NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

CONTOUR TRACKING CONTROL FOR THE REMUS AUTONOMOUS UNDERWATER VEHICLE

by

Alan Robert Van Reet

June 2005

Thesis Advisor: Anthony Healey

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE June 2005	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE: Contour Tracking Control for the REMUS Autonomous Underwater Vehicle	5. FUNDING NUMBERS
6. AUTHOR(S) Alan Van Reet	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited	12b. DISTRIBUTION CODE

13. ABSTRACT (maximum 200 words)

In the interest of enhancing the capabilities of autonomous underwater vehicles used in US Naval Operations, controlling vehicle position to follow depth contours presents exciting potential for navigation. Use of a contour tracking control algorithm in lieu of preprogrammed waypoint navigation offers distinct advantages within new challenges. The difficult nature of this problem lies in the non-trivial connection between the necessary corrective action and the feedback error used in traditional control methods. Stated simply, modern vehicle control algorithms separate horizontal and vertical plane navigation. The autonomous vehicle senses heading error and applies rudder to steer the vehicle to a desired heading. Simultaneously, the vehicle might sense altitude and apply stern plane angles to maintain a safe height above ground. This thesis research examines the new problem of sensing depth and altitude in the vertical plane while steering the vehicle horizontally to find a specified bathymetry contour. While more remains to understand, this research proves the existence of a solution and suggests similar approaches may facilitate tying vehicle navigation to other indirect sensors. This thesis presents two contour tracking control algorithms and examines the performance of each by simulating the response of the REMUS underwater vehicle to ideal and real-world bathymetry models.

14. SUBJECT TERMS Contour Tracking, Autonomous Control, REMUS, Underwater Vehicle, AUV	15. NUMBER OF PAGES 83
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

CONTOUR TRACKING CONTROL FOR THE REMUS AUTONOMOUS UNDERWATER VEHICLE

Alan R. Van Reet
Ensign, United States Navy
B.S., United States Naval Academy, 2004

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
June 2005

Author: Alan Van Reet

Approved by: Anthony J. Healey
Thesis Advisor

Anthony J. Healey
Chairman
Department of Mechanical and
Astronautical Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

In the interest of enhancing the capabilities of autonomous underwater vehicles used in US Naval Operations, controlling vehicle position to follow depth contours presents exciting potential for navigation. Use of a contour tracking control algorithm in lieu of preprogrammed waypoint navigation offers distinct advantages within new challenges. The difficult nature of this problem lies in the non-trivial connection between the necessary corrective action and the feedback error used in traditional control methods. Stated simply, modern vehicle control algorithms separate horizontal and vertical plane navigation. The autonomous vehicle senses heading error and applies rudder to steer the vehicle to a desired heading. Simultaneously, the vehicle might sense altitude and apply stern plane angles to maintain a safe height above ground. This thesis research examines the new problem of sensing depth and altitude in the vertical plane while steering the vehicle horizontally to find a specified bathymetry contour. While more remains to understand, this research proves the existence of a solution and suggests similar approaches may facilitate tying vehicle navigation to other indirect sensors. This thesis presents two contour tracking control algorithms and examines the performance of each by simulating the response of the REMUS underwater vehicle to ideal and real-world bathymetry models.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to thank my thesis advisor, Distinguished Professor Anthony J. Healey, for lending me his expertise, guidance, help, and most of all, time. While I learned much from my efforts, and managed to enjoy my struggles, I would not be able to claim success without his help.

I would particularly like to thank my wife, Bobbi Van Reet, for supporting and encouraging me. Not only has all her work given me the time to devote to this endeavor, but also she still manages to push me to succeed and to do so in a timely manner.

Finally, I would like to thank all my family and friends whose support and understanding has helped to do everything thus far.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. BACKGROUND

Unmanned vehicles provide both civilian and military users with greater access to the varied environments on this planet and beyond. Unmanned vehicles typically enter areas that present conditions impossible for humans to endure, that pose a risk to human life greater than the intended benefit, or that are simply to expensive to reach with a similarly equipped manned vehicle. In the air, on land, or in the sea, specific missions are better suited to certain vehicle types and certain control programs. If there is one absolute truth regarding unmanned vehicles, it is that no single platform will be able to fulfill every possible mission requirement. Though multiple machines will be necessary, designing each robot to perform in as many situations as possible not only keeps vehicle programs simple and cost-effective, but also increases survivability when encountering unpredictable events.

Unmanned Underwater Vehicles (UUVs) employed in naval applications fall into two basic classifications, Remotely Operated Vehicles (ROVs) and Autonomous Underwater Vehicles (AUVs). While neither type carries people onboard, ROVs still require manned control. As the name implies, ROVs must receive continuous control input, or piloting, from a remote user making decisions based on output from the vehicle's sensors, usually video feed. As opposed to air and land vehicles that can easily receive radio wave signals, vehicles underwater cannot yet transmit clear, high-speed signals through their medium over any appreciable distances. Due to this limitation, ROVs remain

connected to the host ship via physical tethers. These tethers have advantages by providing ample power supplies and large communications bandwidths. Conversely, tethering the vehicle limits the distance it can venture from the host ship, and remote operation, while protecting personnel from hostile environments, does not free individuals to perform other tasks or significantly reduce man-hours.

AUVs essentially present opposing capabilities to those of ROVs. AUVs are self-contained units that run off control programs stored in onboard memory. They may communicate with their host as well as other vehicles but not continuously. AUVs execute their stored missions without constant attention from an operator and work without interruption over any distance or duration allowed by onboard battery stores. To maximize battery stores, AUVs must be designed efficiently. This design extends beyond part selection to using vehicle motion efficiently. AUVs manipulate hydrodynamic forces rather than using thrusters as ROVs do, hence they are not suited for station keeping and do not easily stop or back track. AUV control design must anticipate actions and manage each on the fly in order to maximize performance based upon the capabilities of these platforms.

B. MOTIVATION AND RELEVANCE

ForceNet objectives, part of the United States Navy's doctrines set forth by the Chief of Naval Operations in SeaPower 21, dictate the need for collecting vast amounts of information from a network of all available sensing platforms. Assembling this data into a single intelligence model accessible by all combat platforms ensures that every

fighting member enjoys the full information superiority of the entire force (Clark, 2002). AUVs primarily support US Naval Operations by performing surveillance and reconnaissance missions such as beach surveys, ocean sampling, and covert operations in the littoral area.

AUV research began in the 1960s, with the first prototypes emerging in the 1980s (Blidberg, 2001). Present AUV capabilities allow vehicles to follow pre-planned flight paths in order to execute specific mission objectives. Most AUVs navigate by tracking paths constructed linearly between pre-planned waypoints. Waypoint navigation essentially uses Line of Sight guidance to constantly point the vehicle at its present position directly toward the desired waypoint. Other algorithms, such as Cross Track Error, compensate for factors such as steady currents. In order to make the vehicle follow the tracking algorithm's commands, any of a number of control methods may be used to achieve the desired tracking performance. Waypoint tracking algorithms are well developed and the control methods are proven.

One basic limitation of waypoint navigation is the requirement for enough advance knowledge to appropriately locate waypoints during mission planning. The idea of having advance knowledge of an area negates the reason for conducting surveillance of the area. As a minor note, when paths require following curves, tracking straight lines between waypoints is not the most efficient way. Though presently useful, AUVs need to operate successfully while reacting to unknown conditions sensed in real-time in order allow more robust platforms to more effectively fulfill various mission requirements without the need for separate

vehicles. Current work hopes to render vehicles fully autonomous in unknown surroundings and centers around adding obstacle avoidance algorithms to planned path navigation. An example of such work is shown in figure 1.

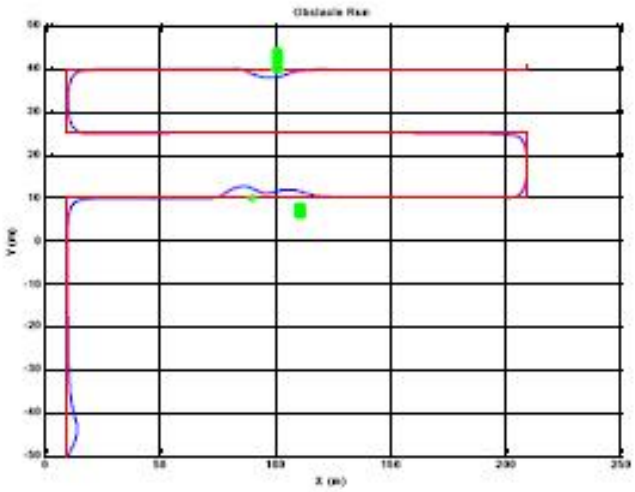

Figure 1. Waypoint Navigation with Obstacle Avoidance (From: Fodrea and Healey, 2003)

The motivation for this thesis is to find an alternative to waypoint navigation that satisfies the need for adapting to the unknown. Navigation is accomplished by tracking a feature of the ocean floor, specifically a specified contour of constant depth. Tracking requires steering the vehicle in the horizontal plane to follow the depth contour. Though an alternative to waypoint navigation, contour tracking would also benefit from similar obstacle avoidance capabilities. Waypoint navigation and contour tracking are each suited to different mission structures, and some missions may benefit from the use of both in conjunction.

C. SENSOR BASED CONTROL

Contour tracking problems belong to a type on control theory known as Sensor Based Control. Other Sensor Based Control problems include tracking plumes in the ocean, following ocean temperature gradients, or steering towards the location of greatest communications signal strength. The name "Sensor Based Control" seems somewhat misleading, as all control theory utilizes feedback of either sensor outputs or errors derived from these outputs. Perhaps the concept might more appropriately be called "Indirect Sensor Control." The distinction comes from an increased complexity in the control method due to the nature of the feedback signal and its indirect relationship with the corrective control action.

As an example, steering to a commanded heading is a comparably simpler problem because the feedback loop ties output from a compass, which measures heading, to the heading command. When the heading error indicates that the vehicle is pointed left of the desired heading, a right turn is clearly in order, and a right rudder command can be easily manipulated mathematically.

1. Contour Tracking

As will be discussed later, autonomous underwater vehicle controls are simplified by separating the effects of motion in the horizontal plane from those in the vertical plane. The previous example of tracking a commanded heading falls into the simpler category largely because both the control model and the sensor output lie in the same plane of motion. In tracking a depth contour, the problem design requires steering the vehicle, a horizontal

plane control, based upon sensors measuring water column depth, a vertical plane output.

Compare this problem to the previous example. Rudder action and vehicle speed determines position at any point in time. At a given position, water depth is determined by the ocean floor's geometry. While the ocean floor is assumed to remain fixed for relevant periods of time, the location of the contour related to the vehicle's turns may change constantly. Though the vehicle may sit to the right of the commanded depth contour, turning left toward the contour's local position may not be ideal if the contour curves right towards the vehicle's position close ahead. The problem is further complicated because the vehicle knows neither future trends of the contour nor local trends when only single sensor values are available, as is the case with the REMUS vehicle.

2. Related Research

Currently, research is underway to control vehicles to track numerically computed gradients for use in following zones of constant temperature in the ocean in three dimensions. This problem relates directly to tracking depth contours in that the gradient at a point in the field forms the basis for the direction and/or magnitude of the control command. Contours of constant value are by definition orthogonal to the gradient, which points in the direction of steepest ascent, making the tracking of either feature mathematically related to the other.

Professor Naomi Leonard of Princeton University has co-authored much of the related research in this field. Particularly relevant to this discussion are efforts

enabling multiple AUVs to climb (Ogren, Fiorelli, and Leonard, 2004) and descend (Moreau, Bachmayer, and Leonard, 2003) gradients, and for the first time this year to use multiple AUVs to track and plot temperature contours (Zhang and Leonard, 2005).

The initial work by Moreau, Bachmayer and Leonard in 2003 focused on tracking the direction of the negative gradient. When each vehicle has enough sensors to measure the full gradient, the closed-loop system becomes Lagrangian. This research allows the calculation of the gradient with only a single sensor vehicle; however, multiple sensors are still required through the use of multiple single sensor vehicles acting together in a single formation. In 2004, Ogren, Fiorelli, and Leonard worked on the related problem of tracking the direction of the positive gradient. These efforts still use multiple single sensor vehicles to construct a single multi-sensor formation; however, in this revision the formation can be reconfigured on the fly without hindering tracking ability. Finally, research conducted by Zhang and Leonard during the same time period as this thesis allows the vehicles to track contours and form contour plots based on collected data. The formation still tracks as a single unit optimally shaped to minimize errors in the gradient calculation. The group may consist of as few as four single sensor vehicles, but tracking still requires the use of multiple sensors and a full numerical gradient calculation.

The related research in this field bears relevance to the efforts of this thesis, yet the methods used differ distinctly. The algorithms used for REMUS focus on

approximating gradients from data at the current and previous locations allowing a single sensor vehicle to successfully track contours without performing numerical gradient calculations.

D. SCOPE OF THESIS

1. Contour Tracking Control

As previously mentioned, the fundamental goal of this thesis is to develop control algorithms that successfully allow AUVs to track constant depth contours. The lessons learned from this work provide insight into the general problems of Sensor Based Control. While the control algorithms developed are applicable to all AUVs that move by manipulating hydrodynamic forces via rudders and planes, these algorithms are specifically tailored to the REMUS vehicle with only currently available sensors in mind. For this reason, a brief discussion of the REMUS vehicle, as used in the Naval Postgraduate School's Center for Autonomous Underwater Vehicle Research, is in order.

2. The REMUS Vehicle

Remote Environmental Monitoring Units (REMUS) are low-cost, lightweight autonomous underwater vehicles originally developed by the Oceanographic Systems Laboratory at Woods Hole Oceanographic Institution. The vehicles operate with a laptop computer and simplify launching and recovery operations due to their compact size and weight. As a package, REMUS incorporates a wide range of onboard sensors and includes an upgradeable payload for the addition of unique sensor packages (Hurst). All of these factors make

REMUS an attractive platform for US Navy missions. Furthermore, research tailored to the REMUS platform has the distinct advantage of being directly relevant to a vehicle already in production and presently deployed by US Navy vessels.

REMUS generally deploys in the Very Shallow Water zone defined by water depths ranging between 40 and 100 feet (Fodrea, 2002). In standard use, REMUS can run from 8 up to 20 hours when traveling at 5 and 3 knots, respectively (Hurst). Figure 2 shows REMUS in a basic configuration along with its impact resistant case, which allows it to be carried or shipped as conventional baggage. Table 1 lists more detailed characteristics of REMUS physical features and functional capabilities.

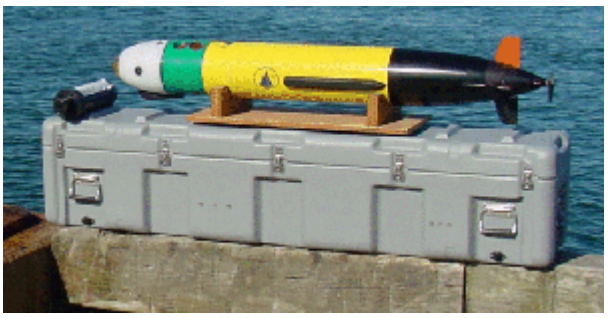

Figure 2. REMUS Vehicle (From: Hurst)

Of the many sensors already carried by REMUS, two are relevant to this thesis. REMUS simultaneously senses its depth under the surface of the water and uses its RDI Doppler sonar to detect its altitude above the ocean floor. For tracking depth contours, summing these two values provides the water column depth at the present position, which also reduces the output from two sensors to a single value useful for feedback. As stated in the previous section on related research, as few as four sensors can produce accurate gradients calculations. The difficulty of

gradient tracking with only single sensor feedback warrants the efforts of this thesis to allow existing REMUS vehicles to perform contour tracking without requiring 4 additional expensive, power consuming sensors.

Table 1. Characteristics of REMUS AUV

PHYSICAL/FUNCTIONAL AREA	CHARACTERISTIC
Vehicle Diameter	7.5 in
Vehicle Length	62 in
Weight in Air	80 lbs
External Ballast Weight	2.2 lbs
Operating Depth Range	10 ft to 66 ft
Transit Depth Limits	328 ft
Typical Search Area	875 yds X 1093 yds
Typical Transponder Range	1640 yds
Operational Temperature Range	+32F to +100F
Speed Range	0.5 knots to 5.6 knots
Maximum Operating Water Current	2 knots
Maximum Operating Sea State	Sea State 2
Battery	1 kW-hr internally rechargeable Lithium-ion
Endurance	20 hours at 3 knots; 9 hours at 5 knots

E. THESIS STRUCTURE

The objective of this research is to develop a stable, robust algorithm for tracking contours of constant water depth. The algorithm is developed to suit the REMUS autonomous underwater vehicle and is tested by simulating the motions of REMUS in a virtual ocean environment. Chapter II explains the necessary motion and ocean models. Chapter III discusses an attempt to use logic feedback to directly control the vehicle's rudder. Chapter IV details the method of using logic to command heading to a stable steering controller. Finally, Chapter V provides general conclusions and recommendations for future work.

II. VEHICLE MODEL AND SIMULATION

A. INTRODUCTION

The purpose of this thesis is not to derive equations modeling underwater vehicle motion, nor to calculate the specific hydrodynamic forces experienced by the REMUS vehicle. Both issues have been adequately addressed prior to this research. The notes from (Healey, 2003) contain complete derivations of the equations in this section, and the thesis by (Fodrea, 2002) contains an additional discussion. The thesis by (Prestero, 2001) calculates the precise values of the hydrodynamic coefficients needed to model a REMUS vehicle with these equations. Though full derivations are not part of this thesis, in order to adequately understand the work presented, a discussion of the relevant equations and assumptions is in order. Furthermore, the methods used to numerically simulate vehicle motion and ocean floor data are included to enhance the reader's comprehension.

B. MATHEMATICAL VEHICLE MODEL

1. Equations of Motion

Vehicle motion is fully modeled by six equations of motion that relate force inputs to resulting motions in three translational and three rotational degrees of freedom. Three reasonable assumptions must be made in order to represent motion by these six equations. The first assumes that the vehicle behaves as a rigid body despite accelerations. The second assumes that acceleration terms can neglect the effects of the earth's

sidereal rate. The third assumption considers only inertial and gravitational forces resulting from thrust, hydrostatic effects, and hydrodynamic lift and drag. The six equations describe surge, sway, heave, roll, pitch and yaw motions and are shown, respectively, in equations 1 through 6 below.

SURGE EQUATION

$$m[\dot{u}_r - v_r r + w_r q - x_G(q^2 + r^2) + y_G(pq - \dot{r}) + z_G(pr + \dot{q})] + (W - B)\sin\theta = X_f \quad (1)$$

SWAY EQUATION

$$m[\dot{v}_r + u_r r - w_r p + x_G(pq + \dot{r}) - y_G(p^2 + r^2) + z_G(qr - \dot{p})] - (W - B)\cos\theta\sin\phi = Y_f \quad (2)$$

HEAVE EQUATION

$$m[\dot{w}_r - u_r q + v_r p + x_G(pr - \dot{q}) + y_G(qr + \dot{p}) - z_G(p^2 + q^2)] - (W - B)\cos\theta\cos\phi = Z_f \quad (3)$$

ROLL EQUATION

$$I_x\dot{p} + (I_z - I_y)qr + I_{xy}(pr - \dot{q}) - I_{yz}(q^2 - r^2) - I_{xz}(pq + \dot{r}) + m[y_G(\dot{w} - u_r q + v_r p) - z_G(\dot{v}_r + u_r r - w_r p)] - (y_G W - y_B B)\cos\theta\cos\phi + (z_G W - z_B B)\cos\theta\sin\phi = K_f \quad (4)$$

PITCH EQUATION

$$I_y\dot{q} + (I_x - I_z)pr - I_{xy}(qr + \dot{p}) + I_{yz}(pq - \dot{r}) + I_{xz}(p^2 - r^2) - m[x_G(\dot{w}_r - u_r q + v_r p) - z_G(\dot{u}_r - v_r r + w_r q)] + (x_G W - x_B B)\cos\theta\cos\phi + (z_G W - z_B B)\sin\theta = M_f \quad (5)$$

YAW EQUATION

$$I_z\dot{r} + (I_y - I_x)pq - I_{xy}(p^2 - q^2) - I_{yz}(pr + \dot{q}) + I_{xz}(qr - \dot{p}) + m[x_G(\dot{v}_r + u_r r - w_r p) - y_G(\dot{u}_r - v_r r + w_r q)] - (x_G W - x_B B)\cos\theta\sin\phi - (y_G W - y_B B)\sin\theta = N_f \quad (6)$$

One additional assumption greatly simplifies control calculations. When developing AUV controls, motion in the horizontal plane is separated from that in the vertical plane. Although designed around two-dimensional control planes, three-dimensional vehicle control can be achieved by simply running the horizontal and vertical control algorithms simultaneously.

For steering in the horizontal plane, only the surge, sway, and yaw equations are important, reducing a six-dimensional problem to just three dimensions. Assuming constant speed only in the forward direction, and reiterating that all vertical plane motions are ignored equations 7, 8, and 9 show the simplified forms of the three horizontal plane equations of motion. Equations 10 and 11 compute changes in the vehicle's Cartesian horizontal plane position based on linear velocity and angular turn rates. For this model, vehicle speed is assumed constant in the forward direction ($u_r = U_0$) and zero current is considered ($U_{cx} = U_{cy} = 0$).

$$m\dot{v}_r = -mU_0 r + \Delta Y_f(t) \tag{7}$$

$$I_{zz}\dot{r} = \Delta N_f(t) \tag{8}$$

$$\dot{\psi} = r \tag{9}$$

$$\dot{X} = U_0 \cos\psi - v_r \sin\psi + U_{cx} \tag{10}$$

$$\dot{Y} = U_0 \sin\psi + v_r \cos\psi + U_{cy} \tag{11}$$

Finally, in order to model the specific behavior of the REMUS, or any other, vehicle submerged in water and responding to inputs from control surfaces, the associated

linearized fluid forces are represented in the equations of motion by coefficient terms multiplied with the appropriate individual motions or control surface angles. The final equations of motion, with forcing effects, are detailed below in matrix equation 12. Equations 10 and 11 for Cartesian position remain unchanged. As previously stated, the equations for this model were obtained from Distinguished Professor Anthony Healey's derivations as found in (Healey, 2003).

$$\begin{bmatrix} m - Y_{\dot{v}_r} & -Y_{\dot{r}} & 0 \\ -N_{\dot{v}_r} & I_{zz} - N_{\dot{r}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{v}_r \\ \dot{r} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} Y_{v_r} & Y_r - mU_0 & 0 \\ N_{v_r} & N_r & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_r \\ r \\ \psi \end{bmatrix} + \begin{bmatrix} Y_\delta \\ N_\delta \\ 0 \end{bmatrix} \delta_r(t) \qquad (12)$$

2. Hydrodynamic Coefficients

The coefficient terms in the previous equations of motion are called hydrodynamic coefficients, and they represent the magnitude of the effects of various propulsive and maneuvering forces on vehicle motion, assuming that the effects are linearly related. The hydrodynamic terms above, which are relevant to horizontal plane motion, represent the following forces:

$Y_{\dot{v}_r}$ = coefficient of added mass in sway
$Y_{\dot{r}}$ = coefficient of added mass in yaw
Y_{v_r} = coefficient of sway force induced by side slip
Y_r = coefficient of sway force induced by yaw
$N_{\dot{v}_r}$ = coefficient of mass moment of inertia in sway
$N_{\dot{r}}$ = coefficient of mass moment of inertia in yaw
N_{v_r} = coefficient of sway moment from side slip
N_r = coefficient of sway moment from yaw
Y_δ = coefficient of rudder moment
N_δ = coefficient of rudder moment

Table 2 gives actual values for the hydrodynamic coefficients that accurately model the REMUS vehicle's maneuvering characteristics. These numerical values were obtained from research found in the thesis by (Prestero, 2001). As an exception, LT Lynn Fodrea modified the rudder moment coefficient values after observing that the Prestero model did not agree with experimental results (Fodrea, 2002).

Table 2. REMUS Hydrodynamic Coefficients for Equations of Motion in the Horizontal Plane

$Y_{\dot{v}_r}$	-3.55e01 kg
$Y_{\dot{r}}$	1.93 kg m/rad
Y_{v_r}	-6.66e01 kg/s (Same as Zw)
Y_r	2.2 kg m/s (Same as Zq)
$N_{\dot{v}_r}$	1.93 kg m
$N_{\dot{r}}$	-4.88 kg m^2/rad
N_{v_r}	-4.47 kg m/s
N_r	-6.87 kg m^2/s (Same as Mq)
Y_δ	-3.46e01/3.5 kg m/s^2
N_δ	5.06e01/3.5 kg m/s^2

C. SIMULATION

1. Vehicle Motion

The equations of motion are ordinary differential equations. Simulation of vehicle motion results from integrating the equations over time and adding a unique initial condition. This simple mathematical concept computes the value of every system state at any moment in the integrated time. Numerical methods are employed to integrate of the differential equations. While any numerical integration method would work, the model in this thesis uses simple Euler integration for computer coding simplicity, and a sufficiently small time step assures reasonable accuracy in the numerical solution.

2. Ocean Bathymetry

Running the control simulation requires the creation of a virtual ocean environment. Three ocean models were developed for this thesis. Two models simulate straight and curved contours using depth data from simple first and second order equations, respectively. Creating bathymetry data from low order equations results in an ideally smooth ocean floor model. The ideal models serve as initial measures of the contour tracking algorithm's performance.

The final test of an algorithm's usefulness uses a virtual bathymetry model constructed from real-world, sampled data. Figure 3 shows the real-world ocean floor model used. The data in this model comes from actual REMUS sampling runs performed in Monterey Bay by the Naval Postgraduate School's Center for AUV Research. This section of ocean floor features generally straight contour

lines with local deviations and one significant dogleg turn. When using real data, the rougher nature of the floor presents a greater challenge for stability concerns.

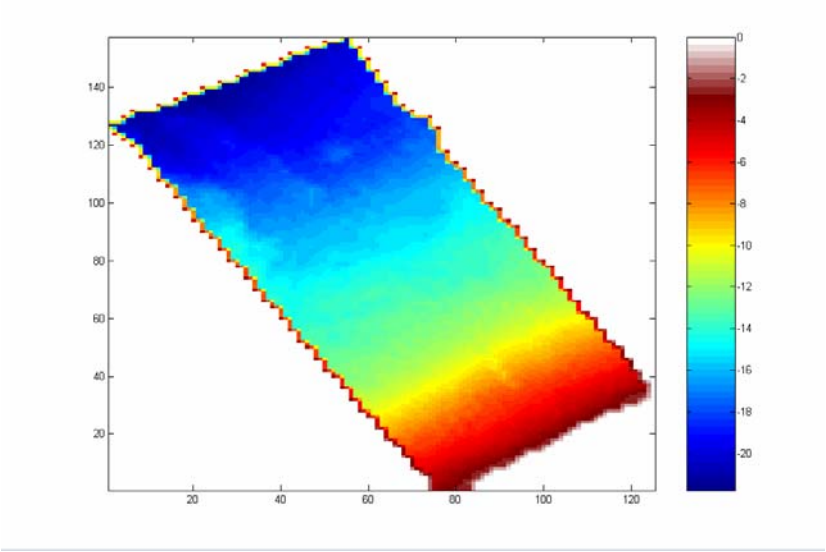

Figure 3. Discretized Bathymetry Data From Monterey Bay

Actual vehicles cover continuous ocean floor receiving discrete sensor feedback at the sampling rate. In creating a virtual model, the ocean floor becomes a discrete field. Though discrete sampling can also be simulated, placing two discretized signals in series compounds adverse effects, so continuous feedback of a discrete signal is used instead. The assumption has been made that this switch does not significantly affect the performance of the simulations, and it is more than reasonable to assume that it has no effect on the stability of the algorithms. Appendix A contains MATLAB code for Bathymetry simulation. The color scaling in these figures and the Monterey Bay bathymetry data are saved in files attached to the electronic version of this thesis. The code in Appendix E simulates sensor output as a single water column depth value.

THIS PAGE INTENTIONALLY LEFT BLANK

III. DIRECT, LOGIC-BASED RUDDER CONTROL

A. INTRODUCTION

In order to successfully construct an autonomous control, one must create a stable, closed-loop system using control feedback. When considering this contour tracking problem, constructing this feedback loop presents such a challenge because the vertical plane depth readings cannot be directly converted to horizontal steering controls through traditional mathematical relations, such as constant feedback gains. Having multiple depth readings at a given time provides enough information to mathematically compute the gradient of the water column field. Gradient computation mathematically links the depth readings to a heading, which is exactly the form of feedback suited to horizontal steering control. Without having multiple sensors available, or when the gradient calculations are computationally burdensome, single depth readings simply cannot relate to steering commands in any similar way.

The first attempt to close the loop with a single depth sensor uses logic states to determine control actions based on certain conditions in the depth field along the vehicle's path. Although the approach theoretically overcomes the feedback obstacle, the algorithm is ultimately unstable when tested with real-world data, where roughness in the real ocean bottom amounts to noise in the depth sensor signal. With direct logic control, relatively small noise levels result in large control requirements and large motions, which by definition is unstable. The results of this algorithm will be presented only briefly because the method did not ultimately succeed; however, it

merits discussion because its limitations hold valuable lessons and the method's success with ideal data highlights elements of the algorithm that may benefit future work.

B. DESIGN

1. Logic Feedback

Logic feedback selects set control actions from the condition of particular states, rather than mathematically relating the actions to the states. The state of the vehicle's path through the depth field separates into two parts. The current depth error can sufficiently tell the vehicle which way it must turn to reach the commanded contour. More information is needed because once the vehicle points generally toward the contour, further turning would cause the vehicle either to reverse direction or to circle indefinitely without reaching the contour. Clearly a vehicle sensing shallow water should initially turn away from shore, and once the vehicle is moving toward deeper water, it can continue forward without turning.

The additional information needed by the algorithm is the current trend in the depth error. The trend essentially replaces a numerically calculated gradient with a very general approximation. The trend state comes from comparing the current depth error with previous depth errors held in memory to determine whether the vehicle is moving into deeper or shallower water. Figure 4 visually represents this direct logic control algorithm in block diagram form. Table three details the logic-based relationship between three possible control actions the states defined by depth error and trend.

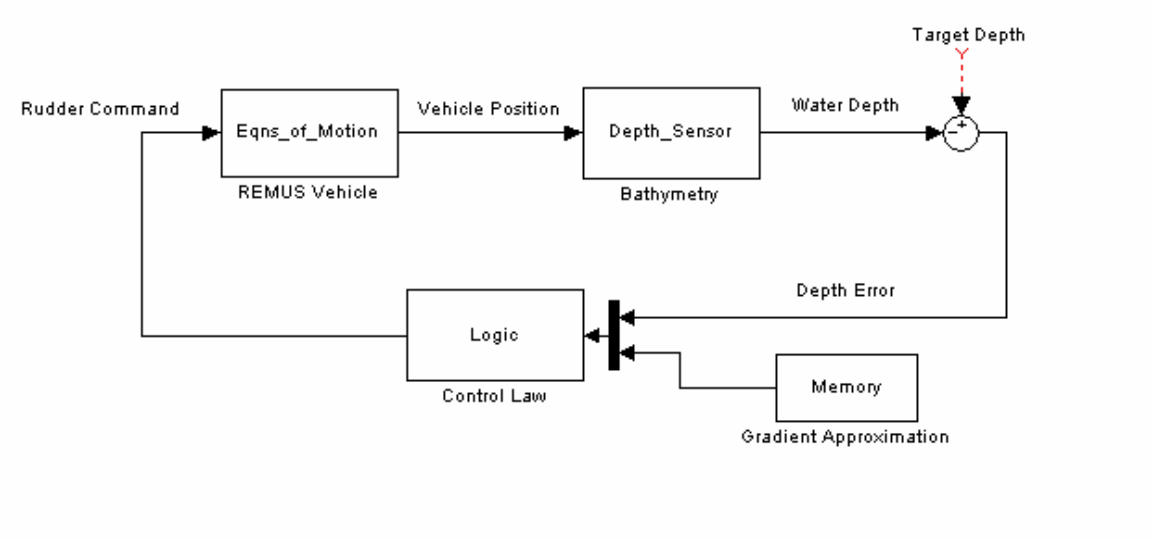

Figure 4. Block Diagram: Logic Feedback

Table 3. Logic States Required for Direct Rudder Control

Logic State	Vehicle's Condition	Required Control Action
0	On Contour	Rudder Amidships
1	Too Deep & Getting Deeper	Turn Towards Shore
2	Too Deep but Getting Shallower	Rudder Amidships
3	Too Shallow & Getting Shallower	Turn Away from Shore
4	Too Shallow but Getting Deeper	Rudder Amidships

2. Gradient Approximation

a. Linear Gradient Approximation (LGA)

With the depth error easily calculated and the necessary logic states established, all that remains to implement this control algorithm is specifying a method by which the error trend is calculated. The simplest trend calculation compares the current and last depth errors and assumes the trend is exactly the difference between the

two. Trends computed with this method are linear approximations, which should have reasonable accuracy assuming that the elapsed time between the depth error values used is sufficiently short. The MATLAB code attached in Appendix B simulates direct logic-based rudder control using a linear gradient approximation.

With the trend calculated according to this linear gradient approximation (LGA) method, the algorithm successfully tracks ideal depth contours whether straight or curved. Figure 5 shows the algorithm's performance when tracking straight contours, and figure 6 shows the same method tracking a circular contour.

Throughout this thesis, the figures of tracking performance show three pieces of information. The central image plots the vehicle's path through the virtual water column field. The heavy black line indicates the vehicle's path and labels clarify the start and end points of the run simulation. The bottom left image shows the water column depth history at every moment in time during simulation. This information has use in determining the vehicle's deviation from the commanded contour and also shows the discrete nature of the depth feedback. Finally, the bottom right image shows the control command history during simulation. For the direct logic control simulations, the control history shows the logic states, which are related to rudder commands as previously specified in table 3.

The direct logic control run using the linear gradient approximation trend, simulated in figure 5, shows that the linear method is well suited to straight-line contours. In all straight-line simulations, the tracking control commands constant 15-meter depth. In this run, the vehicle begins in water just slightly deeper than the command with an initial heading 10 degrees towards deeper water. The vehicle turns toward the desired contour, and tracks the remainder of the run with relatively little control action and depth error.

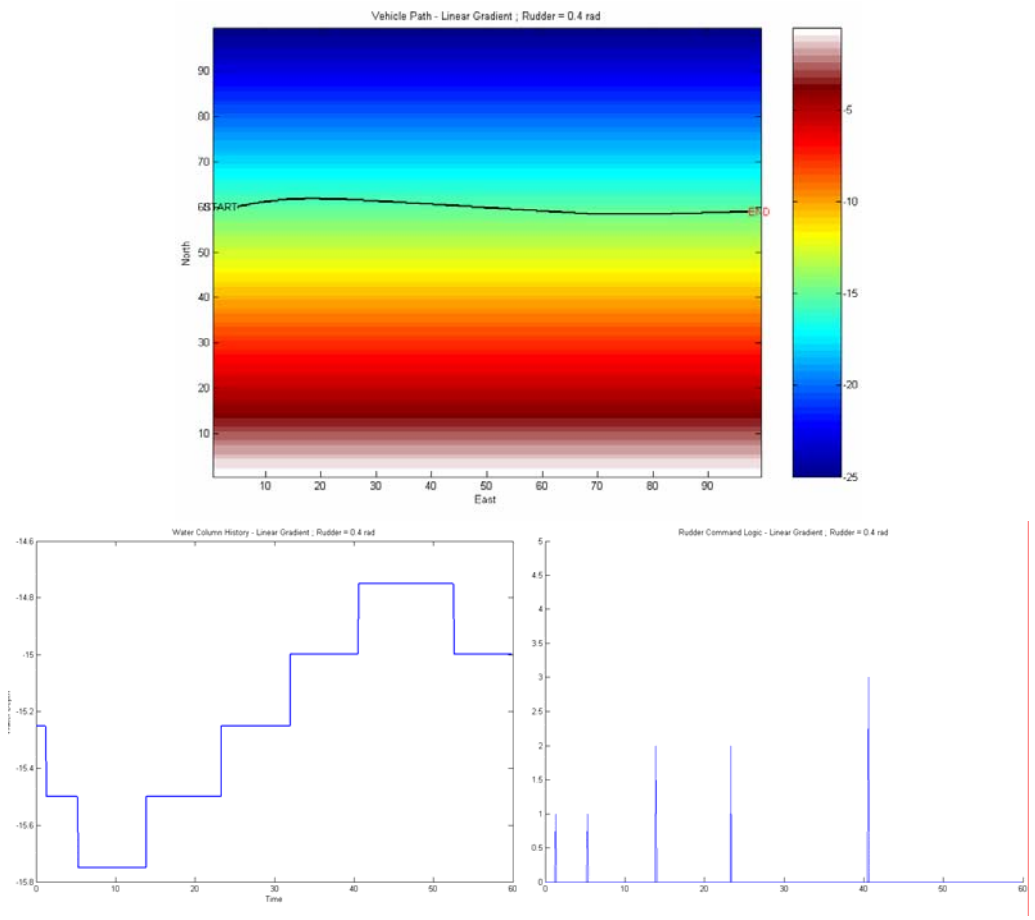

Figure 5. Straight Contour Response: Direct Control, LGA

Figure 6 shows the tracking response over curved contours. In all curved contour simulations, the tracking control commands 10-meter water depth in order to track the longest path through this virtual environment. The vehicle starts in water 1 meter too shallow, again with the initial heading not parallel to the local direction of the contour. Tracking the curve requires significantly increased control action, and the inefficient sinusoidal path results from the bang/bang action of logic control. Bang/bang means the control action is full on even when feedback errors are small. This type of control does not eliminate steady state error causing inefficient tracking of ideal contours.

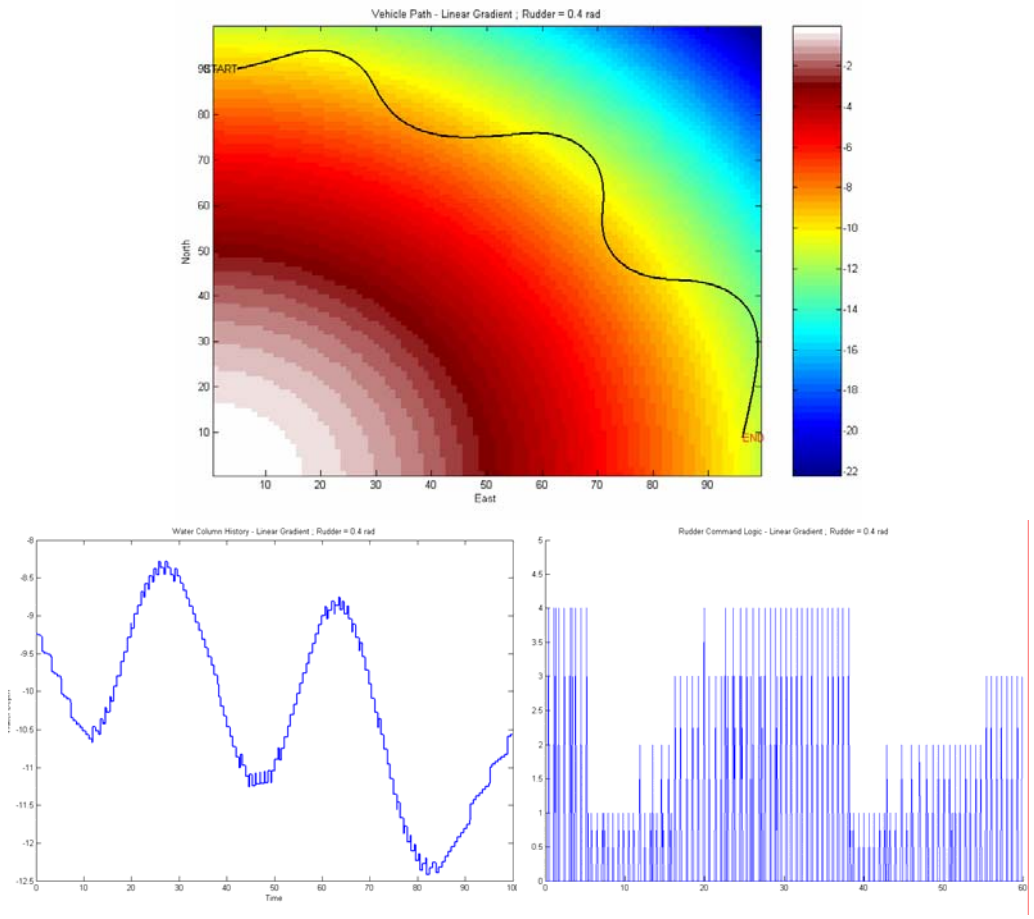

Figure 6. Curved Contour Response: Direct Control, LGA

Testing the control's response to real bathymetry data provides the most meaningful analysis. In all real-data simulations, the tracking control commands 15-meter water depth. Figure 7 shows that the LGA direct logic control fails to track with real data. Although relatively straight, tracking the contour is difficult because the floor slopes gently near the 15 meter depth contour resulting in noisy data as local floor roughness changes much faster than the trend of the general slope. Though the vehicle deviates only a meter from commanded depth, this relates to significant lateral deviations due to the gentle slope, and the error grows with increasing time.

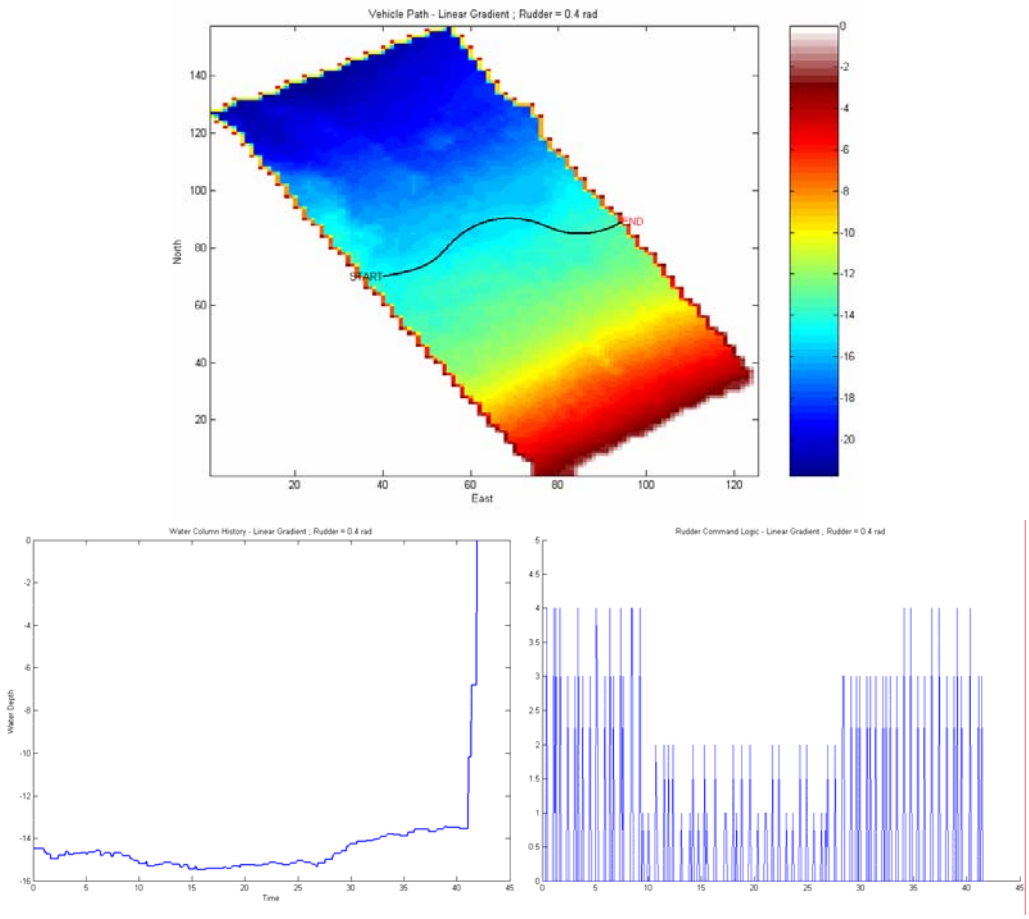

Figure 7. Real Contour Response: Direct Control, LGA

b. Estimated Local Gradient (ELG)

To check whether the unstable behavior of the direct logic control is due to the control's design or caused by representing the gradient by a simple linear trend approximation, an alternative trend approximation is developed. By definition, the linear approximation assumes the vehicle travels in a straight line between depth readings, and that the trend between these readings matches the trend ahead of the vehicle along its present heading. Clearly, this approximation does not account for any turning that occurs between the readings, or the fact that the vehicle may continue to turn ahead. Approximating the trend with the estimated local gradient (ELG) method seeks to account for the effects of turning.

The estimated local gradient considers not only the water depth readings at two locations, but also the vehicle's heading at those locations. Using differences in the two headings, if any, the method tries to distinguish the trend in the x-direction from the trend in the y-direction, which more appropriately approximates an actual gradient. The trend approximation is still first order, but the approximation is two-dimensional instead of one-dimensional. The MATLAB code attached in Appendix C simulates direct logic-based rudder control using the estimated local gradient approximation.

Figures 8 and 9 show that using the ELG approximation with direct rudder control slightly improves efficiency when tracking circular contours but actually hinders efficiency when tracking straight contours. Both figures prove that the direct logic control design tracks ideal data regardless of the approximation method. Figure

10 shows an unstable response when tracking real data, despite calculating trend with the new ELG approximation method.

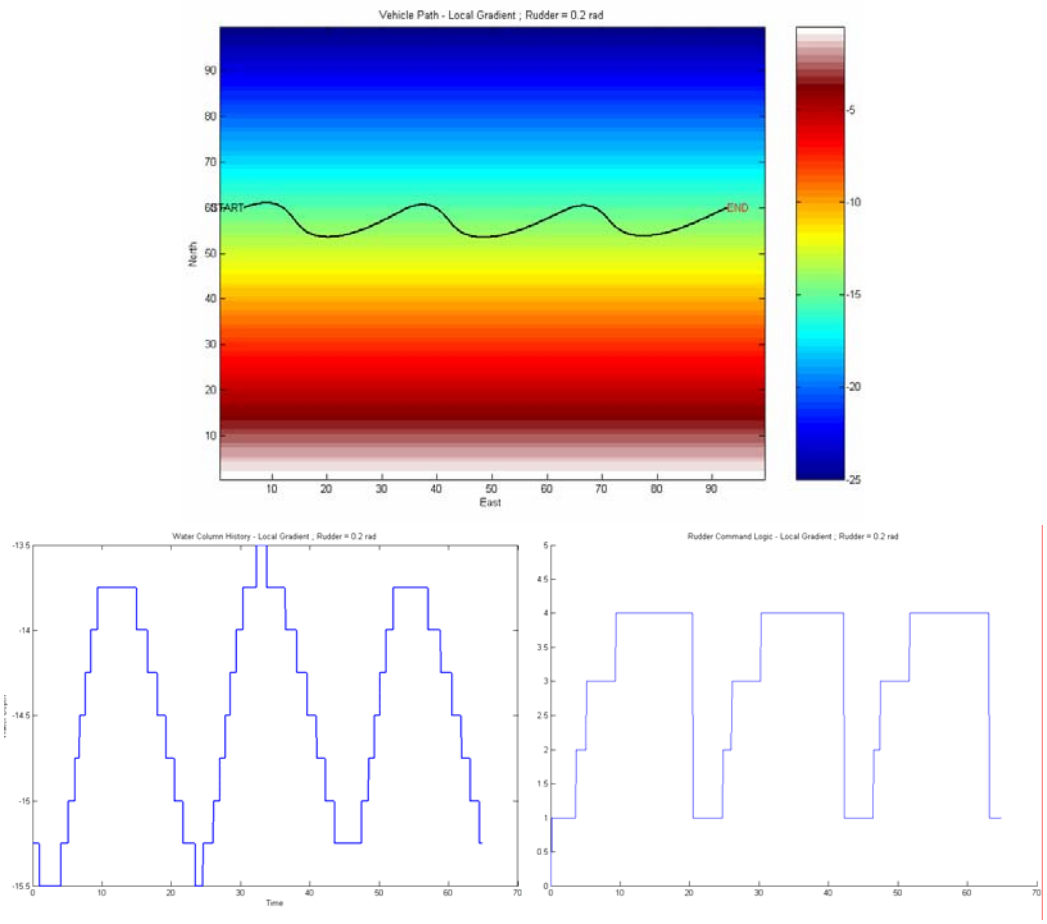

Figure 8. Straight Contour Response: Direct Control, ELG

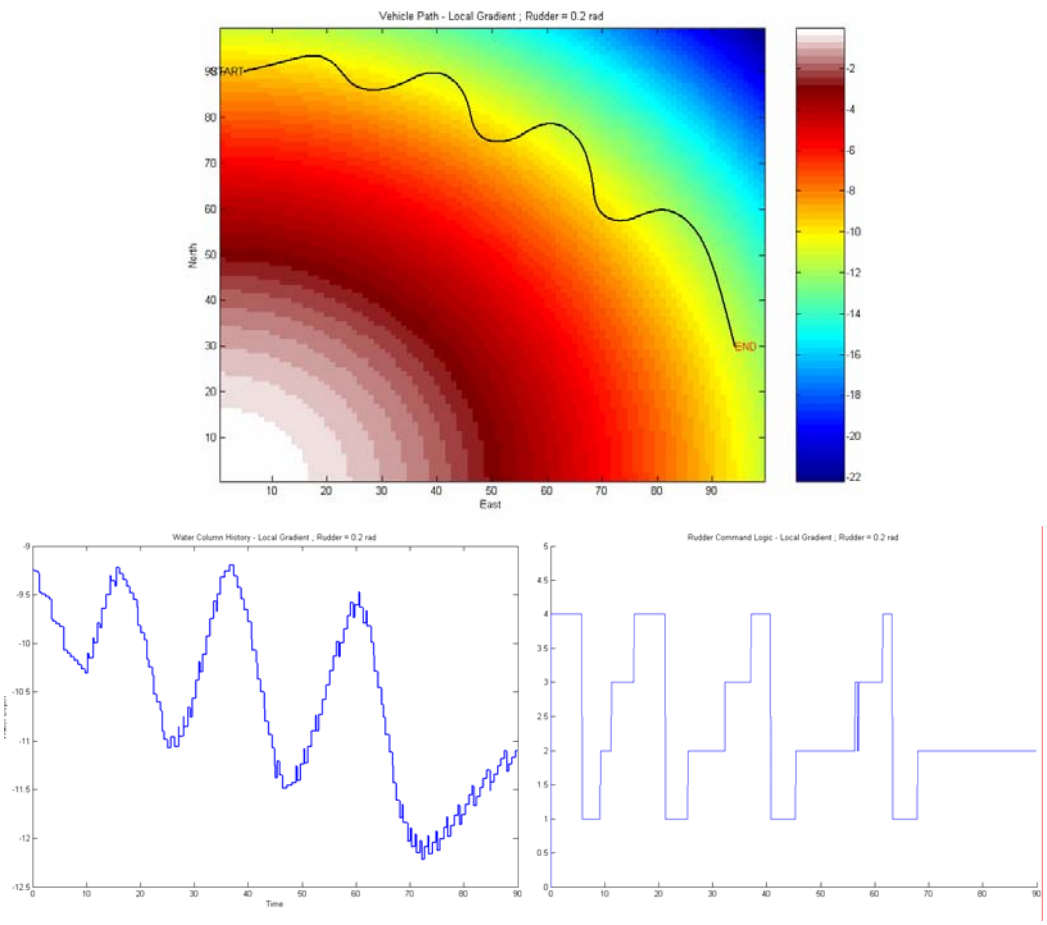

Figure 9. Curved Contour Response: Direct Control, ELG

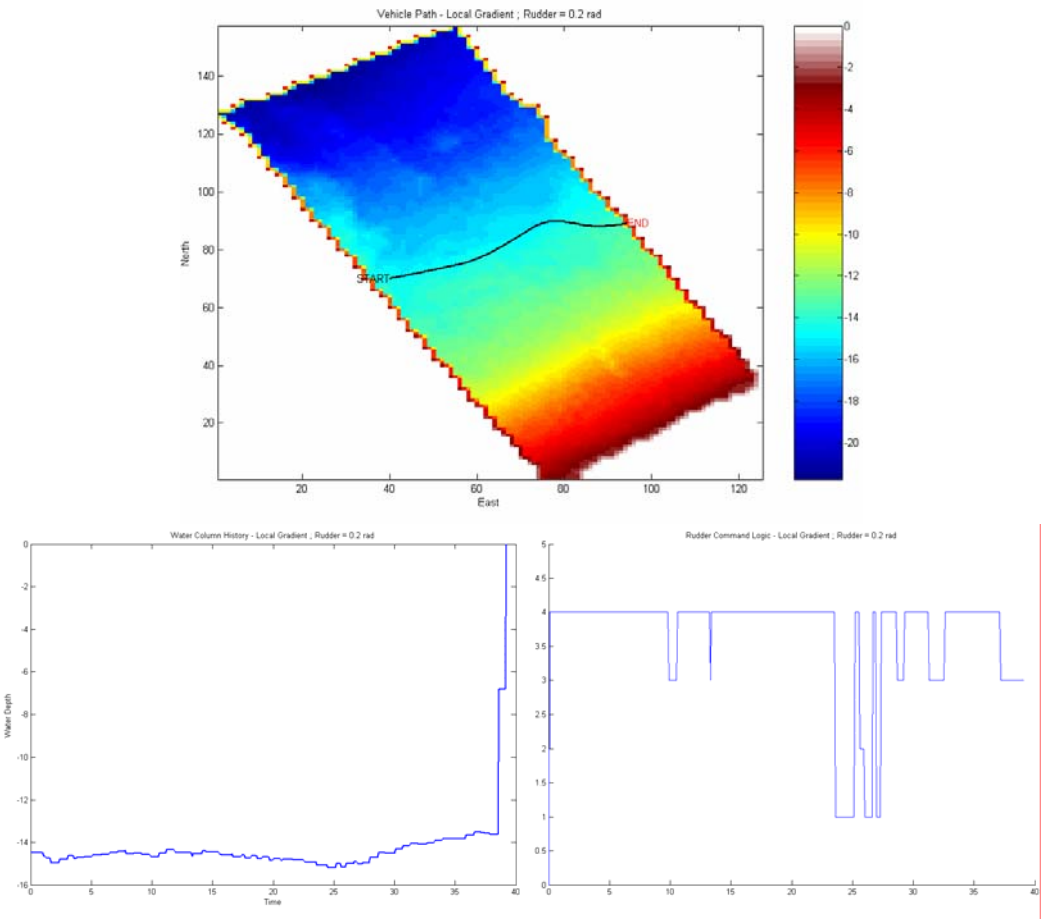

Figure 10. Real Contour Response: Direct Control, ELG

C. SUMMATION

It is appropriate to conclude that the direct-logic control design is responsible for the failure of the tracking control because it cannot track real data using either trend approximation. From this, it is clear that logic feedback alone is insufficient to create a stable closed-loop system when noise is present in the error feedback. What logic feedback offers is the ability to generate control commands from indirectly related sensor output. This observation leads to the next control algorithm, which feeds these commands to a separate stabilized closed-loop control.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. HEADING-STABILIZED LOGIC CONTROL

A. INTRODUCTION

After learning from the response of the first control attempt and reconsidering the problem at hand, it appears that direct logic-based rudder control attempts to "reinvent the wheel" so to speak, in terms of control theory. The primary objective in solving this problem is to relate indirect sensor output to control commands when traditionally mathematical relations are not practical. Inventing a new form of stable closed-loop control feedback need not be part of this research.

Traditional control theory has already solved the problem of autonomously steering a vehicle to track heading commands. The primary observation from the logic feedback attempt is that the model can solve the indirect relationship problem. Using logic to generate appropriate heading commands rather than rudder commands allows the vehicle to use existing steering autopilots to track these commanded headings. This greatly simplifies the control problem, and this method effectively generates heading without gradient computations.

B. DESIGN

1. Heading Command from Logic Feedback

The logic used in the algorithm is similar to the previous logic but simpler. When the vehicle is in water either too deep or shallow, it must point towards or away from shore, respectively. Because heading changes are used instead of rudder deflections, limiting heading changes

about a nominal heading value, rather than adding more logic states can solve the problem of the vehicle reversing direction. The logic states sufficient for stable control are listed in table 4.

Table 4. Logic Required for Heading-Stabilized Control

Logic State	Vehicle's Condition	Required Control Action
0	Too Deep	Turn Towards Shore
1	Too Shallow	Turn Away from Shore

The logic feedback creates an outer heading command loop surrounding the autopilot, which is a control loop issuing rudder commands. Figure 11 visually represents the control structure in block diagram form. The inner/outer loop structure is apparent, as are the unchanged vehicle and depth sensor models. The MATLAB code for this model is attached in Appendix D.

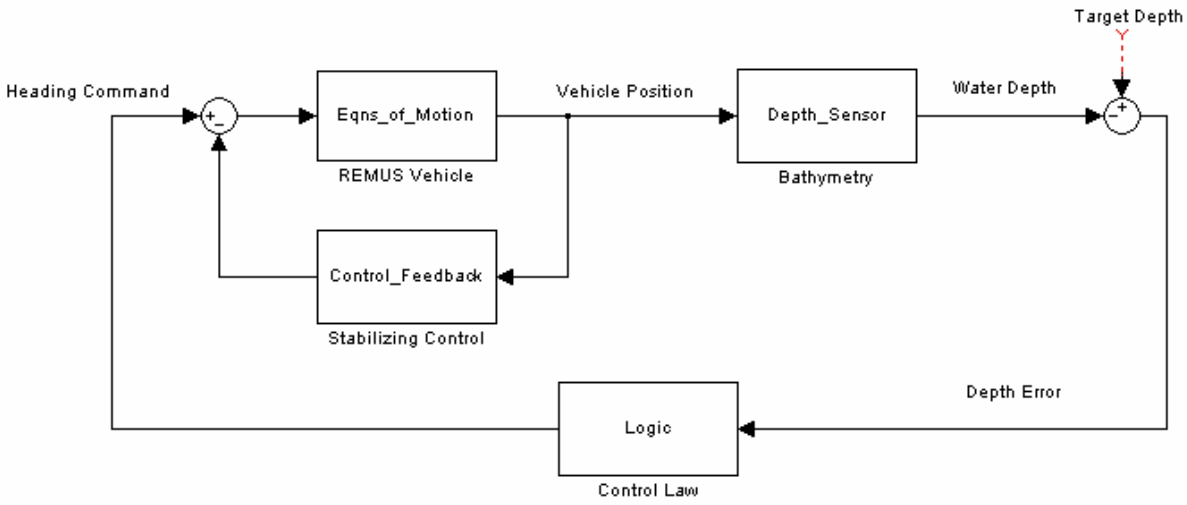

Figure 11. Block Diagram: Heading-Stabilized Logic Feedback

2. Steering Control

The steering controller receives heading commands within a search cone extending 57.3 degrees to either side of the nominal heading. Any stable control design could be used that can successfully track the heading command. The thesis by Fodrea, which was the source for the REMUS vehicle model used in this research, uses a sliding-mode controller for obstacle avoidance. For this research, state-feedback control is used because it is quite simple to implement in MATLAB code and it has desirable performance characteristics.

The state-feedback control law pulls the observable vehicle states, in this model v_r, r, and ψ, and multiplies each state by an individual gain calculated to place the closed-loop poles at locations design to meet specified performance goals. In this method, the control law calculates rudder commands mathematically related to each vehicle state.

As was the case in the previous design, logic feedback behaves as a bang/bang controller. Whatever the nature of the autopilot inner loop control, the overall vehicle motion should have sinusoidal steady-state error because it receives bang/bang controller commands.

C. SIMULATION RESULTS

1. Controller Performance

The vehicle motion model predicts REMUS behavior in the virtual environment that closely agrees with the previous design expectations. The algorithm's response to

ideal, straight line contours, figure 12, shows very quick acquisition of the target depth followed by tight, stable tracking. As predicted, the controller does not eliminate steady state error, and the oscillatory motion associated with the bang/bang control is evident. To note, all simulation figures for the remainder of this discussion present vehicle heading in the lower right image rather than the rudder commands previously shown.

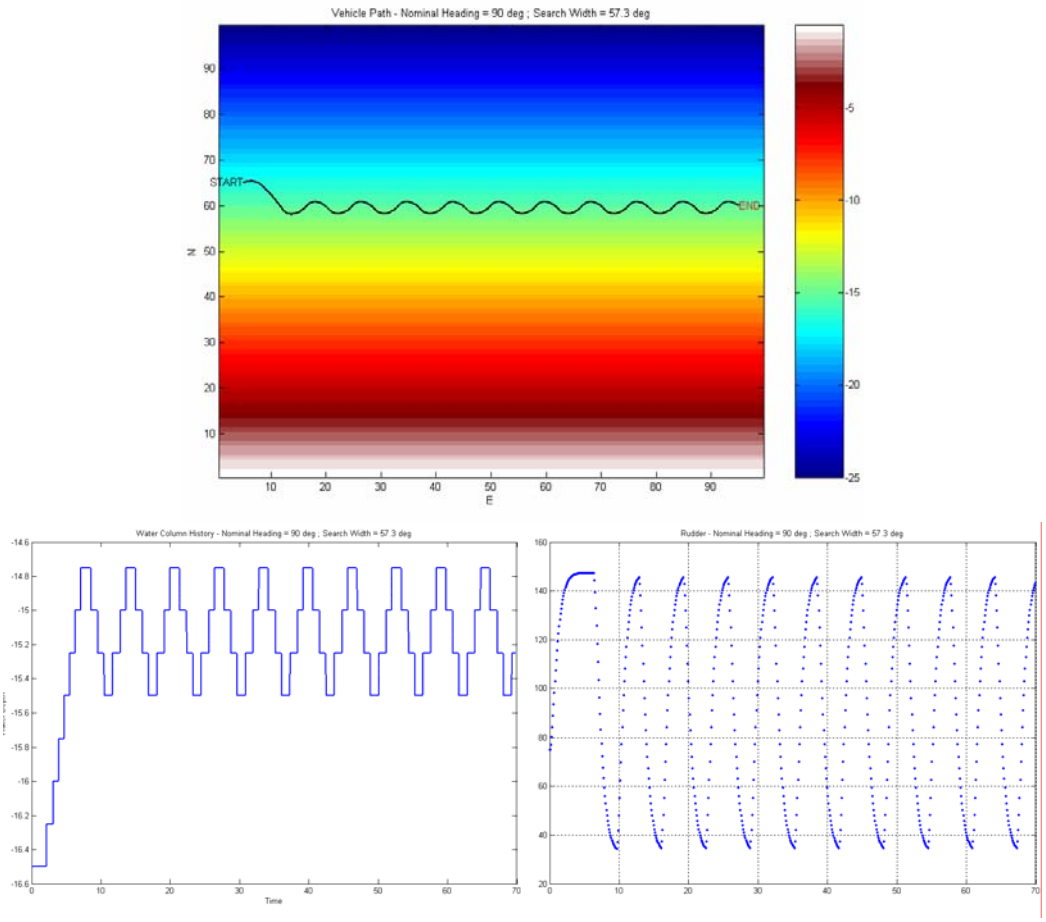

Figure 12. Straight Contour Response

To ensure the controller exhibits stable behavior in all situations, the vehicle's initial start point is perturbed significantly away from the target contour. As seen in figure 13, starting the vehicle in water approximately 8 meters too deep distanced almost 30 meters away from the contour results in stable tracking. As the vehicle acquires the target contour, it tracks the contour without any greater depth error than is seen with the better start point.

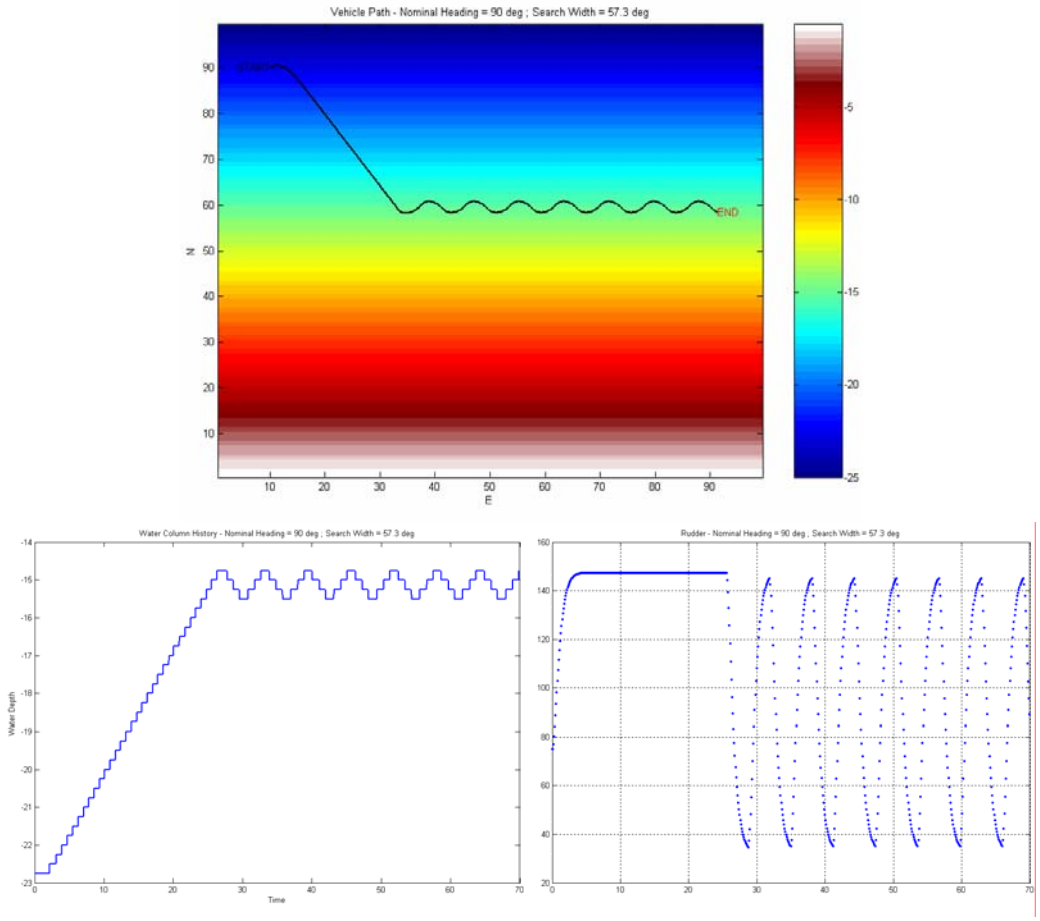

Figure 13. Straight Contour Response: Starting Deep

Figure 14 shows that starting the vehicle in water 10 meters too deep and 40 meters away from the desired contour still results in successful tracking. The vehicle's tracking performance remains as desirable as that achieved with either the deep or best starts.

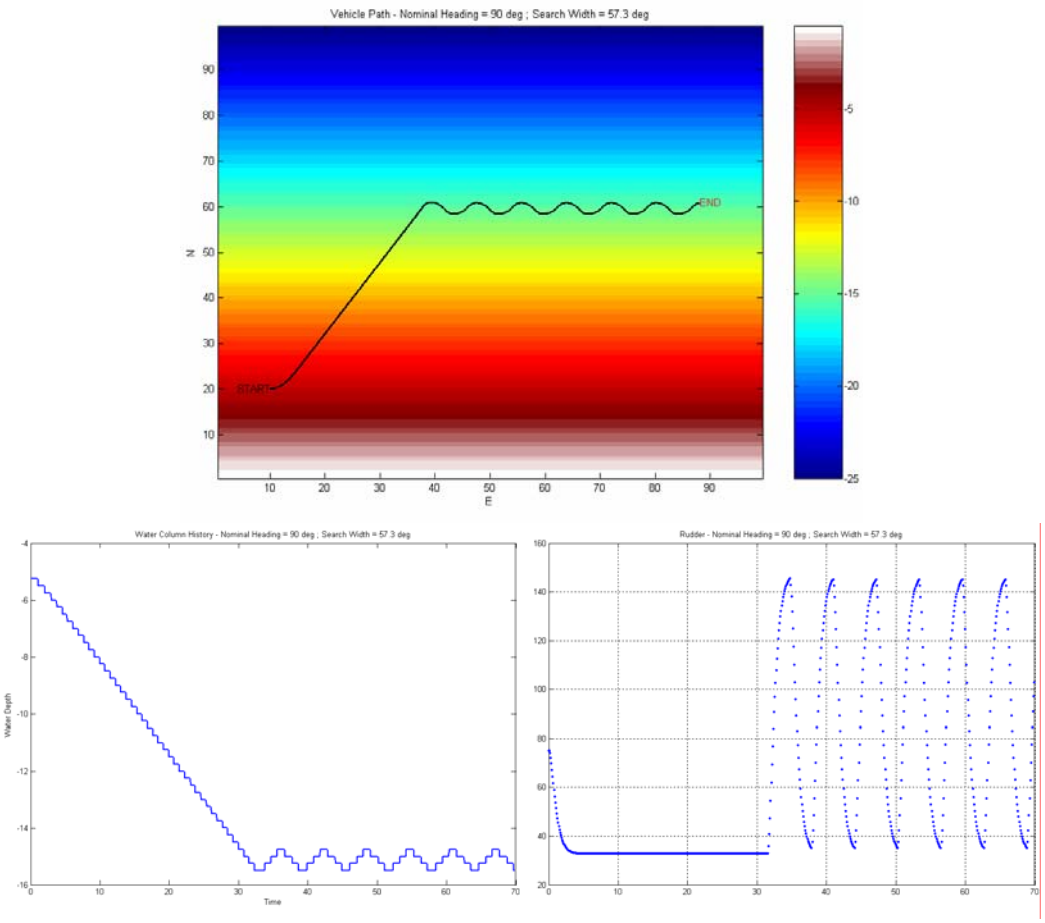

Figure 14. Straight Contour Response: Starting Shallow

The most important result from the heading-stabilized control proves that the algorithm successfully guides REMUS to track depth contours in simulation with real ocean data. Depth error now remains less than half a meter, and lateral deviation is reduced to the order of one or two meters along a 40-meter run.

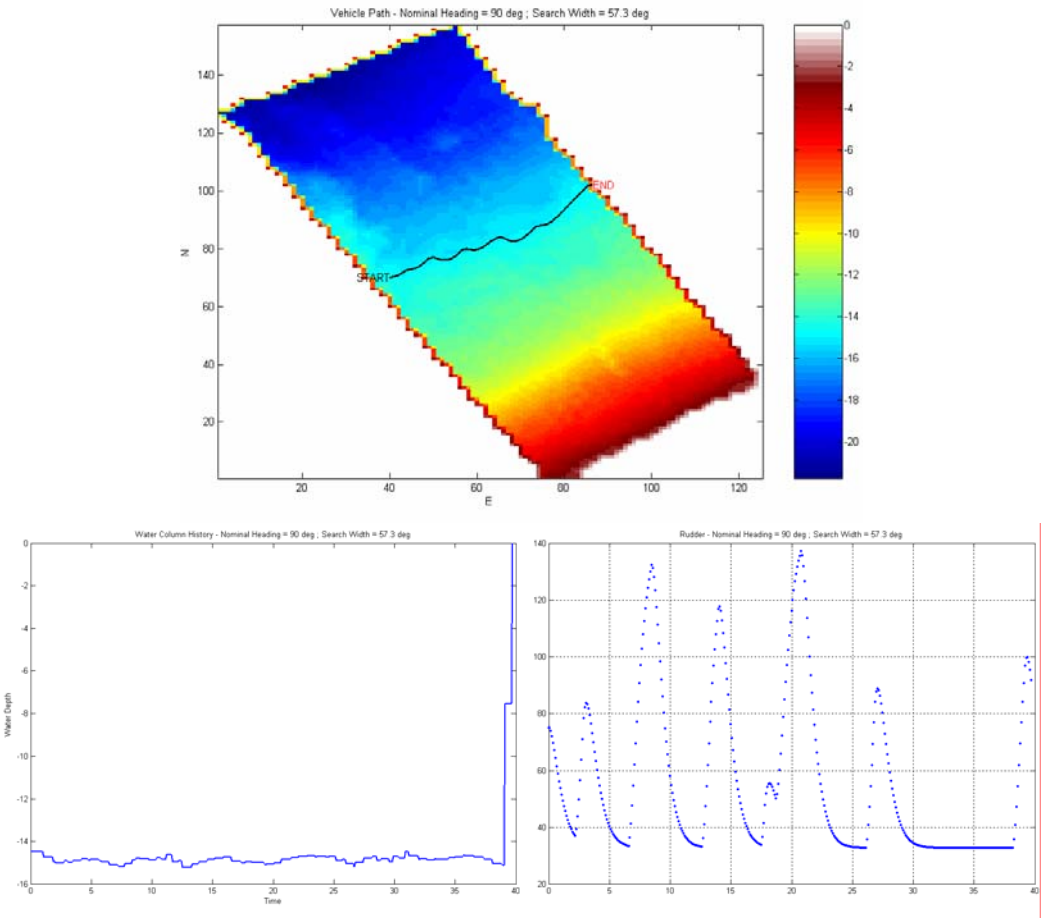

Figure 15. Real Contour Response

Perturbing the vehicle nearly 40 meters away from the primary location of the contour towards deeper water, figure 16 shows the algorithm correctly finds and tracks not only the target depth contour, but also tracks the dogleg in the contour seen along the western edge of the bathymetry sample.

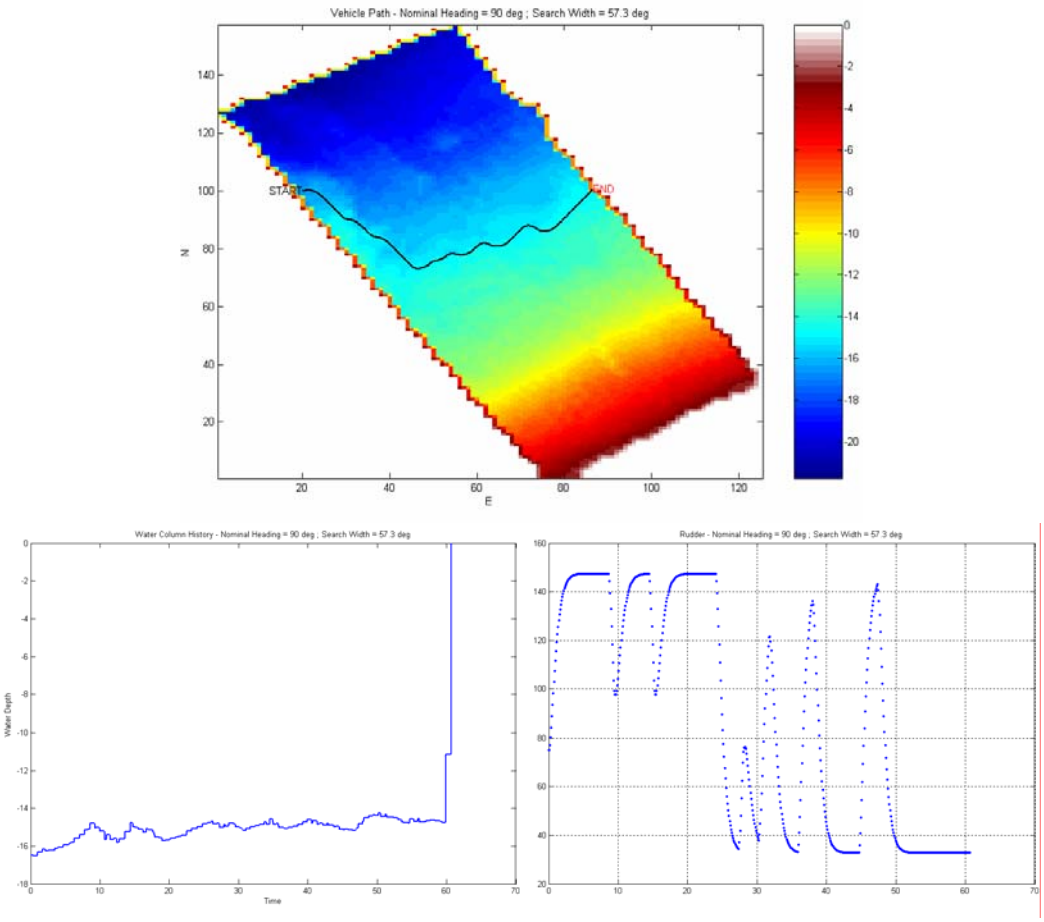

Figure 16. Real Contour Response: Starting Deep

2. Design Limitations

One factor is not immediately apparent in the previous tracking simulations. As stated in the control design section, heading commands are calculated within a finite span about a specified nominal heading. This fact causes problems when the contours points in a direction outside of the search cone. This limitation has one significant implication. The motivation for using contour tracking is to eliminate the need for advance knowledge of an operating area. Having to choose an appropriate nominal heading does not fulfill this objective; however, the amount of advance knowledge required for mission planning has been greatly reduced. Furthermore, it is not necessary to exactly match the nominal heading to the contour direction. The control will track the contour as long as it lies generally inside the nearly 120-degree zone covered by the search cone. Simple modifications suggested in chapter V should eliminate this issue altogether.

Figure 17 shows one example of this limitation. In this situation, the depth contour direction lies well within the search cone. As seen in the previous two simulations, the vehicle is more than capable of tracking this ocean floor model. In this simulation, starting the vehicle in shallow water requires that the vehicle move north to acquire the target contour. Because the contour direction points roughly 20 degrees north of the nominal heading, the 57.3 degree search cone limitation does not allow the vehicle to acquire the contour in any reasonable period of time. This simulation covers 90 seconds of vehicle run time. Arguably, the vehicle would eventually reach the contour and then successfully track it; however,

taking this much time to do so is unacceptable. Conversely, the opposing argument suggests that if the contour direction curves appreciably to the north ahead, the vehicle would never catch its "moving" target.

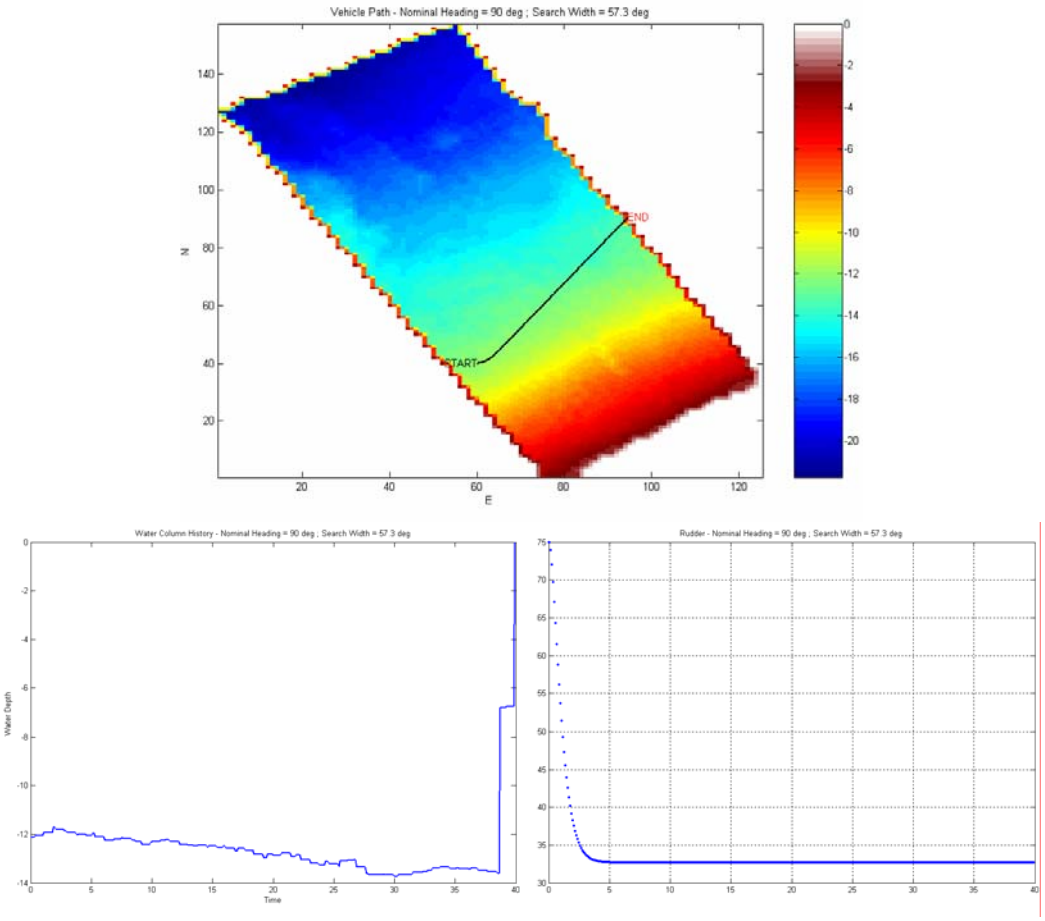

Figure 17. Real Contour Response: Starting Shallow, $\psi_{nominal}=90°$

Choosing a more appropriate nominal heading leads the vehicle to reach the contour much more quickly. Figure 18 shows the same vehicle simulation as figure 17 with identical initial conditions. With the 90-degree nominal heading used in the previous simulation, the vehicle never reached the contour during the run. Changing the nominal heading to 45 degrees, which still does not match the contour direction, the vehicle now acquires the contour in almost as little time as possible, then successfully tracks the contour for the remainder of the run, as seen in figure 18.

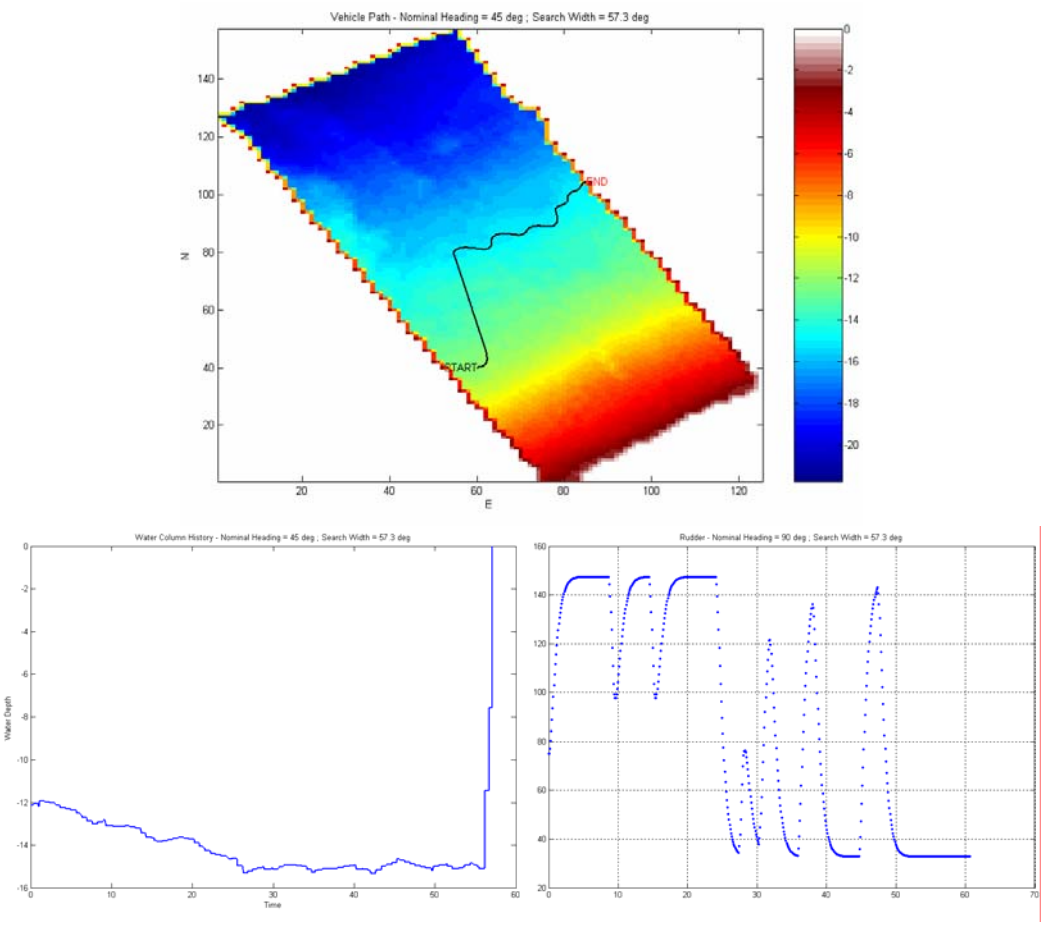

Figure 18. Real Contour Response: Starting Shallow, $\psi_{nominal}=45°$

One additional observation arises from the change in nominal heading. Figure 19 compares the tracking of real ocean data from a best-case initial condition using the 90 degree nominal heading (left) or the 45 degree nominal heading (right). By using the more appropriate nominal heading, it appears that the vehicle tracks the contour more tightly, characterized by the path following more localized contour curvature with less lateral deviation over the majority of the run.

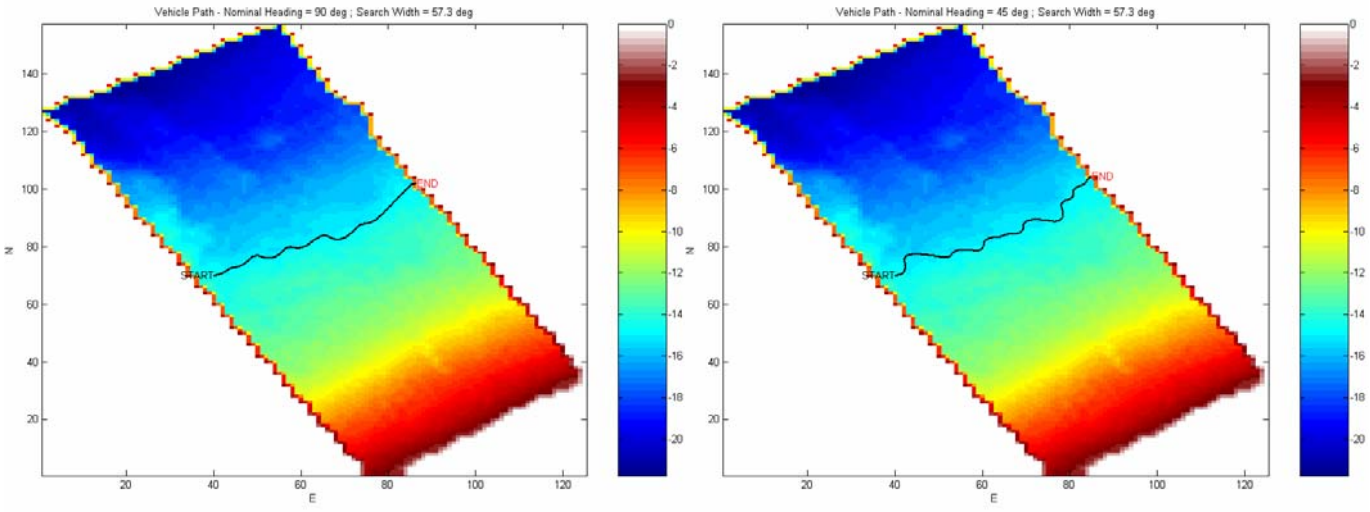

Figure 19. Effect on Performance of $\psi_{nominal}$

Figure 20 provides another example of the search cone limitation. Starting only slightly displaced from the target contour, the vehicle quickly acquires and tracks the contour, exhibiting the sinusoidal motion expected with ideal data. In the situation presented in this simulation, the contour first lies well within the search cone, and then curves continually until it points almost orthogonal to the nominal heading. This run shows that as the contour direction approaches the search cone limit, the vehicle takes longer to reach it, and once the contour points outside the search cone, the command from the controller saturates.

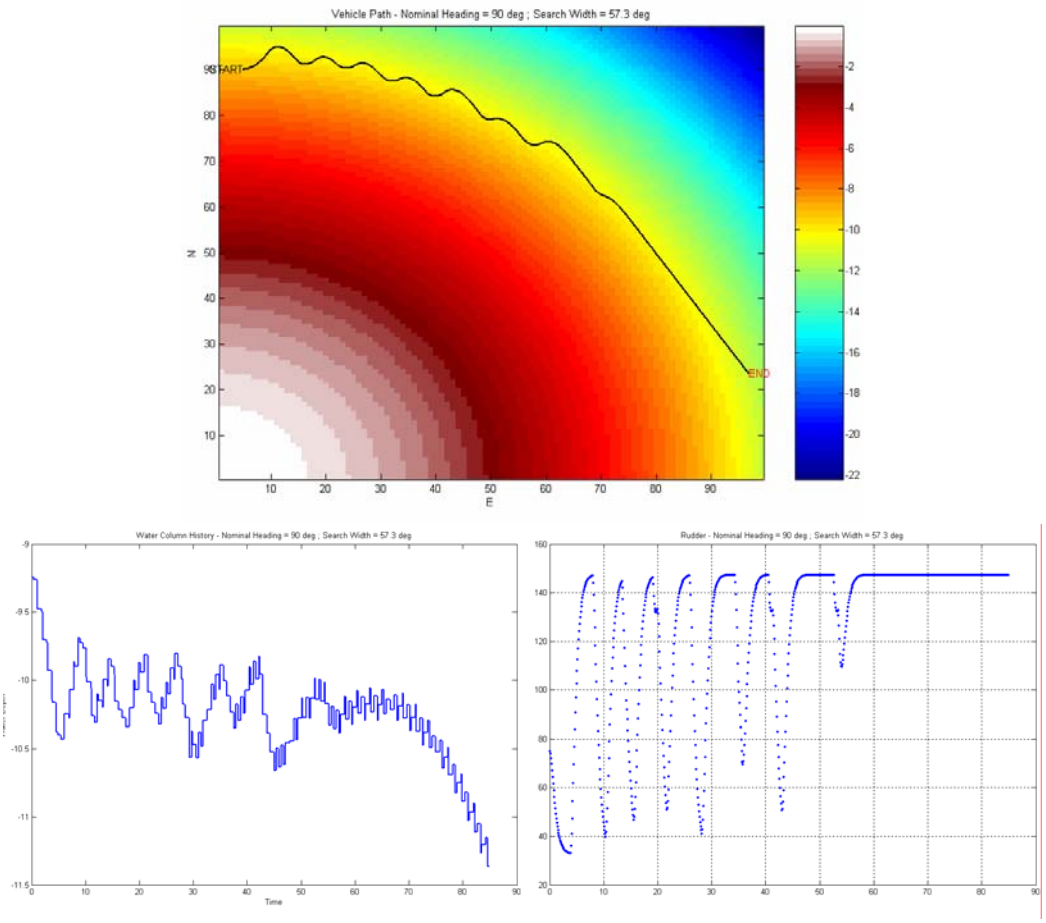

Figure 20. Curved Contour Response: $\psi_{nominal}=90°$

Figure 21 depicts the ideal data simulation showing the improvement associated with selection of a more appropriate nominal heading. During the portions of the run when the nominal heading and contour direction do not agree, the expected slow vehicle response is apparent. For the majority of the run, the contour lies well within the search cone and the vehicle's tracking performs remarkable well. Throughout this portion of the run, depth error remains mostly below 0.2 meters, and lateral deviation remains less than approximately 3 meters despite the oscillation induced by the bang/bang logic commands.

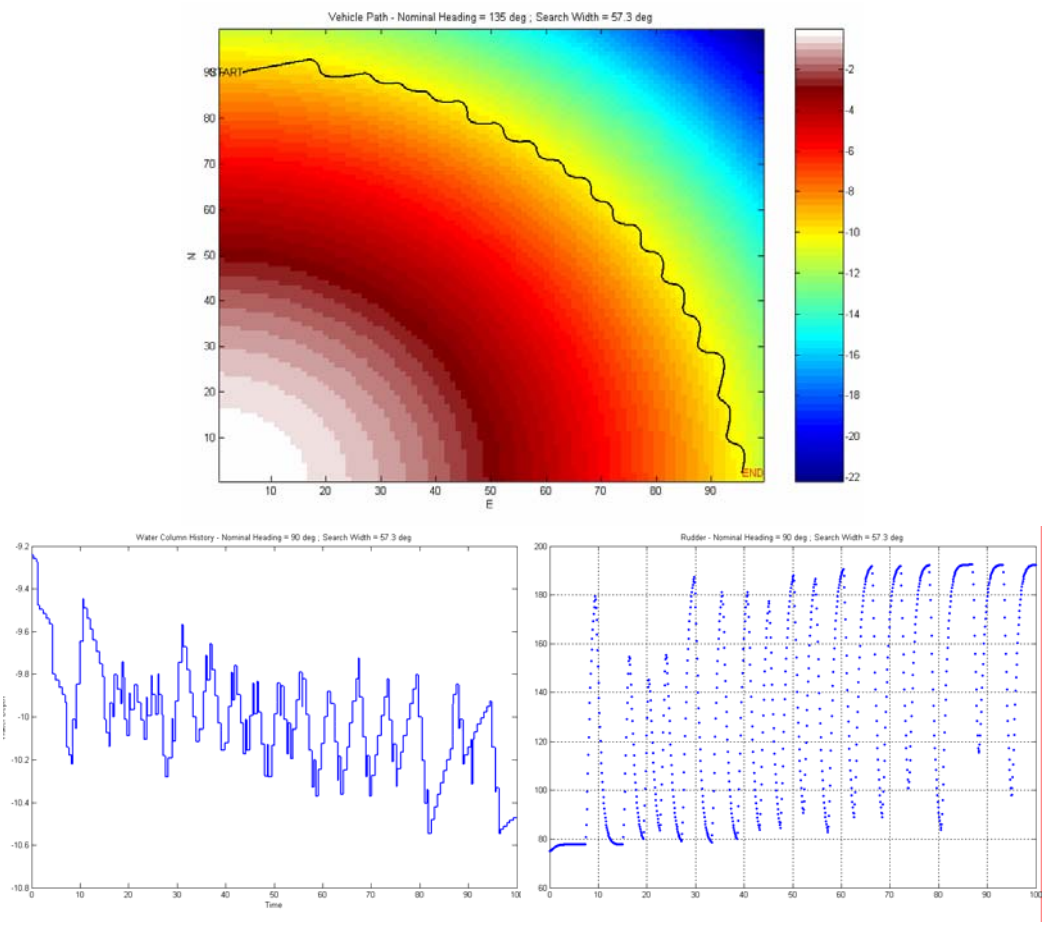

Figure 21. Curved Contour Response: $\psi_{nominal}=135°$

V. CONCLUSIONS AND RECOMMENDATIONS

A. CONLUSIONS

The results of these simulations lead to several important conclusions. The most important conclusion is the realization that the problem of contour tracking has a solution, and that the solution is not only relatively simple but does not require purchasing and installing multiple sensors on the vehicle.

In addition, simulations suggest that logic alone cannot sufficiently create a stable feedback loop. Traditional closed-loop control should be used instead to ensure system designs remain stable. Logic can be better used to derive control commands from outputs that do not have mathematical relationships. The research conducted in the thesis relevantly suggests possibilities for work throughout the field of sensor based control by presenting a method for controlling a vehicle with indirect sensor output.

One additional observation deserves mention due to the possibility that it may affect a very limited number of missions. The contour tracking control algorithms presented in this thesis are designed to track contour lines of constant depth in sloping geographies. These tracking algorithms cannot track minima or maxima features. This limitation is not associated with any particular algorithm, but is instead the result of the initial approach to solving the problem. It is caused because an actual gradient vector is not calculated numerically.

For example, figure 22 shows a shoreline region with a ditch and mound formation similar to that of a sandbar. If

mission objectives required tracking the bottom of the ditch feature, indicated by the red line, the control algorithm would cause the following to happen. Deviation about the ditch bottom in either direction would return shallow water feedback, likely with an unclear gradient approximation. The resulting control command would steer the vehicle away from shore (in the direction of the yellow line) until it reached the matching depth contour on a sloping feature, indicated by the green line. For the remainder of the mission, the vehicle would track the contour indicated by the green line.

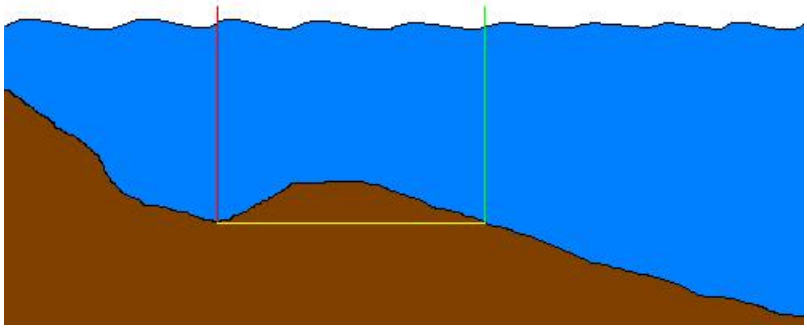

Figure 22. Contour Tracking of Minima/Maxima

B. RECOMMENDATIONS FOR FUTURE WORK

More research must be completed before the theories in this thesis can be trusted to safely control an unmanned vehicle in a real environment. The most apparent drawback in the present algorithm is the proper determination of the fixed nominal heading. Future work should focus on an accurate method for dynamically updating the nominal heading in the control command. This addition would completely solve the issue of tracking without advance knowledge of the operations area. The vehicle would be deployed in the water, given an initial position, and

pointed in the general direction it must travel, either up or down the coastline. As the vehicle acquires the target depth contour, something similar to the trend calculation used in the direct logic-based rudder control could be used to adjust the nominal heading when the vehicle detects the contour curving either left or right. The update must be optimized to adjust only for significant curves of the coastline, not for localized curvature due to uneven sea floor surfaces.

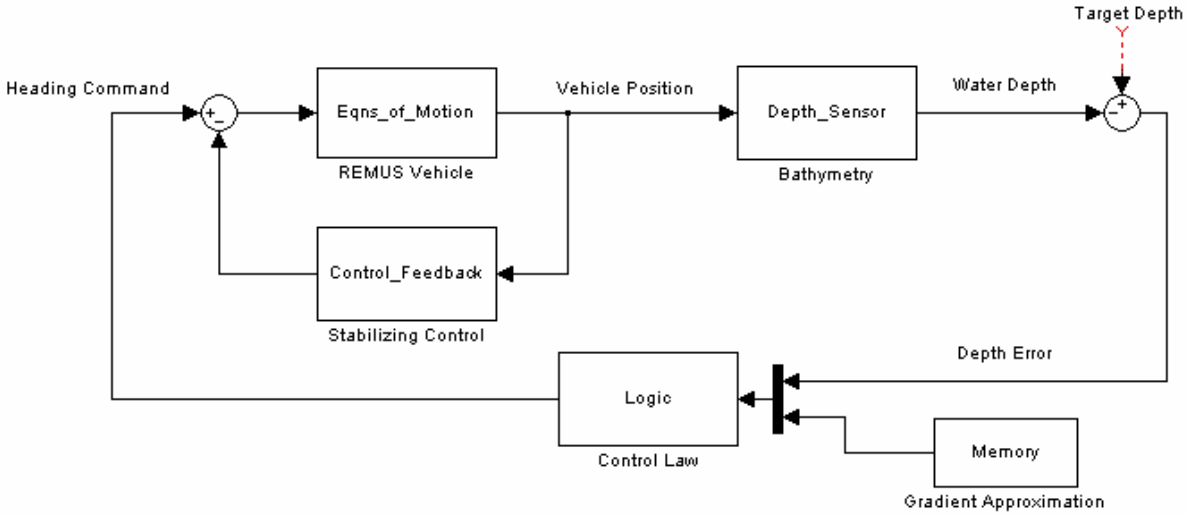

Figure 23. Block Diagram: Dynamically-Updated Heading-Stabilized Logic Control

By updating the nominal heading, the width of the search zone could be significantly reduced. The effect of a narrower search zone would reduce tracking of local curvatures, while the heading update would ensure tracking of the overall contour trend. This combination would produce more efficient vehicle paths, and power conservation is a primary goal for AUV research.

With the addition of a heading update feature, more research is needed not only for path optimization but to ensure that REMUS will safely handle any situation before risking trials with an actual vehicle. A few example concerns are suggested for testing with the control. What would happen if REMUS were unknowingly initiated too close to a sea hill? Would REMUS inadvertently acquire the contour around the hill and circle the hill indefinitely? Also, what would REMUS do when crossing a trench perpendicular to the shoreline? Would REMUS attempt to turn sharply and continue along the trench wall out to sea, or would it recognize the trench as a localized event and continue forward along the general shoreline trend?

One powerful potential is the possibility of pairing obstacle avoidance with contour tracking. Present obstacle avoidance models are already designed to work with the steering autopilot that comprises the inner loop of the heading-stabilized logic control. Such a control program would make a fully autonomous vehicle incredibly robust in almost any underwater environment with very little advance knowledge. Obstacle avoidance may also address the concerns presented in the previous paragraph. The ability to process hills or trenches as obstacles would help REMUS track general trends and not localized events. Additionally, forward look obstacle detection sensors could greatly improve the ability to locate contour lines while still not having to numerically calculate gradient vectors.

APPENDIX A: SEA FLOOR SIMULATION

```
% imports bathymetry data into workspace

global sea_floor

real_floor_temp = dlmread('smoothbathy.txt',',',7,0);
[length,width] = size(real_floor_temp);

for i = 1:50
    for j = 1:50
        straight_floor(i,j) = -i/2;
        curved_floor(i,j) = -((j/15)^2 + (i/15)^2);
    end
end

for i = 1:length
    for j = 1:width
        real_floor(i,j) = real_floor_temp(length-i+1,j);
    end
end

sea_floor = interp2(real_floor);  % Choose Which Bathymetry Model to
Use in Parentheses

load('bathycolormaps');

figure(1)
image(sea_floor,'CDataMapping','scaled');
colormap(colormapwhite);  % choose black or white to set color for zero
depth
%colormap(colormapblack);  % choose black or white to set color for
zero depth
colorbar;
set(gca,'YDir','normal');

%view(50,65);

%global sea_floor
%sea_floor_temp = dlmread('bathy.txt',',',1,0);
%[length,width] = size(sea_floor_temp);
%sea_floor = reshape(sea_floor_temp,length,length,width);
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B: DIRECT-RUDDER CONTROL; LINEAR GRADIENT

```
% This mfile uses corrected hydrodynamic coeff from MIT to develop
% a steering model.  It models REMUS following depth contours via a
% Linear Gradient Approximation

clc, close all, clear all

import_bathy

degrad = pi/180;
raddeg = 180/pi;
right = -1; left = 1;

TargetDepth = -10;
ToShore = right;
Rudder = 0.4;

% Set time of run

stop_time = 100;
dt = 0.1;
tau = [0:dt:stop_time];
%tau = linspace(0,30,1000);

% Set initial conditions

v = 0.0;
r = 0.0;
psi = 75.0*degrad;   % Initial Heading of the Vehicle
North = 90;          % Initial Position of the Vehicle in Meters (Use
70, 40 for Real ; 60, 5 for Straight ; 90, 5 Curved & -10 deep)
East = 5;
init_cond = [v;r;psi;North;East];    % [vr, r, psi, X, Y]

% REMUS Characteristic Specifications:

L = 1.33;           % Length in m
W = 2.99e02;        % Weigth in N
g = 9.81;           % Acceleration of gravity in m/s^2
m = W/g;            % Mass in kg
V = 1.543;          % Max Speed in m/s
rho = 1.03e03;      % Density of Salt H20 in kg/m^3
D = .191;           % Max diameter in m

% State Model Parameters

U = 1.543;          % m/s
Uo = U;
Ucx = 0; Ucy = 0;
Boy = 2.99e02;      % in N
xg  = 0; yg = 0; zg = 1.96e-02;     % in m

Iy = 3.45;          %kg/m^3 (from MIT thesis)
Iz = Iy;
```

```
% MIT REMUS Coeff (Dimensionalized)

Nvdot = 1.93;
Nrdot = -4.88;
Yvdot = -3.55e01;
Yrdot = 1.93;
%Nv = -4.47; should be same as Mw which is stated as +30.7
% should be -9.3 but going by Hoerner eqn, we get about 4.47
Nv = -4.47;
Nr = -6.87; %Same as Mq;
Yv = -6.66e01; %Same as Zw; Note should be -6.66e1 from MIT thesis not
2.86e01
Yr = 2.2 ; %Same as Zq = 2.2; MIT has miscalculation
Nd = -3.46e01/3.5; % Nd and Yd scaled by 3.5 to align w/exp data
Yd = 5.06e01/3.5;

% The Steering Equations for the REMUS are the following.
% These equations assume the primarily horizontal motions ...

MM=[(m-Yvdot) -Yrdot 0;-Nvdot (Iz-Nrdot) 0;0 0 1];
AA=[Yv (Yr-m*Uo) 0;Nv Nr 0; 0 1 0];
BB=[Yd;Nd;0];
A=inv(MM)*AA; B=inv(MM)*BB; C=[0,0,1]; D=0;

% Desired closed loop poles for sliding:

k=place(A,B,[-1.4,-1.45,0.0]);

% Closed loop dynamics matrix

Ac=A-B*k;
[m,n]=eig(Ac');
S=m(:,3);

% ************************************

% Begin Mission Simulation

states(:,1) = init_cond;

for i=1:(max(size(tau))-1)

    depth(i) = sounding(states(4,i),states(5,i));
    mydepth = depth(i) - TargetDepth;       % mydepth is positive if
shallow, negative if deep

    if depth(i) == 0
        disp('Error:  Vehicle Position Has No Depth Data')
        break
    end

    if i<2
        if mydepth < 0                      % Too Deep
            delta(i) = Rudder * ToShore;    % Turn to Shore
        end
        if mydepth >= 0                     % Too Shallow
```

```
            delta(i) = Rudder * -ToShore;   % Turn away from Shore
        end
    else
        trend = depth(i) - depth(i-1);      % trend is positive going
shallower, negative going deeper

        if ((mydepth < 0) && (trend < 0))               % Too Deep or
On Track and Getting Deeper
            depthstate(i) = 1;
        elseif ((mydepth < 0) && (trend > 0))           % Too Deep but
Getting Shallower
            depthstate(i) = 2;
        elseif ((mydepth > 0) && (trend > 0))           % Too Shallow
or On Track and Getting Shallower
            depthstate(i) = 3;
        elseif ((mydepth > 0) && (trend < 0))           % Too Shallow
but Getting Deeper
            depthstate(i) = 4;
        else
            depthstate(i) = 0;
        end

        switch depthstate(i)

            case 0
                delta(i) = 0;

            case 1                                      % Too Deep or On
Track and Getting Deeper
                delta(i) = Rudder * ToShore;        % Turn Rudder to
Shore

            case 2                                      % Too Deep but
Getting Shallower
                delta(i) = 0;                       % Stay Straight or
Rudder Amidships

            case 3                                      % Too Shallow or On
Track and Getting Shallower
                delta(i) = Rudder * -ToShore;       % Turn Rudder away
from Shore

            case 4                                      % Too Shallow but
Getting Deeper
                delta(i) = 0;                       % Stay Straight or
Rudder Amidships

            otherwise                                   % Unknown Condition
                delta(i) = 0;                       % Stay Straight and
Print Error
                disp('Error:  Current Depth State is Unknown')
        end
    end

    dX(1:3,i) = A * [states(1,i);states(2,i);states(3,i)] + B *
delta(i);
```

```
    dX(4,i) = Uo * cos(states(3,i)) - states(1,i) * sin(states(3,i)) +
Ucx;
    dX(5,i) = Uo * sin(states(3,i)) + states(1,i) * cos(states(3,i)) +
Ucy;

    states(:,i+1) = states(:,i) + dt * dX(:,i);

end

vr = states(1,:);
r = states(2,:);
psi = states(3,:);
X = states(4,:);
Y = states(5,:);
Beta = atan2(vr,U);     % Beta = Side slip angle

figure(1)
hold on
plot(Y,X,'k','LineWidth',2);
text(Y(1),X(1),'START','color','k','HorizontalAlignment','right');
text(Y(max(size(Y))),X(max(size(X))),'END','color','r');
title('Vehicle Path - Linear Gradient ; Rudder = 0.4 rad');
xlabel('East');ylabel('North');

figure(2)
plot(tau(1:max(size(depth))),depth,'b','LineWidth',2);
title('Water Column History - Linear Gradient ; Rudder = 0.4 rad');
xlabel('Time');ylabel('Water Depth');

figure(3)
hold on
plot(tau(1:max(size(depthstate))),depthstate)
ylim([0 5])
title('Rudder Command Logic - Linear Gradient ; Rudder = 0.4 rad');
```

APPENDIX C: DIRECT-RUDDER CONTROL; ESTIMATED LOCAL GRADIENT

```
% This mfile uses corrected hydrodynamic coeff from MIT to develop
% a steering model.  It models REMUS following depth contours via an
% Estimated Local Gradient

clc, close all, clear all

import_bathy

degrad = pi/180;
raddeg = 180/pi;
right = -1; left = 1;

% Set Mission Variables

TargetDepth = -15;
ToShore = right;
Rudder = 0.2;

% Set time of run

stop_time = 60;
dt = 0.1;
tau = [0:dt:stop_time];
%tau = linspace(0,30,1000);

% Set initial conditions

v = 0.0;
r = 0.0;
psi = 75.0*degrad;   % Initial Heading of the Vehicle
North = 70;          % Initial Position of the Vehicle in Meters (Use
70, 40 for Real ; 60, 5 for Straight ; 90, 5 Curved & -10 deep)
East = 40;
init_cond = [v;r;psi;North;East];    % [vr, r, psi, X, Y]

% REMUS Characteristic Specifications:

L = 1.33;           % Length in m
W = 2.99e02;        % Weigth in N
g = 9.81;           % Acceleration of gravity in m/s^2
m = W/g;            % Mass in kg
V = 1.543;          % Max Speed in m/s
rho = 1.03e03;      % Density of Salt H20 in kg/m^3
D = .191;           % Max diameter in m

% State Model Parameters

U = 1.543;          % m/s
Uo = U;
Ucx = 0; Ucy = 0;
Boy = 2.99e02;      % in N
```

```
xg  = 0; yg = 0; zg = 1.96e-02;     % in m

Iy = 3.45;          %kg/m^3 (from MIT thesis)
Iz = Iy;

% MIT REMUS Coeff (Dimensionalized)

Nvdot = 1.93;
Nrdot = -4.88;
Yvdot = -3.55e01;
Yrdot = 1.93;
%Nv = -4.47; should be same as Mw which is stated as +30.7
% should be -9.3 but going by Hoerner eqn, we get about 4.47
Nv = -4.47;
Nr = -6.87; %Same as Mq;
Yv = -6.66e01; %Same as Zw; Note should be -6.66e1 from MIT thesis not
2.86e01
Yr = 2.2 ; %Same as Zq = 2.2; MIT has miscalculation
Nd = -3.46e01/3.5; % Nd and Yd scaled by 3.5 to align w/exp data
Yd = 5.06e01/3.5;

% The Steering Equations for the REMUS are the following.
% These equations assume the primarily horizontal motions ...

MM=[(m-Yvdot) -Yrdot 0;-Nvdot (Iz-Nrdot) 0;0 0 1];
AA=[Yv (Yr-m*Uo) 0;Nv Nr 0; 0 1 0];
BB=[Yd;Nd;0];
A=inv(MM)*AA; B=inv(MM)*BB; C=[0,0,1]; D=0;

% Desired closed loop poles for sliding:

k=place(A,B,[-1.4,-1.45,0.0]);

% Closed loop dynamics matrix

Ac=A-B*k;
[m,n]=eig(Ac');
S=m(:,3);

% ************************************

% Begin Mission Simulation

states(:,1) = init_cond;    % [vr, r, psi, X, Y]

for i=1:(max(size(tau))-1)

    depth(i) = sounding(states(4,i),states(5,i));
    mydepth = depth(i) - TargetDepth;       % mydepth is positive if
shallow, negative if deep

    if depth(i) == 0
        disp('Error:  Vehicle Position Has No Depth Data')
        break
    end

    if i<2
```

```
        if mydepth < 0                          % Too Deep
            delta(i) = Rudder * ToShore;        % Turn to Shore
        end
        if mydepth >= 0                         % Too Shallow
            delta(i) = Rudder * -ToShore;       % Turn away from Shore
        end
    else
        grad_x = sounding(states(4,i)+1,states(5,i))-depth(i);
        grad_y = sounding(states(4,i)+2,states(5,i)+2)-depth(i);
        psi = states(3,i);
        psideg = psi*raddeg;
        trend = cos(psi)*grad_y + sin(psi)*grad_x;

%         if (psi > 315 && psi < 45)
%             trend = grad_y;
%         elseif (psi >= 45 && psi <= 135)
%             trend = grad_x;
%         elseif (psi > 135 && psi < 225)
%             trend = -grad_y;
%         elseif (psi >= 225 && psi <= 315)
%             trend = -grad_x;
%         end

        % trend is positive going shallower, negative going deeper

        if ((mydepth <= 0) && (trend < 0))               % Too Deep or
On Track and Getting Deeper
            depthstate(i) = 1;
        elseif ((mydepth < 0) && (trend >= 0))           % Too Deep but
Getting Shallower
            depthstate(i) = 2;
        elseif ((mydepth >= 0) && (trend >= 0))           % Too Shallow
or On Track and Getting Shallower
            depthstate(i) = 3;
        elseif ((mydepth > 0) && (trend < 0))           % Too Shallow
but Getting Deeper
            depthstate(i) = 4;
        end

        switch depthstate(i)

            case 1                                  % Too Deep or On
Track and Getting Deeper
                delta(i) = Rudder * ToShore;        % Turn Rudder to
Shore

            case 2                                  % Too Deep but
Getting Shallower
                delta(i) = 0;                       % Stay Straight or
Rudder Amidships

            case 3                                  % Too Shallow or On
Track and Getting Shallower
                delta(i) = Rudder * -ToShore;       % Turn Rudder away
from Shore
```

```
            case 4                                  % Too Shallow but
Getting Deeper
                delta(i) = 0;                       % Stay Straight or
Rudder Amidships

            otherwise                               % Unknown Condition
                delta(i) = 0;                       % Stay Straight and
Print Error
                disp('Error:  Current Depth State is Unknown')
        end
    end

    dX(1:3,i) = A * [states(1,i);states(2,i);states(3,i)] + B *
delta(i);
    dX(4,i) = Uo * cos(states(3,i)) - states(1,i) * sin(states(3,i)) +
Ucx;
    dX(5,i) = Uo * sin(states(3,i)) + states(1,i) * cos(states(3,i)) +
Ucy;

    states(:,i+1) = states(:,i) + dt * dX(:,i);

end

vr = states(1,:);
r = states(2,:);
psi = states(3,:);
X = states(4,:);
Y = states(5,:);
Beta = atan2(vr,U);     % Beta = Side slip angle

% Display Run

figure(1)
hold on
plot(Y,X,'k','LineWidth',2);
text(Y(1),X(1),'START','color','k','HorizontalAlignment','right');
text(Y(max(size(Y))),X(max(size(X))),'END','color','r');
title('Vehicle Path - Local Gradient ; Rudder = 0.2 rad');
xlabel('East');ylabel('North');

figure(2)
plot(tau(1:max(size(depth))),depth,'b','LineWidth',2);
title('Water Column History - Local Gradient ; Rudder = 0.2 rad');
xlabel('Time');ylabel('Water Depth');

figure(3)
hold on
plot(tau(1:max(size(depthstate))),depthstate)
ylim([0 5])
title('Rudder Command Logic - Local Gradient ; Rudder = 0.2 rad');
```

APPENDIX D: HEADING-STABILIZED LOGIC CONTROL

```
% This mfile uses corrected hydrodynamic coeff from MIT to develop
% a steering model.  It models REMUS following depth contours via a
% Heading Stabilized Logic Controller

clc, close all, clear all

import_bathy

degrad = pi/180;
raddeg = 180/pi;
right = -1; left = 1;

% Set Mission Variables

TargetDepth = -10;
ToShore = right;
psinominal = 135*degrad;     % Nominal Heading about which to Track
searchwidth = 1;            % Angular Tracking Range about Nominal
Heading in Radians

% Set time of run

stop_time = 100;
dt = 0.1;
tau = [0:dt:stop_time];

% Set initial conditions

v = 0.0;
r = 0.0;
psi = 75.0*degrad;   % Initial Heading of the Vehicle
North = 90;          % Initial Position of the Vehicle in Meters (Use
70, 40 for Real ; 60, 5 for Straight ; 90, 5 Curved & -10 deep)
East = 5;
init_cond = [v;r;psi;North;East];    % [vr, r, psi, X, Y]

% REMUS Characteristic Specifications:

L = 1.33;           % Length in m
W = 2.99e02;        % Weigth in N
g = 9.81;           % Acceleration of gravity in m/s^2
m = W/g;            % Mass in kg
V = 1.543;          % Max Speed in m/s
rho = 1.03e03;      % Density of Salt H20 in kg/m^3
D = .191;           % Max diameter in m

% State Model Parameters

U = 1.543;          % m/s
Uo = U;
Ucx = 0; Ucy = 0;
Boy = 2.99e02;      % in N
xg  = 0; yg = 0; zg = 1.96e-02;     % in m
```

```
Iy = 3.45;          %kg/m^3 (from MIT thesis)
Iz = Iy;

% MIT REMUS Coeff (Dimensionalized)

Nvdot = 1.93;
Nrdot = -4.88;
Yvdot = -3.55e01;
Yrdot = 1.93;
%Nv = -4.47; should be same as Mw which is stated as +30.7
% should be -9.3 but going by Hoerner eqn, we get about 4.47
Nv = -4.47;
Nr = -6.87; %Same as Mq;
Yv = -6.66e01; %Same as Zw; Note should be -6.66e1 from MIT thesis not
2.86e01
Yr = 2.2 ; %Same as Zq = 2.2; MIT has miscalculation
Nd = -3.46e01/3.5; % Nd and Yd scaled by 3.5 to align w/exp data
Yd = 5.06e01/3.5;

% The Steering Equations for the REMUS are the following.
% These equations assume the primarily horizontal motions ...

MM=[(m-Yvdot) -Yrdot 0;-Nvdot (Iz-Nrdot) 0;0 0 1];
AA=[Yv (Yr-m*Uo) 0;Nv Nr 0; 0 1 0];
BB=[Yd;Nd;0];
A=inv(MM)*AA; B=inv(MM)*BB; C=[0,0,1]; D=0;

% Desired closed loop poles for sliding:

k=place(A,B,[-1.4,-1.45,-1.5]);

% Closed loop dynamics matrix

% Ac=A-B*k;
% [m,n]=eig(Ac');
% S=m(:,3);

% ************************************

% Begin Mission Simulation

states(:,1) = init_cond;    % [vr, r, psi, X, Y]

%psicom(1)=90*degrad;

for i=1:(max(size(tau))-1)

    depth(i) = sounding(states(4,i),states(5,i));
    mydepth = depth(i) - TargetDepth;       % mydepth is positive if
shallow, negative if deep

    if depth(i) == 0
        disp('Error:  Vehicle Position Has No Depth Data')
        break
    end
```

```
        if mydepth < 0                                              % Too
Deep
            psicom(i) = psinominal + searchwidth * -ToShore;    % Turn
to Shore(South)
        end
        if mydepth >= 0                                             % Too
Shallow
            psicom(i) = psinominal + searchwidth * ToShore;     % Turn
away from Shore(North)
        end

% steering control law

    delta(i)=-k*states(1:3,i)+k(3)*psicom(i);

% state update

    dX(1:3,i) = A * [states(1,i);states(2,i);states(3,i)] + B *
delta(i);
    dX(4,i) = Uo * cos(states(3,i)) - states(1,i) * sin(states(3,i)) +
Ucx;
    dX(5,i) = Uo * sin(states(3,i)) + states(1,i) * cos(states(3,i)) +
Ucy;

    states(:,i+1) = states(:,i) + dt * dX(:,i);

end;   %end of for loop

vr = states(1,:);
r = states(2,:);
psi = states(3,:);
X = states(4,:);
Y = states(5,:);
Beta = atan2(vr,U);     % Beta = Side slip angle

% Display Run

figure(1)
hold on
plot(Y,X,'k','LineWidth',2);
text(Y(1),X(1),'START','color','k','HorizontalAlignment','right');
text(Y(max(size(Y))),X(max(size(X))),'END','color','r');
title('Vehicle Path - Nominal Heading = 90 deg ; Search Width = 57.3
deg');
xlabel('E');ylabel('N');

figure(2)
plot(tau(1:max(size(depth))),depth,'b','LineWidth',2);
title('Water Column History - Nominal Heading = 90 deg ; Search Width =
57.3 deg');
xlabel('Time');ylabel('Water Depth');

figure(3)
plot(tau(1:max(size(psi))),psi*raddeg,'b.')
grid
title('Rudder - Nominal Heading = 90 deg ; Search Width = 57.3 deg')
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX E: WATER COLUMN DEPTH SENSOR

```
function depth = sounding(X,Y)

global sea_floor

X=round(X);
Y=round(Y);

depth = sea_floor(X,Y);
```

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Blidberg, Richard D., "The Development of Autonomous Underwater Vehicles (AUVs): A Brief Summary", Autonomous Undersea Systems Institute, ICRA, Seoul, Korea, May 2001.

Clark, Vernon, "Seapower 21, Projecting Decisive Force Capabilities", United States Naval Institute Proceedings, October 2002, www.usni.org.

Fodrea, Lynn, "Obstacle Avoidance Control for the REMUS Autonomous Underwater Vehicle", M.S.M.E. Thesis, Naval Postgraduate School, December 2002.

Fodrea, Lynn and Healey, A. J., "Obstacle Avoidance Control for the REMUS Autonomous Underwater Vehicle", Proceedings of the IFAC GCUUV Conference, Swansea, Wales, 2003.

Healey, Anthony J., "Dynamics of Marine Vehicles (ME-4823)", Class Notes, Naval Postgraduate School, Monterey, CA, 2003.

Hurst, S., "REMUS", Oceanographic Systems Laboratory, Woods Hole Oceanographic Institution, http://www.whoi.edu/science/AOPE/dept/OSL/remus.html, June 4, 2005.

Moreau, Luc, Bachmayer, R. and Leonard, N.E., "Coordinated Gradient Descent: A Case Study of Lagrangian Dynamics with Projected Gradient Information", Proceedings of IFAC Workshop on Lagrangian and Hamiltonian Methods for Nonlinear Control, 2003, http://www.princeton.edu/~naomi/, June 4, 2005.

Ogren, Petter, Fiorelli, E. and Leonard, N.E., "Cooperative Control of Mobile Sensor Networks: Adaptive Gradient Climbing in a Distributed Environment", IEEE Transactions on Automatic Control, 2004, http://www.princeton.edu/~naomi/, June 4, 2005.

Prestero, Timothy, "Verification of a Six-Degree of Freedom Simulation Model for the REMUS Autonomous Underwater Vehicle," M.S. Thesis, Massachusetts Institute of Technology, Sep 2001.

Zhang, Fumin and Leonard, N.E., "Generating Contour Plots Using Multiple Sensor Platforms", To Appear in 2005 IEEE Swarm Intelligence Symposium, 2005, http://www.princeton.edu/~naomi/, June 4, 2005.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Distinguished Professor Anthony J. Healey, Code ME/HY
   Chairman, Department of Mechanical Engineering
   Naval Postgraduate School
   Monterey, California

4. Naval/Mechanical Engineering Curriculum Code 34
   Naval Postgraduate School
   Monterey, California

5. Doug Horner
   Naval Postgraduate School
   Monterey, California

6. ENS Alan Van Reet
   Naval Nuclear Power School
   Charleston, South Carolina