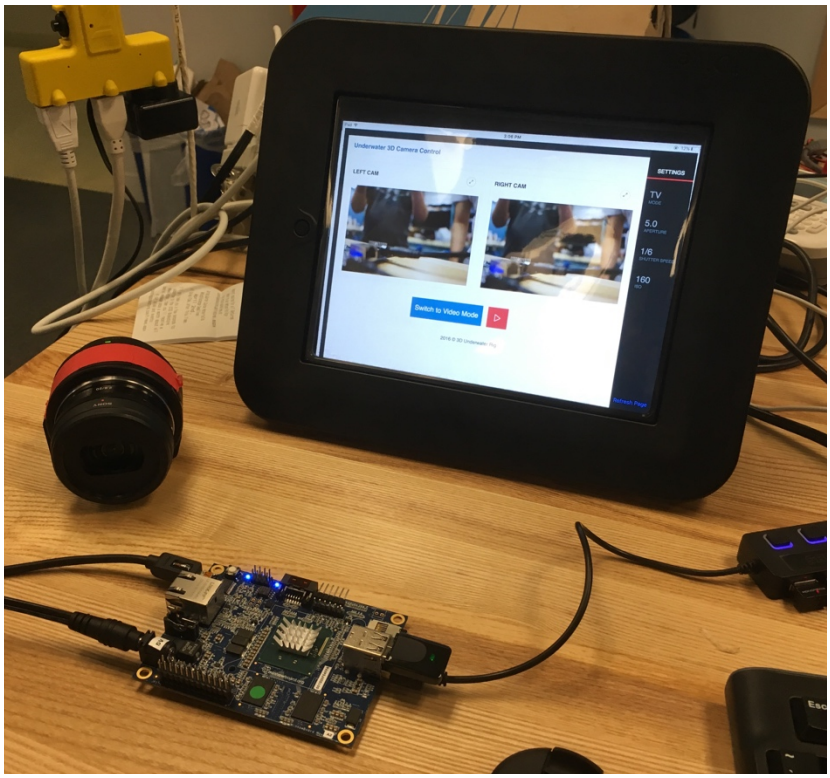
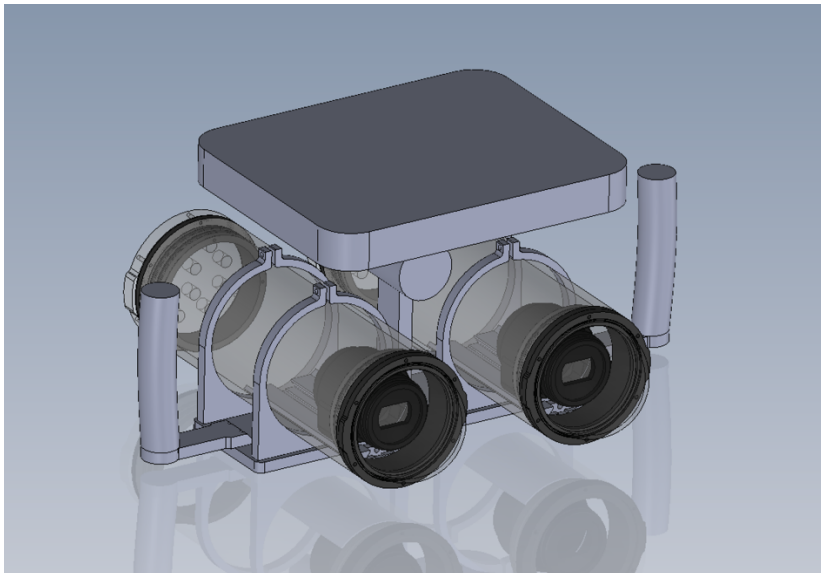
CSE 145 Spring 2016 Final Report

Underwater 3D Stereo Camera Rig

Kevin Xiankun Zhang, Sang Suh, Stephen Gilardi
Advised by Antonella Wilby





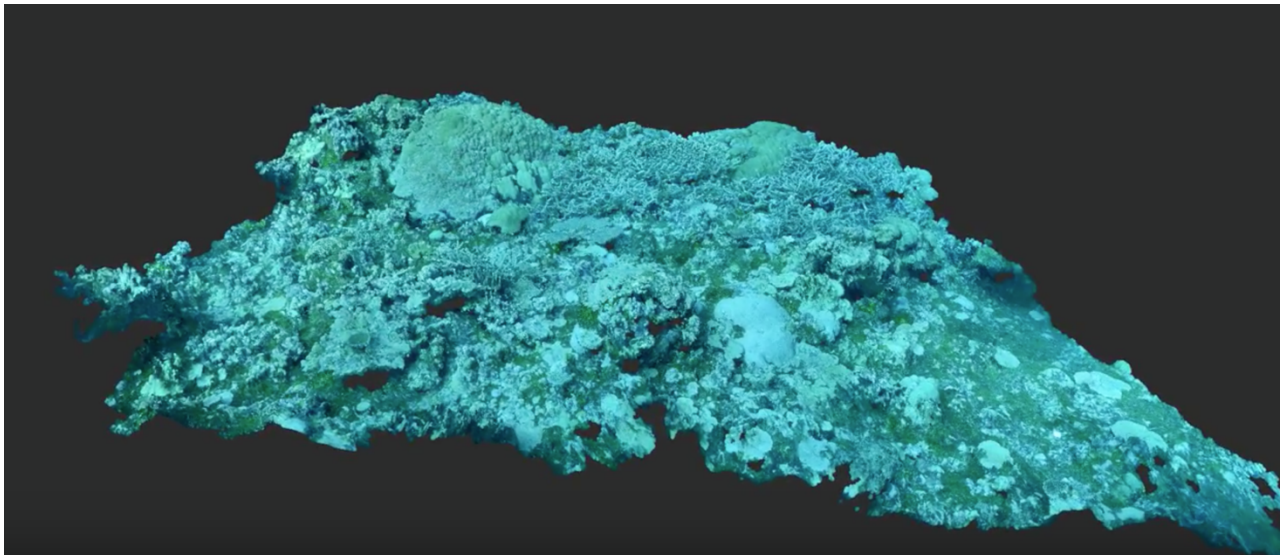
Abstract

Currently there are few general solutions and no commercial products for someone who wishes to create a 3D mapping of an underwater scene. We have contributed to a solution which will be affordable, easy to use, and used to produce high-resolution underwater 3D mapping models. With modifications to pre-existing open source code and using an open-sourced single board computer with an x86 architecture, we can reduce the cost of development even further. The contributions made involve three key components; the embedded web server which controls the cameras, an iPad user-interface to control the rig, and an embedded OpenCV software solution for calibrating and generating a real-time 3D point cloud to be displayed on the iPad.

Introduction

A researcher on the East Coast is one among many who are actively studying a newly found flooded underwater cave in Madagascar which is home to the largest ever found gravesite of lemurs. Flooded caves are of particular interest due to the high quality of preservation that is found regarding the skeletons of the species which may have lived there, and in this case specifically the Lemurs. As quoted in a National Geographic clip regarding this specific cave, “this is probably the largest cache of sub-fossilized lemurs that has ever been discovered”. Since Madagascar is the only place where Lemurs live today, it provides a unique opportunity to study today’s Lemurs variations compared to previous generations which can be found in the cave. The National Geographic clip also states that the skeletons found in the cave can be estimated to be about 300-4,000 years old. Some questions which researchers hope to answer are; how did these lemurs end up in the cave, examination and study into why giant lemurs (hundreds of pounds) have gone extinct, and how various environmental changes may have led to the demise and creation of such a large grave site. Research into this topic is vitally important because it can give us answers into why so many different kinds of Lemurs went extinct, knowledge which hopefully prevent this from happening again, protecting other species even ourselves from following a similar fate. In other words, these answers will only further help us prepare for the future. More information on the National Geographic clip can be found at:
https://www.youtube.com/watch?v=QX46qzzHGNI

To aid in this ongoing research, a solution for mapping this newly found Madagascar underwater cave is needed. With a 3D mapping of the underwater cave, other researchers can study the new findings and contribute in different ways. Currently only those certified to cave dive can look into the new findings which unfortunately leaves many unable to contribute to their full extent. They need a cheap and reliable solution to produce high resolution 3D mappings of the underwater environment to be shared and studied by various other researchers. An example of such a high-resolution 3D mapping is shown on the next page. The goal is to provide similar images of an underwater cave scene taken by professional cave divers who will be swimming and using our custom built stereo camera rig. As stated earlier, the device needs to be cheap to build and maintain, easy to use by cave divers for a lengthy amount of time, and simply just be reliable. The key design choices with the rig is that it will be accessed and controlled using a common iPad interface and will generate the 3D image mappings using two stereo cameras which generates the 3D image by mimicking how eyes work (in which two images from slightly different positions can be used to generated a 3D depth map). The high resolution images will be generated and stitched in post-processing much like the picture below.



Screenshot from https://www.youtube.com/watch?v=wxKfGmxx9B0, "*ANT*" (2015)

Unfortunately building such a complete solution is beyond the scope of a ten-week quarter, so key components were chosen to be worked on and contributed to. In the beginning of our involvement we were given a MinnowBoard, a Sony QX1 Camera, a simple python web server for accessing the camera, and an open source piece of software called OpenCV for generating the 3D point cloud from the two cameras (Stereo Vision). Building upon the initial product, we successfully completed the following sub-sections:

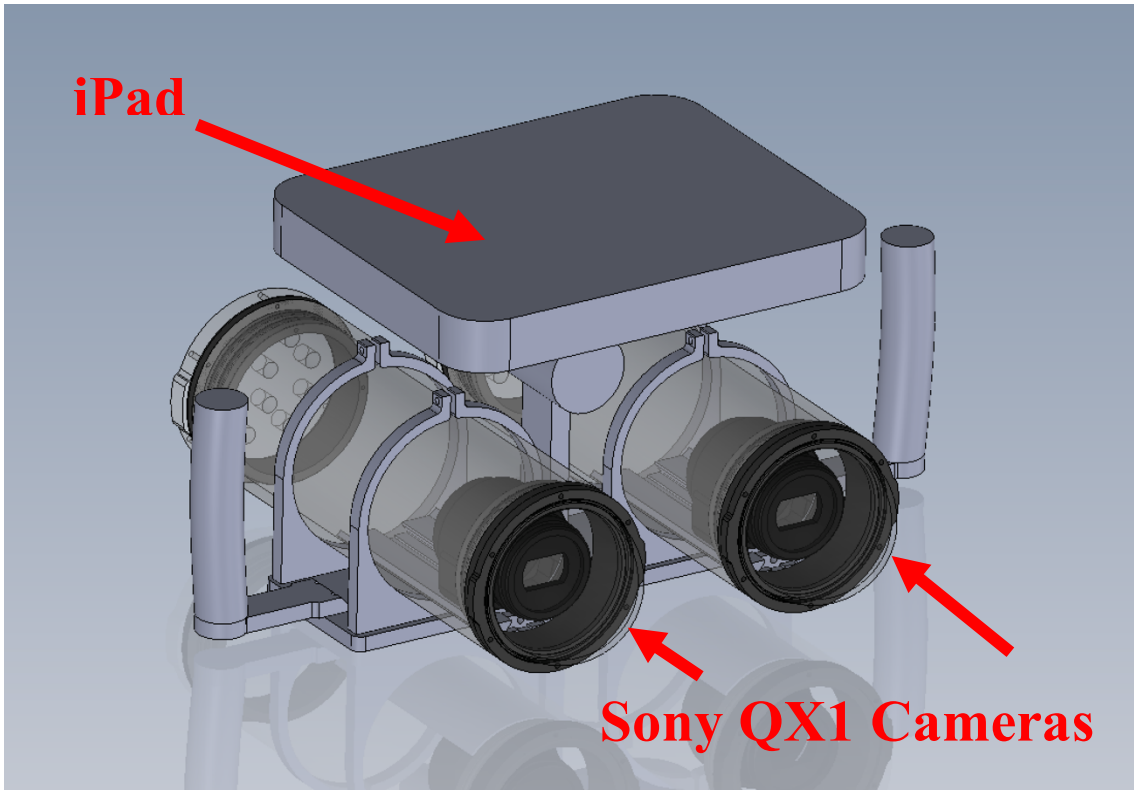
- Embedded Webserver which accesses and controls the Sony QX1 cameras
- Front end UI to be used on the iPad to communicate with the embedded webserver
- iPad Application which frames the GUI for ease of access
- Live Streaming the video feed to the front-end interface.
- Communication between the two MinnowBoard's (Master / Slave)
- Accessing the GPIO Pins on the MinnowBoard
- Configuration software used to configure the two Sony QX1 Cameras
- 3D Point Cloud generated on MinnowBoard

Significant progress has been achieved, and components which were needed have been successfully completed. At the time of this writing, we only have one dedicated MinnowBoard (instead of the two needed), and only one of the Sony QX1 Cameras (instead of the two needed), and don't have the completed housing for the rig. Testing was achieved using similar systems (another laptop, testing with two webcams, etc.). While parts of the testing don't represent the final product, they were chosen and developed in such a way that the results are confident within reason to work just as expected on the final rig with the rest of the components.

Technical Material

To begin looking at the technical aspects of the project, let's looking more specifically into the components of which the rig will be made up of. As shown in the model of our rig below developed in house by a mechanical engineer working at UCSD in correlation with the Sea Lab). At a high level it consists of an iPad and a stereo camera system (two Sony QX1 cameras). The user will interact with our front-end UI on the iPad to control and operate the cameras on the rig. Everything from controlling the camera's various settings (ISO, Aperture, Shutter Speed, White Balance, taking pictures, taking videos, viewing live-stream, etc.) is available on the iPad application.

The next three pages will go into great detail regarding the technical accomplishments and milestones which were met. This includes the iPad, and iPad application, the Minnowboard and its various networks, the Sony QX1 camera, OpenCV and different SLAM implementations, and finally server code which handles all the embeddedlogic.

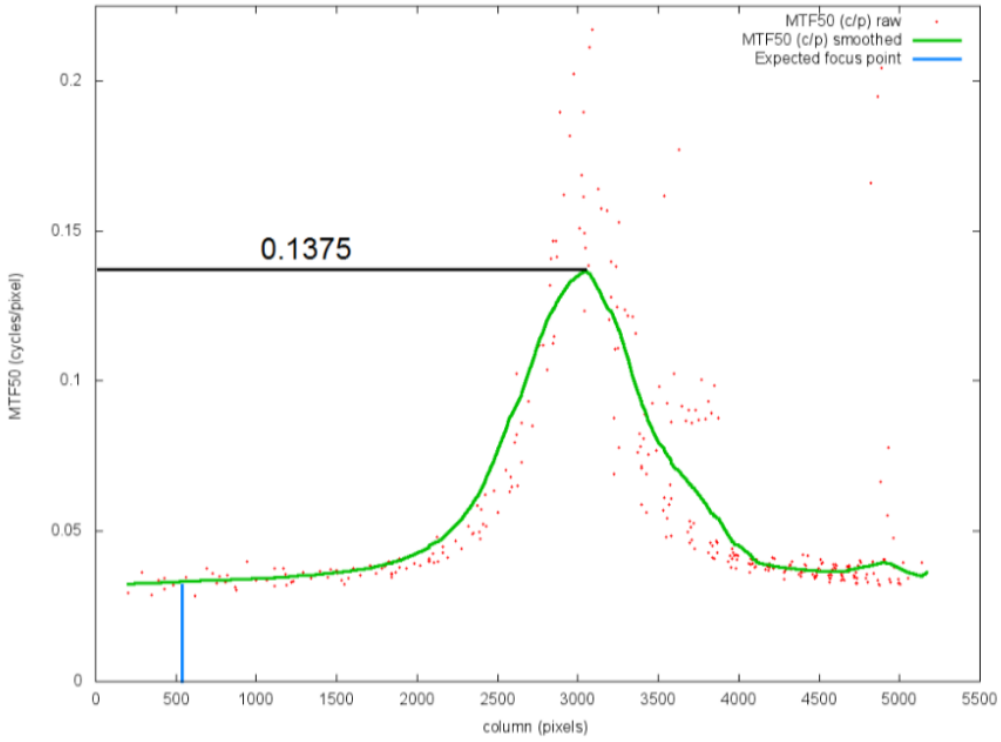


The iPad will be enclosed in a third-party iDive case which is pressure balanced to protect the iPad while underwater and to be safely and conveniently used. The iPad provides the diver with a simple and familiar technology interface where our custom-built application can be easily used to access controls to the rest of the rig. The iPad will be connected to the rest of rig technology using Wi-Fi. The application we created to be run on the iPad is actually a simple web frame which is told to connect to our webserver running on the master MinnowBoard. The MinnowBoard will be broadcasting its own Wi-Fi signal for the iPad to connect to, which in its current state is unfortunately not working as expected.

The user interface acts as a simple portal for all of the necessary controls and commands to be sent to the camera rig and the cameras specifically. The UI was designed with the use case of a diver in mind, using large buttons and convenient menu's to be accessed using the bulky iDive case while wearing diver's gloves. It's vital that the next step be to test the functionality of the interface while underwater and wearing the diver's gloves. Changes will be made as needed in the future to supplement the appropriate and reliable functionality needed by the diver.

As will be explained in more detail when talking about the MinnowBoard in the next few pages, there are multiple networks which are needed for everything to operate as expected. The network between the MinnowBoard and the iPad involves a broadcasted Wi-Fi from the MinnowBoard to the iPad. This involves a DHCP and HostAPD server running on the MinnowBoard which is almost implemented, but left with a bug which is preventing it from working. As a workaround for our purposes, we have introduced an external router which is acting as our broadcast signal for now. It's important to note that this external router is ONLY used to facilitate the communication between the MinnowBoard and the iPad itself.

The two cameras used for our stereo vision are both the Sony QX1. The Sony QX1 is a WiFi controlled camera which provides an easy to use API with which we use to control the various aspects of the camera, and to even grab a live feed from to display onto the iPad. The cameras retail for $399 without the lens, but come in a perfect form factor with a well-supported API. We had to put the cameras under a rigorous form of MTF testing to ensure that the cameras would perform as needed to obtain our high resolution images, even while underwater. MTF stands for Modulation Transfer Function, and essentially measures how well the camera can see small things and differentiate them in a camera scene. For comparisons it was important to document the MTF results for the camera itself, and then again with the camera underwater and in its own housing. For the MTF testing with the cameras themselves (not underwater, and not in the housing), Sang Suh went through a rigorous process in which an experimental method for rating and generating results for the cameras was created. The same tests were run through various scenarios and again with different camera settings multiple times. Not only was this kind of testing important for the comparison of camera performance inside and outside of our underwater housing, but it also emphasized the settings in which the camera worked optimally, thus giving us the option to enable some of those settings as defaults in the cameras themselves when used in the rig.



The picture one the previous page represents the graphical profile for the camera out of the housing and out of the water with a 3:2 aspect ratio using the 20 megapixel setting on the camera. The c/p data can be extracted to calculate the MTF being at about 1/mm. More well documented and explained information can be found in our report on the MTF experiments and calculations, listed in the references at the end of this report. Due to delays with the rig itself (the underwater housing), we were unable to perform comparable MTF testing in the real production settings. The MTF testing with the housing and being underwater is vital to comparing to the performance of what we expect to get out of the camera. These important comparisons can lead to vital design choices for future iterations all in the pursuit of the best quality high resolution image which we can produce for the 3D mapping.

The camera itself is communicated with using its own broadcasted wifi signal. This unfortunately complicates things as we need to connect each camera to its own MinnowBoard to facilitate a reliable connect with each camera. This combined with the broadcasted wifi signal to the iPad, we are operating with three different uniquely broadcasted wifi signals

The on-board computer we're using to host our webserver and the light-weight OpenCV processing is the MinnowBoard MAX. The MinnowBoard is a single board computer much like the popular Raspberry Pi except that it works on the much more popular x86 architecture. The x86 architecture is an important difference because it is very well supported, and essentially behaves very similar to a laptop. For development, this is incredible because a majority of code that we can write and test on a personal laptop is almost guaranteed to work and perform almost exactly the same on the MinnowBoard. The MinnowBoard Max runs an Intel Atom (~1.2GHz) and sufficiently meets our needs for the webserver and OpenCV code performance wise. It's important to note that we haven't been able to stress-test the system in any way yet regarding running everything at once. This will be an important test to perform, and will lead to decisions of reducing framerates / quality of the live streaming images.

The webserver running on the MinnowBoard to control the two cameras is written in Python and utilizes the Flask framework. The webserver on the Master Board is the primary point of access for all other sub-systems which make up the camera rig. It handles JSON requests from the iPad, and then performs the necessary logic for the request. The webserver is connected to one of the Sony QX1 cameras directly, and relays any additional information to the second MinnowBoard which will be connected by an Ethernet cable. This communication between the two boards is facilitated using Python's Web Socket framework, and is used primarily to relay information to the second Sony QX1 camera. For stereo vision requirements, it's important that any camera use is synchronized with very little latency between the two cameras. This is an area which must be tested in the future as we were unable to do so with only access to one MinnowBoard and one camera. The webserver on the master board is also in charge of decoding the images to be sent to the live-view on the iPad. The cameras send a proprietary stream of packets to the webserver, in which we decode and generate a JPEG image to be sent to the iPad. Currently, we are able to run about 15-20 frames per second depending on the strength of the connection between the iPad and MinnowBoard. There are a few bugs present in the live-streaming functionality and in decoding the images quickly enough, a performance test and reliability profiling needs to happen in the future to ensure that specs are met and satisfy various use cases.

A similar server will be operating on the Slave MinnowBoard, except it will be performing much less intensive computations and less networking. Instead of receiving commands directly from the iPad interface, it will be receiving commands from the Master MinnowBoard through a web socket which will be connected via an Ethernet cable. The server will handle these commands and send the appropriate JSON API requests to the second Sony Camera. Not currently started but needs to be completed for final production is the act of sending the live-stream feed from the second camera back to the Master MinnowBoard. Some important consideration to be made for this will be the decision to send the raw feed, or to process the packets and send the decoded JPEG images back to the master server to be quickly sent back to the iPad. Doing the latter easies the computational intensity on the Master MinnowBoard, but may complicate the communication protocols between the two MinnowBoard's themselves.

Code was also written for the two servers to interact with their respective GPIO pins on each MinnowBoard, to be used to remotely power on / off each camera, but the necessary USB cables to communicate with the camera were not received during the course of the 10 weeks, so further code implementation for talking directly to the camera via the GPIO pins is not yet started.

Implementing the SLAM for a sparse point cloud on the MinnowBoard provided its own challenges. While the MinnowBoard provides a lot of processing power for its form factor, it still falls short for full SLAM implementation without modifications. The advantage of portability out weighted the necessity for larger and faster processing power. Starting the investigating into SLAM involved two aspects of research and testing preexisting OpenCV / ROS algorithms with necessary specifications. There was no inertial movement unit (IMU) and the specification required real-time capture and processing, visual odometry, closed loop detection and obviously stereo camera video feed.

Research into SLAM involved finding algorithms that provided the necessary minimum requirements. Since the camera rig used stereo cameras that eliminated many SLAM options. Further investigating involved find solution that were reasonable to implement and had fair amount of research and support. The path led to four reasonable options for SLAM; those are S-LSD, ORB, Stereo, and RTAB-Map.

LSD SLAM is well known, but S-LSD is the relatively new stereo version of LSD SLAM. LSD is Large Scale Direct Monocular SLAM. Instead of using keypoints, it directly operates on image intensities for both tracking and mapping. The camera is tracked using direct image alignment, while geometry is estimated in the form of semi-dense depth maps, obtained by filtering over many pixel-wise stereo comparisons. Then it builds a Sim pose-graph of keyframes, which allows it to build scale-drift corrected, large-scale maps including loop-closures.

ORB SLAM is a method out of University of Zaragoza in Spain. It uses a keyframe and feature-based SLAM. It is able to compute in real-time the camera trajectory and a sparse 3D reconstruction. It is able to close large loops and perform global relocation in real-time and from wide baselines.

Stereo SLAM is a ROS node to execute Simultaneous Localization And Mapping (SLAM) using only one stereo camera. The algorithm was designed and tested for underwater robotics. This node is based on the G20 library for graph optimization and uses the power of libhaloc to find loop closures between graph nodes. It uses a keyframe to multi-keyframe loop closing mechanism, based on keypoint clustering, to improve the SLAM corrections on feature-poor environments like underwater mapping due to light being attenuated in water.

RTAB-Map is a Real-Time Appearance-Based Mapping and is a RGB-D Graph-Based SLAM approach based on an incremental appearance-based loop closure detector. The loop closure detector uses a bag-of-words approach to determinate how likely a new image comes from a previous location or a new location. When a loop closure hypothesis is accepted, a new constraint is added to the map's graph, then a graph optimizer minimizes the errors in the map. A memory management approach is used to limit the number of locations used for loop closure detection and graph optimization, so that real-time constraints on large-scale environments are always respected.

Testing involved first implementing and testing on a laptop and then on to the MinnowBoard. Because there was only one MinnowBoard and extreme issues with version compatibilities and dependencies, testing on the embedded board was limited to only one SLAM method. Two USB cameras were used as the video stream because the actual cameras had not been implemented and there was only one Sony QX1. A lot of work was involved in getting prerequisites needed to implement SLAM. Including errors due to USB bandwidth limitation meant, all live streams resolutions were lowered. RTAB-Map was implemented, but required a program to be written for side-by-side video recording (code is located on the web-site). Code was also implemented to test the calibration and disparity map processing power of the MinnowBoard, before any real slam was implemented on the board itself, which proved successful. Due to images resolution being lowered, it also lowered quality of the mapping and prone to errors of localization and calibration was taking much longer to accomplish (hours). While the feasibility of SLAM on the MinnowBoard is a yes, the quality and reliability requires still requires extensive testing.

In its current state the SLAM code has not been integrated with the webserver, and thus is not present in our current UI. The goal would be to have this information relayed in a low-resolution real time 3D modeling of the environment for the diver to use. This kind of real time feedback would be incredibly beneficial in which it would help detect spots which may not be completed and requiring more pictures to be taken.

All technical material is currently documented in a public Github account used by the team. The goal is to transfer this material to Antonella's BitBucket where it can be continued to be worked on by future students. This paper also serves as a technical note on what has been accomplished so far and some of the technical challenges faced

Milestones

Our project is comprised of three parts – server development, image processing, and embedded controls. At the beginning of this quarter, we promised to deliver the following milestones with the respective deadlines:

1. MTS testing on objects – 4/17/2016
2. Basic layout of UI – 4/23/2016
3. MinnowBoard working with OS and networked – 4/23/2016
4. Get 3D mapping on computer – 5/1/2016
5. MTS testing on optics (with housing and underwater) – 5/11/2016
6. Integrate software, hardware, and UI together – 5/20/2016

As of 5/11/2016, we were able to finish the first 4 milestones, but we did not have the materials available to do the actual MTF testing underwater with the housing. The primary reason being the lack of housing being completed yet, so we planned on doing the underwater testing with the iPad only after the integration would happen at the end of the quarter. We also reevaluated our integration plan with the software, hardware, and UI into more specific atomic tasks to help us keep track of changes for the rest of the quarter. The new milestones are as follows:

1. Integrate Minnowboard with iPad controls – 5/16/2016
2. ROS or OpenCV stereo camera calibration – 5/16/2016
3. Add white balance control – 5/20/2016
4. Implement communication between web server and UI – 5/20/2016
5. Implement communication from camera to OpenCV / SLAM – 5/20/2016
6. Get OpenCV working on Minnowboard – 5/20/2016
7. 3D depth mapping – 5/20/2016
8. Get 3D mapping to work on computer – 5/20/2016
9. Add UI stream to both cameras – 5/23/2016
10. Complete web server controls for the two Sony camera – 5/23/2016
11. MTS testing on optics (with housing and water) – 5/25/2016
12. Integrate hardware / software / UI – 5/27/2016
13. Options on SLAM algorithm for embedded system – 5/30/2016

As of 5/26/2016, wer were able to complete all of the above milestones except for items 5, 6, 7, 10, and 11. The main issue to our progress lies with our constraint with the hardware as we were unable to get a second camera, a second Minnowboard, and the housing for the rig itself. Another technical issue was the complexity with OpenCV which gave us delays with the 3D mapping / SLAM implementations. However, we were able to finish a good majority of our milestones by the end of the quarter and were able to do a demo the embedded camera control functionality.

Conclusion

While significant work and milestones were met over the past ten weeks, there is still plenty of work to be completed in order to finish the final product. We built the primary embedded controls necessary to operate the camera rig while using the iPad interface, and had a significant amount of the work done for the low-resolution. live-view of the current 3D point cloud generated using the live video feed. This work makes up the core product and functionality of the camera rig, and with the additional parts needed we can begin to really finish the first prototype of the system.

The stereo camera rig represents a modular and easy to use system which can ultimately aid the researchers studying the large Lemur gravesite recently found in Madagascar. The stereo camera rig has benefits reaching far beyond just this one cave, and can ultimately be used in any kind of underwater scene benefiting multiple fields of research and general commercial use. The project provides those working on it in the past, the present, and in the future an opportunity to contribute to an incredible project with lots of potential. Some of us hope to contribute to the project beyond the scope of this class, continuing some work in the summer and potentially into the Fall Quarter for even more additional work. Some key points which need to be worked on involve reliability, specifically making sure things always work and that safeguards are in place preventing the system from crashing with small errors.

The primary issue right now are networking bugs, unreliable connections, horrific framerate drops, and other various bugs / issues which pop up occasionally. Once those are fix, the hosted WiFi between the Minnowboard and iPad needs to be address. HostAPD is already configured, but may need to be uninstalled and redone. In an effort to try to finish it by the deadline we may have messed things up, and trying to fix it may be futile at this point. Other than those small things, the project is moving along nicely and we all look forward to seeing it grow and slowly become completed.