UNIVERSITY OF CALIFORNIA SAN DIEGO

Systems Design of Underwater 3-D Camera Modules

A thesis submitted in partial satisfaction of the requirements
for the degree Master of Science

in

Computer Science

by

Patrick Richard Paxson

Committee in charge:

Professor Ryan Kastner, Chair
Professor Patrick Pannuto
Professor Brice Semmens

2022

The thesis of Patrick Richard Paxson is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2022

EPIGRAPH

Fish farmin' ain't easy!

—Luis E. Rodríguez Rivas

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I'd like to thank my parents, without whom I never would have gotten this far. I'd also like to thank my brother, the best mechanical engineer I know. I'm excited to see where you go.

Thank you to Dr. Kastner, for completely changing the course of my time at UCSD, enabling me to explore underwater robotics research, and providing fantastic guidance on leadership, technical work and communication. Thank you to Dr. Semmens, without whom this paper wouldn't exist, and thank you to Dr. Pannuto for joining my advisory committee and supporting my work. Thank you to Nathan Hui, whose guidance and patience were invaluable in bringing our designs to life, and thank you to Peter Tueller for bringing me into this project and sticking with me every step of the way. And thank you to Donovan Drews, my OG friend and one of the most dedicated engineers I know.

Thank you to the entire FishSense team, past or present for all of your incredible hard work, dedication and passion throughout these past couple years, and thank you to everyone on Triton Robosub who believed in me.

Finally, thank you Catherine for being by my side through all of this. You're the absolute best, and I couldn't have done all this without you!

Chapter 2 contains unpublished material coauthored with Avalos, Alberto. The thesis author was the primary investigator and author of this material.

Chapter 3, in part, is currently being prepared for submission for publication of the material. Tueller, Peter; Maddukuri, Raghav; Paxson, Patrick; Suresh, Vivaswat; Miao, Albert; Drews, Donovan; Paxson, Charles; Semmens, Brice; Kastner, Ryan. The thesis author was the primary investigator and author of this material.

Chapter 4, in part, is currently being prepared for submission for publication of the material. Tueller, Peter; Maddukuri, Raghav; Suresh, Vivaswat; Miao, Albert; Drews, Donovan; Paxson, Charles; Semmens, Brice; Kastner, Ryan. The thesis author was the primary investigator and author of this material.

Chapter 5, in part, is a reprint of the material as it appears in IEEE/MTS OCEANS 2021. Tueller, Peter; Paxson, Patrick; Maddukuri, Raghav; Suresh, Vivaswat; Ashok, Arjun; Bland, Madison; Wallace, Ronana; Guerrero, Julia; Semmens, Brice; Kastner, Ryan. The thesis author was the primary investigator and author of this material. Chapter 5, also, in part is currently being prepared for submission for publication of the material. Paxson, Patrick; Tueller, Peter; Maddukuri, Raghav; Suresh, Vivaswat; Miao, Albert; Drews, Donovan; Paxson, Charles; Semmens, Brice; Kastner, Ryan. "FishSense: 3D Capture and Analysis for Fish Detection and Stock Assessment". The thesis author was the primary investigator and author of this material.

ABSTRACT OF THE THESIS

Systems Design of Underwater 3-D Camera Modules

by

Patrick Richard Paxson

Master of Science in Computer Science

University of California San Diego, 2022

Professor Ryan Kastner, Chair

Methods for measuring fish populations are incredibly outdated. Even with rapid improvements in battery technology, compute power and camera systems in the last few years, the most popular techniques for counting fish and measuring their lengths employ humans. In this paper, I present FishSense as an alternative. By utilizing a compact 3-D camera, powerful embedded compute system, and easily-available off-the-shelf parts for assembly, the FishSense modules are sleek, cost-effective and modular systems able to slot into a wide variety of applications. By capturing both depth and color image data, these modules are able to measure the length of individual fish as well, offering an effective non-invasive solution to length & biomass estimation. The hardware design allows for a high degree of modularity, with options for longer battery life,

better performance, and applications in longer-term passive monitoring. The FishSense handheld model has been rigorously tested in several deployments, ranging from a small testing pool with toy fish, to a 70,000 gallon aquarium tank, to an active aquaculture center, with more planned in upcoming months, and preliminary software results from these deployments demonstrate the efficacy of the FishSense modules at measuring fish populations and sizes.

Figure 0.1: The FishSense logo, designed by Catherine Murphy.

Chapter 1

Introduction

1.1 Background

1.1.1 The Problem

Humans are drastically taxing the ocean's fish populations. In 2017, 34% of the total fish caught were harvested from overexploited populations, meaning that that local population of fish didn't have enough healthy adults to replenish their numbers sustainably [4]. In other words, that's 60 million tonnes of fish coming from populations that couldn't afford to lose those individuals, which equates to $136 billion of market value [5]. As global demand for seafood has nearly quadrupled in the last 50 years, this substantial yearly taxation of fish populations is causing entire wild fisheries (natural hotspots in both freshwater and saltwater that are home to many varieties of fish) to dry up, and in coastal ecosystems over 90% of fish populations had suffered a population decrease of over 50% [6], with 38% suffering losses of over 90% and 7% becoming extinct entirely.

It's worth noting that the marine population drops are not just caused by overfishing. Ocean acidification through increased carbon dioxide in the environment due to human fossil fuel

use is interfering with the metabolism and immune responses of fish we eat, and is a leading factor in coral reef bleaching [7], an issue which affects a billion people's food supply and livelihoods [8]. Microplastics introduced by garbage pollution have coated the entire marine ecosystem, from the sea breeze to the beaches to the water to fish living all the way in the Marina Trench [9], and the bioaccumulation of microplastics pose significant risks to fish populations, humans [10] and the food chain at large [11]. Thousands of other studies have been conducted in recent years that document other catastrophic effects of human activity on marine life.

However, regardless of the causes, it is clear that managing fish populations is essential to preserving an important nutrition source for billions of people. In fact, the past two decades have shown it; aquaculture centers (fish farms) have taken on the bulk of the rising demand in fish, and since 2012 have accounted for a higher percentage of the world's yearly fish production than wild fisheries [4]. And in fisheries, both man-made and wild, where the fish are being closely monitored and reported, populations have been stabilizing or even improving. The effort that people are putting in to try and improve population monitoring is working, and there is an increasing focus on bringing in cutting-edge new technology to more efficiently conduct these surveys, which brings us to the question: how are fish monitored today?

1.1.2 Current Solutions

Fisheries assessment today is conducted almost entirely by humans, and relies very heavily on either direct observation and measurement, or monitoring via video. Some use catch-and-release to measure fish, where fishers catch individuals, take them aboard a boat, and measure them directly with a ruler or scale before releasing them back into the ocean [12, 13, 14]. While this method does serve its primary purpose of accurately measuring fish length, it is incredibly invasive and taxing on the fish's health, and it is time-consuming as only a few fish can be measured at a time.

Other fisheries employ divers to conduct censuses of fish populations [15, 16]. Oftentimes,

these divers are sent with nothing but a ruler and clipboard, expected to count, measure, and track hundreds of fish. The complexity of this task, and the limited capabilities of the tools means that more time must be invested into training, and requires hours upon hours of diving time to completely evaluate even small areas. Some divers are able to employ tools like stereo vision rigs, as seen in Figure 1.1, which amount to little more than two GoPros set a fixed distance from one another. While this setup is easier to manage than a ruler and clipboard, the stereo vision rig is unwieldy because of how far apart the cameras must be placed, applying postprocessing on the data collected takes time and expertise, and length measurements can be inaccurate even with perfect calibration. Still others use laser-based devices, which may lend more accuracy to length measurements but require the fish to remain still, which rarely happens naturally.

Figure 1.1: Diver with a stereo GoPro camera rig. Photo by Tiago Peixoto, copyright Grouper Moon Project 2020.

Even more approaches to counting fish involve manually annotate underwater videos

collected either from fixed cameras set up in aquaculture cages or videos captured from rigs like the stereo vision rig described above [17, 18, 19, 20, 21, 22, 23]. Just like with the previous methods, assigning humans to manually study hours of content is very time consuming and taxing on the people tasked with counting the fish in the videos. Others don't even utilize videos, instead assigning someone to count fish passing through a section of river from atop a 20 foot tower. And with these methods, information like length are all but lost unless a stereo vision rig was utilized.

The most cutting-edge technology currently available today, like the Open Ocean Cam[1], begin to introduce AI into the equation, but the use of only one imager means that accurately measuring the length or biomass of fish becomes an incredibly challenging task. Meanwhile, fisheries scientists advocate for better data collection methods [24], and as the AI revolution has proven, if there is big data than machine learning will follow.

In interviews I conducted across the course of this project with researchers, scientists, and fisheries managers across the United States, I heard firsthand accounts of these methods, often established upwards of half a century ago, still in use today. And across the board, when faced with the prospect of modernizing the counting and measurement techniques, my interviewees cited the extremely tight budgets and lack of technological expertise among the top reasons to stick with the tried-and-true methods. And indeed, there are thousands of small-scale, local fisheries around the world that simply do not have the money or resources to invest in utilizing machine learning or other automated counting techniques, let alone automated biomass estimation.

One such local aquaculture center that we tested a FishSense module at spent most of their time, manpower and budget on counting and maintaining their fish populations. With over 50 farming cages (Figure 1.2) situated around an island miles into the Pacific Ocean, divers made weeks-long rotations around the island to visit each cage, attempting to count and monitor the health and average biomass of the fish in each one. This data collection was crucial to the success of the entire operation, as the company spent approximately two-thirds of their overall revenue

[1] https://www.openoceancam.com

Figure 1.2: Some of the cages on the east side of the island. Each cage used netting to enclose tens of thousands of fish, and groups of cages surrounded three sides of the island.

on just feeding the fish! Appropriately budgeting food to ensure that each cage got exactly the right amount of food was a core part of the fishery's daily operations, and took dozens of divers as well as a crew of people monitoring camera feeds all day. The massive amount of man-hours, resources, and plain old hard work that went into fish farming made a lasting impression on me, and served as a primary motivator for me in pursuing a solution to their massive scale issue.

The technology employed in oceanic applications are years or even decades behind what can be found on land. The next section will delve further into the progress made in more conventional applications of machine learning.

1.1.3 Current Technology

In the past 10 years, we have seen an explosion of development and innovation in the fields of machine learning and computer vision. When AlexNet [25] won the ImageNet competition in 2012, Alex Krizhevsky and his colleagues made history by winning with the first image classifier

that utilized deep Convolutional Neural Networks (CNNs) ever entered in the competition. This was made possible by Krizhevsky's use of Graphics Processing Units (GPUs) during training to parallelize the expensive computations needed, which even then took 5-6 full days to train on two Nvidia GTX 580 GPUs.

While the individual concepts of CNNs [26] and utilizing GPUs to speed up their training [27] weren't new, Krizhevsky's innovative 8-layer model architecture to optimize training speed blew past the competition, and was only surpassed three years later by an even bigger, deeper 100 layer CNN from Microsoft [28]. AlexNet's astounding 10% accuracy edge over the competition shined a bright spotlight on the power of deep neural networks, value of parallelization using GPUs (and nowadays Tensor Processing Units (TPUs)[2]), and the development of even faster, more efficient embedded computing systems.

This new attention led academic institutions, tech companies, and passionate engineers to begin applying Artificial Intelligence (AI) to everything from natural language processing, to playing games, to image processing, and models got increasing more powerful, layered and complex with each passing day. OpenAI researchers quantified this growth in computing cost, finding that while Moore's Law only projected three doubling periods for the number of transistors in use in new computers (an 8x increase in hardware computing power), deep learning models have gotten over 300,000 times more powerful from just 2012 to 2018! [1] This growth can be visualized in Figure 1.3.

This mind-boggling increase in computing capability can be attributed to a number of advancements in both hardware and software; while researchers continued developing clever new model architectures, companies like Google and Nvidia pushed the boundaries of how fast they could compute large matrix multiply-adds (a crucial computation in AI, graphics processing, and more). Libraries like CUDA[3] or OpenCL[4], which provide developers with a framework for taking

[2]https://cloud.google.com/tpu/docs/tpus
[3]https://developer.nvidia.com/about-cuda
[4]https://www.khronos.org/opencl/

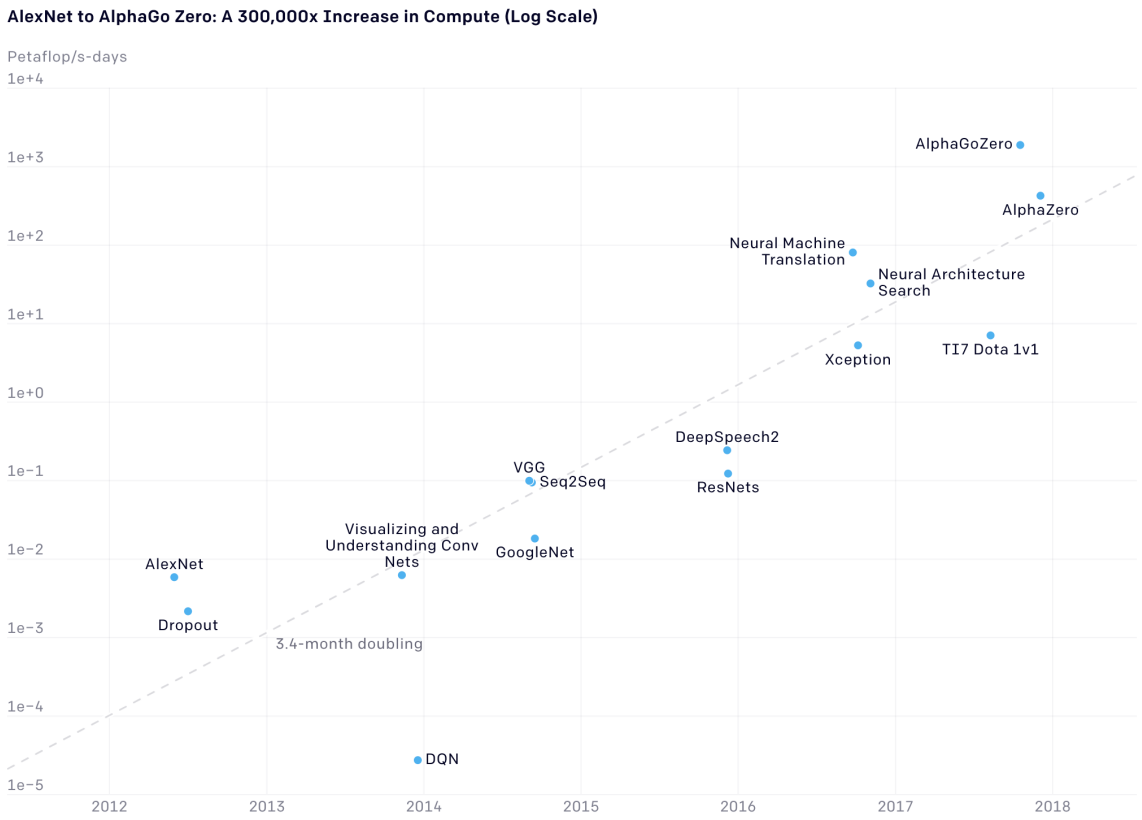


Figure 1.3: Log-scaled graph of the increase in computing cost for landmark AI networks developed from 2012 to 2018. Copyright OpenAI [1].

advantage of hardware acceleration and parallel computation, started becoming cornerstones of any machine learning engineer's toolkit. Meanwhile, new GPUs were being snatched up by researchers, cryptocurrency enthusiasts and gamers, lending greater importance to innovation on the hardware side to meet an increasingly hungry market.

This hardware revolution has not been limited to GPUs, either. As the ceiling for what volume of data can be processed continues to increase, so too do our methods for collecting this data. While single-lens cameras and basic images or videos have remained a popular subject of study, one major field that has been quickly growing in the past few years has been operating on 3-dimensional (3-D) data, which began with data collection from the Microsoft Kinect [30] in 2010. This video game peripheral, originally aimed at detecting player movement, was quickly

(a) (b)

(c) (d)

Figure 1.4: Recent 3-D sensors: (a) Microsoft/Xbox Kinect, released in 2010 for ∼$150 and discontinued in 2017 [29]. (b) StereoLabs Zed 2, released in 2020 for ∼$450. (c) Intel Realsense D455 Depth Camera, released in 2020 for ∼$240. (d) Intel RealSense L515, released in 2019 for ∼$600.

adopted by researchers, roboticists, and even medical professionals because of its availability, low cost, and integrated depth sensor. More recently, and with the retirement of the Kinect and other older Xbox development, StereoLabs' ZED camera[5], Intel RealSense's D400 series depth cameras[6] and other 3-D cameras have been seeing commercial success in a wide variety of applications for their ability to record depth information alongside Red/Green/Blue color (RGB) images in one small package. Intel RealSense[7], Apple [31], and other companies have also been creating relatively low-cost LiDAR sensors, enabling more people to get their hands on the technology, although this has reached a smaller market due to the higher cost, among other things. A few examples of these 3-D sensors can be viewed in Figure 1.4.

Putting these versatile, cheap sensors in the hands of consumers has led to petabytes of color and depth (RGB-D) data for tons of applications, including creating virtual representations of physical spaces for augmented reality applications [32], image segmentation using 3-D point cloud data [33], simultaneous localization and mapping (SLAM) for autonomous vehicles [34], facial and skeletal tracking to help patients manage disease or recover from injuries [35], and 6-D object pose estimation for robotic arm grasping and object manipulation [36], just to name a few. The low cost, small form factor and data quality of these sensors enable incredible versatility, and the use cases continue to grow.

A decade later, the concepts that made AlexNet revolutionary, like CNNs, ReLU, max pooling and GPU training are taught in undergraduate machine learning courses, and a PyTorch implementation running on Google Colab can achieve the original paper's accuracy of 78.1% in 6 or 7 epochs, which took around 10 minutes to run [37]. What took Krizhevsky at least 7000 minutes and 2 GPUs to achieve, I could do now with a web browser in 30, and I could even surpass his original highest accuracy[8]. The GTX 580s that Krizhevsky used are 1000x less powerful than the cutting-edge RTX 3090 today [38], and the benchmark datasets themselves

[5]https://www.stereolabs.com/zed-2/

[6]https://dev.intelrealsense.com/docs/stereo-depth-camera-d400

[7]https://dev.intelrealsense.com/docs/lidar-l500

[8]https://colab.research.google.com/drive/1AYlgPOQhyNINdXnDWGewmNwnZzeVSXpK

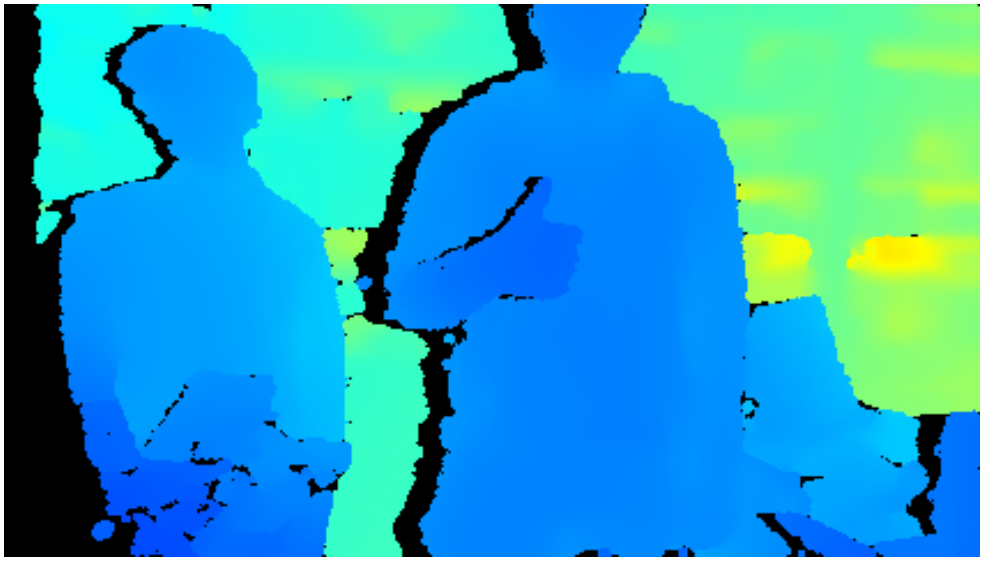


Figure 1.5: Colorization of a depth image of two people. The blue colors are closer to the camera, while the yellows and greens are shelves further away

have only grown larger and more complex as the hardware and software involved with computer vision and machine learning have improved.

1.2 Research Goal

Clearly, there is a major discrepancy between the new wave of advancements in machine learning and computer vision (ML/CV), and the techniques used to count and measure fish in industry. FishSense seeks to close this gap by bringing 3-D sensors underwater, enabling a literal new dimension of image segmentation and classification for accurate automated population tracking and length measurement. In this paper, I will discuss how I approached the systems design of self-contained FishSense hand-held modules, which utilize the incredible developments in camera, computing and machine learning technology in a compact form factor.

Creating these low-cost, modular systems opens the door to a world of possibilities. Modules can be placed in the hands of citizen divers for large-scale data collection in hundreds of unique ecosystems, to improve conservation and improve ocean health. Fixed-position or diver-held systems can be deployed in aquaculture centers to monitor population levels, estimate

biomass and ensure that fish are growing healthily. Camera trap systems can be placed in mangrove ecosystems to protect their populations and make sure the thousands of fish that call them home are safe. They can even be deployed on AUVs or boats to monitor deep-sea populations or the most recent catch. The viability of 3-D cameras on powerful underwater embedded computing platforms work for a whole slew of oceanic applications, both conservation-wise and commercially.

In order to explore the viability of FishSense modules for conservation, we partnered with the Reef Environmental Education Foundation (REEF)[9] and The Nature Conservancy (TNC)[10], which secures us enough funding to develop 5 modules aimed at a June 2022 deployment. These 5 modules will be placed in the hands of citizen divers, who will capture raw data and pass it off to REEF volunteers for species classification and other annotations for machine learning. In addition, Brice Semmens's lab will be utilizing several modules to survey the Pacific coast, directly comparing the cameras against existing solutions. With enough data, model accuracy and physical systems, FishSense can revolutionize the fisheries monitoring industry, and bring about meaningful change in combating overfishing and modernizing blue tech.

[9] https://www.reef.org
[10] https://www.nature.org/

Chapter 2

The FishSense System

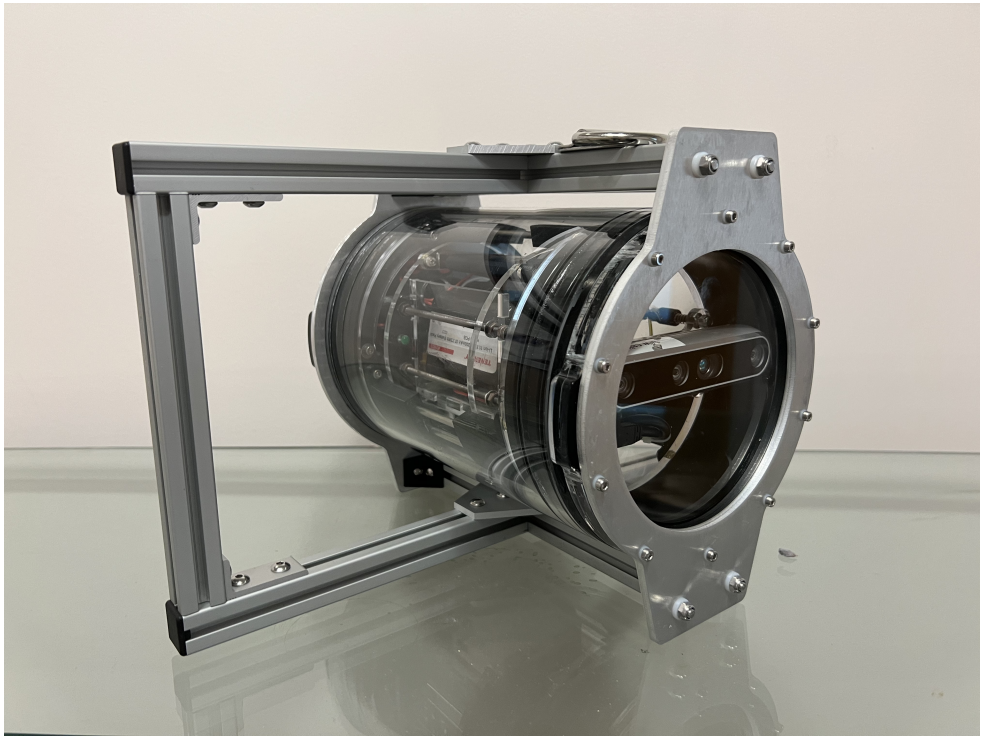


Figure 2.1: FishSense Version 2 prototype, with handles.

2.1 Overview

The FishSense Version 2 prototype (depicted in Figure 2.1) utilizes an Intel RealSense D455 depth camera, feeding into an Nvidia Jetson TX2 that is managing saving the RGB-D data onto a Solid-State Disk (SSD), a 1-terabyte volume that can be filled in the matter of a couple hours. The system is powered by two 3S2P batteries, and a diver can turn it on and off with the turn of a penetrator. Once the system is on, a diver can activate or deactivate recording using a magnetic wand held against the back plate, and is projected to last for approximately 2 hours while continuously recording. The block diagram for how power and data flow through the system can be viewed in Figure 2.2, while the firmware state diagram can be viewed in 5.1.

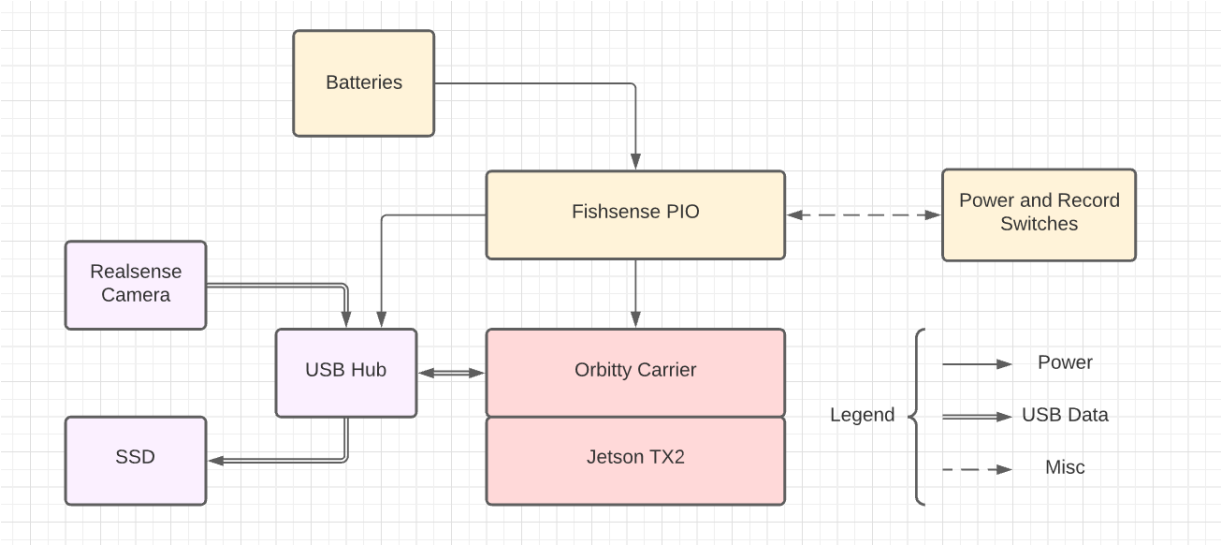
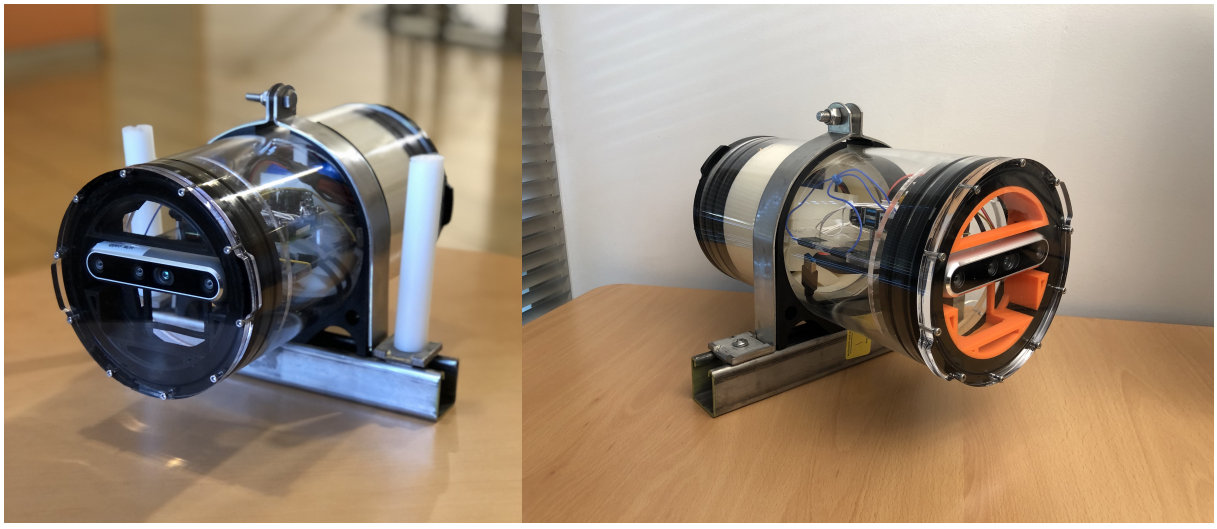


Figure 2.2: Systems diagram for FishSense Version 2.

The electronics are mounted onto an easily-removable electronics tray, which holds the depth camera against the front plate, holds the batteries and SSD in place, and presses the TX2 against the aluminum backplate for cooling. The electronics are enclosed in an acrylic tube, which has a 6 inch internal diameter, 6.5 inch external diameter, and 9.5 inch length. Finally, T-series aluminum bars compose the handles, which are fastened to the electronics enclosure via 10 bolts on the front plate.



(a) (b)

Figure 2.3: FishSense Version 1 prototype, with and without handles. (a) With handles (b) Without handles

This version addressed the issues we faced in developing and testing version 1 (Figure 2.3), and managed to achieve a balance of low cost, modularity and power, as nearly all the parts are off-the-shelf. These issues will be discussed in greater detail in the following few chapters. Should future opportunities allow exploration of some of the form-factor extensions discussed in Section 6.2, this existing design can be easily modified to have a longer battery life, a different processor or different deployment parameters. Extra blanks, as well as the inclusion of a FathomX tether board means that the system can be hooked up to a computer on the surface for real-time viewing, and it can be mounted to river beds or aquaculture cages to serve as static monitoring systems.

2.2 Sensor Choice

The initial goal of FishSense was to utilize the recent developments in affordable depth sensing technology in an underwater setting. My first experiment was a feasibility test to determine whether the Intel RealSense L515 was a viable sensor underwater. In order to test this,

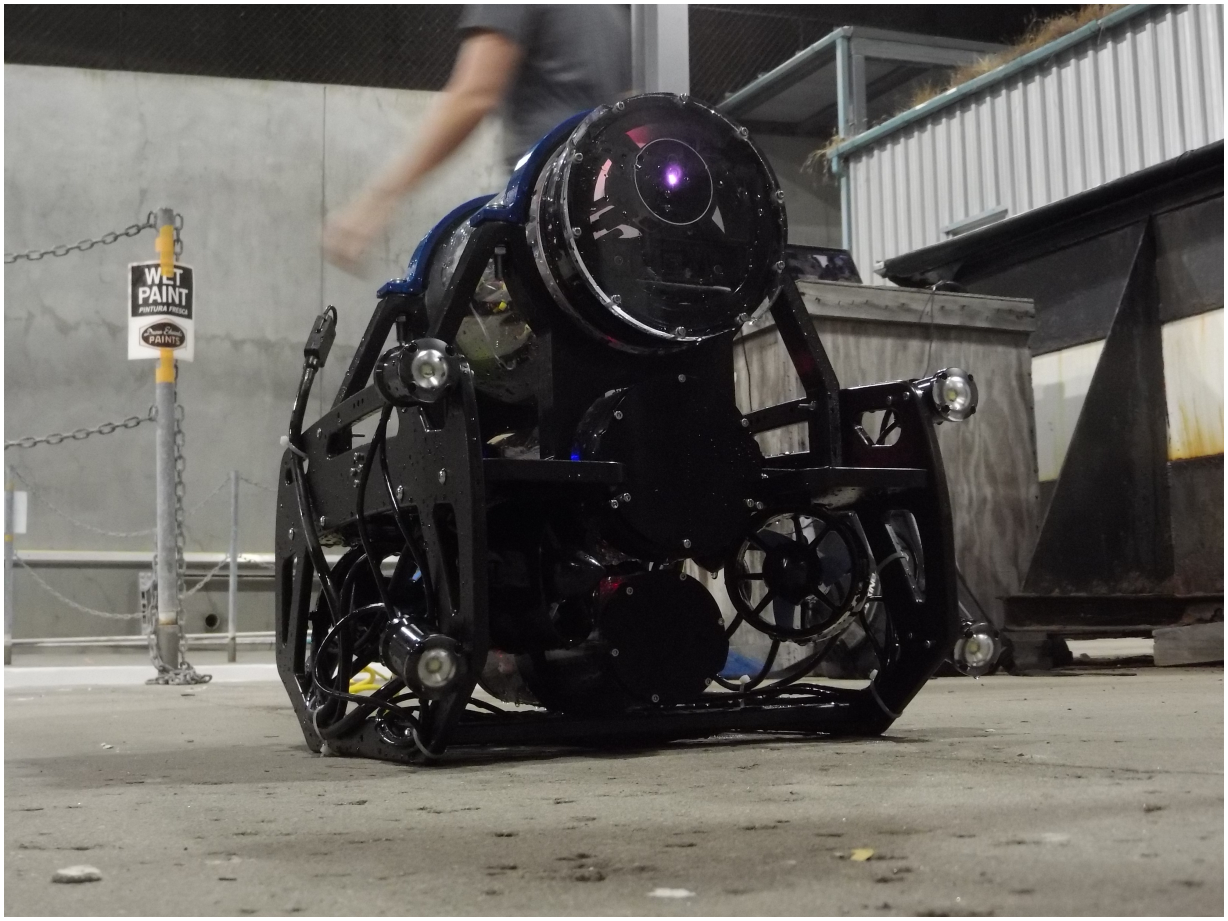


Figure 2.4: Triton Robosub[1] AUV equipped with the Intel RealSense L515. The purple light emitting from the sensor are the LiDAR beams.

we designed a 3-D printed frame to hold it in place against the acrylic endcap of an Autonomous Underwater Vehicle (AUV). This testing setup, which consisted of a Blue Robotics [2] BlueROV with an extra 6-inch tube to house an Nvidia Jetson TX2 and other sensors, can be viewed in Figure 2.4. While the testing rig proved effective, the L515 did not; as later lab tests demonstrated, the LiDAR beams emitted by the sensor were not able to penetrate the acrylic endcap, reflecting too much between the camera face and the endcap and diffracting too much through the endcap itself to provide accurate data back. Because LiDAR sensors expect the return signals reflected back into the sensor to have minimal environmental interference, the very fact that the sensor had

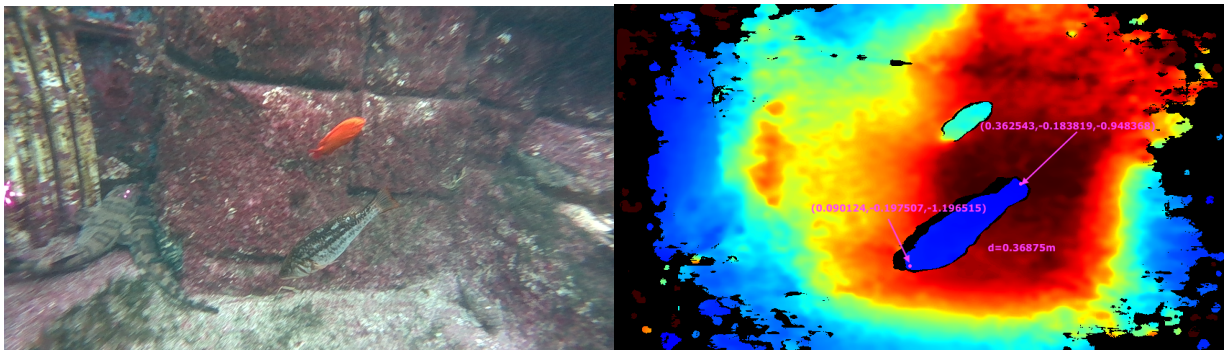
[2]https://bluerobotics.com

Table 2.1: Materials with properties advantageous for underwater LiDAR and RGB imaging. We chose a refractive index that best bridged the difference between air and water, minimizing the interference of diffraction.

Material Name	Material Class	Refractive Index	Durability in Fresh Water	Durability in Salt Water	Transparency	Price (USD/kg)
Fluorinated ethylene propylene (Unfilled)	Polymer	1.34 - 1.35	Excellent	Excellent	Transparent	10 - 15
PLA fiber (Ingeo)	Fiber and particulate	1.35 - 1.45	Acceptable	Acceptable	Transparent	4.22 - 5.67
polychlorotrifluoroethylene PCTFE (unfilled)	polymer	1.38 - 1.4	Excellent	Acceptable	Transparent	100 - 195
Soda Borosilicate	glass	1.43 - 1.5	Excellent	Excellent	Transparent	4.07 - 6.81
Cyclic Olefin Copolymer High Flow	polymer	1.44 - 1.46	Excellent	Excellent	Optical Quality	15 - 16
Polyactide / Polyactid Acid (General Purpose)	polymer	1.44 - 1.46	Acceptable	Acceptable	Transparent	2.82 - 3.71
Silica 7913 (96%)	glass	1.45 - 1.47	Excellent	Excellent	Transparent	6.01 - 10
Silica (Quartz fused)	glass	1.45 - 1.47	Excellent	Excellent	Transparent	6.21 - 10.4
Silica 7940 (Fused)	glass	1.45 - 1.47	Excellent	Excellent	Optical Quality	6.21 - 10.4
Borosilicate glass, corning 7070	glass	1.46 - 1.48	Acceptable	Acceptable	Transparent	4.49 - 7.48
Borosilicate - KG33	glass	1.46 - 1.48	Excellent	Excellent	Transparent	4.15 - 6.22

to shine through something posed an immediate challenge. This doesn't even consider the fact that the beams need to shine through water without getting absorbed after getting through the endcap. While we did briefly investigate alternate materials to build the endcap out of, summarized in Table 2.1, this ultimately proved unnecessary as our tests with the L515's cousin proved much more effective.

The Intel RealSense D435 and D455 were the next depth cameras tested. Both cameras utilize stereoscopic vision as the main method for determining depth, with an infrared projector to lend even greater accuracy to the depth as well as a color camera to match RGB with depth. While we found that both cameras overcame the endcap limitations of the L515 (as light diffraction was much less of a concern with stereoscopic images, as it only affects the edges of the frame and not the center), the D455's increased range due to its larger size made it a much more ideal candidate for underwater use. Through a similar experimental setup to the L515 tests, we found that the D455 could consistently identify the depth of objects that were up to 5-6 meters away, which was more than enough to move forward with developing a system to support the camera. While



(a) (b)

Figure 2.5: Underwater Color and Depth images collected by a FishSense system tested at the UCSD Birch Aquarium. (a) Color image of two fish (orange and black/gray, in center of frame). (b) Depth image of the same two fish, with length of larger fish computed. The light blue and dark blue outlines correspond with the fish.

the ZED camera was considered in lieu of RealSense sensors, the power of the RealSense SDK, availability of the cameras, and reports from another AUV team that attempted to use the ZED camera in their system [39] that it was very incredibly difficult to calibrate underwater, led us to stick with the D455. However, because the Intel RealSense team has been discontinued [40], other camera options may prove necessary, and this will be discussed in further detail in Section 6.1.1.

The D455's RGB and depth channels (Figure 2.5) provided ideal data sources for identifying fish length and biomass. Previous underwater image detection modules trained on 2-D color images, have been able to achieve respectable performances of over 98% accuracy [41, 42]. Introducing the depth channel provided us with clear segmentation between individual fish, which meant that RGB detection models could be easily augmented to be even more accurate. Section 5.2.3 discusses my approach for this segmentation improvement in further detail. In addition to the improvements in detection, the depth data enables us to determine fish length by measuring distances from the camera at the fish's snout and tail fork, and applying some basic trigonometry. This will be further discussed in section 5.2.4.

2.3 Processor Choice

With the right 3-D camera in mind, the next most important decision to make was in the computation department. For the FishSense version 1 prototype, the Raspberry Pi 4 [3] was selected for its low cost, low power requirements, and USB 3.1 ports. Initial lab tests confirmed that the camera required modern USB data transfer rates, meaning that we needed to match the RealSense's USB 3.1 spec in our compute board.

However, lab tests revealed that the USB port was not the limiting factor in how fast data could be saved; the Pi itself was. The limited processing and USB peripheral power of the computer rendered saving full point clouds impossible; instead the closest we got was a depth gradient image for each RGB frame, which could run at 5-10 frames per second. This did not give us enough concrete depth data to train specialized models, but did allow us to evaluate the usability of the camera underwater, as well as a nice volume of color images.

However, the Jetson TX2, which we had been using in the AUV testing platform, was capable of saving full point clouds, and got up to 30 frames per second in the lab! In addition, the TX2 offered CUDA support, meaning that detection could be optimized for real-time use on the device for any future applications that need it. While other Jetson products like the Nano, AGX Orin, or Xavier NX are more modern systems, the cost-to-performance ratio of the TX2 (viewable in Table 2.2), the availability of a compact carrier board, the lab-tested compatibility with the D455, and the team's greater comfort with a more familiar board led us to choose the TX2 over newer Jetson models. However, the Nano offers enough performance to generate high quality data at a large cost saving, and the AGX Orin and Xavier offer much more processing power; the feasibility of these boards will be explored in Section 6.1.2.

The ability to save depth images at high frame rates was crucial for our ability to conduct meaningful software postprocessing, so switching to the TX2 was absolutely necessary for the

[3] https://www.raspberrypi.com/products/raspberry-pi-4-model-b/

Table 2.2: A comparison of price, performance, and framerate benchmarks for the Jetson boards and Raspberry Pi we considered as the system's main compute unit [2] [3].

	TX2	**Pi 4**	**Nano**	**AGX Xavier**	**AGX Orin**
Cost	$400	$55	$100	$1100	$2000
OS	Ubuntu	Raspbian	Ubuntu	Ubuntu	Ubuntu
CPU Freq	2GHz	1.5 GHz	1.43 GHz	2.03 GHz	2 GHz
CPU Cores	4	4	4	8	12
GPU Freq	1.3 GHz	N/A	921 MHz	1.21 GHz	1 GHz
GPU Cores	256	N/A	128	512 GPU 64 TPU	2048 GPU 64 TPU
RAM	8 GB	4 GB	4 GB	32 GB	32 GB
Power	7.5-15 W	15 W	5-10 W	20-40 W	15-40 W
Size	87 mm × 50 mm	85 mm × 49 mm	69.6 mm × 45 mm	100 mm × 87 mm	100 mm × 87 mm
AI Performance	1.33 TFLOPs	N/A	472 GFLOPs	32 TOPs	200 TOPs

FishSense systems to deliver their intended value. With both the camera and compute unit in place, requirements could be defined for the electrical and mechanical subsystems, which I will discuss in further detail in Chapters 3 and 4.

2.4 Acknowledgements

Chapter 2 contains unpublished material coauthored with Avalos, Alberto. The thesis author was the primary investigator and author of this material.

Chapter 3

Electrical System

3.1 Overview

The electrical design of the FishSense modules were heavily dictated by the system-level decision to use an Intel RealSense D455 as the primary camera for this module, which influenced the decision to use the Nvidia Jetson TX2. These two decisions impose hard power and peripherals requirements on the electrical system: normally, the TX2 is attached to a development carrier board that gives the computer Wi-Fi capability, multiple USB ports, HDMI, Ethernet, and more. However, this carrier board is approximately 150mm by 150mm, while the Jetson module itself is only 70mm by 45mm, less than a sixth of the size. The solution to this issue is to use a more suitable daughter board, the ConnectTech[1] Orbitty carrier board. This carrier is the same size as the TX2, but includes the essential ports: GPIO, Ethernet, HDMI, one USB 3.0 port, and power, and even an SD card slot, all of which can be seen in Figure 3.1. The TX2 with this Orbitty carrier board required an input voltage of at least 9 volts, and necessitated the use of a USB hub to plug both the RealSense camera and the SSD into the one USB port.

[1] https://connecttech.com

(a) (b)

Figure 3.1: The Nvidia Jetson TX2 on its stock developer board and Orbitty carrier board. In each, the actual TX2 module is the grey metal cuboid. (a) TX2 on the Nvidia stock developer board. The TX2 module is covered by a heat sink and fan. (b) TX2 with ConnectTech Orbitty carrier board. There is no heat sink on this module.

Based on the power requirements of the electrical components selected, we chose to use two 3s2p Lithium-Ion (Li-Ion) battery packs to power the Version 2 system, rather than the 4s4p batteries used to power the Version 1 (where "s" represents the number of cells wired in series, and "p" represents the number wired in parallel). We chose 4s4p batteries for the first prototype because the Raspberry Pi only needed 5 volts (V) to perform [43], and a step-down converter for this power differential was relatively easy to source on an online marketplace like DigiKey. The small form factor and long dive times of the 4s4p configuration were favorable for given the lower power consumption of the overall system, and the voltage conversion chip was quite efficient.

However, in the version 2 prototype, the Orbitty board's power requirements of 9-14V maximum [44] presented an issue. One cell of a Li-Ion battery ranges between 3.2-4.2V, with a nominal voltage of 3.7V, meaning that 4s configurations would have an average voltage of 14.8V. This was already more than the supported voltages of the Orbitty, and the batteries could be

charged all the way to 16.8V. In addition, a step-down voltage converter was much more difficult to source, and would prove much less efficient, for this variable range, so we decided to lower the number of cells in our battery packs from 4 to 3. 3s batteries fit nicely within the range of the Orbitty board's power requirements, and 2 of them in parallel (the "p" in the naming scheme) meant that we could maximize our battery life because each individual cell would discharge at 50% of the speed. Switching to the lower cell count had the added bonus of lowering the cost of the batteries, too.

With the components, batteries and constraints laid out, the electrical design was just missing one thing: a way to tie all the pieces together.

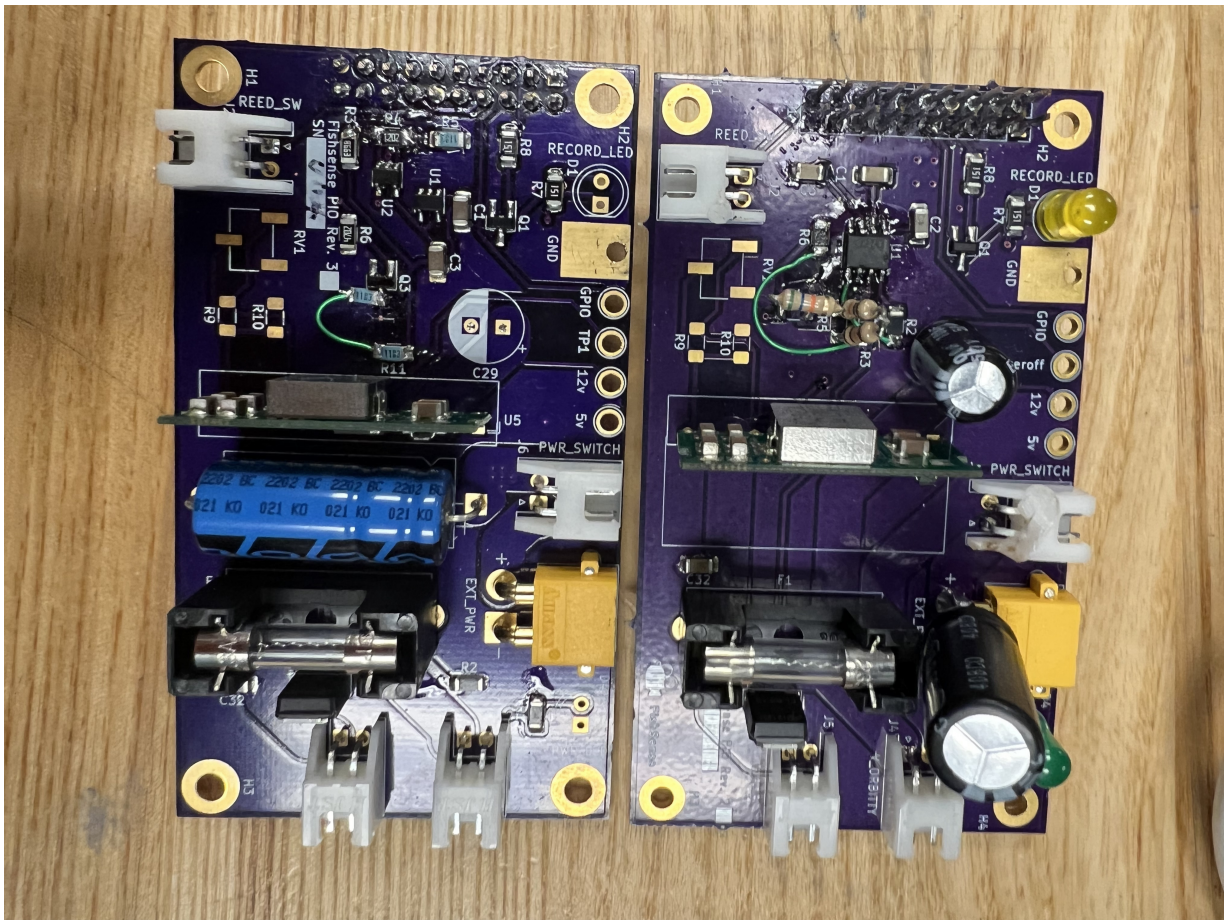


Figure 3.2: Reworked comparisons of Revisions 2 and 3 of the PIO board. Revision 3 is on the left, while revision 2 is on the right.

3.2 Power/Input/Output Board Design

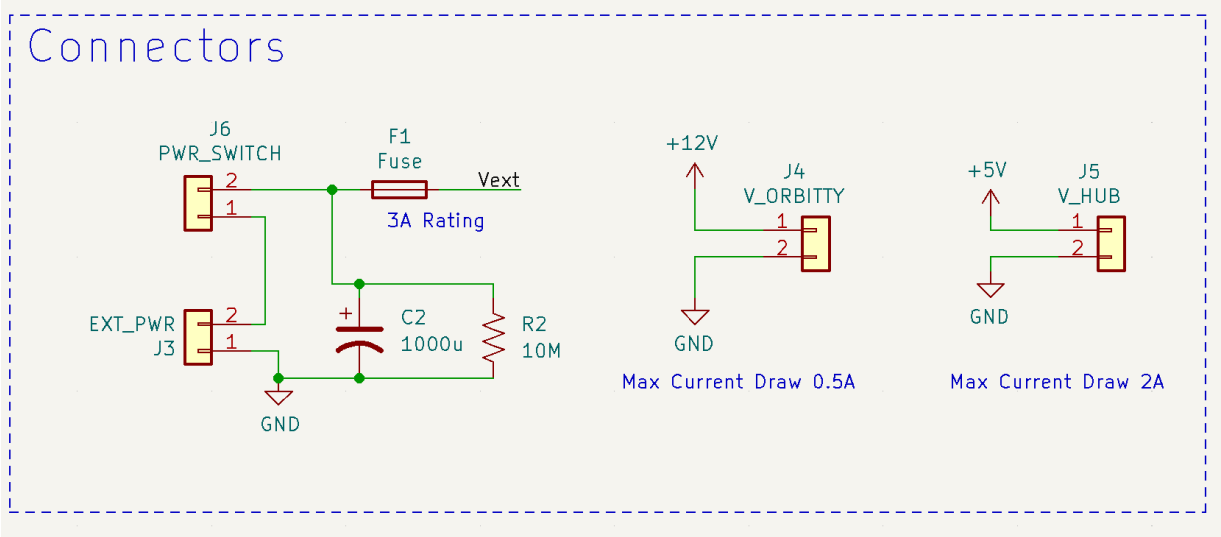


Figure 3.3: Schematic of the power connectors.

The Power/Input/Output (PIO) board is the custom Printed Circuit Board (PCB) that connects the FishSense modules' commercial off-the-shelf parts. It serves several key purposes: powering the TX2 and the USB hub, turning off the system when the power penetrator is unscrewed, ensuring the system shuts down if battery power drops too low, passing signals from a reed switch to the TX2 via GPIO, and driving indicator LEDs. The first iteration of the PIO board, Revision 1, was used in the Version 1 prototype, and included a similar but more limited set of features, tuned to the Raspberry Pi over the TX2.

The most crucial part of the PIO board is the external power circuit, depicted in the left half of Figure 3.3. An in-rush capacitor (C2) and a bleeder resistor (R2) lie directly connected to the batteries (I'll discuss the purpose of these components in Section 3.2.2), but all other parts are protected by a fuse. The filament in this fuse (F1 in the figure) will burn if the amount of current drawn by the system is greater than 3 amps (A), which provides a physical failsafe should any unexpected current spikes occur. The other two connectors in the figure connect the PIO board to the Orbitty power input and the USB hub input, respectively. As noted on the figure, the Orbitty

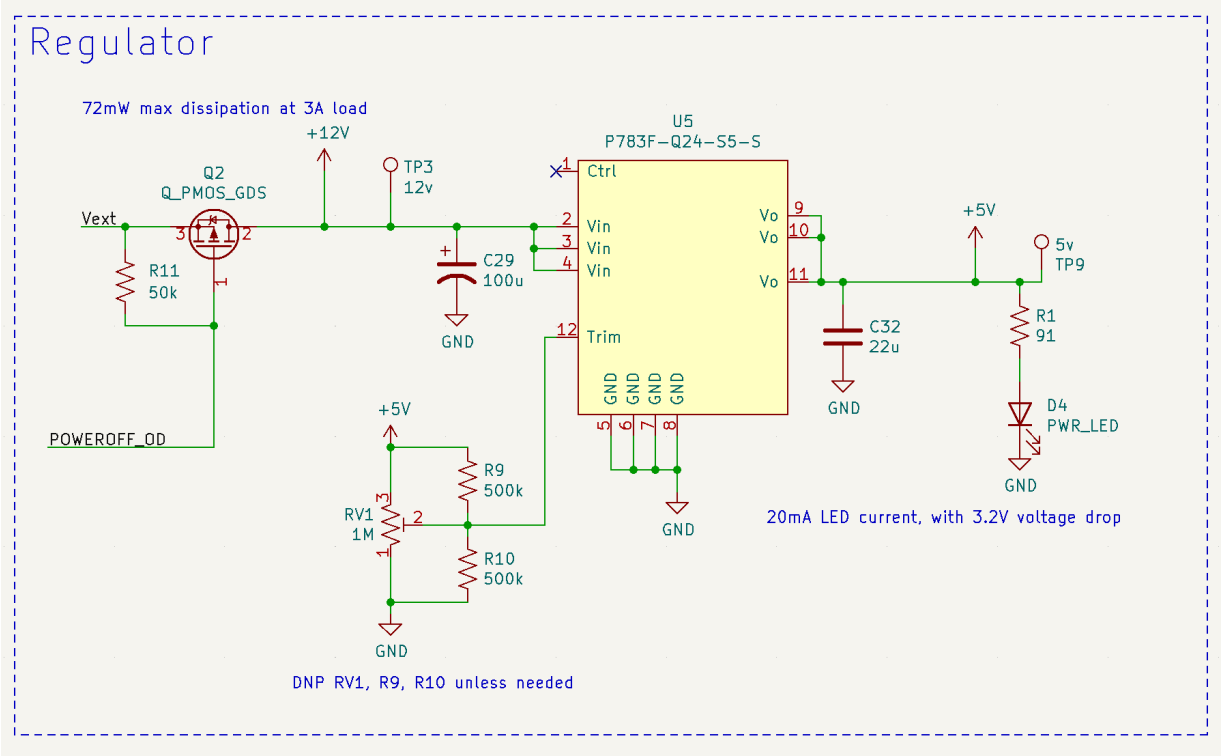


Figure 3.4: Schematic of the power regulator.

board will draw a maximum current of 0.5A and the USB hub (and the D455 camera it powers) will draw a maximum current of 2A, placing the total amperage of the system at 2.5A, safely under the 3A limit imposed by the fuse.

After the fuse, the power signal passes through a PMOS (Q2 in Figure 3.4), which essentially operates as a kill-switch. As long as the power-off signal remains low, current flows from the battery power to the Orbitty directly (indicated by the +12V signal in the figure), as well as the step-down voltage converter U5. However, if the voltage supervisor detected that the battery voltage has dropped below the safe level of ~9.5V, the `POWEROFF_OD` signal will go high, cutting power to the boards. Sections 3.2.1 and 3.2.2 will delve into this voltage supervision in further detail, as these chips gave us the most trouble.

The last component that was needed for this Version 2 prototype PIO board was a GPIO header for the Orbitty board. This header, depicted in Figure 3.5, illustrates how the reed switch

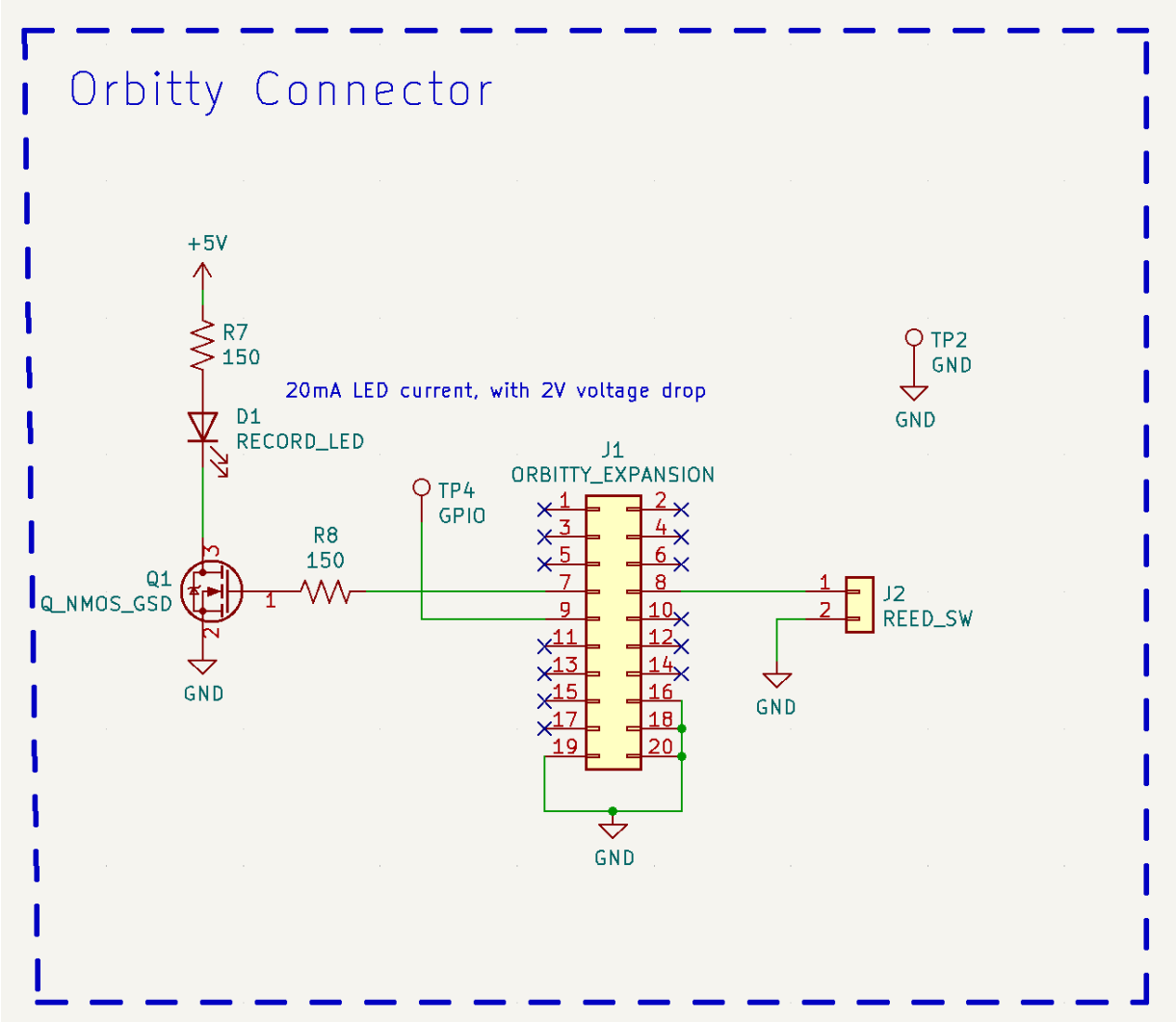


Figure 3.5: Schematic of the Orbitty GPIO pins.

connects to one of the GPIO pins to control the start and stop of recording, while another pin drives the recording indicator LED via another MOSFET (this time, an NMOS).

This PIO board, while relatively straightforward in design, took many hours of design reviews, assembly, and testing to achieve a complete working version, and still requires a further revision to be perfect. However, the test points we added to be able to measure the 5V line, 12V (Orbitty power) line, and `POWEROFF_OD` signals proved invaluable in evaluating the correctness of the board.

3.2.1 Revision 2

The main three schematics depicted in the above figures remained mostly the same between Revision 2 and Revision 3; however, you may have noticed that I did not include the voltage supervisor. This is because we entirely switched supervisors between the two revisions, and much of the debugging process was centered around achieving the right hysteresis thresholds.

LiPo batteries, like most batteries, lose voltage as they discharge. This is why I presented a range of voltages for each cell, and this also means that power electronics need a way to handle when the lowest safe threshold for battery power has been reached. If the power falls below this threshold, it becomes dangerous to recharge the battery because of the potential of explosion.

This is where the voltage supervisor comes in: it monitors the external battery voltage, turns off the system when the power falls below 9.5V, and doesn't allow it to power back on until the battery voltage reads at a sufficiently high threshold to avoid power-cycling the rest of the system. The difference between the power-off and power-on thresholds is called hysteresis, and it is crucial for ensuring that the system is not attempting to power off and on continuously as the batteries hover around the shut-off voltage. The hysteresis also serves to protect both the battery and the rest of the system from the increasing internal resistance of the battery as it discharges, because this increase causes a decrease in the total current the battery can supply. Even if the battery has enough voltage, the decreased current means that the high-current D455 can suffer performance issues, and the TX2 could potentially run out of power to save the data.

The first chip that we selected to control this hysteresis was the Texas Instruments TL7702A[2], which is depicted as U1 in Figure 3.6. We selected values for R4 and R5 based on the datasheet to achieve a shut-off threshold of 9.5V, with 0.3V of hysteresis [45], and used the $\overline{\texttt{RESET}}$ pin to determine the poweroff signal. In addition, a feedback loop was added to increase the hysteresis threshold, as recommended by a note in the datasheet.

[2]https://www.ti.com/product/TL7702A

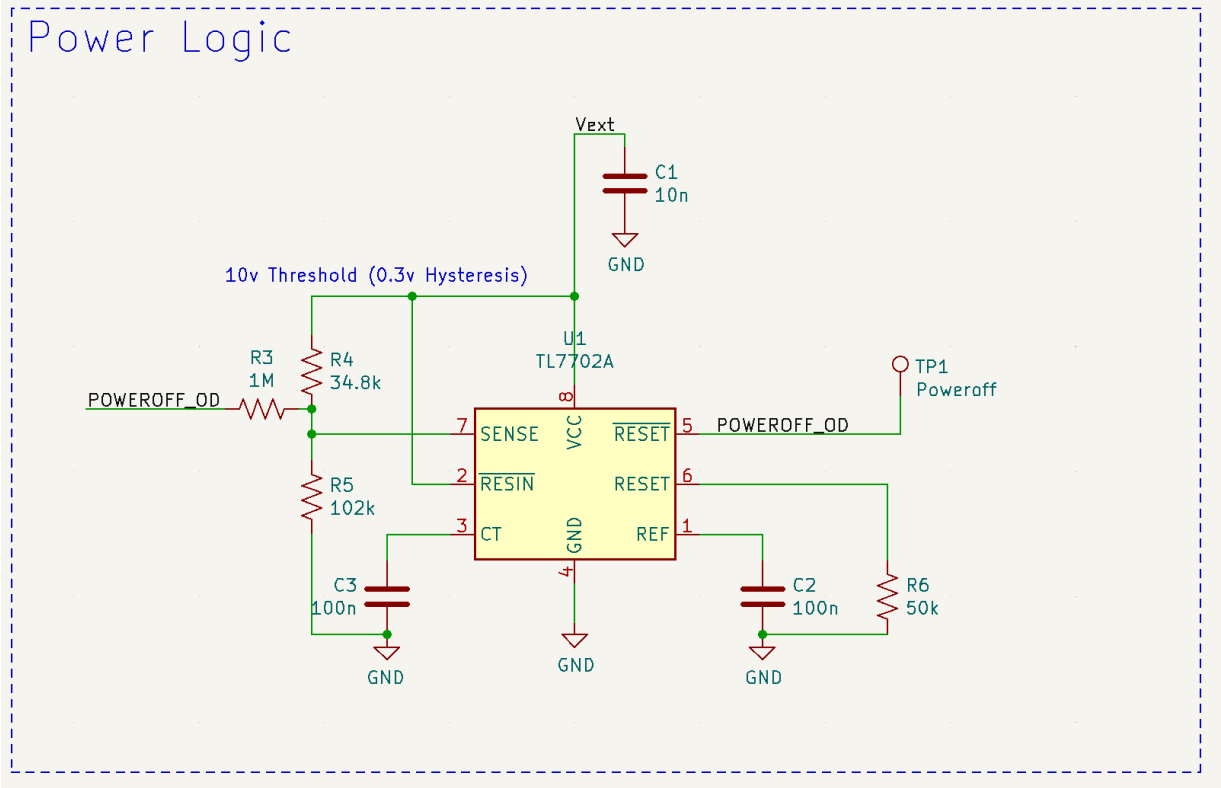


Figure 3.6: Schematic of the Revision 2 voltage supervisor.

Once we completed a design review, we ordered the parts and assembled a test board. After powering the board, we found the first issue with the design: we had hooked up the POWEROFF_OD signal to the wrong pin; a misreading of the datasheet led us to conclude that $\overline{\text{RESET}}$ was the pin we wanted, when in fact it was RESET that we needed to utilize. In order to fix this, we depopulated R6 to allow the reset signal to be utilized, then added a jumper wire from R3 to one of the R6 pads to connect RESET to the POWEROFF_OD line. However, correcting this issue still did not power on the board, meaning that there was something else at fault.

The next thing we tried to do was remove the feedback loop, which appeared to be interfering with the hysteresis thresholds; once this was gone, we were able to power the PIO board on, but we were not able to achieve effective hysteresis. Instead of turning on at the high threshold and staying on until the low threshold is reached again, the PIO board would turn off above the low threshold and power right back on when the power was turned up. We were able

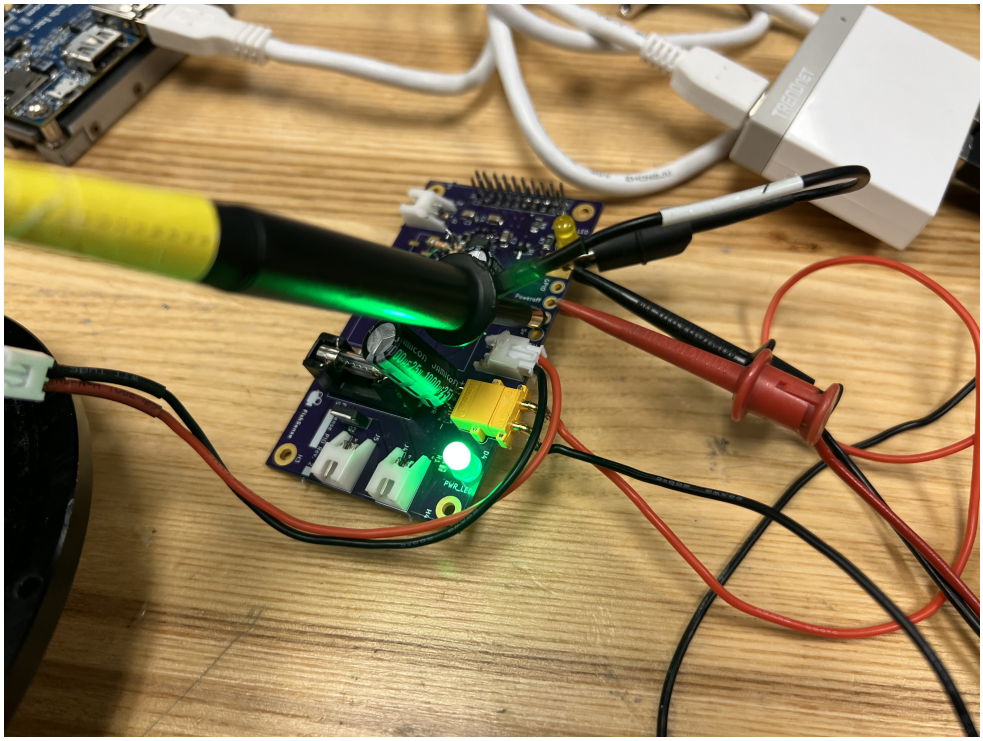


Figure 3.7: Experimental setup for testing Revision 2 hysteresis.

to test this by attaching leads from the Poweroff and 12V lines to an oscilloscope, as shown in Figure 3.7, and observed oscillation consistent with the LED behavior we saw in the Poweroff signal, as shown in Figure 3.8. These periodic spikes indicated that the voltage supervisor was continuously recognizing the system should be powered off, powering it off, then recognizing that it needs to be powered on again, restarting the cycle.

At this point, we decided to put the hysteresis issue to the side, because we wanted to ensure that the other parts of the PIO board were behaving properly. Now that the board could power on on its own we could clearly see that the LEDs were working, and that the 12V and 5V lines were reading properly. We began testing it with the TX2, and found that the extra current drawn by the TX2 upon startup caused a temporary drop in voltage to maintain consistent power draw from the batteries, which triggered the voltage supervisor to shut the system back down. We fixed this by experimenting with several large capacitors, settling on 1000 microfarads (μF) as the smallest capacitor that could power on the system. This capacitor solved the issue because

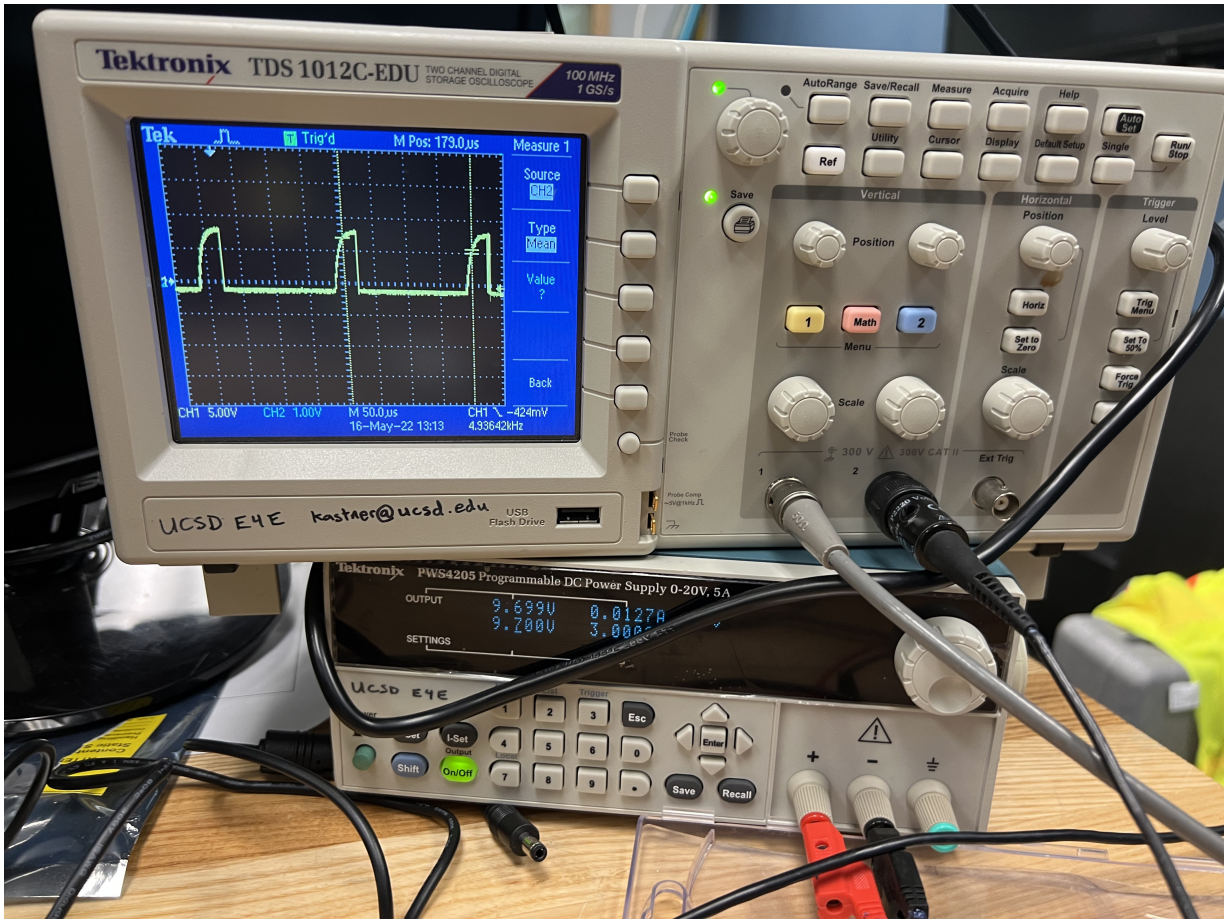


Figure 3.8: Oscillation of the poweroff line in reworked Revision 2.

it allowed the extra inrush current to be drawn from the capacitors instead of the battery itself, maintaining normal voltage thresholds during the microsecond or so it took to draw the initial power for startup.

Finally, with the TX2 powering on, we began to test the firmware, and found a couple other issues with the PIO board that had been overlooked in the design review. The GPIO header for the Orbitty carrier was upside-down because it needed to go on the bottom of the board rather than the top, and we also found that the mounting holes were too small. We fixed the former issue by cutting the unnecessary pins off of the Orbitty carrier, and the latter by drilling out larger holes. This allowed us to send a nearly fully-functioning FishSense Version 2 prototype to a deployment in the Caymen Islands, because the system could power on, start/stop recording, and save the

data properly. As long as the divers were careful about the dive lengths, they could avoid issues introduced by the faulty hysteresis.

We began to delve further into the issues with the hysteresis, constructing a testbench using the voltage supervisor attached to a breadboard, following the schematic depicted in Figure 3.9. We wanted to analyze the effect of the resistor choices in R1 and R2 (R3 and R4 in Figure 3.6), and by using some standard resistor values we were able to determine that increasing the resistance of R1 and R2 while holding the ratio between them constant yielded linear increases in the observed hysteresis threshold, as shown in Figure 3.10.

Using this testbench, we also determined several noncompliant behaviors exhibited by the TL7702A. For one, the feedback loop seemed to decrease the hysteresis threshold by an unpredictable amount, which went unmentioned in the datasheet. In addition, the hysteresis region displayed oscillation issues even with no load, indicating that there was something wrong with the chip or the way that the datasheet presents it. Given these issues, as well as the other major mistakes with Revision 2, we decided to begin development on Revision 3 to simply replace the supervisor and fix the remaining faults.

3.2.2 Revision 3

In our design and development process for Revision 3, we intended to fix all of the issues we ran into with Revision 2. This began with the design review process; we created a checklist of key changes and ensured that they were all present in the new version. In addition, we double-checked the datasheet for the new voltage supervisor, the Microchip MIC2279L[3] to ensure that the right resistors were chosen, to avoid the implementation issues we faced earlier.

The MIC2279L required an extra voltage regulator to lower the input voltage from the variable voltage from the battery to a consistent 3.3V, but was much simpler to set up than the TL7702A in terms of voltage monitoring. The equations provided in the datasheet were much

[3] https://www.microchip.com/en-us/product/MIC2779

clearer, and the testing in-lab demonstrated behavior consistent with what was listed [46].

After repeating the part-ordering and assembly process, we dove into testing. Our testing setup is depicted in Figure 3.12. This time, when we plugged the PIO board into power, it was able to turn on properly, so we jumped right to plugging in a TX2. This is where we ran into our first issue: our breadboard testing with the Revision 2 board failed to take into account the capacitance of the Orbitty carrier board, and our new filtering capacitor for the inrush was not delivering enough current with out the voltage dropping. Similar to the Revision 2 board, the voltage supervisor would detect a low voltage because of the current, and shut the system back down. By monitoring the poweroff and 12V test points, we could see how the system power and the voltage monitor were working together, allowing us to visualize the poweroff trigger in Figure 3.13.

To fix this issue, we wanted to add a soft start to the kill-switch PMOS, which was achieved by adding a gate drive resistor. In other words, we slowed the activation of the PMOS input signal for long enough that the system powers on, allowing normal boot-up by spreading out the Orbitty capacitance charging over a longer period of time. We initially chose a 50Kω resistor due to what we had on hand and some ballpark estimates, and tested its effectiveness. With just this change, we were able to achieve a slow start, but it wasn't enough to keep the system powered on, as seen in Figure 3.14. The solution we had was to simply increase the resistance, which yielded the results in Figure 3.15 and got the system to power on properly!

From here, we could verify that the new voltage supervisor had correctly-tuned hysteresis values; by powering on the system and decreasing the input voltage until it powered off, then turning it back up, we were able to determine experimentally when it powered back on. Figure 3.16 illustrates the oscilloscope outputs as the power reached the higher threshold of hysteresis, while Table 3.1 summarizes the experimental thresholds for both Revision 2 and Revision 3. This time, our experimental setup aligned with our theoretical hysteresis calculations, but we still ran into an issue with the signal oscillating around 10.3V (depicted in Figure 3.17).

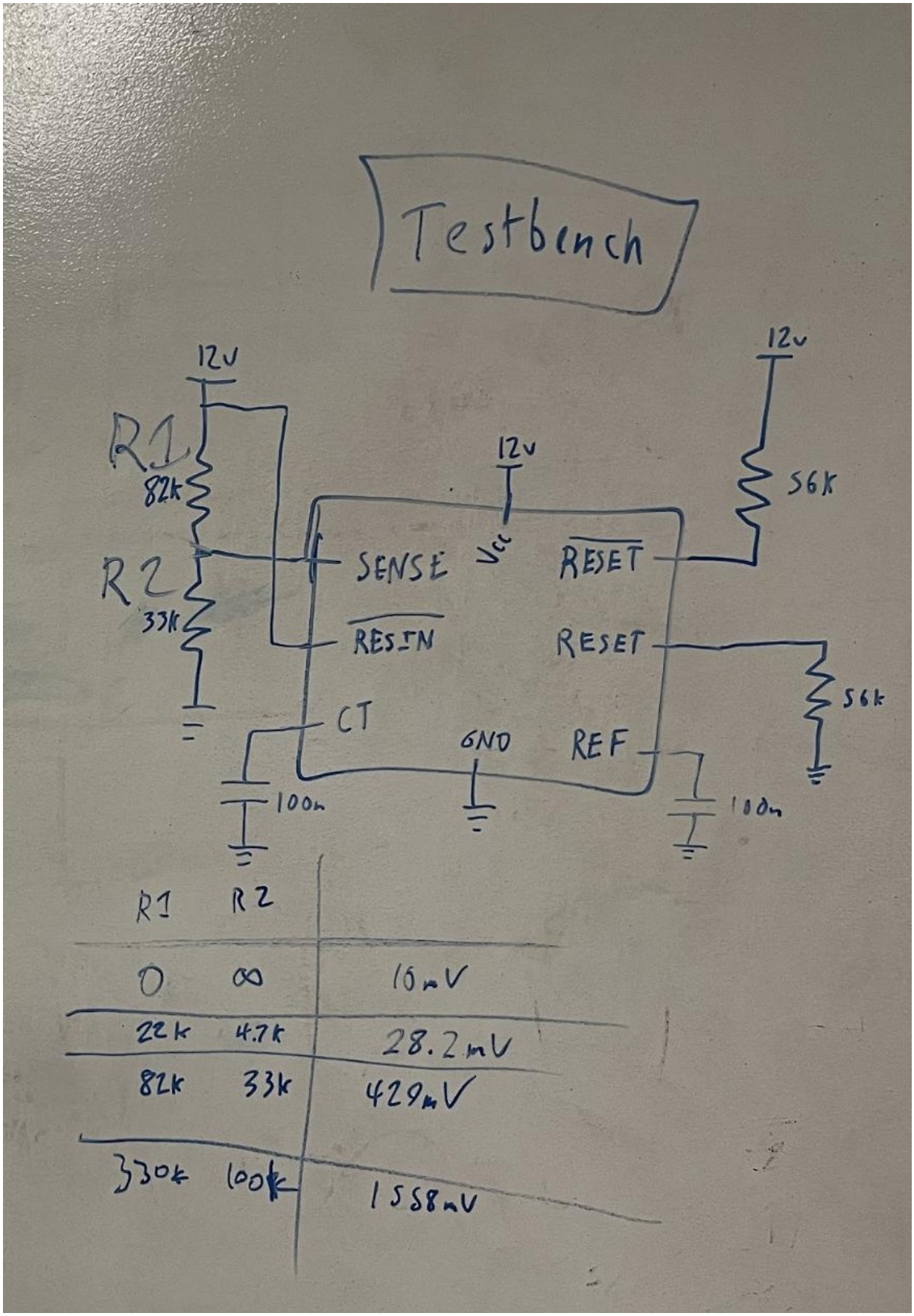
Table 3.1: Summary of hysteresis behavior between Revision 2 and Revision 3 (all values in volts).

	Rev 2	**Rev 3**
Shut-off Threshold	9.5	9.7
Power-on Threshold	11.1	10.7
Oscillation Issues	10.3	–

This oscillation was again due to capacitance issues, and artificially increasing the gate capacitance via an external capacitor solved the issue. Finally, with the reworked version of Revision 3, we were able to achieve a perfectly functioning PIO board. One unintended side effect of our soft start solution, however, is that adding the extra gate resistance/capacitance meant that a slow stop was achieved as well. When the system is first powered on, as the filtering capacitors for the inrush current are charging, the system will go through a temporary start/stop sequence that lasts approximately a millisecond. Once these capacitors are charged, however, the PIO board performs exactly as intended, and is ready for a final revision and larger-scale production.

3.3 Acknowledgements

Chapter 3, in part, is currently being prepared for submission for publication of the material. Tueller, Peter; Maddukuri, Raghav; Paxson, Patrick; Suresh, Vivaswat; Miao, Albert; Drews, Donovan; Paxson, Charles; Semmens, Brice; Kastner, Ryan. The thesis author was the primary investigator and author of this material.



33

Figure 3.9: Testbench schematic for testing the resistor choice effect on hysteresis.

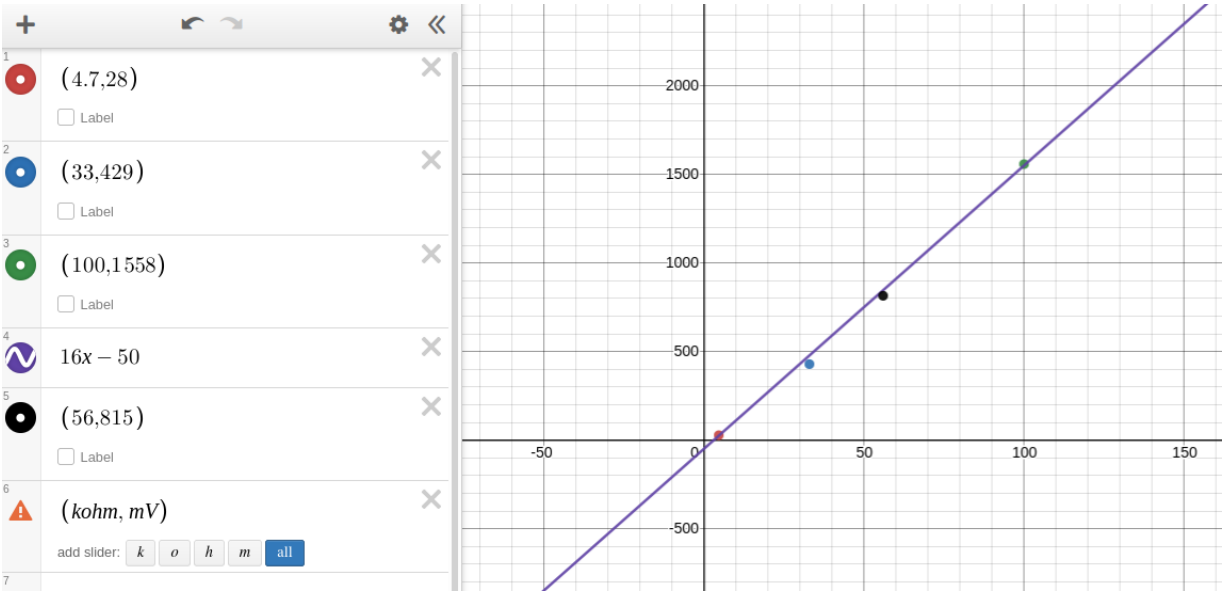


Figure 3.10: Observed hysteresis threshold vs. R2 resistance (in kΩ).

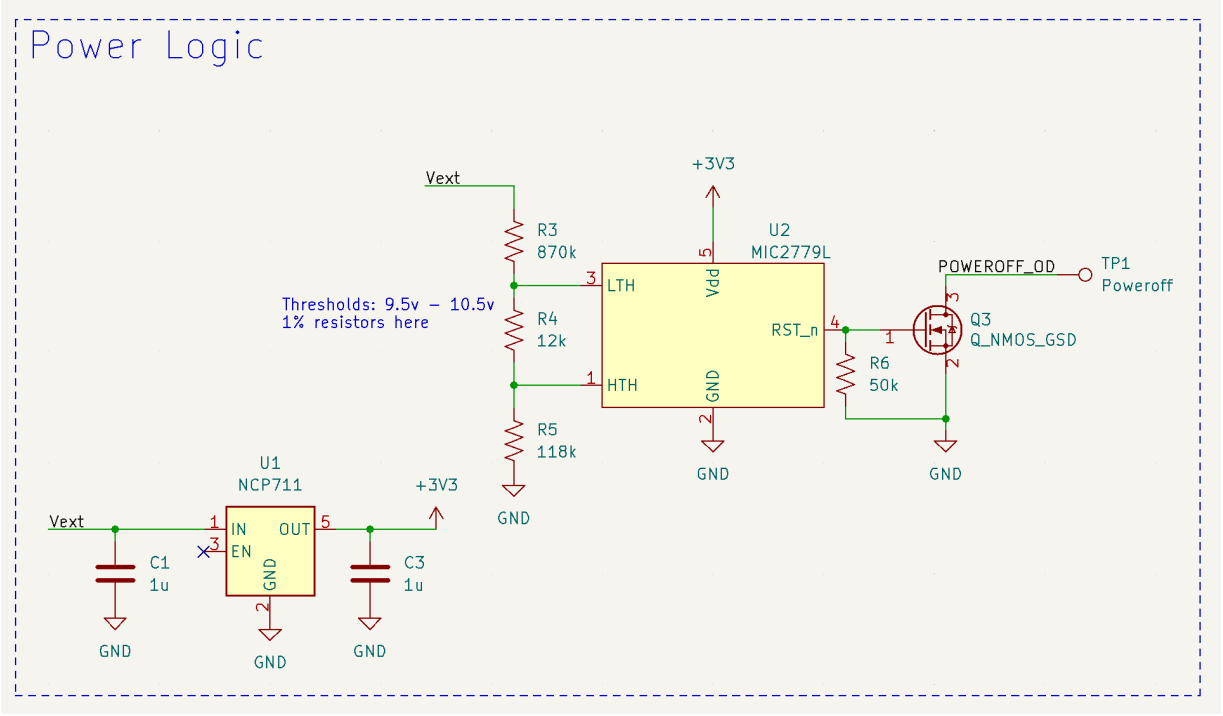
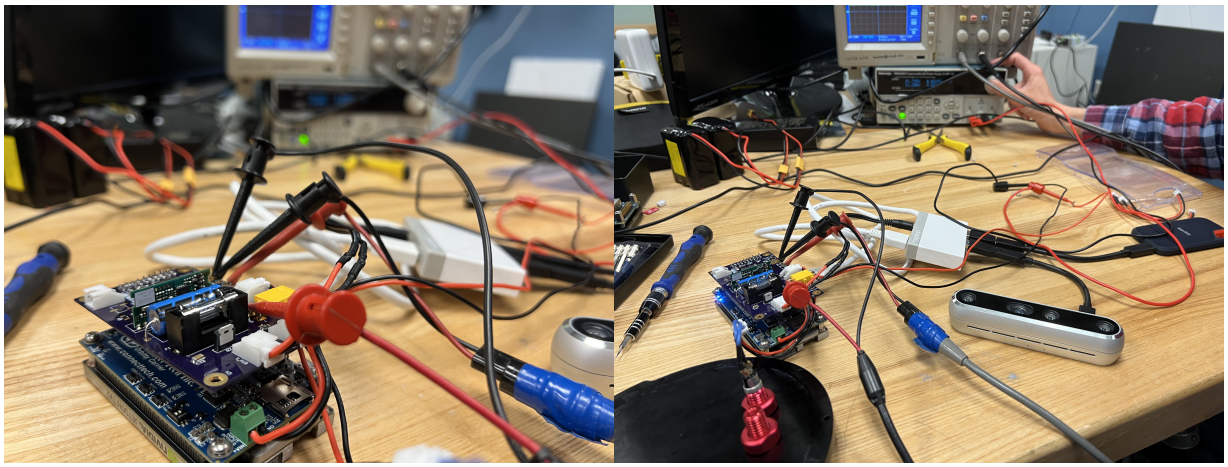


Figure 3.11: Schematic of the Revision 3 voltage supervisor.



(a) (b)

Figure 3.12: Experimental testing setup for Revision 3 PIO boards.

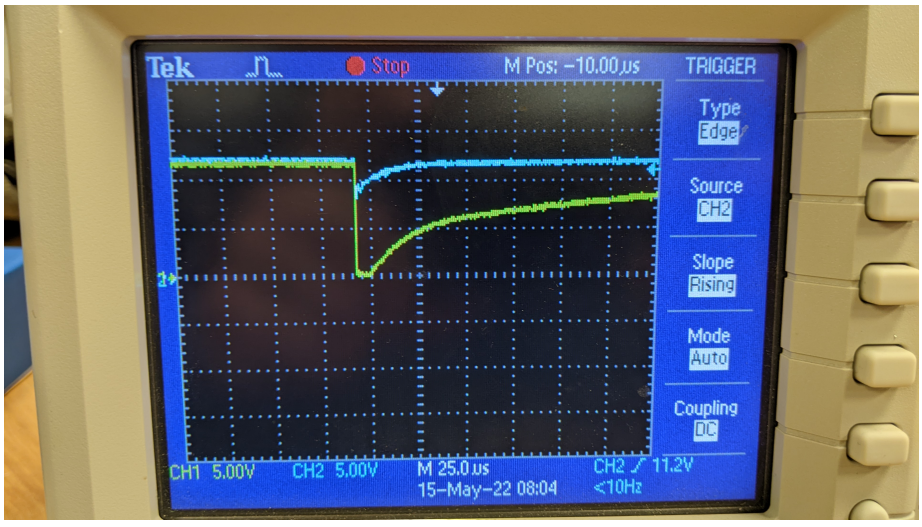


Figure 3.13: Oscilloscope output of Revision 3 without starting. Yellow represents the poweroff signal, while Blue represents the voltage.

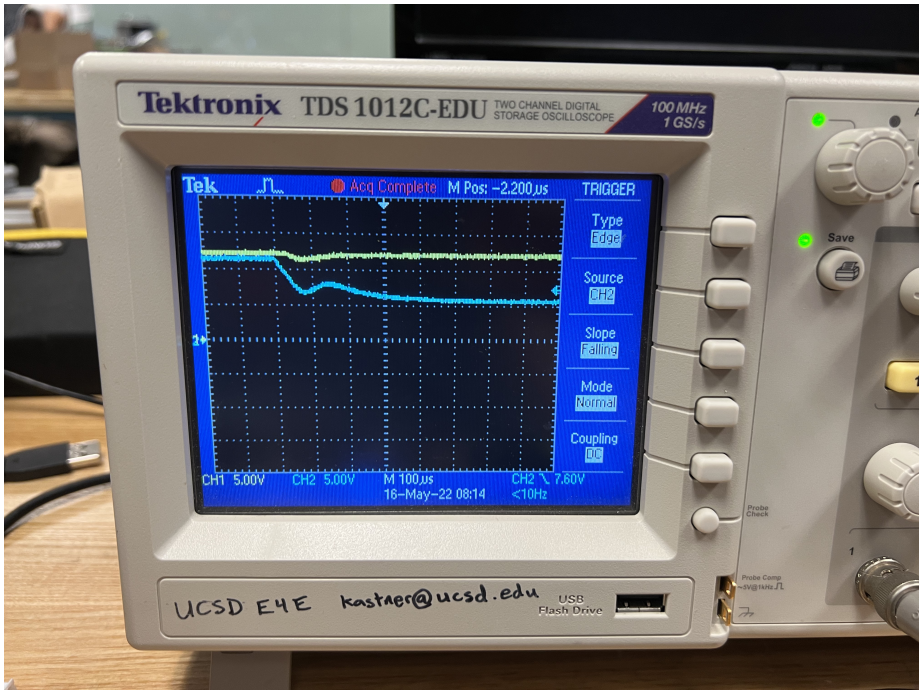


Figure 3.14: Oscilloscope output of Revision 3 with a weak slow start. Blue represents the poweroff signal, while yellow represents the voltage.

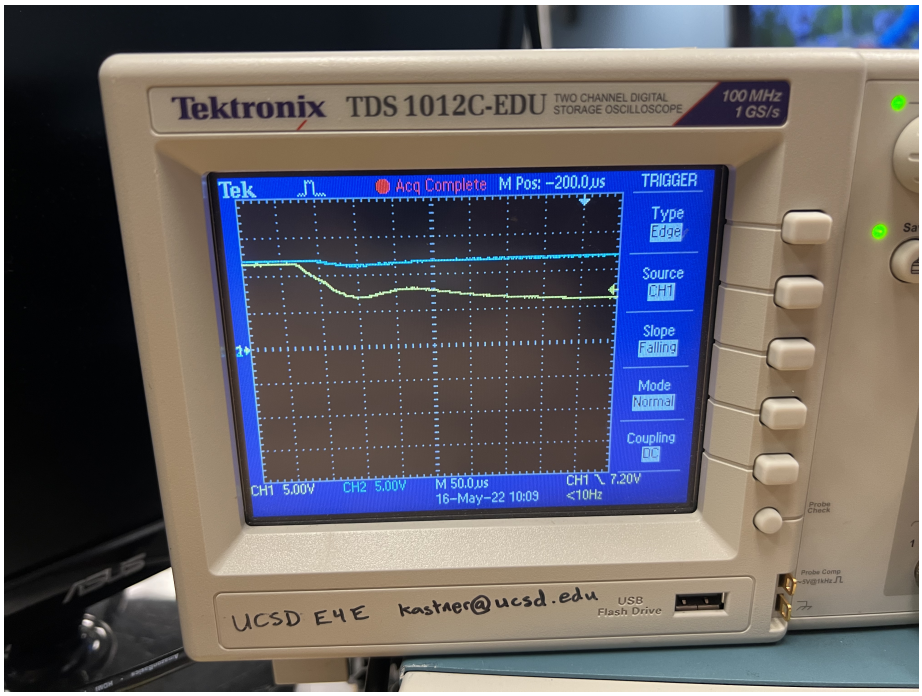


Figure 3.15: Oscilloscope output of Revision 3 with a strong slow start. Yellow represents the poweroff signal, while blue represents the voltage.

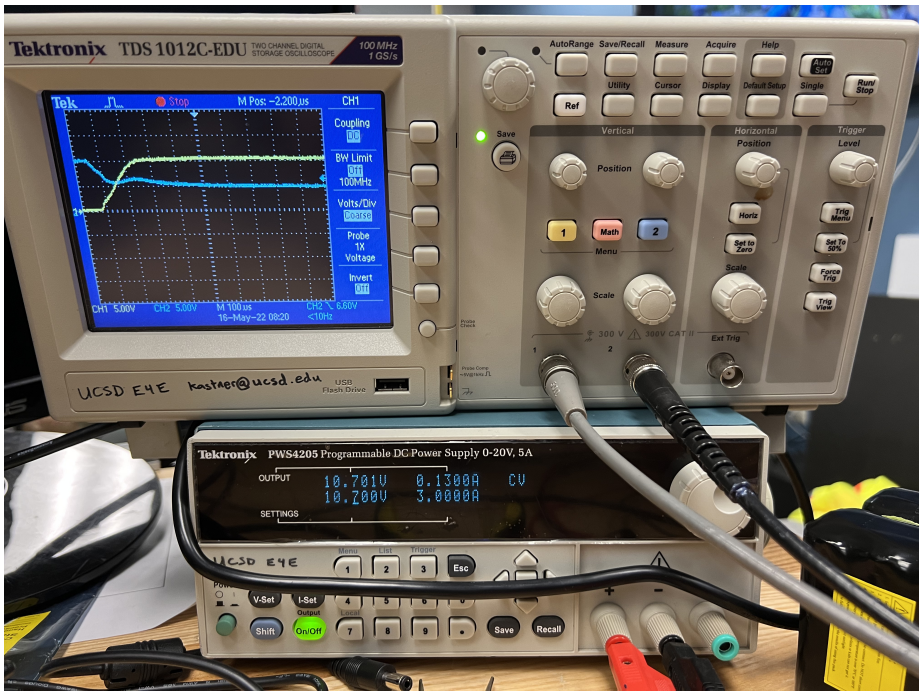


Figure 3.16: Oscilloscope output of Revision 3 hysteresis. Blue represents the poweroff signal while yellow represents the voltage.

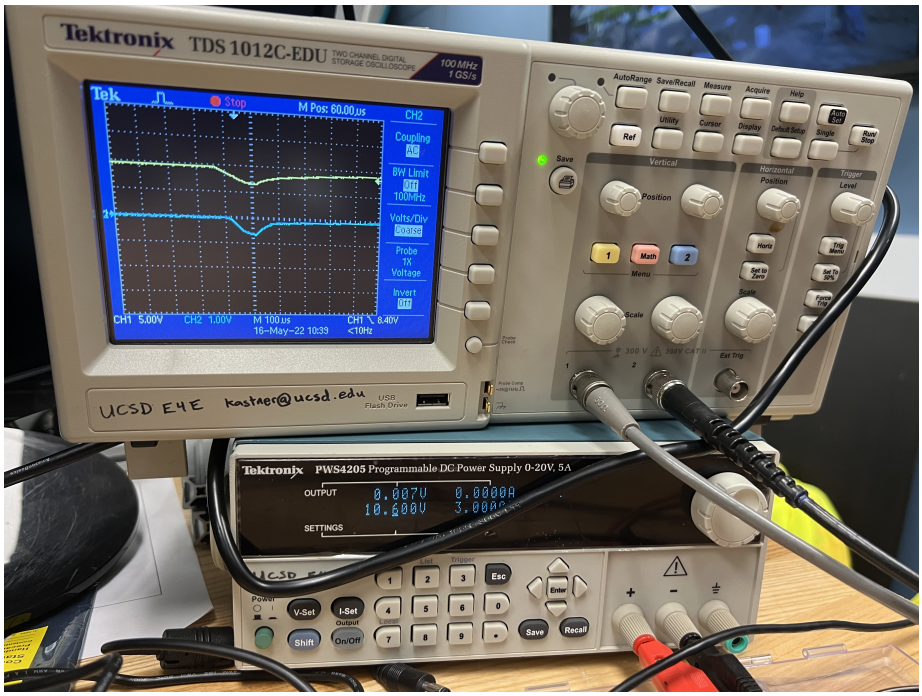


Figure 3.17: Oscilloscope output of Revision 3 poweroff oscillation. The poweroff signal (blue) reads negative because of the AC coupling from the oscilliscope. The yellow signal represents the voltage.

Chapter 4

Mechanical System

The mechanical system design was driven by the key principles of affordability, usability and repeatability. Ever since our earliest experiments, we found that Blue Robotics acrylic hulls suited the D455 camera well, as the 6in internal diameter fit the 5̃in camera neatly. However, as we quickly learned with our Version 1 prototype (Figure 4.1), the double O-ring flange was challenging to put on or take off, and removal was necessary to get at internal electronics.

In addition, much of the space in the Version 1 prototype ended up going unutilized, as the standard 12in length proved to be unnecessary for the electronics we wanted to place inside. After repeated use across many deployments and pool tests, the unconnected internal components began to fall into disarray as well, with 3-D-printed parts rotating and sliding around inside the tube. Even the handles felt like an afterthought, with a bulky metal bar holding the handles onto the tube via a friction-fit clamp that would slide up and down the length of the tube.

Version 2 of the mechanical system (depicted in Figure 4.2) sought to completely overhaul the first design, taking the elements of the hull that worked and complementing them with a set of well-thought-out features that makes maintaining and operating the FishSense modules very easy. Between the experimental test setup that was attached to an AUV and the Version 1 prototype, we

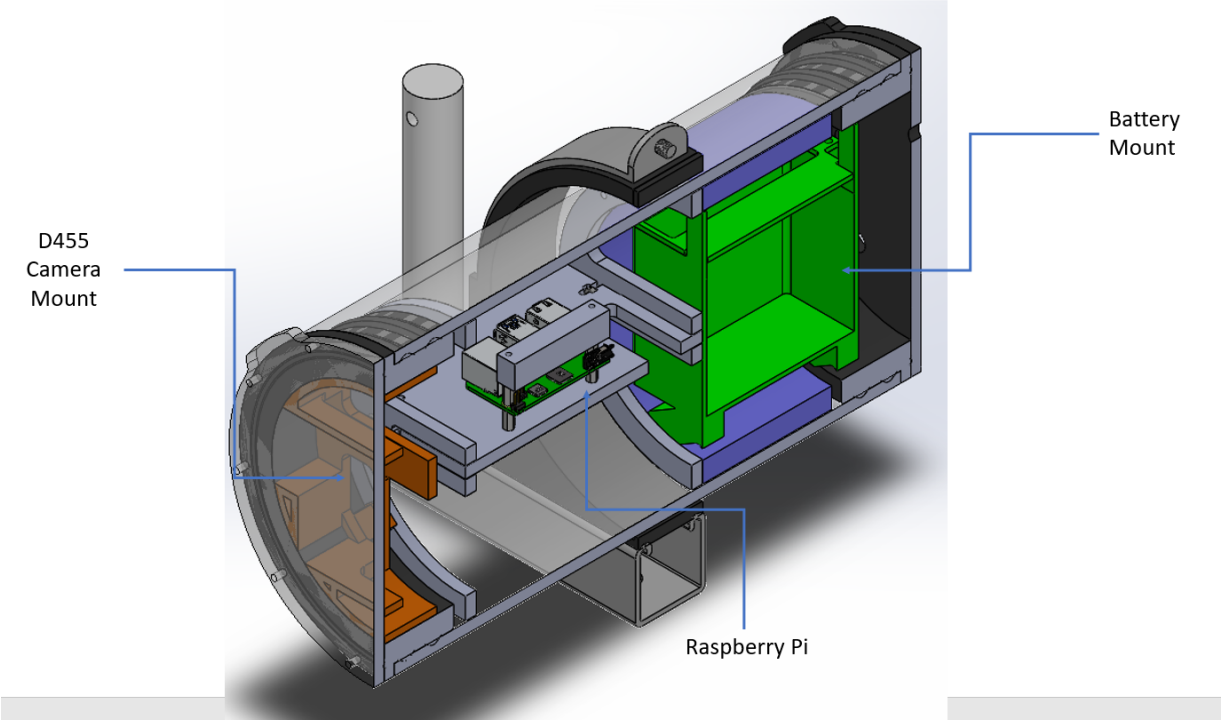


Figure 4.1: Labeled isometric cross-section of the FishSense Version 1 prototype.

collected lots of feedback from divers and other engineers on what the struggles were with using the system, and factored solutions to these issues into the new design.

4.1 Hull Design

The most consistent part of our design across all three underwater testing configurations we have utilized is the BlueRobotics hull. We chose to use BlueRobotics parts for several reasons: their reliability, affordability, and flexibility. Unsurprisingly, waterproofing electronics chambers is a bit of a challenging task; welds are very difficult to fully waterproof, and custom O-ring seals need to be manufactured with relatively low tolerance because of how easily O-rings can become flattened and rendered ineffective. BlueRobotics has streamlined and refined this manufacturing process to the point that they are able to sell endcaps, flanges, and O-rings in bulk through their online store, and at fractions of the cost of other underwater enclosure manufacturers. Because

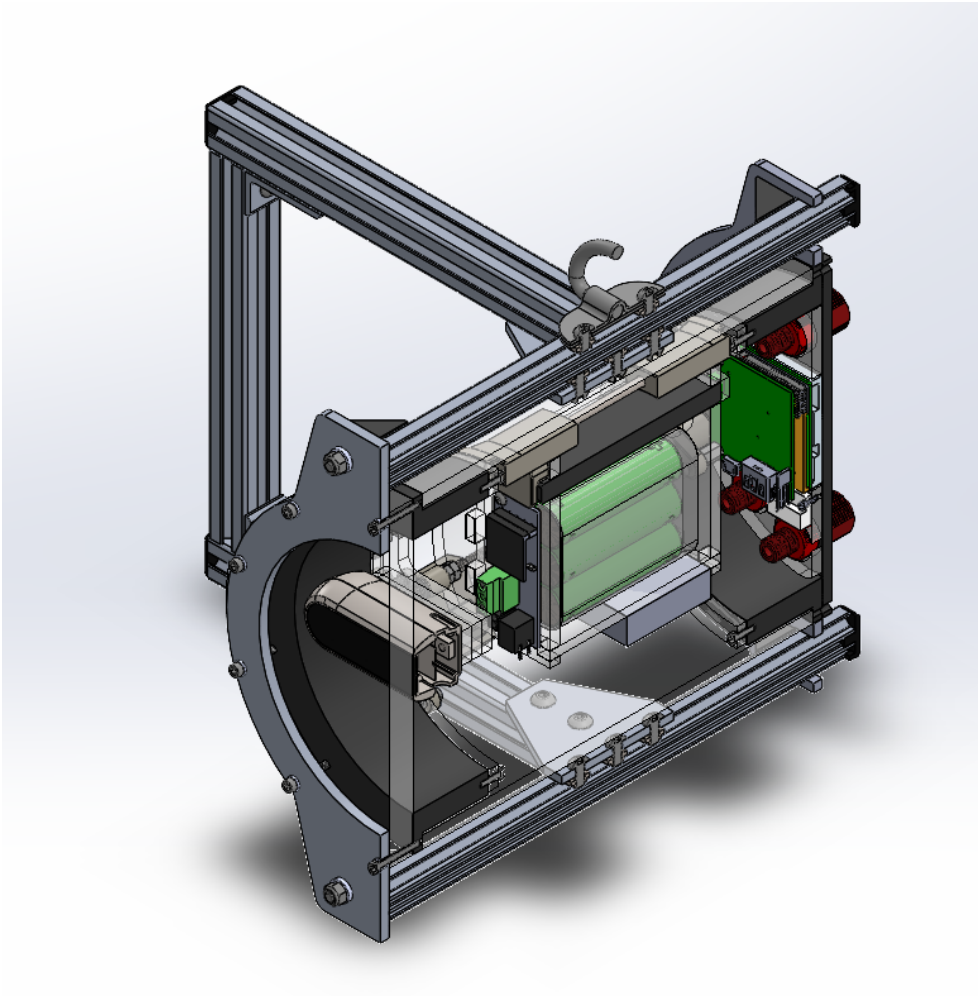
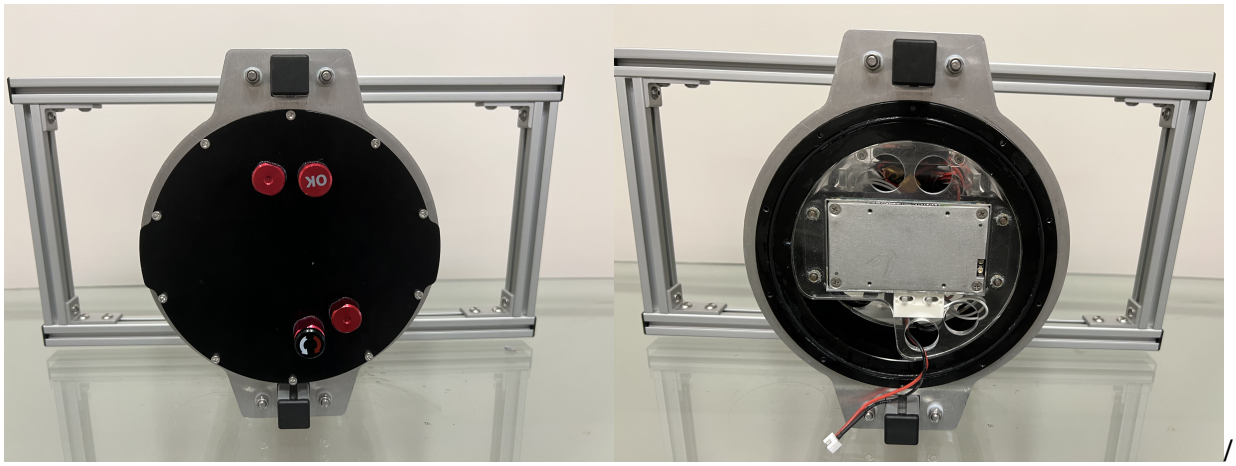


Figure 4.2: Isometric cross-section of the FishSense Version 2 prototype.

BlueRobotics's target market is the researcher or backyard engineer rather than the big oil or defense customers that underwater robotics typically caters towards, their hull served us much better in terms of applications and cost, and had the added benefit of customizable tube lengths and penetrator port placement. We managed to cut down on the length of the tube from 12in to 9in, making our system incredibly compact compared to the previous version.

In addition to the BlueRobotics hull, they provide lots of very useful penetrators, like a vacuum test valve or a power switch penetrator. This enabled us to simplify our design, and more of their custom penetrator options will be discussed in Section 6.1.3. We chose to include 4 penetrator ports for the Version 2 prototype, placed in a custom-drilled configuration that allows



(a) (b)

Figure 4.3: Penetrator port layout on the Version 2 prototype.

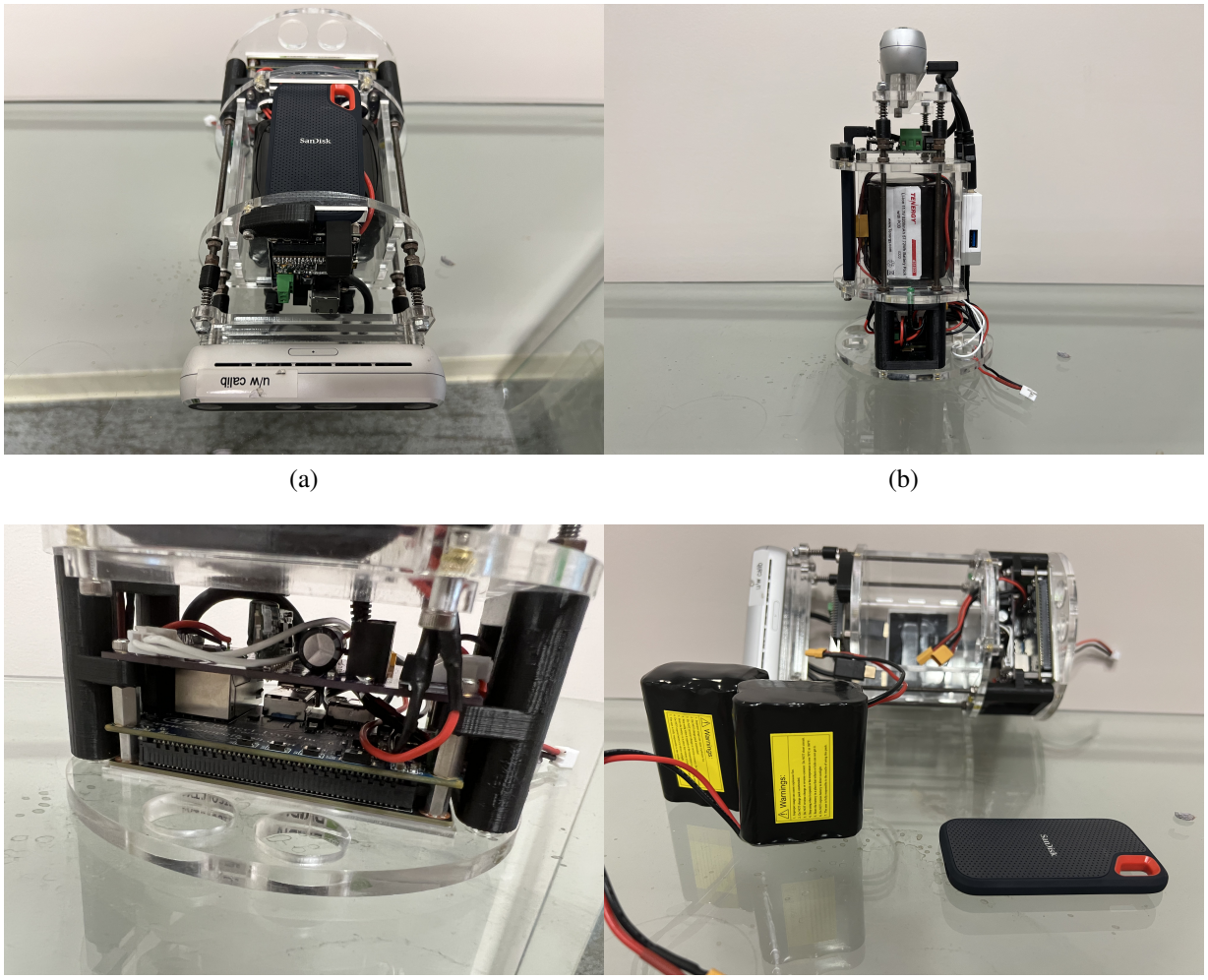
us to press the TX2 against the aluminum endcap for cooling, while maintaining options for real-time communication with the surface, external sensors or devices, or power from the surface. We only utilized two, one as a pressure release valve and another as the power switch for the system.

The last major upgrade we made to the Version 2 system was with the handles. These new handles utilize off-the-shelf T-series aluminum rods instead of custom-cut steel and 3-D printed handles, and is mounted to the system via the screws on the frontplate, which makes for a much stronger mount than the previous iteration. The design of the rods means that other sensors, handles, or hooks could be attached to the handles with little more than a screw, and a crane mounting point is placed at the top of the system to make deployments easier as divers move equipment from the deck of a boat into the water.

The only major downsides with the current hull design are the ease of assembly and the buoyancy. The amount of screws makes the outer hull a bit tedious to assemble and disassemble, but luckily most users shouldn't need to remove anything but the back plate for battery recharging. The module is also currently a bit bottom-heavy, meaning that it has a tendency to point up as divers are using it underwater. This presents a bit of a challenge in terms of usability, but luckily

it can be corrected with weights and skill.

4.2 Electrical Rack Design



(a) (b)

(c) (d)

Figure 4.4: Electronics rack for FishSense Version 2 prototype.

The electrical rack in the Version 2 system (Figure 4.4) is vastly different than its predecessor; instead of treating the camera mount, electronics rack, and battery holder as separate components, we chose to roll all three sections into one removable design. By connecting everything together, servicing parts becomes much easier, as you no longer need to remove the O-ring

flange to get access to the batteries or the compute board, as depicted in Figure 4.5. In addition, the parts are much more effectively locked down, making it much less likely for a wire or other connection to break through jostling. The tray has 3-D printed guide rails to make inserting and removing the electronics very straightforward, and it also forces the rack into a consistent orientation, lending further immovability to the parts.

Figure 4.5: FishSense Version 2 prototype with the electronics rack half-removed.

The electronics rack presses the D455 camera up against the front plate using springs. Ensuring contact with the front plate diminishes reflections between the acrylic and camera face from interfering with the D455's readings, and allowed for a noticeable improvement in the data quality during initial field tests.

The battery holder, placed in the middle of the rack to optimize balance, contains the two 3s2p battery packs that power tye system. In order to hold them in place, we used the SSD as a lock, holding them in the box from the top. A trigger, as well as the custom-measured holes in the acrylic frame, holds the SSD in place. We had some extra space in between the camera and

the battery holder, which we utilized by adding a FathomX tether board. This board, powered off of the same 5V PIO output as the USB hub, can be hooked up to allow a FishSense system to stream in real-time to a computer on the surface, which proved very useful in our initial testing with the AUV, which was equipped with the same board.

Finally that brings us to the TX2. The heat sinks that came with the development and Orbitty carriers were not very useful in this context, because a heat sink functions to dissipate heat from the processor to the fins on the heat sink. However, in an enclosed underwater system like this one, that heat has nowhere else to go within the tube, so the extra volume taken up by a heat sink is not very well-utilized. In order to cool off the processor in the water, we instead take advantage of the high thermal conductivity of the aluminum backplate, as well as the heat-absorbing power of the water around the module to dissipate heat. This ensures that the TX2 is not being heat-throttled, but when we assembled the initial prototypes, we found that the tolerance on the measurements was too loose, and the processor did not achieve thermal contact with the endcap.

With the major components planned out in the design, we needed to add a few more I/O peripherals to complete the rack design. The reed switch for enabling and disabling recording was placed on the back endcap next to the TX2, and the indicator LEDs for power were mounted off of the board and inside a piece of acrylic to maximize their visibility.

This left us with a final design that was affordable (with 3-D printed or laser-cut custom parts holding the commercial components together), easy to use, and effective for gathering high-quality RGB-D data underwater.

4.3 Acknowledgements

Chapter 4, in part, is currently being prepared for submission for publication of the material. Tueller, Peter; Maddukuri, Raghav; Suresh, Vivaswat; Miao, Albert; Drews, Donovan;

Paxson, Charles; Semmens, Brice; Kastner, Ryan. The thesis author was the primary investigator and author of this material.

Chapter 5

Software

Software for the FishSense system consists of two key components: the firmware and the computer vision. The firmware enables communication between the TX2, D455, PIO board, and the user, while computer vision utilizes image processing techniques and machine learning applied to the data captured from systems in the field. Much of the focus of the project thus far has been hardware-focused, because we needed to assemble some complete modules in order to collect large volumes of field data to refine software post-processing, but with the complete systems and summer deployments scheduled, post-processing strategies will become a lot more effective.

5.1 Firmware

The goal and the implementation of the firmware was simple: we needed a script that could set up the camera, await user input, and start/stop recording based on said input. A state diagram for the firmware can be seen in Figure 5.1, and it was implemented with little more than a bash script and a C++ program.

One of the major advantages of the D455, as discussed previously, is the SDK that Intel

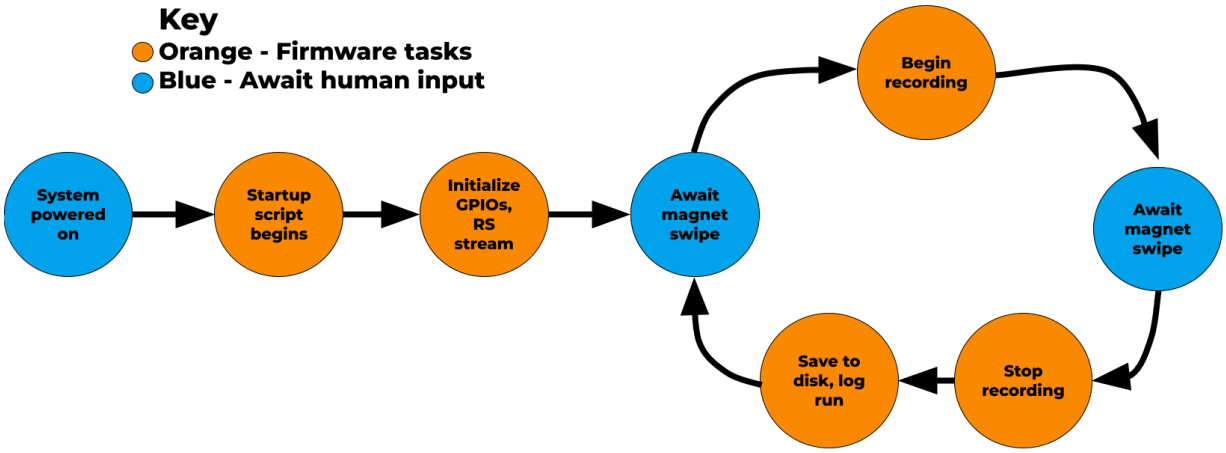


Figure 5.1: Firmware state diagram for all FishSense systems.

provides alongside the hardware. The RealSense SDK was able to handle setting the camera frame rate, saving locations, and actual recording through its own API, so all we had to do was call the appropriate functions at the right time. We use GPIO in order to read the input from the reed switch, which toggles the saving loop until the reed switch is activated again; GPIO is also used to toggle the recording light. Finally, a bash script is run as soon as the TX2 boots up, which begins the C++ program in the background once the system is ready to go.

5.1.1 RealSense Integration

One of the most challenging aspects of the RealSense firmware integration was the calibration of the cameras. Typical RGB camera calibration involves using a checkerboard pattern to measure the distortion introduced by slight manufacturing inconsistencies, and the RealSense depth cameras were no different. The only hangup is that they also need to have a calibration protocol for the depth channel, which is where the calibration board comes in (Figure 5.2). This person-sized dual checkerboard worked with a custom calibration software created by the RealSense team to ensure that proper depth readings were scanned by the camera, and we quickly ran into issues just using it on land.

Figure 5.2: RealSense D400 Calibration Board.

The calibration protocol required the camera to be a very particular fixed distance away from the checkerboard, facing a specific angle towards the left or right depending on the stage of calculation. Very little feedback was given to the user if these alignments were incorrect, and the software was very sensitive to inconsistencies in the calibration setup. Most of our attempts to calibrate the camera on land led to vague error screens like the one depicted in Figure 5.3, and this was before we even got it near water.

The calibration board was made with high-density foam, which made for a sturdy material on land, but an absurdly buoyant material underwater. This meant that we couldn't get the calibration board to even sink just below the surface, making underwater calibration impossible. To circumvent this, we sent cameras to the RealSense team so that they could calibrate them for us; they had an underwater calibration pipeline already set up, so it was straightforward for them

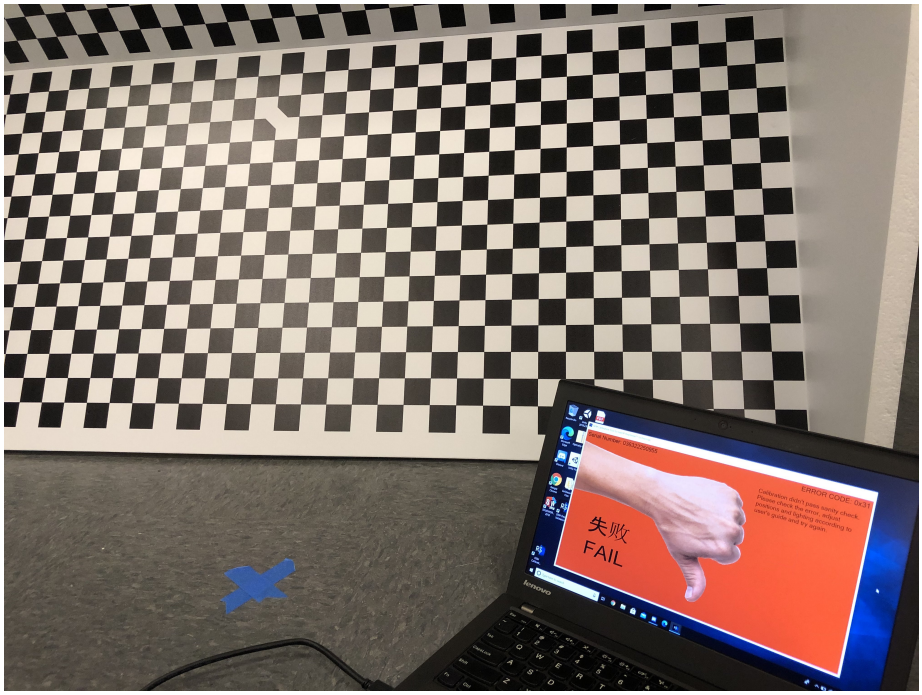


Figure 5.3: Failed calibration attempt.

to run the calibration and send us back the calibrated cameras. To further avoid the issue during initial testing, we copied the calibration settings from these two cameras and averaged them to approximate an underwater calibration setup for uncalibrated cameras, yielding decent results. Further investigation needs to be done to figure out how to take the checkerboard underwater and set up our own calibration pipeline.

The RealSense SDK was similarly tricky to use; inconsistent and outdated documentation often led to strange errors, and the processing pipeline often became backed up when attempting to save at the full 30 frames per second. However, with some trial and error we were able to configure the C++ script to save effectively, reducing the framerate to 15 frames per second due to limitations in the PIO board's maximum current draw.

5.1.2 TX2 Integration

Each TX2 needed to be configured with Nvidia's Jetpack SDK, as well as ConnectTech's custom firmware for port interfacing. Using a combination of official Nvidia documentation [47] and ConnectTech resources[1], we were able to load this required firmware onto the TX2s fairly consistently.

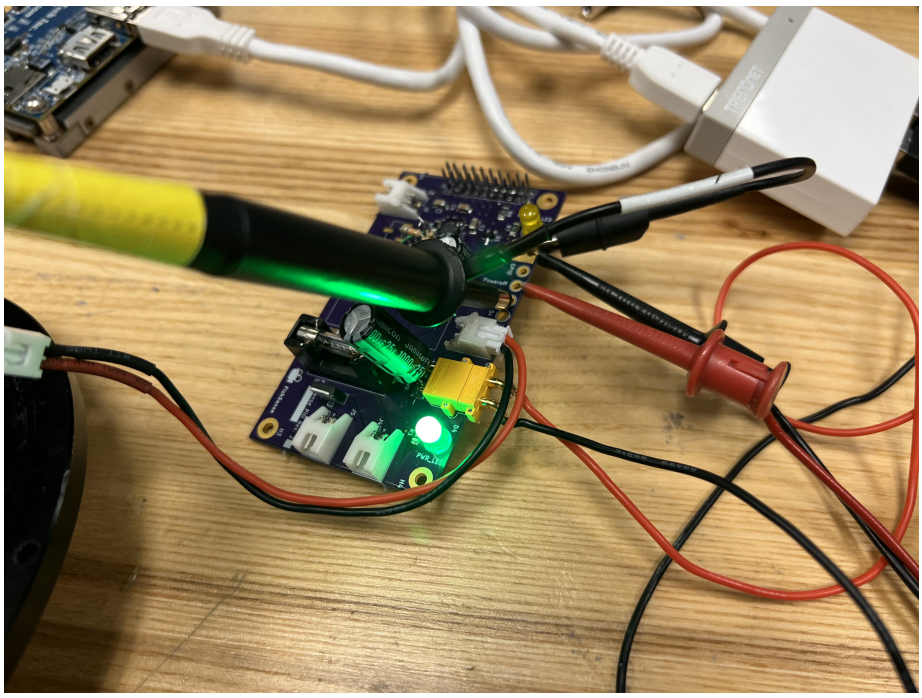


Figure 5.4: LED on the PIO board.

In order to load our custom firmware, we created a Makefile to compile our custom code, copy it to `/usr/bin`, set up the GPIO ports, and connect to the SSD. Once the saving script begins, we interface with the LEDs through the Orbitty's GPIO ports, which turns on LEDs like in Figure 5.4 by simply driving the GPIO pins high or low. When starting up, both LEDS will remain on until setup is complete, at which point the yellow LED will blink then turn off, indicating that it is ready to record. Once the system is actively capturing data, the yellow light will turn back on, and when the system is powered off, the green light will be too. The yellow

[1] https://www.youtube.com/watch?v=9uMvXqhjxaQ

Table 5.1: Summary of major FishSense tests and deployments.

Date	Location	Description	Fish Types	Data Types
December 2020	Birch Aquarium Kelp Forset	70,000 aquarium tank home to scores of fish. Livestreamed to YouTube[2]).	Leopard sharks, Moray eels, Giant black sea bass, many more	RGB `.png` Depth `.png` Depth `.ply`
September 2021	Key Largo, Florida	REEF deployment to the Florida Keys, and the first images from the handheld module in the open ocean.	Lionfish, Greater soapfish, Blue tangs, several more	RGB `.png` Depth `.csv` Species Annotations
July 2021	Birch Aquarium Kelp Forset	70,000 aquarium tank home to scores of fish.	Leopard sharks, Moray eels, Giant black sea bass, many more	RGB `.png` Depth `.csv`
August 2021	Pacifico Aquaculture Center	Fish farm with over 50 cages home to hundreds of thousands of fish.	Striped Bass	RGB `.png` Depth `.csv`
March 2022	Birch Aquarium Kelp Forest	70,000 aquarium tank home to scores of fish.	Leopard sharks, Moray eels, Giant black sea bass, many more	RGB-D `.bag`

LED indicates recording or system status, while the green LED serves as the power indicator.

5.2 Computer Vision

Our experimental setup for capturing image and depth data involved two major sources, a fake fish and real fish. The fake fish, nicknamed Fred, can be seen in Figure 5.5, and RGB-D images of him were collected in lab, in pool tests at the Keck/OAR pool, and at the Birch Aquarium. RGB-D images of real fish were captured through several deployments, and Table 5.1 provides an overview of the dates, testing locations, types of fish, and types of data captured. Two locations in particular, the Pacifico Aquaculture Center[3] and the Birch Aquarium's Giant Kelp Forest[4], provided invaluable data for the testing outlined in the following sections. Custom labels were applied to our datasets by hand, primarily using the Computer Vision Annotation

[2]https://youtu.be/f8-bzkFM1fQ
[3]https://www.pacificoaquaculture.com
[4]https://aquarium.ucsd.edu/visit/exhibits/giant-kelp-forest

Tool (CVAT) [48] and LabelImg [49] to draw bounding boxes on the RGB images of the fish. In addition to data collected through FishSense modules, we utilized the Google Open Images dataset [50] as well as the OzFish dataset [51] to bolster our limited collection of labelled fish pictures, although many from the collection needed to be pruned due to poor label or image quality. We reduced the dataset to normally-shaped, unexotic fish in an attempt to optimize the model for the broadest use case.

Figure 5.5: Fred, the fake fish. He is 12.5" snout to tail fork.

Using results from these pool tests and deployments, we were able to experiment with several computer vision processing techniques. At first, non-ML-based approaches were pursued to minimize computational intensity on the Raspberry Pi and become familiar with the data, but the limitations of these methods were quickly found. From there, our focus turned to utilizing a YOLO image detection network trained on our RGB data due to the limited size of our depth dataset. Now, however, with the Version 2 prototype's increased saving and processing capabilities, I can take full advantage of the depth channel's data to create a U-Net segmentation pipeline and most accurately locate fish within a frame.

5.2.1 Motion & Edge Detection

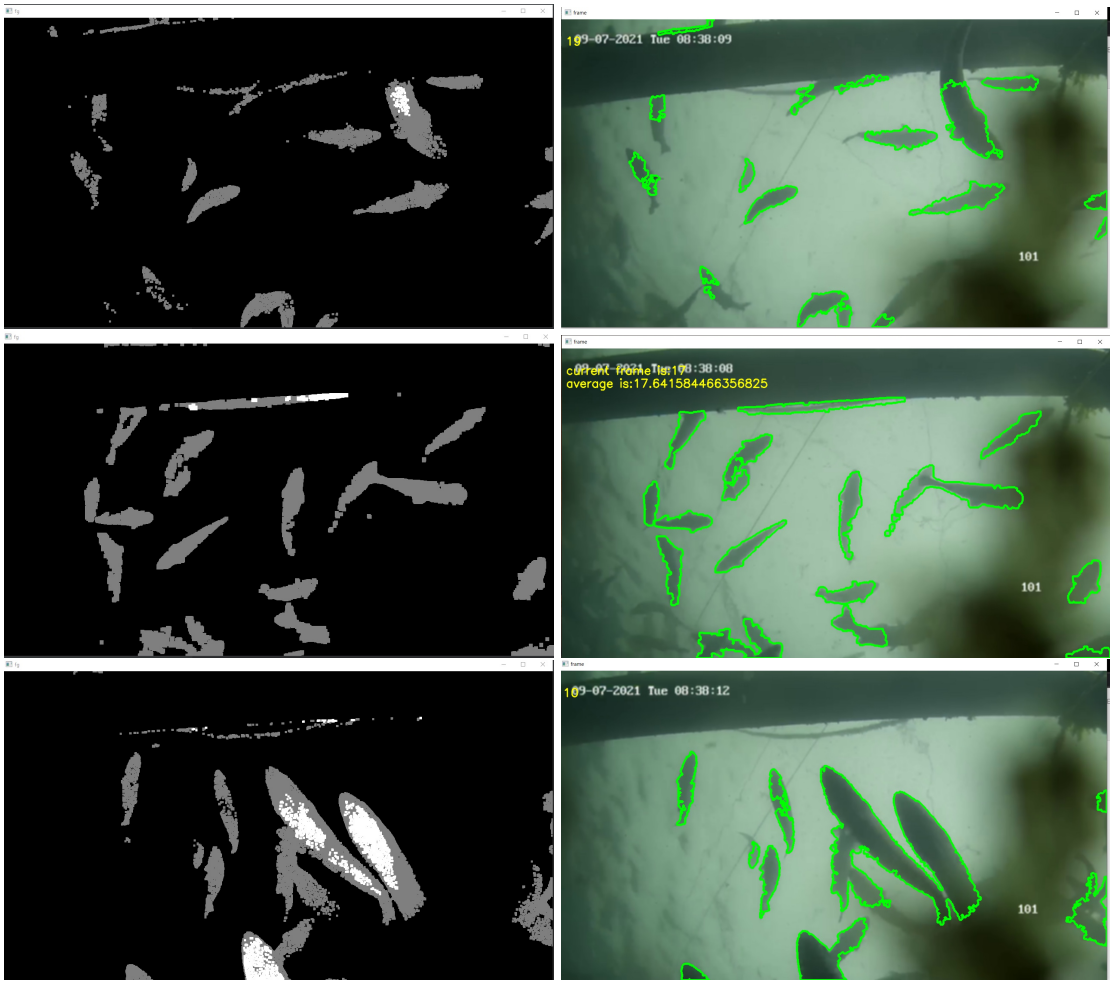


Figure 5.6: Motion detection algorithm applied to Pacifico data. The left column displays the motion filter, while the right column illustrates the motion bounding boxes applied to the original RGB image.

When we first began experimenting with RealSense data, we wanted to utilize relatively simple processing techniques; the limited performance of the Raspberry Pi meant that our data and our real-time processing capabilities were limited. We began with motion detection, as it was the most straightforward; by ignoring the pixels that weren't changing and drawing bounding boxes around the ones that were, we could extract the shape of a lot of fish at once [52]. We applied this technique to the image data collected from Pacifico Aquaculture because their cameras were

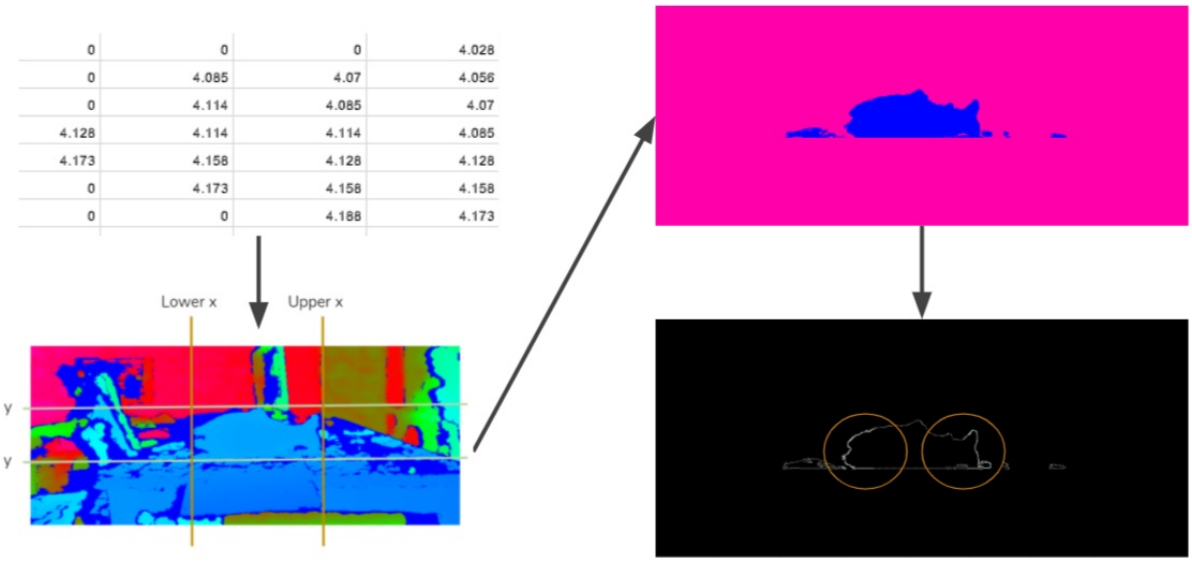


Figure 5.7: Canny edge detection pipeline.

fixed in one position, minimizing the impact of camera movement in the detection. As you can see in Figure 5.6, this technique performed reasonably well on drawing bounding boxes around fish in frame, but had a fairly high false positive rate with all of the other movement caused by tides in the water, as well as a good deal of false negatives for fish in the background.

The next thing we attempted was Canny edge detection [53]. This classic algorithm was created in the 1980s, and thus has an extremely low compute cost compared to machine learning-based approaches. Figure 5.7 illustrates the process of turning depth readings into a segmentation mask, and how we are able to extract the edges from the mask to find the fish. While this performed decently well in lab, the messiness of underwater data caused this approach to struggle a great deal, and we eventually abandoned both of these approaches in favor of more modern techniques.

5.2.2 YOLO Model

You Only Look Once (YOLO) V3 [54] is a powerful and robust object detection network renowned for its incredible speed and ease of training, in addition to its accuracy. Figure 5.8

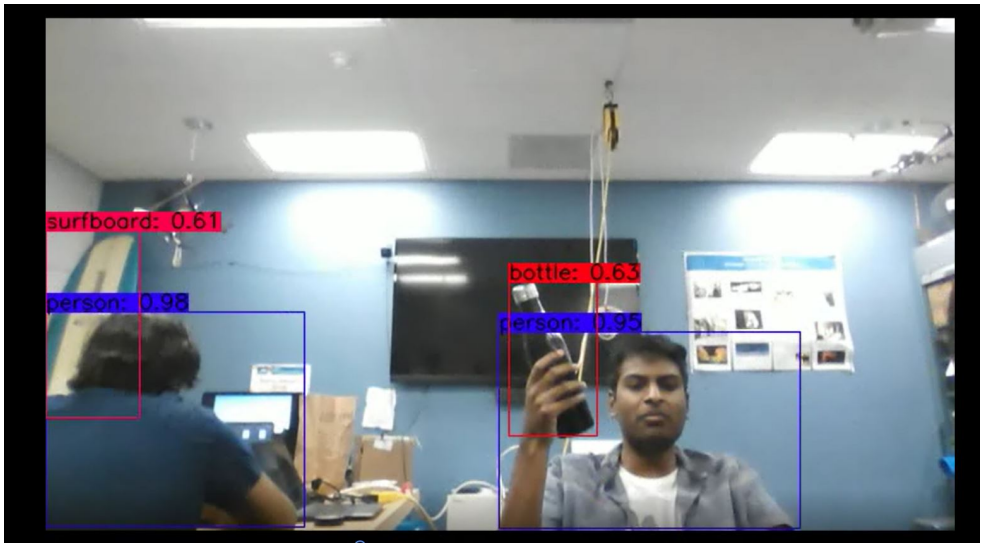


Figure 5.8: Stock YOLO results trained on CocoNet dataset.

illustrates the model's ability to identify a variety of objects within a frame, and we took the baseline model and trained it on several different combinations of data. To start off, we focused exclusively on the data that we collected from Birch Aquarium to train a detector, which yielded some promising results on the data from Birch (as seen in Figure 5.9). However, this model struggled to generalize to other environments, as evidenced by Figure 5.10. As you can see, when the number of fish increased, the model began to struggle a lot more with identifying all of the individuals in a frame, especially as they got further away from the lens.

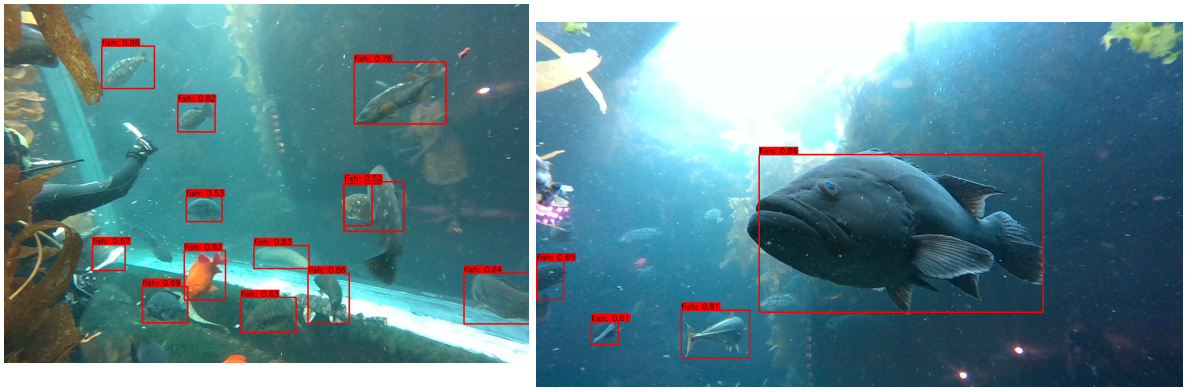


Figure 5.9: Custom YOLO model results trained only on Birch Aquarium data.

When we augmented the training with the OpenImages and OzFish datasets, we found that

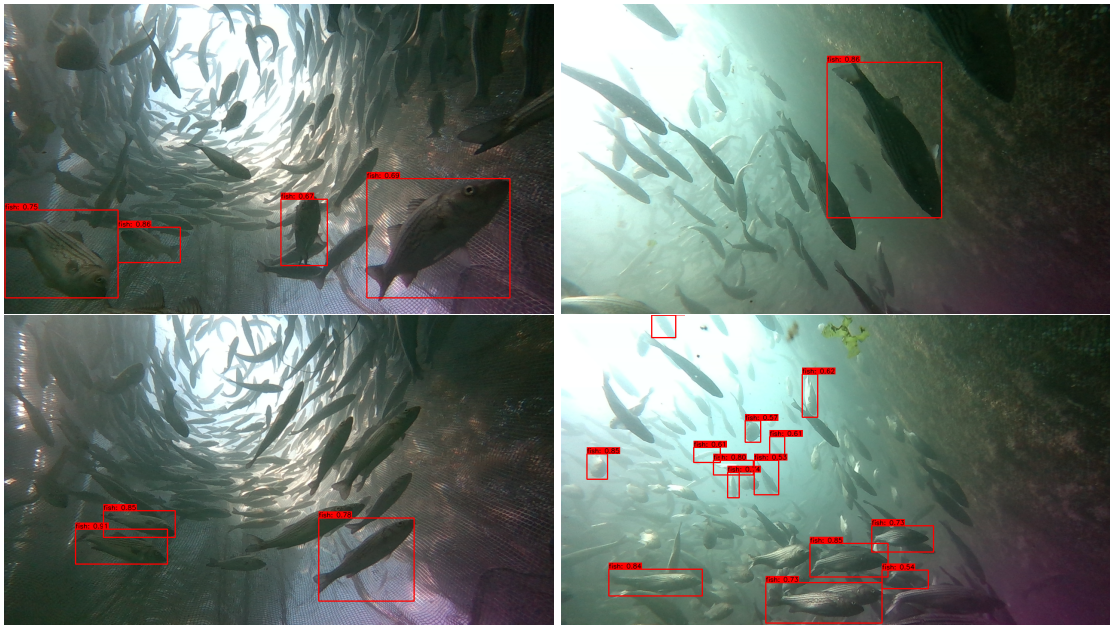


Figure 5.10: Custom YOLO model tested on Pacifico Aquaculture tank data.

the performance actually dropped. This is most likely because the model was now underfit to the specific environments we had captured, meaning that either more training or more hyperparameter tuning needed to be conducted to improve results.

To further improve the model, we implemented YOLO V4 [55], which added even more speed and accuracy to the detection. In order to speed up the convergence of training, we started off by using the YOLOV4-Tiny [56] and YOLOV5 [57] models, which had fewer parameters and thus faster training times. Finally, we ended the training with YOLO V4, which added robustness to the detection results.

In this new iteration of the model, we also added independent classes for the head and tail of fish, allowing us to move closer to fully-automated length measurement. A test result can be viewed in Figure 5.11, in which both the fish and its head/tail were detected. However, as you can see, the head/tail detection still needs a lot of fine-tuning, and even with the improvements offered by YOLO V4, the best accuracy we could achieve was around 80%. This accuracy is not adequate for applications in conservation or fisheries management, so further post-processing

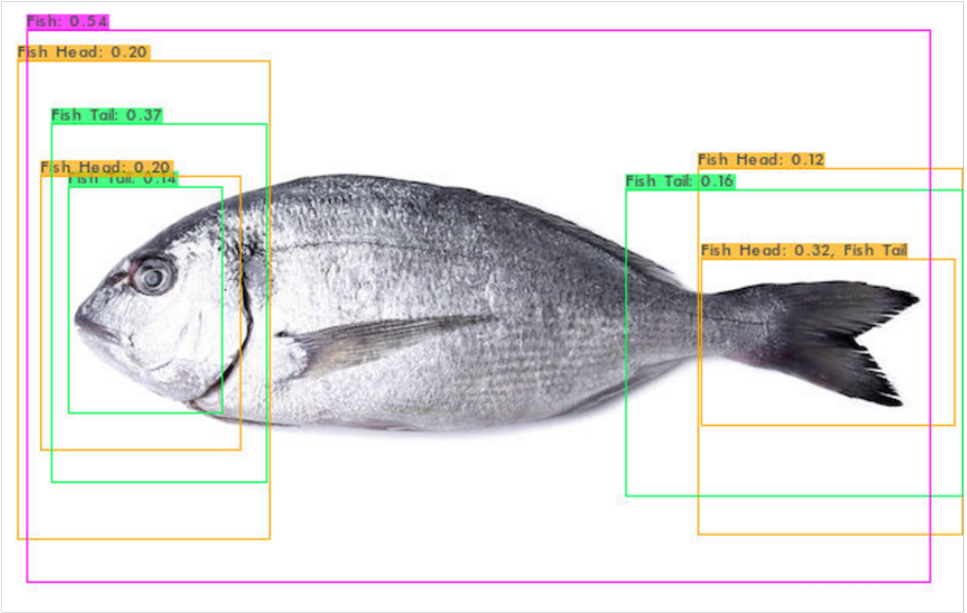


Figure 5.11: YOLO V4 results on test image.

refinement must be done as we begin to collect more data.

5.2.3 Segmentation with Depth Data

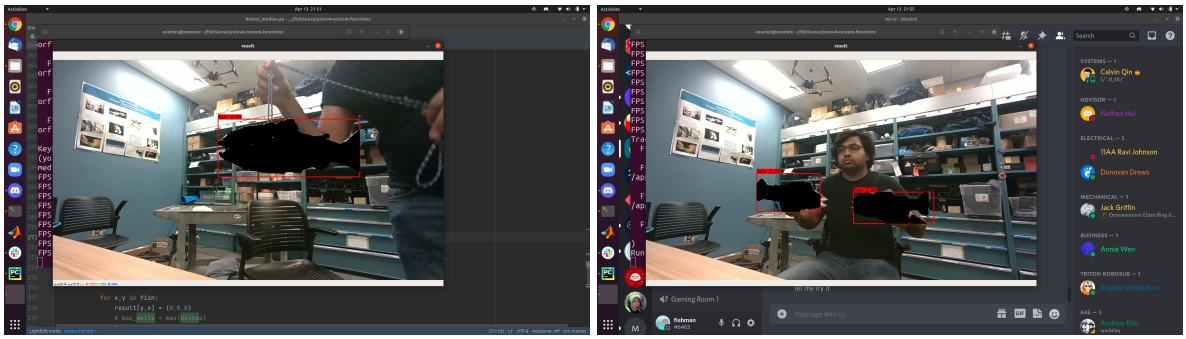


Figure 5.12: Depth gradient results

There are several different approaches that we have tried to improve our fish identification via segmentation. The first, and again most computationally straightforward, is simply finding a depth gradient between fish and background. This works decently well when there's only one or two fish, as Figure 5.12 illustrates, but the strategy begins to fall apart as the image quality

strays further from the ideal. Figure 5.13 depicts one such case, where the fish is too close to its background for the algorithm to differentiate the two. One can imagine how poorly this method will perform as we get the hundreds of fish that aquaculture centers like Pacifico process on a daily basis, and this method was quickly replaced.

Figure 5.13: Depth gradient results

Another segmentation strategy we employed utilized the depth gradient images generated by the RealSense camera. As you can see in Figure 5.14, depth gradient images of the large schools of fish contain fairly clear fish segmentations, and these segmentations can be extracted to determine individual fish lengths.

5.2.4 Length Measurement

Once we have fish, length measurement is relatively straightforward; we just need to be able to identify parts on a fish. Length measurement for fish is typically done from the tip of the head to the meeting point of the upper/lower halves of the tailfin (called the "fork"), but

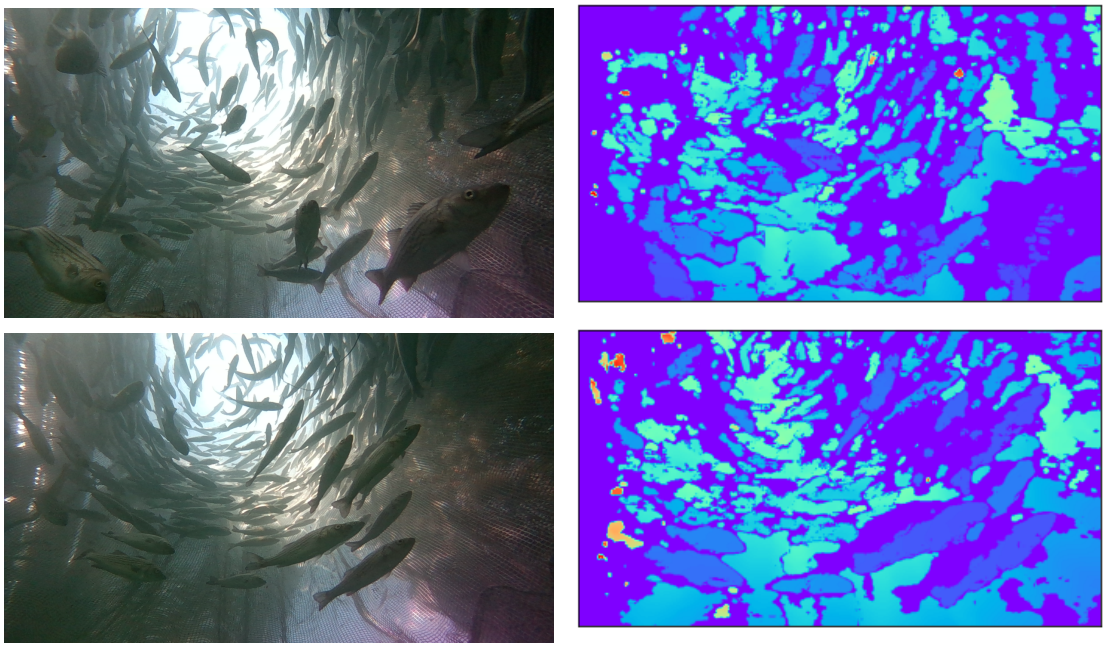


Figure 5.14: Depth gradient images corresponding with RGB data captured at Pacifico Aquaculture.

the automated head and tail detection attempted with YOLO is not quite accurate enough to be utilized for measurement yet. As we collect more data and improve the model, we will be able to locate the center of the head and tail bounding boxes, and calculate the fish's length.

In the meantime, we are utilizing a median segmentation algorithm to determine length given a bounding box. Using a Depth-First Search (DFS) algorithm starting from the center of a bounding box, we can measure points within the frame that are a certain threshold away from the median, and gather the maximum and minimum points to calculate the length of the fish with trigonometry. This method performs fairly consistently in air, but initial experimentation has show that the added noise of the underwater environment introduces inaccuracies with the depth measurements that throw this algorithm off.

5.3 Acknowledgements

Chapter 5, in part, is a reprint of the material as it appears in IEEE/MTS OCEANS 2021. Tueller, Peter; Paxson, Patrick; Maddukuri, Raghav; Suresh, Vivaswat; Ashok, Arjun; Bland, Madison; Wallace, Ronana; Guerrero, Julia; Semmens, Brice; Kastner, Ryan. The thesis author was the primary investigator and author of this material. Chapter 5, also, in part is currently being prepared for submission for publication of the material. Paxson, Patrick; Tueller, Peter; Maddukuri, Raghav; Suresh, Vivaswat; Miao, Albert; Drews, Donovan; Paxson, Charles; Semmens, Brice; Kastner, Ryan. "FishSense: 3D Capture and Analysis for Fish Detection and Stock Assessment". The thesis author was the primary investigator and author of this material.

Chapter 6

Future Work

Truly, the possibilities for the technology utilized in FishSense modules are endless. Throughout my time interviewing scientists, entrepreneurs, marine technology experts, and even during casual conversations with people interested in this work, new ideas for how else these cameras could be used are given to me. Even while writing this paper, I pondered the value in using the camera to identify garbage accumulating on the ocean, and have manipulators or a conveyor belt remove the pollutants, but alas I digress. This section outlines some of the ideas we have begun to explore, and can serve as a launching point for future development.

6.1 Electrical

6.1.1 Intel RealSense Alternatives

One primary issue with the current systems is the fundamental reliance on the Intel RealSense depth cameras for its stereo vision. Because Intel has dissolved the RealSense team and discontinued most of their product offerings, the cameras will not be on store shelves forever, and the SDK will grow more and more outdated with each passing year. Exploring alternatives to

the RealSense camera, such as custom stereo solutions, infragreen or infrablue imagers, or other commercial cameras will eventually be necessary as the D400 series cameras are phased out of Intel's product offerings.

6.1.2 Increasing Computing Power

Using more powerful boards such as the Nvidia Xavier or Orin will vastly improve the performance of this camera in real-time. Being able to offload population surveys and length measurements to the embedded system will not only save people hundreds of hours in data upload times, but it can also give them real-time feedback for more time-sensitive operations like feeding a tank of fish. While the TX2 is quite capable in and of itself, the Xavier and Orin boards both unlock a new dimension of processing speeds, and with CUDA acceleration on the machine learning processing pipeline, one of these systems could easily perform detection and data collection in real-time.

6.1.3 Other Sensors

Utilizing a wider variety of sensors can add much more valuable information for scientists. For example, the addition of a depth sensor would be a straightforward matter of replacing one of the blanks on the current system with a BlueRobotics pressure sensor penetrator, and hooking it up to the TX2. Just this simple addition would allow the diver to record more information on where these fish are found, and could automate the saving process by triggering recording based on current depth.

Other sensors, like sonar, could be utilized for local or global positioning, allowing for more refined location tracking for certain species of fish. Adding a GPS that grabs the last known coordinates of the system can make speedboat surveys much more practical as well.

Finally, the addition of different imagers, like a high-quality RGB camera or imagers in different spectrums, can further improve detection and enable us to classify different kinds of fish

that could not be told apart using visual spectrum processing.

6.2 Mechanical

6.2.1 Boat-mounted System

Creating a mounting system so that the FishSense modules can be utilized on boats adds more practicality and use-cases for these systems. FishSense cameras will be much more reliable in-air, and can identify fish and measure their lengths much more effectively than a human could.

6.2.2 Camera Traps

Modifying the FishSense system's hull to have a longer battery life and baited frame can allow the system to be used as a camera trap. Modules could be placed in locations like river beds, mangrove ecosystems, or coral reefs to image a wide variety of fish and capture important population information for conservation in these crucial areas. The possibility of solar-powered systems also means that these camera traps could be set up permanently, quietly collecting data in parts of the Alaskan wilderness for scientists everywhere to use.

6.2.3 Other Form Factors

While the Version 2 prototype's mechanical design has been refined, it is still lacking a couple of key features like endcap locks typical of marine science equipment today. Improving the form factor to fix the buoyancy issues, allow for more visible LED indicators, and moving away from magnets can vastly improve the usability for divers in the field. For the latter in particular, diver suits come equipped with several kinds of magnets, and accidental triggers were relatively common from preliminary tests.

In addition, changing the form factor entirely may help divers feel more comfortable with

the system; for example, mounting the main electronics on a diver's back with just a self-enclosed D455 module in their hand or on their head would make using the system much easier and more familiar, as many other cameras employ a similar form factor.

6.3 Software

6.3.1 Computer Vision Improvements

First and foremost, collecting, annotating, and utilizing more training data will vastly improve the computer vision results. The lack of high-quality depth data, as well as the lack of things like species annotations, limits the current power of our software post-processing, but the upcoming deployments should change this limitation.

In addition, training hybrid models that utilize both the RGB and depth data can yield results far better than models trained on one or the other. Vectorizing each pair of datastreams to be processed in parallel rather than in series can add much more context to the model, and give it more accuracy in both detection and length measurement.

6.3.2 Low-power System

A low-power mode controlled by another board that wakes the TX2 up when activity is detected and shuts it off otherwise. Some experimentation has been done with an STM32 board; we were able to power-cycle the TX2 using the low-power STM32 using a time-based transition, but further requirements must be defined for the specific applications of this low-power system before real progress can be made. However, the implementation of this strategy would mean that camera traps, increased compute power, and many other optimizations listed here are viable from a battery life perspective.

6.3.3 CUDA Optimization

Running the detection on the system in real-time means that we should capitalize off of the CUDA capabilities of the Nvidia board. As mentioned, the real-time detection would save hours of manual processing and data-loading, and will be crucial for the module's continued success.

Appendix A

List of Abbreviations

3-D: Three-dimensional

AI: Artificial Intelligence

AUV: Autonomous Underwater Vehicle

CNN: Convolutional Neural Network

CUDA: Compute Unified Device Architecture

CV: Computer Vision

D455: Intel RealSense D455 Depth Camera

DFS: Depth-First Search

GPIO: General-Puropose Input/Output

GPU: Graphics Processing Unit

HDMI: High-Definition Multimedia Interface

L515: Intel RealSense L515 LiDAR Camera

LiDAR: Light Detection and Ranging

ML: Machine Learning

PCB: Printed Circuit Board

PIO: Power/Input/Output

REEF: Reef Environmental Education Foundation

RGB: Red, Green, Blue (Color)

RGB-D: Red, Green, Blue, Depth (Color and depth)

RS: Intel RealSense

SONAR: Sound Detection and Ranging

SSD: Solid State Disk

TNC: The Nature Conservancy

TPU: Tensor Processing Unit

TX2: Nvidia Jetson TX2

USB: Universal Serial Bus

YOLO: You Only Look Once

Appendix B

Development Repositories

A high-level overview of FishSense development repositories can be found here: https://github.com/UCSD-E4E/fishsense-docs

The PIO board PCBs can be viewed at https://github.com/DonDrews/fish-hat2.

Our current software can be found on the UCSD-E4E FishSense repository: https://github.com/UCSD-E4E/FishSense.

Bibliography

[1] D. Amodei and D. Hernandez, "Ai and compute," *OpenAI*, 2018. [Online]. Available: https://openai.com/blog/ai-and-compute/

[2] Nvidia, "Jetson modules," 2022. [Online]. Available: https://developer.nvidia.com/embedded/jetson-modules

[3] "Raspberry pi datasheets," 2019. [Online]. Available: https://datasheets.raspberrypi.com

[4] H. Ritchie and M. Roser, "Fish and overfishing," *Our World in Data*, 2021, https://ourworldindata.org/fish-and-overfishing.

[5] Food and A. O. F. of the United Nations, *The State of World Fisheries and Aquaculture 2020. Sustainability in action.*, 2020, https://doi.org/10.4060/ca9229en.

[6] B. Worm, E. B. Barbier, N. Beaumont, J. E. Duffy, C. Folke, B. S. Halpern, J. B. Jackson, H. K. Lotze, F. Micheli, and S. R. Palumbi, "Impacts of biodiversity loss on ocean ecosystem services," *science*, vol. 314, no. 5800, pp. 787–790, 2006.

[7] K. R. N. Anthony, D. I. Kline, G. Diaz-Pulido, S. Dove, and O. Hoegh-Guldberg, "Ocean acidification causes bleaching and productivity loss in coral reef builders," Nov 2008. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2580748/

[8] H. E. Rivera, A. N. Chan, and V. Luu, "Coral reefs are critical for our food supply, tourism, and ocean health. we can protect them from climate change," Aug 2020. [Online]. Available: https://sciencepolicyreview.org/2020/08/coral-reefs-are-critical-for-our-food-supply-tourism-and-ocean-health-we-can-protect-them-from-climate-c

[9] A. J. Jamieson, L. S. R. Brooks, W. D. K. Reid, S. B. Piertney, B. E. Narayanaswamy, and T. D. Linley, "Microplastics and synthetic particles ingested by deep-sea amphipods in six of the deepest marine ecosystems on earth," Feb 2019. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6408374/

[10] G. E. De-la Torre, "Microplastics: An emerging threat to food security and human health," May 2020. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7171031/

[11] A. V. Kontrick, "Microplastics and human health: Our great future to think about now," Jun 2018. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5962470/

[12] N. Strachan, "Length measurement of fish by computer vision," *Computers and electronics in agriculture*, vol. 8, no. 2, pp. 93–104, 1993.

[13] R. Froese and K. Kesner-Reyes, "Impact of fishing on the abundance of marine species." ICES Council Meeting Report CM, 2002.

[14] D. J. White, C. Svellingen, and N. J. Strachan, "Automated measurement of species and length of fish by computer vision," *Fisheries Research*, vol. 80, no. 2-3, pp. 203–210, 2006.

[15] J. Goetze, S. Jupiter, T. J. Langlois, S. Wilson, E. S. Harvey, T. Bond, and W. Naisilisili, "Diver operated video most accurately detects the impacts of fishing within periodically harvested closures," *Journal of Experimental Marine Biology and Ecology*, vol. 462, pp. 74–82, 2015.

[16] K. D. Schramm, E. S. Harvey, J. S. Goetze, M. J. Travers, B. Warnock, and B. J. Saunders, "A comparison of stereo-bruv, diver operated and remote stereo-video transects for assessing reef fish assemblages," *Journal of Experimental Marine Biology and Ecology*, vol. 524, p. 151273, 2020.

[17] N. Harman, E. S. Harvey, and G. A. Kendrick, "Differences in fish assemblages from different reef habitats at hamelin bay, south-western australia," *Marine and Freshwater Research*, vol. 54, no. 2, pp. 177–184, 2003.

[18] R. Dunbrack, "In situ measurement of fish body length using perspective-based remote stereo-video," *Fisheries Research*, vol. 82, no. 1-3, pp. 327–331, 2006.

[19] B. Schlining and N. J. Stout, "Mbari's video annotation and reference system," in *OCEANS 2006*. IEEE, 2006, pp. 1–5.

[20] A. M. Olsen and M. W. Westneat, "Stereomorph: An r package for the collection of 3d landmarks and curves using a stereo camera set-up," *Methods in Ecology and Evolution*, vol. 6, no. 3, pp. 351–356, 2015.

[21] P. Rennert, O. Mac Aodha, M. Piper, and G. Brostow, "Videotagger: User-friendly software for annotating video experiments of any duration," *bioRxiv*, p. 272468, 2018.

[22] J. L. Boldt, K. Williams, C. N. Rooper, R. H. Towler, and S. Gauthier, "Development of stereo camera methodologies to improve pelagic fish biomass estimates and inform ecosystem management in marine waters," *Fisheries research*, vol. 198, pp. 66–77, 2018.

[23] SeaGIS. (2021) Eventmeasure - event logging 3d measurement. [Online]. Available: https://www.seagis.com.au/event.html

[24] M. Adkison, "Adkison advocates increased fisheries data gathering," 2007. [Online]. Available: https://web.archive.org/web/20070711045411/http://www.sfos.uaf.edu/news/story/?ni=184

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2012. [Online]. Available: https://doi.org/10.1145/3065386

[26] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation Applied to Handwritten Zip Code Recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 12 1989. [Online]. Available: https://doi.org/10.1162/neco.1989.1.4.541

[27] K. Chellapilla, S. Puri, and P. Simard, "High performance convolutional neural networks for document processing," 10 2006. [Online]. Available: https://hal.inria.fr/file/index/docid/112631/filename/p1038112283956.pdf

[28] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015. [Online]. Available: https://arxiv.org/abs/1512.03385

[29] D. Reisinger, "Microsoft has finally killed the kinect xbox sensor." [Online]. Available: http://fortune.com/2017/10/25/microsoft-kinect-xbox-sensor/

[30] Z. Zhang, "Microsoft kinect sensor and its effect," *IEEE MultiMedia*, vol. 19, no. 2, pp. 4–10, 2012.

[31] Apple, "Apple unveils new ipad pro with breakthrough lidar scanner and brings trackpad support to ipados," *Apple Newsroom*, 2020. [Online]. Available: https://www.apple.com/newsroom/2020/03/apple-unveils-new-ipad-pro-with-lidar-scanner-and-trackpad-support-in-ipados/

[32] Y.-S. Yoon, S. Hwang, D. Lee, S. Lee, J.-W. Suh, and S.-U. Jung, "3d mesh transformation preprocessing system in the real space for augmented reality services," *ICT Express*, vol. 7, no. 1, pp. 71–75, 2021. [Online]. Available: https://doi.org/10.1016/j.icte.2021.02.001

[33] C. Qi, L. Yi, H. Su, and L. Guibas, "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space," in *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2017. [Online]. Available: https://arxiv.org/pdf/1706.02413.pdfhttps://arxiv.org/pdf/1706.02413.pdf

[34] D. V. Nam and K. Gon-Woo, "Solid-state lidar based-slam: A concise review and application," in *2021 IEEE International Conference on Big Data and Smart Computing (BigComp)*, 2021, pp. 302–305.

[35] F. L. Siena, B. Byrom, P. Watts, and P. Breedon, "Utilising the intel realsense camera for measuring health outcomes in clinical research," vol. 42, no. 53, 2018.

[36] Z. Liu, W. Liu, Y. Qin, F. Xiang, S. Xin, M. Roa, B. Calli, H. Su, Y. Sun, and P. Tan, "OCRTOC: A Cloud-Based Competition and Benchmark for Robotic Grasping and Manipulation," in *IEEE Robotics and Automation Letters (RA-L)*, 2021. [Online]. Available: https://arxiv.org/pdf/2104.11446.pdf

[37] N. Ahmed, "Writing alexnet from scratch in pytorch," 2022.

[38] UserBenchmark, "Nvidia rtx 3090 vs nvidia gtx 580." [Online]. Available: https://gpu.userbenchmark.com/Compare/Nvidia-RTX-3090-vs-Nvidia-GTX-580/4081vs3150

[39] M. Jaffee, "Autonomous underwater vehicles team at berkeley (auvs) robosub 2019: project blue," 2019. [Online]. Available: https://robonation.org/app/uploads/sites/4/2019/10/UC-Berkeley_RS19_TDR.pdf

[40] M. Clark, "Say goodbye to intel's realsense tech by remembering its incredible demos," Aug 2021. [Online]. Available: https://www.theverge.com/2021/8/17/22629528/intel-realsense-3d-camera-tech-wind-down-business-product-demos

[41] M. Moniruzzaman, S. M. S. Islam, M. Bennamoun, and P. Lavery, "Deep learning on underwater marine object detection: A survey," in *International Conference on Advanced Concepts for Intelligent Vision Systems*. Springer, 2017, pp. 150–160.

[42] H. Qin, X. Li, Z. Yang, and M. Shang, "When underwater imagery analysis meets deep learning: A solution at the age of big visual data," in *OCEANS 2015-MTS/IEEE Washington*. IEEE, 2015, pp. 1–5.

[43] R. Pi, *Raspberry pi documentation*. [Online]. Available: https://www.raspberrypi.com/documentation/computers/raspberry-pi.html#typical-power-requirements

[44] C. T. Inc., *Orbitty Carrier for Nvidia® Jetson™ TX2/tx2i - connect tech inc.*. [Online]. Available: https://connecttech.com/ftp/pdf/CTIM-ASG003_Manual.pdf

[45] T. Instruments, "Tl77xxa supply-voltage supervisors," 2016. [Online]. Available: https://www.ti.com/lit/ds/symlink/tl7702a.pdf

[46] Microchip, "Datasheet for mic2779l-2ym5-tr microchip voltage supervisors." [Online]. Available: https://octopart.com/datasheet/mic2779l-2ym5-tr-microchip-66801105

[47] Nvidia, "Jetson tx2 developer kit user guide | nvidia developer," 2020. [Online]. Available: https://developer.nvidia.com/embedded/dlc/jetson_tx2_developer_kit_user_guide

[48] OpenVINO, "Computer vision annotation tool," 2018. [Online]. Available: https://github.com/openvinotoolkit/cvat

[49] T. Lin, "Labelimg," 2015. [Online]. Available: https://github.com/tzutalin/labelImg

[50] "Open images dataset v6 + extensions," 2020. [Online]. Available: https://storage.googleapis.com/openimages/web/index.html

[51] "Ozfish dataset - machine learning dataset for baited remote underwater video stations," 2020. [Online]. Available: https://doi.org/10.25845/5e28f062c5097

[52] N. Singla, "Motion detection based on frame difference method," *International Journal of Information & Computation Technology*, vol. 4, no. 15, pp. 1559–1565, 2014.

[53] W. Rong, Z. Li, W. Zhang, and L. Sun, "An improved canny edge detection algorithm," 2014. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6885761/

[54] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," Apr 2018. [Online]. Available: https://arxiv.org/abs/1804.02767

[55] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," Apr 2020. [Online]. Available: https://arxiv.org/abs/2004.10934

[56] J. Solawetz, "Yolov4-tiny darknet object detection model." [Online]. Available: https://models.roboflow.com/object-detection/yolov4-tiny-darknet

[57] G. Jocher, "Yolov5." [Online]. Available: https://pytorch.org/hub/ultralytics_yolov5/