AUV Guidance, Navigation and Control Improvements

Project 900402 Final Report

Monterey Bay Aquarium Research Institute
7700 Sandholdt Road
Moss Landing, CA 95039-9644
rob@mbari.org

Rob McEwen

Revision 1.0

January 3, 2005

Contents

List of Figures

Abstract

A substantially improved depth control algorithm was designed, coded, simulated, and successfully tested at sea on the MBARI mapping AUV. An obstacle avoidance algorithm was also designed, coded, and tested. At-sea tests of the obstacle avoidance system were inconclusive due to the limited range versus angle-of-incidence characteristic of the forward-looking sonar.

1 Introduction

The main purpose of this project was to design, code and test an Obstacle Avoidance (OA) algorithm for the mapping AUV. The algorithm uses a forward-looking sonar, the Datasonics Psa916, to detect a fixed obstruction ahead. Ultimately, the OA system may be duplicated on the CTD AUV, if space, power and resource limitations will permit a new sonar in the nose. Moving obstacles are not considered in this report.

Obstacle avoidance is important because MBARI's AUVs are presently at risk of colliding with an obstacle ahead, particularly a steep upslope when flying at altitudes of 20 meters or less. This risk is borne out by the fact that on at least two occasions an AUV has returned with mud caked on it, clearly having hit the bottom. The mapping AUV has reason to fly at a low altitude because it improves the quality of the sub-bottom profiling data. Additionally, the CTD vehicle often does bottom-terminated yoyos to within 20 meters or less.

There are two main requirements for obstacle avoidance; first is to develop a guidance algorithm that commands maneuvers based on sonar measurements, particularly the forward-looking sonar. We will concentrate on vertical-plane maneuvers, since the most likely obstacle is a steep upslope. More advanced obstacle avoidance algorithms that involve horizontal-plane maneuvering and path planning around obstacles are not feasible using the Psa916 because this is a single-dimensional range finder, and cannot detect obstacles to either side.

The second major obstacle avoidance requirement is to speed up the vehicle depth control so that it can respond to the avoidance command quickly enough. The present depth control has a very low bandwidth, and the ground the vehicle covers before it can change it's depth exceeds the expected range of the Psa916.

There is a large body of research available on obstacle avoidance. Much of this is for land robots, where the problem differs significantly from the AUV application in two ways. First, a wheeled land robot is confined to the horizontal plane; it cannot fly over an obstacle like the AUV can. Second, a wheeled robot can stop, pause, and back up, where a flying AUV cannot.

There is a smaller literature on AUV obstacle avoidance. Most of these algorithms assume a two-dimensional forward-looking sonar is available, usually a multi-beam.

Petillot *et al.* [7] develop a 2-D obstacle avoidance and a path planning algorithm for an AUV using a multibeam sonar. The algorithm uses constructive solid geometry to model the obstacles in the workspace, and then uses a nonlinear search routine for path planning and moving obstacle avoidance.

Antonelloi *et al.* apply a Virtual Force Field (VFF) approach to AUV obstacle avoidance [1]. This paper has only simulation results. It uses a scanning sonar. See [2] for a development of VFF.

A Naval Postgraduate School group has applied fuzzy logic to the AUV obstacle avoidance problem, using a Remus vehicle and an unspecified forward-looking 2-D sonar [4].

The goals of this project are summarized below in Section 2. The project results are summarized Section 3, page 5, along with some incidental minor discoveries, which nonetheless are important. The conclusion of this report, in Section 7, page 42, suggests future work.

2 Summary of Project Goals

This section summarizes the original project goals, but also includes additional sub-goals that became necessary during the course of the project.

1. Develop, code and test a simple obstacle avoidance algorithm, using the existing Psa916 range finder, that gives the vehicle the capability to sense a fixed object ahead, such as a hill, and climb over it.

2. Simulate the OA algorithm, which requires a non-flat bottom model.

3. Measure the operational range of the Psa916 in the mapping vehicle, and characterize its performance.

4. Improve the transient response of the depth control. This is also a prerequisite for Item 1.

5. Test the OA system and the improved depth control at sea.

6. Develop a navigation algorithm that takes compass and AHRS inputs and computes geodetic location (latitude and longitude) without converting to UTM.

3 Summary of Project Results

The results of this project are summarized here:

1. This report developed an obstacle avoidance algorithm. It works by climbing over obstacles without altering the vehicle's course. This is achieved by modifying the altitude calculation of Equations 19 and 20, page 12, to incorporate the forward-looking

sonar. Figure 7, page 16, summarizes how this is done. Comparing the estimated bottom (blue/white interface) in the top strip of Figure 9, page 21, to Figure 10, page 23, shows the improvement. These are simulations.

2. Removing the pitch integrator and adding the depth rate control term resulted in an increase of a factor of four in the closed-loop depth-control bandwidth. Comparision of Figures 19, page 35, and 20, page 36, show the improvement in step response. These are plots of AUV sea-trial data.

3. The existing Dvl-only altitude calculation, in use since the inception of AUVs at MBARI, was found to be geometrically incorrect. The error is 6% of altitude at a vehicle pitch angle of 30°. See Equation 14, page 11, and Figure 4, page 11, in Section 4.2. The new altitude equation was verified by simulation; compare the dashed magenta trace in the top strip of Figure 9, page 21, to Figure 10, page 23.

4. The yoyo mission of Figure 24, page 41, top strip, shows that there is still some other altitude measurement problem. The bumps on the bottom are clearly not real, and are apparently proportional to altitude but independent of pitch. The error is about 13% of altitude, which is large.

5. The Psa916 performs very poorly when its line-of-sight forms a small angle-of-incidence with the bottom. It is particularly vulnerable to a misleading side-lobe return. See the black trace at around 2km in the bottom strip of Figure 22, page 39. Both the 12 meter range measurement, and the decreasing range reading that starts when the vehicle ascends, are from sidelobes. The Psa916 never found the correct range to the bottom.

6. The Qnx simulation was modified to model the five sonar beams passing over a non-flat bottom. This was accomplished with some trigonometry and a bisection root finder. The simulation can now model topography composed of simple functions such as sinusoids or piece-wise linear functions. For an example, see Figure 9, page 21.

This project originally had an objective to develop a navigation algorithm that takes compass and AHRS measurements in, and directly computes location in latitude and longitude. This would skip the present step of converting into and out of UTM. This objective was eliminated in favor of pursuing the higher-priority OA work.

4 Guidance: Obstacle Avoidance

Figure 1, page 7, shows the basic geometry and the relative ranges of the Dvl and Psa916 sonars. The Psa916 has an expected range of 50 meters.

The most likely obstacle the vehicle will encounter will be a hill or cliff. Thus the OA system must detect rising ground ahead, and be able to either climb over it or alter its course. The simplest approach, and the one used here, is to do this by using only the vertical-plane depth/pitch control loop to climb over, and leave the waypoint/heading control unconstrained. This avoids much complication that will occur if both the OA system and a waypoint or setpoint behavior are simultaneously trying to command heading.

There are two probable cases using only depth control; first, the Dvl detects the bottom, and the Psa916 does not. This will be common, since the Psa916 will look mostly ahead, and will not have much downward range compared to the Dvl. Second is when both sonars return a range to the bottom. Before exploring each of these cases further, we will first select a mounting angle for the Psa916 in the following section. This is done early in the report, because it was done early in the project, since any mounting bracket design and mechanical modifications are best done first.

A third, but improbable, case is if the Psa916 should detect an obstacle ahead, but the Dvl doesn't. This is unlikely in mid-latitude waters, because the Dvl look-ahead angle is 22.5°, which means that it will see about 80 meters ahead, 200 meters below the vehicle. A Psa916-only detection would mean the AUV is approaching an overhang, or something floating in the water. This case will need to be accounted for should we run a vehicle in the Arctic, where it is much more likely to occur. Ice keels underneath pressure ridges can extend well below the surface ice. Embedded icebergs can extend further, although they are less frequent.

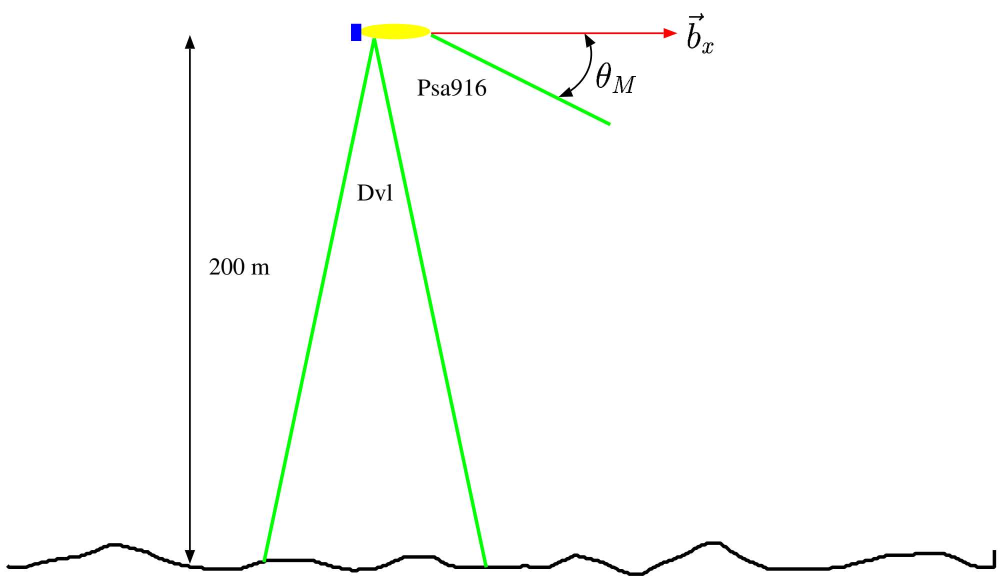

Figure 1: Dvl and Forward-Looking Psa916 Sonars

4.1 Sonar Mount Angle

The objective of this section is to determine the Psa916 mount angle. This section assumes an ideal sonar that works independently of angle-of-incidence with the bottom. During testing a low angle-of-incidence was found to be problematic. See Section 6.2, page 35.

Let r_P be the range of the Psa916, and θ_M be the mount angle as shown in Figure 1. Further suppose that the vehicle is in straight and level flight, with a pitch angle of zero. Assume that the maximum range of the Psa916 is 50 meters. Then the maximum altitude before the Psa916 looses sight of the bottom is given by

$$h_M = r_P \sin\theta_M. \tag{1}$$

The Psa916 beam intersects the ground x_M meters ahead of the vehicle, where x_M is given by

$$x_M = r_P \cos\theta_M. \tag{2}$$

It is also interesting to see by how much a lower altitude, h meters, reduces the ahead distance, x.

$$x = \frac{h}{\tan\theta_M}. \tag{3}$$

Figure 2, page 9 shows plots of the above three equations. From Figure 2, we would like a mount angle of 30°. This gives an ahead range of about 43 meters at the maximum altitude of 28 meters. The ahead range drops to 35 meters when the vehicle flys at 20 meters altitude.

Investigation of the existing nose cone and Psa916 bracket shows that we can achieve 27° without mechanical modifications, which we then select as the final mount angle.

4.2 Dvl Detects Bottom, Psa916 Does Not

Altitude has historically been computed using only Dvl measurements, since the forward-looking sonar was not available. If the Psa916 does not get an echo, the range computation will revert to the Dvl-only case.

The Dvl alone is insufficient for detecting obstacles ahead at low altitudes, since the forward beams are only $\pi/8$ (22.5°) ahead of vertical. At an altitude of 20 meters this is 8.3 meters, about the turning radius of the vehicle.

Up until December 2004, altitude was calculated by averaging the four beams to compute a range along the Dvl centerline, and then compensating the range for the vehicle attitude.

$$r = \frac{\cos\alpha}{N}\sum_{i=1}^{i=N} r_i \tag{4}$$

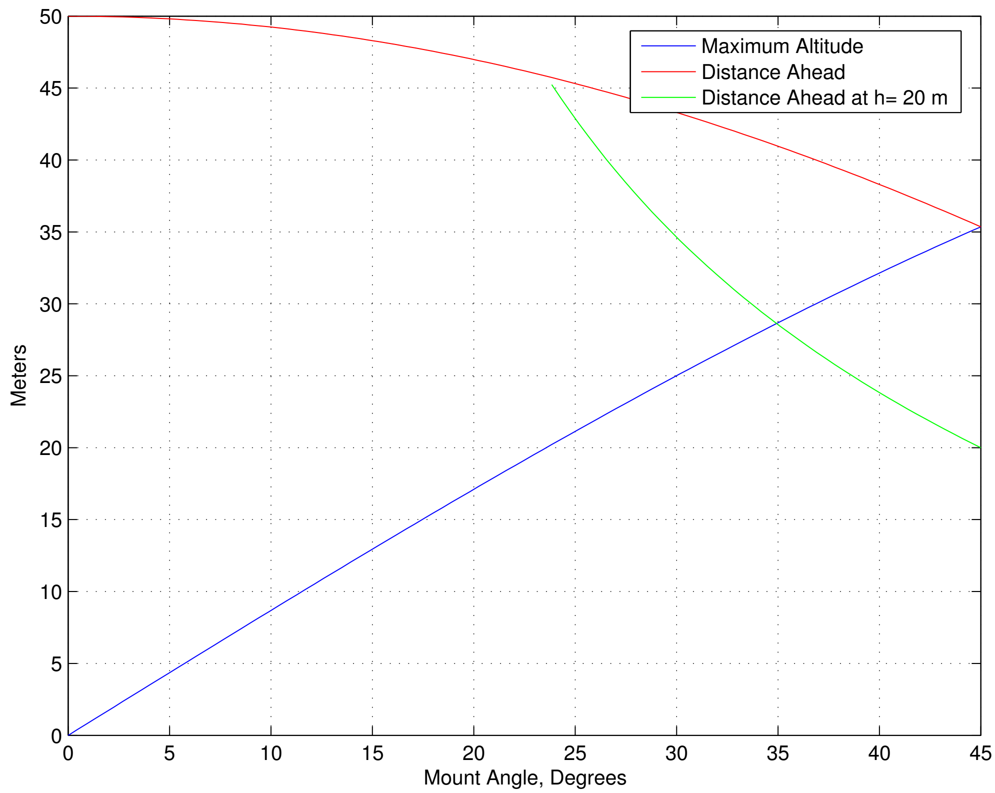

Figure 2: Maximum Altitude and Distance-Along-Bottom Versus Mount Angle

$$h = r\cos\theta\cos\phi. \tag{5}$$
$$\alpha = \frac{\pi}{6} \quad (30^\circ). \tag{6}$$

$0 < N \leq 4$ is the number of valid Dvl range measurements.

During the course of this project, this was discovered to be geometrically incorrect. The problem is illustrated by the planar case, in which the above Equations reduce to

$$r = \frac{(r_{Df} + r_{Da})\cos\delta}{2} \tag{7}$$
$$h = r\cos\theta \tag{8}$$
$$\delta = \frac{\pi}{8} \quad (22.5^\circ). \tag{9}$$

Here r_{Df} and r_{Da} are forward and after Dvl beams, respectively. See Figure 3, page 10.

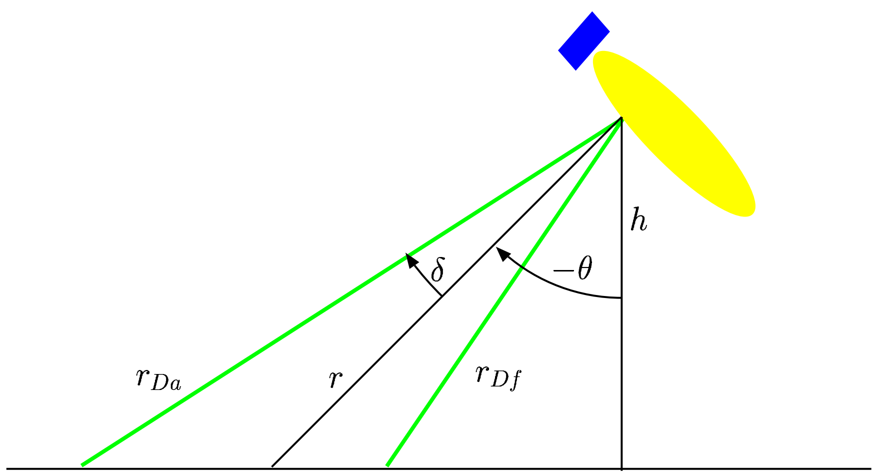

Figure 3: Altitude Computation of Equation 5

From planar mensuration formulae [8], the length of the bisector of the angle formed by the forward and after Dvl beams is

$$r = \frac{2 r_{Df} r_{Da}\cos\delta}{r_{Df} + r_{Da}}. \tag{10}$$

Equation 10 and Equation 7 are not the same. Since Equation 7 has been in use for quite some time, it is worth investigating the amount of error. It is apparent from Figure 3 that

$$h = r_{Df}\cos(\theta+\delta) \tag{11}$$
$$= r_{Da}\cos(\theta-\delta). \tag{12}$$

We can then get r_{Da} in terms of r_{Df},

$$r_{Da} = r_{Df}\frac{\cos(\theta+\delta)}{\cos(\theta-\delta)}. \tag{13}$$

Substituting Eqn 7 into Eqn 8 and subtracting Eqn 10 from the result from gives the altitude error. Dividing this by Eqn 10 to get a ratio, and simplifying,

$$h_e \quad = \quad \left(1+\frac{\cos(\theta+\delta)}{\cos(\theta-\delta)}\right)\frac{\cos\theta\cos\delta}{2\cos(\theta+\delta)}. \tag{14}$$

Figure 4 contains a plot of h_e. Thus, at a pitch angle of 30° and altitude of 100 meters the error is about 6 meters.

The altitude error is proportional to altitude, and so becomes proportionally smaller as the vehicle comes close to the bottom. Also, the error decreases with pitch angle, and goes to zero when the vehicle is flying horizontally over a level bottom.

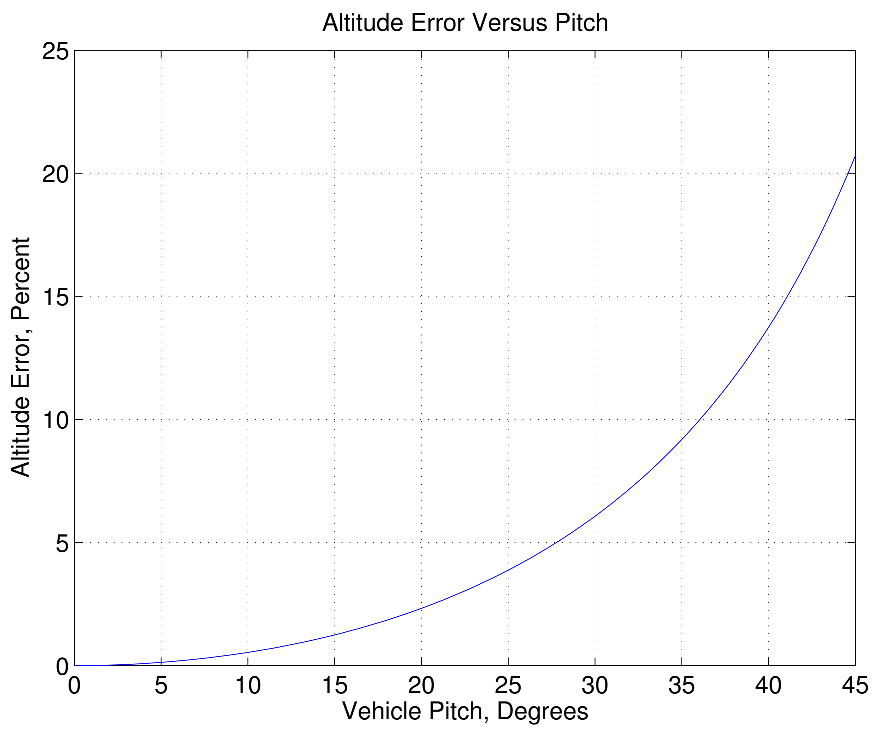

Figure 4: Error Between Old and New Altitude Calculations

We can easily generalize Equation 10 to the three-dimensional case. Let $\vec{b}_i$ be a unit vector along the i^{th} Dvl beam, coordinatized in the vehicle body frame. Then, from the Dvl geometry,

$$\vec{b}_1 = \begin{bmatrix} \frac{1}{2\sqrt{2}} \\ -\frac{1}{2\sqrt{2}} \\ \frac{\sqrt{3}}{2} \end{bmatrix}, \ \vec{b}_2 = \begin{bmatrix} -\frac{1}{2\sqrt{2}} \\ \frac{1}{2\sqrt{2}} \\ \frac{\sqrt{3}}{2} \end{bmatrix}, \ \vec{b}_3 = \begin{bmatrix} \frac{1}{2\sqrt{2}} \\ \frac{1}{2\sqrt{2}} \\ \frac{\sqrt{3}}{2} \end{bmatrix}, \ \vec{b}_4 = \begin{bmatrix} -\frac{1}{2\sqrt{2}} \\ -\frac{1}{2\sqrt{2}} \\ \frac{\sqrt{3}}{2} \end{bmatrix}. \tag{15}$$

The 2-1 Euler direction cosine matrix taking a vector in the body frame, B, into the Earth-fixed frame, N, is

$$T_{N/B} = \begin{bmatrix} \cos\theta & \sin\theta\sin\phi & \sin\theta\cos\phi \\ 0 & \cos\phi & -\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix}, \tag{16}$$

and then,

$$\vec{b}_N = T_{N/B}\,\vec{b}_B \tag{17}$$

$$= \begin{bmatrix} \cos\theta & \sin\theta\sin\phi & \sin\theta\cos\phi \\ 0 & \cos\phi & -\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix} \vec{b}_B \tag{18}$$

The altitude from the i^{th} beam is the third component of $\vec{b}_N$ in Equation 18, multiplied by the measured range,

$$h_i \triangleq h_{Ni_3} = r_i\left(-b_{Bi_1}\sin\theta + b_{Bi_2}\cos\theta\sin\phi + b_{Bi_3}\cos\theta\cos\phi\right). \tag{19}$$

Finally,

$$h = \frac{1}{N}\sum_{i=1}^{i=N} h_i, \tag{20}$$

where $0 < N \leq 4$ is the number of valid beams. Equations 4-5 were replaced by Equations 19 and 20 in onboard/Navigation/Dvl.cc, in December 2004.

Keep in mind that the altitude is computed from each beam under the assumption that the bottom is a horizontal plane. Averaging the four beams accounts for the case where the vehicle is over a sloping bottom.

4.3 Dvl and Psa916 Detect Bottom

There are basically two options should the Psa916 detect something ahead. First, and simplest, is to assume that there is a hill, and climb to maintain the commanded altitude. This can be achieved by combining the Psa916 and Dvl ranges in an extended altitude calculation, and then passing the altitude error to the depth controller. This is how altitude following works at present, but using only the Dvl altitude. Sections 4.3.1, 4.3.2, and 4.3.3 develop the new altitude calculation.

The second option is to make a brief slalom type maneuver to sweep the Psa916 beam back and forth to look for a way around the obstacle. This approach could lead to a more advanced path planning and obstacle detection algorithm. The drawback is that the vehicle must first maneuver blindly to sense what's ahead. This option is not pursued further in this report.

The following derivations of altitude are reduced to the planar case by first combining the lateral components of the Dvl measurements using Equation 10.

The following altitude computations differ from the previous ones in Equations 19-20 in an interesting way. The vehicle attitude (Euler) angles from the AHRS are not required. This is possible because below we compute the altitude perpendicular to the bottom, which doesn't depend on the attitude of the vehicle with respect to the vertical, it only depends on the attitude of the vehicle relative to the bottom.

The altitude calculation below is developed in three stages. The first stage, Section 4.3.1, incorporates all four Dvl beams. It became clear during the course of this derivation that a better way was to combine only the forward two Dvl beams with the Psa916, because the rear two beams are measuring altitudes of points on the bottom that the vehicle has already passed. This second-stage derivation is done in Section 4.3.2. The first stage is retained in the report because the math in the second stage builds upon it. Finally, a modification is covered in the last stage, Section 4.3.3.

4.3.1 First Computation of Altitude

The altitude h can be computed from the two measured ranges, r_P and r_D, as shown in Figure 5. Knowledge of the vehicle pitch is not required. Let r'_P and r'_D be the two lengths of the sides of the triangle, from the vertex to the ground. Let h' be the altitude of the vertex. Then from plane geometry,

$$h' = \frac{r'_P r'_D \cos\theta_M}{\sqrt{(r'_P)^2 + (r'_D)^2 - 2r'_P r'_D \sin\theta_M}}. \tag{21}$$

Then, assuming we know d, from Figure 5

$$r'_D = r_D + d\tan\theta_M \tag{22}$$

$$r'_P = r_P + \frac{d}{\cos\theta_M} \tag{23}$$

Now it remains to find the distance $h' - h$, which will then allow us to solve for h. To

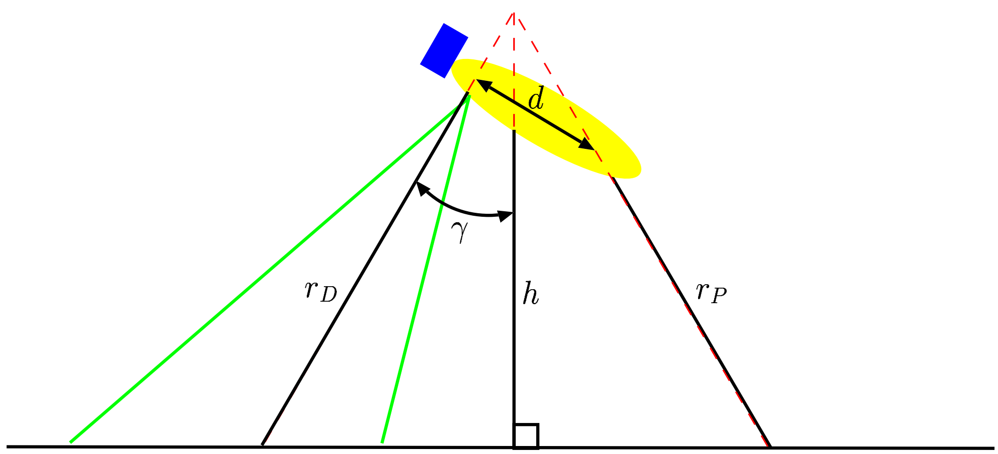

Figure 5: Altitude Computation of Equation 21

this end, the angle γ between the lines h and r_D is

$$\gamma \quad = \quad \arccos\left(\frac{h'}{r'_D}\right). \tag{24}$$

Then, since the lines d and r_D are perpendicular,

$$h' - h \quad = \quad \frac{d\tan\theta_M}{\cos\gamma}. \tag{25}$$

Substituting Equation 24 into 25, and solving for h gives

$$h = h' - \frac{r'_D\, d\tan\theta_M}{h'}. \tag{26}$$

For $|\gamma| < 15°$,

$$h \approx h' - d\tan\theta_M. \tag{27}$$

When $|\gamma| = 15°$, the error between Eqns 26 and 27 is 3.4%.

Figure 6 shows the geometry when the vehicle encounters a steep incline.

A drawback of the above is that when the vehicle approaches a steep incline, the rear two beams tend to drag the average down, which overestimates altitude. See Equations 19

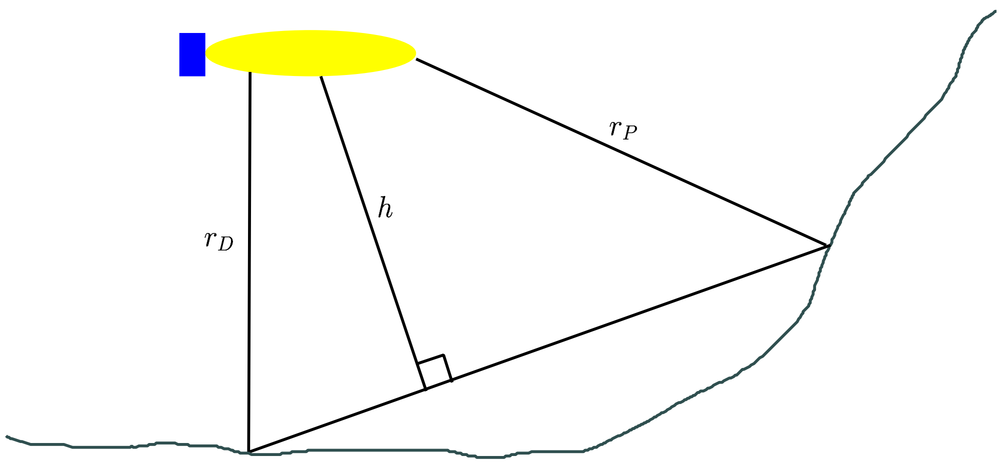

Figure 6: Geometry when AUV Encounters an Incline

and 20. A better way is to only use Dvl beams 1 and 3, the two forward-looking beams, in the computation. These beams are $\pi/8$ (22.5°) forward of the body-fixed z axis. Figure 7 shows this situation, where $\delta = \pi/8$.

We will now have to go back and rework the math.

4.3.2 Second Computation of Altitude

First, let r_D now be the average of the forward two beams, which are beams one and three, instead of all four of them.

Consider a triangle that has sides of length r'_P, r'_D, and c, with opposing angles A, B and C respectively. The law of cosines states that the length of side c is given by

$$c^2 = (r'_P)^2 + (r'_D)^2 - 2r'_P r'_D \cos C. \tag{28}$$

From plane mensuration formulae, the altitude of the triangle is

$$h' = \frac{r'_P r'_D \sin C}{c}. \tag{29}$$

Applying this to Figure 7, we see that

$$C = \pi/2 - \theta_M - \delta, \tag{30}$$

and that

$$\cos C = \cos(\pi/2 - \theta_M - \delta) \tag{31}$$
$$= \sin(\theta_M + \delta) \tag{32}$$
$$\sin C = \cos(\theta_M + \delta). \tag{33}$$

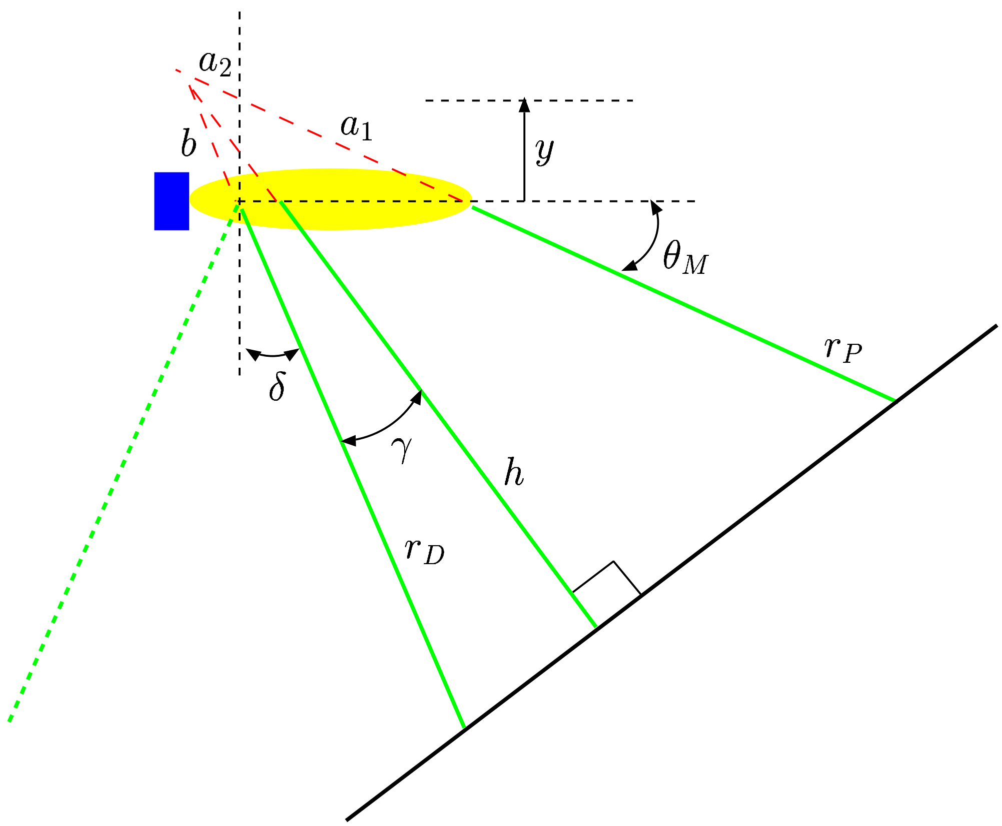

Figure 7: Altitude Computation Using Forward Two Dvl Beams

Substituting the positive square root of Equation 28 into 29, and then substituting Equations 32 and 33 into the result gives

$$h' = \frac{r'_P r'_D \cos(\theta_M + \delta)}{\sqrt{(r'_P)^2 + (r'_D)^2 - 2r'_P r'_D \sin(\theta_M + \delta)}}. \tag{34}$$

Let

$$a = a_1 + a_2, \tag{35}$$

then, from Figure 7,

$$r'_P = a + r_P, \tag{36}$$
$$r'_D = b + r_D. \tag{37}$$
$$\gamma = \arccos \frac{h'}{r'_D} \tag{38}$$

Beware that the definitions of r'_D and γ are different than in the previous section, which can be seen by comparing Figures 5 and 7.

Now it remains to find the altitude h, which will require us to find the lengths of the sides a and b. Again from Figure 7 it is apparent that

$$\sin\theta_M = \frac{y}{a_1}, \tag{39}$$
$$\cos\theta_M = \frac{d}{a_1}, \tag{40}$$
$$\tan\theta_M = \frac{y}{d}. \tag{41}$$

The above Equations yield solutions for a_1 and y,

$$a_1 = \frac{d}{\cos\theta_M} \tag{42}$$
$$y = d\tan\theta_M. \tag{43}$$

Applying the Law of Sines to Figure 7,

$$\frac{b}{\sin(\theta_M + \pi/2)} = \frac{y}{\sin(\pi - \theta_M - \pi/2 - \delta)}. \tag{44}$$

Substituting Equation 43 into the above and solving for b,

$$b = \frac{d\sin\theta_M}{\cos(\theta_M + \delta)}. \tag{45}$$

The Law of Sines also gives a solution for a_2,

$$a_2 = \frac{d \tan\theta_M \sin\delta}{\cos(\theta_M + \delta)} \tag{46}$$

Substituting Equation 46 and Equation 42 into 35,

$$a = \frac{d \tan\theta_M \sin\delta}{\cos(\theta_M + \delta)} + \frac{d}{\cos(\theta_M)} \tag{47}$$

Finally, the Law of Sines gives the length $h' - h$,

$$\frac{h' - h}{\sin(\pi/2 + \delta)} = \frac{b}{\sin(\pi - \gamma - \delta - \pi/2)}. \tag{48}$$

Solving for h,

$$h = h' - \frac{\cos\delta}{\cos(\gamma + \delta)} b. \tag{49}$$

Substituting Equation 45 into the above,

$$h = h' - \frac{\cos\delta}{\cos(\gamma + \delta)} \frac{d \sin\theta_M}{\cos(\theta_M + \delta)}. \tag{50}$$

4.3.3 Further Modification of Computation of Altitude

It turns out that Equation 50 is not quite what we want when γ is large. For example, suppose that the vehicle is approaching a vertical wall. Figure 7 shows that, in this case, h' is the distance from the apex of the triangle formed by a_2 and b, and is parallel to the vehicle centerline. h doesn't even exist, since it is defined to be the portion of h' below the vehicle centerline. What we want instead is the distance from the Dvl head, parallel to the h of Equation 50, which we call h_D. See Figure 8.

h_D can be found relatively easily. From Figure 8,

$$h_D = h' - b \cos\gamma. \tag{51}$$

An alternative form is

$$h_D = r_D \cos\gamma. \tag{52}$$

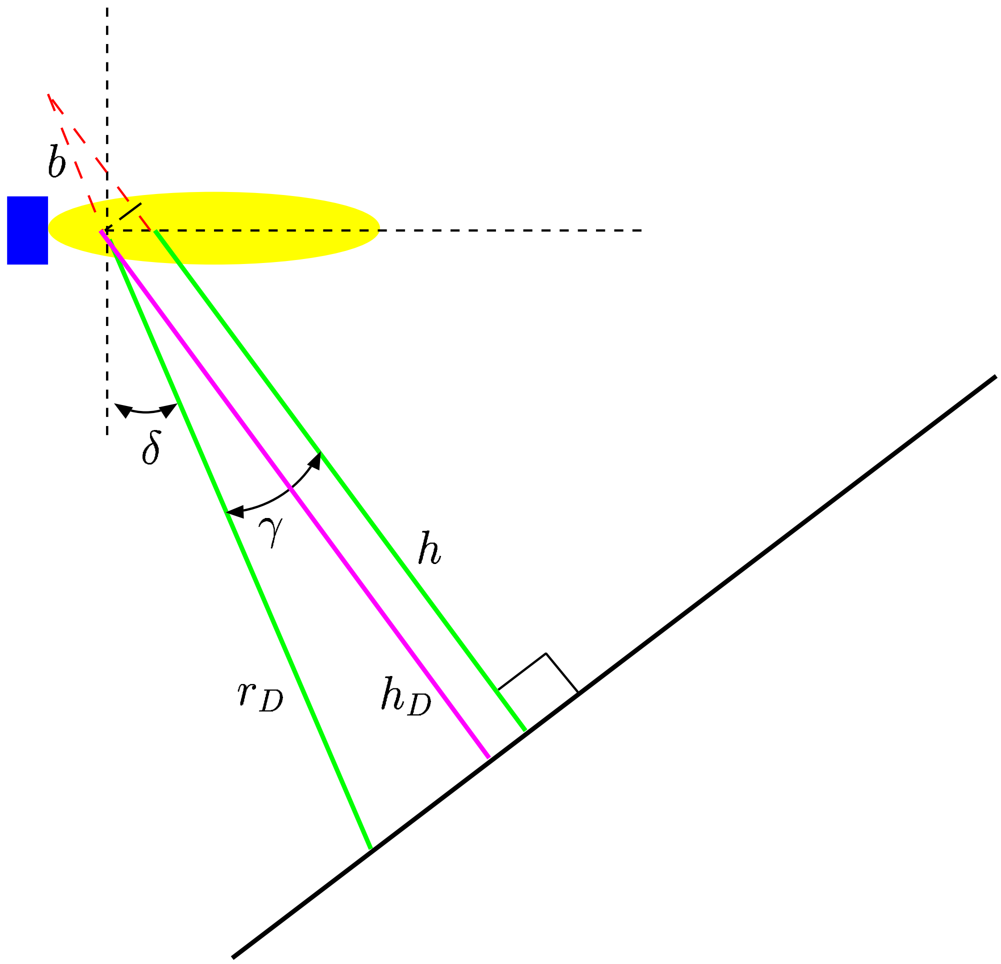

Figure 8: Definition of h_D, Parallel to h

4.3.4 Simulation Results

During the course of this project the Qnx simulation was modified to simulate Dvl and range-finder sonar measurements over a non-flat bottom. This involved adding coordinate transformations similar to Equation 18. The simulation also required a root solver to find the length of each beam; this was successfully accomplished by implementing a version of the classic bisection algorithm.

The simulation was and is very useful in visualizing different altitude-tracking situations, and illustrating and isolating various design issues.

The upper strip of Figure 9 shows the vehicle passing over a piece-wise linear ridge that is 20 meters high. The green lines show the simulated bottom, and the black curve (separating blue from white) shows where the vehicle thinks the bottom is, using the combined Dvl and Psa altitude Equation 50. The magenta curve shows the estimated bottom using the original Dvl-only Equations 4 and 5. The two bottom-depth estimates result from summing vehicle depth with estimated altitude. The orange trajectory shows the path of the vehicle, which is in the process of diving to a commanded depth of 60 meters, when it detects the ridge ahead, and reacts by reversing the dive and starting a slow climb. The response time of the depth control loop is an important issue and is discussed further in Section 5. This section concentrates on altitude estimation.

Figure 9 shows a number of interesting aspects. First, the error in the original altitude computation (magenta trace, top strip) of Equations 4 and 5 is apparent at about 20 meters distance (horizontal axis), where it makes a step jump up to 75 meters. This is because the back two Dvl beams have exceeded their maximum range, and drop out of the calculation. Only the forward two beams are valid, and these are shorter since the vehicle is pitched forward. Obviously this jump does not reflect the real (simulated) bottom in green. This is part of the problem with the original calculation in Equations 4 and 5, and is corrected in the new method in Equation 50. The other part of the problem is the geometric error, as discussed in Equation 14 and Figure 4. This is apparent after 300 meters along the bottom, where the vehicle pitches up to ascend, and the estimated bottom drops below the actual. The magenta and black curves are coincident here because the Psa916 has exceeded its (simulated) maximum range of 120 meters, and so the altitude computation reverts to the original ones in Equations 19-20.

The Dvl beams have a maximum range of 200 meters. The second strip of Figure 9 shows the individual beam ranges. The Dvl beam lengths are set to zero when the distance exceeds the maximum range because the real instrument does this. The Psa916 outputs its maximum range when it doesn't receive an echo.

At about 80 meters, just after the rim of the incline, the original altitude computation overestimates altitude because the rear two Dvl beams are dragging the estimate down. This averaging effect can be seen in reverse at at about 175 meters where the vehicle passes over the descending segment of the ridge.

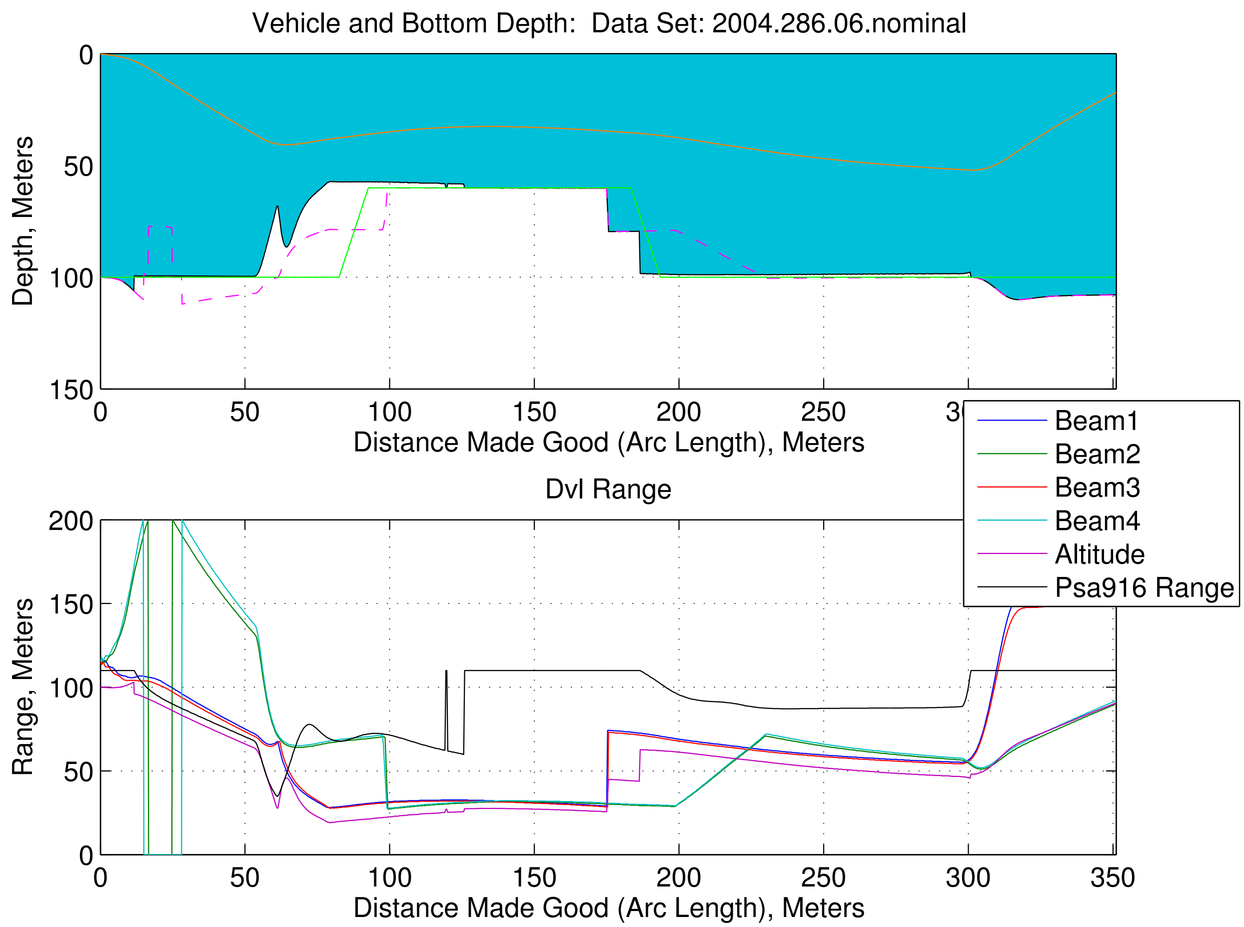

Figure 9: New Psa/Dvl Bottom Estimate Computed With Equation 50 (Black Line, Top), and Original Dvl-Only Bottom Estimate of Equations 4 and 5 (Magenta Dashed Line, Top)

The peak in the new altitude computation at about 60 meters occurs when the Psa916 beam passes over the edge of the incline on to the level part, while the Dvl beams are still intersecting the inclined segment. The line connecting the two points then passes through the cliff edge, causing the altitude estimate to increase. This artifact, while not desirable, is acceptable since the algorithm is still predicting an altitude decrease before the vehicle reaches the wall. Bear in mind that we are attempting to derive information about arbitrarily complicated topography by only measuring two points on the surface, so some error is inevitable.

Another issue here is at 100 meters distance, where the vehicle starts to pass over the flat top of the ridge, the new altitude estimate is in error by around two meters, yet both Dvl beams and the Psa916 beam are intersecting a flat surface. This is caused by the definition of altitude in Equation 50, which was used in this simulation run, and is corrected by the improved altitude calculation of Equation 51.

Figure 10, page 23, shows the same simulation, but with the new altitude Equation 51 replacing 50, and with the corrected Dvl-only altitude Equations 19 and 20 replacing Equations 4 and 5. The improvements are apparent.

4.4 Psa916 Internal Filter

The Psa916 contains an internal first order FIR filter given by

$$r_P(k) = \frac{1}{4} r_{PM}(k) + \frac{3}{4} r_{PM}(k-1) \tag{53}$$

where r_{PM} is the measurement and r_P is the filtered value. This algorithm additionally rejects any new measurement that "differs significantly" [3] from the previous one. This outlier rejection scheme probably contributes to the tendency of the Psa916 to lock on to the side-lobe return, as discussed in Section 6.2, and demonstrated in Figure 21, page 38. See the accompanying text for explanation.

5 Control: Underdamped Depth/Pitch Transient

The AUV has had a longstanding problem with depth control in that the transient response is dominated by a very long, underdamped oscillation. This has been well predicted by the linear control and dynamics model. See Figure 11, page 24.

Section 6 compares results from this analytic mathematical model with the real system.

Figure 12 shows a diagram of the depth control loop model that produces Figure 11.

Equation 54 below is the linearized heave and pitch dynamics referenced in the block diagram of Figure 11. See [6] and [5] for the derivation and list of symbols. See Appendix A for the list of numerical values.

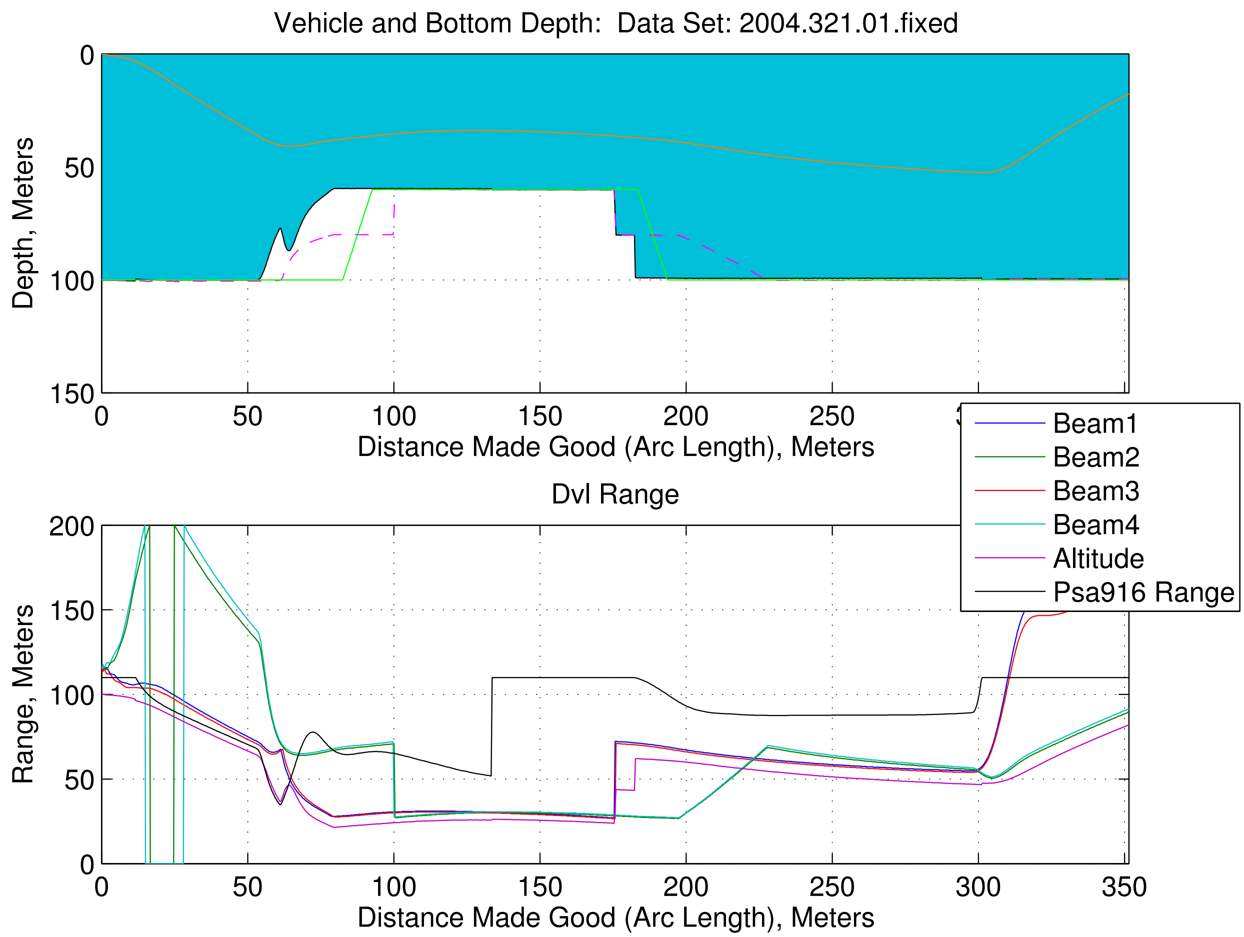

Figure 10: New Psa/Dvl Bottom Estimate Computed With Equation 51 (Black Line, Top), and Corrected Original Dvl-Only Bottom Estimate of Equations 19 and 20 (Magenta Dashed Line, Top)

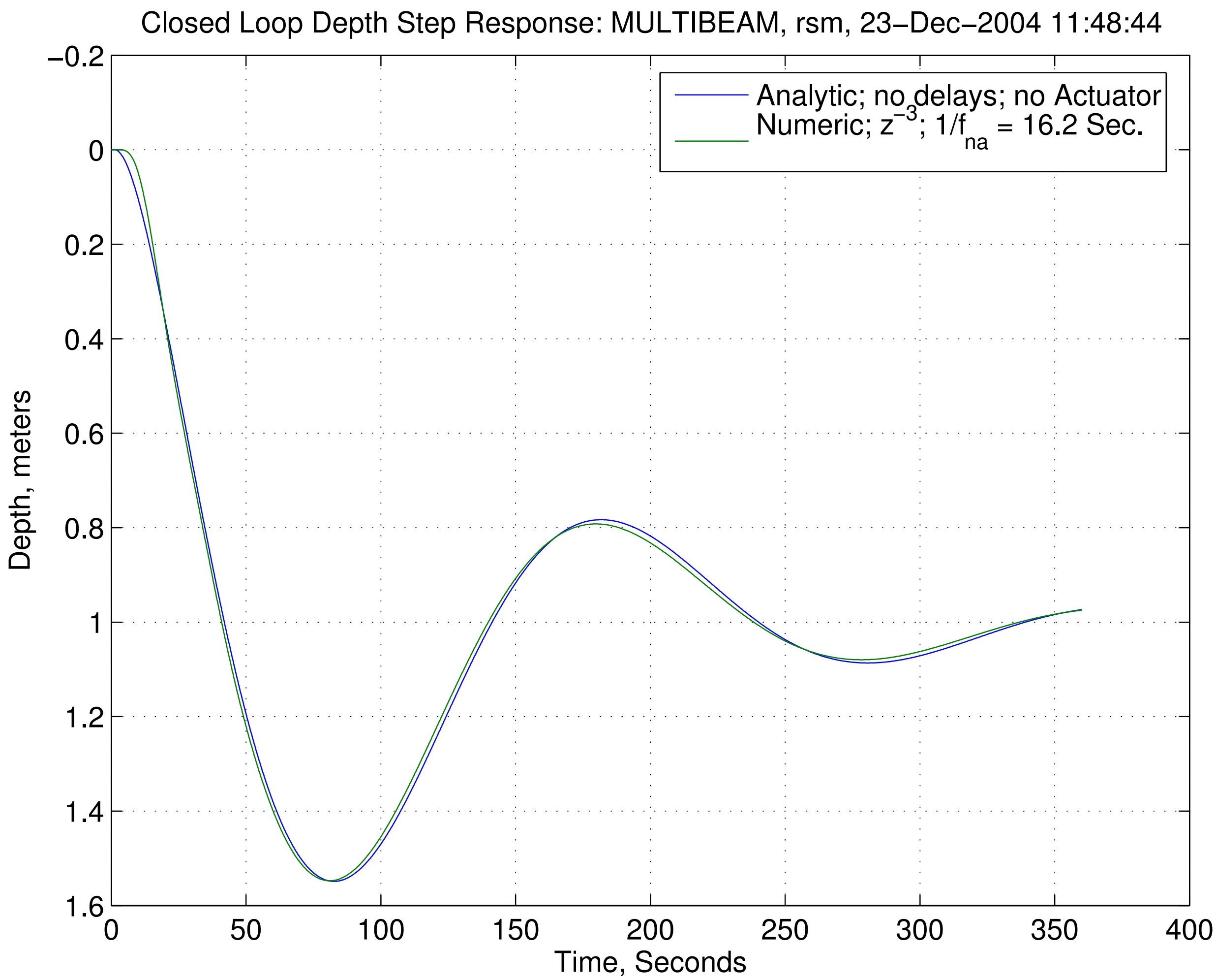

Figure 11: Closed Loop Step Response of Depth Control Loop Predicted by Figure 12 and Equation 54

.

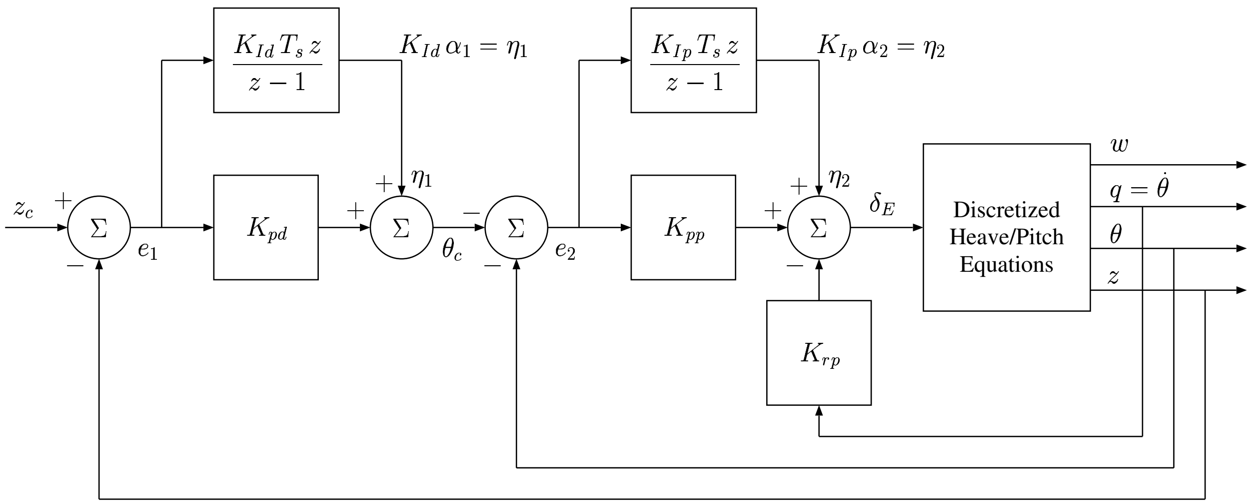

Figure 12: Block Diagram of the Depth Control Loop

Heave/Pitch

$$
\begin{bmatrix} m - Z_{\dot w} & -mx_G - Z_{\dot q} & 0 & 0 \\ -mx_G - M_{\dot w} & I_{yy} - M_{\dot q} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot w \\ \dot q \\ \dot z \\ \dot\theta \end{bmatrix}
$$
$$
= \begin{bmatrix} (Z_w - Z_{RF})U & (Z_q + m + x_R Z_{RF})U & 0 & 0 \\ (-M_w + x_R Z_{RF})U & (-M_q - mx_G - x_R^2 Z_{RF})U & 0 & W(z_B - z_G) \\ 1 & 0 & 0 & -U \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} w \\ q \\ z \\ \theta \end{bmatrix}
$$
$$
+ \begin{bmatrix} -(Z_R U^2 + T_P) \\ (Z_R U^2 + T_P)x_R \\ 0 \\ 0 \end{bmatrix} \delta_E \tag{54}
$$

Due to the axial symmetry of the hull we have:

$$Z_w = Y_v \tag{55}$$
$$Z_q = -Y_r \tag{56}$$
$$M_w = N_v \tag{57}$$
$$N_r = -M_q \tag{58}$$
$$Z_R = Y_R \tag{59}$$
$$Z_{RF} = Y_{RF}. \tag{60}$$

In addition to the fourth order rigid body dynamics in Equation 54 above, the model contains a second order elevator actuator model in series with the δ_E shown in Figure 12. The actuator model is developed in Section 5.1, page 33. After taking the Z transform, we add the two accumulators as shown in Figure 12, and also a three sample period delay. The final system is consequently 11^{th} order.

Figure 13 shows a root locus of this model, with gain variation at the input δ_E. The transient response will be dominated by the poles and zeros near $z = 1$, and so an enlargement of that region is shown in Figure 14, and a further enlargement in Figure 15. Although individual gain changes will alter the shape of the locus, this locus gives a good indication as to what the effect will be of overall gain increases or decreases. Also, it produces the gain margin at the input, which is the traditional measure of robustness.

It is helpful to first understand where the open-loop poles lie. The three poles near the $z = 0$ origin are time delays. The rigid-body dynamics have poles at

```
1.0000
0.7455
0.9972
0.9260.
```

The actuator model has its two poles at $z = .9253$, and finally the two integrators add a double pole at $z = 1$.

We can see that the open-loop rigid body dynamics are not oscillatory.

The pole at $z = 1$ is from the kinematic equation $\dot{\theta} = q$.

The complex zero pair, shown in Figure 15, is problematic, because it attracts two poles at a low frequency and damping. As long as the two integrators are present, experience shows that the zero pair doesn't move much. As the gain increases, the actuator pole pair at $.977 \pm j.055$ (magenta diamonds in Figure 14) continue to become more oscillatory and underdamped. This response shows up primarily in pitch. The outer depth control loop has a much lower bandwidth and tends not to respond to pitch oscillations at this frequency.

The depth control in Figure 12 has two integrators, one for the inner pitch control loop, and one for the outer depth control. This arrangement is convenient for switching between pitch control (outer loop is deactivated) and depth control. However, when the depth control is active, the pitch integrator is unnecessary, because the depth control integrator will still force the pitch to be whatever it needs to be to maintain zero steady-state depth control error. Since the pitch integrator adds a pole at the origin, it tends to degrade the transient performance. So, we will remove it for depth control.

Additionally, the (new in the Mapping Vehicle) Kearfott navigation system provides depth rate, which will provide damping for the depth control loop.

This revised depth control system is shown in Figure 16, with the new damping feedback in red.

A new gain set was determined for the configuration in Figure 17.

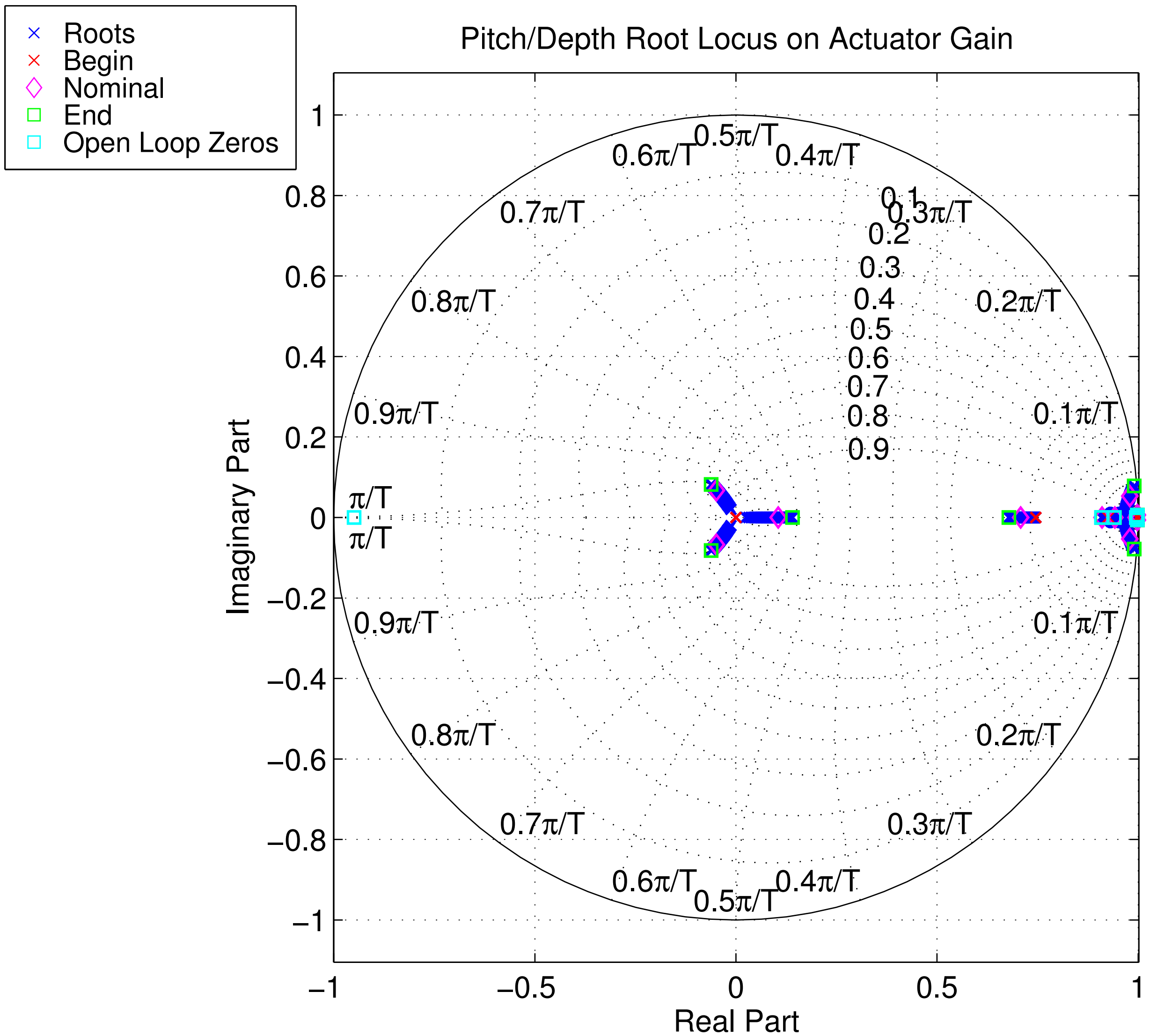

Figure 13: Root Locus of the Depth Control Loop

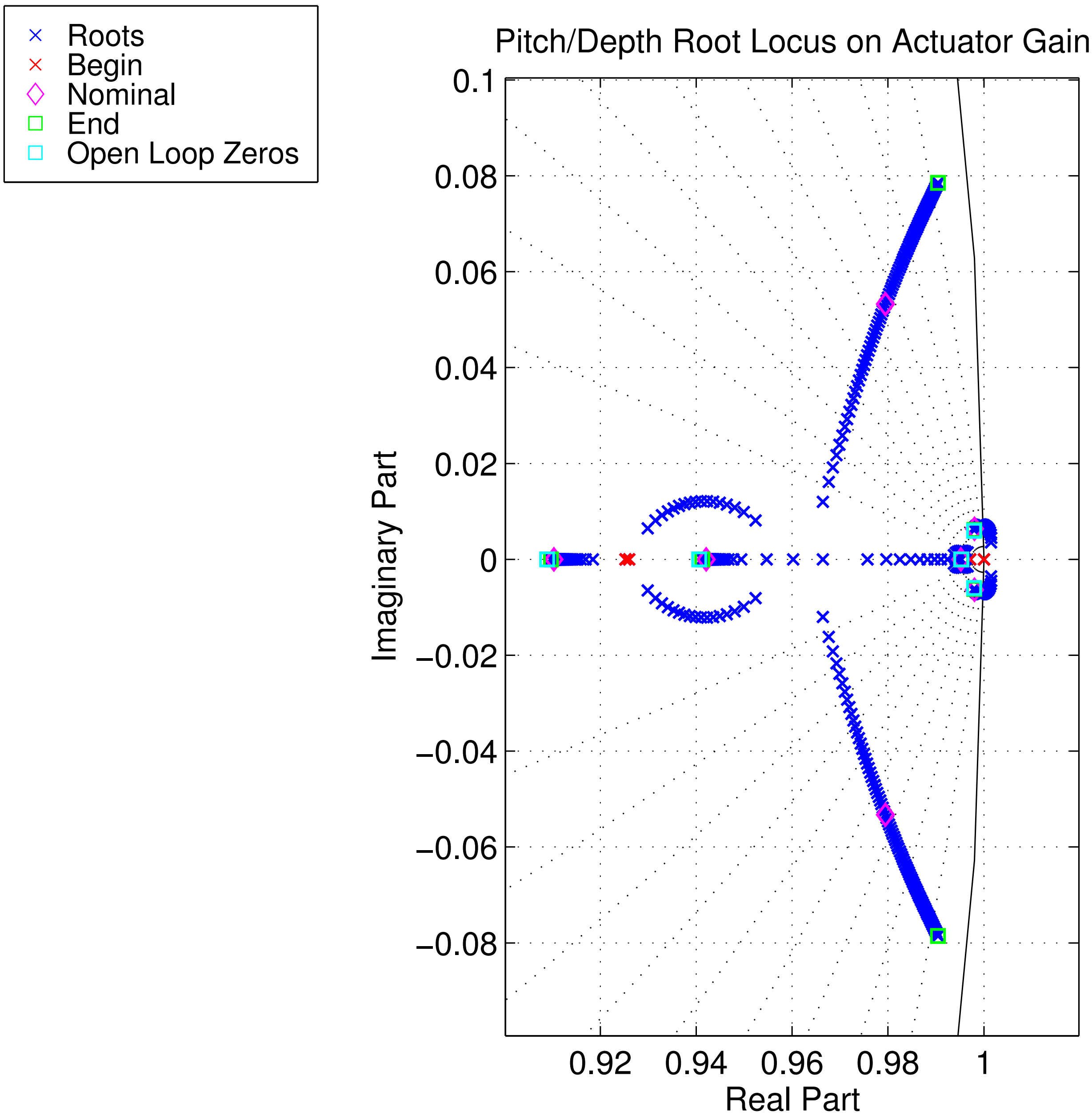

Figure 14: Root Locus Plot of Figure 13 Enlarged at $z = 1$

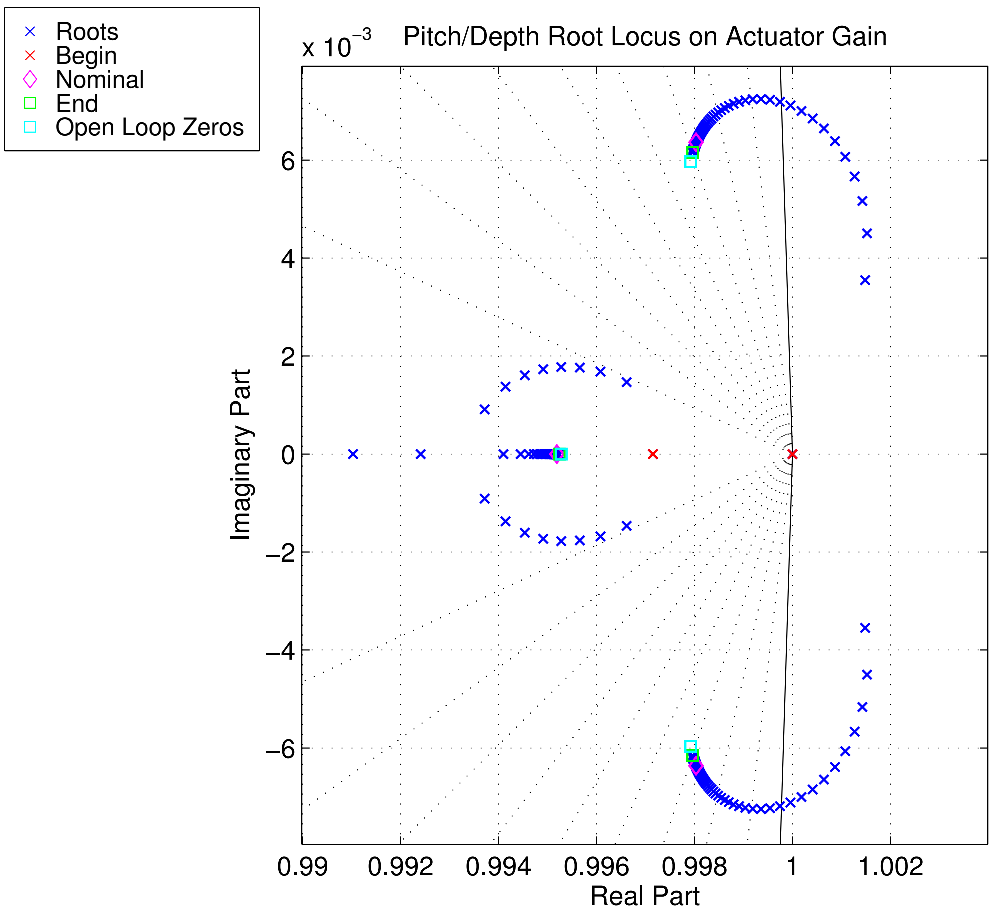

Figure 15: Root Locus Plot of Figure 14 Enlarged Further

.

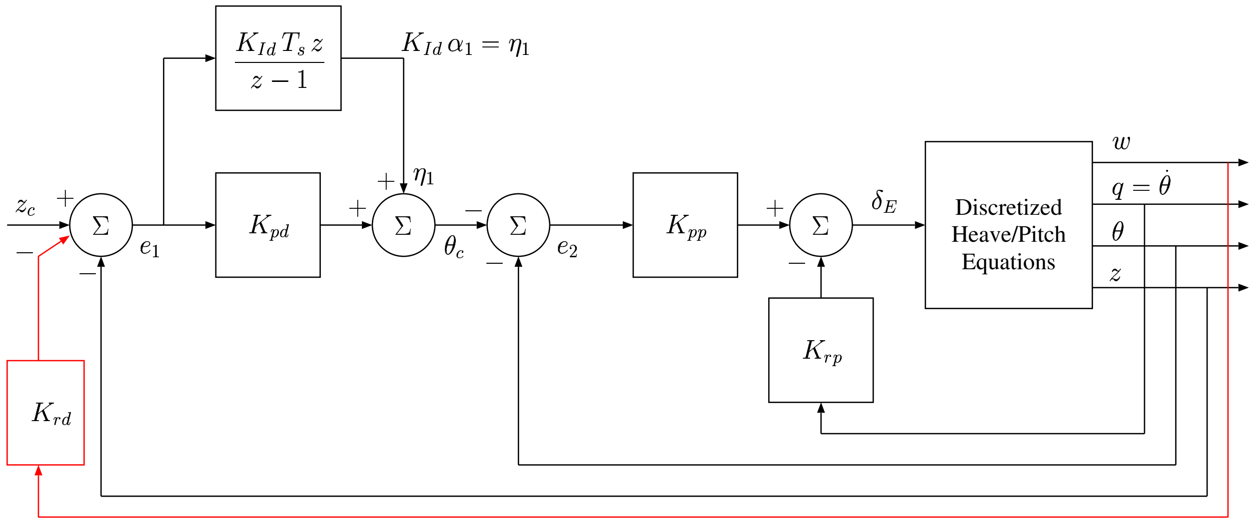

Figure 16: Revised Depth Control Loop

```
0.000000        : (Kip) Pitch integral gain, 1/sec.
0.060000        : (Kpd) Depth position gain, rad/meter.
0.000600        : (Kid) Depth integral gain, rad/meter/sec.
0.120000        : (Krd) Depth rate gain, rad-sec/meter.
```

The original gains are on page 45. The new gains shown above were determined by Ziegler-Nichols style trial-and-error with the analytic model of Figure 16, followed by at-sea tuning with the vehicle.

Figure 17 shows the depth step response of the new control system of Figure 16. Comparison of Figure 17 to Figure 11 shows that the model predicts a major improvement in transient response.

Figure 18 shows the root locus of the new system. Comparing this to Figure 14 we see that the problematic zero pair has moved further into the unit circle, and increased in frequency, and consequently so has the dominant pole pair.

The actuator pole pair at $.977 \pm j.055$ in Figure 14 has moved only slightly to $.973 \pm j.048$ in Figure 18, indicating that the pitch control loop has retained its underdamped transient response.

Section 6 compares similar step responses of the actual vehicle.

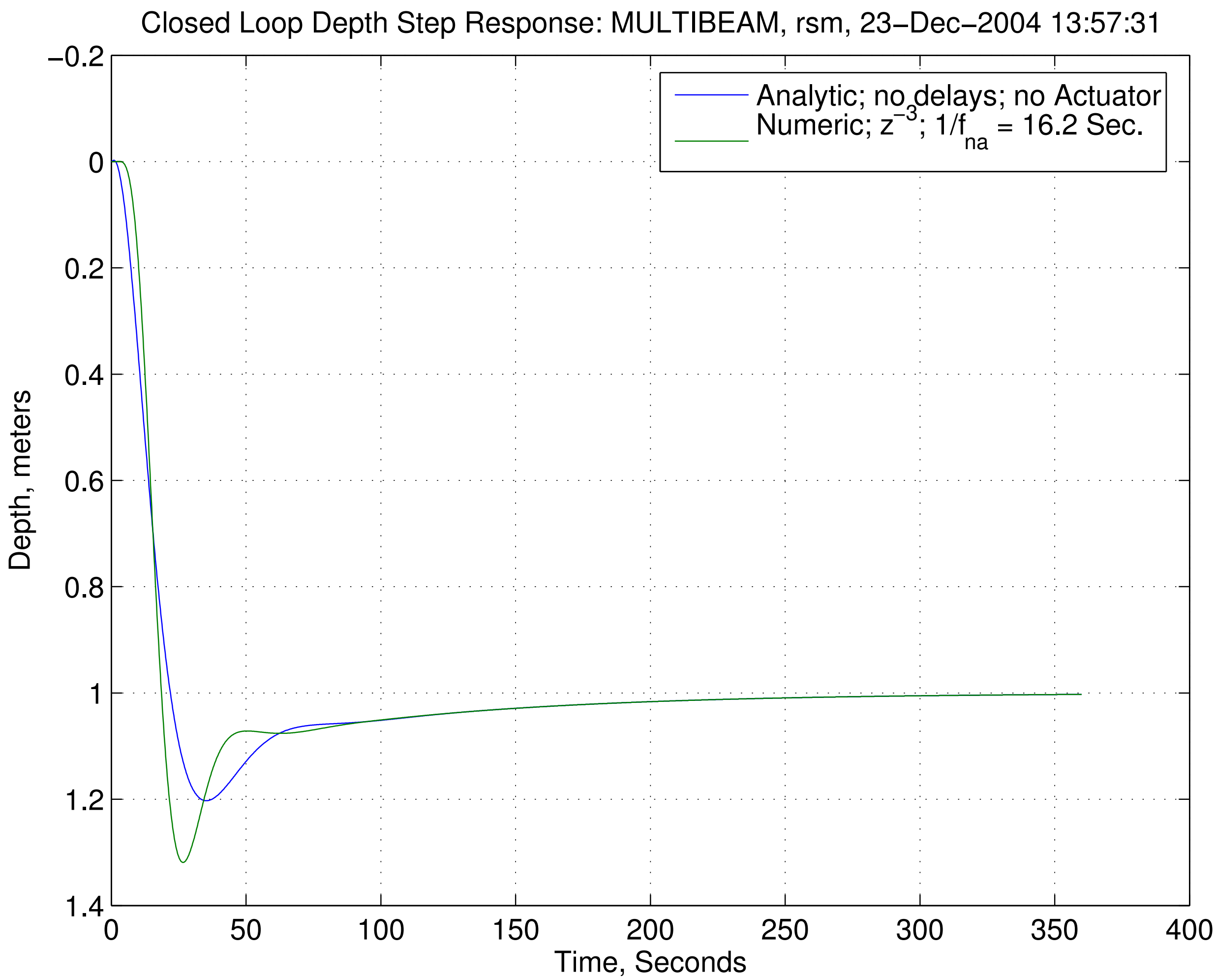

Figure 17: Closed Loop Step Response of Depth Control Loop Predicted by Figure 16 and Equation 54

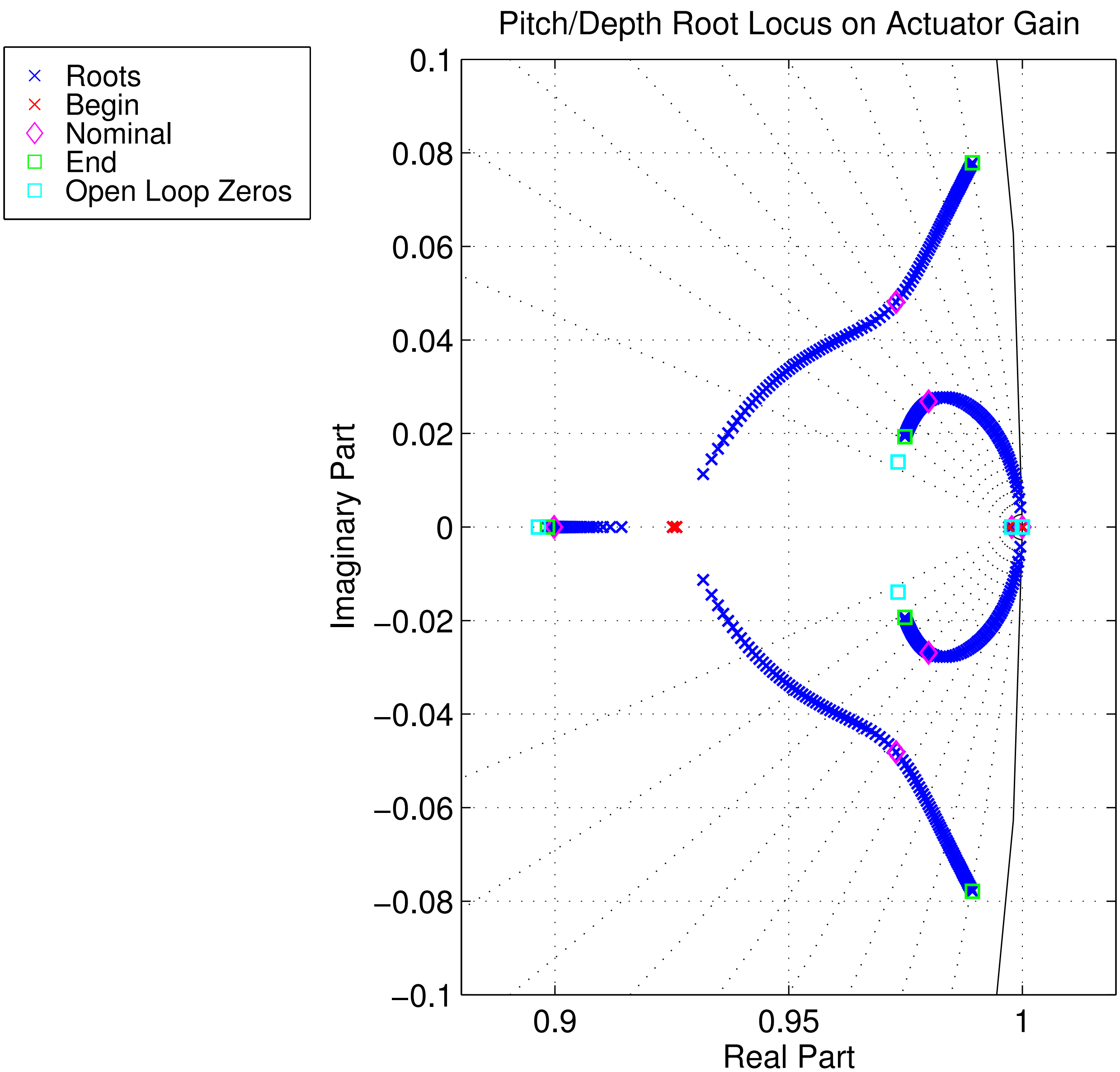

Figure 18: Root Locus of Depth Control Loop of Figure 16 and Equation 54

5.1 Tailcone Actuator Model and Bandwidth

The tailcone elevator actuator is a stepper motor that rotates the elevator at a maximum rate of

$$\dot{\delta}_{EM} = 15°/\text{sec}. \tag{61}$$

The motor controller parameters were selected so that the motor can maintain this rate independent of the expected maximum load. The problem now is how to model the actuator with a linear system, since the bandwidth of a linear model will depend on the amplitude of the input. To clarify this, imagine that the elevator has a sinusoidal commanded input,

$$\delta_{Ec} = A \sin(\omega t). \tag{62}$$

The actuator will then try to move the elevator with rate

$$\dot{\delta}_E = \omega A \cos(\omega t), \tag{63}$$

and so the maximum rate that the motor can achieve is

$$\dot{\delta}_{EM} = \omega A. \tag{64}$$

Experience with the vehicle shows that during straight and level flight, the elevator motions are slight, and not ususally more than $\pm 1°$. Substituting in the numerical values above,

$$\omega_M = \frac{\dot{\delta}_{EM}}{A} = \frac{15}{1} = 15 \ \text{rad/sec}, \tag{65}$$

or about 2.4 Hz. To summarize, the elevator can track a sinusoid with amplitude $\pm 1°$, and with frequency up to 2.4 Hz before it begins to rate limit.

Consequently, we choose a second order transfer function as the actuator model. The elevator does not overshoot, so $\zeta = 1$.

At this point, the actuator model doesn't match the observed performance of the pitch control loop. Pitch control has an underdamped transient response with a period of about 16 seconds. The pitch control model of Figure 12 will produce a similar response if the actuator bandwidth is set to

$$\omega_M = .25 \ \text{rad/sec}. \tag{66}$$

This is drastically lower than the value determined by the analysis of Equation 65. There are two reasons that might explain this. First, the elevator has been observed to have about $0.5°$ of play, due to the play in the linkages with the stepper motor. This is very large given that nominal control motions are only twice that. Also, the tailcone chassis is not perfectly rigid, and the elevator control surface bends noticably without applying much external force. These two causes are flagged in the conclusion to be investigated further.

The actuator model is then

$$\frac{\delta_E}{\delta_{Ec}} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n + \omega_n^2} \tag{67}$$

$$\omega_n = \frac{\omega_M}{\sqrt{\sqrt{2} - 1}} \tag{68}$$

$$\omega_M = .25 \ \ \text{rad/sec} \tag{69}$$

$$\zeta = 1.0. \tag{70}$$

6 Sea Trial Results

The AUV team conducted sea trials on 6 December 2004. There were four objectives:

1. Test the new depth control and adjust the control gains.

2. Determine the range of the forward-looking Psa916 sonar.

3. Test the new altitude calculation that includes the Psa916, given in Equations 50 and 51.

4. Test the corrected version of the original altitude calculation, given in Equations 19 and 20

An additional objective, not part of this project, was to test the new shipborne dropweight watchdog algorithm. This new algorithm development resulted from the near loss of the vehicle on 15 October 2004 when it failed to drop its weight, after sinking to the bottom during our first attempt at the tests listed above. A tenacious air bubble had remained in the vehicle during the ballasting procedure in the test tank, and didn't dispel until the first mission at sea, causing the vehicle to lose buoyancy.

6.1 Depth Control

Figure 19 shows the depth step response of the Mapping Vehicle with the orginal gains on Page 45. It corresponds well with the linear model's step response of Figure 11, although the natural frequency of the model's response is somewhat higher.

Figure 20 shows a similar depth step but with the new gains determined by the analysis of Section 5. This again compares well with the predicted response of Figure 17. The model's response is a little faster and has more overshoot, however, the real system has a pitch limit of

±30°, which was reached between 740 and 750 seconds in Figure 20. This partially explains the increased response time and decreased overshoot of the real system.

Comparision of Figures 19 and 20 show that the analysis and gain changes of Section 5 has resulted in a dramatic improvement in depth control.

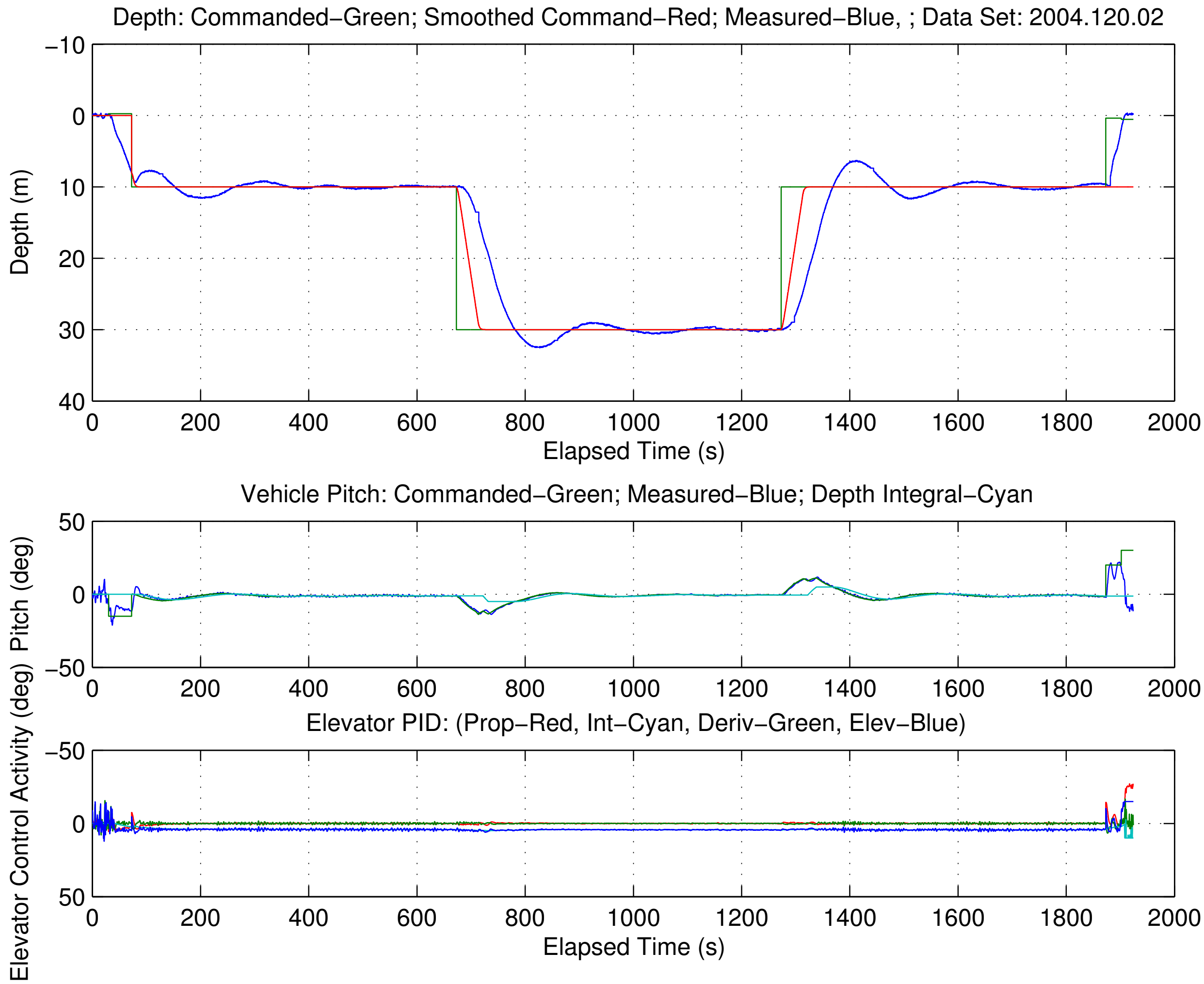

Figure 19: Depth Step Response with the Original Gains Given on Page 45

6.2 Psa916 Range

Unfortunately the Psa916 (S/N 636) was ineffective and, to be blunt, practically useless, during the sea-trials, which lasted only a single day. It is possible that future operational procedures, or mechanical modifications, such as baffling, would improve performance.

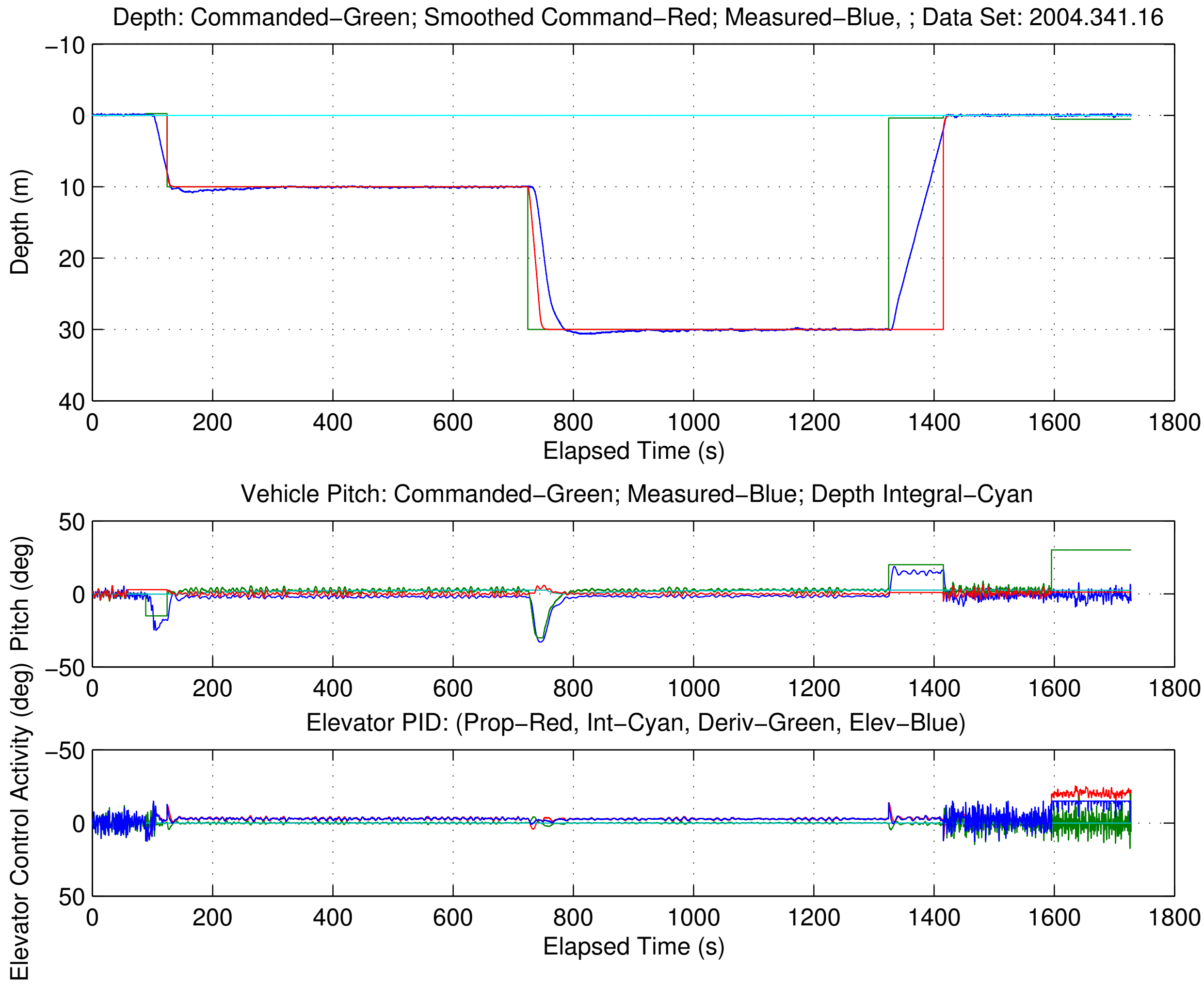

Figure 20: Depth Step Response with the News Gains Given on Page 30

The Psa916 range-measurement missions during the sea trials did bring to light several related issues:

1. Using the forward-looking sonar to detect the bottom over relatively level ground may not be feasible. Apparently much of the transmit energy continues ahead because of the low angle of incidence, making the return echos faint. This problem may be solvable, perhaps with a different sonar, and requires further investigation.

2. The Psa916 is vulnerable to a dangerously misleading side-lobe return, as discussed below. Account for this problem during future OA algorithm development.

3. The Dvl altitude measurement and calculation still has significant error and needs further investigation.

Figure 21, page 38 shows the AUV flying an altitude-following out-and-back mission over a relatively flat bottom. The black trace in the bottom strip shows very poor performance by the Psa916. The Psa begins to get returns around 2 km on the horizontal axis, but the range is disturbingly wrong. The Dvl is measuring an altitude of 8.8 meters. The Psa916 is mounted 27° down from the vehicle centerline, so the Psa916 should measure a range of around $8.8/\sin 27° = 19$ meters. Yet it is reporting 11 meters. The most likely reason for this is that it is detecting a return from a sidelobe, which is pointed about 26° down from the Psa916 line of sight.

Since the primary sonic pulse strikes the seafloor at a large oblique angle, most of the energy continues forward, and consequently the return is faint. Also, the sidelobe pulse will strike the bottom at a more vertical angle, which reflects a higher percentage of the energy back. The point at which the sidelobe pulse strikes the ground is closer than the primary spot, so less energy is lost to attenuation. Finally, the Psa916 electronics are fairly simple and just take the first echo, and discard subsequent echos. These reasons indicate that the Psa916 is returning a false reading from the sidelobe during the altitude track portion of the mission, particularly around 2 km in Figure 21.

A sidelobe return from the surface is evident in the lower strip when the vehicle begins its ascent at 2160 meters. This dangerously shows decreasing range to a surface, when in fact the range from the bottom is increasing.

Figure 22, page 39 shows that the vehicle did not return by quite the same path during this mission, accounting for the asymmetry of the bottom in Figure 21.

To determine if the sidelobe problem would be solved by a more perpendicular surface, we ran an out-and-back yoyo mission, shown in Figure 23, page 40. The middle strip shows the vehicle pitching down at 30°, meaning that the angle of incidence of the Psa916 should be 57°. The bottom strip of Figure 24, page 41 shows the Psa916 data. This strip has been enlarged to show a single up and down cycle from 1300 to 1600 meters.

Several things are apparent from the bottom strip. As the vehicle ascends, the range of Dvl beams 2 and 3 increases, and when the vehicle dives the range of 1 and 4 increase,

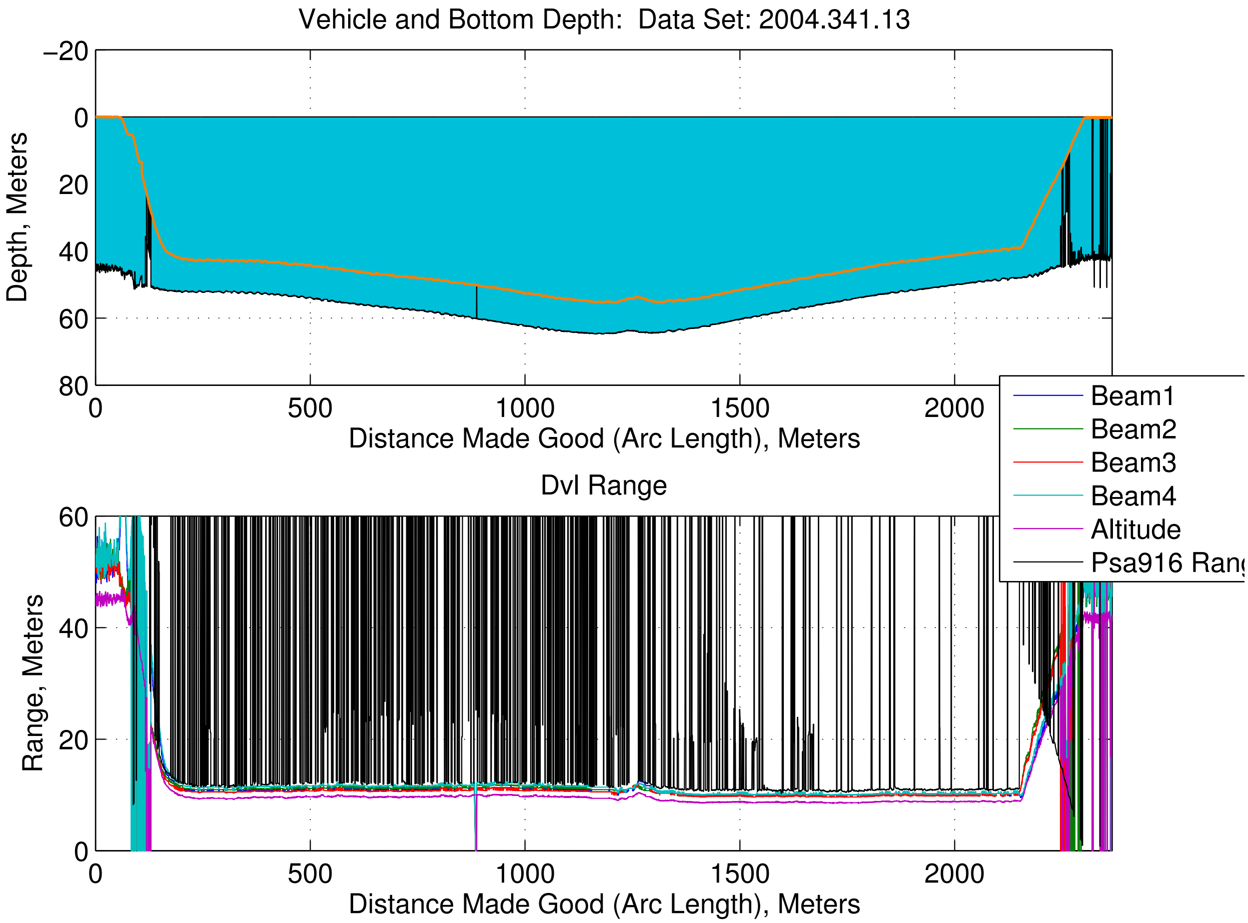

Figure 21: Ten Meter Altitude-Track Run

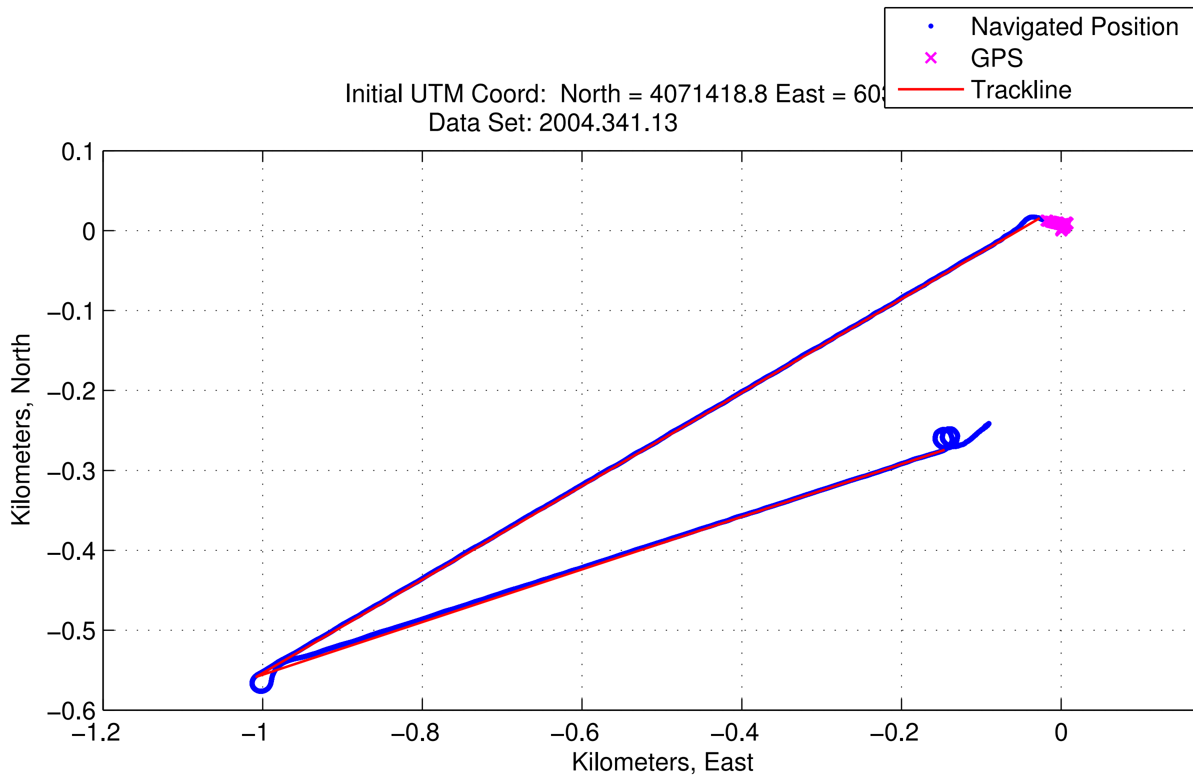

Figure 22: Vehicle Position for Ten Meter Altitude-Track Run

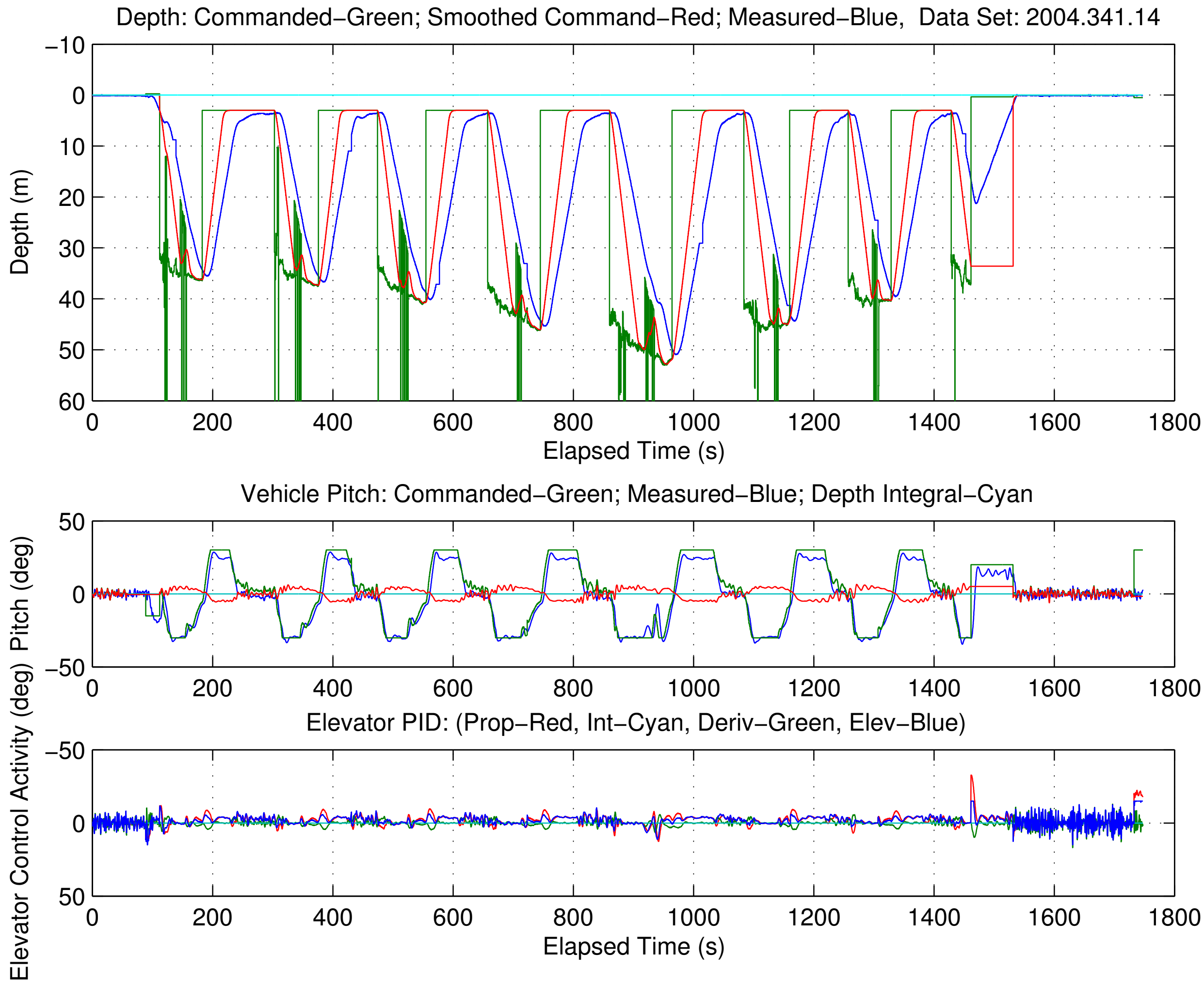

Figure 23: Vehicle Depth, Pitch and Elevator for a Yoyo Mission

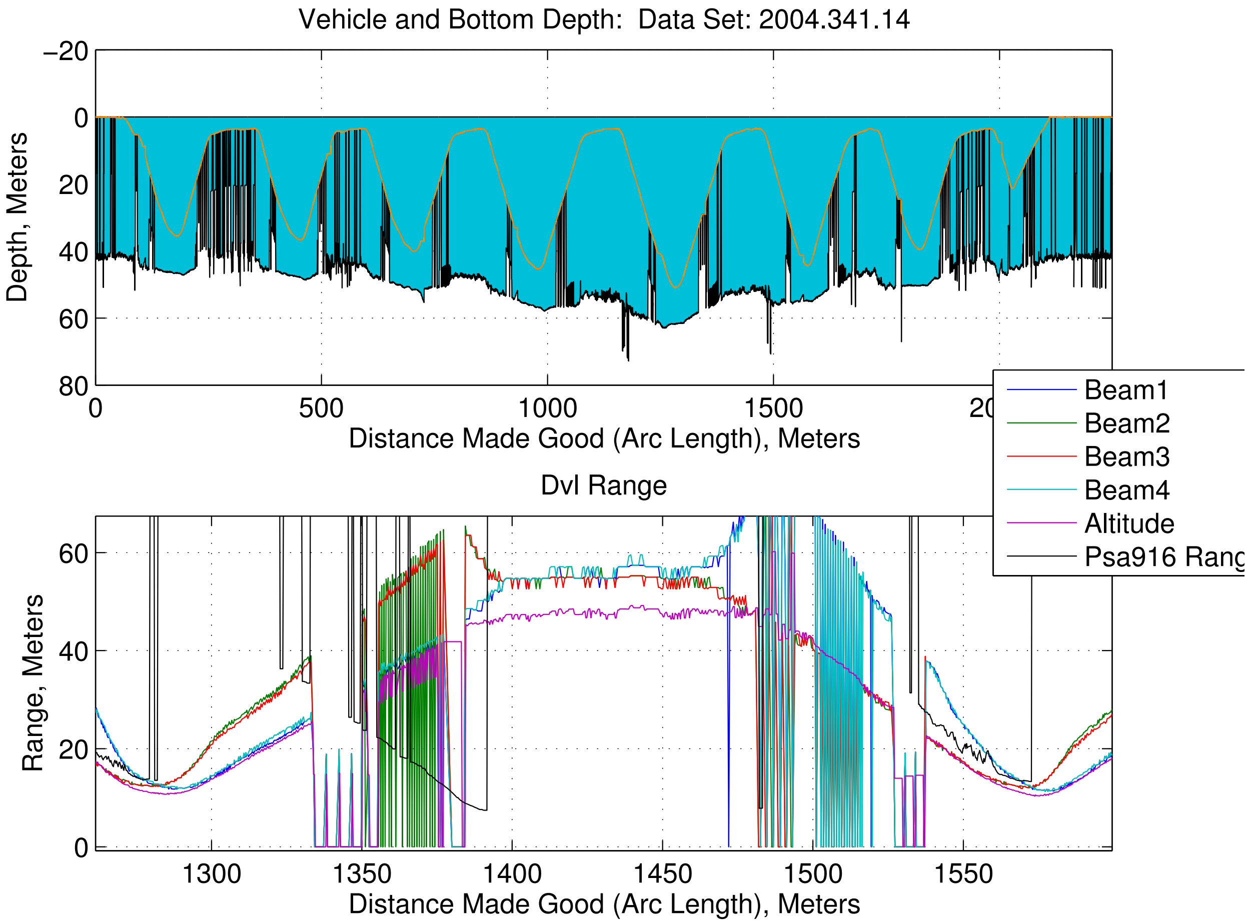

Figure 24: Yoyo Mission

which indicates the Dvl head is mounted with beams 2 and 3 forward. This is nonstandard; normally beams 1 and 3 are forward.

On the ascent, beams 2 and 3 start to drop out, and on the descent it's beams 1 and 4. This repeats on most of the ascent/descent cycles. It is probably because the increased angle of incidence decreases the return echo strength.

The Psa916 apparently is detecting a side-lobe echo from the surface on the ascent, which is between 1325 and 1375 meters. The Psa916 does correctly detect the bottom between 1525 and 1575 meters, as we had hoped it might. However, the range is only about 35 meters, which is on the low side.

Thus, this mission shows that the Psa916 can detect the bottom if the angle of incidence is at least 57°. Unfortunately the range seems quite small; about 35 meters. This will probably increase as a function of increasing angle of incidence.

6.3 Obstacle Avoidance

The obstacle avoidance algorithm has been tested in simulation, as described in Section 4.3, page 12. At-sea testing has not yet been possible due to the limited performance of the Psa916, as discussed in the preceding section.

7 Conclusions and Future Work

The specific results of this project are summarized in Section 3, page 5. Simple obstacle avoidance with a single-dimensional range finder is feasible in terms of the algorithm and the vehicle control, however the sensing problem of poor range finder sonar performance at a low angle-of-incidence must be solved. Solutions are suggested below.

The depth control improvements were very effective and resulted in vastly decreased step-response settling times. See Section 3, page 5 and the rest of the report for details.

Suggested future work:

1. Test the Tritech sonar (in stock here) at a low angle-of-incidence, and determine if it will perform better than the Psa916.

2. Investigate ways to reduce side-lobe interference at a low angle-of-incidence, including mechanical baffling, and possibly more advanced detection electronics and firmware that can accept multiple echos and select the correct one.

3. Investigate an OA system where the forward-looking sonar is mounted horizontally so that it's line-of-sight is along the vehicle centerline. This would lead to a different algorithm than the one here, because there is no need to combine the range measurement

with the Dvl. As soon as the forward-looking sonar detects something in straight-and-level flight, the vehicle can begin an evasive maneuver. Beware that this still will not solve the low angle-of-incidence problem.

4. Investigate the feasibility of a path-planning OA algorithm that utilizes the lateral information in two forward Dvl beams, since they also look to the side. This would only be effective at altitudes greater than about 20 meters. It would require combining horizontal-plane and vertical-plane guidance. One approach to sensing would be to compute the normal to the plane defined by the three points (two Dvl beams and the forward-looking beam).

5. Simulate a scanning sonar. Develop, or at least implement existing Obstacle Avoidance algorithms that do path planning, rather than just climbing. This would not account for moving obstacles.

6. Port the improved depth controller to the CTD vehicle. This will require developing a depth rate estimator that uses depth, accelerometer, and Dvl measurements.

7. Code and test an emergency OA mode where the vehicle reverses course and then either shuts off or ascends under power. This mode will be triggered if the angle γ, shown in Figure 7, page 16, exceeds a specific threshold when the Psa916 range r_P is below a specified minimum.

8. The data taken with both Dvl and Psa916 operating indicate that there may be sonic interference between the two. Determine the extent and nature of the interference, and develop a way to reduce or eliminate it.

9. Simulate the situation where small pitch oscillations cause the Psa916 to keep losing and acquiring bottom lock.

10. Incorporate bending and backlash into the tailcone actuator model. Rework the model to better match the observed transient response of the pitch control.

11. Find the cause of the 10 second pitch control mode. Perform pitch control step responses in straight flight to determine if horizontal-plane dynamics are coupling with the pitch response.

12. Further adjust the analytic model of Section 5 to account for the slightly slower transient response observed in the real system, described in Section 6

13. Expand the Qnx non-flat bottom simulation to use table lookup with topographic data.

References

[1] Gianluca Antonelli, Stefano Chiaverini, Roberto Finotello, and Riccardo Schiavon. Real-time path planning and obstacle avoidance for rais: An autonomous underwater vehicle. *IEEE Journal of Oceanic Engineering*, 26(2):216–227, April 2001.

[2] J. Borenstein and Y. Koren. Real-time obstacle avoidance for fast mobile robots. *IEEE Trans. Syst., Man, Cybern.*, 19:1179–1187, 1989.

[3] Datasonics, Inc., P.O. Box 8 - 1400 Route 28A, Cataumet, MA 02534. *PSA-916 Sonar Altimeter User's Manual*, March 1997.

[4] Lynn Fodrea. Obstacle avoidance control for the remus autonomous underwater vehicle. Master's thesis, Naval Postgraduate School, Monterey, CA, December 2002.

[5] M. Gertler and G. Hagen. Standard equations of motion for submarine simulation. Report 2510, David W. Taylor Naval Ship Research and Development Center, 1967.

[6] R. S. McEwen and K. Streitlien. Modeling and control of a variable-length auv. In *12th International Symposium on Unmanned Untethered Submersible Technology*, University of New Hampshire, Durham, NH, August 2001.

[7] Yvan Petillot, Ioseba Ruiz, and David Lane. Underwater vehicle obstacle avoidance and path planning using a multi-beam forward-looking sonar. *IEEE Journal of Oceanic Engineering*, 26(2):240–251, April 2001.

[8] Samuel M. Selby, editor. *CRC Standard Mathematical Tables*. The Chemical Rubber Company, 18901 Cranwood Parkway, Cleveland OH, 44128, 18 edition, 1964. Student Edition.

A Numerical Values: Mass Properties and Stability Derivatives

```
MULTIBEAM PARAMETERS
-- Mass Properties --
1017.803571    : (mass) Mass, kg.
1766.644276    : (Izz) Inertia about (principal) z axis, kg-m^2.
34.117332      : (Ixx) Inertia about (principal) x axis, kg-m^2.
5.240475       : (len) Length, Meters.
17.193159      : (len) Length, feet.
0.533400       : (dia) Diameter, Meters.
21.000000      : (dia) Diameter, Inches.
-2.620237      : (xR) Distance from CG to CP of ring, Meters.
-2.696438      : (xP) Distance from CG to pivot point, Meters.
0.120494       : (xG) x CG offset, Meters.
0.000000       : (yG) y CG offset, Meters.
0.006800       : (zG) z CG offset, Meters.
0.120494       : (xB) x CB offset, Meters.
0.000000       : (yB) y CB offset, Meters.
0.000000       : (zB) z CB offset, Meters.
-0.076200      : (xPcp) Dist from pivot to ring CP, Meters.
-3.000000      : (xPcp) Dist from pivot to ring CP, In.
0.063500       : (aR)  Dist from c.l. to rudder actuator base, m
0.063500       : (rR2) Dist from c.l. to rudder actuator pivot, m
0.025400       : (rR1) Body x Dist from Tailc. Pivot to Rudder Act. Pvt, m
-- Vehicle Control System --
0.200000       : (Ts) Control System Sampling Interval, Sec.
3.000000       : (vel) Foward speed, knots.
52.000000      : (Tp) Thrust, Newtons.
0.570000       : (Kph) Heading position gain, unitless.
1.700000       : (Krh) Heading rate gain, seconds.
0.000000       : (Kvh) Sway rate (v) gain, rad/(meters/sec).
0.017000       : (Kih) Original Heading integral gain, 1/sec.
0.000000       : (Kwp) Sway (Waypoint) gain, rad/meter.
0.000000       : (Kiwp)Sway (Waypoint) integral gain, rad/sec/m.
0.650000       : (Kpp) Pitch position gain, unitless.
1.300000       : (Krp) Pitch rate gain, seconds.
0.015000       : (Kip) Pitch integral gain, 1/sec.
0.014000       : (Kpd) Depth position gain, rad/meter.
0.000600       : (Kid) Depth integral gain, rad/meter/sec.
0.000000       : (Krd) Depth rate gain, rad-sec/meter.
```

```
-- Tailcone --
15.000000      : (maxDeltaR) Max. Rudder Deflection, Deg.
2400.000000    : (maxActDisp) Max Act. Displacement, counts.
1200.000000    : (maxActRate) Max Actuator Displacement Rate, Counts/Sec
240.000000     : (maxActDispTs) Max Act. Displacment in one Ts, Counts/sample
1.150000       : (ampsAct) Actuator current, full on, amps.
-- Control Surface Hydrodynamic Properties --
0.129677       : (TotalArea) Area of ring and struts, M^2.
370.168619     : (YdeltaRFixed) Sway force due to fixed rudder.
369.442145     : (YdeltaR) Sway force coeff. due to rudder deflection.
   - Fins or Struts -
0.010161       : (StrutArea) Control surface Area, Meters^2.
15.750000      : (StrutArea) Control surface Area, Inches^2.
0.077803       : (LinCtrlArea) Equivalent control surface area, Meters^2.
120.595454     : (LinCtrlArea) Equivalent control surface area, Inches^2.
6.280000       : (dCl) Coeff. of lift slope.
0.810000       : (Cdc) Crossflow drag coefficient
0.012000       : (Cd0) Constant component of drag coeff.
3.111111       : (Ar) Ring/strut aspect ratio.
0.900000       : (Ef) Oswald efficiency factor.
   - Ring -
0.048387       : (RingArea) Control surface Area, Meters^2.
75.000000      : (RingArea) Control surface Area, Inches^2.
4.800000       : (dClR) Coeff. of lift slope.
0.810000       : (CdcR) Crossflow drag coefficient
0.010000       : (Cd0R) Constant component of drag coeff.
4.260000       : (ArR) Ring/strut aspect ratio.
0.900000       : (EfR) Oswald efficiency factor.
-2.620237      : (xR) Ring lever arm, meters.
   - Antenna -
0.019355       : Antenna surface Area, Meters^2.
30.000000      : Antenna surface Area, Inches^2.
4.400000       : (dClF) Coeff. of lift slope.
0.810000       : (CdcF) Crossflow drag coefficient
0.005000       : (Cd0F) Constant component of drag coeff.
3.333333       : (ArF) Antenna aspect ratio.
1.000000       : (EfF) Oswald efficiency factor.
-1.858237      : (xF) Antenna lever arm, meters.
```

```
 -- Fuselage Hydrodynamic Properties --
// Added Mass:
 Yvdot  = -971.596409;  // Yvdot, kg.
 Zwdot  = -971.596409;  // Zwdot, kg.
 Xudot  =  -26.133175;  // Xudot, kg.
 Mqdot  =-1550.034028;  // Mqdot, kg-m^2.
 Nrdot  =-1550.034028;  // Nrdot, kg-m^2.
 Kpdot  =    0.000000;  // Kpdot, kg-m^2.
 Kvdot  =    0.000000;  // Kvdot, kg-m.
 Mwdot  =  105.740207;  // Mwdot, kg-m.
 Zqdot  =  105.740207;  // Zqdot, kg-m.
 Nvdot  = -105.740207;  // Nvdot, kg-m.
 Yrdot  = -105.740207;  // Yrdot, kg-m.
 Ypdot  =    0.000000;  // Ypdot, kg-m.
// Stability Derivatives:
 Kpabp  =   -0.287857;  // Kp|p| , kg-m^2
 Nuv    = -822.553014;  // Nuv   , kg
 Nur    = -458.011607;  // Nur   , kg-m
 Xvv    =  -65.247857;  // Xvv   , kg/m
 Xww    =  -65.247857;  // Xww   , kg/m
 Xvr    =  971.596409;  // Xvr   , kg
 Xwq    = -971.596409;  // Xwq   , kg
 Xrr    =  105.740207;  // Xrr   , kg-m
 Xqq    =  105.740207;  // Xqq   , kg-m
 Yuv    = -133.391380;  // Yuv   , kg/m
 Yur    =   75.647182;  // Yur   , kg
 Nrabr  =-2218.563545;  // Nr|r| , kg-m^2
 Mqabq  =-2218.563545;  // Mq|q| , kg-m^2
 Nvabv  =  452.404158;  // Nv|v| , kg
 Ywp    =  971.596409;  // Ywp   , kg-m
 Yrabr  =    0.000000;  // Yr|r| , ?
 Yvabv  = -855.618925;  // Yv|v| , kg/m
 Zwabw  = -855.618925;  // Zw|w| , kg/m
 Mwabw  = -452.404158;  // Mw|w| , kg
 Zqabq  =    0.000000;  // Zq|q| , ?
 Muq    = -458.011607;  // Muq   , kg-m
 Muw    =  822.553014;  // Muw   , kg
 Mpr    = 1550.034028;  // Mpr   , kg-m^2
 Npq    =-1550.034028;  // Npq   , kg-m^2
 Zuq    =  -75.647182;  // Zuq   , kg
 Zuw    = -133.391380;  // Zuw   , kg/m
 Zvp    = -971.596409;  // Zvp   , kg
```