October 27, 2003

Dr Andrew Rawicz
School of Engineering Science
Simon Fraser University
Burnaby, British Columbia
V5A 1S6

Re: ENSC 340 Design Specification for a Beam Pattern Measurement System

Dear Dr. Rawicz:

The attached document, *Design Specification for a Beam Pattern Measurement System,* describes functional details for the SonarWorks ENSC 340 group project.

We are in the process of designing and building a transducer beam pattern measurement system, primarily for use by researchers in the Underwater Research Lab here at SFU. The purpose of this document is to provide technical details of the system and sub-component design, including justification for design choices, where applicable.

This document describes the design of the various software and hardware components, including testing and integration considerations. In addition, an overview of the complete system architecture is provided. Where possible, we have chosen to use existing hardware in an effort to reduce costs, and to recycle used components.

If you have any questions about our design specification, please feel free to contact me by phone at (604) 420 3149 or by email at sonar-group@sfu.ca.

Sincerely,

Roβ Huxtable

Rob Huxtable
CEO
Sonar Works

Enclosure: *Design Specification for a Beam Pattern Measurement System*

Design Specification:

Multichannel Beam Pattern Measurement System

Project Team: Rob Huxtable
Tim Warner
Annie (Wan Chin)Wu
Robin Prest

Contact Person: Robin Prest
sonar-group@sfu.ca

Submitted to: Dr. Andrew Rawicz- Ensc 340
Steve Whitmore – Ensc 305
School of Engineering Science
Simon Fraser University

Issued Date: October 27, 2003
Revision: 1.1

Executive Summary

A key component of a sonar system is the transducer array, which is responsible for receiving acoustic waves and converting them to an electrical signal. Accurate sonar measurements depend on characterizing imperfections and cross-talk in the transducer array and understanding the problems they pose. A transducer's beam pattern describes its response with respect to the angle of arrival of a signal, and must be characterized before reliable data can be measured. Sonar Works is developing a system for testing multi channel sonar transducers in partnership with the Simon Fraser University Underwater Research Lab. Our goal is to produce an accurate beam pattern measurement system (BPMS) that is useable with minimal technical training. It is our intention to have a working proof-of-concept in operation by December 2003.

We have developed design specifications that provide a low-level description of the BPMS environment. The basic design requirement of the BPMS system is to generate acoustic waves from a sonar transducer which are then received by the transducer array under-test. The angle of incidence between the signal and receiver must be varied and the received signal recorded for each angle of incidence. A user interface provides control over the transmitted signal and displays the amplitude of the received signal versus the angle of arrival. Several options are available for displaying and interpreting the received signal.

To meet the requirements described above, the BPMS consists of several distinct components:

1. Matlab-based control and analysis software.
2. Sonar transmitter.
3. Sonar receiver and amplifier.
4. Motor and associated control hardware.
5. Computer with analog-to-digital and digital-to-analog converters.
6. Power supplies.

The components of the BPMS have been chosen to be robust and tolerant of a water environment, while at the same time providing full functionality and performance. Likewise, the user interface has been designed to be both intuitive and capable, so that users of all skill levels will find value in its operation. This design specification provides a detailed account of how we have achieved this combination of power with ease of use, examining each component of the BPMS and its operation.

Table of Contents

List of Tables

List of Figures

Acronyms

A/D Analog to Digital
BPMS Beam Pattern Measurement System
D/A Digital to Analog
DOF Degrees-Of-Freedom
GUI Graphical User Interface
PTFE Polytetrafluoroethylene (Teflon)
URL Underwater Research Laboratory
Tx Transmitter
Rx Receiver

1 Introduction

Accurate sonar measurements depend on characterizing imperfections and cross-talk in transducer arrays. Sonar Works is developing a system for testing multi channel sonar transducers in partnership with the Simon Fraser University Underwater Research Lab. This document provides technical details of the system and sub-component design, including justification for design choices. The design of the various software and hardware components is described and an overview of the complete system architecture is presented.

2 System Architecture

Figure 1 shows the block diagram of the system architecture, organized in terms of the software, electrical and physical layers. The beam pattern measurement system (BPMS) is built upon existing subsystems. This document describes the design of the principal components that were required in addition to these existing subsystems. In particular, the design of application software, motor control and mechanical hardware are discussed.

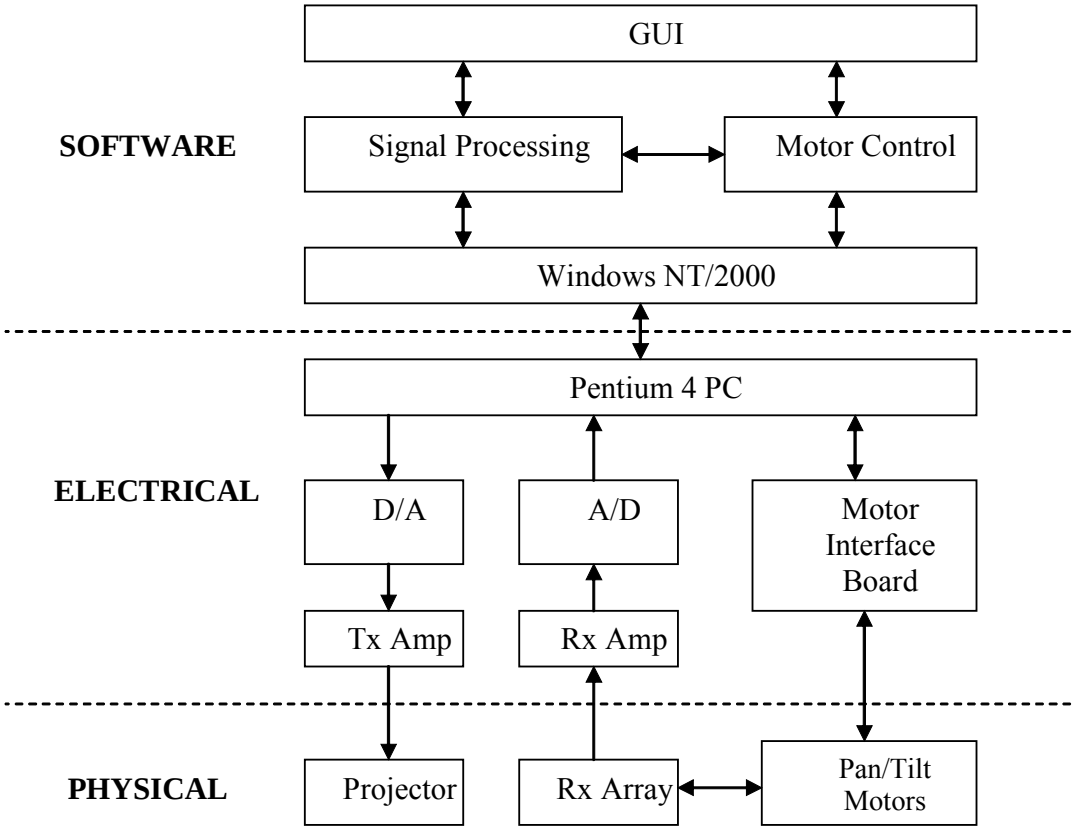


Figure 1: Block Diagram of Sonar System

3 High-level Software Design

3.1 Overview

This section describes the design of the software that controls the scanning sequence and beam pattern display, including both the user interface software and underlying functionality.

3.1.1 Architecture

The high-level software launches from the MatLab 6.5 command line and features separate scan control and display modules. A schematic representation is shown in Figure 2.

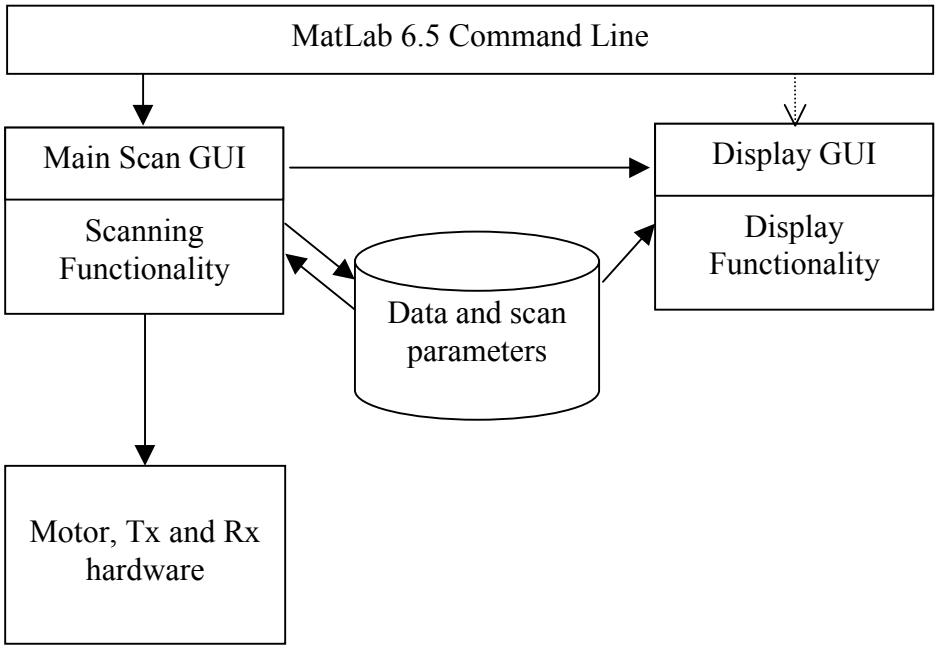


Figure 2: Schematic diagram of high-level software

Data is shared between the scan application and the display application through shared files.

3.1.2 Design Philosophy

The system architecture supports a modular development approach. Aside from shared files, the scan control application and beam pattern display applications are loosely coupled to allow for independent development and ease of modification. Each application can be called separately from the MatLab command line and does not share any variables with the other application.

While the system architecture supports separate development of the two applications, the GUI's for each have been designed to have a similar look and feel, and the display application can be invoked seamlessly from the scan application immediately after completing a scan so that the user does not need to know anything about the file sharing.

The software employs a functional design approach, as Matlab does not lend itself to object-oriented design.

3.1.3 Matlab Development Environment

Matlab was chosen as a platform for running the high-level software due to its power, ease of use and familiarity to anticipated users. Advanced signal processing capabilities are available, along with powerful high- and low-level graphics functions. In addition, Matlab features GUIDE, a development environment that enables the creation of GUIs to interface with Matlab programming scripts known as m-files.

Using GUIDE, a GUI is created by drag-and-drop placement of objects (buttons, list boxes, axes, etc.) onto a layout template. GUIDE automatically saves the design in a Matlab figure file and generates and m-file with functions associated with the objects in the figure. By editing the auto-generated m-file functions, it is possible to interface the objects with user-defined functions to implement the required functionality.

3.2 Main Application

3.2.1 User Interface

A screen shot of the main application user interface is shown in Figure 3. The main application is still under development, but the general layout and organization of the application is complete. The application has been designed so that the system is operated from one interface screen, thus making the user interface easier to navigate, particularly for new users.

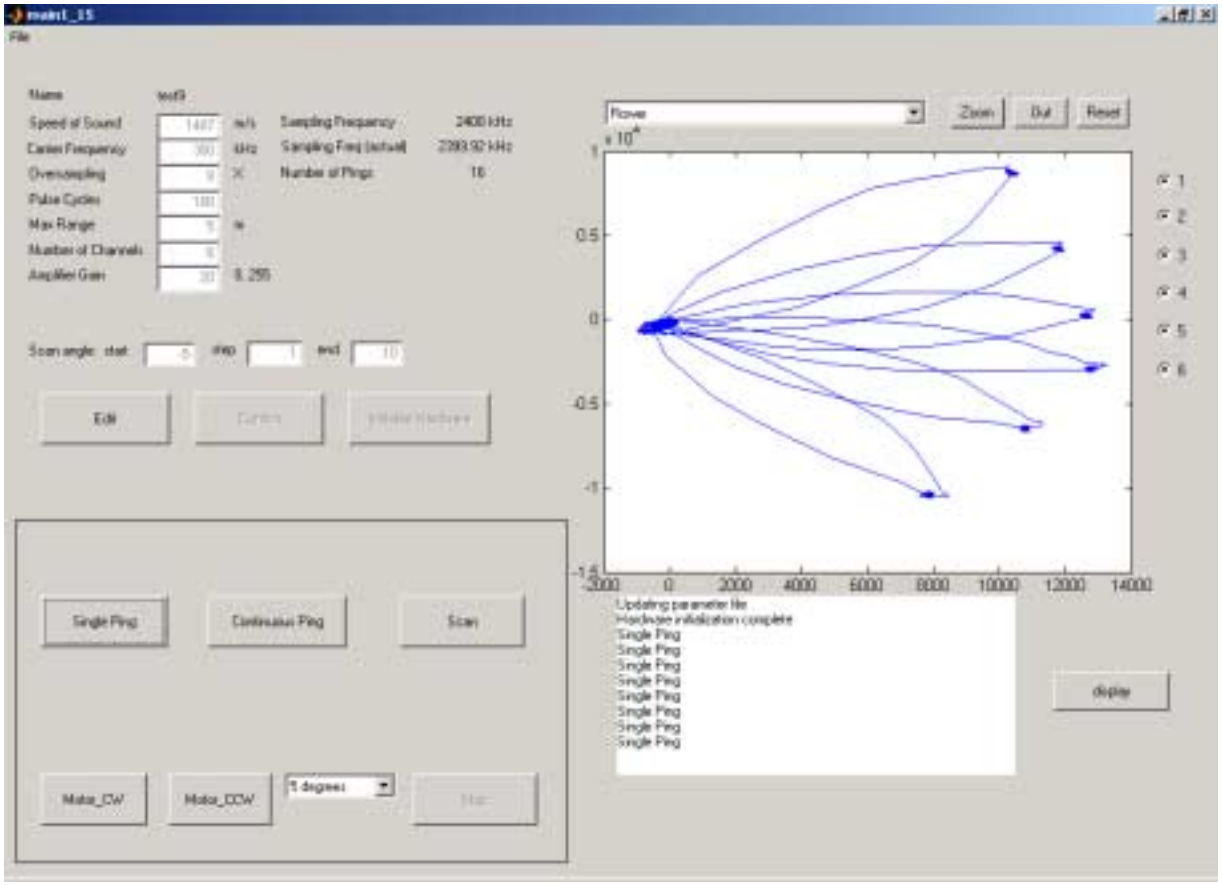


Figure 3: Screen Shot of the Main Application User Interface

The user interface has a standard "File" pull-down menu on the top left corner of the screen. From this menu, the user can choose to load parameters from a previously saved scan, or to start a new scan.

The upper left quadrant of the interface displays the loaded parameters (default parameters in the case of a new scan), and allows the user to *edit* and *confirm* them. Once the parameters have been confirmed, the hardware (A/D, D/A and amplifier) is setup via the *initialize hardware* button. The user interface prevents incorrect actions by enabling/disabling (i.e., "graying out") buttons where appropriate.

The lower left quadrant of the interface contains controls for manually moving the motor, aligning the transducer, and running a scan.

The lower right quadrant contains a scrolling dialog box that provides the most recent feedback on the most recent events. Feedback includes completion of tasks (e.g., hardware initialization) and errors that may have occurred.

The upper right quadrant of the interface contains the plotting area. Above the plotting area are plot selection (e.g., flower, envelope or oscilloscope plots) and zoom controls. To the right of the plot are radio buttons for selecting/de-selecting separate data channels corresponding to individual transducer elements.

Once a scan has been completed the display application is launched via the *display* button on the lower right of the screen.

3.2.2 Functionality

The primary role of the main application is to record beam pattern measurements and store them in file. The main application allows different parameters to be set (e.g., pulse length, pulse frequency, sampling rate), and provides initial feedback on the resulting measurements. For example, pressing the *single ping* button results in a single set of data that is then plotted. The plotted data can then be used to determine what changes need to be made to the parameters, if any.

Two files (same name, but different extension) are used by the main application:

- The configuration (`*.cfg`) file stores the parameters
- The data (`*.dta`) file stores the recorded data

Figure 4 shows a typical configuration file. The configuration file is designed to be human-readable to assist with debugging, and to provide useful output for the user.

```
name test1
csound 1487.000000
fcarrier 300.000000
nover 8
pulsecycles 150
max_range 5.000000
num_channels 6
amp_gain 20
angle_start -14.000000
angle_step 3.000000
angle_end 10.000000
```

Figure 4: Typical Configuration File (test1.cfg)

The basic operation of the main application is summarized in Figure 5, in the form of psuedocode.

```
load/enter/edit the parameters
initialize hardware
center the array (zero-degree position)
move motor to start of scan
for angle = start_angle to end_angle
{
        ping
        record data
        save data to *.dta file
        move motor clockwise by angle_step
}
call display application to display results
```

Figure 5: Psuedocode for Basic Operation of Main Application

Error! Reference source not found. shows a graphical representation of the data organization after it has been loaded into a matrix from a data file. The data is recorded linearly in the file, in such a way that the matrix (represented in **Error! Reference source not found.**) is easily generated using the MatLab `reshape` command. With reference to the Figure, each of the channels comes from an individual transducer. The angle data is recorded for every ping on every channel. Although this arrangement results in some redundancy, it does allow the data from a single ping to be easily extracted from the matrix, along with its associated angle.

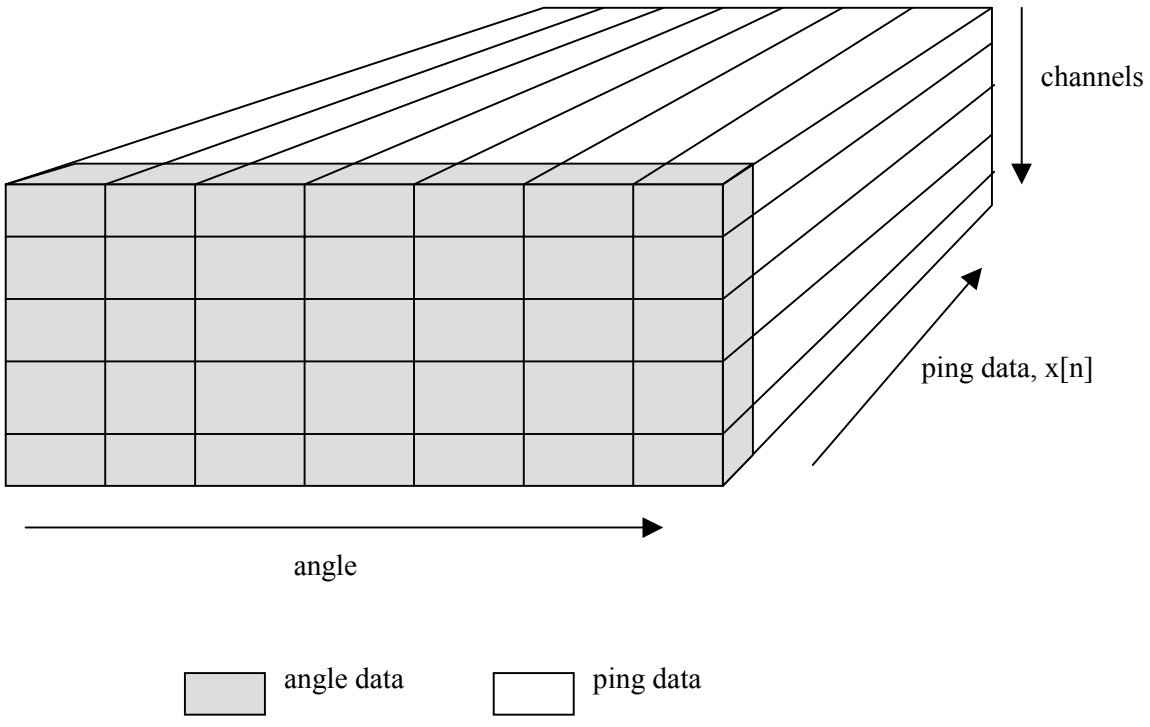


Figure 6: Graphical Representation of Data Organization

3.3 Display Application

3.3.1 User Interface

The user interface for the display application is shown in Figure 7.

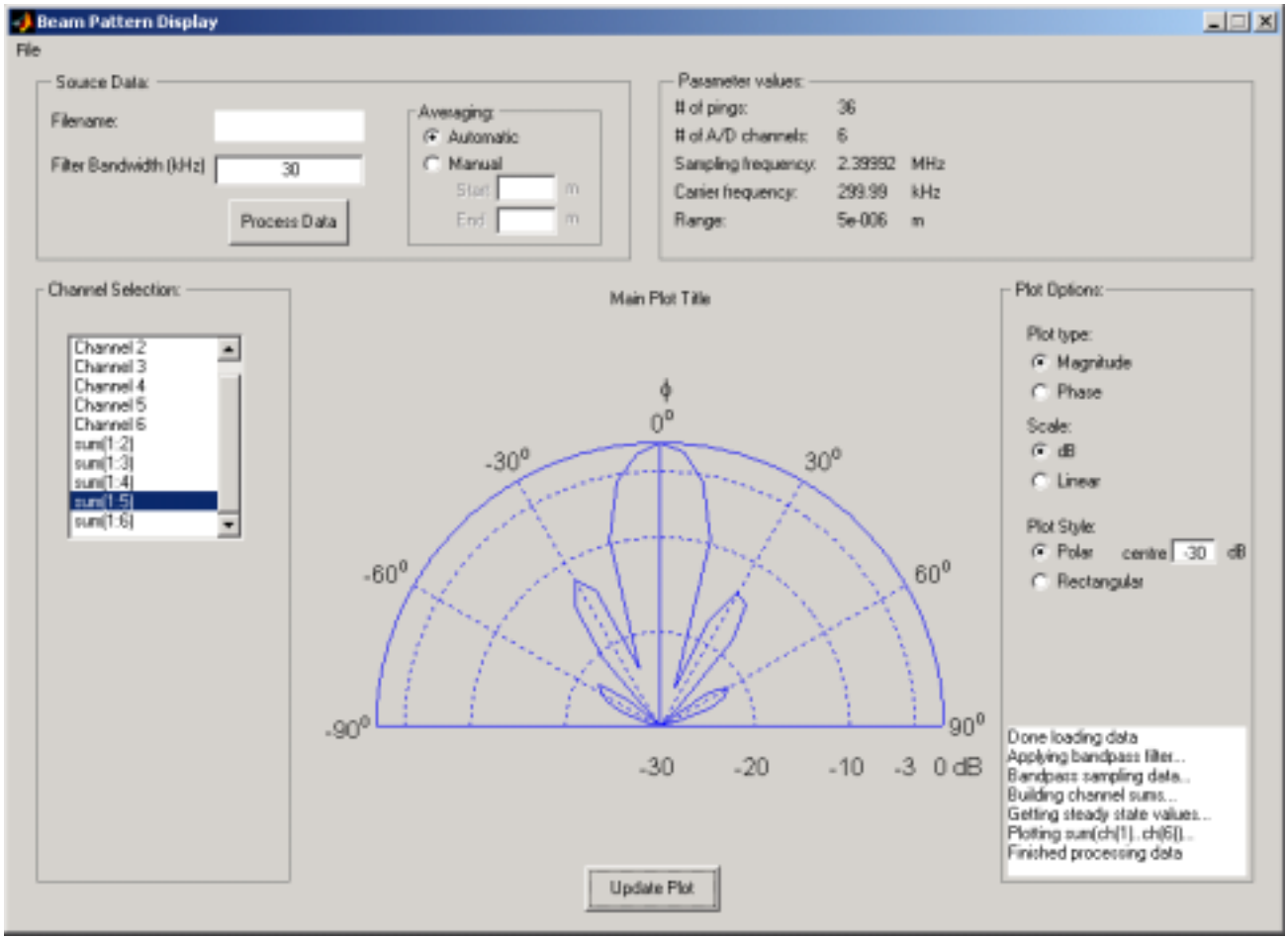


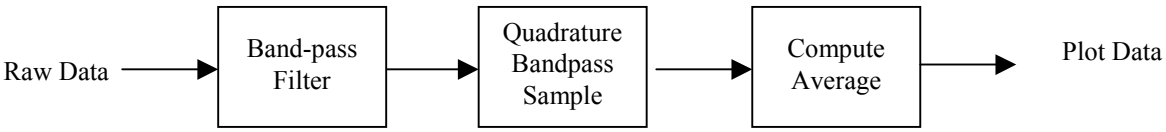
Figure 7: Screen shot of display application

A button on the scan control interface launches the display GUI and passes a file name parameter. If the display GUI is launched in this manner, it will automatically load the data and scan parameters from the appropriate file. Otherwise, the design allows for the user to browse for previously stored data sets.

When the “Process Data” button is pushed, a function is called to perform the appropriate signal processing as described in the following section. By means of providing a list box of channels and radio buttons to select the plot style, the interface allows for the user to easily change what is displayed in the plot window.

3.3.2 Functionality

The role of the display application is to read in raw data from a file created by the scan application, then to filter, sample, average and display the associated beam pattern for each channel of data collected and for combinations of channels. Figure 8 shows the



signal processing block diagram.

Figure 8: Signal processing block diagram

The band-pass filter uses MatLab signal processing tools to compute and apply a 256-point finite impulse response filter kernel. The bandpass filter is centered at the carrier frequency of the acoustic signal and is used to suppress noise.

A technique known as quadrature band-pass sampling is used to simultaneously down-sample the data and detect the signal envelope. By sampling at two points per carrier cycle separated in phase by 90^{o}, and adding the two sets of points in quadrature, one obtains a complex signal that has a magnitude equivalent to the envelope of the acoustic signal.

The signal sent by the transmitter is a rectangular pulse modulating a 300 kHz sine wave carrier. Assuming the pulse duration is much longer than any transient decay time, the envelope amplitude detected at the receiver is equivalent to its sinusoidal steady state response. The beam pattern is defined as the power response as a function of angle of arrival. As a result, to obtain the steady-state response, the filtered data is then averaged over the middle half of the received pulse for each angle of arrival.

Once the signal processing is complete, the data is reduced to a matrix of values representing the response for each channel at each angle of arrival. Depending on the user's selections, the amplitude response can be plotted in decibels or linear units, on either polar or rectangular axes. The user may also choose to plot the phase response.

In addition, the display application allows the user to print or export the plots to other graphics formats.

4 Motor Control

4.1 Software

The software program has been developed to control the motor and allow bidirectional motor rotation. The software will be responsible for controlling the yaw and roll motors for horizontal and vertical rotation. The program read in the signals from the parallel port which is connected to the motor control board. The program is written in Microsoft Visual C++[1] and compiled into a Mex file so it can be incorporate with rest of the application software.

Figure 9 shows the flowchart of the overall application:

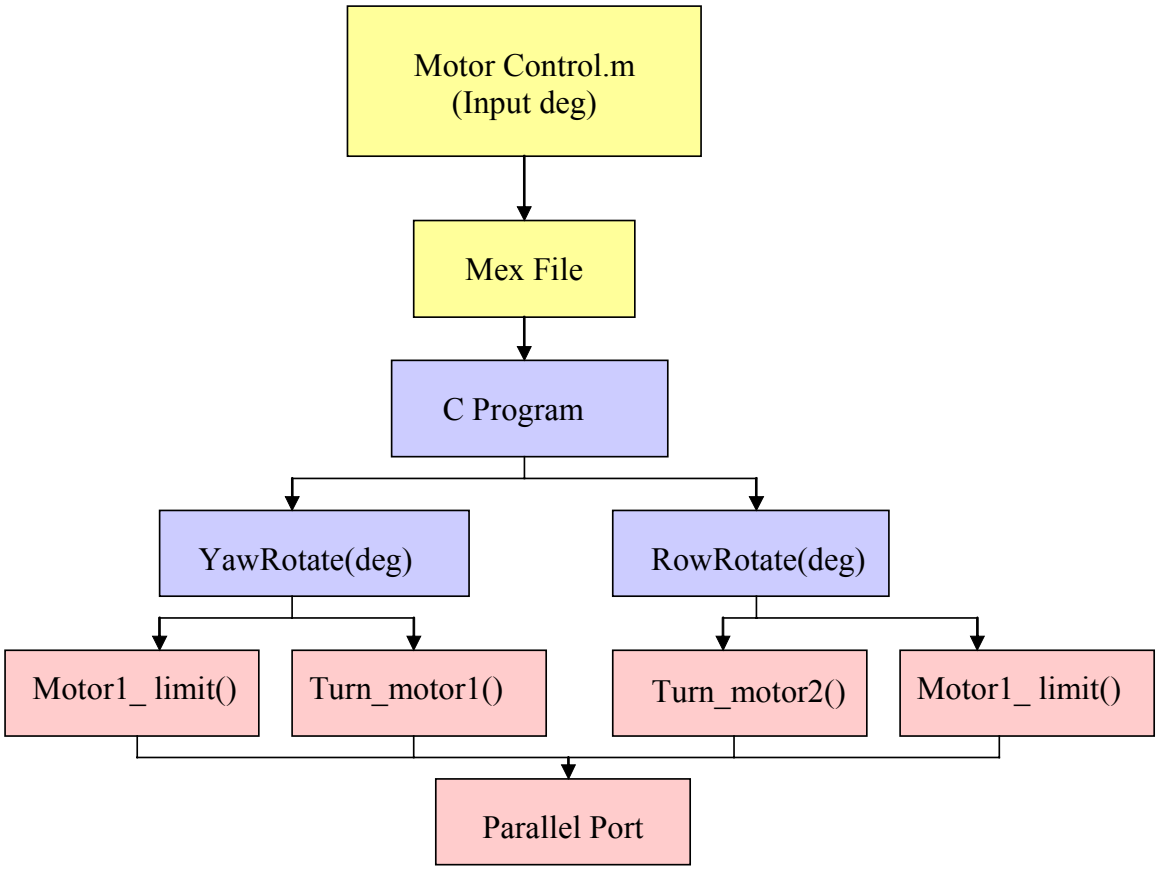


Figure 9: Flow chart for the Motor Control Software

[1]The reason for developing the code in Visual C++ is because some motor control code has been developed by another student, Victor Leung, and several necessary algorithms are supplied by the manufacturer in C++; however, the overall system does not match with our application needs, hence extensive modification of the code is required.

The overall program can be broken down into two parts, Matlab and C. The section below will illustrate how each section works:

4.1.1 Matlab

For the Matlab function code, the motor control software that is written in C was wrapped up inside a Mex File. Mex files allow programs that are written in other languages to be compiled and executed from within Matlab.

The main program can control motor through following code:

[motor_limit,deg] = motor_rotate(yaw or row,deg);

There are two inputs which the controlling program will need to specify, with the first indicating which motor to rotate and the second telling the system by how much it will need to rotate. Positive degree values indicate a clockwise rotation and negative values indicate a counter clockwise rotation. The Mex file will pass back two variables to the main program indicating how many degrees the specified motor has rotated and whether the motor limit has been reached. These variables are important, because the physical constraints of the stepper motor will prevent it from exactly rotating the amount requested. These physical constraints include the finite step size of the stepper motor, which forces the yaw motor to move in steps of 0.056 degrees and the roll in steps of 0.112 degrees.

4.1.2 C Programming

The overall C++ program can be broken down into two functions: *yaw rotate* and *roll rotate*. For both functions, each will call the sub functions *turn motor* and *motor limit*, the system will send commands through the parallel port to the motor control board to turn the motor, while at the same time checking to make sure that motor has not yet reached its physical limit.

Two *turn motor* functions have been written: one is to control yaw rotation and the other is used to control roll rotation. Their main functionality is to drive the motor by sending the correct eight step switching sequence. After eight correct steps, the rotor has turned one full tooth, which is equivalent to 0.056 degree in this case. Some delay is added in between stepping to give rotor sufficient time for rotation.

The *motor limit* functions are used to check limit bits on the parallel port. For both yaw and roll motors, one bit is used to indicate whether the yaw or roll has reached its maximum or minimum position and another gives information on whether the yaw and roll has reached its home position. The program will send the motor to its home position when it is first initialized and then use the home position as a reference for calculating the current motor position.

4.2 Electrical Hardware

4.2.1 Stepper Motor

A NEMA 23 Slo-syn unipolar stepper motor was chosen to implement the BPMS because of its high repeatability, which allows easy comparison between test runs. The major features of the stepper motor are listed below:

Manufacturer:	Superior Electric
Type:	NEMA 23 Slo-syn unipolar, part #M061-LS-311
Voltage supply:	20V DC
Current per winding:	0.22A
Nom. resistance per winding:	91Ω
Typical step time:	12ms
Degrees of freedom:	2
Yaw step size:	0.056 degrees
Yaw range of motion:	200.6 degrees
Roll step size:	0.112 degrees
Roll range of motion:	84 degrees

4.2.2 Cables/Shielding

The cabling used to connect the stepper motor to the motor control board contains a total of ten conductors which are contained within 4 shielded packets and a single jacket. Two of these cables exit the stepper motor assembly, one each for the yaw and roll motors. Shielding ensures that the enclosed wires do not interfere with the signals received from the sonar transducers themselves, which are susceptible to noise.

The connection between the motor control board and the stepper motor has been made using BNC24-type plugs. These plugs were chosen specifically because they are non-standard for most electronics equipment, and therefore run little risk of being mistakenly plugged into the wrong socket—a very probable mistake had a standard D-connector been used, considering that the control board must also be connected to a computer parallel port, which itself uses a D-connector. The pins of the plug are geographically divided so as to separate the conductors for each motor, and so that DC power lines separate the hall sensor lines from the motor control lines to prevent then from interfering. The pinout for the BNC24 plugs is shown in Figure 10, with the signal associated with each pin detailed in Table 1.

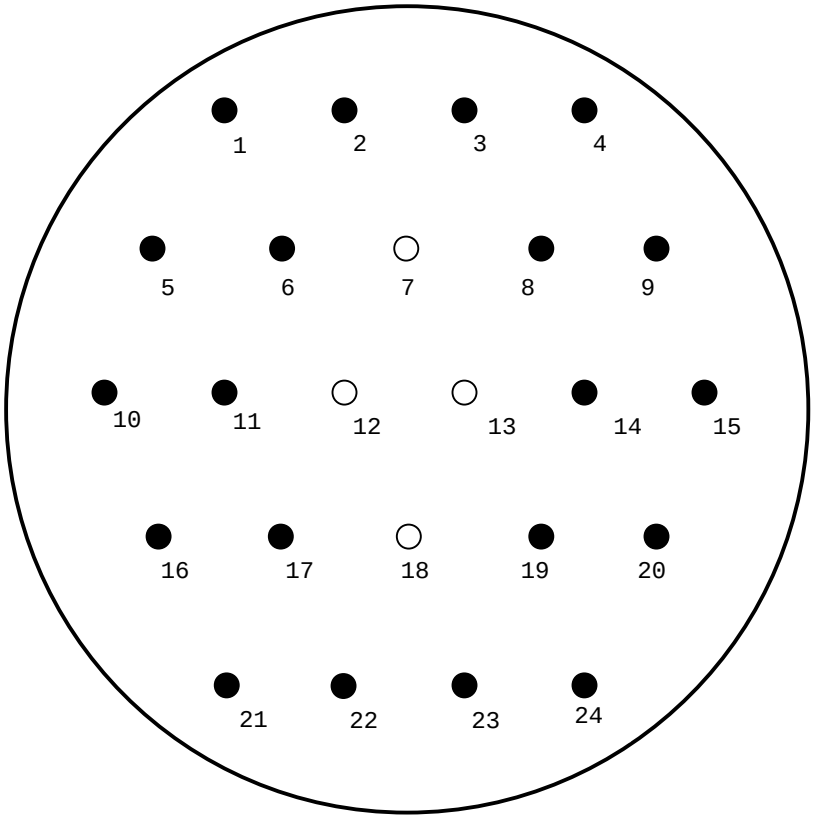


Figure 10: Motor BNC 24-Pin Connectors (female perspective)

Table 1: Signals associated with Figure 10 pinout

Pin	
1	Roll hall sensor 1
2	Roll +5V
3	Yaw +5V
4	Yaw hall sensor 1
5	Roll hall sensor GND
6	Roll hall sensor 2
7	NC
8	Yaw hall sensor 2
9	Yaw hall sensor GND
10	Roll signal 1
11	Roll GND 1
12	NC

Pin	
13	NC
14	Yaw GND 1
15	Yaw signal 2
16	Roll signal 3
17	Roll signal 2
18	NC
19	Yaw signal 1
20	Yaw signal 3
21	Roll signal 4
22	Roll GND 2
23	Yaw GND 2
24	Yaw signal 4

4.2.3 Waterproofing

An obvious concern for sonar hardware is its protection from water submersion. The stepper motor itself is sealed using an o-ring and silicone water sealant, thus enabling it to be submerged 80% underwater. The top of the motor casing is open to permit the escape of the necessary wiring, and this therefore limits the motor from being completely

underwater. A splashguard has been added to the motor top to prevent casual splashing from entering the casing.

4.2.4 Motor Control Board

The motor control board used to interface with the stepper motor is a legacy hardware piece and was not designed by Sonar Works. It is responsible for decoding instructions from the computer, logic-inverting the hall sensor outputs to make them compatible with the computer's parallel port, and providing protection against possible back EMF from the motor. The wire connectors between the motor control board and BNC24 plug are shown in Figure 11, with the individual pin assignments given in Table 2.

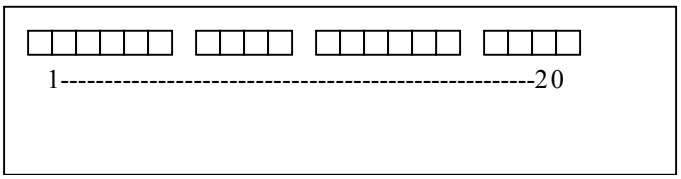


Figure 11: Wire connections between motor control board and BNC24 plug

Table 2: Pin assignments between BNC24 plug and motor control board

Motor control board pin	BNC 24 plug pin	Signal function
1	10	Roll signal 1
2	11	Roll GND 1
3	17	Roll signal 2
4	16	Roll signal 3
5	22	Roll GND 2
6	21	Roll signal 4
7	3	Yaw +5V
8	9	Yaw hall sensor GND
9	8	Yaw hall sensor 2
10	4	Yaw hall sensor 1
11	19	Yaw signal 1
12	14	Yaw GND 1
13	15	Yaw signal 2
14	20	Yaw signal 3
15	23	Yaw GND 2
16	24	Yaw signal 4
17	2	Roll +5V
18	5	Roll hall sensor GND
19	6	Roll hall sensor 2
20	1	Roll hall sensor 1

4.2.5 Enclosure

A metal enclosure was constructed to house the motor control hardware. The enclosure design can be seen in Figure 12. The choice of a metal enclosure was made in order to better dissipate heat, to provide electromagnetic shielding, and to easily allow machining. The disadvantage of metal is that it is not as resistant to water and rust as plastic casing. The enclosure also allows the use of the BNC24 plug for motor control signals and standard banana cables for the power supply.

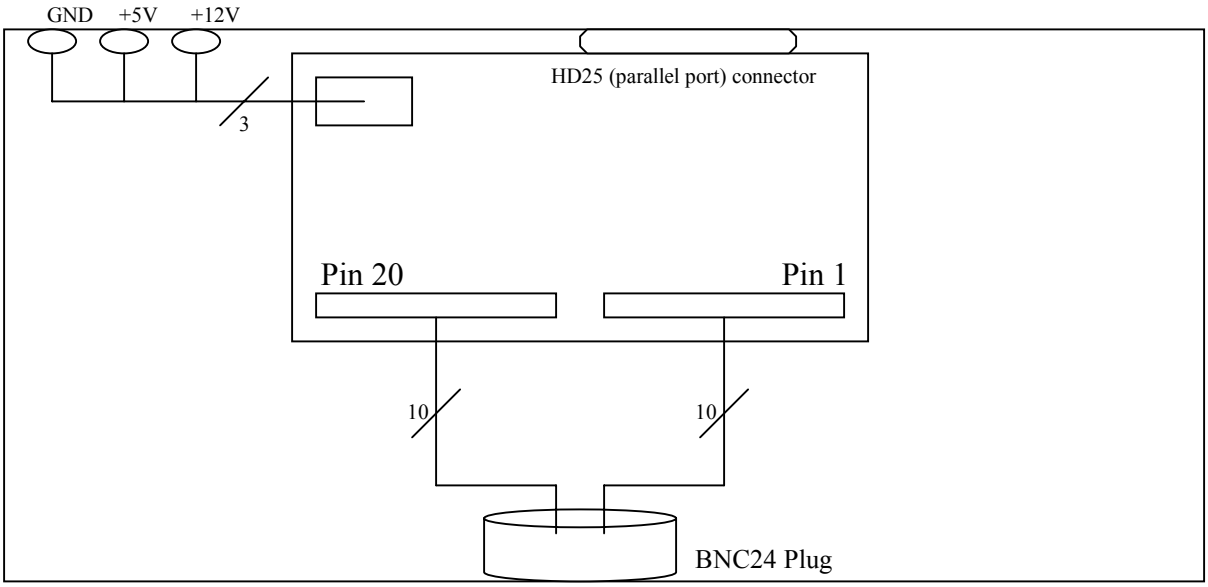


Figure 12: Enclosure for motor control board

4.3 Mechanical Hardware

The general arrangement of the transducer mounting is shown in Figure 13. The basic elements of the required hardware are already present in the Underwater Research Lab. In the interests of economy and recycling, the mechanical design will utilize as much of the existing hardware as practicable. However, the existing hardware needs to be carefully checked and/or modified to ensure ease-of-use requirements are met, and to ensure watertight and structural integrity.

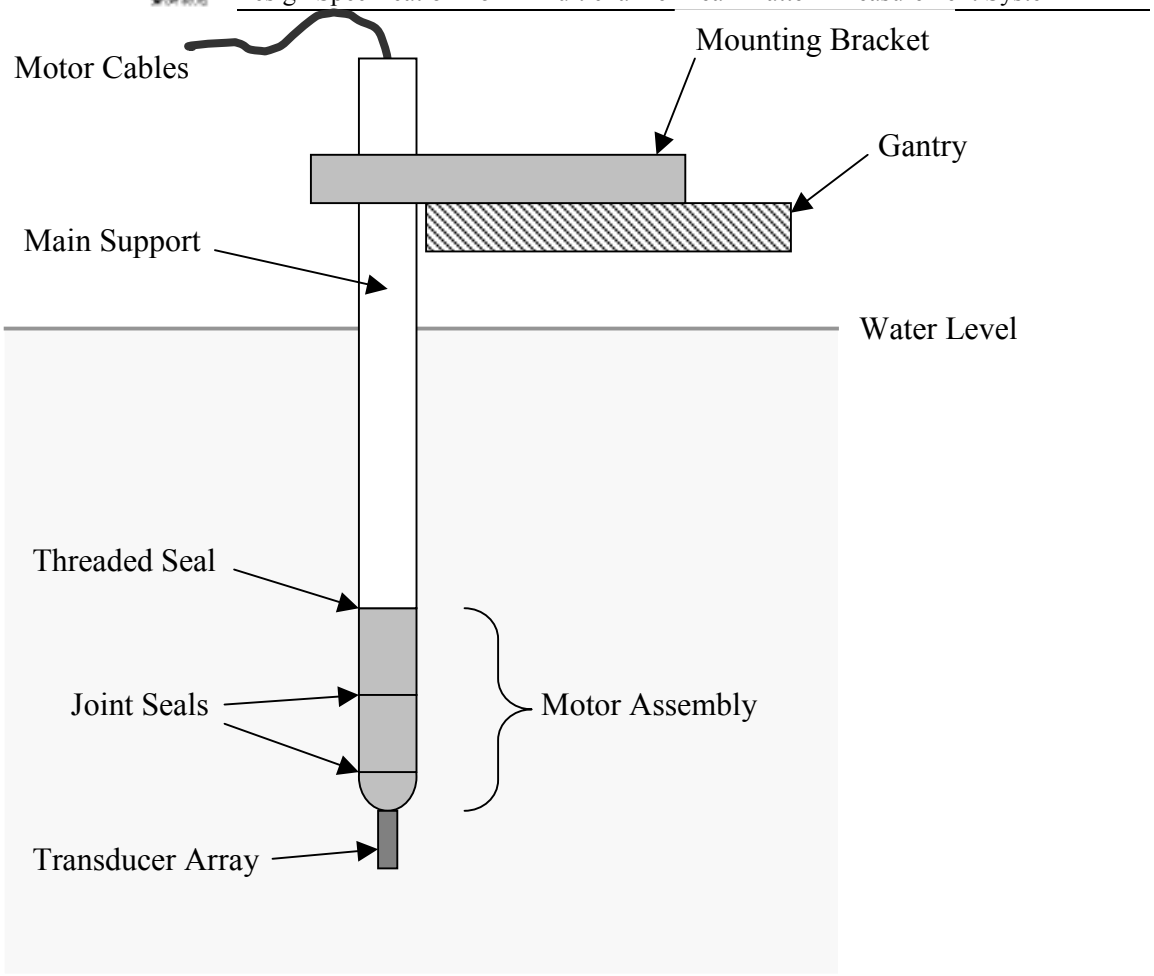


Figure 13: General Arrangement of Transducer Mounting

4.3.1 Mounting Bracket

The mounting bracket connects the motor assembly to the gantry via the main support (see: Figure 13). Any wobbling of the transducer array affects the quality of the beam pattern measurements. In addition, the mounting needs to be secure – if the motor assembly is dropped, damage may occur to both the motor assembly and the tank. Therefore, the mounting bracket must hold the main support very firmly. Further requirements are that the bracket needs to be easily opened and closed by one person, and that the mounting arrangement must be suitable for operation in a wet environment Portability of the mounting arrangement (for in-field testing) is also desirable.

Given these requirements, the bracket will be made of a suitable metal, e.g.:

- Aluminum alloy – heated treated and aged to give suitable strength and hardness. The main support (ref: Figure 13) is made of 6061-T6 aluminum alloy. This alloy, or something similar, would suitable for the bracket.
- Coated Steel – steel provides great strength, while the coating provides protection against corrosion in the wet environment.

The existing bracket in the Lab is made of coated steel and is of sturdy construction. However, it does not allow easy operation by one person. Given the time and cost constraints of this project, modifying the existing bracket to improve its usability is probably the best design choice.

4.3.2 Watertight Seals

The motor assembly is a 2-DOF mechanism and operates under water (see: Figure 13). Therefore, seals that allow relative motion between the components (i.e., joint seals) are required.

The motor assembly is connected to the main support via a screw thread. This connection needs to be made watertight. The design will employ a rubber "O" ring with a 75 mm diameter (lubricated with silicone oil), with PTFE tape on the thread to ensure a tight connection.

4.3.3 Test Plan for Mechanical Hardware

1. Motor operation – after sealing the motor assembly, check that the motors function normally, and that the mechanism has full freedom of movement.

2. Immersion test – test the watertight integrity of the unit, including the seals. The sealed unit will be immersed in the tank at maximum operating depth (≈1.5 m) while disconnected from the power (to reduce risk of shorting due to water ingress). The unit will then be inspected after being immersed for 24 hours to see if any water has entered. If not, the unit will be immersed and the motors operated, to ensure that the unit works correctly while under water, and that the joints seals do not leak. Finally the unit will be checked again to see if any water entered.

3. Calibration test – this test is to determine the accuracy/precision of the mechanism.

4. Usability test – check to ensure that one person can easily mount the mechanism in the tank with minimum risk of damage to mechanism and tank.

5 Conclusion

This document provided technical details of the system and sub-component design, including testing considerations. The design of the various software and hardware components was described and an overview of the complete system was presented.

6 References

Haintz, P. September 2003. Multi Channel Coherent Beam Pattern Measurement System. *Draft Master's Thesis.* Simon Fraser University.

