Northeastern University

*College of Engineering*

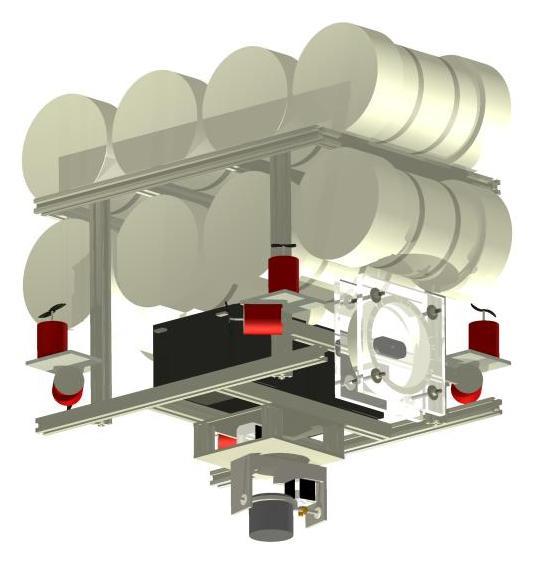
**Department of Electrical and Computer Engineering**

**Boston, MA**

**Capstone 1 Design Proposal June 22, 2012**

***S.A.L.T.S***

Submersible Acoustically Linked Transmission System



Pat Cunniff -- Coleman Johnston -- James King

Steven Mackinnon -- Ryan Penny -- Rafal Premik

**Table of Contents**

[INTRODUCTION](#h.31abf665d7fd)

[Project Description and Applications](#h.390a9155d1d0)

[Similar Products](#h.aab1caa658f1)

[Challenges](#h.a96f7a32f8cc)

[SYSTEM OVERVIEW](#h.a1474f7e20ff)

[Project Purpose](#h.4282632106c1)

[Subsystems](#h.09d672e69c31)

[ROBOTIC DESIGN & BUILD](#h.bc0cbe78ee45)

[ROBOTIC CONTROL PROGRAMMING](#h.d5394b89c0df)

[VIDEO PROCESSING](#h.a51df06a53e4)

[MODEM DESIGN](#h.cb0d8d02dbce)

[Transmitter Design](#h.a22f85b20fbf)

[Receiver Design](#h.96b43a77f049)

[ACOUSTIC COMMUNICATION SYSTEM](#h.44d4b5bb2388)

[Differential Quadrature Phase Shift Keyed modulation](#h.9ecb4f719533)

[Orthogonal Frequency Division Multiplexing](#h.9d74b2232354)

[BASE STATION](#h.7ed6346b170a)

[USER INTERFACE](#h.e49d4ffc2c07)

[Overview](#h.25e1ce949123)

[Data Transmission](#h.ebc91f83bf0f)

[Video Decoding & Display](#h.03d861725171)

[Robot Control](#h.bed1bfc5f7dc)

[Data received from robot](#h.97e833d97c9e)

[Object detection/following](#h.e5ef8cec9bbb)

[POWER CONSIDERATIONS](#h.f4f660f33e21)

[BUDGET](#h.0e79a4207fce)

[SCHEDULE](#h.6c441182964f)

**Figures**

[Figure 1: Neptune SB-1](#h.2b3ede55d63a)

[Figure 2: OpenROV](#h.5b96dd8b6270)

[Figure 3: North Carolina Station University AUV](#h.d5c479b8778e)

[Figure 4: System Overview (Laptop, Base Station, X-ducers, Robot)](#h.fc0e4ee2bd64)

[Figure 5: First Iteration Mechanical Design](#h.0f54d5ef8d08)

[Figure 6: Motor Map (Front of Machine at Top Center](#h.41ff35cf0cdd)

[Figure 7: Buoyancy Equivalence](#h.fe6211cc2e08)

[Figure 8: PID Loop Block DIagram](#h.dfd4adac0ec8)

[Figure 9: Transmitter Hardware Block Diagram](#h.c8a955bea24f)

[Figure 10: Receiver Hardware Block Diagram](#h.19c1787e3f87)

[Figure 11: DQPSK Encoding/Timing Diagram](#h.13cd745b1a99)

[Figure 12: OFDM Visualization](#h.cab88586bc74)

[Figure 13: Sample GUI Application Layout Created in Visual Studio Forms Designer](#h.f501bbd85b94)

[Figure 14: Xbox 360 Controller mapped to control our robot's movement](#h.1814ee8a7914)

[Figure 15: Power Supply Distribution](#h.b722640fe799)

[Figure 16: Estimate Budget](#h.495a10b4d816)

# INTRODUCTION

## Project Description and Applications

We have decided to make an unmanned underwater vehicle (UUV) which will transmit video and receive out-of-water commands via an acoustic link. We chose to go with acoustic transmission because electromagnetic waves are easily dampened. We believe that acoustic communication will work more effectively than optical communication because of its ability to work at long ranges underwater. Current state of the art acoustic modems operate at lower data speeds than we would need to transmit video so we are going to have to design our own. To date, no one has successfully transmitted video underwater and our goal is to accomplish this task.

There are obvious benefits to receiving data in real time rather than having to wait for a UUV to surface in order to see video footage. All real time video UUVs today are tethered by cable, which can cause problems in the ocean environments and it is not a very clean system for the ever-increasing push for interconnected network communications. The Submersible Acoustically Linked Transmission System (SALTS) can extend the range of current tethered robotic designs while still communicating in real time, acting as a gateway to future underwater robot designs.

SALTS can be used in many underwater applications, such as forensic work in underwater crime scenes, sunken salvage recovery, wildlife surveillance for scientific research, and incognito surveillance of high-risk targets on surface. Normally, a team would have to wait for their UUV to surface before viewing any sort of video data being collected or a person would need to be present with the robot underwater. As of now, there is no way of providing feedback to the robot from a user out-of-water. This is a very important aspect of underwater exploration because of the lack of knowledge we have about the environment and the lack of vision we have of the robot performing its tasks. A user having semi-autonomous control of the robot will allow for a more controlled system without the user having to sit in front of the UI for the entire process.

## Similar Products

We considered several existing designs before deciding to make our own robot. We were considering the option of modifying the Neptune SB-1 RC submarine to perform acoustic communication. Unfortunately, the submarine is too small for the large sonar transceivers we will be using and the heavy modem that it would need to carry in order to serve its purpose.



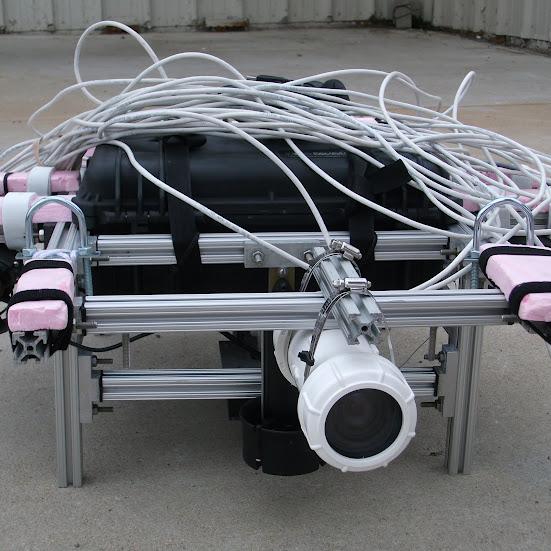
**Figure 1: Neptune SB-1**

Another design we considered was the OpenROV kit that is pictured below. This marine rover is engineered and balanced to maneuver standalone. Once again the vehicle is too small to house the acoustic communication hardware. Its dimensions are 30cmx20cmx15cm and cost is well over $800 dollars for the disassembled kit, which would surpass our budget.



**Figure 2: OpenROV**

We also considered North Carolina State University’s Robotics Team’s UAV (pictured below). This underwater robot had a simple mechanical design and wouldn’t be as hard as the previous projects to adapt our system. NCSU’s robot is not commercially available though.



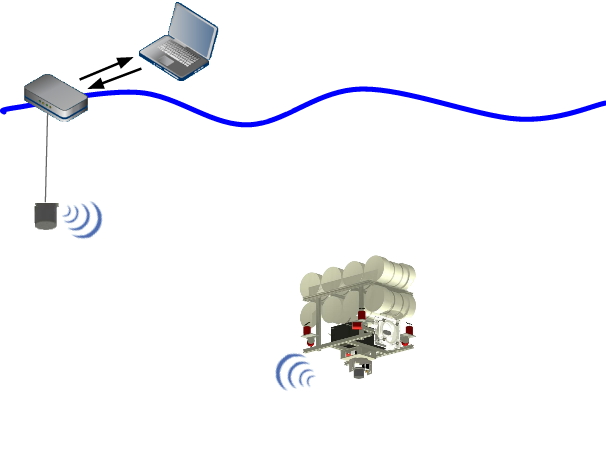
**Figure 3: North Carolina Station University AUV**

## Challenges

Since we will be sending data though an underwater channel, we will have some challenges that are to be abstracted from the end-user. One of these challenges will be multipathing. When transmitting acoustic signals underwater, the signals will tend to take multiple paths to the destination, causing there to be a period of time at the receiver where the same data will be received multiple times. Testing will be performed in a pool so in addition to this normal multipathing, we will also experience multipathing due to reflections off of walls and the water’s surface. This extreme case of multipathing will make the fading in our channel very difficult to overcome. Another challenge in our underwater channel is attenuation. The properties of water will automatically attenuate acoustic signals, causing our signal at the receiver to be less distinguishable and more affected by noise in and on either side of the channel. Not only will our medium of transmission affect our acoustic transmission signal, but it will cause our camera to have a slightly refracted view of the objects it is trying to see. Since we will have to robot actively look for objects, this will be another concern to address. All of these challenges with our channel will severely cut down our communication rate and though we will have to deal with the complications of our channel, a major challenge for us is to compress the video and modulate it in a way to keep our data rate as high as we can in order to see a decent image on the user side.

Designing a robot to autonomously stabilize itself and control its own movements is a bit different than creating an autonomous land robot. Our robot will need to move in 3 dimensions and keep buoyancy. Not only will we be attempting to do this but we are also going for full holonomicity, meaning we should not have to turn our robot in order to move in the desired direction.

# SYSTEM OVERVIEW



**Figure 4: System Overview (Laptop, Base Station, X-ducers, Robot)**

## Project Purpose

We will develop a modem capable of higher data rates than the current 5 kbps rate and fit this modem with a sonar transceiver to establish an underwater acoustic link. This link will be a directional transceiver and will be able to handle commands to the submersible and low quality video on the same link. This project is also a proof of concept regarding underwater acoustic video transmission, which can be further extended to become a robust system capable of high transmission rates and stability and confidence with the UUV controls.

### Subsystems

This project has the following subsystem groups:

\* Robotic Design & Build

\* Robotic Control Programming

\* Video Processing

\* Modem Design

\* Acoustic Communication System

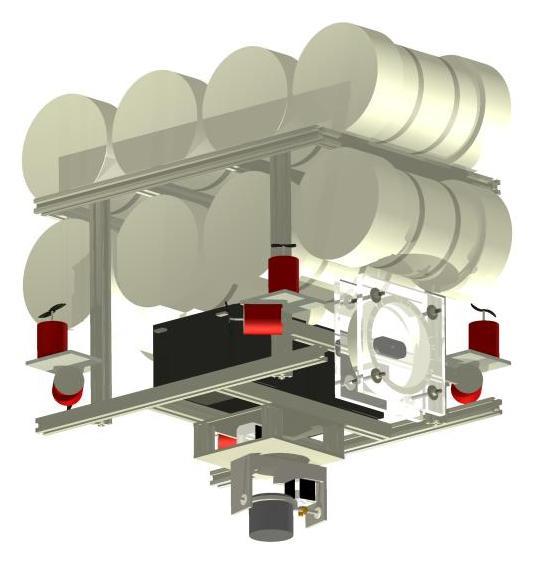
\* Base Station

\* User Interface

\* Power Solutions

# ROBOTIC DESIGN & BUILD

Robotic design is driven by three requirements: make the machine capable of underwater operation, make the machine controllable in all six dimensions, and maintain sonar transducer alignment. There are many sub-components to mechanical design and control. Mechanically, issues of chassis material, pressure vessel materials and sealing, center of buoyancy, and center of mass must be considered. Further, maintaining positive control of the machine in a six dimensional space (3 spatial coordinates, roll, pitch, and yaw) presents a challenging task given the nonlinear characteristics of the machine and environment. Our anticipated design is shown below.



**Figure 5: First Iteration Mechanical Design**

The choice of frame material choice was driven by the dimensional requirements of the machine. Power will be supplied by two lead acid batteries weighing approximately 18 kg each in air, which is in addition to the mass of the frame its self, pressure vessels, sonar transducer, and other electronics, resulting in a machine that will likely weigh in excess of 50 kg. Polyvinyl Chloride (PVC) pipe was initially considered as a frame material. This material was chosen against due to its low strength, which poses a problem for transporting the machine, its lack of rigidity, and tendency to shatter under shock loading conditions (e.g. a collision between the pool wall and the machine). A workable solution was found using a structural aluminum extrusion, colloquially known as “80-20”. It provides the strength and rigidity necessary to build a machine of this size at a cost only marginally higher than PVC pipe. Further, it provides easy mounting and adjustment of other components without drilling holes or other machining work.

Propulsion was the next step to making a workable design. We decided that a holonomic drive provided the best solution for our movement needs. This means that the machine can move in any direction without turning, and can rotate around its yaw, pitch, and roll axes with an effectively zero turning radius. This provides a much higher degree of control than the classic submarine control scheme, where forward motion must be maintained to turn the vessel. Implementing this type of drive requires 6 motors at minimum; however, our design uses 8 to provide redundancy and additional torque. Four of the motors are placed vertically to apply thrust in the ±Z direction, and are used to adjust depth (Z), pitch, and roll. The other four motors are mounted on the corners of the square chassis, and are placed at 45° such that their thrust vectors point along a line drawn from one side of the corner and the other. This configuration allows for movement in any direction in the X-Y plane, and to twist, or control the yaw, on the spot. The motors themselves will be taken from bilge pumps, as they are already water tolerant and provide sufficient torque when paired with a large hobby propeller. The motors are shown relative to the bottom most part of the frame in the image below. Motor control is discussed in a paragraph below.



**Figure 6: Motor Map (Front of Machine at Top Center**

Maintaining pressure vessel integrity presents another problem, which must be solved to ensure the safety of the electrical components in the machine. Schedule 80 PVC pipe was chosen as a suitable material for the pressure vessels. One end of the pipe can be satisfactorily sealed using an end cap and conventional PVC welding techniques. This solution is not workable at the other end; however, as the electronics must be accessible for troubleshooting and maintenance. For this end, aluminum end caps will be machined to contain two o-rings, which will be held in compression about the exterior of the PVC pipe. Although a single o-ring is capable of sealing against water pressures as high as 200,000 psi (orders of magnitude larger than the ~7 psi that will be encountered in the pool), a two o-ring system was decided upon due to the dire consequences of seal failure. As an additional precaution, pressure vessels which contain electronics may be filled with mineral oil, making the vessel effectively incompressible. Bronze cable glands rated to 300 feet have been found to allow the passage of wires into and out of the pressure vessels, which will be placed on the aluminum end cap to avoid compromising the strength of the PVC pipe.

As we intend to use a directional transducer on our machine, a system to maintain alignment between the transducer and base station was mandatory. Mechanically, this system uses stepper motors to rotate the transducer continuously though 360° of rotation and 180° of tilt on a two dimensional gimbal. Initially servos were chosen for this application, but they were decided against due to the difficulty in waterproofing them, their lack of torque, lack of continuous rotation, and high degree of mechanical and electrical noise. The stepper motor solution solves these problems at the expense of a marginally more complicated initialization sequence and electrical connections. The stepper motors will be geared on a self locking reduction worm gear, eliminating the possibility of back driving the motors from torque applied to the transducer. The gearing will also reduce the 1.8° steps of the motor to a much more precise 0.09° or lower if necessary.

The final set of mechanical considerations involve buoyancy. On the design level, we require that the center of mass of the machine be directly below the center of buoyancy to prevent any tendency to roll or pitch the machine. Further, active control of the machine in the Z dimension is required. We decided against passive buoyancy control due to the complexity of controlling air bladders and the other hardware necessary for this solution. Instead, we aim to make the machine within a few pounds of neutrally buoyant by designing the machine to be only slightly negatively buoyant and adding sections of closed cell foam to add buoyancy. So long as the resulting buoyant force is below the maximum thrust of the four motors oriented vertically, the depth may then be actively controlled. Once all other systems have been designed and their weight and displacement determined, calculations will be done on the volume of foam necessary to bring the machine to neutral buoyancy. As a beginning step, the buoyancy of the battery was compared to the buoyancy of the pressure vessels; three pressure vessels ( at +30 N each) are needed to float a single lead acid battery ( -85 N).

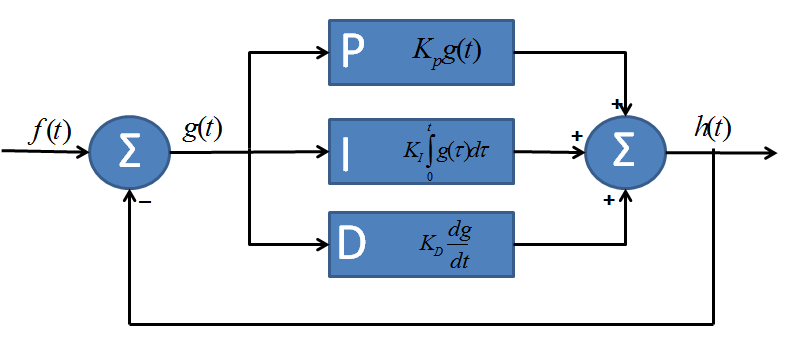


**Figure 7: Buoyancy Equivalence**

## ROBOTIC CONTROL PROGRAMMING

Driving the hardware listed above will be accomplished using a PIC-16. Inputs to the PIC will be object coordinates coming from the video processor and commands from the user. Stabilization will be done with a motion processing unit and we will output PWM signals to H-bridges between the PIC and motors. The motion processing unit will be continuously polled by the PIC for acceleration information in all 6 directions. On board the PIC, this information will be used to determine the actual orientation of the machine. We intend to control pitch, roll, yaw, and depth with individual PID control loops. Motion in the X-Y plane will run open loop. Initially, user input will be a desired velocity vector in the X-Y plane, a yaw angle, and a depth, with roll and pitch having a hard coded set point of 0°, unless user control of these variables is deemed necessary. Motor outputs are determined by taking the dot product between the desired velocity vector and the trust vector of each motor. Then this value is adjusted as determined by the PID loop to correct for the angular motion of the machine, and then scaled to a value for a corresponding PWM output on the PIC. The sonar transducer will be autonomously driven using the derived positional information to maintain sight of the receiver.

PID (proportional, integral, derivative) loops determine the value for a control variable by comparing a process variable to a set point. As an example, the actual depth of the machine would be a process variable, the set point the desired depth, and the control value would be the motor outputs to adjust the depth. The first step of PID loop operation is determination of the error between the process variable and set point. Once this is known, the proportional term is taken to be this error weighted by a constant. Thus the farther the machine is from the target depth, the faster the motors will run to get there. The next correction is the integral term, which provides a correction factor by multiplying the time integral of the error by a constant. This helps to correct a slight but persistent deviation from the set point which would not otherwise be large enough to overcome the dead band. For depth,this might be caused by briefly passing through an area of light downdraft, and being slightly below the target depth after. Lastly, the derivative term is computed by weighting the time derivative of the error by a constant. This term measures how quickly the process variable is changing, and it is weighted such that it prevents the proportional factor from overshooting the set point by throttling the motor output as the set point is approached. In our example, it slows the machine down to prevent it from overshooting the target depth as it is approached. Once the target depth is obtained, the derivative term also more quickly ramps up the correction for faster deviations from the set point. The sum of all the terms is taken to determine the output of the loop. Although this scheme does not necessarily provide optimal control of the system, it will be more than sufficient for our purposes. Should the nonlinear handling conditions of our machine be worse than anticipated, gain scheduling, or changing the weighting constants mentioned above for different operating regimes, can be implemented.



**Figure 8: PID Loop Block DIagram**

The object following side of the control scheme will have an algorithm as follows:

A desired trajectory will be continuously sent to the PIC but will only be utilized once a command from the user has been sent. User commands will include positional changes and locations of individual objects to be followed from the video feedback line. Should the directional transducer link get broken or faded, this will be a first priority to fix before any other actions can be completed. The second priority of the robot control will be its stabilization; once this is considered satisfactory, user commands will determine where to go and/or what task to complete. The movement from point A to point B and the stabilization along the way will be controlled using the techniques described in the above paragraphs.

# VIDEO PROCESSING

The video processing unit we have decided to go with is the BeagleBone. It has optimal processing speeds for our tasks to be completed; we have confirmed this with benchmark fast fourier transform (FFT) testing. It has an embedded operating system for ease of use with professional video compression, object detection, and FFT libraries. Raw data will come in though a USB line in order to keep as much of the video data as possible for the video processing. The input video will go to the object detection code and the compression code concurrently and independantly. This process will be most effectively optimized for speed using operating system’s scheduling algorithms and thread usage.

OpenCV will be used to detect objects in the water. Use of the OpenCV libraries allow for fast and easy manipulation of images. A combination of edge detection and color differentiation will allow for relatively accurate object detection underwater. Normally, refraction would be a problem with images underwater but our camera has a lens angle that should cut down on the refraction of the image coming into the processor. Since our video processor will be running an embedded operating system, we will be able to port our testbed OpenCV code from a desktop computer to the BeagleBoard with only minor setup changes.

The raw video will also be coming straight into our image compression code. We will be using FFmpeg with either H.264 or MPEG-4 (depending on performance) to encode our video to a rate of 64 kbps. We will be using spatial image compression and temporal motion compensation to reduce the number if bits being sent over the channel. Because of our high bit rate error, we will have to implement an error checking scheme otherwise the spatial image compression and temporal motion compensation will cause the video at the destination to be distorted because of the propagation of errors effect; we must do this without introducing too much lag into the video transmission. The video will also most likely have to be compressed to gray-scale and have a reduced frame rate of ~1 frame/sec along with a decrease in resolution in order to keep our compressed video equal to or below the channel bottleneck we will have. We will also need to compress this so it may be decoded and played real-time. This will require us to use a streaming protocol like RTP/RTCP with modifications to account for our error prone channel. We could do this either by sending a constant stream of data or sending packets of data. For error checking, we might go with added parity bits, start/end bits, checksums, repetition codes or ACK packets if we are able to reserve certain frequencies for data coming back from the base station.

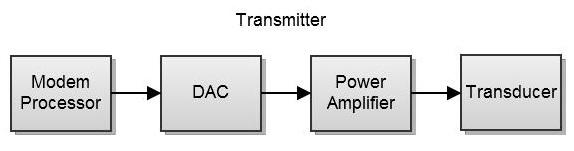
After the video gets compressed, it will be sent through our modulation code. This code is currently being developed in MATLAB and is based off of Rameez Ahmed's in-air acoustic transmission system. A big portion of the processing power needed for this code will be the FFT and IFFT's. This will be the code associated with the modem in our system.

# MODEM DESIGN

Our acoustic modem will be designed to transmit the video from the camera as well as basic locational data to the base station. In addition it will receive basic controls from the user via the base station and relay them to the PIC mictrocontroller. To reliably transmit the video, we plan to achieve a data rate of 64 kbps. In order to accomplish this data rate the hardware must reliably transmit the generated signal to the transducer with little distortion and sufficiently amplify that signal to overcome the adverse channel conditions. Any received signals must be conditioned to filter out any unwanted signals, then amplified to appropriate levels across the spectrum.

Design of the modem consists of two main hardware blocks: the transmitter and the receiver. The transmitter consists of the processor, digital to analog converter (DAC), and a power amplifier (PA). The receiver section consists of a lower noise amplifier (LNA), an automated gain control amplifier (AGC), an analog to digital converter (ADC), and the processor. A piezoelectric transducer will be used to transfer the data across the underwater channel.

## Transmitter Design



**Figure 9: Transmitter Hardware Block Diagram**

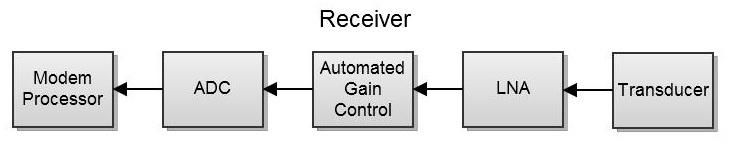
The transmitter portion of the modem design will include four major blocks. The major components are the common modem processor, digital-to-analog converter, a power amplifier, and the common transducer. The modem processor will be a BeagleBone Development board utilizing OpenMAX code libraries along with the fftw-arm NEON SIMD enabled library to efficiently run the IFFT and FFT functions required. Benchmark testing has been conducted to verify that the ARM Cortex-A8 processor operating at 720MHz will be sufficient for calculating the IFFT and FFT results within our real-time constraints.

The digital-to-analog converter (DAC) block will use a DAC with a sampling rate of 300k samples per seconds. This will satisfy the Nyquist-Shannon sampling theorem as the max operating frequency will be no greater than 70kHz. This DAC will connect to the BeagleBone utilizing the Parallel Port to allow for the least latency between processor transmission and transducer transmission. A dedicated DMA channel will allow for signal tranmission through the DAC while the next frame is being generated by the modem processor. Utilizing this scheme, real-time constraints will be easily met.

The next stage of the transmitter will be a Class D power amplifier. Operational capability has been verified via reference to the Woods Hole Oceanographic Institute’s uModem design which utilizes a class D power amplifier very successfully. Given operation in the ultra-sonic band, strictly linear amplifiers are not a complete requirement since most industry class D power amplifiers contribute only 0.01% distortion to the signal being amplified. The error introduced by a class D amplifier will be outweighed by its high efficiency in the 90%+ range. A class D amplifier will also greatly assist with the need to run an embedded system with high power considerations. See the power considerations section for more information regarding the power supply operation.

The final transmitter stage is the common transducer that will be used for both transmitting and receiving of data. This transducer is currently being specified but will operate in the 50kHz-70kHz region with a conical beam width of 20 degrees on the robot and 30 degrees at the base station. The conical beam pattern has been chosen to overcome the multipathing challenge that will ultimately reduce total data throughput.

## Receiver Design



**Figure 10: Receiver Hardware Block Diagram**

The receiver hardware consists of two blocks, the signal conditioning and equalizing, and the signal processing block. The signal conditioning and equalizing will be accomplished using a low noise amplifier (LNA) and an automated gain control (AGC) amplifier. The LNA will be amplifying the lower powered signals received without adding additional noise to the system itself. This amplified signal will then be passed to an AGC amplifier. The AGC amplifier is responsible for monitoring the average power of the incoming signal. Naturally the signal will be attenuated at certain places. The function of the AGC is to make sure the less attenuated sections are not amplified too much and throw off the digitizing of the signal in the ADC.

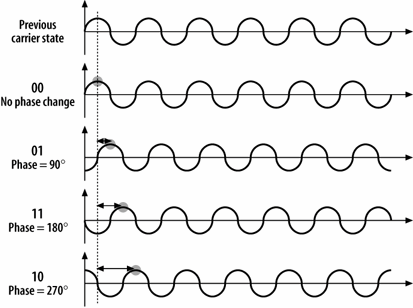
The software signal processing block consists of the analog to digital converter and the same modem processor as is used in the transmitter block. The ADC is responsible for sampling the analog signal and digitizing the signal for input into the processor. The challenge with this is that the incoming signal often has noise-like characteristics, meaning there is a high peak to average power ratio of the signal. Most of the data in the signal is stored in a very dense area, but outlier data points can throw off the sampling of signal. The processor is responsible for demodulating the signal and reproducing the original bitstream. The processor will perform fast Fourier transforms on the incoming signal separating the transmitted frequencies from each other. Each frequency is the demapped with a DQPSK symbol map and the transmitted data is reproduced.

# ACOUSTIC COMMUNICATION SYSTEM

We will be using Differential Quadrature Phase Shift Keyed (DQPSK) as our modulation scheme and Orthogonal Frequency Division Multiplexing (OFDM) to utilize our bandwidth more effectively. Both of these methods provide solutions to the challenges specific to our project, such as multipathing, short bandwidth, and channel distortion.

### Differential Quadrature Phase Shift Keyed modulation

DQPSK offered us several advantages over other modulation techniques, which we found to best fit our project. First, DQPSK is not amplitude modulated, which benefits us greatly, as the underwater channel can greatly attenuate signals at different levels across the frequency spectrum. Not having to have multiply transmission power levels helped us save in power, as that is already a large factor in our system: more power= more heavy batteries. Frequency modulation would have required we use two frequencies per OFDM channel, thus effectively cutting our bandwidth in half. Finally DQPSK over regular QPSK provides immunity from phase shifts in the channel. Encoding data symbols with respect to phase can be very difficult in water, as the phase shift across the channel can be different for every subcarrier frequency. As the receiver can not reliably determine how far each carrier has shifted from the phase it was encoded to, a different reference must be used that is known by the receiver. In DQPSK, data is stored between frequencies, the difference in the phase of two carriers encodes the data. As the carriers are fairly close to each other, it can be assumed that whatever phase shift one has incurred, the other has as well.

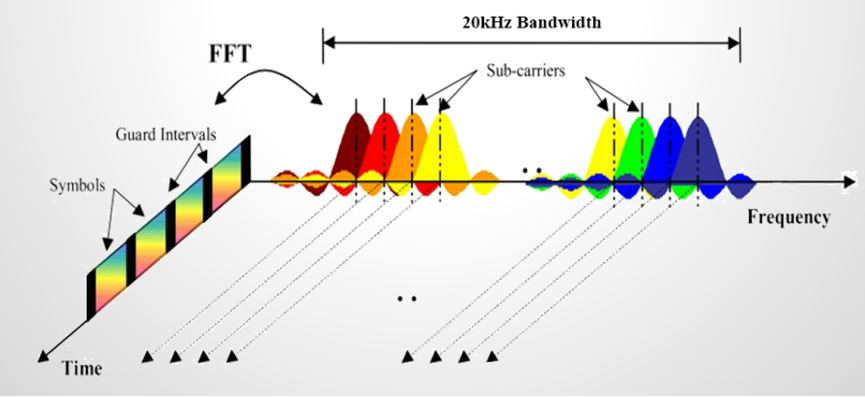


**Figure 11: DQPSK Encoding/Timing Diagram**

### Orthogonal Frequency Division Multiplexing

OFDM solves many of the problems we face transmitting in the underwater channel. Splitting the data amongst many lower data rate channels spaced by a calculated frequency will allow us to greatly reduce the inter-symbol interference from our system. Each transmitted symbol is followed by a guard interval, which allows any multipathing to resolve. This multipathing causes inter-symbol interference, which is successive symbols bleeding into each other’s time slots. As all of the carriers are orthogonal over the symbol period, there will be minimal inter-carrier interference as well. Splitting the data amongst many carriers across the bandwidth also minimizes the amount of data lost due to channel fading at certain frequencies. Interleaving can also mitigate the amount of data lost due to frequency selective channel fading. OFDM is also very advantageous for the receiver as well. As the symbol period is much larger than the periods of any of the carriers and a cyclic prefix is appended to the signal during the guard interval, the sampling window and can be moved around to accommodate the delays of all the carriers.

OFDM, of course, presents several of its own challenges that must be overcome. An OFDM signal has a very high peak to average power ratio, which means, amplifiers must be able to amplify much larger than the average transmitted power of the signal. An amplifier with higher voltage rails must be used in the design. This can also tax the dynamic range of your DAC or ADC. In addition tp high PAPR, OFDM also is very sensitive to doppler shift. A small shift in the frequency of a carrier can lead to loss of orthogonality, which may cause inter-carrier interference or even carriers to shift entirely into another carrier’s frequency slot. This can be resolved through the use of preambles to transmitted packets, which can notify the receiver of the transmitted frequency.



**Figure 12: OFDM Visualization**

# BASE STATION

The base station will be the complimenting side of our acoustic link and will consist of an exact replica of the modem and transducer we will be using on the robot minus the servos needed to position the directional transducer. The other side of the base station modem will initially be connected directly to the computer out of the water.

Placement of the base station is our largest concern in its design. As the transducer will be fixed in position, we will not be able to actuate it to achieve maximum signal strength. Thus we must have a broader angle on the base station transducer to cover the majority of our operating environment. After measuring the dimensions of the pool we are to be testing in, we have found that about a 30° transducer cone width will be sufficient if we place the base station transducer at the bottom of the pool about 12m away from our working environment. This will present us with a maximum working distance of about 30m, and near complete coverage of the working environment.

Overall, the base station utilizes our modular modem approach to be created. Tuning will be necessary to have cabling running out into the water at a longer distance (~2m) but there should be minimal difference between it and our robot modem.

# USER INTERFACE

The system's graphical user interface (GUI) application will serve several purposes:

· Display the camera's video feed

· Allow the user to manually control the robot's movement

· Display debugging data received at the base station (and save it)

· Display the power levels of the system's batteries

· Display the robot's X, Y, and Z coordinates relative to the base station

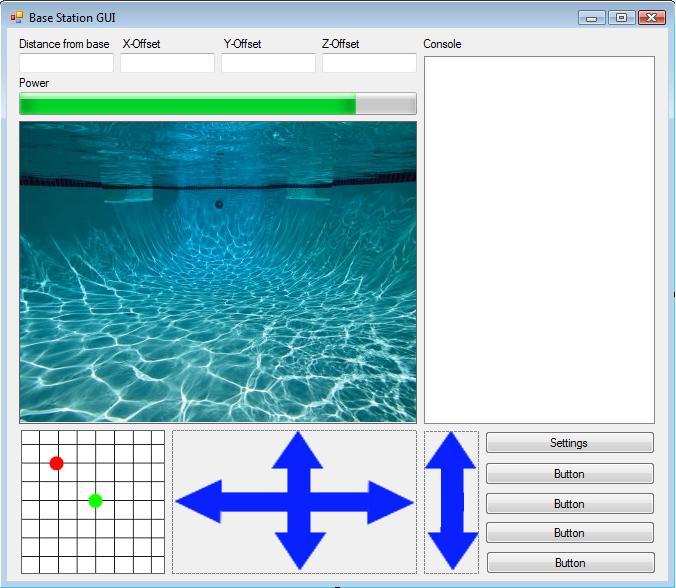
· Allow the user to specify which object(s) the robot should follow

· Display a map of the robot, the base station, and any detected objects

· Send commands to the robot

## Overview

The GUI application will be written in C#, which utilizes Microsoft's .NET Framework, and it will run on a Windows machine. The advantage of using C# is that it allows us to use the Windows Forms API, which provides access to native Microsoft Windows user interface elements. Additionally, Microsoft Visual Studio contains a tool called Windows Forms Designer, which allows us to easily design a GUI by dragging and dropping UI elements onto a form. The behavior of the elements (i.e. reaction to a button click) can then be easily defined in code.



**Figure 13: Sample GUI Application Layout Created in Visual Studio Forms Designer**

## Data Transmission

In order to send and receive data between the base station's modem and the C# application, we will packetize the data and transmit it over an Ethernet line. For our video data, we will most likely use UDP as our application-level protocol because although it does not guarantee packet delivery, it requires less overhead than TCP, hence a shorter capture-to-display lag time. The arrival of our control and debugging data is much more crucial, therefore we will send it to and from our application in TCP packets, which guarantee packet delivery. This will be easily implemented in our application because C# contains a Socket class, which makes it easy to access the network and to send and receive packets.

## Video Decoding & Display

To display the camera's video feed, we will have to use Microsoft's DirectShow API; an architecture for streaming media on Windows. This will be used to both decode and render the MPEG video stream. DirectShow is written in C++, so we will not be able to use it directly in C#. Fortunately, there are a large number of reliable code samples, tutorials, and documentations out there which show how to use DirectShow and properly interface it with C#, so this should not be much of an issue for us.

## Robot Control

To manually control the robot, we have several options to consider:

· Clicking arrow buttons in the GUI (blue arrows in the figure above)

· Pressing keys on the base station PC's keyboard (i.e. up, down, left, right)

· Using an external controller

The method we choose to implement depends on how quickly and frequently we can deliver data from the base station's modem to the robot's modem. Since the bulk of our communication will be sending compressed video data from the robot's modem to the base station, the time slot in which we can send data from the base station to the robot will be limited. This being the case, it might be best to control the robot using the arrow buttons in the GUI. If we took this approach, we would send commands that prompted the robot to rotate a fixed number of degrees, or moved up, down, left, or right a fixed distance. Furthermore, we could allow the user to specify either coordinates for the robot to travel to, or the angle to rotate and the distance to move in each direction.

If we are able to send data from the base station to the robot frequently (more than 10 times per second), then it could be possible to achieve finer control. The easiest way to do this would be to use an external controller, such as an Xbox 360 controller in the figure below; the x, y, roll, pitch, yaw, and height can easily be controlled using this controller. Microsoft has a free XInput API which provides access to an Xbox 360 controller connected to a PC via USB. This will make it easy to receive input from the Xbox 360 controller and format robot movement commands appropriately. We could also allow the user to control the robot by assigning keys on the base station PC’s keyboard corresponding to these six degrees of movement.



**Figure 14: Xbox 360 Controller mapped to control our robot's movement**

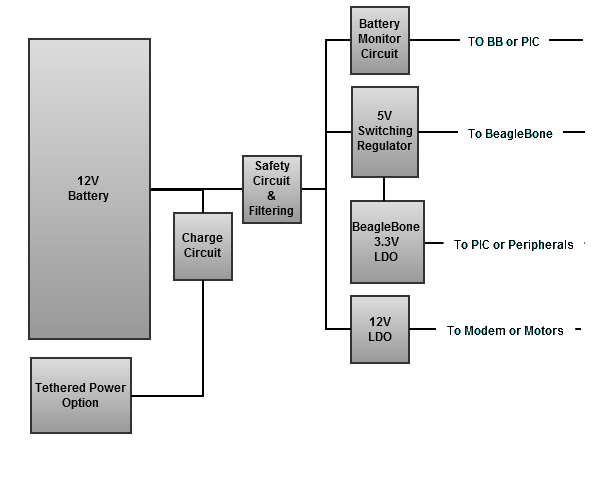
### Data received from robot

The robot’s modem will periodically send non-video data to the base station’s modem. This will include debugging data from the Beagle Board and/or PIC, robot coordinates, battery power levels, and any other data we find important. The base station’s modem will packetize this data in TCP and send it over Ethernet to a different port than video data is sent to. Our UI application will listen on this port and when data arrives it will format it appropriately, update the corresponding UI elements with the new data, and save debugging and coordinate data locally in text files.

### Object detection/following

If the user clicks within the video display control of the UI application, the application will send a TCP packet to the base station’s modem containing the coordinates of the mouse click and a command to follow the object at these coordinates. The base station will then relay this message to the robot during its next transmission. This message will be received by the robot and sent to the Beagle Bone, which will check if there is an object located at these coordinates. If the Beagle Bone’s object detection algorithms do detect an object, it will command the robot to follow this object. If there is no object, it will either do nothing or send a message back to the base station during the next non-video transmission indicating that there was no object found at the specified coordinates.

# POWER CONSIDERATIONS



**Figure 15: Power Supply Distribution**

The robot, the modem, and the base station will use the same universal power supply source. The power supply will generate 3.3V@500mA, 5V@10A, and 12V@30A. The actual power supply will include a PMIC controller for charging of the battery and battery level monitoring by the robots main control systems. A tethered external connection will also be available with its leads being diode ORed to the main power bus within the PMIC circuitry.

Power to each system will be accomplished by providing a 12V Lead-Acid Car Battery. For the base station, a 12V DC Wall Charger will be used to provide the power necessary for transmission.

Before the 12V value is brought to any DC-to-DC converters, safety circuitry and filtering will be done. This section will be mostly safety circuitry since much of the filtering will be done on the voltage regulators. There will be large capacitive bulk and high frequency filtering located here and fuses as well. Fuses will be rated to 15A so 12V zones of power may be created to accommodate the high power needs of the motors and the modem transmitter components.

Once safety and filtering circuitry are addressed, the battery voltage will be supplied to the 5V Switching Regulator and 12V LDO. These regulators will supply the correct voltage and adequate current to the BeagleBone Board and the robot motors. The BeagleBone’s internal LDOs will convert some of the power supllied by the 5V regulator to 3.3V to be supplied to the PIC microcontroller and MPU since these components will never exceed 500mA of current needs.

The 12V LDO will be used to create a clean 12V power source to be converted to +/-24V rails for the modem and 12V power for the motor control. Each of the (8) eight robot motors will require 1A of peak current pull for operations. The 12V LDO will be sized for 20A of supply since the power amplifer that will be located on the modem may require up to 200W of power. This will mean nearly 20A of current pull on the 12V LDO when supplying a +24 and -24 voltage rail.

Due to these high power demands, (2) Two Lead Acid Batteries will be used for operation. These will provide a total of 60Ah of power to the system. With peak power requirements of 30A, these will allow for 0.5C battery drain and roughly 4hrs of system endurance since peak power pull will occur only a fraction of the operational time.

# BUDGET

|  |  |  |  |
| --- | --- | --- | --- |
| **Robot** |  |  |  |
| Frame | Machined Aluminum Frame |  | $60.00 |
| PVC Tubing | 6"x4' PVC Electronics Housing |  | $50.00 |
| Plexiglass Box | Clear Box for Camera |  | $50.00 |
| Misc Hardware | Nuts, Bolts, Etc |  | $100.00 |
| Stepper Motors | 2x Transducer Steppers |  | $30.00 |
| Motors | 8x 12V DC Bilge Pump Motors |  | $128.00 |
| Batteries | (2) 12V Lead Acid |  | $120.00 |
| Hbridges | 8x for motor control |  | $20.00 |
|  |  | **Total** | **$558.00** |
| **Robot Electronics** |  |  |  |
| MPU | Accelerometer/Gyroscope |  | $15.00 |
| Video Processor | BeagleBone |  | $79.00 |
| Robot Plrocessor | Pic32 |  | $35.00 |
| Base Station Processor | BeagleBone |  | $79.00 |
| Camera | Logitech C210 |  | $23.00 |
| Circuit Boards | Estimate 1 8"x8" 4 Layer PCBs |  | $70.00 |
| Misc Components | Res, Caps, FETs, etc |  | $50.00 |
|  |  | **Total** | **$351.00** |
| **Modem** |  |  |  |
| Transducers | SONAR Transducer |  | $0.00 |
| Amplifiers | Power Amplifier/Gain Controller |  | $120.00 |
| UP/Down Converters | Step up and step down |  | $100.00 |
| Signal Processing | Signal modulation/ demodulation |  | $79.00 |
| Circuit Boards | Estimate 3 4"x8" 2 Layer PCBs |  | $100.00 |
| Misc Components | Res, Caps, FETs, etc |  | $50.00 |
|  |  | Total for 1 | $449.00 |
|  |  | **Require 2** | **$898.00** |
| **Other costs** |  |  |  |
| Lifeguards | $36/hr (est 6hrs) |  | $0.00 |
| Test Bench | Small Pool for qualification |  | $90.00 |
| Machining | $15/hr (est 5hrs) |  | $75.00 |
| Tools |  |  | $0.00 |
|  |  | **Total** | **$165.00** |
| **Totals** |  |  |  |
| Robot Mechanical Total | $558.00 |  |  |
| Robot Electronics Total | $351.00 |  |  |
| Modem Total | $898.00 |  |  |
| Other Total | $165.00 |  |  |
| **Total Project Cost** | **$1,972.00** |  |  |

**Figure 16: Estimate Budget**

# SCHEDULE