WALL-E
Fall Quarter Review

Franklin Tang, Karli Yokotake, Karthik Kribakaran, Veena Chandran, Vincent Wang, Wesley Peery

Hardware Components

Adafruit Feather Huzzah

The microcontroller used to communicate with external modules. Its purpose is to interface to the the microSD card reader module and the GPS module to record the timestamps of the video files when the video capture is triggered.

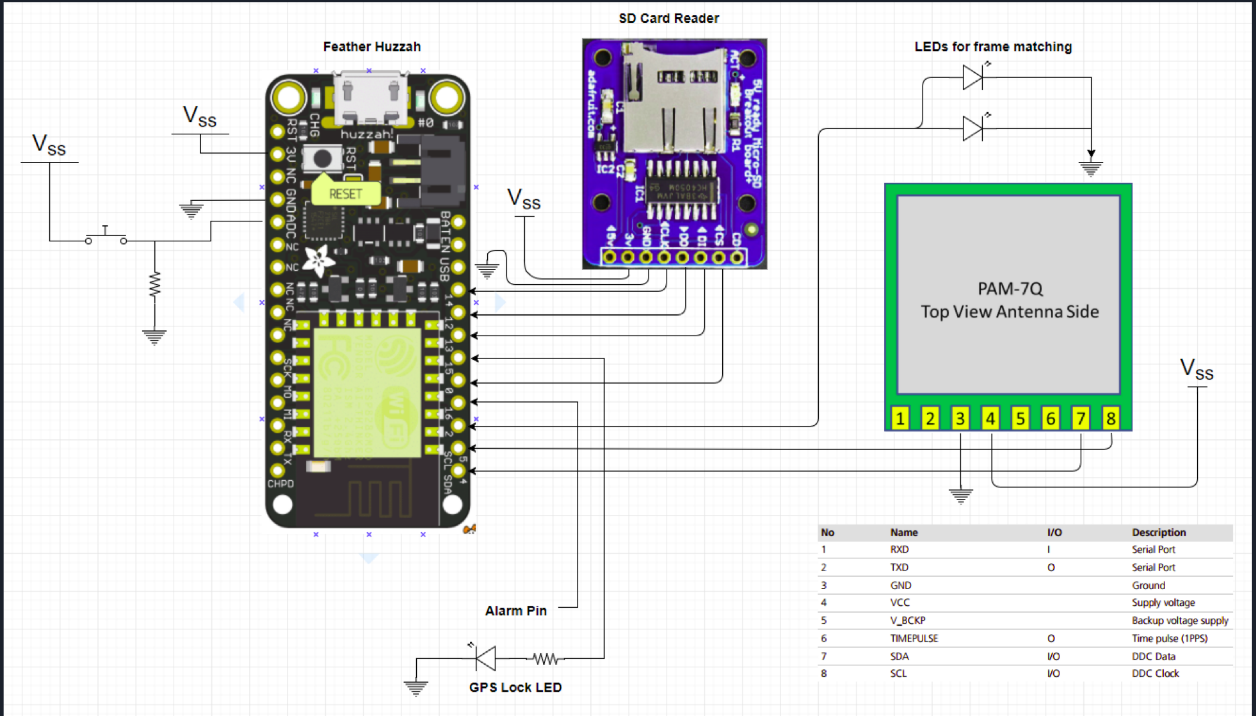
MicroSD card breakout board+

The card reader is used to write the timestamps of the video files into a text file on the microSD card. It interfaces to the Feather using SPI.

PAM-7Q-0 U-Blox GPS Module

The GPS module is used to get the current real time, latitude, and longitude to write to an SD card. It interfaces to the Feather using I^2C.

Hardware Design



No	Name	I/O	Description
1	RXD	I	Serial Port
2	TXD	O	Serial Port
3	GND		Ground
4	VCC		Supply voltage
5	V_BCKP		Backup voltage supply
6	TIMEPULSE	O	Time pulse (1PPS)
7	SDA	I/O	DDC Data
8	SCL	I/O	DDC Clock

State Machine

State 1: User starts in idle state. In this time, WALL-E will be searching for a GPS signal. Once it acquires a signal, we move into state 2.

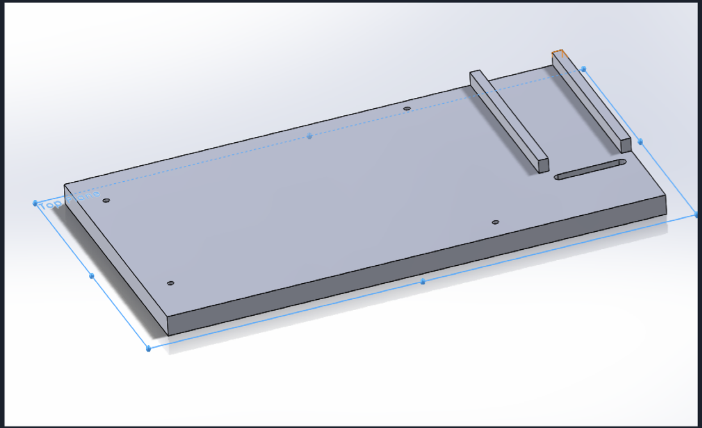
State 2: An LED in the main compartment of WALL-E will start blinking, indicating that the GPS is ready and the user can begin recording. Once the user is ready to record, they will push a button, which brings us to state 3.

State 3: At this time, the GPS output (time, latitude, and longitude) is written to the microSD and recording begins. Also, red LEDs in both camera cases flash 3 times for the purpose of synchronizing the two feeds. The user presses the button once more to move into the last state.

State 4: Recording ends, final state. Ready for recording again.

Mounting the Components

- Currently all of the components are loose inside of the acrylic case
- We will be 3d printing this (10”x 6”) tray to mount the pcb and battery
- The battery will mount in between the two raised rails



Post-processing CV Pipeline

Frame Matching (Intensity Gradient Approach)

Problem: Left and right video feeds are not guaranteed to be synchronized

Algorithm overview:

- Find intensity gradient of each video feed
- Find frame offset that minimizes gradient difference between each feed

Frame Matching (Intensity Gradient Approach) cont.

Left feed:

$$lf(x,\ y,\ n)\ \in\ \{0, 1, 2\ \dots\ 255\}$$

Right feed:

$$rf(x,\ y,\ n)\ \in\ \{0, 1, 2\ \dots\ 255\}$$

Gradient calculation:

$$G_{l,i} = (0.9)G_{l,i-1} + (0.1)\sum_{x=0}^{width}\sum_{y=0}^{height}|lf(x,\ y,\ i) - lf(x,\ y,\ i+1)|$$

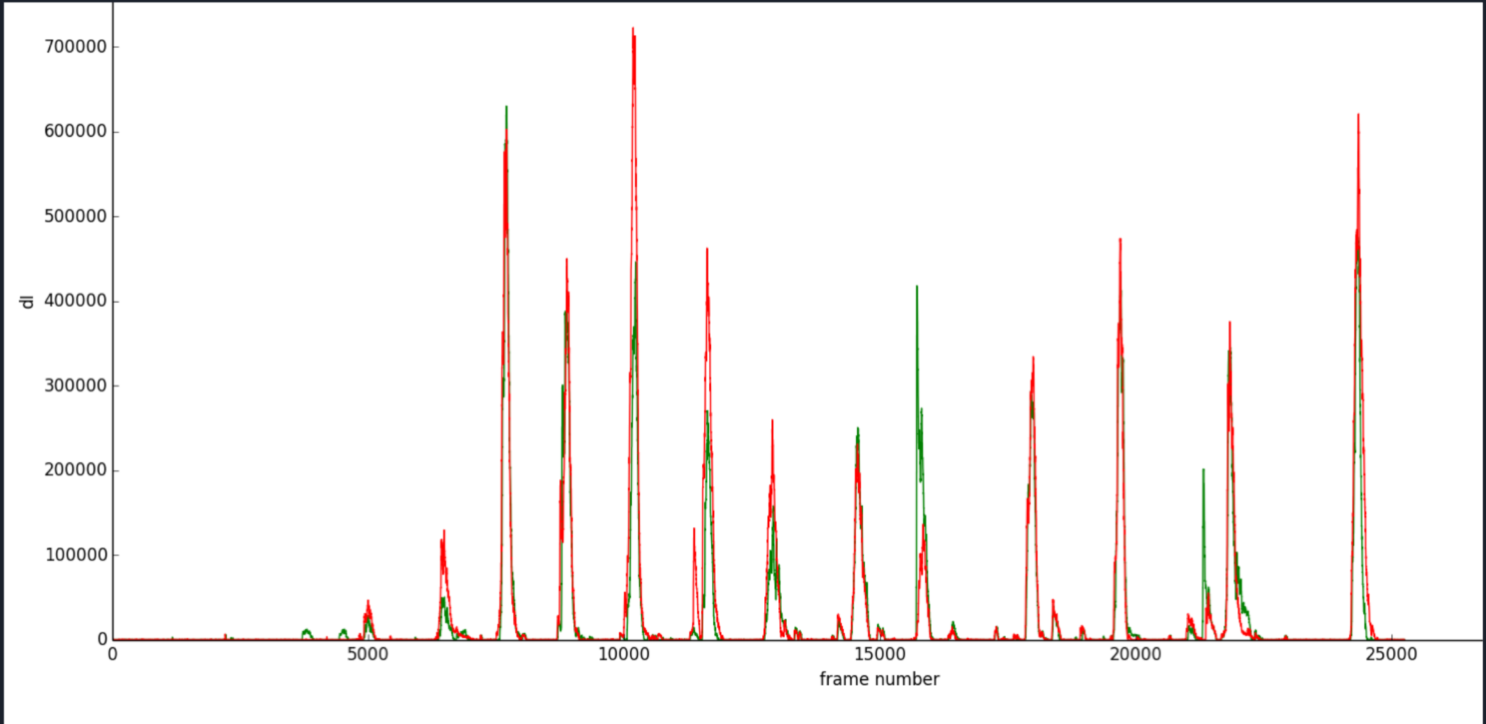
$$G_{r,i} = (0.9)G_{r,i-1} + (0.1)\sum_{x=0}^{width}\sum_{y=0}^{height}|rf(x,\ y,\ i) - rf(x,\ y,\ i+1)|$$

Frame Matching (Intensity Gradient Approach)

Find offset value that minimizes the following equation:

$$\frac{1}{numframes - offset_r} \sum_{i = max(0, offset_r + i)}^{min(numframes - 1, numframes + offset_r)} \left| G_{r,i} - G_{l,i} \right|$$

Frame Matching (Intensity Gradient Approach)



Frame Matching (Intensity Gradient Approach)

Frame Matching (LED Approach)

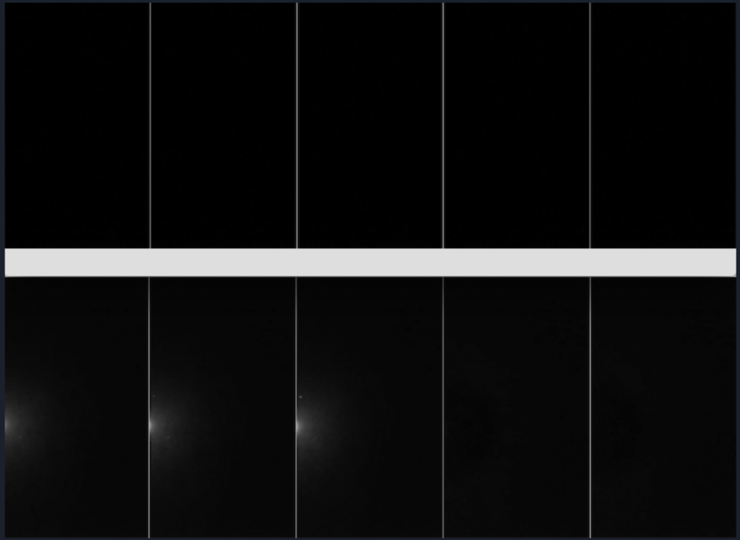
Hardware:

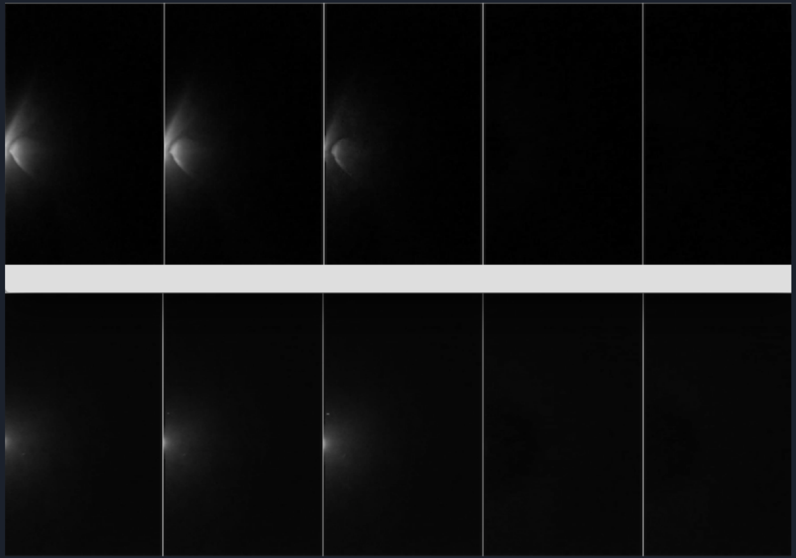
- One LED in each camera tube
- Flash LEDs at the same time during the first few seconds of recording

Algorithm Overview:

- Builds on previous algorithm
- Calculate the intensity gradient of the first 500 frames of each feed
- Find the frame largest increase in intensity of each video
- Calculate the offset between the frames

Frame Matching (LED approach)





Before

After

Stereo Rectification

- Goal: To align the left and right images so that their Y axis align
    - An object that is captured by both of the stereo cameras will have the same Y coordinate
    - There will only be offset between the X values
- Benefits:
    - Point matching becomes easier
    - It becomes easier to calculate the (x,y,z) coordinate of the object

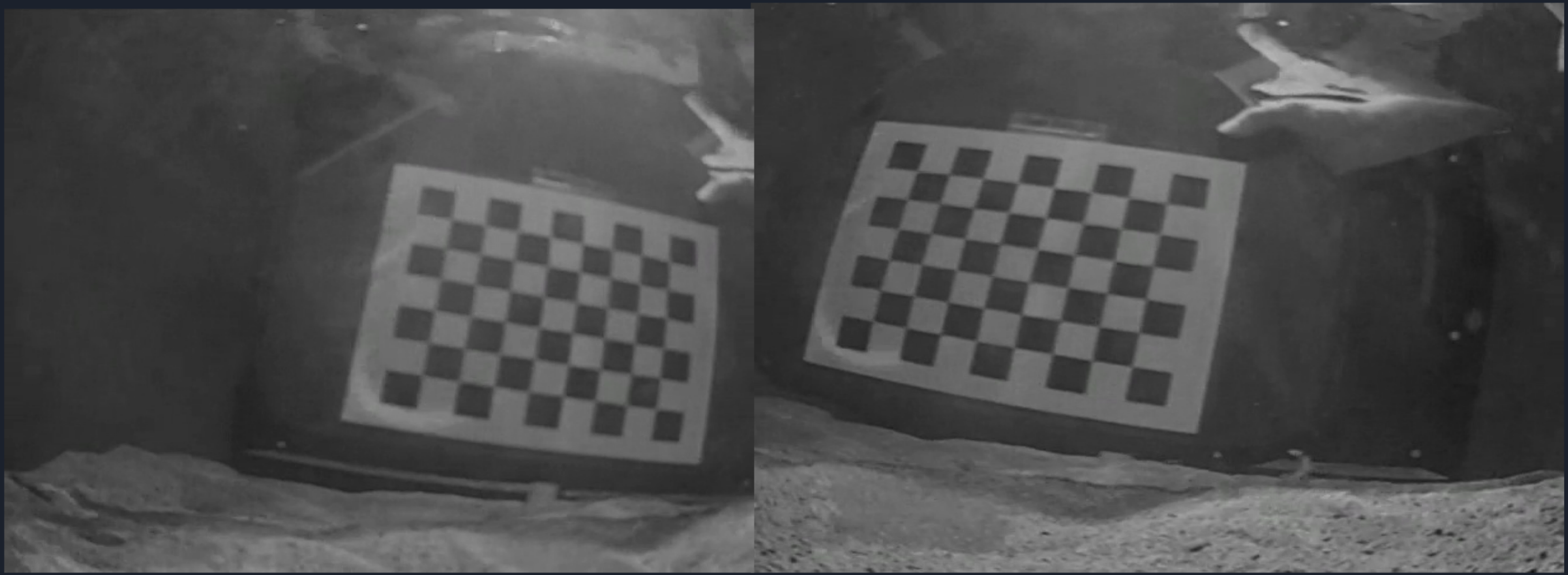
Stereo Rectification - Checkerboard Initialization

- We used a checkerboard to fix camera rotation and y-axis discrepancies in video feeds
- Utilized visual cues in checkerboard to orient the video feed frames correctly
- This involved taking footage of a checkerboard in a controlled environment
- Accomplished with the help of OpenCV fisheye library

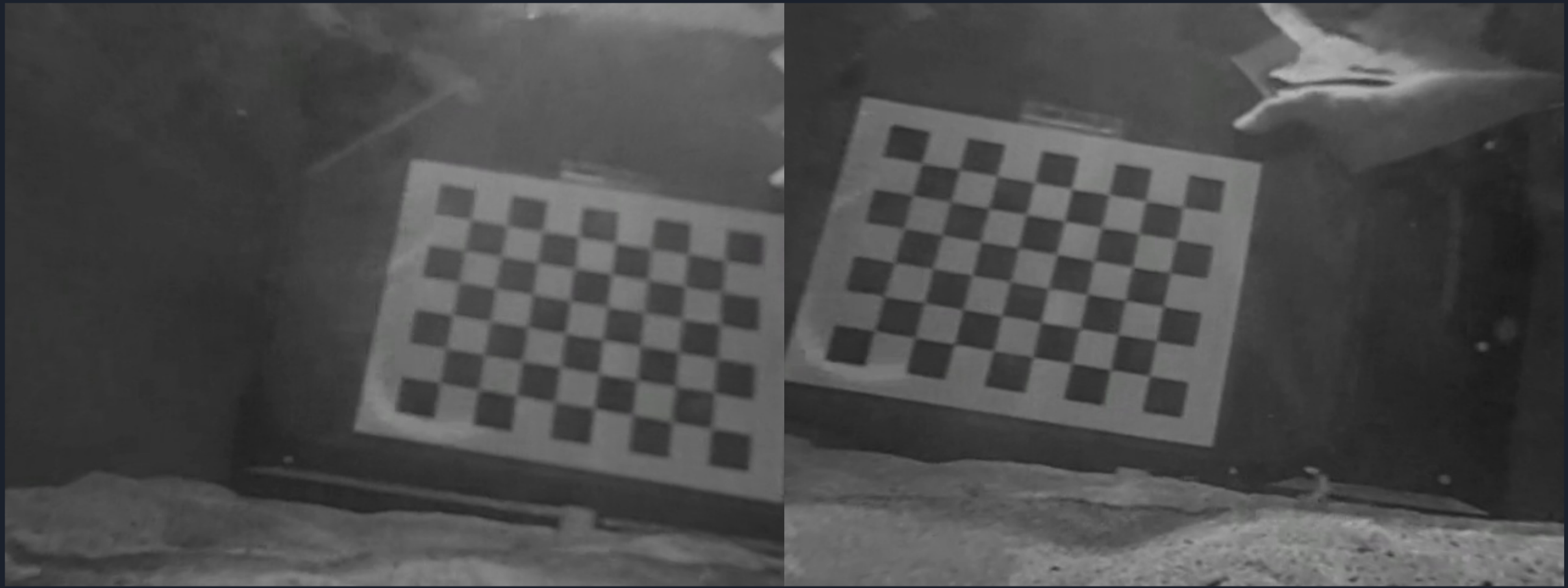
Results of Stereo Rectification

- It was a challenge to generate quality results, it required a lot of tinkering with the openCV scripts
- We achieved the best results after we deinterlaced the videos (changing the resolution from 680x480i to 680x478p)
- We also wrote a script that took two videos and applied the transformations on each frame

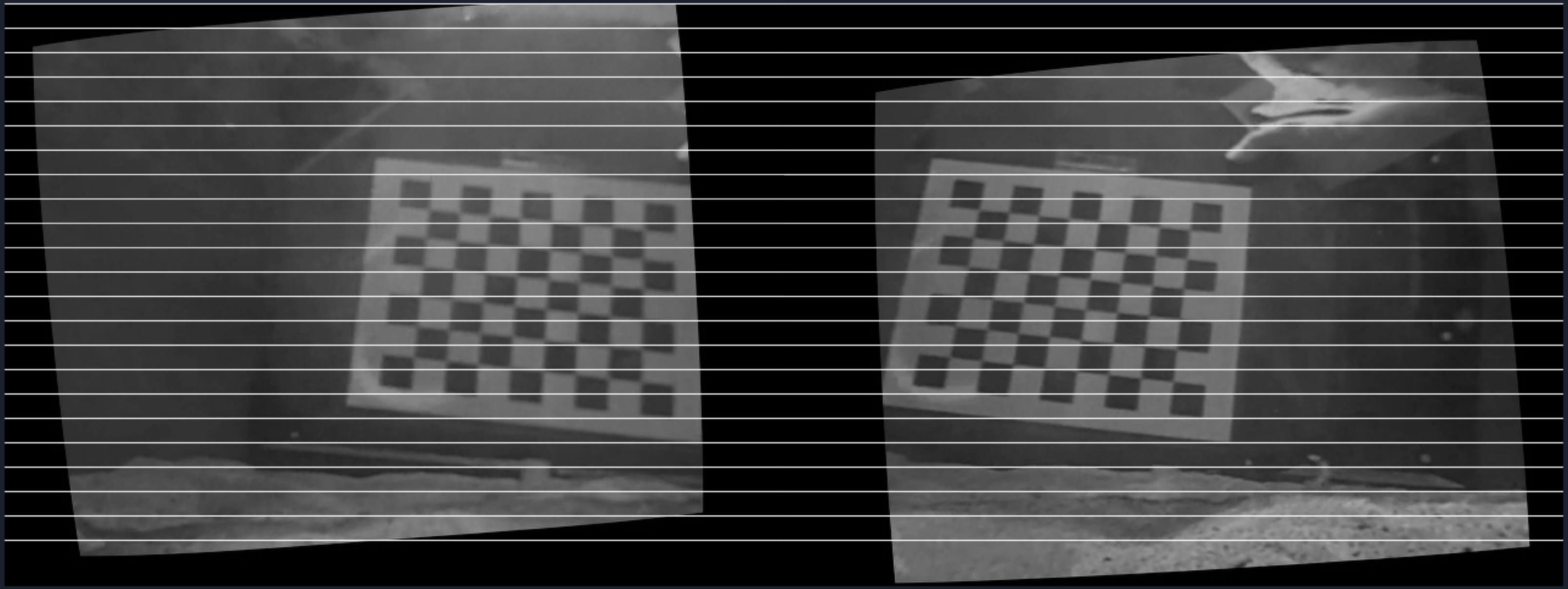
Stereo Rectification Results (original)



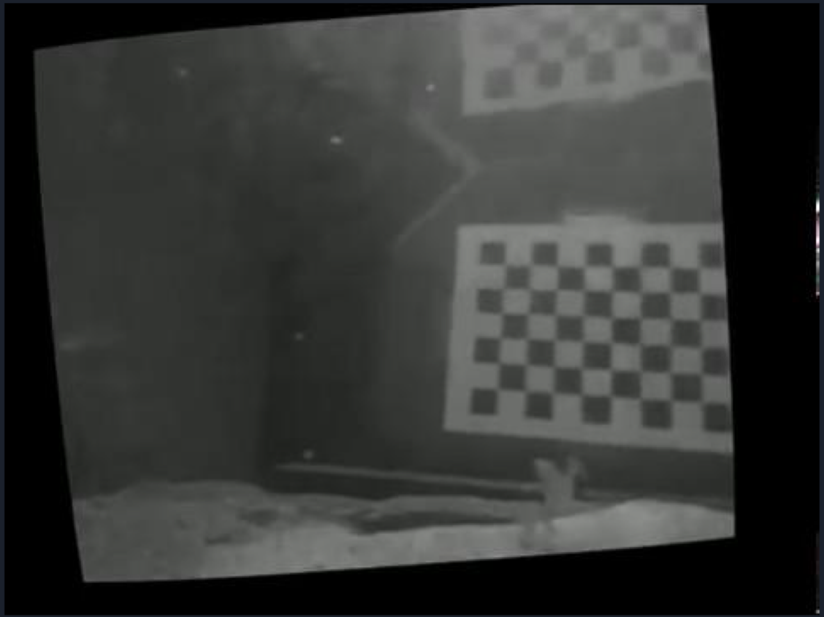
Stereo Rectification Results (undistorted)



Stereo Rectification Results (stereo rectified)



Two Stereo Rectified Videos



Special thanks to:
Yoga
Professor Oakley
Caio
Celeste
Trinity Locker-Cameron

Questions?