearby objects and alert through auditory cues
   - Technologies to explore: Sensors, Cameras, LIDAR

4. Accelerated Traffic Light Automation System (ATLAS)
   - Problem: 28 million tons of CO2 emitted from idling at inefficient traffic lights
   - Solution: Optimize traffic light states using computer vision
   - Innovative approach: Developing a system without traditional traffic lights

We thought about each decision and took each into consideration a fair amount. After 2 rounds of voting and some deliberation, **we chose GUIDE as our project.**

Rationale for selection:
1. Significant social impact
2. Aligns with team's technical skills
3. Feasible within project timeframe and budget

Initial hardware requirements brainstorming:
- Depth camera
- Microcontroller (possibly Raspberry Pi, arduino, or esp32)
- LIDAR sensor
- Portable power source

Action items:
1. Research potential components and create preliminary list
2. Investigate similar existing technologies
3. Draft initial project timeline

Questions for advisor:
1. Any recommendations for specific hardware components?
2. Have any similar projects been attempted in recent years? What hardware/software did they use and how successful were they?

8/27/2024 (Tuesday, 11:10 AM, EABA Room 121)
Team progress meeting.
Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah Kilpatrick

Tasks accomplished:
1. Began creation of initial bill of materials (BOM)
2. Ordered depth camera
https://www.amazon.com/dp/B0BWM21YK8?ref=ppx_yo2ov_dt_b_fed_asin_title

Our current Bill of Materials

Part Type	Cost	Link	Pros/Cons
Walking Stick	$17.99	Walking Stick - Amazon	
7.4V Lithium Ion Battery	$13.99	7.4V Battery - Amazon	4.2 Rating, very little reviews
6V Lithium Ion Battery	$14.99	6V Battery - Amazon	4.5 Rating, fast charging and comes with charger.
TFmini-S LiDAR	$45	12m LiDAR - Amazon	4.5 Rating, UART compatible, 0.704 oz, 1.65 x 0.71 x 0.59
TF-Luna	$24	8m LiDAR - Amazon	4.5 Rating, UART compatible, 5g, 35 x 19 x 21.25mm
Vibrating Motor	$7.59	Vibrating Motor	4 Rating, Light Weight, 0.63 oz, simple, small
RPLiDAR	$99.00	LiDAR - Amazon	
Ovonic Air 500mAh 11.1V Battery	$25.99	11.1V Battery - Amazon	
Supulse Battery Charger	$12.99	Li Po Battery Charger - Amazon	

Concerns:
- Potential compatibility issues between chosen components (Are all the communication protocols sufficient and reliable [UART, SPI, I2C, etc.])
- Need to determine power requirements for portable operation (Each MWh matters in the context of a battery pack and limited power supply.

Next steps:
1. Research power consumption of selected components
2. Begin drafting system architecture diagram

8/29/2024 (Thursday, 11:10 AM, EABA Room 121)
Focus: Hardware component research

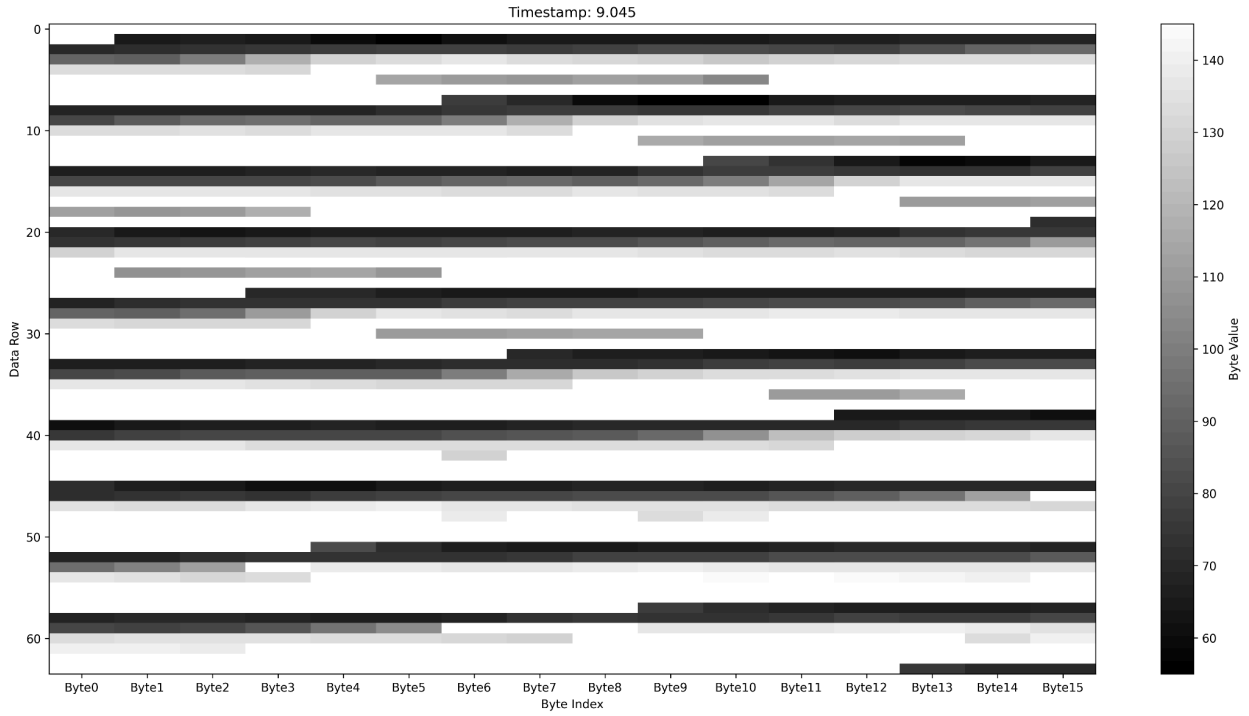
Key findings:
1. Discussed potential suppliers with pros and cons]
2. Identified potential issues with LIDAR in varying light conditions

The depth camera that was ordered arrived. I have began researching documentation and coding by making a python script that accesses the serial port.

```
try:  # open port
   print(f"Attempting to open {port}...")
   ser = serial.Serial(port, 115200, timeout=1)
   time.sleep(3)  # Allow some time for port initialization
   print(f"Successfully opened {port}")
```

```
while True:
   data = ser.read(1024)  # Adjust buffer size as needed
   print_raw_data(data)
```

The final product will be an embedded C/C++/ Micropython binary loaded onto a microcontroller, but this gives us a good idea of what to do/expect with the data coming from the depth camera serial port.



This is a frame of the data with the program I wrote. It definitely gives low values when the camera is close to an object, and high values when the program is far away from an object. I will need to adjust the way the camera is giving data back from
Jack Couture is attempting to properly decode this with the FFMPEG C++ libraries.

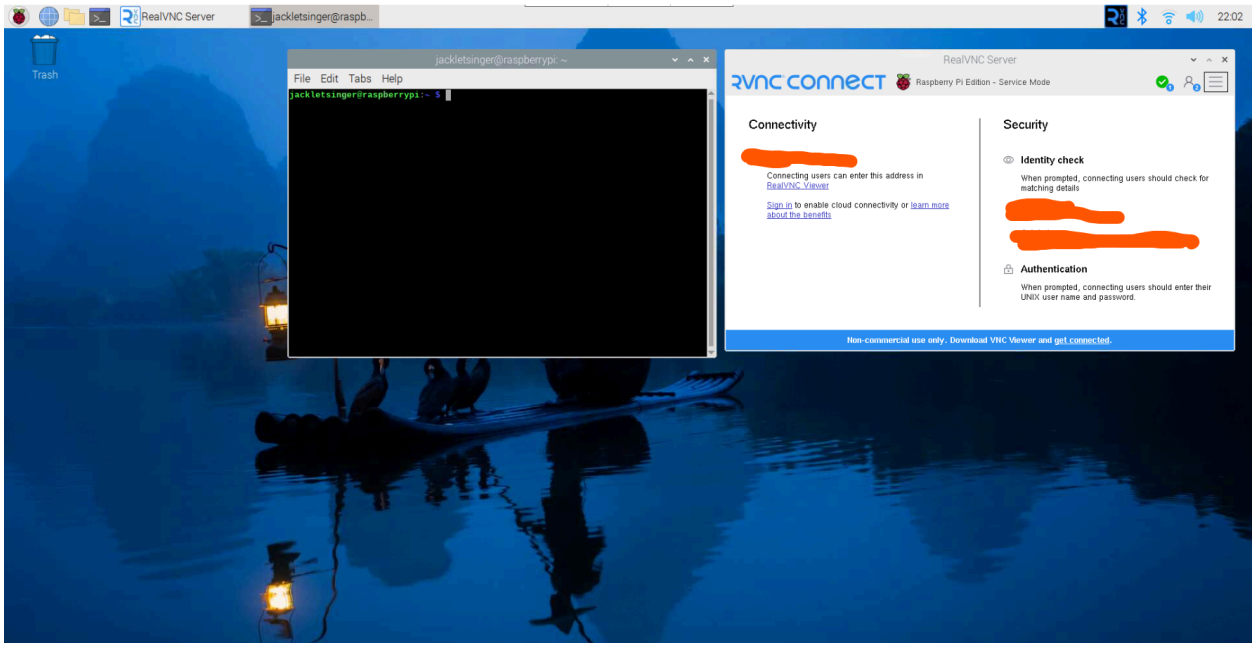
Action items:
1. Share findings with team
2. Schedule meeting with faculty advisor to discuss component selection

9/03/2024 (Tuesday, 11:10 AM, EABA Room 121)
Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah Kilpatrick
Agenda: Finalize parts list

Decisions made:

Concerns:

I set up a Virtual machine with hardware hosted in order to facilitate the development process of the project.



Next steps:
1. Place orders for remaining components
2. Begin drafting testing procedures for each component

Setup Raspberry PI

9/05/2024 (Thursday, 11:10 AM, EABA Room 121)
Team meeting: Hardware and software integration discussion
Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah Kilpatrick

Here I have gotten the Raspberry Pi Pico 2 to display what pins are available on the pinout, as well as the ADC (Analogue to Digital Converter) to show voltages on the pin. This is through Thonny and Micropython, and is an intermediary step in learning how the hardware/software works

Key points:
1. Software team proposed initial architecture
2. Identified potential interface challenges between [specific hardware] and [specific software module]

Action items:
1. Research communication protocols for chosen microcontroller
2. Draft hardware section of project proposal
3. Schedule joint hardware-software testing session

[Include basic block diagram of hardware-software interface]

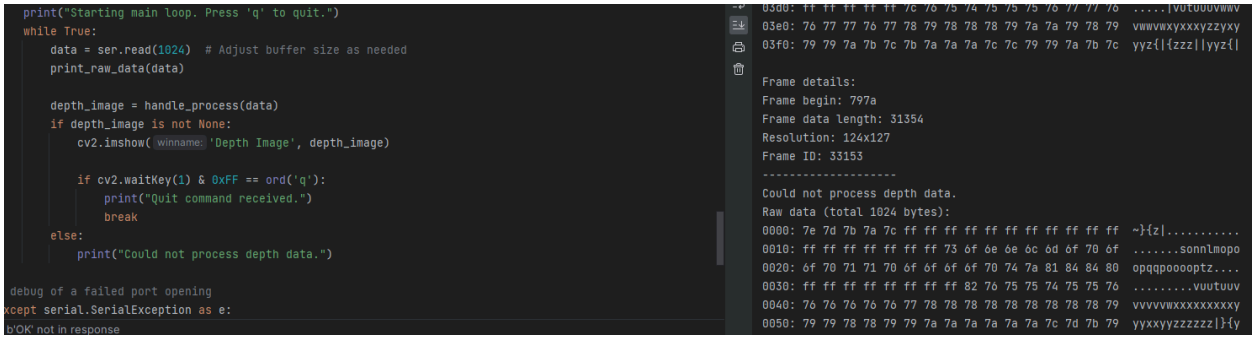
9/10/2024 (Tuesday, 11:10 AM, EABA Room 121)
Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Noah Kilpatrick
Focus: Proposal preparation and component ordering

Tasks completed:
1. Ordered Raspberry Pi Pico (Model: [specify], Cost: $20)
2. Studied Raspberry Pi SDK documentation
3. Researched UART capabilities of Raspberry Pi Pico 2

Key learnings:
[List 2-3 important points about Raspberry Pi Pico capabilities]

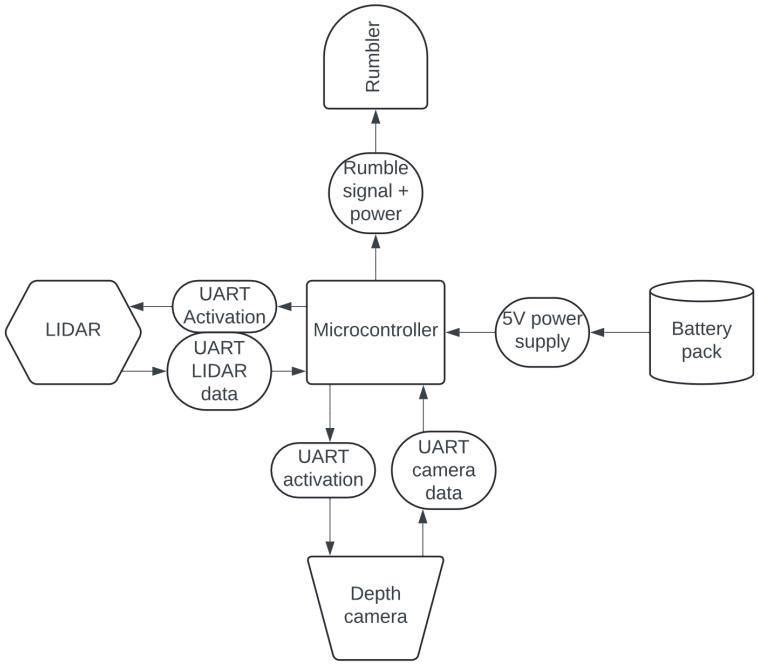
Next steps:
1. Complete hardware section of proposal
2. Prepare initial setup instructions for Raspberry Pi Pico

As for the depth camera, more has been done to understand the serial port interface



9/12/2024 (Thursday, 11:10 AM, EABA Room 121)
Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah Kilpatrick
Focus: Proposal finalization and additional component ordering

Accomplishments:
1. Created hardware interconnectivity diagram



Proposal status:
- Hardware section draft completed
- Pending review from team lead

Action items:
1. Prepare 5-minute presentation on hardware architecture for proposal defense
2. Begin drafting testing procedures for newly ordered components

9/17/2024 (Tuesday, 11:10 AM, EABA Room 121)
Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah Kilpatrick

Proposal: Presentation Day

Question: **Why a raspberry pi pico 2?**

After debating between an ESP32, Arduino, or raspberry pi 2
Arduino is good for beginners because of the Arduino IDE, but lacking in depth in more complex projects.
ESP32 is good, but the version differences, lack of dedicated IDE & baremetal support leaves much to be desired.

Raspberry Pi Pico 2

- Extremely power efficient
- UART/Serial port capabilities (as opposed to I2C or SPI)
- Python / C/C++ capabilities, access to the Raspberry PI SDK.

Key feedback on hardware section (From professor and others):
1. We needed a Gantt chart or time
2. We needed a test and integration plan
3. We needed better descriptions of device specifications during the presentation.

2. Suggestions for alternative battery pack to consider

Action items based on feedback:
1. Research suggested alternative component
2. Update BOM if changes are necessary

9/19/2024 (Thursday, 1:00 PM, Lab 204)
Team meeting: Post-proposal planning
Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah Kilpatrick

Completed all aspects of the GUIDE proposal, listened to the remaining presentations.

Hardware update: I had to enable port forwarding on my raspberry pi in order for other team members to

Status update:
- All hardware components finalized and ordered

- Began drafting assembly and testing procedures

Next phase planning:
1. Assign specific testing responsibilities to team members
2. Set up regular check-ins for component arrival and initial testing
3. Begin designing mounting system for hardware components

Questions for next advisor meeting:
1. Recommendations for robust testing environments?
2. Suggestions for user safety considerations during testing?

9/24/2024 (Tuesday, 11:10 PM, EABA Room 121)

During this hardware meeting we evaluated the materials we were given. We made the choice to strip the wire header of the lidar so it could be used with the pinout, as opposed to having to wait to order whatever 6-pin header was needed. The wire could be disconnected or replaced if needed, and is not permanently attached to the Lidar. Ryan and Jack soldered the Raspberry Pi pico pin headers, allowing it to be used on a breadboard.

9/24/2024 (Tuesday, 11:10 AM, EABA Room 121) Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah

All ordered hardware components arrived, marking the start of the physical build phase. Initial setup and testing of the Raspberry Pi Pico began, focusing on verifying the functionality of each component. Voltage tests were conducted using male-to-male wires and a multimeter, confirming the proper operation of the Raspberry Pi Pico and associated wiring.

However, the LiDAR sensor proved unresponsive during initial tests. This unexpected issue led to the consideration of alternative communication protocols, specifically I2C, to troubleshoot the LiDAR's functionality. The team planned to investigate whether the LiDAR required a specific UART activation signal or if the unit itself was defective, setting a goal to determine the viability of the LiDAR by the following day.

9/26/2024 (Thursday, 11:10 AM, EABA Room 121) Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah

The development environment configuration began in earnest, with the setup of a virtual machine dedicated to data collection and testing. This virtual environment was crucial for facilitating collaborative development and ensuring consistent testing conditions across the

team. A small test program was written to verify the basic functionality of the Pi Pico, serving as a foundation for more complex implementations.

During this process, an important discovery was made regarding the USB serial connection of the Pi Pico. When uploading programs that don't explicitly initialize the USB serial connection, the Pico ceased to appear as a COM port in Windows. This behavior was due to the Pico's default USB stack being replaced by the uploaded program. To address this, the team planned to implement a library that would maintain USB serial functionality post-program execution, enabling real-time debugging of the LiDAR and other components.

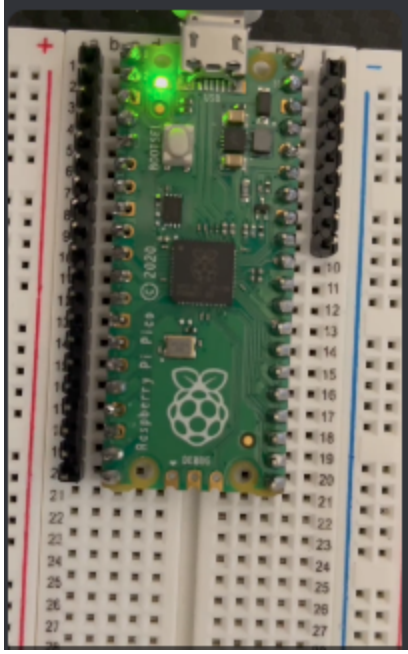
```
if (YY):
  continue // skip

else:
  #function for decoding protocol data byte format
```

```
const int LED_PIN = 25;

void setup() {
  pinMode(LED_PIN,OUTPUT);
}

void loop() {
  digitalWrite(LED_PIN, HIGH);
  delay(1000);
  digitalWrite(LED_PIN, LOW);
  delay(1000);
}
```



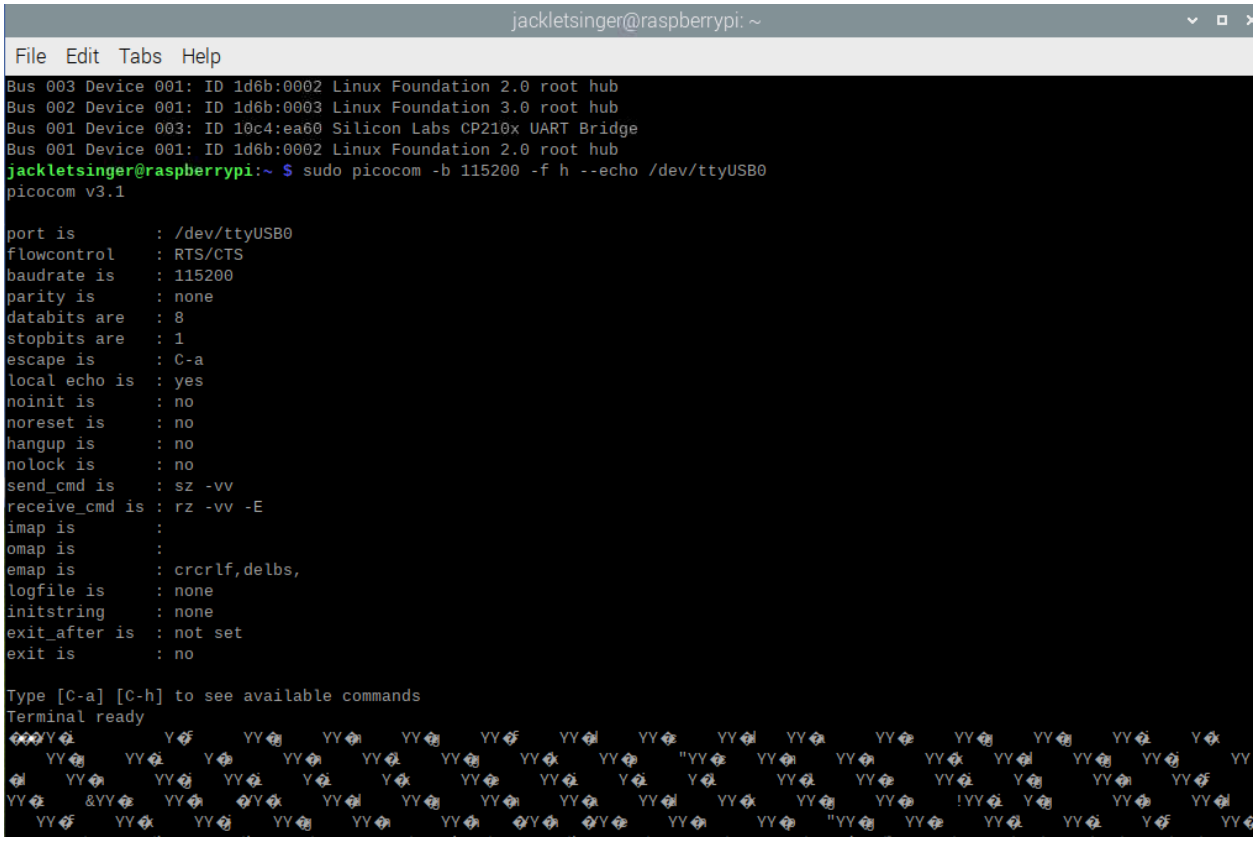
Bit of an update, I am using the Arduino IDE and can write code that changes led/voltages etc. I understand the problem in greater detail now. When you upload a program to the Pico that doesn't explicitly initialize the USB serial connection, the Pico stops appearing as a COM port in Windows. This is because the Pico's default USB stack is being replaced by your program, which doesn't include USB serial functionality. I have class now. When I return home, I'm going to try out a library that puts USB serial functionality onto the pi after the program has booted/executed (If you are asking why I am doing this, its so I can read values when whatever program is running, so we can more effectively debug the LiDAR in real time)

10/1/2024 (Tuesday, 11:10 AM, EABA Room 121) Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah

Started testing sensor capabilities. Verified LiDAR and depth cameras are working as expected outside of an embedded environment.

10/3/2024 (Thursday, 11:10 AM, EABA Room 121) Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah

Continued sensor testing. Confirmed both sensors are within project specifications for range and accuracy. Gave the Software team the assignment of decoding the packets coming from the camera and LiDAR.

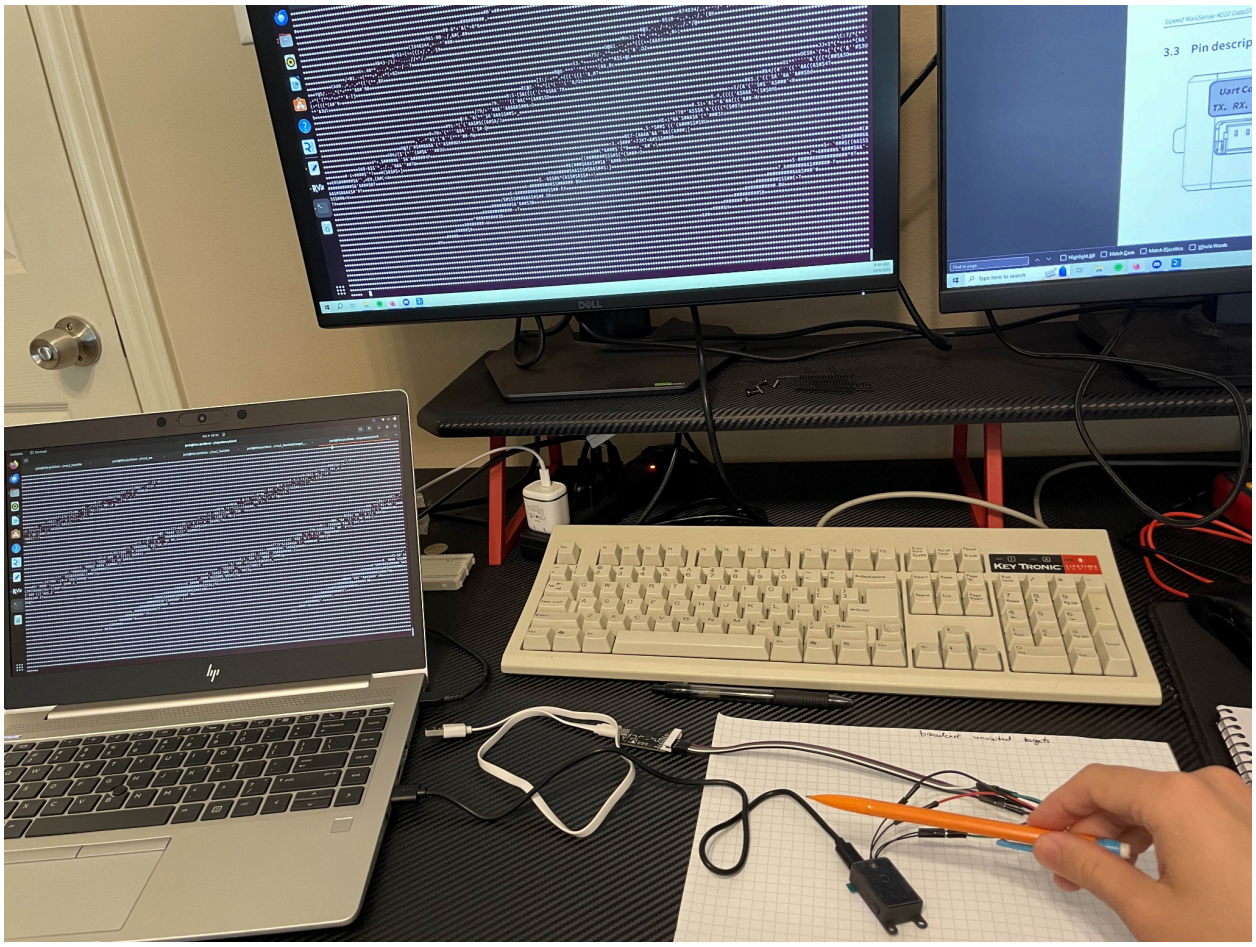


10/8/2024 (Tuesday, 11:10 AM, EABA Room 121) Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah

Sensor testing entered a critical phase as the team began verifying the capabilities of the LiDAR and depth camera outside of the embedded environment. This step was crucial in establishing a baseline for sensor performance before integration with the Raspberry Pi Pico. The focus was on ensuring that both sensors could accurately detect obstacles within the project's specified 2-meter range.

The team meticulously documented the performance characteristics of each sensor, paying particular attention to detection accuracy, response times, and consistency of readings. This

data would be essential for fine-tuning the obstacle detection algorithms and ensuring the final product met the project requirements for user safety and reliability.

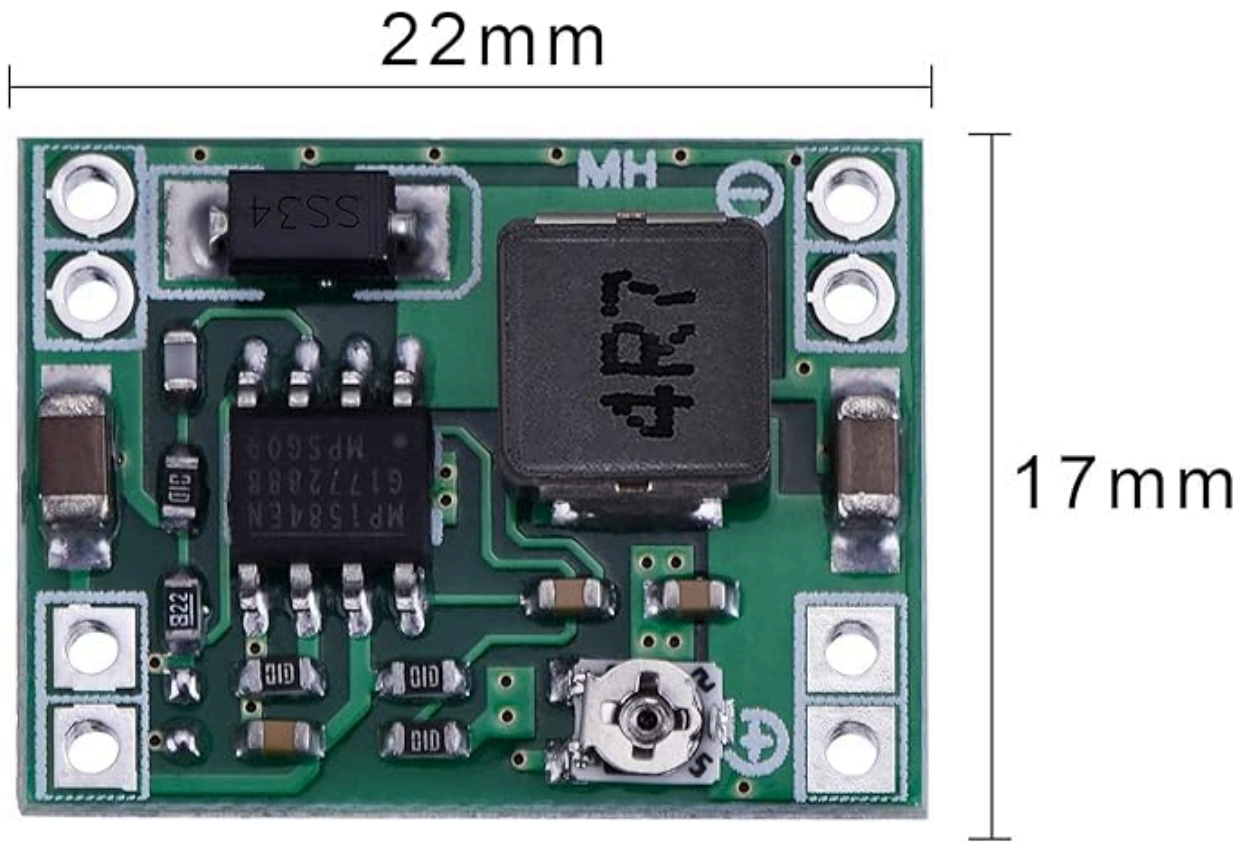


Got the camera working with the UART, as well as the UART bridge (all in Linux)
Look at the pattern on the screen and what I'm holding over the depth camera
I don't know how, but the camera is literally transmitting the image in ASCII or something to create a physical representation of what's above it. I'm debugging it right now to figure out what other capabilities it has

10/10/2024 (Thursday, 11:10 AM, EABA Room 121) Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah

The integration of sensors with the Raspberry Pi Pico commenced, marking a significant milestone in the project's hardware development. The team focused on establishing robust power management systems and creating reliable electrical connections between components. This phase was crucial for ensuring the longevity and reliability of the GUIDE device in real-world usage scenarios.

A major breakthrough occurred with the successful establishment of UART communication with the depth camera. The team was surprised and intrigued to observe an ASCII-like image representation in the camera's output. This unexpected capability sparked discussions about potential additional features that could be implemented using the depth camera's data, potentially enhancing the obstacle detection capabilities of the GUIDE system.

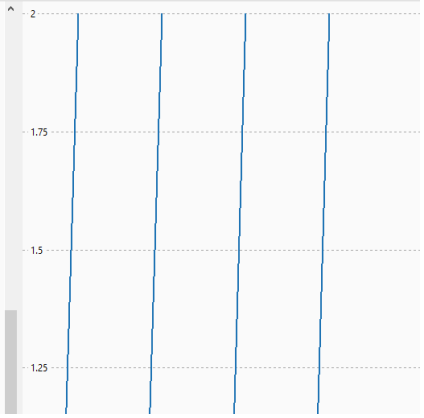


<untitled> * | [main.py]

```
69
70      try:
71          if uart.any():
72              print('Testing 1')
73              # Read data from UART
74              data = uart.read()
75              print('Testing 2')
76              # Decode bytes to string and print
77              print(data)
78              #print(data.decode('utf-8'), end='')
79              print('Testing 3')
80          # Small delay to prevent busy waiting
```

Shell

```
b"'1\xff#.*$\xff#\xff#G#'*########/(#####$##\xff#\xff#1&#'/>HG0#Ux2&#( …
Testing 3
testing post packet
testing 1
Testing 1
Testing 2
b"\x00\xff '\xff\x00luA\x0c\t\x00\x00\x00ddu\x17#\x00\xff\xff\xff\xff\ …
#$$$##&5999(##$.#####3.1+#+7#\xff#*+'\xffA#'.+##\xff#####**+-,(*#\xff# …
Testing 3
testing post packet
testing 1
Testing 1
Testing 2
b"\x00\xff '\xff\x00lvA\x0c\t\x00\x00\x00dd\x91\x17#\x00\xff\xff\xff\x …
Testing 3
testing post packet
testing 1
Testing 1
Testing 2
b"##-51-&#+03-##*#&+*7D>-'###5<:.&.9?=11>;,+4:?:/1>4#,<@>+##/63,(+31+1 …
Testing 3
testing post packet
testing 1
Testing 1
Testing 2
b"\x00\xff '\xff\x00lvA\x0c\t\x00\x00\x00dd\xba\x17#\x00\xff\xff\xff\x …
###*###$/5###34*#$.5/,$####01####'5##*7>>=6(##0/###--##+/$######'1'### …
```

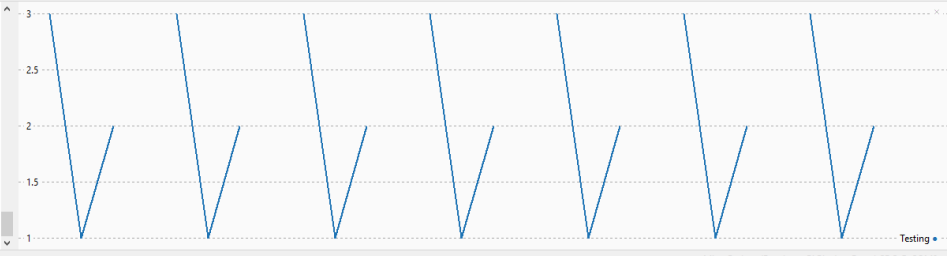


<untitled> * | [main.py]

```
1   from machine import UART, Pin
2   import utime
3
4   # Initialize UART on GP0 (TX) and GP1 (RX)
5   uart = UART(0, baudrate=115200, tx=Pin(0), rx=Pin(1))
6   #uart = UART(1, baudrate=115200, tx=Pin(4), rx=Pin(5))
7   print("Testing 0")
8
9   # Main loop
10  while True:
11      try:
12          if uart.any():
13              print('Testing 1')
14              # Read data from UART
15              data = uart.read()
16              print('Testing 2')
17              # Decode bytes to string and print
18              print(data)
19              #print(data.decode('utf-8'), end='')
20              print('Testing 3')
21          # Small delay to prevent busy waiting
22
23      except:
24          print("Error")
25
26      utime.sleep_ms(1000)
```

Shell

```
4\x11h\t\x0eYY\xc6\x00\x1b\x11h\t\x15Y
Y\xc6\x00\x13\x11h\t\rYY\xc6\x00\x15\x
11h\t\x0fYY\xc6\x00\x16\x11h\t\x10YY\x
c6\x00\x0f\x11h\t\tYY\xc6\x00\x10\x11h
\t\nYY\xc6\x00\x14\x11h\t\x0eYY\xc6\x0
0\n\x11h\t\x04YY\xc6\x00\x10\x11h\t\nY
Y\xc6\x00\x19\x11h\t\x13YY\xc6\x00\x0c
\x11h\t\x06YY\xc6\x00\x13\x11h\t\rYY\x
c6\x00\x16\x11h\t\x10YY\xc6\x00\x12\x1
1h\t\x0cYY\xc6\x00\x12\x11h\t\x0cYY\xc
6\x00\x15\x11h\t\x0fYY\xc6\x00\x13\x11
h\t\rYY\xc6\x00\x0f\x11h\t\tYY\xc6\x00
\x0f\x11h\t\tYY\xc6\x00\x12\x11h\t\x0c
YY\xc6\x00\x08\x11h\t\x02YY\xc6\x00\x0
b\x11h\t\x05YY\xc6\x00\x10\x11h\t\nYY\
xc6\x00\n\x11h\t\x04'
Testing 3
```



MicroPython (Raspberry Pi Pico) • Board CDC @ COM8

```
b'YY\xc6\x00\xe8\x10p\t\xe9YY\xc6\x00\xe6\x10p\t\xe7YY\xc6\x00\xe7\x10p\t\xe8YY\xc6\x00\xea\x10p\t\xebYY\xc6\x00\xec\x10p\t\xedYY\xc6\x00\xe6\x10p\t\xe7YY\xc6\x00\xf0\x10p\t\xf1YY\xc6\x00\xef\x10p\t\xf0YY\xc6\x00\xed\x10p\t\xeeYY\xc6\x00\xeb\x10p\t\xecYY\xc6\x00\xec\x10p\t\xedYY\xc6\x00\xe5\x10p\t\xe6YY\xc6\x00\xe3\x10p\t\xe4YY\xc6\x00\xe3\x10p\t\xe4YY\xc6\x00\xe0\x10p\t\xe1YY\xc6\x00\xe5\x10p\t\xe6YY\xc6\x00\xe2\x10p\t\xe3YY\xc6\x00\xea\x10p\t\xebYY\xc6\x00\xe3\x10p\t\xe4YY\xc6\x00\xf2\x10p\t\xf3YY\xc6\x00\xe5\x10p\t\xe6YY\xc6\x00\xf0\x10p\t\xf1YY\xc6\x00\xe7\x10p\t\xe8YY\xc6\x00\xee\x10p\t\xefYY\xc6\x00\xeb\x10p\t\xecYY\xc6\x00\xea\x10p\t\xebYY\xc6\x00\xe4\x10p\t\xe5YY\xc6\x00\xeb\x10p\t\xecYY\xc6\x00\xe3\x10p\t\xe4YY\xc6\x00\xe6\x10p\t\xe7YY\xc6\x00\xe1\x10p\t\xe2YY\xc6\x00\xea\x10p\t\xeb'
```


This is our packet

Packets always start with

```
YY
```

and end with

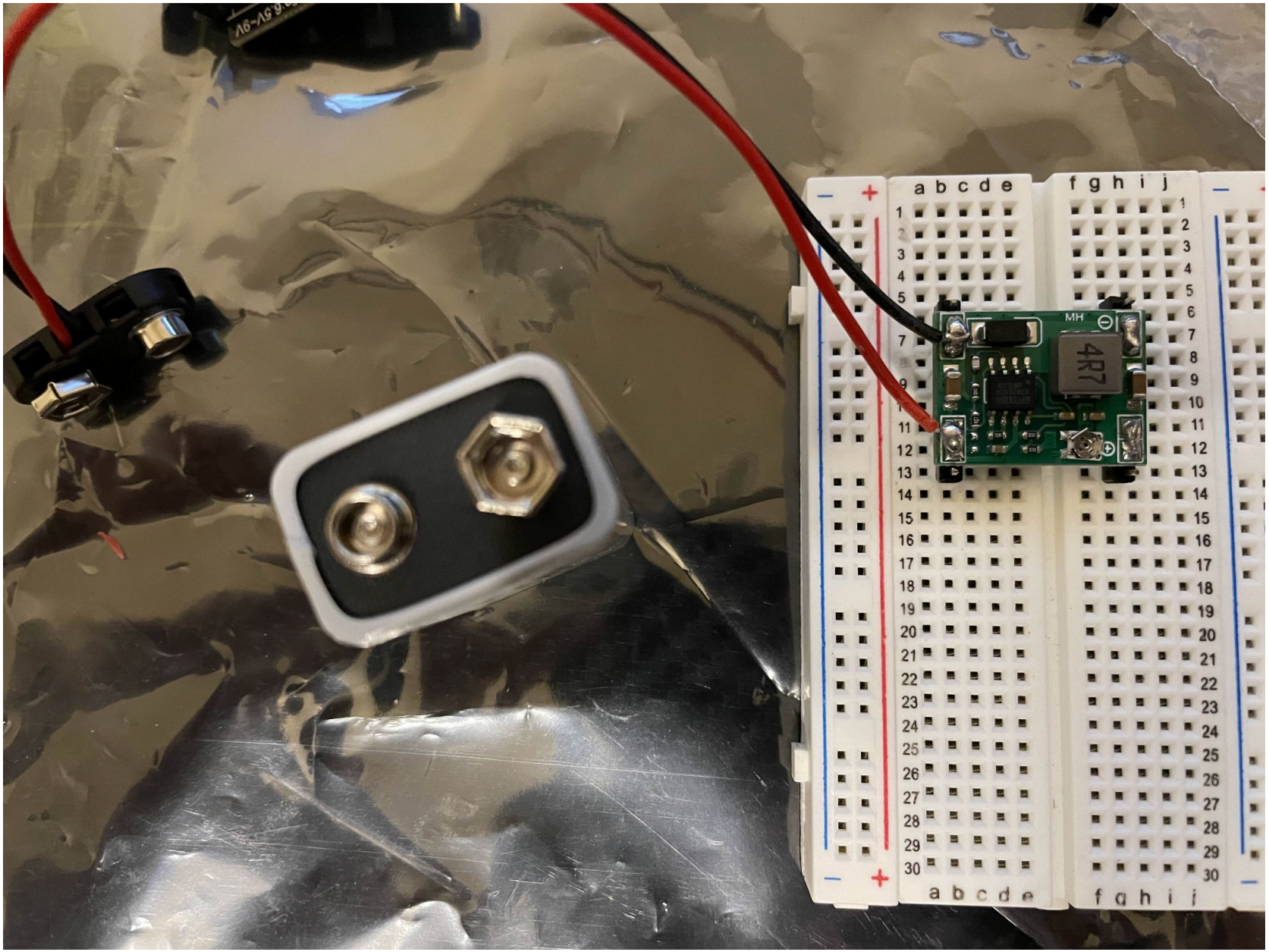
```
\x HEX HEX
```

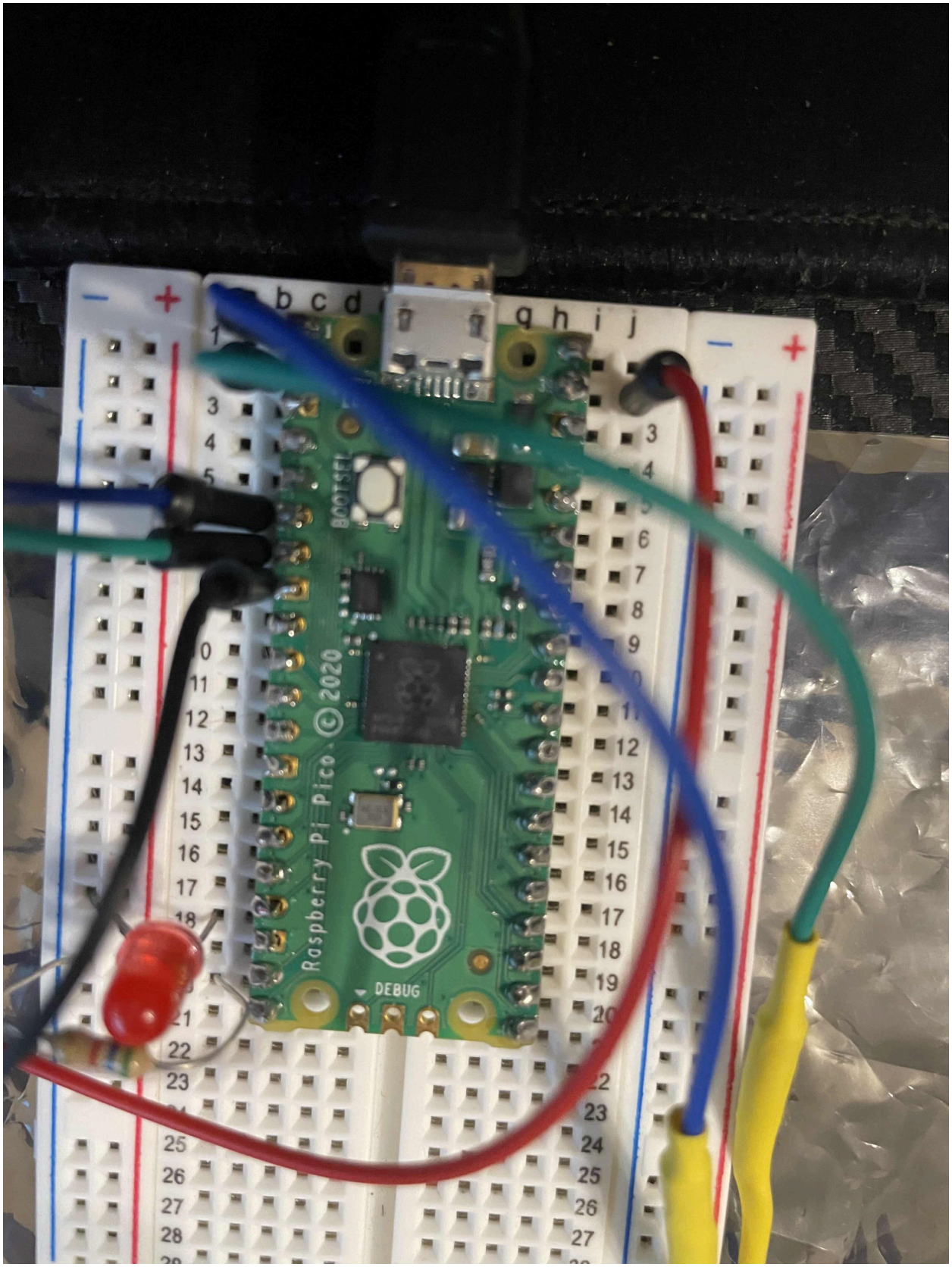

```
b"\x00\xff
'\xff\x00lvA\x0c\t\x00\x00\x00dd\x91\x17#\x00\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\
xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xf
f\x99\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\
xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xda\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xf
f\xff\xff\xff\xff\xff\xff#\xff\xff\xff\xff\xff\xff\xff-
\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff#\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\
xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xf
f\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff#\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff
\xff\xff\xff\xff\xff\xff9\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xffA\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff
$,#######$#####,*####;PB(##'-(1<3##*1/ZvX/DPA.&&9=53+/<8+####,5####-1-###09#G*#\xff#(43,#\xff######51$#*##''##00-
#######10####1>/###*0*5HB4--457<;>HJ###A?:;5###-3*&##&,*+&#######35B################0<&######0/*$####\xff####8=5&###+/&####-##9?
70.45CRK;;<#\xff#^F7A5###9>3/&###########\xff#'-\xff####$&#######,##4/######+96(###,#GpW#/<90'###/5&####(###/0/145>JB+*4##
(M>;@3'##CA########/H9\xff#&-&\xff#\xff###-*######0<H<.1(####.(#0;1''375|\xff~6?6#&(###><#####+,.+$#'03763,
(-5>7##3>ED9#&>;########HgG#\xff#-####E\xff#+/##&##(/-:A7*'##754###0.5EF5'`\x82a9D6#####0@=####,-5>6##&4595*#+35=
<.#&6@II:.+###$##$$#<M3\xff######58###,(####$###4/#(',90.0###.;HE4#+*##84&'**-4957@0#&(-51$#1:6583',0/4;=3561@JB-###-
B,#71######&##$H\xffH#.##'+&,###$#####.0'#.=0#+=FED=+&###(**09>73547<+##/35/*/85+4?9-.676:;>H=/>E>.###/=
(#.###,###3&\xff'c\xff\xff#####+#/'\xff#####\xff##$###+(&.;?:95$#######'8C?655/,,+3:94(##,5:5--./7=5*
(080,694*####&#\xff#\xff#1B#\xff#$##$<-###-,$##+8##\xff######$#####&47+$*$#&$*7-###4>A?;3+,7>>999,##*BM6##99363&####.51+#$.-
#'#####0=*###\xff####L/(>:$##(9#####/"
```


10/15/2024 (Tuesday, 11:10 AM, EABA Room 121) Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah

After a week of intense troubleshooting, the team successfully resolved the UART communication issues that had been hindering progress. Both the LiDAR and depth camera were now communicating effectively with the Raspberry Pi Pico, a crucial achievement for the project. The team verified that the data transmission met the project's stringent 0.5-second delay requirement, ensuring that the GUIDE system could provide real-time feedback to users.

With both sensors operational, the team began testing the sensor fusion capabilities of the system. By combining data from the LiDAR and depth camera, they aimed to improve the accuracy and reliability of obstacle detection. This multi-sensor approach promised to enhance the GUIDE system's performance in various lighting conditions and with different types of obstacles, potentially exceeding the original project specifications.


MH
4R7
a b c d e
f g h i j
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30


BOOTSEL
Raspberry Pi Pico © 2020
DEBUG

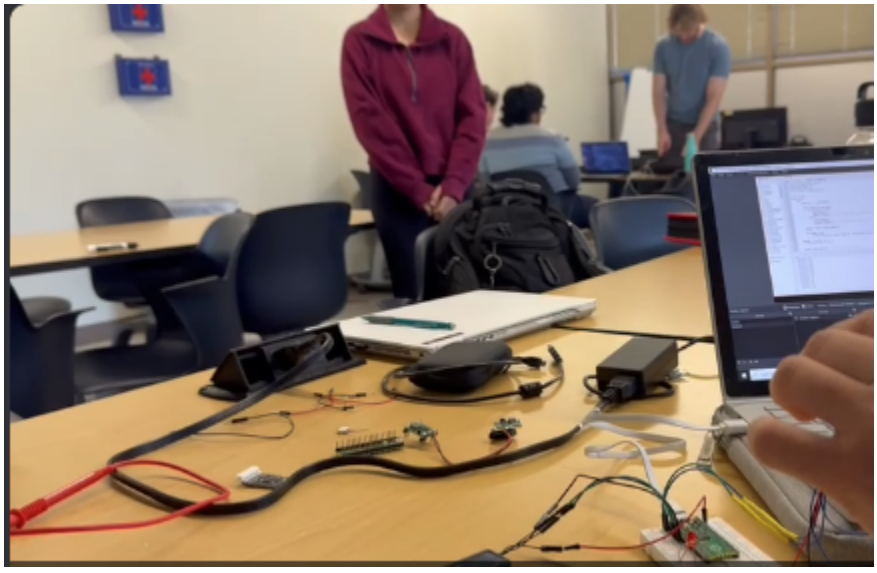
10/17/2024 (Tuesday, 11:10 AM, EABA Room 121) Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah

Completed soldering and component integration. Breadboard prototype fully assembled.

Began working on documentation for Critical Design Review.

The final testing and refinement of the breadboard prototype began in earnest. The team meticulously checked all connections, ensuring they were secure and properly insulated. Each component underwent rigorous individual testing to verify its functionality within the integrated system. This thorough approach was crucial for identifying any potential issues before the final assembly.

A comprehensive series of tests were conducted to verify the system's ability to detect obstacles at various distances, specifically at 2m, 1m, 0.5m, and 0m. The team paid close attention to the accuracy of detection and the appropriateness of the feedback provided at each distance. These tests were critical in ensuring that the GUIDE system would provide reliable and useful information to visually impaired users in real-world scenarios.

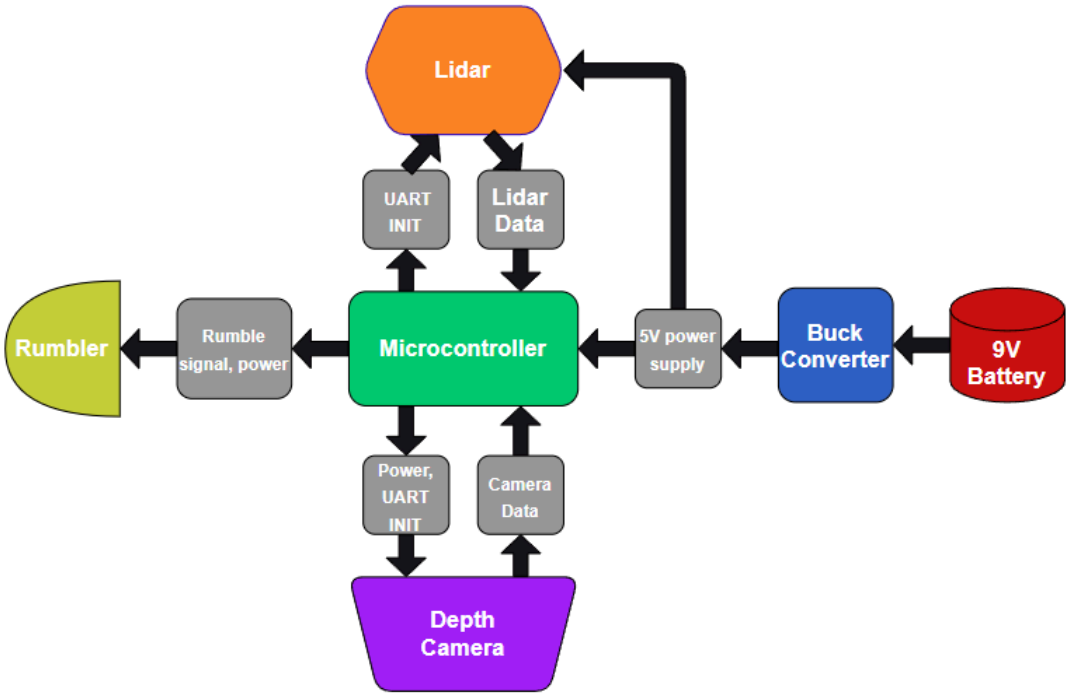


10/22/2024 (Tuesday, 11:10 AM, EABA Room 121) Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah

The handoff of the fully functional breadboard prototype to the software team marked a significant milestone in the project. This transition allowed the software team to begin fine-tuning the obstacle detection algorithms and user feedback mechanisms using the actual hardware.

The hardware team provided comprehensive documentation and guidance to ensure a smooth transition and continued collaboration between the two teams.

Simultaneously, the hardware team completed the detailed documentation for the Critical Design Review. This included comprehensive hardware specifications, step-by-step integration procedures, and preliminary testing results. The team also began planning for the crucial design validation phase, which would involve extensive testing of the GUIDE system in various real-world conditions across the university campus. These tests would be vital in proving the system's reliability and effectiveness in assisting visually impaired individuals in navigating complex environments.

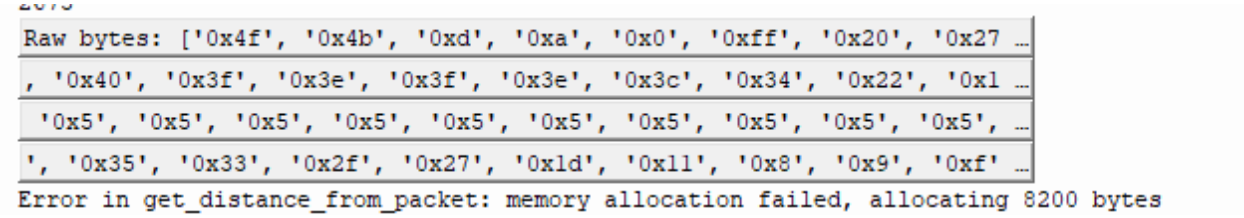


10/24/2024 (Thursday, 11:10 AM, EABA Room 121) Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah

Successfully implemented direct communication between the LiDAR and Raspberry Pi Pico today, eliminating our dependence on the UART bridge. The key discovery was identifying the packet structure - each valid packet begins with 'YY' and ends with specific hex values. This pattern recognition solved our data fragmentation issues.

Building on this success, I also established direct communication with the depth camera. Getting clean data from both sensors directly to the Pi represents significant progress. The next challenge will be implementing these on separate UART channels for simultaneous operation. The data quality improvement is substantial compared to our previous implementation.

The significance of today's progress cannot be overstated - we've essentially eliminated an entire layer of complexity from our system. The direct sensor-to-Pi communication not only reduces potential points of failure but also gives us much more control over our data processing pipeline. I've documented several test packets for future reference, as understanding these data structures will be crucial for our distance calculation algorithms.



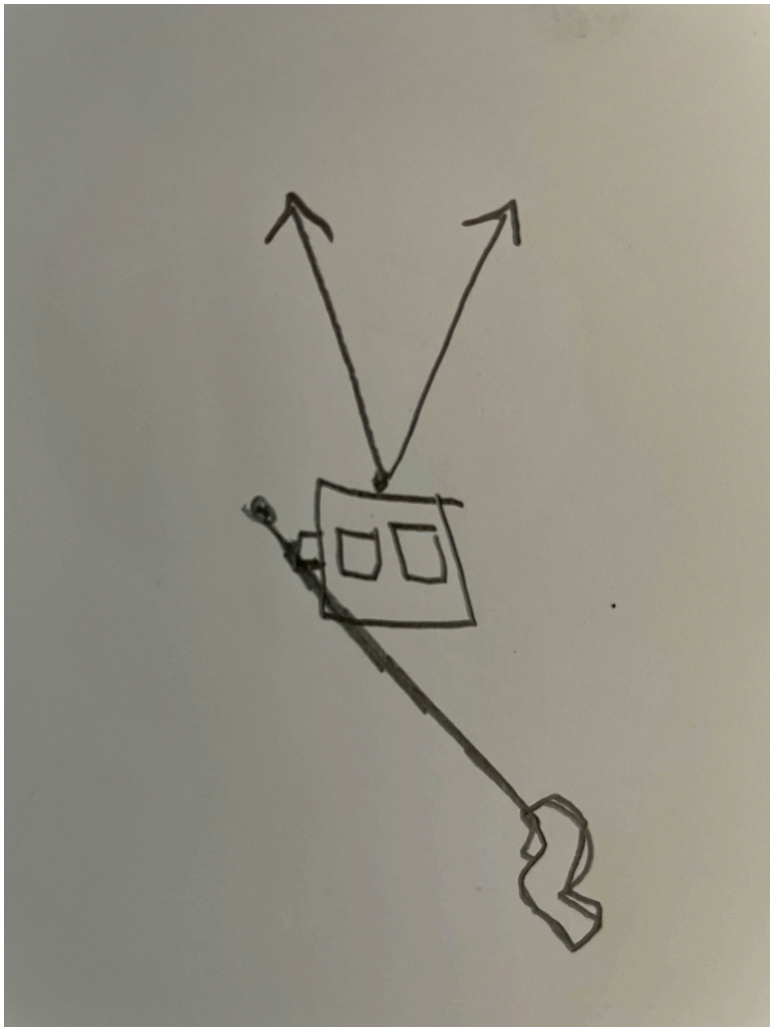
In this screenshot here, we initially were storing the data from the depth camera that appended without clearing the data from an array. Since the Raspberry Pi Pico 2 is an embedded system, we quickly hit a memory limit that would not have been an issue on a laptop or desktop computer. Of course, we cleared the array routinely during code runtime, fixing the problem.

10/29/2024 (Tuesday, 11:10 AM, EABA Room 121) Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah

Received and tested the buck converter today. Initial tests were successful, achieving 4.97V output from 9V input with good stability. While this converter works well for testing purposes, I've noted we'll need to think ahead for the switch configuration as well as how to correctly insert the device. The highlight was successfully powering all components simultaneously, validating our power distribution design.

A key concern throughout testing was maintaining stable voltage under varying load conditions. Our sensors have specific voltage requirements, and any fluctuation could lead to erratic behavior. The current setup demonstrates remarkable stability, but I've noted several potential

improvements for our final design, particularly regarding power efficiency. However, this ended up not being much of a problem or need to implement.



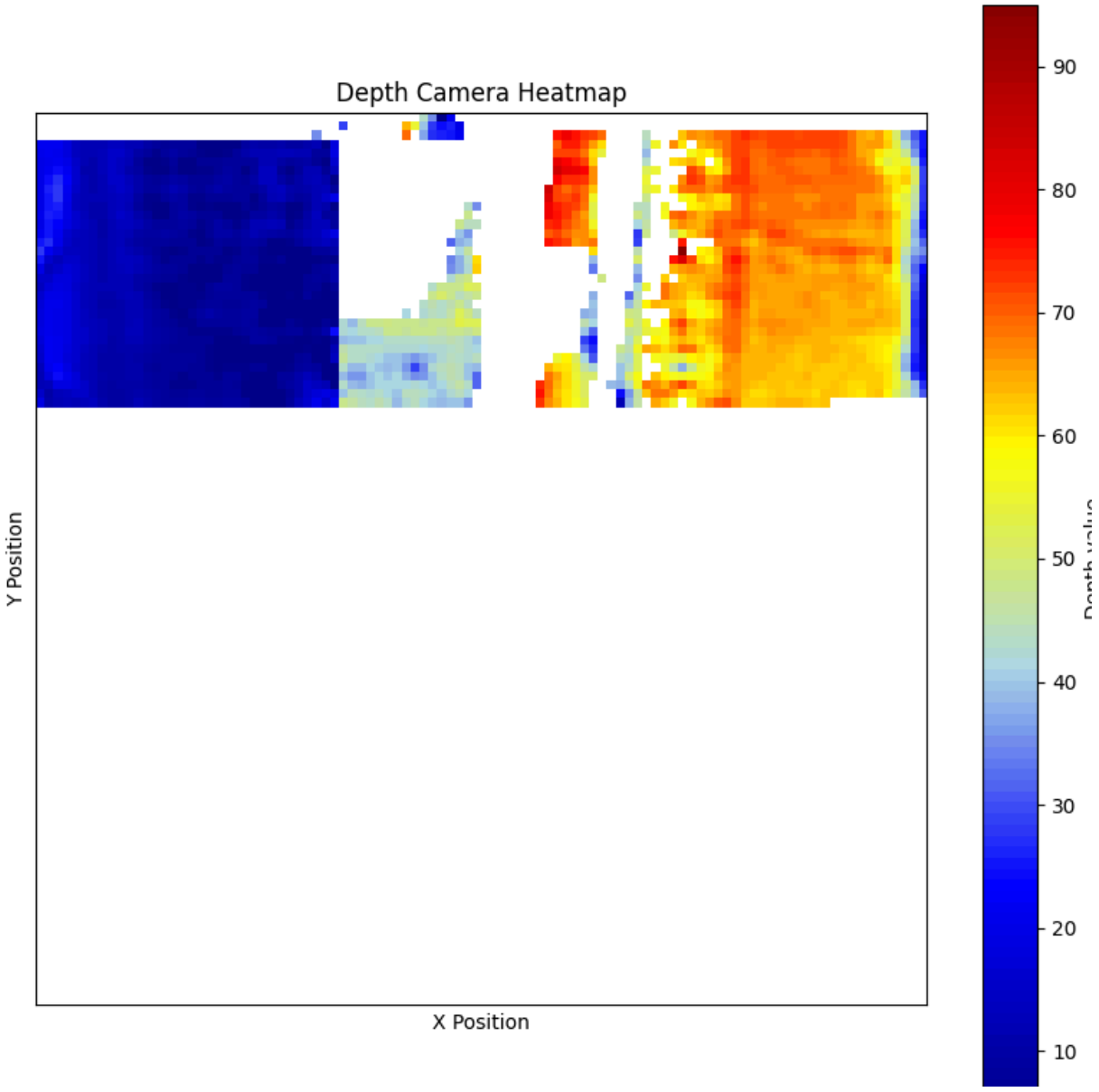
An initial sketch of mine showing an idea that was floated earlier in the semester- What if the two sensors were pointed in different directions at an angle for more data collection. While it was possible to have canted sensors- the 3D print and setup would have been more complicated.

10/31/2024-11/5/2024 (Thursday, 11:10 AM, EABA Room 121) Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah

Focused on sensor mount design optimization today. After analysis, determined optimal positioning requires the depth camera mounted parallel to the stick on top, with the LiDAR angled downward below it. Calculations indicate we need an angle between 15 and 25 degrees for optimal coverage - this should allow the depth camera to detect people at 1-3 meter ranges

while keeping the LiDAR focused on immediate obstacles and ground-level hazards. The theoretical model looks promising, though practical testing will be crucial.

After reviewing our angle calculations extensively, I believe we've found a configuration that maximizes our sensor coverage while minimizing blind spots. I've sketched several alternative mounting designs, taking into account both our technical requirements and manufacturing constraints. The top-bottom arrangement, while not immediately obvious, provides the best balance of detection capabilities while maintaining a manageable center of gravity for the overall device.



Very interesting diagram/illustration of the data captured by a single frame of the depth camera when hooked up to a desktop computer. The frame shows baggage (highlighted in yellow/orange on the right) close to the camera pinhole, while the left shows nothing close in range.

11/7/2024-11/13/2024 (Thursday, 11:10 AM, EABA Room 121) Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah

Made a breakthrough in depth camera data processing. Successfully identified the distance bytes in the data packets and developed a method for individual pixel indexing. However, this revealed significant processing limitations - the single-threaded nature of Python combined with ARM core constraints is causing notable latency in our processing cycle.

Encountered memory allocation limits during testing - an unexpected but informative setback. This has highlighted the need for more efficient data handling strategies. Our current processing approach will need optimization to achieve the response times we're targeting.

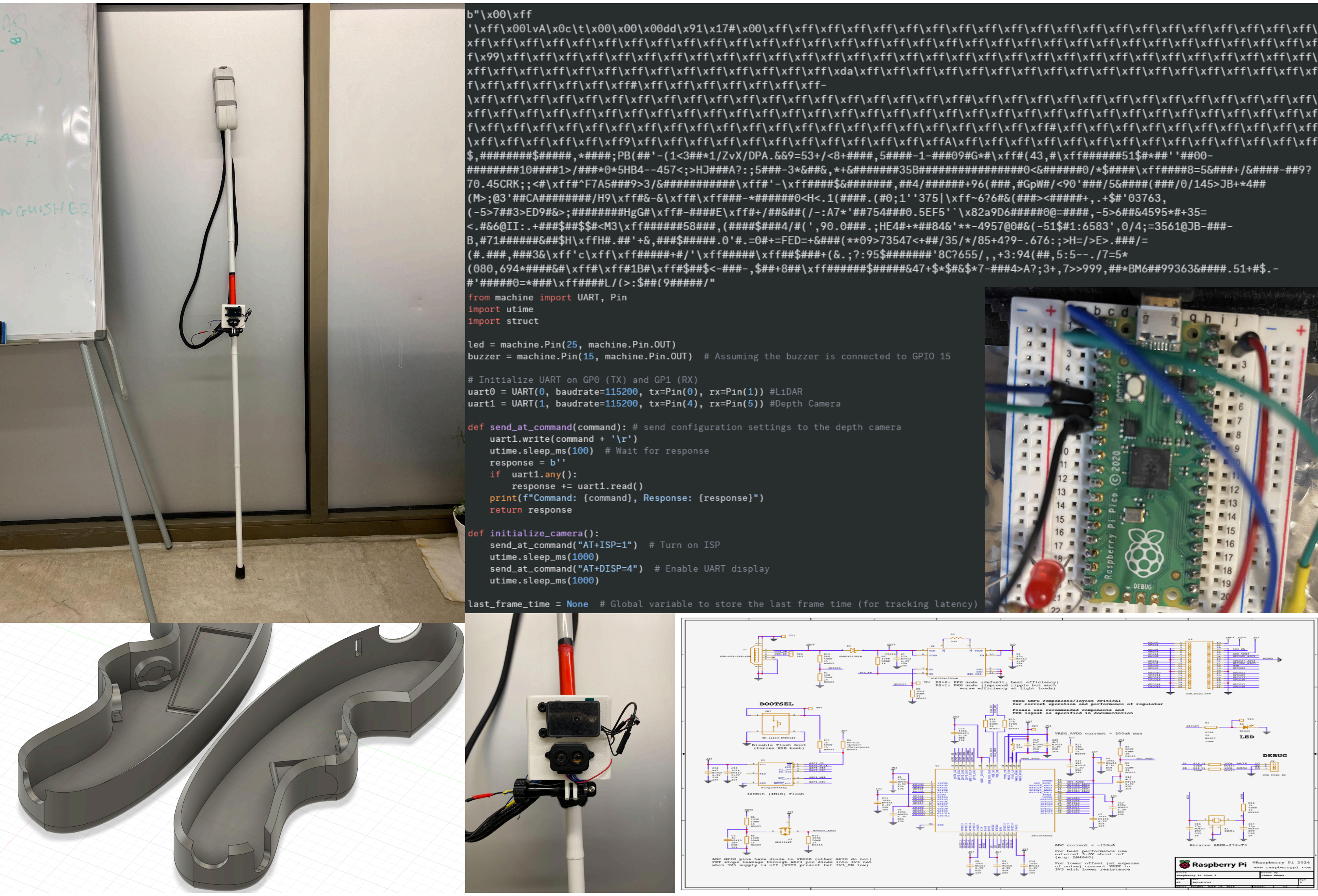
These limitations have led me to begin exploring alternative processing approaches. While full pixel-by-pixel processing would provide the most detailed environmental data, we may need to implement a sampling method that processes only key regions of the image. I've started documenting potential optimization strategies, including selective sampling patterns and more efficient memory management techniques. The challenge will be finding the right balance between processing speed and detection accuracy.

11/14/2024-11/19/2024 (Thursday, 11:10 AM, EABA Room 121) Attendees: Jack Letsinger, Jack Couture, Alyan Tharani, Diana Canchola, Ryan Wu, Noah

Completed final hardware assembly this week. All components are now integrated and communicating properly. Wire management proved challenging - had to modify the housing to prevent cable stress points. Spent considerable time verifying each connection and testing communication pathways.

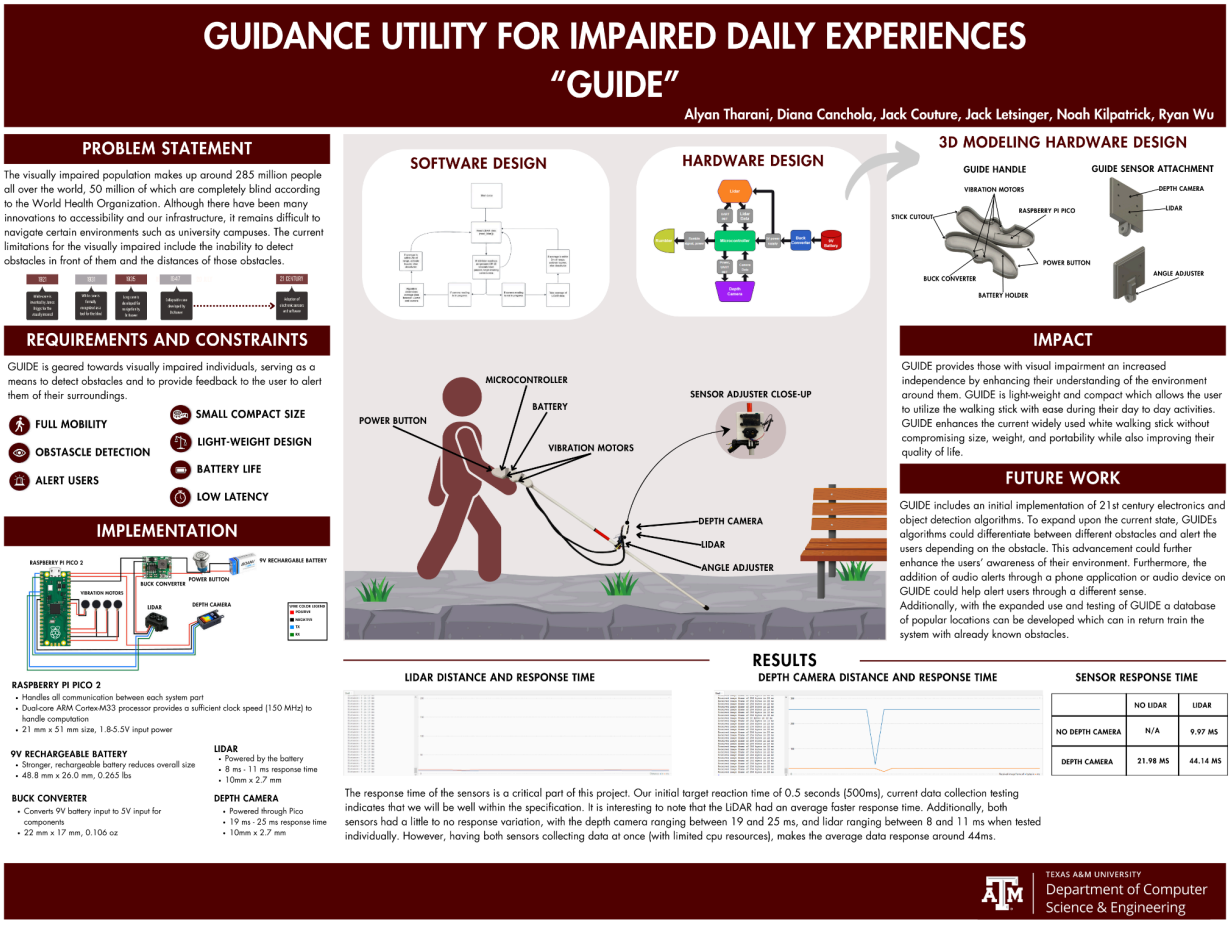
Seeing the complete integration of all subsystems is satisfying. The power management, sensor array, and processing systems are all functioning together as designed. While internal organization could be more elegant, the functionality meets our requirements. Some minor adjustments may still be needed, but the core functionality is solid.

The final testing phase revealed several interesting characteristics of our system that weren't apparent during component-level testing. The interaction between the LiDAR and depth camera data streams is particularly intriguing - when both sensors detect an obstacle, their data correlation provides a more robust detection than either sensor alone. I've begun documenting these synergistic effects, as they could be valuable for future iterations of the design (we chose to call this sensor fusion). The power consumption under full operation is also notably lower than our initial estimates, suggesting our power management design is more efficient than anticipated.



Collage of our progress throughout the semester.

11/22/2024 Friday - MSC



We chose to demo at the capstone expo held at the MSC. We got lots of good information and feedback from judges and classmates. (This poster presentation was made by Diana).