Architecture of a Dynamic Position Autonomous Vessel

Mr. Jonathan Edward Paquette, US Coast Guard

I am a Coast Guard Ensign serving in Cape May, NJ. I received my BS in Electrical Engineering from the Coast Guard Academy.

Thomas Robert Cogley
Dr. Tooran Emami, U.S. Coast Guard Academy

Tooran Emami is Tenure Track Assistant Professor in the Department of Engineering, Electrical Engineering Section, at the U. S. Coast Guard Academy. She received M.S. and Ph.D. degrees in Electrical Engineering from Wichita State University in 2006 and 2009, respectively. Dr. Emami was an adjunct faculty member of the Department of Electrical Engineering and Computer Science at Wichita State University for three semesters. Her research interests are Proportional Integral Derivative (PID) controllers, robust control, time delay, compensator design, and filter design applications, for continuous-time and discrete-time systems.

Lt. Aaron Peder Dahlen, USCG
Dr. Richard J. Hartnett P.E., U.S. Coast Guard Academy

Architecture of a Dynamic Position Autonomous Vessel

Abstract

This paper presents the final work from a one year senior capstone project in Electrical Engineering at the U.S. Coast Guard Academy. In this project three students are asked to apply lessons learned through three previous years' academic experience to an autonomous floating vessel, for the purposes of modeling and real-time heading control. Consistent with the maritime focus of the U. S. Coast Guard, students construct a four by eight foot barge propelled by six commercial Minn Kota trolling motors, in order to study real time heading control algorithms such as those which might be encountered on typical Mobile Offshore Drilling Units (MODU's). Controlling a smaller scale model platform provides students an opportunity to perform system identification and control, and allow students opportunities to expand their knowledge base in control systems, electronic navigation systems, and software design.

In this senior project, students constructed a mobile barge, and made use of a centralized database where system identification and controller data from different control architectures could be captured and evaluated. This centralized database is chosen because it allows multi-user applications/programs to access, query, and write data simultaneously, without closing connections. An AIRMAR PB200 WeatherStation sensor is used to collect vessel data (GPS/WAAS location, vessel heading, and relative wind). Project hardware and software is demonstrated, and experimental data is collected from this vessel at the United States Coast Guard Academy. Here we present the design, plus performance data taken from this platform.

Introduction

The United States Coast Guard Marine Safety Center sponsored a senior project at the U.S. Coast Guard Academy to design a small autonomous vessel for future applications to model a Mobile Offshore Drilling Unit (MODU). To accomplish this, a controller must be developed for the MODU model that is able to maintain a desired heading and position simultaneously in the presence of disturbance inputs (such as wind or current), or loss of a single thruster. An integrated sensor provides WAAS enabled GPS position information, plus heading information. Before beginning the controller design, the system is identified for both rotational and translational dynamics. All data acquisition and control is accomplished wirelessly from shore via computer.

Many researchers have been developing different autonomous dynamic positioning systems around the world. Ker-Wei, Yu and Hsu used the particle swarm optimization method for a ship coordinate system [1]. Alarçïn used a neural network based on an internal model to control the roll motion of a ship [2]. Moan developed the safety and challenges of station-keeping systems in the view of in-service experience [3]. Fu, Ning, and Wei developed a method of on-line reconfiguration of a dynamically positioned vessel's controller by using a virtual thruster in the instance of thruster failures [4]. Xia, Shi, Fu, Wang, and Bian used hybrid Proportional Integral Derivative (PID) controllers with a neural network to make an adaptive control for a ship [5]. Pettersen and Fossen added an integral action in the feedback path to control the orientation and position of a ship with experimental results [6].

In order to learn more about dynamic positioning and autonomous vessel control, three cadets at the United States Coast Guard Academy (USCGA) built a four by eight foot barge driven by six commercial Minn Kota trolling motors. Figure 1 shows this platform.

Fig. 1. Four by eight foot autonomous vessel with six commercial Minn Kota trolling motors and an AIRMAR PB200 WeatherStation sensor

This paper presents some of the successful design architecture that students use to develop autonomous vessel architecture for a capstone project in Electrical Engineering at the USCGA. The educational purpose of this project helps students to apply the following lessons: 1) performing centralized data collection, i.e., software design), 2) linearization of a nonlinear system, i.e., apply math to an engineering practice, 3) utilizes sensor, i.e., apply electronic navigation systems, 4) identification of the system dynamic and controller design, i.e., apply control systems concepts, 5) troubleshooting over the test i.e., re-design an experience. A centralized database is chosen that allows multi-user applications or programs in the present case to access, query, and write data simultaneously without closing connections. In order to reduce the non-linearity of the system with a linear logic controller, a Printed Circuit Board (PCB), designed by Electrical Engineering students at the USCGA, is used to control the motors thrust by a computer. An AIRMAR PB200 WeatherStation sensor is used to collect vessel data (vessel location based on GPS/WAAS, vessel heading, and relative wind).

The milestones for the project are summarized as: performing data storage center, linearization of motor trust, sensor utilization, the identification of system dynamic, initial PID controller design, and finally Logic Proportional Integral Derivative (LPID) controller design. The real time heading controller for the vessel is based on updating a LPID coefficients. Experimental data, taken from this vessel at the USCGA, is used to demonstrate the application of this architecture. All data, including the controller states, are collected in the database for the real time control and post analysis.

This paper is organized as follows. First, the functional networking design is presented. In this section the data storage center, linearization of motor trust, networking and sensor utilization, and autonomous control design are introduced. Next, the simulation result from this architecture is presented. Finally, the conclusion of this paper is presented.

Functional Networking Design

This Section describes the networking between several vessel subsections. One of the subsections is the database center. The motors' thrust linearization and PCB control is described next. After that, the communication networking via a wireless ad-Hoc network between a land based computer and the vessel is discussed. Finally, the autonomous controller architecture for the vessel is described.

A. Data storage center

Initially, all data was stored in text files. However, this method of data storage became an issue due to the system architecture. The benefit of the text files is that they are simple to write to using *file I/O*, but the downfall of text files is that they cannot be opened from two instances of MATLAB® at the same time. The use of text files would have required the data acquisition script to be connected to the controller script to coordinate reading from and writing to the file in order to prevent the two scripts from attempting to access the same file simultaneously.

One of the solutions is to use a local MySQL database. Using the database presents several positives. First, the database is a multi-user application that permits separate users, or programs in the present case, to access, query, and write data simultaneously without closing connections. This permits the system components to retain their autonomy.

The database also eases the process of analyzing the data during testing and in the laboratory. The MySQL Workbench made it easy to view the data during testing on the water to confirm functionality of the controller and sensor systems. Once testing is complete, the entire database is saved to a *.sql* file to be analyzed easily on computers in the laboratory.

B. Linearization of motor trust

The first step in autonomous vessel control is the development of computer controlled motor thrust. Six, 80 lb., thrust trolling motors propel the vessel. In their original state, the motors are controlled using an analog potentiometer located within the motors' handles. Twisting the handle adjusted the voltage applied to the motor head. A PCB is replaced the analog potentiometer to control the motor trust by a computer. The PCB is shown in Figure 2.

Fig. 2. Motor control printed circuit card (PBC)

The PCB creates a passive system. From the perspective of the motor, the input appears identical to that of the potentiometer. The PCB is controlled by an eight-bit Pulse Width Modulated (PWM) signal from a micro controller. A basic flow diagram of the system as implemented on the controlled vessel is shown in Figure 3.

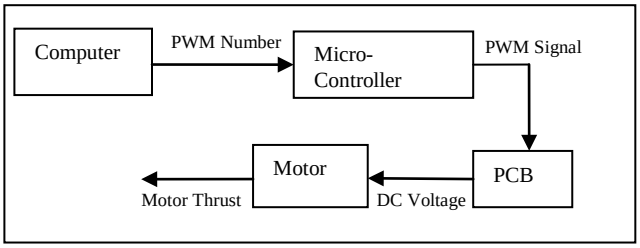

Fig. 3. The motor controller follow chart

The PCB operates on the following principles. First, the PCB isolates the motor from the micro controller using an opto-isolater. The opto-isolater recreates a PWM input signal using the motor head's voltage source. The signal is then passed through a low pass filter and an op-amp, which transforms the signal to a DC voltage. The PCB permits 256 states of thrust.

The next step is to identify and analyze motor response to input signals. The motor does not change linearly to a linear change in percent modulation. The measurement data of the motor response is shown in Figure 4-a. This non-linearity is countered by using MATLAB® to compare the actual motor values to a desired linear set of values ranging from minimum to maximum motor current. The percent modulation, which corresponds to the desired current, is placed in a look up table, which is represented in Figure 4-b. Finally, the motor values are band limited to 20% of maximum possible thrust. The thrust limitation is implemented because water testing of the motors showed motor values over 20% is powerful enough to twist motors off their housing on the vessel. The final set of linearized PWM values are shown in Figure 4-c.

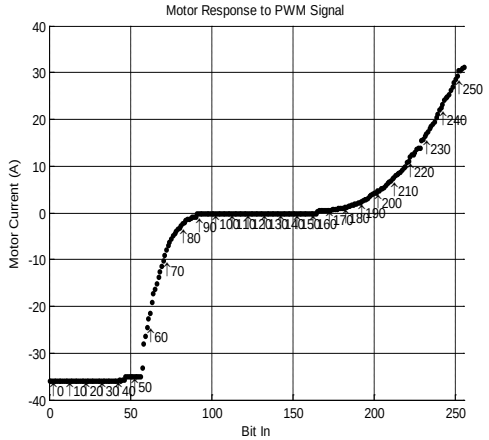

Fig. 4-a. The motor output current to 256 states of motor thrust

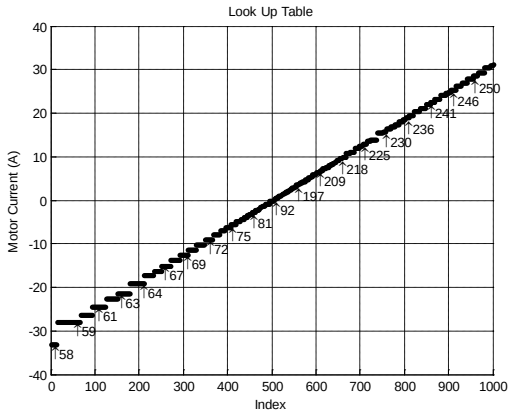

Fig. 4-b. Lookup table

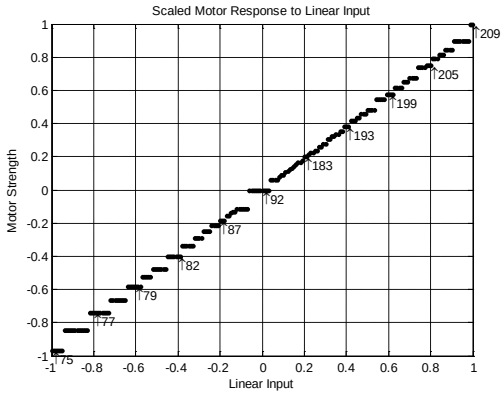

Fig. 4-c. Final linearization values

C. Networking and sensor

A basic flow diagram of the system networking between the on-board computer, ad-Hoc wireless network, AIRMAR PB200 sensor, and six motors are shown in Figure 5.

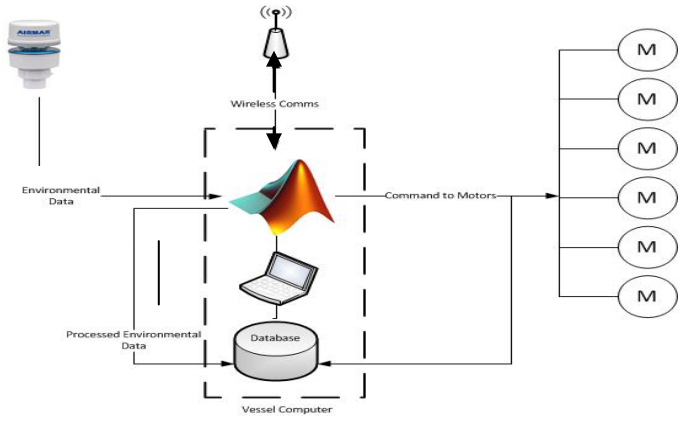

Fig. 5. Vessel communication network

Figure 5 shows that when data is read from the sensor it is written to the local database. The automatic controller reads the position and heading data from the database, which it uses to determine the commands to the motors. The commands are also written to the database for future analysis. The whole process is observed remotely via an ad-hoc network through Microsoft Remote Desktop®, which allows the operator to control the platform from a distance.

For the controller to operate properly, it must have access to information such as position, heading, and the environment. An AIRMAR PB200 sensor is utilized to collect the aforementioned data. The AIRMAR PB200 outputs data in National Marine Electronic Association (NMEA) 0183 strings. This data format is desirable because each data string contains a header indicating the type of data in the string and each string contained a checksum that can be used to eliminate strings with errors. Furthermore, these strings are comma separated which eases the process of extracting data from the strings.

The sensor interfaces with MATLAB® and outputs its data into the buffer. Using the code below, each string is parsed, utilizing the commas as the delineation character.

```
while true
    [str, in_string] = strtok(in_string, ',');
    if isempty(str)
       break;
    end
    count = count + 1;
    holder{1,count} = str;
end
```

The variable, *in_string*, contains the string that is received from the sensor. The string is parsed until each comma-separated section has been extracted from the string and stored in the *holder* cell array so that it could be used later. This method extracts every piece of data from the data string and does not scan the string for a specific piece of data, such as the latitude if the data string was a positional string. There are several benefits of processing the data in this manner.

Primarily, as the platform design is still in the testing phase, the goal is to collect as many types of data as possible. By temporarily holding each piece of data, students are able to pick which data is important to permanently record for further analysis. Secondly, this method reduces the amount of code needed to parse a string because it could be used for every type of data string. If each string was parsed based on the type of data string, it could nearly triple the code needed to do the same operation. Lastly, the current method allows easy access to the string checksum.

The checksum of a string is calculated by performing an *XOR* operation of the first character in the NMEA string with the remaining characters in the string excluding the leading *'$'* and the ending '*'. This operation is completed in the MATLAB® script below.

```
for i = 2:length(scanned_string)
   checksum = bitxor(checksum,uint8(scanned_string(i)));
end
```

In the above code, the variable *scanned_string,* contains the NMEA 0183 string from the sensor with the leading *$* and *suffix* * removed. The output of this operation is converted to a two digit hexadecimal number that represents the checksum of that string.

If the calculated checksum of the incoming data string does not match that of the received checksum, then the corrupt data string is stored with a time stamp to be used for sensor error analysis. However, if the calculated checksum of the incoming data string matches that of the received checksum, then the data received in the string is accepted to be stored in the database to be used by the controller algorithm. Once the data is parsed and the checksum is evaluated, that data is concatenated to form an *SQL* string to store the data in the database. This function is completed by utilizing the *exec* function in the MATLAB® database toolbox.

D. Autonomous controller

Consider the closed-loop SISO system shown in Figure 6, where $G_p(z)$ and $G_c(z)$ is the discrete time plant and initial PID controller, respectively. The sampled controller input and the output signals are $E(z)$ and $U(z)$, respectively. The sampled reference input and the output signals are $\theta_d(z)$ and $\theta(z)$, respectively.

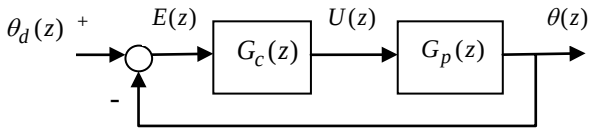

Fig. 6. Block diagram of the closed loop system

Prior to calculating motor thrust values, the proportional, derivative, and integral coefficients must be loaded into the controller. Once system identification is completed, these coefficients will be set values. However, tuning of the controller requires that these can be modified while testing the vessel. The LPID controller will update via the real data over the test condition of vessel as the following controller output:

$$u(k) = K_p e(k) + K_i \sum_{n=0}^{k} e(k) + K_d \left(e(k) - e(k-1) \right) \qquad (1)$$

The heading angle error, $e(k)$, can be defined from the real data over the test as the difference between the desired heading angle, θ_d, and measurement heading angle, θ,

$$e(k) = \theta_d(k) - \theta(k) \qquad (2)$$

The final system to develop is the automatic controller. The controller is composed of a set of MATLAB® callback functions, all of which are controlled by a graphical user interface (GUI). The callback functions serve to read position data from the database, compare current location to a desired location in order to determine motor values to regain position, and write outputs to the motors. There are also a number of functions which store motor values, a desired geographical location, and other controller information in the database for later review. Figure 7 shows the GUI for the vessel control.

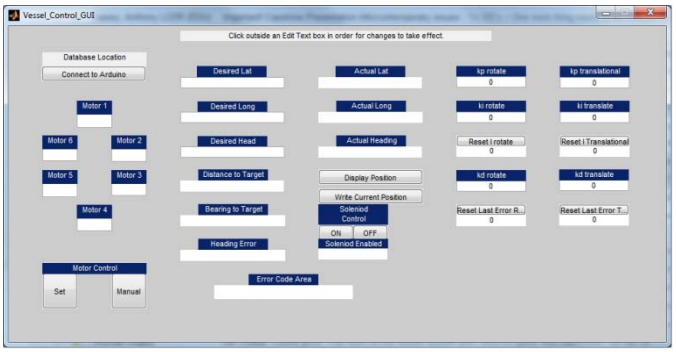

Fig. 7. Graphical user interface for the vessel control

The GUI allows the user to 1) set a desired location and heading, 2) manually control of the thrusters, 3) temporarily store error values for tuning integral and derivative components of the controller and 4) display current information about the vessel, including current motor thrust values, position, heading, and distance to the desired location.

Experimental Results

This section presents a post analysis of measurement data to design the initial discrete-time PID controller via simulation to regulate the heading angle of the vessel.

The vessel has been tested on the open water shown in Figure 8. Test I is marked with a green pin labeled *Test I (Tether)*. An AIRMAR PB200 WeatherStation sensor is used to collect the measurement data from the vessel. The input to the vessel is the motor thrust value and the output is the heading angle of the vessel. The discrete-time experimental measurement heading angle data for the vessel are shown in Figure 9.

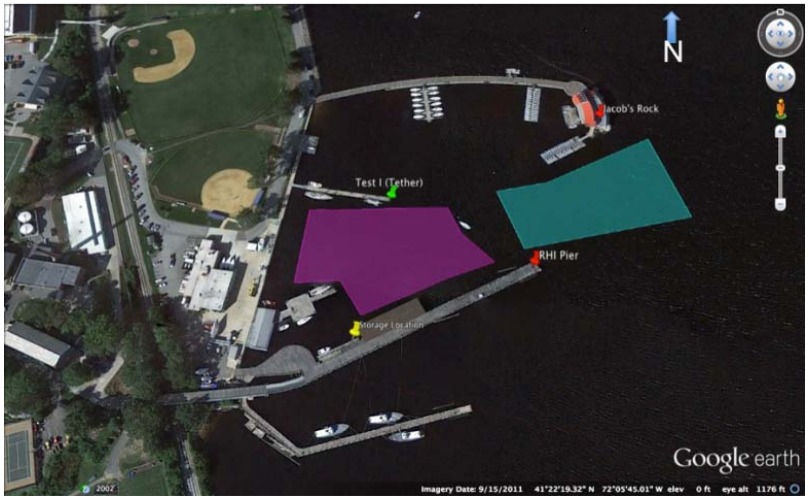

Fig. 8. Testing Area Map

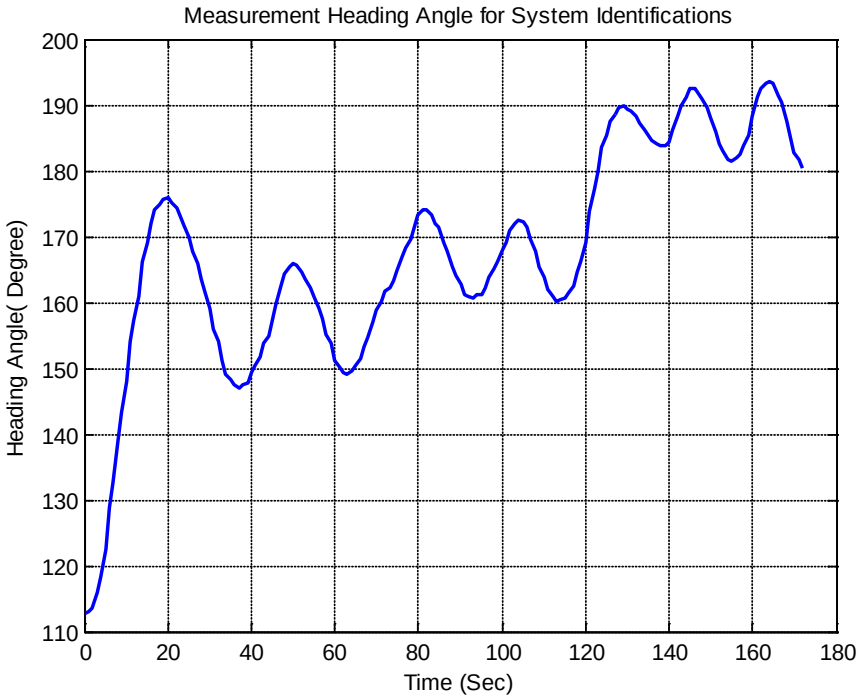

Fig. 9. Discrete-time experimental measurements showing heading angle (output) data from random motor thrust inputs.

Future goals are to employ the linear least squares methods described in [7]and [8] to identify the discrete time step-invariant equivalent transfer function of motor thrust input to heading output, and that research remains to be done. Following the plant identification, the PID controller will be designed such that the heading controller performance will exhibit an optimization of steady state error, settling time, and the percent overshoot. Finally, students would tune the LPID controller coefficients while testing the vessel. The LPID controller would be updated in MATLAB® from (1) and (2) via a real time data collection while testing the vessel.

Conclusions

This paper introduced the architecture of an autonomous floating vessel for real time heading control. A centralize database was chosen that allows multi-user application in the present case, to access, query, and write data simultaneously without closing connections. This permits the system components to retain their autonomy. A Printed circuit board (PCB) was used to control the motors thrust by a computer. The controller outputs were combined and normalized by the computer before being sent to the PCB. An AIRMAR PB200 WeatherStation sensor was used to collect vessel data. All data including controller states were collected in the database for real time control and post analysis. The automatic real time controller consisted of discrete time Logic Proportional Integral Derivative (LPID). This architecture was demonstrated by experimental data collected from the vessel at the United States Coast Guard Academy.

Acknowledgment

The authors extend special gratitude to ENS Hichem Rehouma for his work through his senior year on this project, focusing especially on the power distribution for the platform. We wish to convey a special thanks to all U.S. Coast Guard Academy employees who helped our team with building and testing the vessel on the water. We also wish to express our thanks to the reviewers of this paper for their supportive and constructive comments.

Bibliography

[1] Ker-Wei, Yu and Jai-Hao Hsu, "Fuzzy gain scheduling PID control design based on particle swarm optimization method," *IEEE, Innovative Computing Information and Control Conference*, September 2007.
[2] F. Alarçïn, "Internal model control using neural network for ship roll stabilization," Journal of Marine Science and Technology, vol. 15, no. 2, pp 141-147, 2007.
[3] T. Moan, "Safety management of deep water station-keeping systems," Journal of Marine Sci. Appl., no. 8, pp-83-92, 2009.
[4] M. Fu, J. Ning, and Y. Wei, "Fault-tolerant control of dynamic position vessel by means of a virtual thruster," *Proceedings of the 2011 IEEE International Conference on Mechatronics and Automation*, Beijing, China, August, 2011.
[5] G. Xia, X. Shi, M. Fu, H. Wang, and X. Bian "Design of dynamic position systems using hybrid CMAC-based PID controller for a ship," *Proceedings of the 2011 IEEE International Conference on Mechatronics and Automation*, Niagara Falls, Canada, July, 2005.
[6] K. Y. Pettersen and T. I. Fossen, "Underactuated dynamic positioning of a ship-Experimental results," *IEEE Transactions on control systems technology*, vol. 8, no. 5, September, 2000.
[7] C. L. Phillips and H. T. Nagle, *Digital Control System Analysis and Design*. Prentice Hall, Inc. Upper Saddle River, New Jersey, 1995.
[8] T. Emami, R. J. Hartnett, and J. M. Watkins, "Estimate of Discrete-Time PID Controller Parameters for H-Infinity Complementary Sensitivity Design: Autonomous Sailboat Application," *Proceedings of the 2013 American Control Conference*, Washington DC, June 2013.