A STABILITY AND CONTROL SYSTEM FOR A HEXAPOD UNDERWATER ROBOT

Olivia Min Yee Chiu

Mechanical Engineering
McGill University, Montréal

August 2008

A Thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfilment of the requirements for the degree of
Master of Engineering

Abstract

Aqua is an underwater hexapod robot that uses paddles to propel and orient itself. The system is highly non-linear and coupled, and thus far a controller has not been implemented on the robot. In this work, three different controllers were developed and utilized on the robot. The design of a controller for the vehicle began with the development of a stability augmentation system (SAS). In order to study the stability of the system, the model needed to be linearized and this was accomplished using numerical differentiation by finite differences. Using this method state space matrices were derived for three different steady state velocities and corresponding SAS's were designed based on the system's eigenvalues. These SAS's were implemented in a non-linear simulation and were shown to need further refinement. The refined SAS's were then designed and were successfully implemented on the physical robot in fresh and sea water. The design of an autopilot to operate with the SAS followed which included a proportional and a proportional-integral controller. The controllers were tested in simulation and in experiment with inconsistent results. Finally, the SAS was modified to compensate for possible faults that may occur during the operation of the robot. It was found that the original SAS was sufficiently robust to compensate for the case of a missing flipper. However, the case of a flipper stuck at a fixed angle required a modification to the SAS and this was accomplished by analyzing the additional drag forces created by the fault. The modified SAS was implemented on the robot in a set of experiments with successful results.

Résumé

Aqua est un robot hexapode sous-marin utilisant des palmes comme moyen de propulsion et de direction. Ce robot forme un système non linéaire et couplé, et présentement il n'y a pas un contrôleur sur le robot. Dans cette œuvre, trois contrôleurs différents ont été développé et implanté sur le robot. La conception du contrôleur a débuté par le développement d'un système d'augmentation de stabilité (SAS). Pour étudier la stabilité du système, le modèle a été linéarisé, ceci par une méthode de différences finis. Avec cette méthode, les matrices du système ont été obtenues pour trois vitesses différentes et les SAS's ont été conçu a partir des valeurs propres de système. Ces SAS's ont été utilisés dans les simulations non linéaire et ils ont dû être ajustés. Ces SAS's ont été implantés sur le robot en eau douce et salée. Puis un contrôleur de pilote automatique a été conçu pour fonctionner avec le SAS, incluant des contrôleurs proportionnels et proportionnel-intégrales. Les contrôleurs ont été utilisés en simulation et en expérience avec les résultats variables. Finalement, le SAS a été modifié pour compenser les fautes possibles qui peuvent se passer pendent l'utilisation du robot. On a trouvé que le SAS original était suffisamment robuste pour compenser pour la faute d'une palme manquante. Mais, pour le cas d'une palme coincée à un angle fixe, il a fallu modifier le SAS. Ceci a été fait à partir d'une analyse des forces additionnelles créées par la faute. Le SAS modifié a été mis en pratique sur le robot avec des résultats positifs.

Acknowledgments

I would like to express my sincere gratitude and appreciation to everyone that helped and supported me during my time at McGill. First, I would like to thank my supervisor, Prof. Meyer Nahon, for his guidance, advice and patience throughout my research. Thank you to the Natural Science and Engineering Research Council of Canada (NSERC) for granting me the funds for my research.

I would like to express my gratitude to my fellow researchers in the Mechatronics Locomotion and Mobile Robotics Labs. Many thanks to Chris Prahacs for answering all my questions about AQUA and his never ending patience as I learned how things worked on the robot and around the lab; Philippe Giguere for his invaluable advice and work on the code and help during the experiments; Junaed Sattar for helping me with anything and everything related to Linux and the robot code; Nicolas Plamondon for his help with the simulation and for his work on linearizing the system for me; John-Paul Lobos, Alec Mills, and Yogesh Girdhar for their help during the field trials; and Ioannis Rekleitis for helping me with LaTeX and his scuba diving advice.

Finally, I would like to thank my friends and family for their continual support and encouragement. Thanks go to my parents, Clement and Patsy Chiu for their encouragement in my choices of study. I am thankful for my dear friends Justin Sutherland, Jessica Topple and Jeffrey Bates for their constant friendship through thick and thin. And finally, thanks to my boyfriend Gordon Stuart for his love and support. Thank you for believing in me and always encouraging me to follow my dreams.

TABLE OF CONTENTS

LIST OF FIGURES

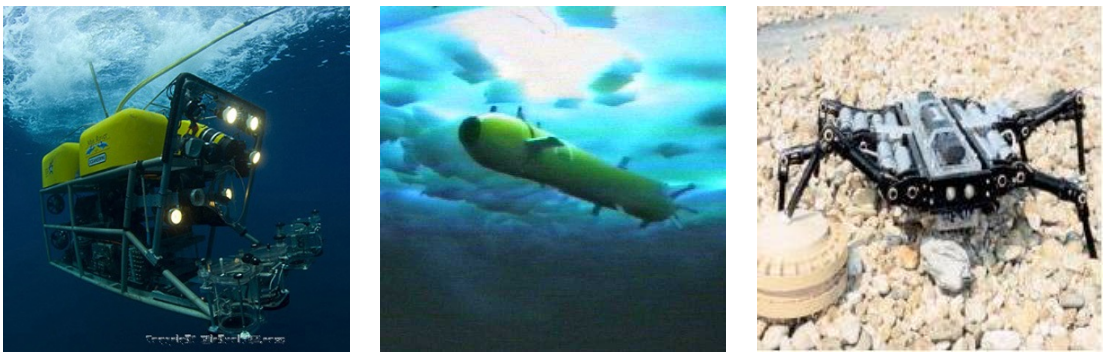
LIST OF TABLES

CHAPTER 1

Introduction

1.1 Overview

Underwater robotic vehicles are widely used for research and industrial endeavours that are often too dangerous or tedious for human divers. They are able to access deeper waters, maneuver through dangerous environments and stay underwater for longer periods of time. There are a number of different types of vehicles used for these purposes and most can be divided into one of three classes: Remotely Operated Vehicles (ROVs), Autonomous Underwater Vehicles (AUVs) and Legged underwater vehicles.



(a) (b) (c)

FIGURE 1.1. Examples of the different classes of underwater robotic vehicles: (a) A ROV, (b) An AUV, and (c) a Legged vehicle

ROVs, such as the one shown in Figure 1(a), are usually box-shaped vehicles with thrusters for locomotion and cameras to observe the surroundings. Some ROVs are more complex and have dexterous manipulators on them to collect samples or to perform tasks. They usually have a cable or umbilical running from them to a remote operator which provides power and allows for communication between a pilot and the robot. However this umbilical often limits the speed and distance the robot may go. By removing the umbilical from the vehicle, it can travel farther and faster. These tetherless vehicles are referred to as an AUV, seen in Figure 1(b) and often resemble torpedoes of various lengths. They are also equiped with thrusters for movement and various sensors. Since these vehicles are not connected to an operator by a cable, they are preprogrammed with predetermined destination and the data is recovered when the mission is over. In general, AUVs are designed to be able to move very quickly in one direction, whereas ROVs are more omnidirectional. The final category of underwater vehicle are Legged vehicles, which are robots with multiple legs that are often modeled after crabs or lobsters. They are able to walk on land and on the ocean floor but they cannot swim or navigate the waters above the ocean bottom.

1.2 The AQUA robot

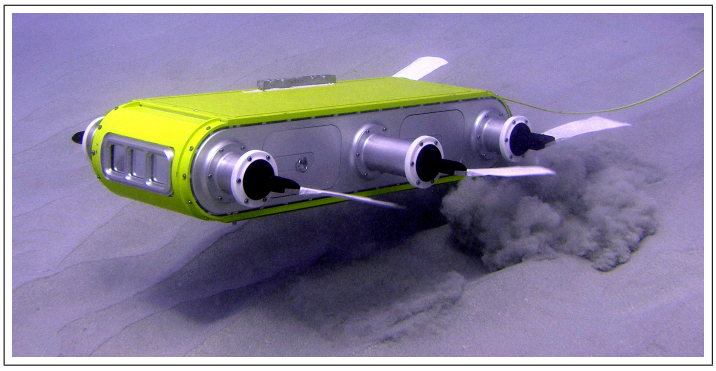


FIGURE 1.2. The AQUA robot.

AQUA is an underwater vehicle different from those described in the previous section. It is a six-legged amphibious robot, shown in Figure 1.2, that can walk on

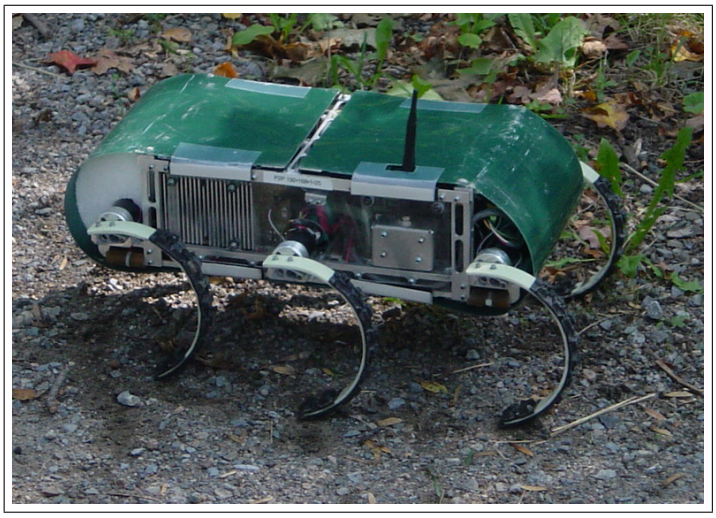


FIGURE 1.3. RHex

land with the use of semi circular legs and swim with the use of oscillating paddles. This design is based on land-based robot called Rhex, shown in Figure 1.3, which is a terrestrial six-legged robot developed in a collaboration between the Ambulatory Robotics Laboratory at McGill University, the University of Michigan, the University of California at Berkley and Carnegie Mellon University, with sponsorship from DARPA [**4, 35**]. RHex is a power autonomous robot with compliant re-circulating half-circle shaped legs and it uses a clock-driven alternating open-loop tripod gait to walk and run. These half-circular legs were replaced with flippers and the outer shell was re-designed such that it could survive in a water environment, creating AQUA [**14**]. With these flippers, AQUA is able to directly control roll, pitch, yaw, surge and heave, thus making Aqua unique compared to other underwater vehicles which use thrusters to propel themselves. The thrust created by the flippers can be regulated by changing the period or amplitude at which they oscillate and the direction of the thrust can be adjusted by changing the centre position or offset of the oscillations. At the moment AQUA has a tether, which allows for communication and data to be exchanged between the robot and a pilot.

AQUA is able to sense its movements with the use of a Microstrain 3-axis inertial measurement unit (IMU) which gives the robot's linear and angular acceleration,

angular velocity, roll angle and pitch angle. The yaw of the robot is measured with the use of an onboard magnetometer-based compass from True North Technologies which does not work particularly well due to the surrounding electronics within the robot. The robot also has three Point Grey cameras onboard (two facing the front and one facing behind the robot) which are used to record what the robot sees as it swims around.

1.2.1 *Experimental Set-up*

Experiments are usually done in either the McGill swimming pool or in the sea at the McGill Bellairs Institute in Barbados. The McGill swimming pool has 8 lanes and is 25 meters long. Most experiments take up the entire length of the pool but there is a preference to use the deeper end when robot is required to change its pitch. At the Bellairs Institute, some experiments take place off the shore in shallow water and others are done in deeper waters. Experiments done by the shore are simpler to set up but AQUA often needs to contend with the surf when swimming and the visibility is often hindered by the sand kicked up by the swimmers. In the deeper waters, the water tends to be calmer under the surface and the visibility is better. But the pilot needs to sit in a boat to control the robot and everything needs to be powered by Li-ion batteries. Therefore the number of experiments and their durations are much shorter.

Figure 1.4 shows the general set-up used when performing experiments with AQUA. The upper figure is a schematic of the set of the equipment and the lower portion of the figure shows an actual set up used during an experiment. A 200m fibre optic tether is attached to the back end of the robot to allow for communication and data transfer between the robot and a pilot on land. The other end of the fibre optic is connected to an Operator Control Unit(OCU) which converts the signals between the fibre optic cable on the robot and the serial cable from the pilot's computer. On top of the OCU is a video screen that displays the images that are being captured by any one of the three onboard cameras. Using this, the pilot is able to see where

FIGURE 1.4. The general experimental set-up with AQUA

AQUA is swimming. It also allows any swimmers or divers with AQUA to signal to the pilot to stop if a problem occurs. The OCU is also connected to the pilot's computer, which is used to monitor various sensors in AQUA, to send commands to the robot and to log whatever data that associated with an experiment.

The pilot is able to do all this through a Graphical User Interface (GUI), shown in Figure 1.5. The top of the GUI allows the pilot to switch into the swimming mode of the robot and to calibrate the flippers. Beneath this is the mode panel which is of the most interest during an experiment. The upper portion of the panel displays the orientation of the robot and the angular commands given by the pilot and the controllers. Below this and to the left, the pilot can see the oscillation period,

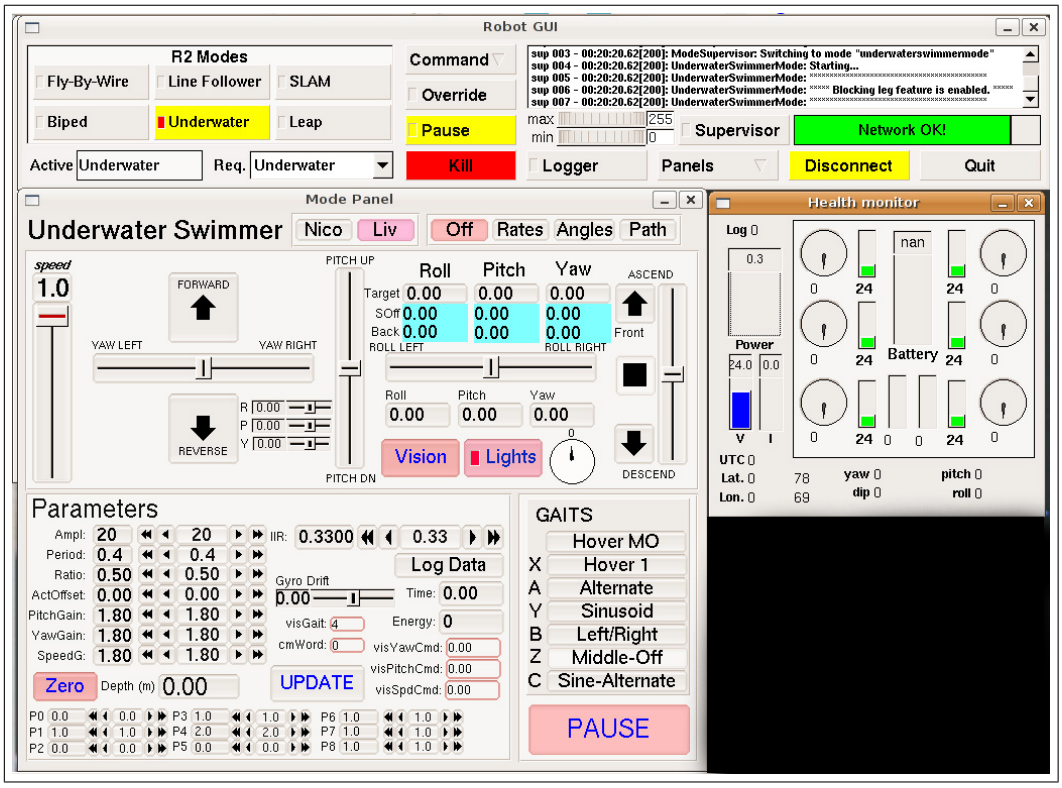


FIGURE 1.5. The Graphical User Interface seen used to send commands to the robot and to monitor the data coming from the robot.

amplitude and offset of the six paddles and is able to change these parameters. At the top of the panel, the pilot has the option to turn on a controller and chose which controller to use. At the bottom of the mode panel are nine parameters which are associated with the gains of the controllers and can be updated on the fly during an experiment. Data from the various sensors or variables in the robot code can be selected in a separate panel and logged as the robot performs an experiment. On the far right of the GUI, there is a health monitor that shows the pilot the power consumption, leg positions and battery power of the robot.

While the pilot could control AQUA by moving the sliders in the GUI, it is much easier to drive the robot with a wireless gamepad, which is shown in Figure 1.6. The gamepad sends the commands from the pilot to the computer and is then sent to the OCU and finally to the robot. The joystick on the left controls the yaw and heave of the robot, while the joystick on the right controls the roll and pitch of the vehicle.

FIGURE 1.6. The gamepad used to control the robot.

The speed of the robot is dictated by the period and amplitude of the oscillations of the flippers. While it's possible to change these parameters during the operation of the robot, they are usually left unchanged throughout an experiment. The various buttons on the gamepad allow the pilot to select different modes (such as calibrate, stand or underwater) or to start or stop the robot's movements.

During experiments, AQUA is often accompanied by two or three divers or swimmers whenever experiments are performed. The swimmers and divers keep an eye on the robot throughout the experiment to make sure it does not swim into any obstacles and to communicate with those on land about what is happening. One of the swimmers or divers often has an underwater video camera to record everything that AQUA does during the experiments.

1.2.2 *Previous Work on AQUA*

AQUA is an interdisciplinary project with many students contributing to the overall performance of the robot. Thus far, a vehicle dynamics model has been developed in MATLAB based on tests done in a stagnant tank and a swimming pool [**13, 15**].

It includes a model of the thrust generated by the six flexible flippers and allows for predictions of the vehicle motion in response to the flipper oscillations. The model derived in [**13**] was based on experimental results of oscillating rigid and flexible paddles. The lift and drag forces produced by the paddles were measured and were the basis of the nonlinear model used in this thesis. Communications between a diver and the robot are being developed where a diver can use specific tags and gestures to communicate with the robot [**9**]. Amongst the different commands that may be given to the robot, one command is to tell the robot to follow a target and this has been successfully implemented on the vehicle. This uses a vision-based control system, where the robot is able to identify specific colours from a ball or a diver, and follow the object as it moves in the water [**36, 37**]. Another control system being developed allows the vehicle to hover, which has had some successful results thus far [**8**]. If a controller is not used to maneuver the robot, a pilot is able to drive the robot with the use of a game pad. With the game pad the pilot is able to control AQUA's roll, pitch and yaw angles and the amplitude, period and the center position of the paddle oscillations.

Despite these advancements on AQUA, one problem that has been expressed by its pilot is the lack of stability control while the robot swims. Therefore, a stability augmentation system (SAS) is presented in this thesis to aid a pilot driving the robot and to aid the vision-based controller. An autopilot was also developed in this thesis to be used in conjunction with the stability augmentation system. The motivation for this is to facilitate the use of AQUA for surveillance and research of coral reefs or for hull inspections. These tasks can be too tedious or dangerous for a diver, instead AQUA could be given a pre-determined trajectory to follow and it could accomplish the task without any human interference. However, these tasks could be fairly lengthy and take place in hazardous conditions, which may lead to failures of one or more of the flippers. Previous experiments with AQUA have experienced the loss of a flipper as well as a motor failure which caused a flipper to be stuck at some angle. Thus later in this thesis the SAS was adapted to compensate for a such failures.

1.3 Literature Review

Although the use of paddles for propulsion is unique and is not widely used on underwater vehicles, there has been extensive research on controllers for use on conventional marine vehicles. This section will attempt to give an overview of previous work done related to the controllers presented in this thesis.

1.3.1 *Stability Control*

The environment in which an underwater vehicle operates is often unpredictable. Varying currents, marine life, incomplete knowledge of fluid structure interactions, etc. can result in perturbations on the robot. Thus ways to stabilize a vehicle are desirable. Although work has been done on the effects of the shape of vehicles [**16**] and the placement of the center of mass [**24**], ultimately a control system is needed to compensate for external disturbances acting on the vehicles. Such a system is called a stability augmentation system (SAS) and, as the name suggests, it is a controller that is used to reduce the influences of external forces on a system, thereby making the system more stable. It is widely used in flight control to aid pilots by reducing the effects of perturbations to an aircraft, thus making it easier for a pilot to operate the vehicle. Various examples of such a system include one by Oliva for a Boeing 747 [**30**]. He created a proportional-integral SAS that was then used to help design an altitude-hold autopilot. Fullmer et al. proposed a preliminary SAS to assist pilots with hover control on the Daedalus remotely piloted vehicle [**12**]. The SAS was used to reduce the effects of inherent instabilities in the system and to improve the response of the aircraft. Helicopters are particularly difficult aircraft to fly and the added stability provided by a SAS allows the pilot to concentrate on other aspects of flight. One such SAS was developed by Kumar et al. using LQR methods and successfully met the handling quality standards outlined by the US Army [**19**].

These are a few of many examples of SAS's in aircraft and one should note that SAS's are usually used in conjunction with a higher level controller. Such systems are not as commonly found in underwater vehicles. The stabilization and tracking

control of an underwater vehicle have been studied separately using linear and non-linear methods. Approaches used to stabilize AUV's include energy-Casimir [**23**], logic-based switching [**1**], sliding mode [**18**] and time-varying [**32**]. It is also of note that all these controllers have been designed for thruster-based vehicles. To the author's knowledge, no work has been done on creating a stabilization system for a marine vehicle that uses paddles for locomotion. However, despite these proposed stability systems, the issues of stability were ultimately incorporated into the design of the tracker rather creating a separate controller to improve stability [**7, 27, 25**]. The main reason for this is that SASs are used to reduce perturbations to zero and thus they also tend to counteract the commands from an autopilot.

The tradeoff between the stability and the maneuverability of marine animals have been investigated by a few researchers. F.E.Fish studied the physical attributes of these animals and how their attributes are used to stabilize, control and dictate the way they move in the water [**10**]. D. Weihs compared the stability versus the maneuverability of animals and how these concepts can be applied to underwater vehicles [**40**]. He discusses how the placement of fins can affect the behaviour of a fish or vehicle and mentions that a level of instability is needed to increase the maneuverability of a body.

1.3.2 *Trajectory Tracking*

Previous work on trajectory tracking controllers for underwater vehicles is much more extensive. Approaches range from simple controllers such as proportional gain to more complex ones such as adaptive control. A survey had been done by Yuh [**42**] of the design and control of AUV's, which included a listing of controllers developed up until the year 2000. These included sliding control, nonlinear control, adaptive control, neural network control and fuzzy control. All of these controllers are complex but have the advantage of being self-tuning, allowing the vehicle to compensate for changes in the vehicle's dynamics. For example, they are able to adjust to changes in the vehicle's speed or to adapt to changes in the vehicle's environment. Other

control methods not mentioned by Yuh are backstepping [**6, 2**], periodic forcing [**22**], vision-based [**34**], and learning [**5, 38**].

Other experimental comparative studies have also been done by Lea et al. [**20**] and Smallwood et al. [**39**]. Lea et al. implemented three different types of speed controllers on an AUV (PID, fuzzy logic and sliding mode) and compared the performance and complexity of each controller. They found the PID to be the simplest of the controllers but since the system is nonlinear, the PID controller took longer to respond compared to the others. The fuzzy logic and sliding mode controllers gave better results but the fuzzy logic controller required a fair bit of tuning, whereas the sliding mode controller required a great deal of system modelling. They concluded that there was no one optimal controller and the preference would depend on what advantages or disadvantages the user preferred.

Smallwood et al. compared the performance of five different model-based controllers on a ROV: a PD controller, a fixed model-based controller with and without linearization, and two types of adaptive model-based controllers. They tested the controllers with different trajectories, gains and parameters. They found that the PD controller was outperformed by all the other controllers and that the adaptive model controllers tended to provide better tracking at higher velocities.

This is by no means a complete list of all controllers available for underwater vehicles, however it must be emphasized that all the above mentioned controllers were designed for systems that utilize thrusters for locomotion. There have been very few studies into controlling foil-based vehicles. To the author's knowledge, one of the earilest studies on control using an oscillaing foil was done by Barrett et al. who designed the RoboTuna [**3**]. They successfully built and tested a robotic fish that used an oscillating tail to propel itself, much like a biological tuna. The design was complicated, using 7 parameters to dictate the movement of one foil to create motion in the forward direction.

A simpler use of oscillating foils was used by Hsu et al. on a 4 paddle robot called Gamera [**17**]. The 4 paddles were placed in an X pattern on the same horizontal plane

around the vehicle and motion was achieved by changing the amplitude, frequency and center position of the oscillation for each paddle. Simple proportional heading and depth control was used to maneuver the vehicle to a target position, but the trajectory that the robot used to reach the target was dependent on the current orientation of the robot and the distance from the target. Another four foil vehicle was designed by MIT and used heuristically tuned feedback proportional gains to control the turtle-like robot [**25**]. A two foil design was used by Narasimhan et al. who placed a set of pectoral fins on an AUV and used an optimal control system to control the yaw angle.[**28**].

1.3.3 *Fault Tolerance*

The work done on fault tolerance can fall into two main categories: fault detection or control reconfiguration. Research on fault detection usually consists of comparing the behaviour of the vehicle to a model and looking for discrepancies. A fault is detected when the discrepancies exceed a predetermined threshold [**33, 31**]. The common approach to compensating for faults is by creating controllers that are as robust as possible [**21, 11**]. However, if the original controller is not robust enough, the control system may need to be reconfigured to compensate for the detected fault. One such example was presented by Yang et al. whose system included a thruster control matrix, which relates the outputs of the controller to the force created by each individual thruster [**41**]. When a fault is detected and indentified, the system eliminates the failed thruster from the matrix and the control outputs are redistributed accordingly. Another reconfiguration scheme was presented by Ni et al. who used multiple sliding-mode controllers for different possible failures [**29**].

1.4 Thesis Organization

The work presented in this thesis involves the design and implementation of three different controllers for the AQUA robot. In Chapter 2, the design and implementation of a stability augmentation system is presented. A different method of

linearization of the system was used in order to study the stability of the system. The next chapter presents the autopilot controllers designed to work with the stability augmentation system. Two different autopilot controllers were designed and the results of these are discussed. Chapter 4 introduces possible faults that may occur in the operation of the robot and the controller designed to compensate for these faults. Finally, Chapter 5 presents conclusions and recommends topics for future research.

CHAPTER 2

Stability Augmentation System

Traditionally, controllers designed for underwater vehicles have focused on the vehicle's trajectory tracking. The performance of such controllers can be improved by the inclusion of a stability augmentation system (SAS). The objective of such a system is to reduce the impact of external disturbances and create a more stable system which a trajectory following controller can act on. In the past, SAS's have been designed for airplanes and helicopters to assist the pilot in flight. However, there has not been much work done towards the development of a SAS for underwater vehicles, particularly one that uses paddles to propel itself. In this chapter, a stability augmentation system for Aqua is presented, and its design and successful implementation will be discussed.

Since the Aqua project is an ongoing interdisciplinary project with many people working on various parts simultaneously, a description of some past and current work done by others is needed to understand the work presented in this thesis. Section 1 covers these works by describing the dynamics model of the robot and how it was linearized in order to study the vehicle's stability. Section 2 discusses the design of the SAS and the results from simulation. Finally, Section 3 presents the experimental evaluation of the SAS.

2.1 Dynamics Model

A nonlinear dynamics model is useful for evaluating time histories of the robot's motion in response to particular paddle motions. The nonlinear model of the Aqua robot was developed in [**13**], based on a component breakdown approach to evaluate the hydrodynamic forces as the vehicle moves through the water. The model also includes a comprehensive model of the paddle force generation which calculates the lift and drag forces created by the paddle based on a steady flow. This model has been validated experimentally in a stagnant tank. The nonlinear model also takes into account the Coriolis, hydrodynamic and inertial forces. The Coriolis forces give rise to the coupling between the 6 degrees of freedom in the system and both the Coriolis and hydrodynamic forces contribute to the non-linearities in the system. The center of mass and the center of buoyancy were presumed to coincide with one another and therefore the gravitational and buoyancy forces cancelled each other out. Omitting

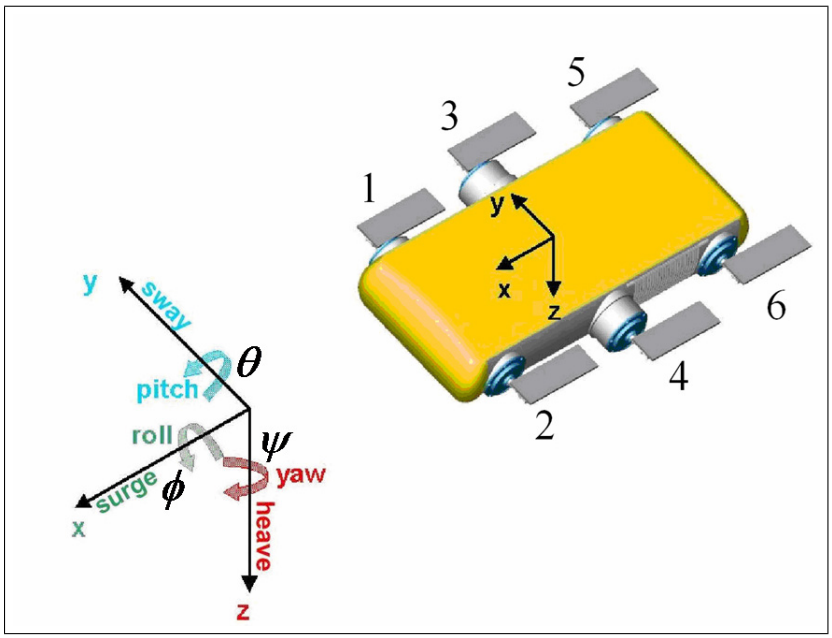


FIGURE 2.1. Six degrees of freedom of the robot and numbering of the paddles.

the details, the nonlinear model can be written at a high level as:

$$\dot{\mathbf{x}} = f(\mathbf{x}, \boldsymbol{\tau}) \tag{2.1}$$

where $\mathbf{x}$ is the state vector $\mathbf{x} = \begin{bmatrix} u & v & w & p & q & r & x & y & z & \phi & \theta & \psi \end{bmatrix}^T$ with u, v, w as the body-frame components of the robot's velocity; p, q, r are the body-frame components of the angular velocity; x, y, z are the vehicle's centre of mass position in the inertial frame, and ϕ, θ, ψ are the vehicle's Euler angles. These are shown in Figure 2.1 along with the numbering of the paddles.

The vector $\boldsymbol{\tau}$ represents the propulsive force due to the paddles whose magnitude and direction depends on the period, amplitude and offset of oscillation of each paddle. It can be decomposed into x and z-components as follows:

$$\boldsymbol{\tau} = \begin{bmatrix} f_{x1} & \dots & f_{x6} & f_{z1} & \dots & f_{z6} \end{bmatrix}^T \tag{2.2}$$

where f_{xi} represents the force provided by paddle i in the x-direction and f_{zi} in the z-direction.

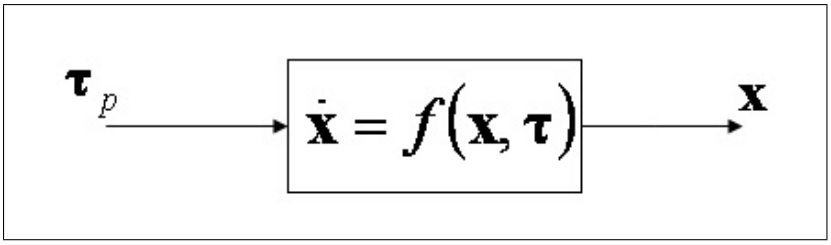


FIGURE 2.2. Open-loop block diagram of the system

The system can also be represented by a block diagram, which is shown in Figure 2.2. $\boldsymbol{\tau}_p$ is the command from a higher level controller such, as a pilot or an autopilot controller. At the moment those commands come from a human pilot on the surface and the process is open loop.

2.1.1 *Linearization*

A disadvantage of the non-linear model is that it does not provide quantitative information about the stability of the robot, and prevents the use of linear controller design theory. In order to overcome these drawbacks, the system needed to be linearized to allow the eigenvalues and eigenvectors of the system to be studied. This

linear model typically takes the following form:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\boldsymbol{\tau} \tag{2.3}$$

where $\mathbf{A}$ and $\mathbf{B}$ are 12x12 matrices and $\mathbf{A}$ is defined as:

$$\mathbf{A} = \left[\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right]_e = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_{12}} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_{12}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_{12}}{\partial x_1} & \frac{\partial f_{12}}{\partial x_2} & \cdots & \frac{\partial f_{12}}{\partial x_{12}} \end{bmatrix}_e \tag{2.4}$$

and $\mathbf{B}$ is similarly defined as $\mathbf{B} = \partial \mathbf{f}/\partial \boldsymbol{\tau}$. The subscript e denotes that the partial derivative is evaluated at a specific equilibrium condition.

With this in mind, the derivation and verification of the linear model will be discussed in the next two sections. This work was done primarily by a fellow graduate student, Mr. Nicolas Plamondon.

2.1.2 *Methodology*

There are several methods that can be used to linearize a system. A direct and common approach is through numerical differentiation by finite differences. It requires that the system first be in an equilibrium condition. A perturbation is then applied to the state vector $\mathbf{x}$, and the consequent response is evaluated. The elements of the state matrix $\mathbf{A}$ are then evaluated as:

$$A_{ij} = \frac{\Delta f_i}{\Delta x_j} \tag{2.5}$$

where f_i is the ith element of $\mathbf{f}$ and x_j is the jth element of $\mathbf{x}$.

In applying this approach to the Aqua robot, a difficulty arose because the magnitude of the paddle forces fluctuated periodically as shown in Figure 2.3, even when the vehicle was in nominal straight and level cruise. The vehicle velocity correspondingly fluctuated periodically as shown in Figure 2.4. This implied that the robot was never in a clear equilibrium, and conventional linearization techniques could not be

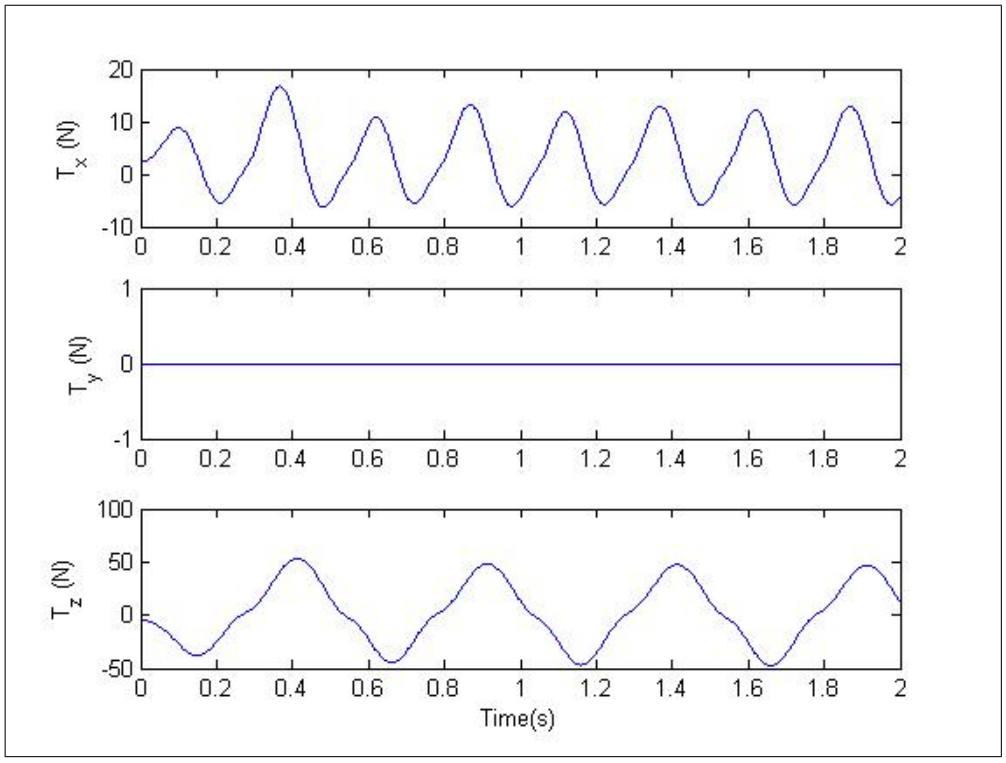


FIGURE 2.3. Net propulsive force acting at the robot mass center (x,y,z components).

used directly. A new approach was therefore designed to deal with the 'oscillating equilibrium'.

An additional issue arose when trying to design an adequate method: the response to a disturbance depended on the position of the paddle. It was therefore important to evaluate the response of the system to a disturbance over a full period of oscillation to get every possible paddle configuration. Thus, an impulse disturbance was applied at various instants over one period of oscillation. This can be seen in Figure 2.5 for the case of a disturbance of 0.02m/s in the first state variable u. For this specific case, the nominal steady-state condition was a steady forward speed of 0.16 m/s. With this approach, the response of the system for a wide variety of paddle configurations was obtained. These responses were then averaged over all the configurations and each entry of matrix $\mathbf{A}$ was calculated using the following expression:

$$A_{ij} = \frac{\overline{\Delta f_i}}{\Delta x_j} \tag{2.6}$$

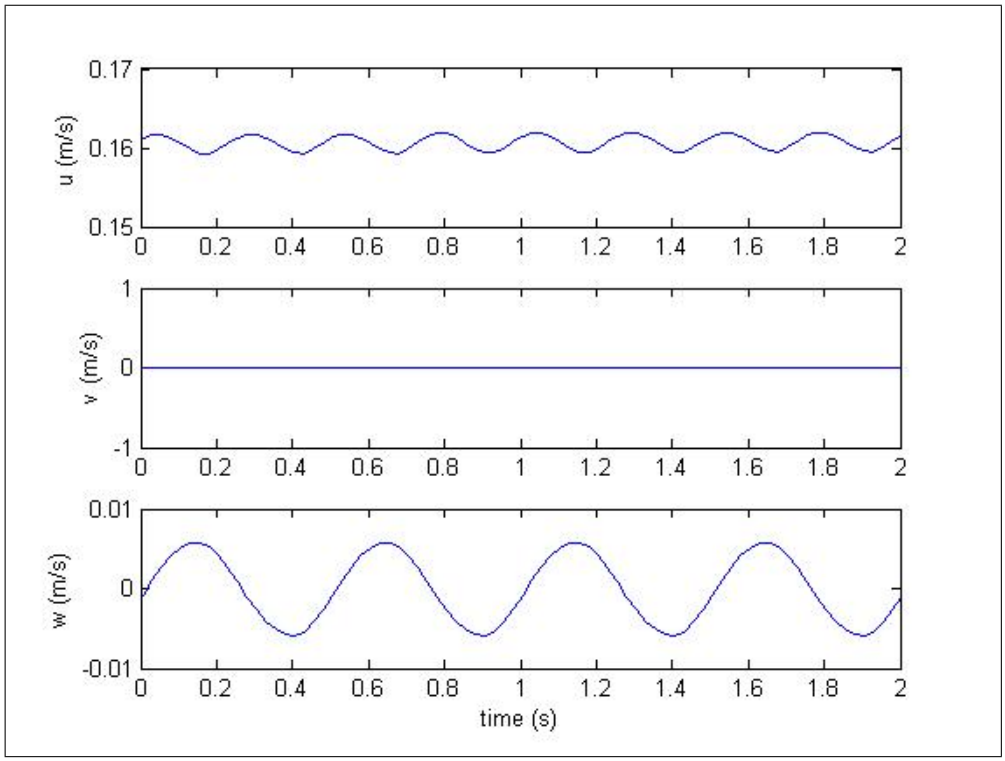


FIGURE 2.4. Translational velocity of the centre of mass (x, y, z components).

where Δx_j is the disturbance, the overbar denotes an average value of Δf_i, and Δf_i was obtained by subtracting the response with disturbance (Figure 2.5) from the response without disturbance (Figure 2.4). By taking the average, the average response of the robot over one period of oscillation was modeled and the entries of $\mathbf{A}$ represented the average dynamics of the robot.

The RMS of the entries of $\mathbf{A}$ was also calculated and compared to the average by computing the ratio of RMS over the average value. This ratio gave information about the relative magnitude of the constant and oscillating components in each entry. A small ratio implied that the constant part of the entry was dominant and that the average gave a good approximation. A large ratio implied that the oscillating part was dominant and that most of the dynamics were not accounted for by the average entry calculated using Eq. (2.6). Since the dynamics of the robot change drastically with velocity, evaluating $\mathbf{A}$ at a single operating point would not fully model the robot accurately. Thus, this procedure was applied at three different velocities (0.16m/s,

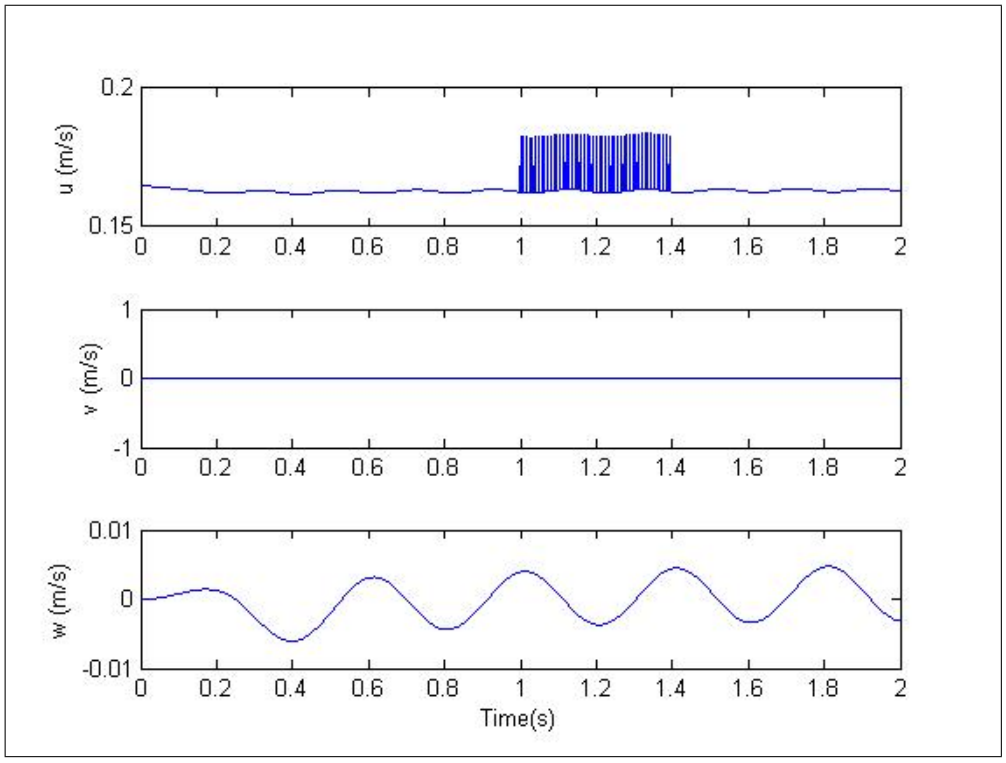


FIGURE 2.5. Translational velocity of centre of mass with disturbance in the surge velocity (x, y, z components).

0.49m/s and 0.75m/s) that spanned the operating range of the robot. The resulting **A** matrices were block diagonal and consisted of two independent subsystems; a longitudinal one and a lateral one.

The **B** matrix in Eq. (2.3) was obtained using a similar approach: the components of $\boldsymbol{\tau}$ were perturbed at various instants over one period of oscillation and Δf_i was evaluated. In contrast to the elements of **A**, it was found that the entries of **B** were not time-periodic and also remained the same for all surge velocities.

2.1.3 *Validation*

The validity of the state matrix **A** was verified by comparing the response of the linear and nonlinear models to perturbations in the initial conditions. This was done in MATLAB by giving an initial perturbation of 0.1m/s to the velocity variables from the initial condition. Results of this comparison are displayed in Figure 2.6. As can

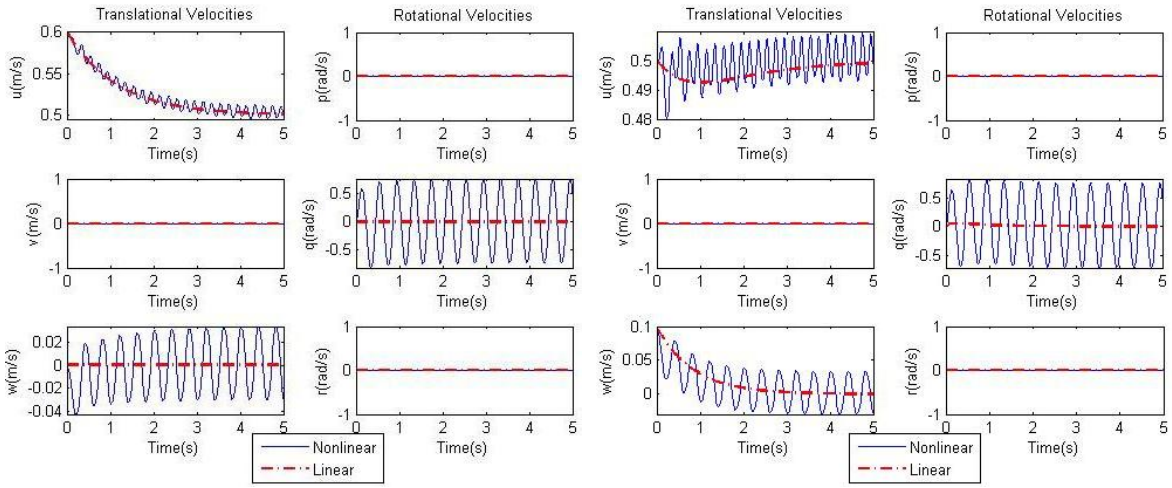


FIGURE 2.6. Comparison of linear and nonlinear simulation results for steady-state velocity of V=0.5m/s and an initial disturbance of (a) Δu=0.1m/s and (b) Δw=0.1m/s.

be seen in the figures, there is a good agreement between the linear and nonlinear model—the linear model yielded an average response consistent with the nonlinear model, but did not contain any of the fluctuations due to the paddle oscillations. As expected, the velocity returned to the steady-state condition after the disturbance, indicating a stable response in these degrees of freedom.

Matrix $\mathbf{B}$ was validated using a similar approach. The amplitude of oscillation of each paddle was increased by 0.1 radian and the period of oscillation remained unchanged. This had the effect of changing $\boldsymbol{\tau}$. Figure 2.7 illustrates that there is a good match between the linear and nonlinear simulation. The effect of increasing the amplitude of oscillation increased the thrust produced in the x-direction which had the effect of increasing the steady-state speed of the robot.

The linear model did not match the nonlinear model very well in the lateral degrees of freedom. The problem is that the linear system is unstable in the lateral degrees of freedom while the nonlinear model is not. The system described by Eq. (2.1) is coupled and highly nonlinear. The linearization discarded the nonlinearities, as

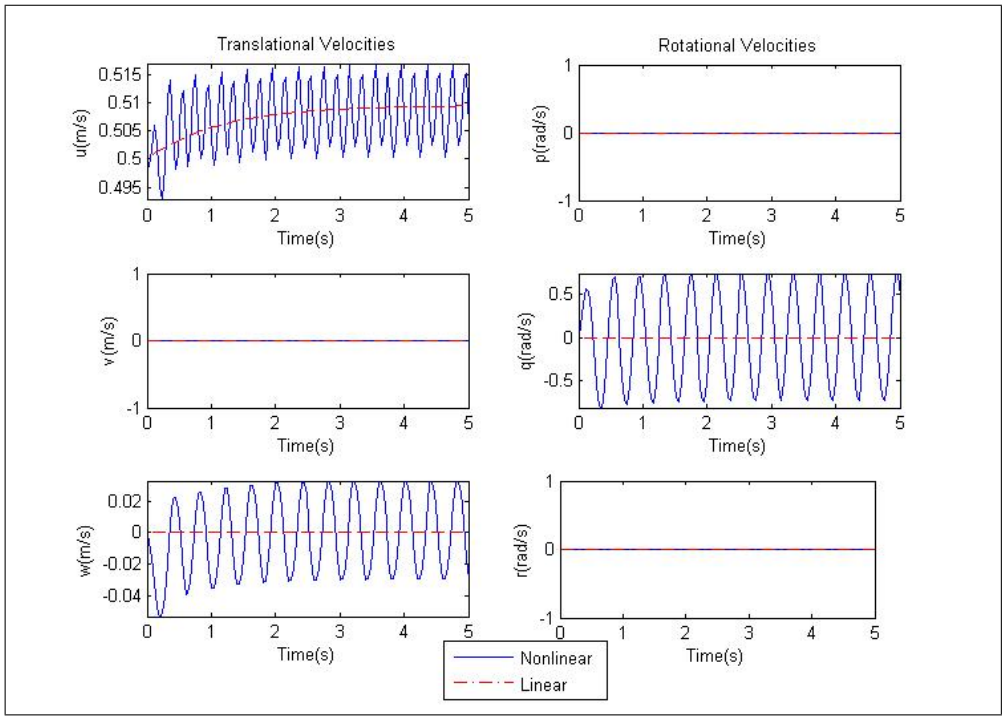


FIGURE 2.7. Comparison of linear and nonlinear simulation results for steady-state velocity of V=0.5m/s and a change in amplitude of oscillation of 0.1 radian.

well as some coupling terms which are of second order, and it appears that these may stabilize the nonlinear model.

2.2 Stability Augmentation System

Now that state space representations for three different forward velocities (0.16m/s, 0.49m/s and 0.75m/s) have been obtained, a stability augmentation system (SAS) can be designed. A stability augmented system differs from an autopilot in that it does not ensure that the robot follows a trajectory. Rather, it aims to return all state perturbations to zero, and thus reduces the impact of external disturbances on the system. This can be done by closing the feedback loop in the system and returning the measured states of the robot to the controller as seen in Figure 2.8. In this figure, the control input $\boldsymbol{\tau}_{SAS}$ is provided by the SAS, while the input $\boldsymbol{\tau}_p$ is provided by the high-level controller. The general notion is that the high-level controller 'sees' a new

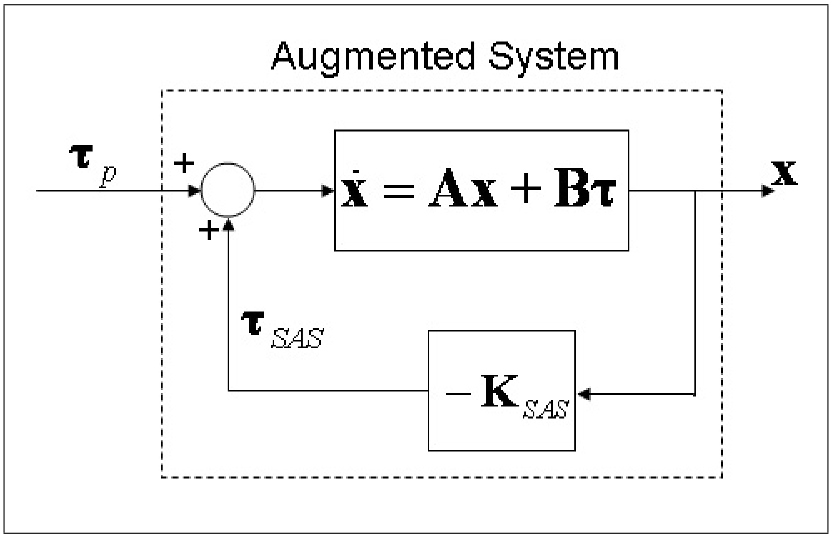


FIGURE 2.8. Block diagram of a SAS.

augmented system that is more stable than the original unaugmented system (with $\boldsymbol{\tau}_{SAS} = \mathbf{0}$). The controller can be a proportional one and take the form

$$\boldsymbol{\tau}_{SAS} = -\mathbf{K}_{SAS}\mathbf{x} \tag{2.7}$$

where $\mathbf{K}_{SAS}$ is a 12x12 gain matrix of the form:

$$\mathbf{K}_{SAS} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & -K_r & 0 & 0 & 0 & 0 & 0 & -K_\psi \\ 0 & 0 & 0 & 0 & 0 & K_r & 0 & 0 & 0 & 0 & 0 & K_\psi \\ 0 & 0 & 0 & 0 & 0 & -K_r & 0 & 0 & 0 & 0 & 0 & -K_\psi \\ 0 & 0 & 0 & 0 & 0 & K_r & 0 & 0 & 0 & 0 & 0 & K_\psi \\ 0 & 0 & 0 & 0 & 0 & -K_r & 0 & 0 & 0 & 0 & 0 & -K_\psi \\ 0 & 0 & 0 & 0 & 0 & K_r & 0 & 0 & 0 & 0 & 0 & K_\psi \\ 0 & 0 & 0 & K_p & -K_q & 0 & 0 & 0 & 0 & K_\phi & -K_\theta & 0 \\ 0 & 0 & 0 & -K_p & -K_p & 0 & 0 & 0 & 0 & -K_\phi & -K_\theta & 0 \\ 0 & 0 & 0 & K_p & 0 & 0 & 0 & 0 & 0 & K_\phi & 0 & 0 \\ 0 & 0 & 0 & -K_p & 0 & 0 & 0 & 0 & 0 & -K_\phi & 0 & 0 \\ 0 & 0 & 0 & K_p & K_p & 0 & 0 & 0 & 0 & K_\phi & K_\theta & 0 \\ 0 & 0 & 0 & -K_p & K_p & 0 & 0 & 0 & 0 & -K_\phi & K_\theta & 0 \end{bmatrix} \tag{2.8}$$

where K_p is the gain on the roll rate, K_q is the gain acting on the pitch rate, K_r is the gain acting on the yaw rate, K_ϕ is the gain acting on the roll angle, K_θ is the gain acting on the pitch angle and K_ψ is the gain acting on the yaw angle. The rows of the gain matrix $\mathbf{K}_{SAS}$ are associated with the propulsive forces given by Eq. (2.2) while the columns are associated with the states $u, v, w, p, q, r, x, y, z, \phi, \theta$, and ψ. Moreover, the first 6 elements (one for each paddle) of each column of $\mathbf{K}_{SAS}$ are related to a force in the x-direction, while the last 6 elements correspond a force in the z-direction. To ensure that the effort was evenly distributed amongst the paddles, it was desirable for the entries in $\mathbf{K}_{SAS}$ associated with a particular state and a particular direction to have equal magnitudes. Finally, negative signs were given to some elements in $\mathbf{K}_{SAS}$ because to successfully counter certain disturbances the resulting force on one side of the robot needs to be in the negative direction to the force on the other side. For example, to make the vehicle yaw, the three flippers on one side of the robot need to create a force in the opposite x-direction compared to the flippers on the other side.

If Eq. (2.7) is substituted into Eq. (2.3) then it becomes

$$\dot{\mathbf{x}} = (\mathbf{A} - \mathbf{B}\mathbf{K}_{SAS})\mathbf{x} + \mathbf{B}\boldsymbol{\tau}_p \tag{2.9}$$

and $\mathbf{A} - \mathbf{B}\mathbf{K}_{SAS}$ can be replaced by $\mathbf{A}'$ such that

$$\dot{\mathbf{x}} = \mathbf{A}'\mathbf{x} + \mathbf{B}\boldsymbol{\tau}_\mathbf{p} \tag{2.10}$$

This new or augmented system represented by $\mathbf{A}'$ is more stable (i.e. all eigenvalues with negative real parts, and more highly damped natural motions) and it is what the high-level controller then acts on. Since the augmented system is more stable, the design of the autopilot can focus on following a trajectory without being concerned about the stability of the vehicle.

2.2.1 *Eigenvalues*

The stability of a linear system can be evaluated by looking at the eigenvalues of the $\mathbf{A}$ or $\mathbf{A}'$ matrix. The eigenvalues of the augmented system can be found by

solving the following equation

$$\det |s\mathbf{I} - (\mathbf{A} - \mathbf{B}\mathbf{K}_{SAS})| = 0 \tag{2.11}$$

Since the eigenvalues are also the poles of the system, any eigenvalues with a positive real part would indicate instability of the system. By analyzing the eigenvectors that correspond to the positive eigenvalues, it is possible to see which states are likely to go unstable.

This was done with the three $\mathbf{A}$ matrices derived in the previous section (one for each of three forward speeds) and setting all the entries in $\mathbf{K}_{SAS}$ to zero. At each velocity, there was only one eigenvalue with a positive real part, six zero eigenvalues and the remaining five had negative real parts. At V=0.16m/s there was one pair of complex conjugate eigenvalues (with negative real parts), indicating the presence of an oscillatory mode at that speed. At the other speeds, all eigenvalues were purely real.

The eigenvector with the positive real eigenvalue was non-dimensionalized in the following way: the linear velocities (u, v, and w) were divided by the equilibrium forward velocity V: the angular velocities (p, q, and r) were divided by V/b, V/l, and V/b respectively where b is the width of the robot and l is the length of the robot; the linear positions (x and z) were divided by the length of the robot and y was divided by the width of the robot; and the orientation (ϕ, θ, ψ)of the robot were not modified. Next the eigenvector was normalized by dividing all the elements in the vector by the element with the largest value. The elements within this eigenvector with larger magnitudes are more significant and are the states that will likely go unstable. It was found that regardless of the velocity of the robot, ψ and r were the most unstable states, followed by ϕ, p, θ and finally q.

This was the order in which the gains were determined when designing the SAS and the process of choosing gains in the gain matrix $\mathbf{K}_{SAS}$ constitutes the design of the SAS. The gains acting on each state was found one at a time using Maple. The range of gains that stabilized each of the states was found by solving for $\mathbf{K}_{SAS}$ in Eq. (2.11)

such that the eigenvalues of $\mathbf{A} - \mathbf{B}\mathbf{K}_{SAS}$ all had negative real parts. Within this range, trial and error was then used to determine what gains would reduce each state perturbation to zero.

Since ψ was the first state to go unstable, it was the first state to be fed back. This was done by setting all the elements in $\mathbf{K}_{SAS}$ to zero except for the top 6 elements in the last column associated with ψ. The values for K_ψ that ensured stability were those that also ensured that all the poles have negative real parts. Solving Eq. (2.11), it was found that for $V = 0.16$m/s, only $K_\psi > 0$ could stabilize the unstable mode. Thus K_ψ was assigned the value 1 N/rad, which reduced the yaw angle to zero in steady state when implemented in the linear model. However, with this feedback in the linear model, the roll and pitch angles were not close to zero at steady state and the yaw rate had some oscillations in it. Thus, keeping K_ψ=1N/rad in the $\mathbf{K}_{SAS}$ matrix, the gain for the yaw rate was found using the same method by solving Eq. (2.11) in Maple. Similarly, the roll and pitch gains were found and added one at a time to the $\mathbf{K}_{SAS}$ matrix using the same method as with the yaw angle.

A similar process was used to create the gain matrices for $V = 0.49$m/s and $V = 0.75$m/s. The gains at higher speeds were determined to be somewhat higher than those at lower speeds. For example, the gain needed to stabilize the yaw angle at $V = 0.75$m/s was $K_\psi > 3$ N/rad. Table 2.1 summarizes the gains in the $\mathbf{K}_{SAS}$ matrix at the three different steady state velocities.

TABLE 2.1. Angular and angular velocity gains at three different steady state velocities

V (m/s)	K_p N·s/rad)	K_q (N·s/rad)	K_r (N·s/rad)	K_ϕ (N/rad)	K_θ (N/rad)	K_ψ (N/rad)
0.16	0	0	1	1	1	1
0.49	0	0	1	1	2	3
0.75	0	0	1	1	3	4

2.2.2 *Performance*

With the SAS implemented in the linear system, the poles of the closed-loop system were now found to all lie on the left hand side of the complex plane, thus indicating that the linear system had been stabilized. At each of the steady state velocities, the SAS also created a complex conjugate pair of eigenvalues and caused three of the zero poles to become negative real.

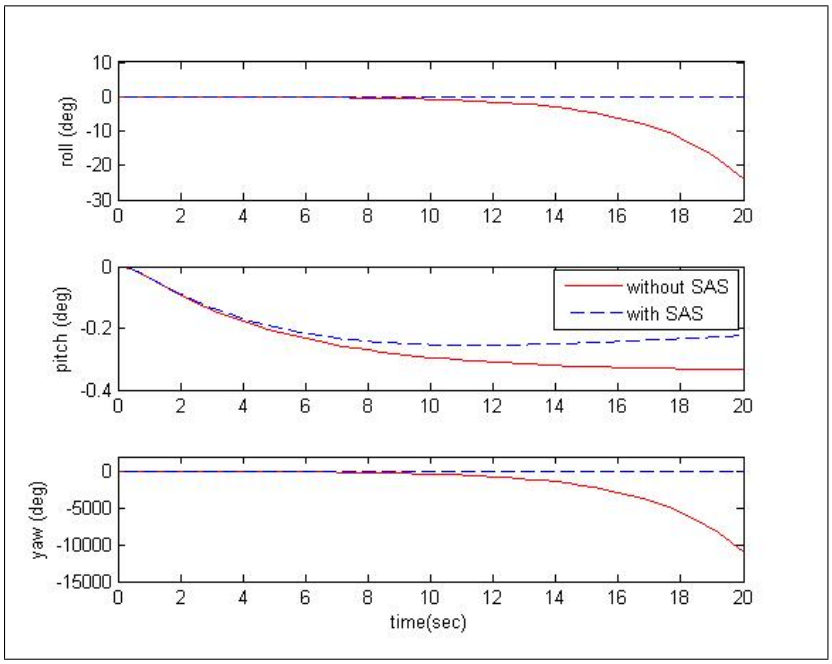


FIGURE 2.9. Linear angular response to an initial impulse at u=0.16m/s.

The gain matrices were then implemented in a closed-loop linear model of Aqua's behaviour using MATLAB. Figure 2.9 shows the angular response of the system at $V = 0.16$m/s to a unit impulse paddle force in the x-direction from paddle 1 at $t = 0$s, using the MATLAB function *impulse*. The figure clearly shows that the effects of the impulse was greatly reduced. Without the SAS the roll and yaw angles grew without bounds, however, with the SAS, these angles remained very close to zero ($|\phi_{ss}| < 0.001$deg, $|\theta_{ss}| < 0.3$deg and $|\psi_{ss}| < 0.01$deg).

To further evaluate the system performance, the gain matrices were then implemented in the non-linear model which showed that the responses were not as good as

those seen in the linear simulations. The system became a little more stable at the lowest velocity but at higher velocities the system would become or remain unstable with the SAS.

Further refinement of the gain matrices for each of the steady state velocities was done by adjusting the gains for each of the states separately. The states were stabilized in the same order as with the linear system, beginning with yaw angle. The process was done using trial and error in the non-linear simulation until a reasonable response was observed. It was found that the gains that needed to be changed the most were the ones associated with yaw, pitch and yaw rate. At $V = 0.16$m/s, most of the gains needed to be doubled but the gain for yaw needed to be 6 times higher. At $V = 0.75$m/s, some of the gains were up to 6 times higher than those used in the linear system. Furthermore, at higher velocities non-zero gains were needed to control the pitch rate in order to keep the system stable. Although these gains were relatively low compared to the other non-zero gains, they were still needed to minimize the effects of the disturbances. Table 2.2 summarizes the new gains in the $\mathbf{K}_{SAS}$ matrix at the three different steady state velocities.

TABLE 2.2. Angular and angular velocity gains at three different steady state velocities

V (m/s)	K_p N·s/rad)	K_q (N·s/rad)	K_r (N·s/rad)	K_ϕ (N/rad)	K_θ (N/rad)	K_ψ (N/rad)
0.16	0	1	2	2	1	6
0.49	0	1	6	1	4	8
0.75	0	2	6	1	19	10

Figures 2.10, 2.12 and 2.14 display the angular and forward velocity responses from the nonlinear model at $V = 0.16$m/s, 0.49m/s and 0.7m/s respectively. In these simulations approximately 0.7N·m moment disturbances (in the roll(a), pitch(b) or yaw(c) direction) were applied at $t = 10$s that lasted for 5 seconds. Figures 2.11, 2.13 and 2.15 show the forward velocity response due to the disturbances in the roll, pitch or yaw direction.

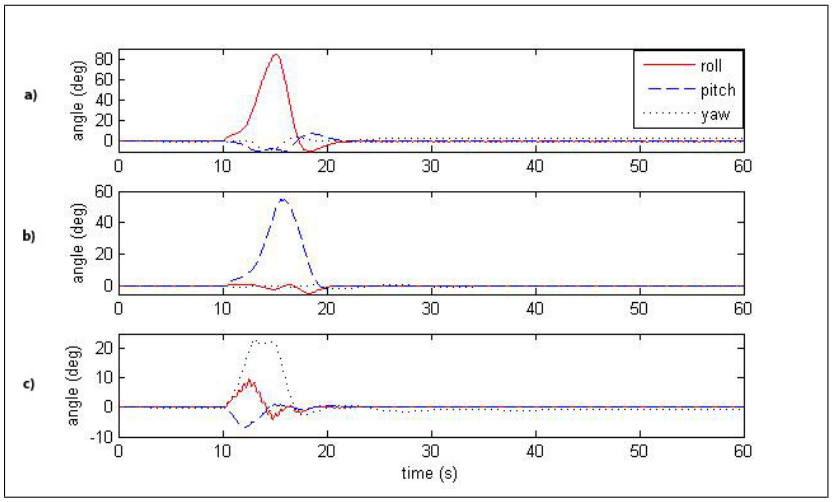


FIGURE 2.10. Angular response to angular disturbances applied in (a)roll, (b)pitch, (c)yaw directions at steady-state velocity of V=0.16m/s.

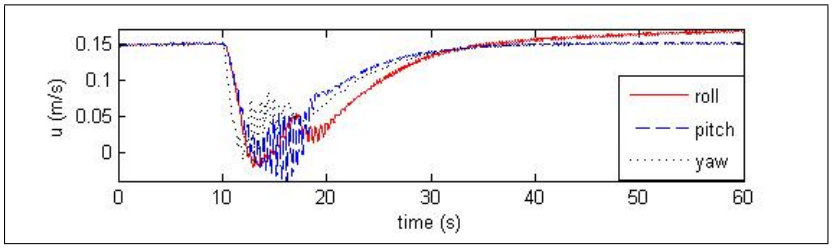


FIGURE 2.11. Forward velocity response to various angular disturbances at steady-state velocity of V=0.16m/s.

The simulations indicate that the gain matrix used for each of the steady state velocities was effective in moderating the effect of the disturbances. In each trial, the angles returned to the same values they had before the disturbance was applied. In some cases the angles were hardly perturbed by the external force. Similarly the forward velocity of the robot always returned to the same value it had prior to the disturbance. In the case of $V = 0.16$m/s, the forward velocity has a negative value for a few seconds when the angular error is at its maximum. This is simply due to the large angle that the robot has turned by the disturbance. Once the robot's angular errors returned to zero, the forward velocity returned to what is was before the disturbance.

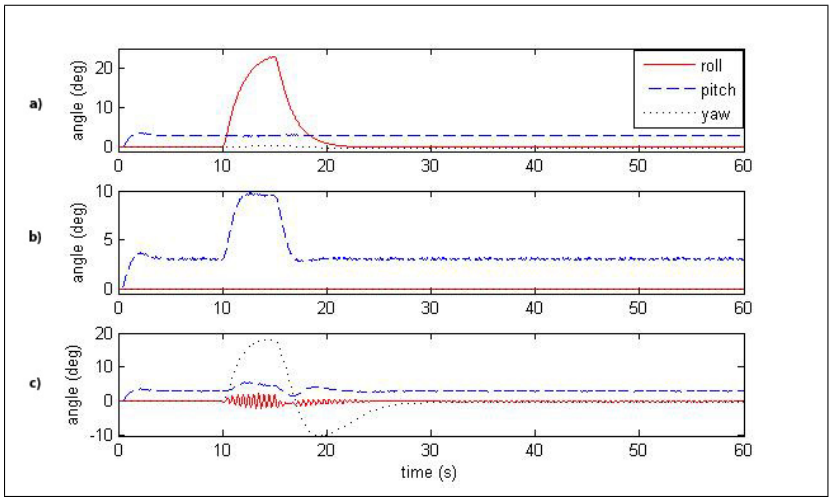


FIGURE 2.12. Angular response to angular disturbances applied in (a)roll, (b)pitch, (c)yaw directions at steady-state velocity of V=0.49m/s.

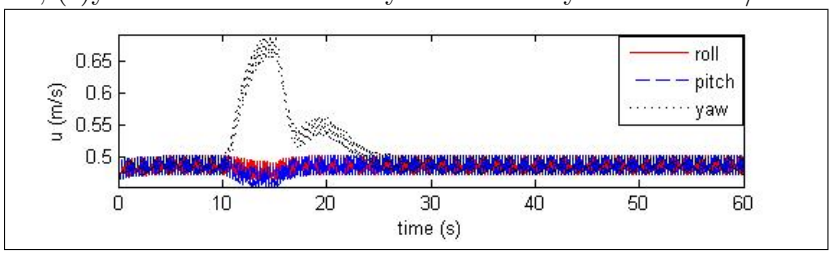


FIGURE 2.13. Forward velocity response to various angular disturbances at steady-state velocity of V=0.49m/s

The figures also show that the oscillations increased as the vehicle velocity increased. This was due to the increased amplitude of the paddle motion required to create the needed thrust in order for the robot to move forward faster. As the amplitude of the oscillating paddles increased, so did the oscillation amplitude of the vehicle velocity.

2.2.3 *Gain Scheduling*

With the SAS designs discussed in the preceding sections, it was now possible to keep the system stable for three distinct forward velocities. However, during normal operation, the robot would function at all speeds within this range. Therefore, a gain schedule was needed to determine a suitable gain matrix to use at a given velocity. A simple gain schedule was implemented that interpolated linearly between the three

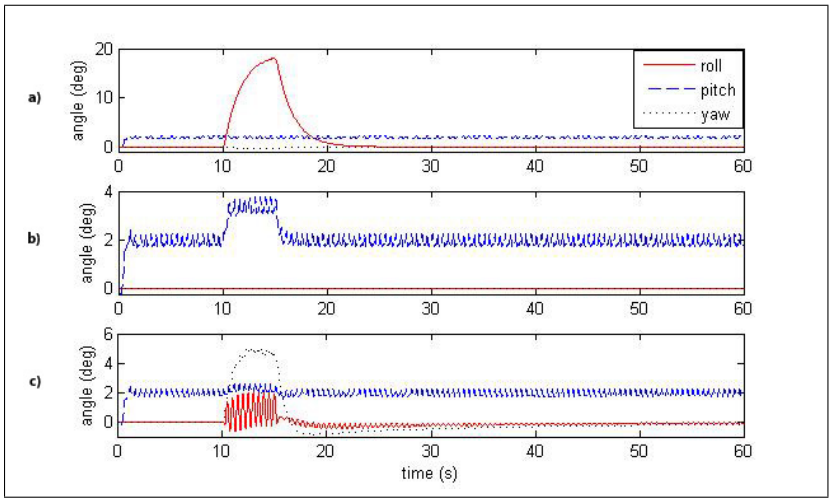


FIGURE 2.14. Angular response to angular disturbances applied in (a)roll, (b)pitch, (c)yaw directions at steady-state velocity of V=0.75m/s.

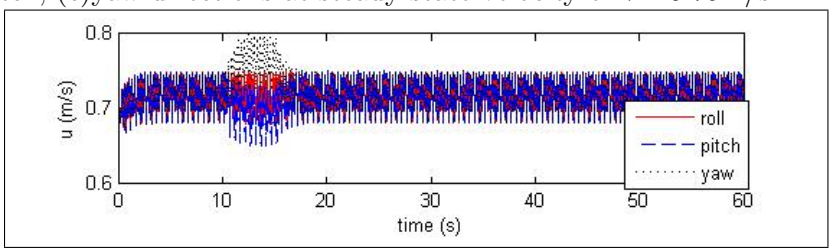


FIGURE 2.15. Forward velocity response to various angular disturbances at steady-state velocity of V=0.75m/s.

gain matrices designed above. This gain scheduled SAS was then evaluated, as shown in Figure 2.16. In this simulation, the robot was initially traveling at a forward velocity of $V = 0.16$m/s and at $t = 10$s, slowly increased its velocity. At $t = 40$s, the vehicle reached approximately 0.49m/s which it maintained for another 10 seconds. It then began to increase its velocity and reached a final velocity of 0.75m/s at $t = 80$s, which it maintained until the end of the simulation.

Disturbances were applied to the robot during the time it was transitioning between the steady state velocities. At t =15s, the disturbance in the form of a moment in the yaw direction (approximately 0.25N·m) was applied for 5 seconds. A pitch moment disturbance (approximately 0.37N·m) was then applied at $t = 65$s lasting for 5 seconds. As the figure indicates, the robot was able to recover from the disturbances very well and was able to maintain its velocity.

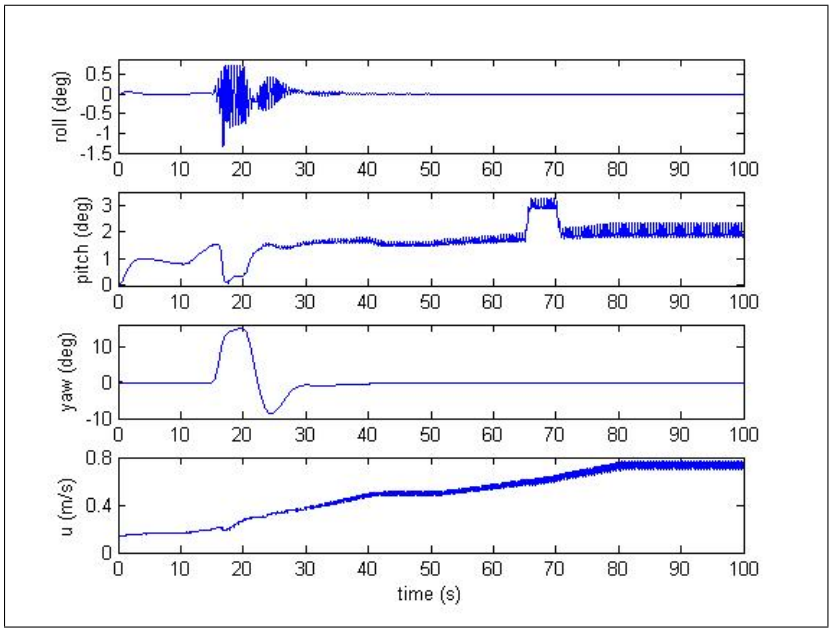


FIGURE 2.16. Angular responses and forward velocity of the robot in a simulation with the use of a gain schedule.

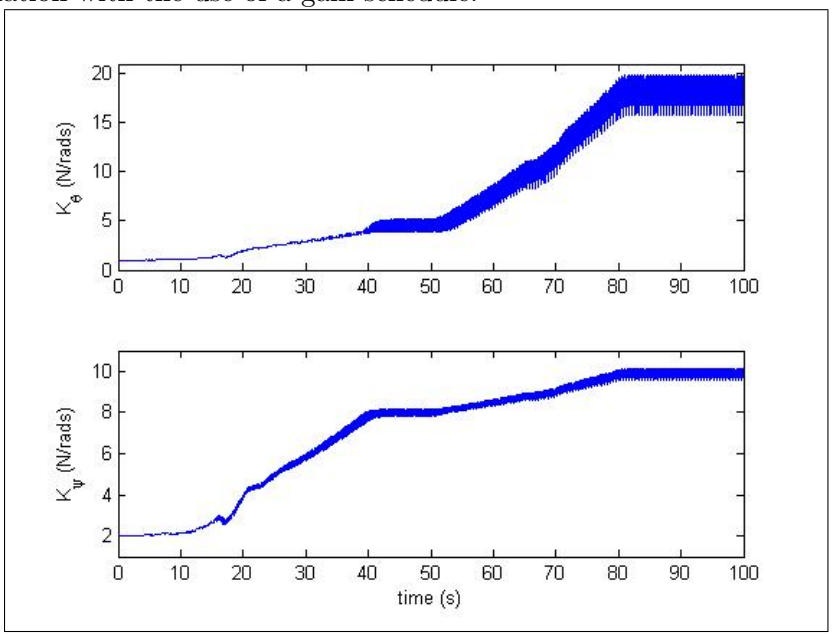


FIGURE 2.17. The pitch and yaw gains throughout the simulation using gain scheduling.

Figure 2.17 shows how the gains associated with the yaw and pitch angle changed during the simulation. Since the forward velocity oscillated with respect to time, the gains also oscillated at the same frequency. The amplitude of the oscillations

depended on the particular gain. For example, at $V = 0.49$m/s, $K_\theta = 4$N/rad and at $V = 0.75$m/s, $K_\theta = 19$N/rad but $K_\psi = 8$N/rad at $V = 0.49$m/s and $K_\psi = 10$N/rad at $V = 0.75$m/s. Therefore, K_θ fluctuated more than K_ψ after t = 50s (when the velocity remained above 0.49m/s). In general, oscillations in the gains should be avoided as they could lead to instabilities and the incorporation of a filter would have helped in this regard.

2.3 Experimental Results

The stability augmentation system was implemented on the actual robot in a swimming pool and in sea trials. In each of the trials, the data from the inertial measurement unit (IMU) was logged along with the gains used. The IMU was the only sensor feedback used and therefore only Aqua's orientation, angular velocities and accelerations were logged. Since only the orientation and angular velocities were available for feedback, the gain matrices were simplified to only 6 columns, each associated with p, q, r, ϕ, θ, and ψ. The resulting gain matrix was

$$\mathbf{K}_{SAS} = \begin{bmatrix} 0 & 0 & -K_r & 0 & 0 & -K_\psi \\ 0 & 0 & K_r & 0 & 0 & K_\psi \\ 0 & 0 & -K_r & 0 & 0 & -K_\psi \\ 0 & 0 & K_r & 0 & 0 & K_\psi \\ 0 & 0 & -K_r & 0 & 0 & -K_\psi \\ 0 & 0 & K_r & 0 & 0 & K_\psi \\ K_p & -K_q & 0 & K_\phi & -K_\theta & 0 \\ -K_p & -K_p & 0 & -K_\phi & -K_\theta & 0 \\ K_p & 0 & 0 & K_\phi & 0 & 0 \\ -K_p & 0 & 0 & -K_\phi & 0 & 0 \\ K_p & K_p & 0 & K_\phi & K_\theta & 0 \\ -K_p & K_p & 0 & -K_\phi & K_\theta & 0 \end{bmatrix} \tag{2.12}$$

and the state vector became

$$\hat{\mathbf{x}} = \begin{bmatrix} p & q & r & \phi & \theta & \psi \end{bmatrix}^T \tag{2.13}$$

Unfortunately, the sensor information of the yaw angle was not reliable due to the noise created by the other electronics within the robot and therefore the gain on the yaw angle was $K_\psi = 0$ for all the trials.

2.3.1 *Quantitative Results*

The first set of trials had Aqua swimming between two swimmers separated by approximately 6 meters. The robot initially had its paddles oscillating at period 0.5s and at amplitude 0.5rads, which corresponded to a forward velocity of approximately 0.5m/s. A third swimmer stood between the two swimmers and when the robot crossed his path, he applied a moment disturbance on the robot by pushing on various parts of the robot. For example, to apply a pitch moment, he pushed on either end of the robot. After the disturbance was applied the robot was allowed to respond without any further interference. This was done multiple times with various gains acting on the angles and angular rates of the robot. The following plots show the results of the experiments performed using the stability augmentation system.

Figure 2.18 presents the responses of the robot with gains acting on only the roll angles. The arrows in the plots indicate the times at which the swimmer applied a roll disturbance to the robot. Since multiple trials were done and the swimmer was not given any specific information about when to apply disturbances, some trials shown in the plot have multiple instances of disturbances. Furthermore, there was an attempt to line up the data such that disturbances from different trials are lined up. This was done to make is easier to analyse the performance of the SAS. The solid line in Figure 2.18 represents the response of the robot without a stability augmentation system and only one disturbance is shown in the plot. It can be seen that after 8 seconds the roll angle was not able to return the same value as before the disturbance was applied. However, the dashed and dotted lines, which represent

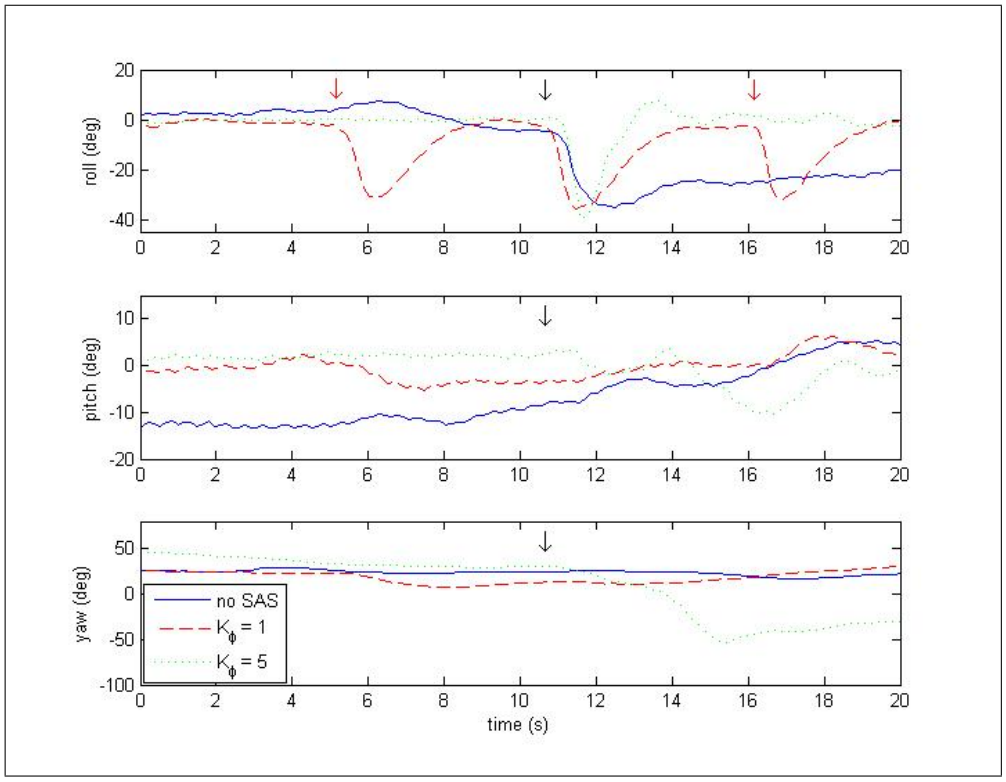


FIGURE 2.18. Experimental results with gains acting on only the roll angle.

responses with a SAS acting on the roll angle, demonstrate a noticeable improvement in the roll response of the robot. The dashed line, which had $K_\phi = 1\text{N/rad}$ shows the robot returning to the original roll angle within 3-4 seconds after each of the three instances of disturbance. The dotted line, which had $K_\phi = 5\text{N/rad}$, shows an even faster response to the one disturbance shown in the plot. The roll angle was able to return to a value close to zero within 2 seconds with a bit of overshoot. Furthermore, the behaviour of the robot before the introduction of the disturbance was also improved with the use of a stability augmentation system. As illustrated in Figure 2.18, the roll angle did not fluctuate as much with the SAS compared to the response without the SAS.

Next, a set of experiments was done using the pitch angle and rate of the robot as feedback, the results of which can be seen in Figure 2.19. Once again, the arrows in the plots indicate when a disturbance was applied by the swimmer. The solid line

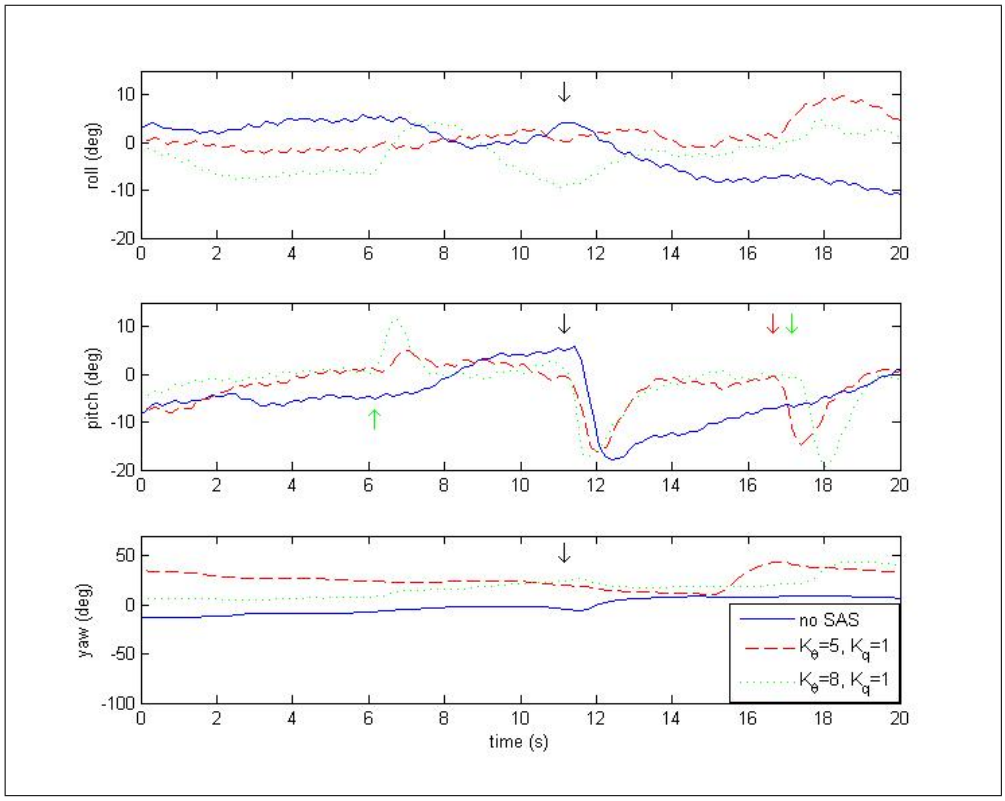


FIGURE 2.19. Experimental results with gains acting on only the pitch angle and rate.

shows the response without the use of a SAS and it can be clearly seen that after a disturbance was introduced, the robot slowly returned to a state of zero pitch but this took more than 8 seconds to accomplish. In contrast, with the use of feedback, the robot was able to return to pitch angles close to zero degrees within 3 seconds. Furthermore, the behaviour before the disturbance was again shown to be smoother with the use of the SAS.

Finally, the roll and pitch gains were used together (K_q=1N·s/rad, K_ϕ=4N/rad, K_θ=8N/rad) in a few trials and the results of one of them can be seen in Figure 2.20. The figure illustrates that the robot was able to swim forward with only small oscillations and that it was able to recover quickly from external perturbations. The perturbations were caused by the swimmer pressing down on one of the corners of the robot which resulted in a combination roll and pitch disturbance. After each perturbation, Aqua was able to recover within one or two seconds.

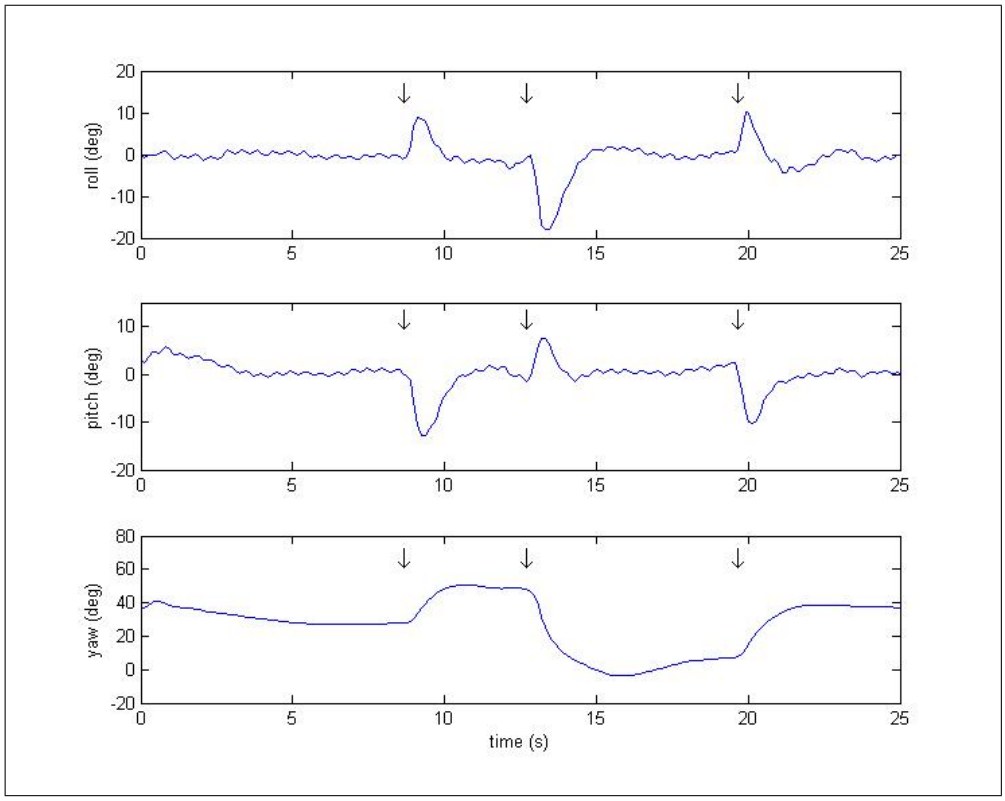


FIGURE 2.20. Experimental results with gains acting on the roll and pitch angles.

Trials were also conducted in open sea water at the McGill Bellairs Institute in Barbados to test the SAS's ability to counter the external forces caused by waves and currents. These experiments were performed just off the shore of the institute, where the surf is an additional source of disturbance. The experiments were executed in a similar fashion to those in the pool except the impulses applied to the robot were given as control input disturbances by a pilot with a game pad rather than by a swimmer. As illustrated in Figure 2.21, the use of the SAS made the behaviour of the robot much smoother. The dashed line, which represents the behaviour without the SAS, shows that the robot was not able to maintain zero roll or pitch. However, with the SAS (represented by the solid line), the robot's behaviour was much more steady and that it was able to better maintain zero roll and pitch angles at zero for longer.

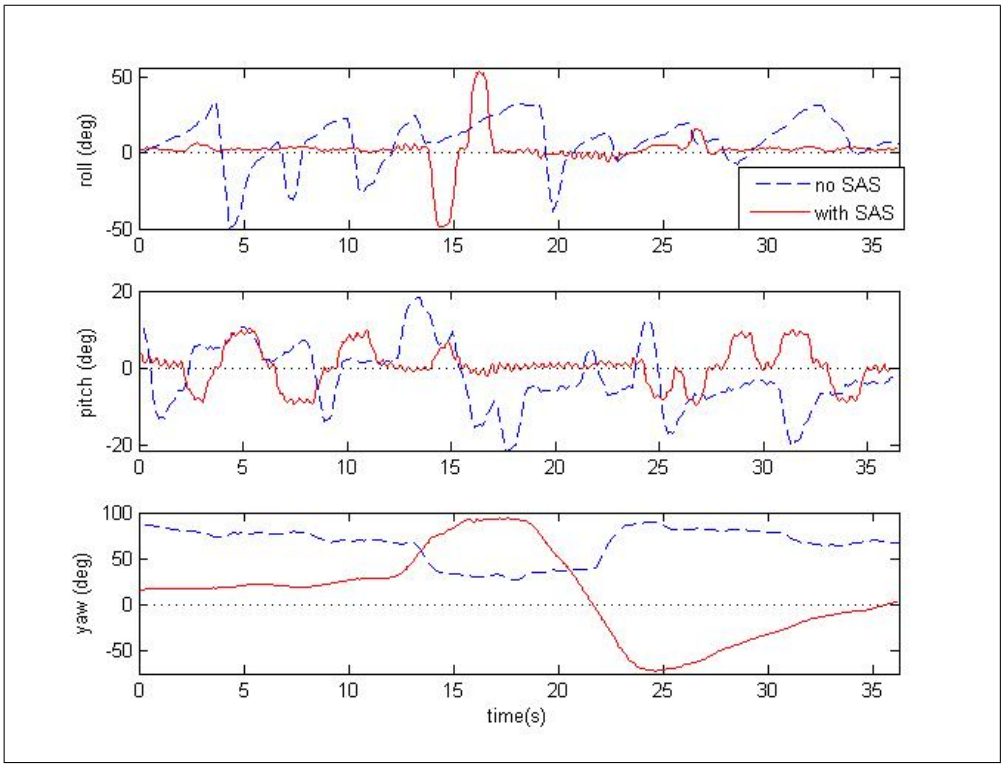


FIGURE 2.21. Sea trial results with the SAS.

2.3.2 *Qualitative Piloting Evaluation*

A qualitative piloting evaluation was also done in the sea by having an expert pilot who had used the robot in the past perform transects of coral reefs. These transects consisted of having the robot swim back and forth in a grid pattern and recording video of a coral reef below. This driver usually sat in a small boat and drove the robot from the surface of the water. Therefore, he often had to contend with a rocking boat and poor visibility while trying to keep the robot steady as it swam over the reef. When the SAS was implemented for him to use, he found that the SAS was a considerable improvement. By having the robot swimming steadily on its own, the driver was able to concentrate more on getting the video images that he wanted.

Another test performed with various non-expert pilots driving the robot with the use of a game pad. They drove the robot for a few minutes with and without the

SAS. Whether the SAS was active either in the first few or last few minutes was determined randomly and the pilots were not told when the SAS was present. Each pilot was asked afterwards whether they felt the robot was easier to control in the first few or last few minutes. Most of the pilots had little previous experience piloting the robot and they were allowed to drive the robot in any direction they desired. The results of the surveys showed that half of the group found the robot easier to control when the SAS was present and the other half of the group found it easier when the SAS was absent. The main criticism by the group that preferred the absence of the SAS was that the robot's response was more sluggish when they attempted extreme maneuvers such as back flips or sharp banked turns. In essence, they were simply "joyriding".

CHAPTER 3

Autopilot

Now that a stability augmentation system has been designed for the robot, one can proceed to design an autopilot to work in conjunction with the SAS. In this chapter, a proportional and a proportional-integral controller are designed. The simulation and experimental results from these controllers are also presented. Since experiments are usually done with the robot swimming at about 0.49m/s, all controller designs were done for this forward velocity. Section 1 describes the control system which includes the autopilot control and Section 2 describes the trajectory used to test the autopilot controller. Section 3 presents the design and the experimental results of the proportional control. Finally, section 4 presents the design of the integral-proportional controller.

3.1 Control System

In Chapter 2, a SAS was designed to provide an augmented vehicle that is more stable. An autopilot "sees" this stable vehicle and is designed such that it acts on the stable vehicle to track a desired path. Figure 3.1 illustrates how an autopilot is added to the block diagram of Figure 2.8. The measured states of the robot are fed back and compared to the desired states of the pre-determined path. The error between the measured and the desired states are then fed to the autopilot. The output from

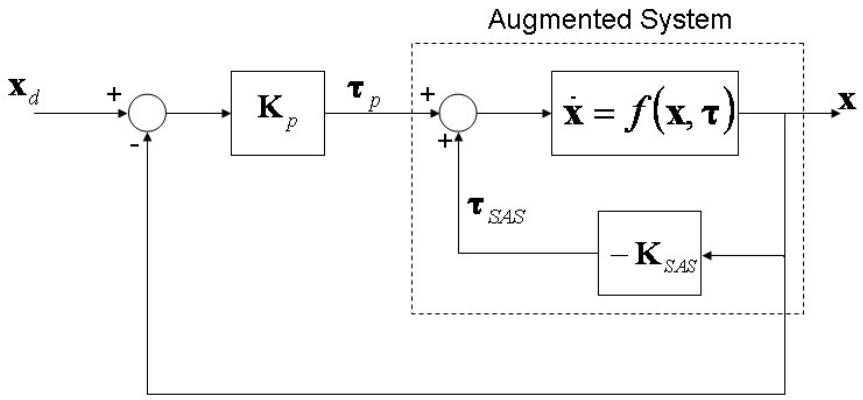


FIGURE 3.1. Block diagram of the control system with autopilot

the autopilot is added to the output of the SAS and the sum is then applied to the robot.

Before designing the autopilot, the SAS was modified slightly such that all six legs of Aqua would produce the same moment for a purely roll or yaw action. Legs 3 and 4 (the two middle legs) are at a different lateral distance from the centre of mass compared to the other four legs and they therefore generate different roll and yaw moments. In order to account for this the SAS gain matrix became

$$\mathbf{K}_{SAS} = \begin{bmatrix} 0 & 0 & -K_r & 0 & 0 & -K_\psi \\ 0 & 0 & K_r & 0 & 0 & K_\psi \\ 0 & 0 & -\frac{a}{b}K_r & 0 & 0 & -\frac{a}{b}K_\psi \\ 0 & 0 & \frac{a}{b}K_r & 0 & 0 & \frac{a}{b}K_\psi \\ 0 & 0 & -K_r & 0 & 0 & -K_\psi \\ 0 & 0 & K_r & 0 & 0 & K_\psi \\ K_p & -K_q & 0 & K_\phi & -K_\theta & 0 \\ -K_p & -K_q & 0 & -K_\phi & -K_\theta & 0 \\ \frac{a}{b}K_p & 0 & 0 & \frac{a}{b}K_\phi & 0 & 0 \\ -\frac{a}{b}K_p & 0 & 0 & -\frac{a}{b}K_\phi & 0 & 0 \\ K_p & K_q & 0 & K_\phi & K_\theta & 0 \\ -K_p & K_q & 0 & -K_\phi & K_\theta & 0 \end{bmatrix} \tag{3.1}$$

where a=0.185 is the distance between legs 1, 2, 5, 6 and the body's x-axis in m, and

b=0.24 is the distance between legs 3, 4 and the body's x-axis in m.

In order to easily add the outputs of the SAS and autopilot, the autopilot initially took the form of a proportional controller, similar to the form of the SAS:

$$\boldsymbol{\tau}_p = \mathbf{K}_p(\hat{\mathbf{x}}_d - \hat{\mathbf{x}}) \tag{3.2}$$

where $\mathbf{K}_p$ is a 12x6 gain matrix similar to Eq. (2.8),

$\hat{\mathbf{x}}_d$ is the desired states vector,

$\hat{\mathbf{x}}$ is the actual states vector, and

$\boldsymbol{\tau}_p$ is the command force vector from the autopilot controller.

3.2 Test Trajectory

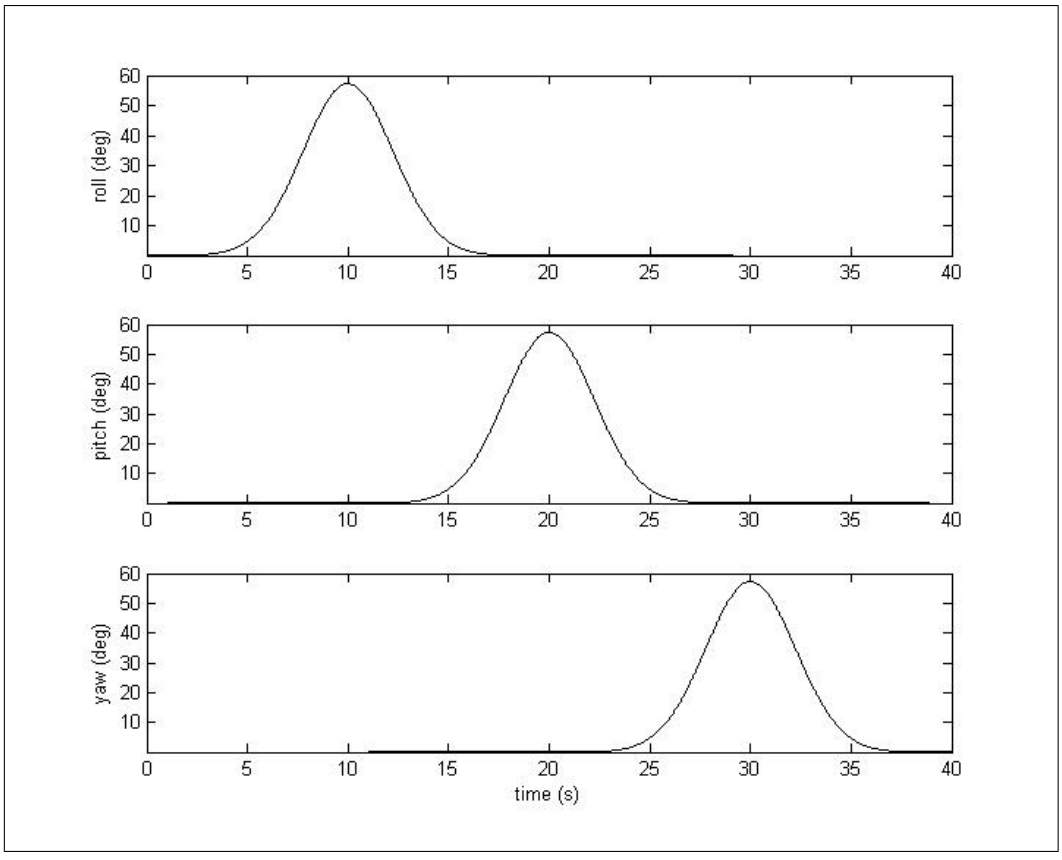


FIGURE 3.2. Desired path for the robot to follow.

To determine the performance of the autopilot controllers derived in this chapter, a test trajectory was needed. The desired path had to be smooth and the associated velocities should be smooth as well. Abrupt changes in either the orientation or the velocity would have caused spikes in the output of the controller which may have led to instabilities. Thus it was decided that the desired path was to have each angle (roll, pitch or yaw) follow a Gaussian curve. A Gaussian and its derivatives are always smooth curves, which makes it an ideal path to follow. Furthermore, a Gaussian curve allows the robot to begin with zero orientation, slowly increase one of its angles to a maximum and then slowly return to zero. Figure 3.2 shows the desired path chosen to test the autopilot controllers.

3.3 Proportional Control

3.3.1 *Derivation of Proportional Gains*

The SAS could be seen as an autopilot where the desired state is always $\mathbf{0}$, so if the robot is neutrally buoyant and the center of buoyancy coincides with the center of mass, at first glance, it would seem that the autopilot could use the same gains as the SAS. However, the SAS is designed to keep the robot swimming straight and leveled, and therefore it will tend to oppose any command given by the autopilot. Thus, the autopilot needs to accomplish two tasks: a) it must be able to counteract the commands from the SAS and b) it must be able to accomplish the task of tracking the desired path. If the gains of the autopilot are twice those of the SAS, it is possible that it would be sufficient to both counteract the effects of the SAS and follow a trajectory. Thus, the first thing attempted for the autopilot was

$$\mathbf{K}_p = 2\mathbf{K}_{SAS} \tag{3.3}$$

However, as one can see from Figure 3.3, the angles only reached approximately 60% of the desired values. The reason that this approach does not work is that as the robot nears a desired angle, the output force from the autopilot reduces but the

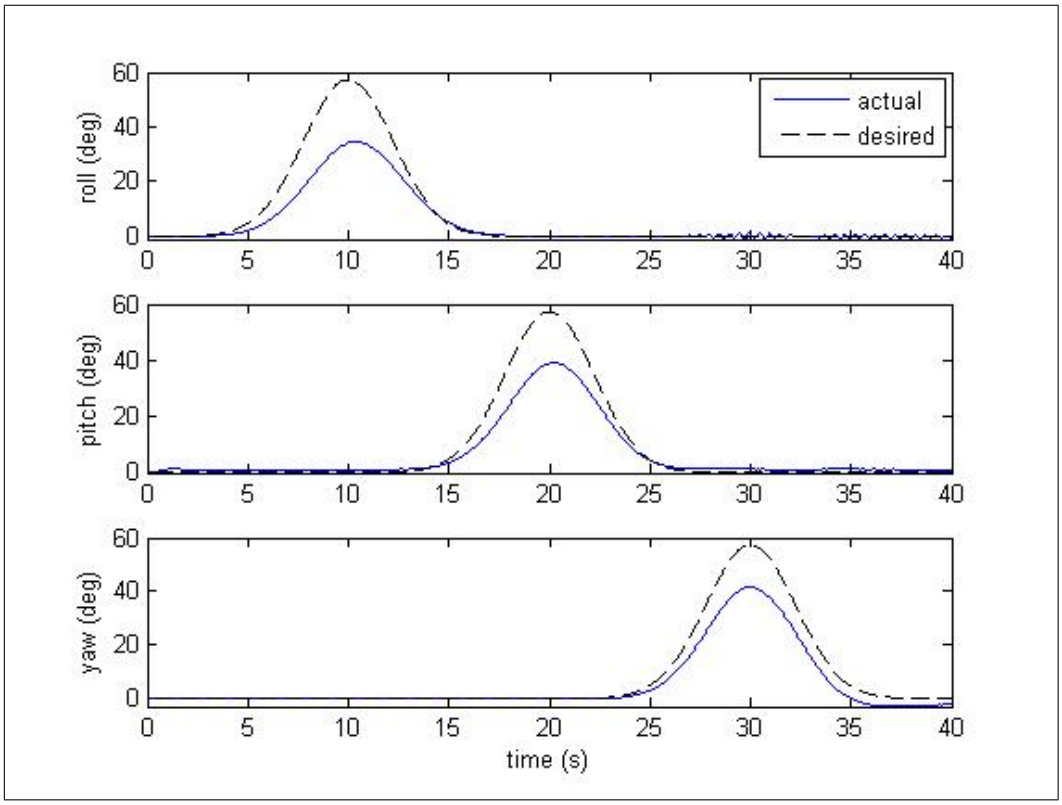


FIGURE 3.3. Response using $\mathbf{K}_p = 2\mathbf{K}_{SAS}$.

output force from the SAS increases. The plot effectively represents the equilibrium that was achieved between the two controllers.

From this point on, trial and error was used to determine the new gains for the autopilot. The goal was to have the autopilot give significantly higher commands in order to overcome the commands from the SAS. As with the SAS, the autopilot gains for the yaw angle and yaw rate were found first because these are the most unstable of the three angular states. This was done by increasing the gains on the yaw angle until the angular error for the yaw angle was minimal. The gain for yaw rate was increased if there were large oscillations in the response. Next, the gains for the roll angle and rate were determined and then finally the pitch angle and rate. The resulting gain

matrix was as follows:

$$\mathbf{K}_p = \begin{bmatrix} 0 & 0 & -10 & 0 & 0 & -100 \\ 0 & 0 & 10 & 0 & 0 & 100 \\ 0 & 0 & -8 & 0 & 0 & -80 \\ 0 & 0 & 8 & 0 & 0 & 80 \\ 0 & 0 & -10 & 0 & 0 & -100 \\ 0 & 0 & 10 & 0 & 0 & 100 \\ 5 & -5 & 0 & 35 & -80 & 0 \\ -5 & -5 & 0 & -35 & -80 & 0 \\ 4 & 0 & 0 & 27 & 0 & 0 \\ -4 & 0 & 0 & -27 & 0 & 0 \\ 5 & 5 & 0 & 35 & 80 & 0 \\ -5 & 5 & 0 & -35 & 80 & 0 \end{bmatrix} \tag{3.4}$$

Comparing this matrix to $\mathbf{K}_{SAS}$ in Eq. (2.12) and Table 2.2, notice that $\mathbf{K}_p$ has

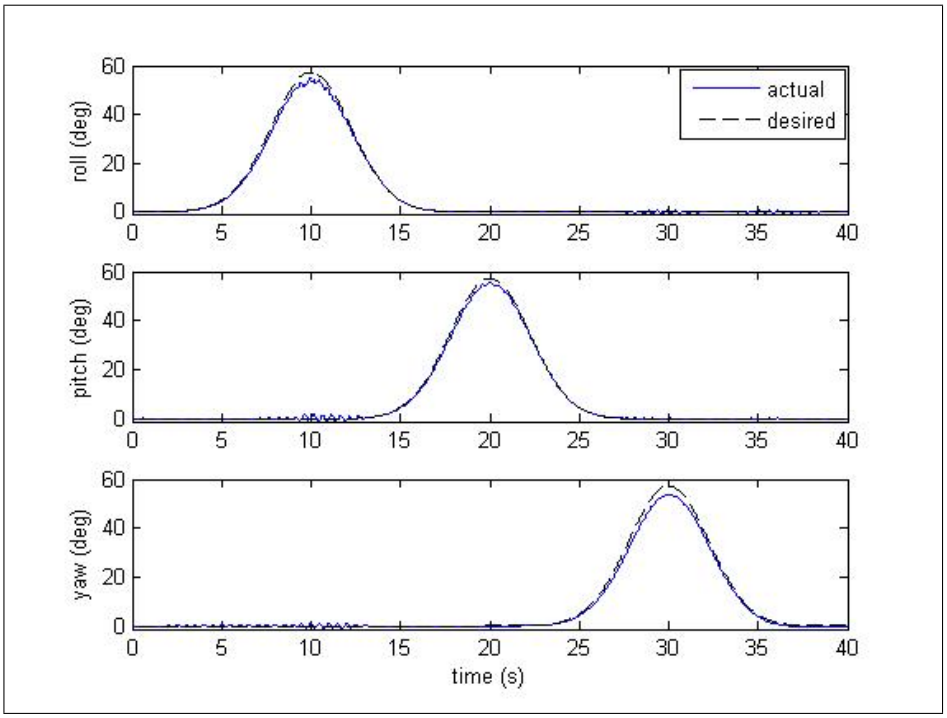


FIGURE 3.4. Response using proportional controller.

significantly higher gains. Figure 3.4 shows the results from the non-linear simulation of Aqua using the proportional autopilot and it shows that the autopilot is able to have the robot track the path fairly closely. The plot also shows small oscillations

occurring when the each desired angle approached its peak. This may be accounted by the large commands coming from the SAS and the autopilot in these time intervals. Since the autopilot gains are very high, small changes to the one of the robot's angles result in large changes to the force command from the autopilot.

3.3.2 *Experimental Results*

The proportional controller was implemented on Aqua during a set of sea trials. Given space and time limitations the desired path was reduced to the first 20 seconds of Figure 3.2 and only the desired roll angle was varied. The gains found in the previous section were much too high and lower gains were, consequently, used in the trials. Using the gains from Eq. (3.4) resulted in unstable behaviour of the robot where

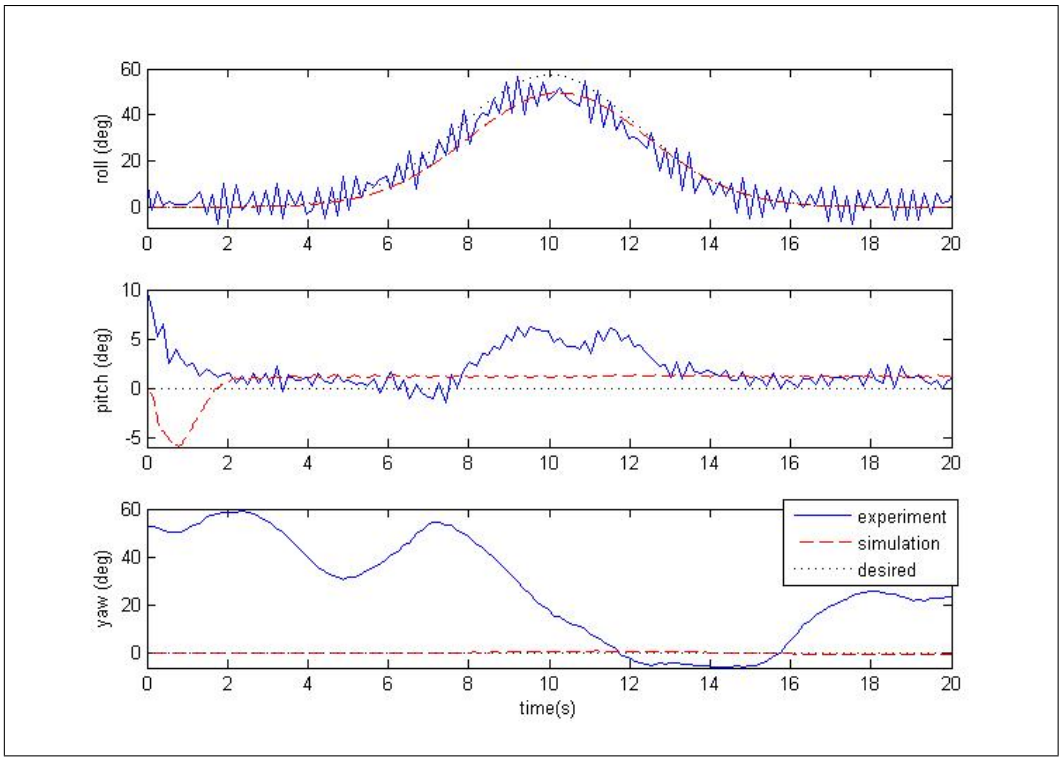


FIGURE 3.5. Sea trial and simulation results of a proportional autopilot.

the center of oscillations of each paddle were constantly at a maximum. It was later discovered that the process of translating the output from the autopilot to an angle

offset for the paddle oscillations was done differently on the robot than in simulation. Figure 3.5 compares the results from the sea trial to the results from simulation. The simulation results used the same gains as those used in the trials which were: $K_p = 0.5\text{N}\cdot\text{s/rad}$, $K_q = 2\text{N}\cdot\text{s/rad}$, $K_r = 0\text{N}\cdot\text{s/rad}$, $K_\phi = 7\text{N/rad}$, $K_\theta = 7\text{N/rad}$ and $K_\psi = 0\text{N/rad}$. While the simulation showed a very smooth behaviour in the robot, it can be seen that there were large oscillations in the actual behaviour of the robot on the order of ± 10 degrees.

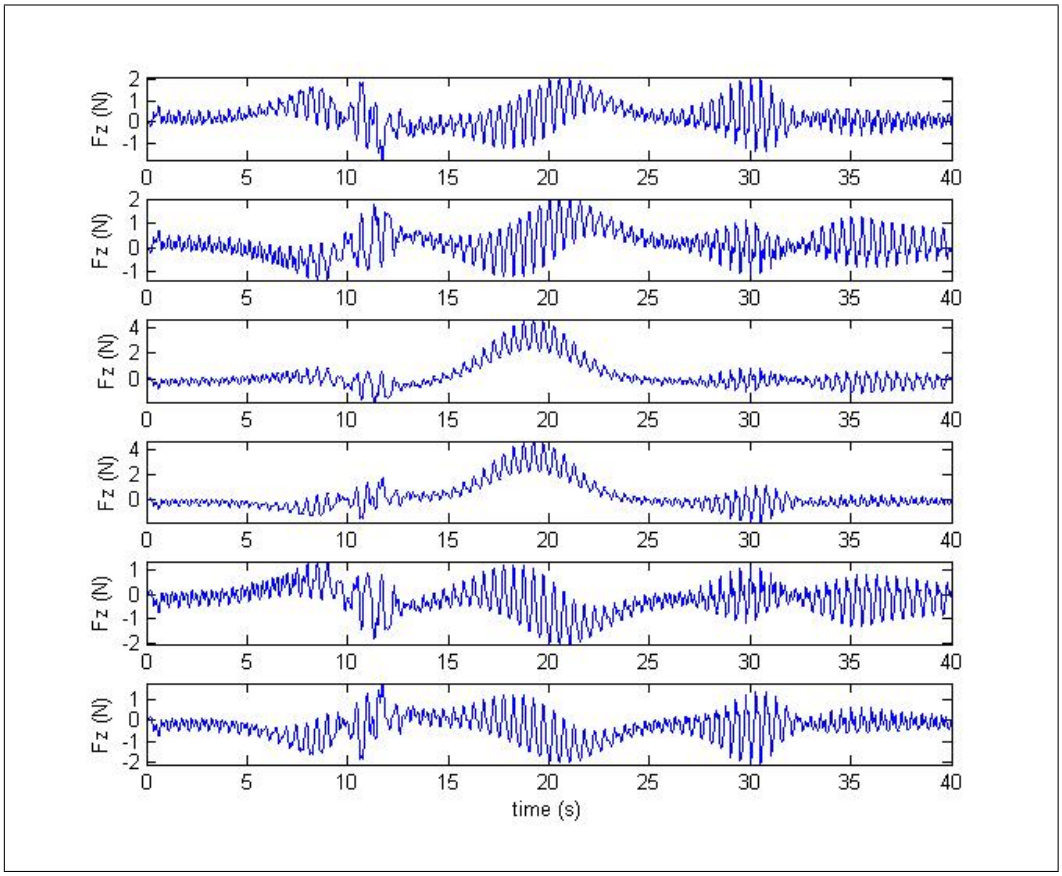


FIGURE 3.6. Total force commands in the z-direction using the P controller

These oscillations may be explained by the fact that the gains of the controller were very high and thus any changes in the actual angles of the robot resulted in large changes of the output forces from the controller. This can be seen in Figure 3.6, which shows the command forces in the z-direction for each paddle while following the path shown in Figure 3.2. These are the forces from the simulation and as one can

see, the forces oscillated up to ±2N. However, on the actual robot, the allowed force commands were limited to ±1N. Therefore, the commands were constantly saturated which led to poor results.

3.4 Proportional-Integral Control

The controller needed to be modified such that the control forces would not fluctuate so much and that the commands would not exceed the limits on the actual robot. Since the autopilot thus far has used the angles and angular velocities of the robot as feedback, it was decided to include the integral of the angles to the design of the autopilot. The function of the integral term is to ensure that, the longer Aqua strays from the desired path, the greater the control force exerted causing it to drive toward that path.

3.4.1 *Derivation of Integral Gains*

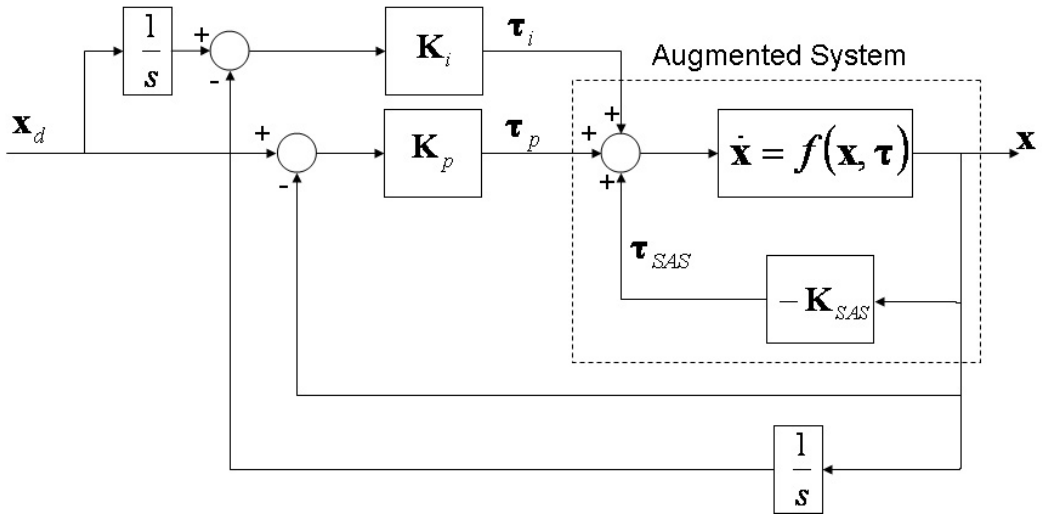


FIGURE 3.7. Block diagram of the control system with an integrator

Figure 3.7 shows the modified block diagram that includes the integral controller. The gains of the proportional controller, for the time being, was reset back to twice the SAS gains. The integral gain matrix, $\mathbf{K}_i$, was a 12x6 matrix with its elements arranged in a similar way as $\mathbf{K}_p$.

$$\mathbf{K}_i = \begin{bmatrix} 0 & 0 & -K_{ir} & 0 & 0 & -K_{i\psi} \\ 0 & 0 & K_{ir} & 0 & 0 & K_{i\psi} \\ 0 & 0 & -\frac{a}{b}K_{ir} & 0 & 0 & -\frac{a}{b}K_{i\psi} \\ 0 & 0 & \frac{a}{b}K_{ir} & 0 & 0 & \frac{a}{b}K_{i\psi} \\ 0 & 0 & -K_{ir} & 0 & 0 & -K_{i\psi} \\ 0 & 0 & K_{ir} & 0 & 0 & K_{i\psi} \\ K_{ip} & -K_{iq} & 0 & K_{i\phi} & -K_{i\theta} & 0 \\ -K_{ip} & -K_{iq} & 0 & -K_{i\phi} & -K_{i\theta} & 0 \\ \frac{a}{b}K_{ip} & 0 & 0 & \frac{a}{b}K_{i\phi} & 0 & 0 \\ -\frac{a}{b}K_{ip} & 0 & 0 & -\frac{a}{b}K_{i\phi} & 0 & 0 \\ K_{ip} & K_{iq} & 0 & K_{i\phi} & K_{i\theta} & 0 \\ -K_{ip} & K_{iq} & 0 & -K_{i\phi} & K_{i\theta} & 0 \end{bmatrix} \tag{3.5}$$

To determine the gains for the integral portion of the controller, the forces required to follow the desired path were investigated. When following the trajectory, the maximum command force occurred when the desired angular velocity is at a maximum. This is because the drag force is one of the largest forces that the robot must overcome. By taking the second derivative of the desired trajectory, the times ($t = t_{max_v}$) at which the maximum angular velocity occurred in each direction were found. For example, in figure 3.2, $t_{max_p} = 7.76s$ for the desired roll angle. The integral gains were determined by using the information at t=t_{max_v} and the following:

$$K_{ix} = \frac{F}{\int_0^{t_{max_v}} x_d dt - \int_0^{t_{max_v}} x dt} \tag{3.6}$$

where x is the angle begin tracked,

$\int_0^{t_{max_v}} x_d dt$ is the desired integral of angle x at $t = t_{max_v}$,

$\int_0^{t_{max_v}} x dt$ is the actual integral of angle x at $t = t_{max_v}$ using $\mathbf{K}_p = 2\mathbf{K}_{SAS}$,

F is the maximum allowed command force at $t = t_{max_v}$, explained below and

K_{ix} is the integral gain for angle x.

For example, with the desired path shown in Figure 3.2, the maximum roll velocity occurred at $t = 7.76$s and at this time the desired roll integral was $\int_0^{7.76} \phi_d = 0.89$ rad·s. When only proportional gain ($\mathbf{K}_p = 2\mathbf{K}_{SAS}$) was used, the integral of the actual roll angle at t= 7.76s was $\int_0^{7.76} \phi dt = 0.43$ rad·s. On the actual robot, the maximum allowed command force in the z-direction is 1N. However, if the robot reaches the desired angle at $t = 7.76$s, then the SAS would generate an opposing force that would reduce the total command force. Therefore the maximum allowed output from the autopilot is the sum of the force from the SAS and 1N ($F = \phi_d(K_\phi + 1\text{N})$. Using this information and solving Eq. (3.6) for $K_{i\phi}$, the integral gain for the roll angle was found. A similar process was used to find the integral gain for the pitch angle. Since there was not a limit on command forces in the x-direction, the integral gain for the yaw angle was done through trial and error.

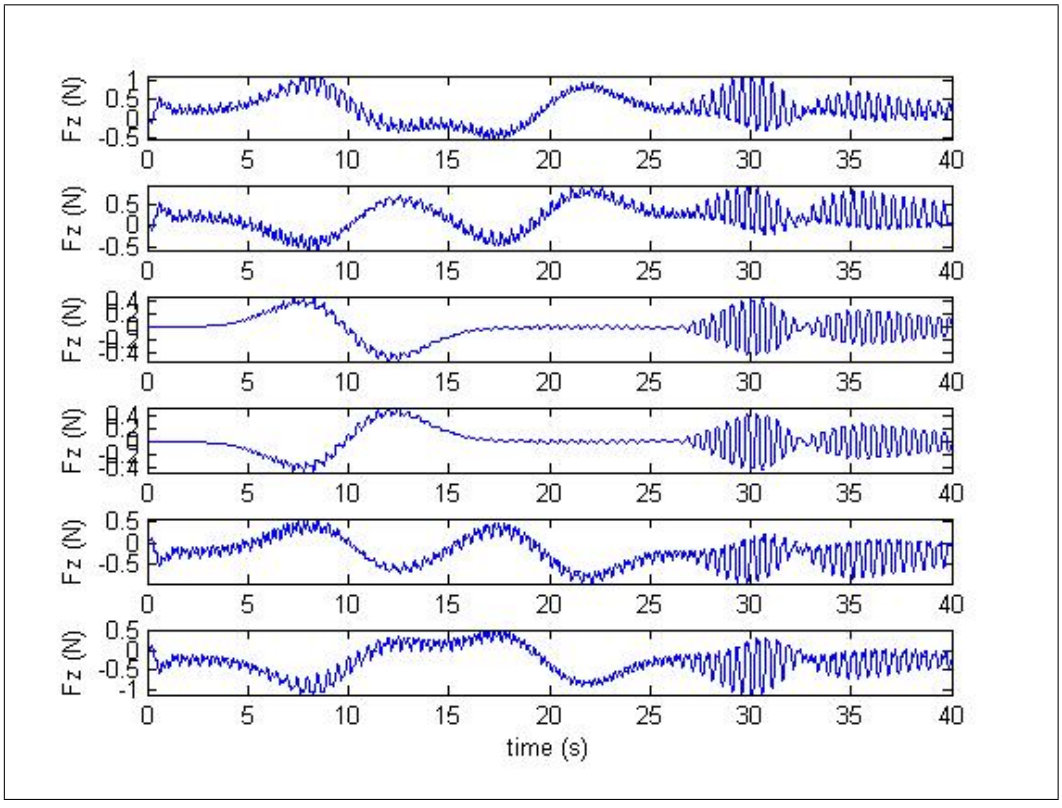


FIGURE 3.8. Total command forces in the z-direction using the PI controller.

By adding an integrator to a system, a lag in the response often occurs. Therefore, the gains in $\mathbf{K}_p$ were correspondingly adjusted. The resulting gain matrices were:

$$\mathbf{K}_p = \begin{bmatrix} 0 & 0 & -12 & 0 & 0 & -16 \\ 0 & 0 & 12 & 0 & 0 & 16 \\ 0 & 0 & -9 & 0 & 0 & -12 \\ 0 & 0 & 9 & 0 & 0 & 12 \\ 0 & 0 & -12 & 0 & 0 & -16 \\ 0 & 0 & 12 & 0 & 0 & 16 \\ 2 & -2 & 0 & 2 & -8 & 0 \\ -2 & -2 & 0 & -2 & -8 & 0 \\ 1.5 & 0 & 0 & 2 & 0 & 0 \\ -1.5 & 0 & 0 & -2 & 0 & 0 \\ 2 & 2 & 0 & 2 & 8 & 0 \\ -2 & 2 & 0 & -2 & 8 & 0 \end{bmatrix} \mathbf{K}_i = \begin{bmatrix} 0 & 0 & -24 & 0 & 0 & -20 \\ 0 & 0 & 24 & 0 & 0 & 20 \\ 0 & 0 & -18 & 0 & 0 & -15 \\ 0 & 0 & 18 & 0 & 0 & 15 \\ 0 & 0 & -24 & 0 & 0 & -20 \\ 0 & 0 & 24 & 0 & 0 & 20 \\ 8 & -17 & 0 & 3 & -12 & 0 \\ -8 & -17 & 0 & -3 & -12 & 0 \\ 6 & 0 & 0 & 2 & 0 & 0 \\ -6 & 0 & 0 & -2 & 0 & 0 \\ 8 & 17 & 0 & 3 & 12 & 0 \\ -8 & 17 & 0 & -3 & 12 & 0 \end{bmatrix} \tag{3.7}$$

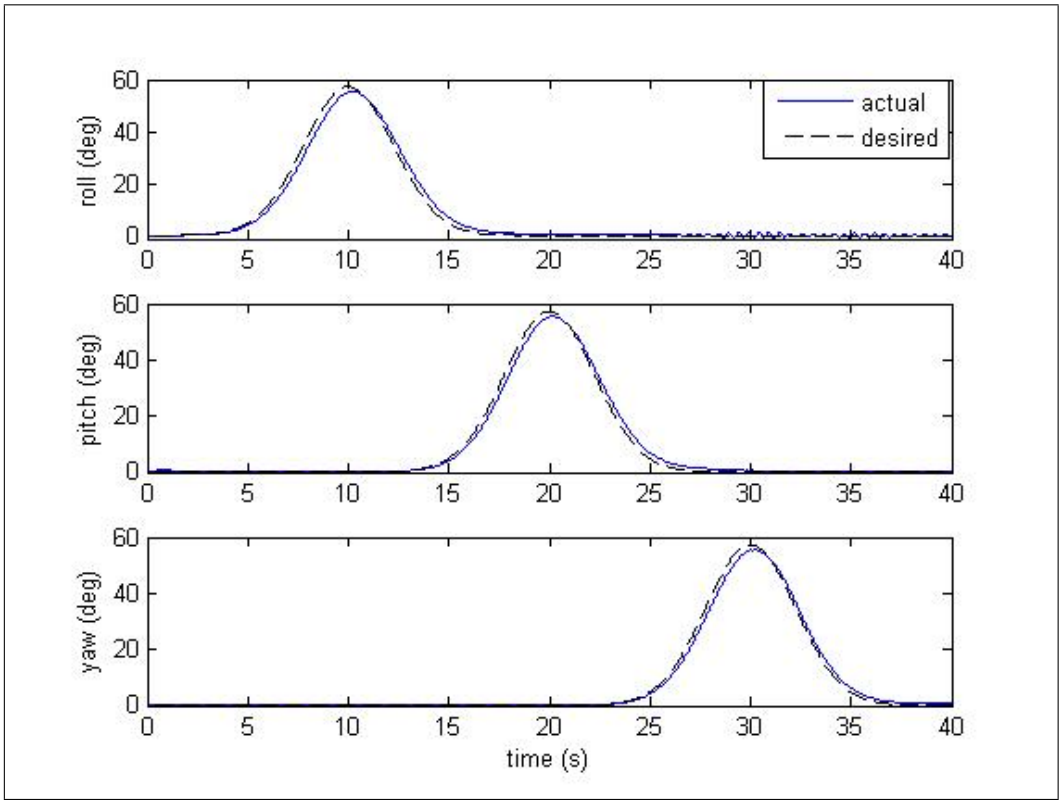


FIGURE 3.9. Response of the robot using the PI controller

As one can see, the proportional component of the autopilot had significantly lower gains than those for the proportional-only controller, which led to smaller oscillations in the force commands. This can be seen in Figure 3.8, where the net command from the autopilot and the SAS resulted in forces with magnitudes equal to or less than 1N. These were the commands that resulted from tracking the path described in Figure 3.2.

Figure 3.9 shows the results from simulation and one can see that the PI autopilot was able to track the desired path very well. The simulation with the P-only autopilot showed small oscillations in the angular response when a desired angle approached its peak value. Figure 3.9 indicate that these small oscillations are also reduced with the PI autopilot. Since the proportional gains are smaller, the small changes of the actual orientation of the robot do not cause the proportional autopilot force command to change as much.

To show that the PI autopilot was not designed to only track the path in Figure 3.2, another test trajectory was used to evaluate the autopilot. The alternate trajectory is more complex than the previous one. There are abrupt velocity transitions along the path and it requires the robot to follow two different angles at once. The path consists of a triangle wave in each of the robot's angles. The desired roll angle traces the triangle wave first and when it reaches its peak value, the desired pitch angle begins to increase and trace a triangle wave. When the pitch angle reaches its peak value, the desired yaw angle begins to change. This trajectory is represented by the dashed line in Figure 3.10 along with the simulated response. The response in simulation which is represented by the solid line shows that the robot was able to follow the desired path closely, which indicates that the PI controller is capable of tracking more complicated trajectories.

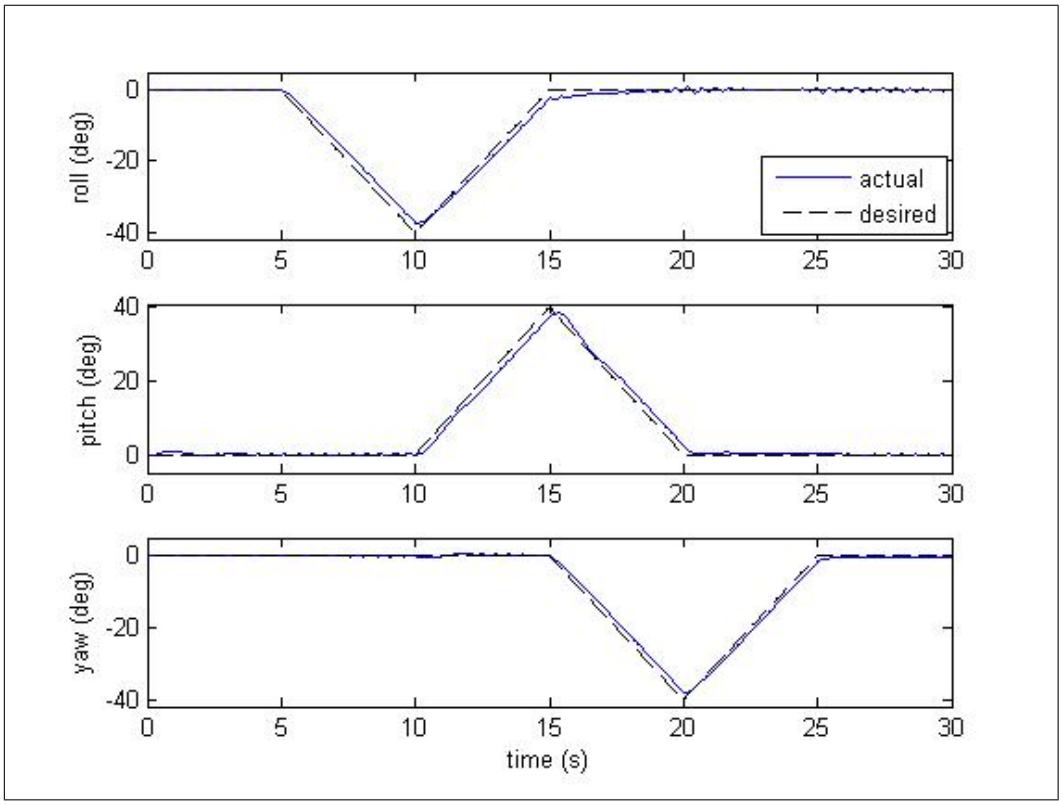


FIGURE 3.10. Response of the robot using the PI controller to follow a different trajectory.

CHAPTER 4

Fault Tolerance

Aqua is an underwater robot that is meant to be used in conditions that are not always safe for human divers. Long missions and unpredictable environments may lead to certain failures in the robot. Such failures could include the loss of mobility or outright loss of one of its flippers. These failures can be caused by internal malfunctions, such as a motor breakdown, or by external interferences, such as collisions with other objects.

Ideally, the SAS designed in Chapter 2 is robust enough to compensate for the failures. However, this is not the case with both types of failures. In this chapter the SAS is reconfigured and the approach used assumed that the existence of a failure could be accurately detected. However, the modified SAS is designed such that it is not necessary to know exactly which leg had failed. This avoids having multiple SASs which depend on correctly identifying what exactly the failure that has occurred. Since the experiments are often done at $V = 0.49\text{m/s}$, the SAS was modified for this velocity but the concepts could be applied to other velocities. Section 4.1 will briefly explain a few methods of detecting failures. Then, in Section 4.2 the case of a missing leg is investigated, while Section 4.3 looks at the situation of a leg stuck at a fixed angle.

4.1 Fault Detection

The detection of these failures could be accomplished in a number of ways. For example, a current sensor could be implemented to monitor each of the flipper motors. The current is directly related to the amount of torque provided by the motor and could indicate a failure in the part. For example, if a motor becomes stuck, the system would try to send more current to the motor to move it. Monitoring the current would be one of the simplest methods to detect a failure but unfortunately motor current sensors do not presently exist in AQUA. Another method of detection can be done by monitoring the encoders in each of the motors. These encoders give the position of each of the flippers at a given time. The error signal between the angle at which the flipper is at and the angle it should be at, can give an indication to whether a flipper is stuck. If the flipper angle of a leg is observed to maintain an error above some threshold over a certain period of time, it is a possible sign of a motor failure. However, the motors and the encoders are connected to one another and although a motor failure does not necessarily include an encoder failure, it is possible that the two will coincide depending on the cause of the failure.

A more appealing method of detection is through observing the orientation error signal of AQUA. When a failure has occurred in a leg on the robot, the robot will roll, pitch or yaw in a certain way, which will be shown later in this chapter. By monitoring the angular error of the robot, it is possible to deduce from the error whether a failure has occurred. A threshold can be set such that when the error has surpassed this threshold, it will indicate a failure. However, AQUA is an oscillating system and to prevent false reports of failure, the error signals should be filtered. The filtered signal can be taken over a fault-free interval of time that is long enough to capture the oscillations of the robot. Then the filtered angular errors could be monitored as AQUA swims and if these become larger then the fault-free filtered errors by some predetermined threshold, then a failure can be presumed.

A similar but more robust method is presented in [**26**] where Principal Component Analysis and statistics are used to monitor particular states of the vehicle's motors.

The statistical mean and deviation of the observable states is taken over the fault-free period of time. This is then used to normalize the residuals of the measured states which are calculated as the vehicle is in operation. If the residuals surpass a threshold then a fault has been detected. This technique can be extended to the angular errors measured on AQUA to detect failures. However, this method is sensitive to underwater current disturbances that would cause the robot to have nonzero angular error.

4.2 Missing Leg

The first fault considered was the case of a missing leg. The nonlinear simulation was used to study the behaviour of the robot under this condition. The SAS from Eq. (2.8) was enabled throughout the simulations, however the autopilot was not used and no disturbances were introduced to the system. This was done with each of

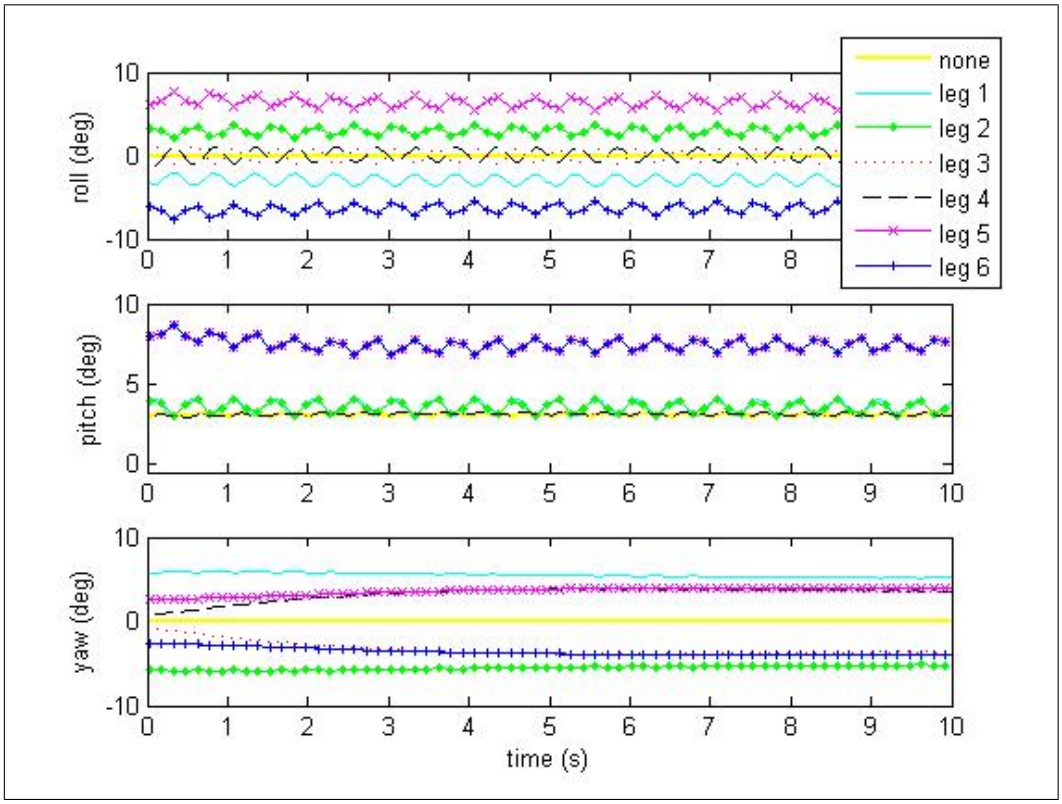


FIGURE 4.1. Angular response with the loss of various legs

the six legs missing on Aqua and Figure 4.1 is the result of these simulations. The figure illustrates that the SAS was able to compensate for a missing leg, albeit with the presence of a small steady state error. The errors have magnitudes less than 10 degrees and this was deemed to be an acceptable error. These results also show that the back legs had the largest effect on the roll and pitch angles, while the front legs affected the yaw angle the most. The loss of the middle legs did not appear to make a significant change to the behaviour of the robot.

It is of some interest to see what effect a lost leg has on the output of the SAS and how it compensated for the failure. From Chapter 2, the output of the SAS is:

$$\boldsymbol{\tau}_{SAS} = \begin{bmatrix} f_{x1} \\ \vdots \\ f_{x6} \\ f_{z1} \\ \vdots \\ f_{z6} \end{bmatrix} = \begin{bmatrix} 0 & 0 & -K_r & 0 & 0 & -K_\psi \\ 0 & 0 & K_r & 0 & 0 & K_\psi \\ 0 & 0 & -K_r & 0 & 0 & -K_\psi \\ 0 & 0 & K_r & 0 & 0 & K_\psi \\ 0 & 0 & -K_r & 0 & 0 & -K_\psi \\ 0 & 0 & K_r & 0 & 0 & K_\psi \\ K_p & -K_q & 0 & K_\phi & -K_\theta & 0 \\ -K_p & -K_p & 0 & -K_\phi & -K_\theta & 0 \\ K_p & 0 & 0 & K_\phi & 0 & 0 \\ -K_p & 0 & 0 & -K_\phi & 0 & 0 \\ K_p & K_p & 0 & K_\phi & K_\theta & 0 \\ -K_p & K_p & 0 & -K_\phi & K_\theta & 0 \end{bmatrix} \begin{bmatrix} p \\ q \\ r \\ \phi \\ \theta \\ \psi \end{bmatrix} \tag{4.1}$$

Notice that Eq. (4.1) does not include the a/b factor. The analysis in the rest of the chapter will also not include the a/b introduced in Chapter 3.

From Eq. (4.1), it can be seen that the force outputs in the x-direction are dependent on the yaw rate and angle only, while the force commands in the z-direction are dependent on the roll rate, pitch rate, roll angle and the pitch angle. Expanding

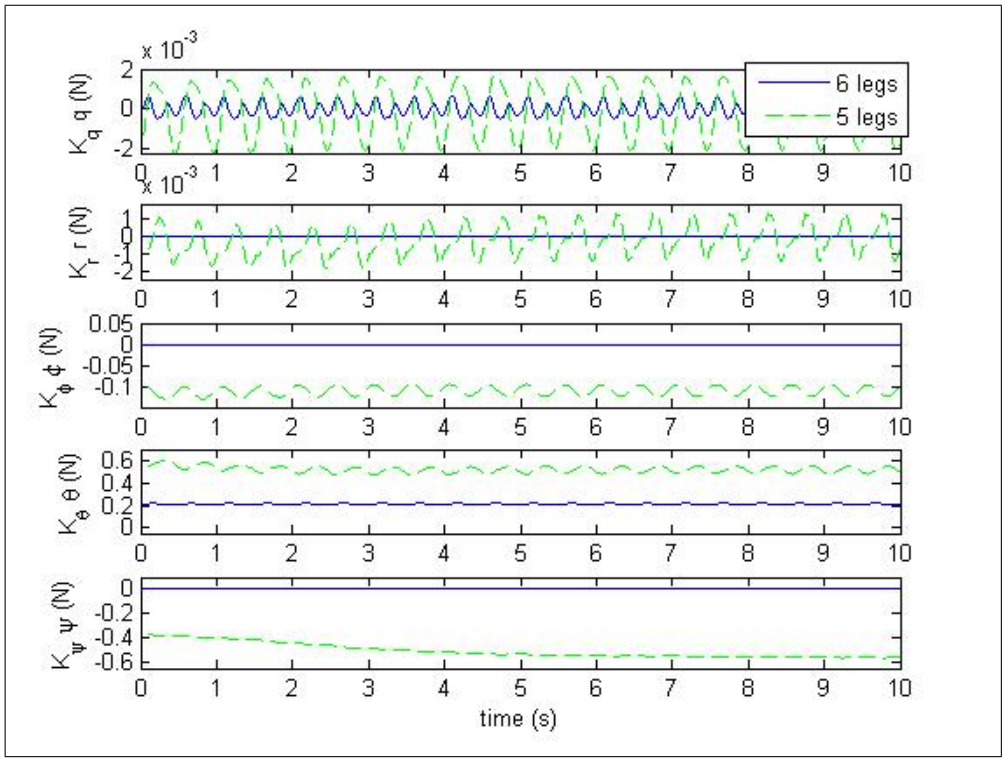


FIGURE 4.2. SAS force contribution from each angle and angular rate

the right hand side of the equation gives:

$$\begin{bmatrix} f_{x1} \\ \vdots \\ f_{x6} \\ f_{z1} \\ \vdots \\ f_{z6} \end{bmatrix} = \begin{bmatrix} -K_r \cdot r - K_\psi \cdot \psi \\ K_r \cdot r + K_\psi \cdot \psi \\ -K_r \cdot r - K_\psi \cdot \psi \\ K_r \cdot r + K_\psi \cdot \psi \\ -K_r \cdot r - K_\psi \cdot \psi \\ K_r \cdot r + K_\psi \cdot \psi \\ K_p \cdot p - K_q \cdot q + K_\phi \cdot \phi - K_\theta \cdot \theta \\ -K_p \cdot p - K_q \cdot q - K_\phi \cdot \phi - K_\theta \cdot \theta \\ K_p \cdot p + K_\phi \cdot \phi \\ -K_p \cdot p - K_\phi \cdot \phi \\ K_p \cdot p + K_q \cdot q + K_\phi \cdot \phi + K_\theta \cdot \theta \\ -K_p \cdot p + K_q \cdot q - K_\phi \cdot \phi + K_\theta \cdot \theta \end{bmatrix} \tag{4.2}$$

The changes that occur in the components of the force commands due to the SAS can be seen in Figure 4.2. In this plot, the back left leg was removed and these are the contributions from each of the states and gains to the output forces of the SAS. The plot shows that the output force from the pitch and yaw angles account for most of the difference between having all 6 legs and missing the back left leg. This makes sense since the gains on the pitch and yaw angles are the highest ($K_\theta = 4$N/rad and $K_\psi = 8$N/rad). Furthermore, it also makes sense that the force from the yaw angle would be affected when a leg is missing since each paddle is normally asked to produce 4.5N of force in the x-direction to maintain forward velocity $V = 0.49$m/s. Similarly, with a missing paddle 0.2N in the z-direction to correct for the pitching motion from the oscillating paddles is lost. Thus there is also a significant increase in the command from the pitch angle error.

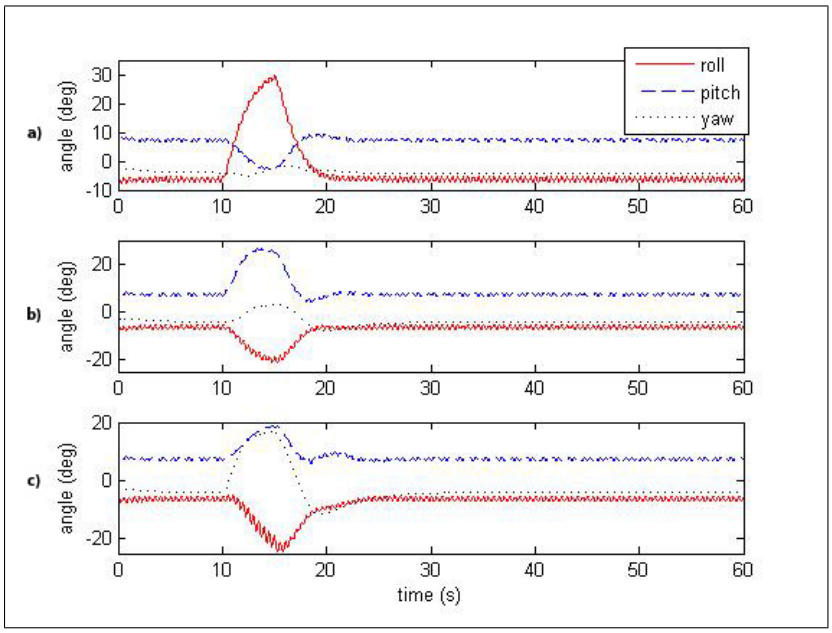


FIGURE 4.3. Angular response to various angular disturbances while missing the back left leg

The fault tolerance was further tested by applying disturbances to the robot missing a flipper. Figures 4.3 show the response of the robot to 0.7N·m moments in the (a)roll, (b)pitch and (c)yaw directions. These disturbances were applied at

t=10s for 5 seconds and then removed. The simulation result shows that the SAS designed in Chapter 2 is able to maintain the robot's stability despite the loss of a leg. The responses show the robot was able to return to the same steady state angles as before the disturbance was introduced. Similar responses were obtained when the other flippers were removed.

4.2.1 *Experimental Results*

The stability augmentation system from Chapter 2 was implemented on the actual robot in a swimming pool and the back left flipper was removed. Similar to the previous experiments, the data from the IMU was logged along with the gains used. Once again the sensor information of the yaw angle was not reliable due to the electronic noise within the robot and therefore the gain on the yaw angle was $K_\psi = 0$ for all the trials. In these trials AQUA's paddles were oscillating at a period of 0.4 s and an amplitude of 0.5 rad, which corresponds to a forward velocity of approximately 0.55m/s. AQUA swam back and forth along the length of the pool (approximately 25 meters) and disturbances were applied to the robot by the pilot with the use of a game pad.

Figure 4.4 displays some of the results from the trials. The solid line illustrates part of one trial where the SAS was not used and it is evident that the roll and pitch angles were not able to stay at zero degrees. From the beginning of the data shown, the roll angle was deviating from zero and at approximately $t = 6$s the pilot attempted to correct the roll angle for 1s. After the input was removed at $t = 7$s, it can be seen that the roll angle began to rise again. Similarly, the pitch angle was not constant and by $t = 9$s angle was fairly large. The pilot attempted to correct the pitch angle for 2s, after which the pitch angle rose again but then stayed somewhat constant around 10 degrees with no further commands from the pilot.

In contrast the dashed and dotted lines in Figure 4.4 represent trials where the SAS was used and it presents a great improvement in the behaviour of the robot. In the trial represented by the dashed line, roll disturbances were applied to the robot at

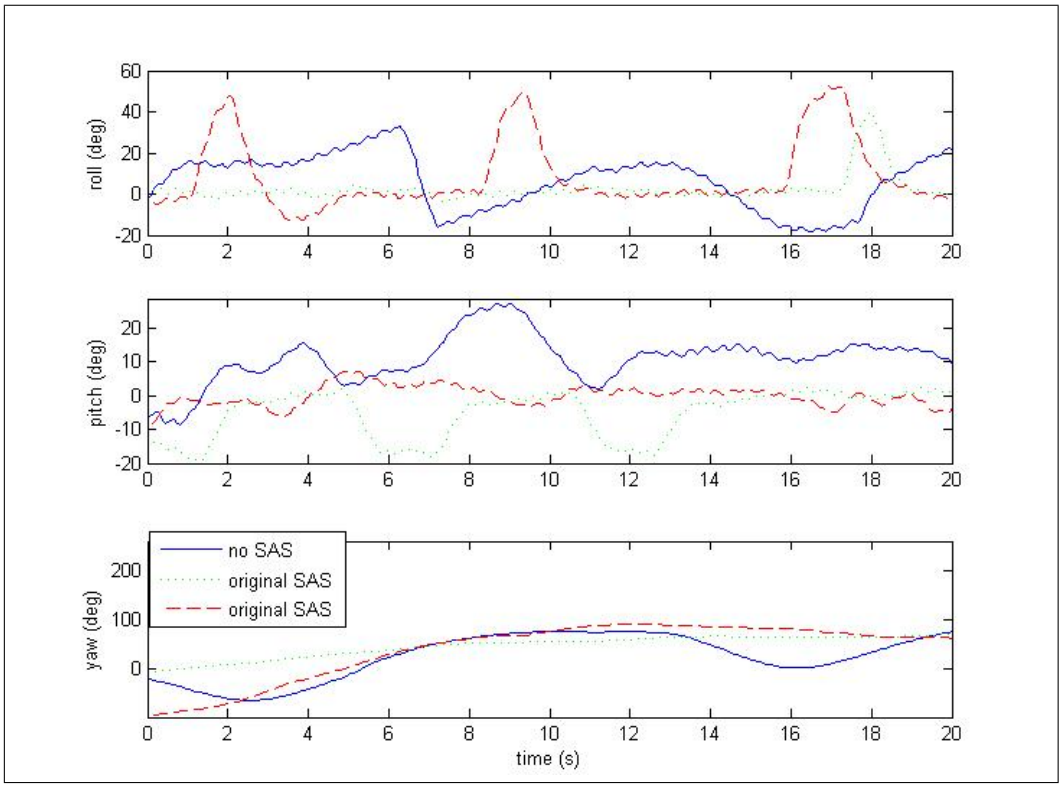


FIGURE 4.4. Experimental results with the back left leg missing

$t = 1$s, 8s and 16s. These disturbance were applied for approximately 1 second and AQUA was allowed to react without any further interference. As the figure shows, the roll angle returned to zero within 1 second after the disturbance was removed and there was a small overshoot after the first disturbance. The time between the disturbances shows that the SAS was able to maintain the roll angle at ± 2 degrees and the pitch angle at ± 5 degrees.

The dotted line represents yet another trial where the SAS was used to stabilize the robot. Here pitch disturbances were applied at $t = 0$s, 5s and 11s for approximately 2 seconds and a final roll disturbance was applied at $t = 17$s for 1 second. Again, the robot was able to recover from each disturbance within a second after the disturbance was removed. Furthermore, the roll and pitch angles were maintained at ± 5 degrees in the time intervals between disturbances.

4.3 A Stuck Leg

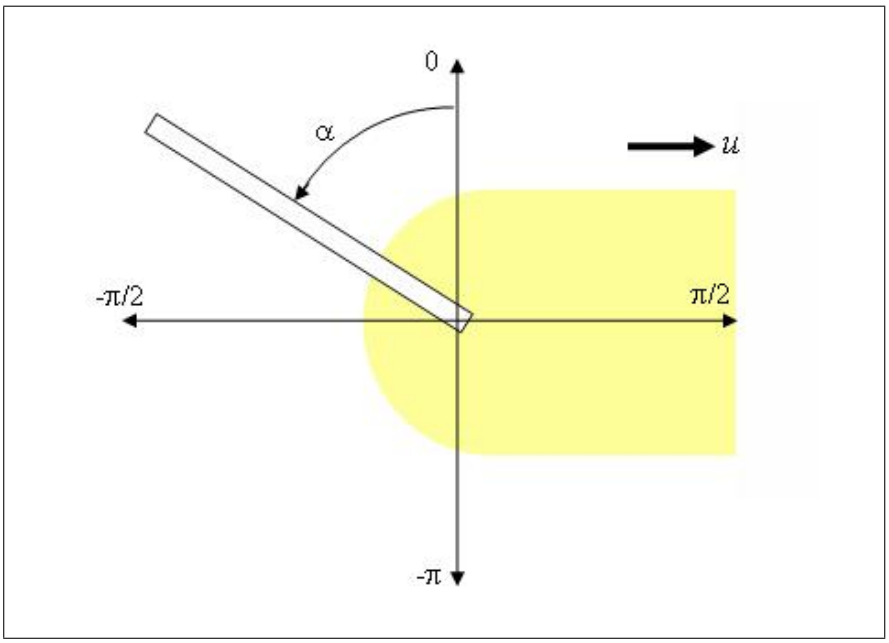


FIGURE 4.5. Flipper angle

The next fault investigated was the case of a leg being stuck at a fixed angle. The goal was to design a SAS that would compensate for the failure without knowing what angle the leg was stuck at or which leg was stuck. Figure 4.5 illustrates how the flipper angle, α, is measured with respect to the rest of the robot.

Simulation of the robot was done for various legs stuck at various angles between 0 and -π, since Aqua's flippers stay within this range in normal operations. In these simulations the SAS from Eq. (2.8) was implemented but the autopilot was not used, nor were there any external disturbances. Furthermore, it was assumed that the stuck flipper was rigidly set and that forces from the movement of the robot would not move the flipper from its stuck angle. Figure 4.6 is an example of a set of simulations and it shows the effect of the back left leg stuck at various angles. Similar responses were found with the other five legs stuck at various angles and will be discussed later on in this section.

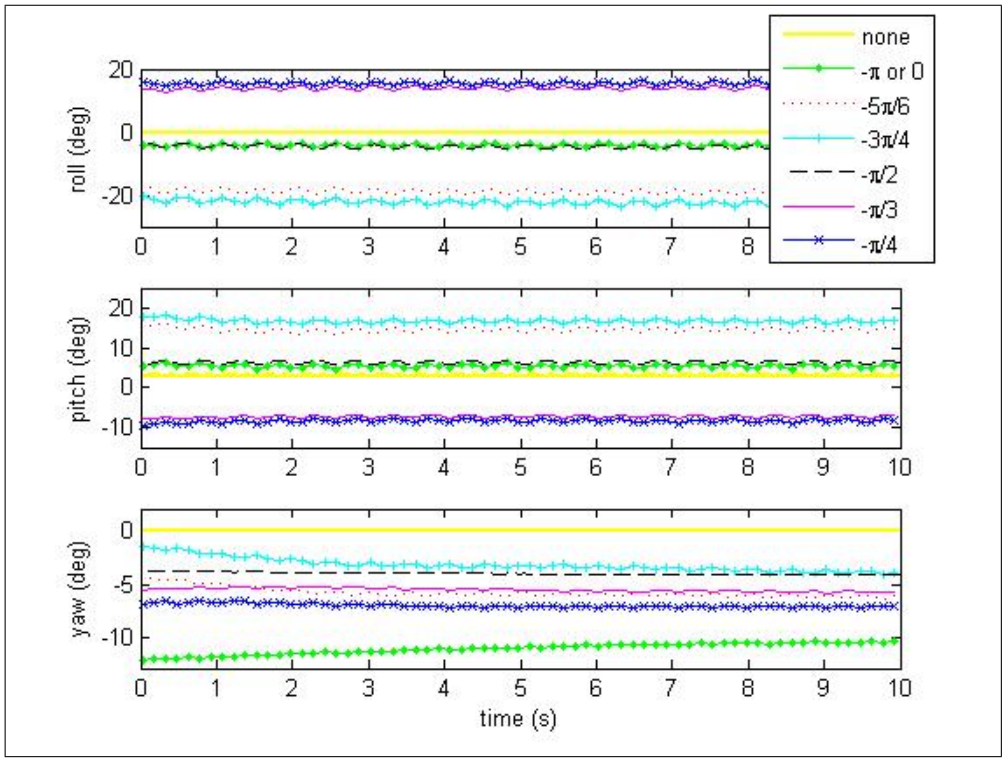


FIGURE 4.6. Response of Aqua to having the back left leg stuck at various angles using the original SAS.

From the figure, it appears that the SAS is unable to compensate for a stuck leg. The robot did not become unstable but settled to a different steady state with a large orientation error. In most cases the steady state error is larger than 10 degrees and this is no longer an acceptable error. There is a more noticeable effect from a stuck leg than the missing leg in the previous section. This is because in addition to the loss of the force in the x-direction to keep a steady forward velocity, there is also an added drag force acting on the robot. The combination of these two effects is large enough that the baseline SAS cannot compensate for it. Since all the angles are strongly coupled to one another, when one angle deviated from zero, others also strayed from the zero. We note that, in many cases, the robot's behaviour appears counterintuitive. For example, the pitch motion is the same whether the leg is stuck at 0 or π. We expect that this is due to the manner in which the SAS tries to compensate for the failure; though this could be investigated further.

It is also worth noting that the case where the leg is stuck at -$\pi/2$, is very similar to the case of the missing leg. This is because at this angle, there is very little to no drag produced by the stuck leg in the direction of forward velocity. Therefore, the only consequence of the stuck leg is the loss of the force in the x-direction to keep a steady forward velocity, which is the same as when a flipper is lost.

Figure 4.6 also shows that the roll and pitch were most affected when a flipper was stuck at -$\pi/4$ or -$3\pi/4$. On the other hand, the yaw angle was most affected when the flipper is stuck at 0 or -π. This makes sense since at 0 and -π, the flipper is perpendicular to the direction of motion and thus the drag forces in the x-direction are at a maximum. Since the greatest effects were observed at these angles, the design of a new SAS was based on them. It was assumed that if the new SAS could compensate for these extreme cases, it could also compensate for other intermediate angles.

To begin, the drag forces acting on the robot in the extreme cases (flipper angles -π and -$3\pi/4$) were examined. When the flipper is at -π, the drag force is acting only in the x-direction and is defined as:

$$F_d = \frac{1}{2}\rho v^2 C_d A \tag{4.3}$$

where $\rho = 1000\text{kg/m}^3$ is the density of water

$v = 0.49\text{m/s}$ is the forward velocity of Aqua

$C_d = 1.2$ is the drag coefficient (taken from [**13**])

$A = 0.014\text{m}^2$ is the area of a flipper

Based on this simplified analysis, the drag force created by the flipper when it is perpendicular to the forward velocity of the robot is approximately 2.1N. Figure 4.7 illustrates the case where the back left flipper is stuck and the shows the drag force created by the flipper. Since the flippers are approximately 0.185m from the x-axis of the robot frame, a moment of 0.389N·m is produced and if the back left leg is stuck, it causes the robot to yaw in the negative direction (to the left).

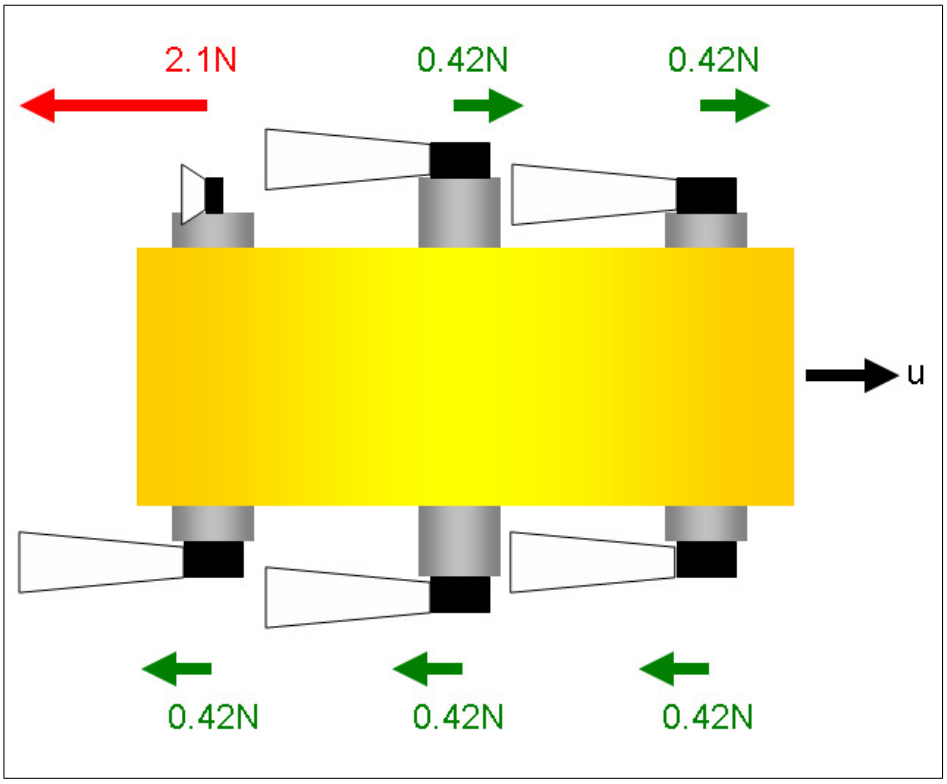


FIGURE 4.7. Forces contributed by each flipper

Therefore, the stuck leg creates an additional drag force and drag moment on the robot. The remaining working five flippers need to compensate for both of these effects by changing the forces they produce. It is possible to resolve both the drag force and drag moment by solving the following equations for F_i:

$$F_1 + F_2 + F_3 + F_4 + F_5 = 2.1 \tag{4.4}$$

$$-aF_1 + aF_2 - bF_3 + bF_4 - aF_5 = 0.389 \tag{4.5}$$

where F_i is the force produced by the ith leg

$a = 0.185$m is the distance between legs 1, 2, 5, 6 and the body's x-axis, and

$b = 0.24$m is the distance between the legs 3, 4 and the body's x-axis

Notice that there are five forces to solve for and only two equations, which makes it an under determined system and there are an infinite number of solutions. It is possible to obtain a minimum norm solution to this system of equations. Unfortunately, the

results obtained showed an uneven distribution of forces to be produced by each of the flippers. Specifically, the flippers on one side of the robot must produce much larger forces than the flippers on the other side to compensate for the drag force and moment. This is undesirable because this would mean a different solution if the fault was in another leg, which would lead to having multiple sets of gains that depend on where the fault has occurred.

Therefore, another approach was used and only the drag moment was resolved. This approach aims to have the remaining working five flippers produce the same thrust to compensate for the fault. Since a flipper stuck at 0 or -π creates a drag moment of 0.389N·m, then the other working flippers need to produce a net moment of 0.389N·m about the robot's z-axis to prevent the robot from yawing. Assuming the other five flippers are also 0.185m away from the x-axis, then this is equal to having each of the five flippers produce an additional 0.42N in the x-direction to counteract the drag moment. This is shown in Figure 4.7 where, if the back flipper was stuck, then it would mean the two legs on the same side of the stuck flipper would have to provide additional forces in the positive x-direction and the flippers on the other side would have to create differential forces in the negative x-direction.

To calculate the gain needed to do this, the five functional flippers needed to provide 0.42N in the x-direction when the yaw angle is at some acceptable steady state error. The decision for an acceptable error was based on the results from chapter 2, where the yaw angle remained between 2-4 degrees when the SAS was used. Thus with an error of 2 degrees or $\pi/90$ rad, the extra gain that needed to be added to the existing K_ψ was calculated in the following way:

$$\Delta K_\psi = \frac{F_d}{\pi/90} = 12N/rad \tag{4.6}$$

Since a similar analysis for the effect on roll and pitch would have been overly involved, we choose to simply increment for K_ϕ and K_θ by 8.5N/rad, as this value was found to give reasonable results.

The gains on the angular rates were also investigated but it was found that they did not affect the results much. Thus, ultimately only the gains on the roll, pitch and yaw angles were changed. With these new gains, the SAS gain matrix was

$$\mathbf{K}_{SAS} = \begin{bmatrix} 0 & 0 & -6 & 0 & 0 & -20 \\ 0 & 0 & 6 & 0 & 0 & 20 \\ 0 & 0 & -5 & 0 & 0 & -15 \\ 0 & 0 & 5 & 0 & 0 & 15 \\ 0 & 0 & -6 & 0 & 0 & -20 \\ 0 & 0 & 6 & 0 & 0 & 20 \\ 0 & -1 & 0 & 9.5 & -12.5 & 0 \\ 0 & -1 & 0 & -9.5 & -12.5 & 0 \\ 0 & 0 & 0 & 7.5 & 0 & 0 \\ 0 & 0 & 0 & -7.5 & 0 & 0 \\ 0 & 1 & 0 & 9.5 & 12.5 & 0 \\ 0 & 1 & 0 & -9.5 & 12.5 & 0 \end{bmatrix} \tag{4.7}$$

Simulations were run again with the back leg stuck at various angles. Figure 4.8 shows the results using the same angles as in Figure 4.6, and there is a definite improvement in the angular response. The roll angle is reduced to ± 4, the pitch angle is reduced to between -4 and 6 degrees and the yaw angle is reduced to between -1 and -4 degrees. This level of error is considered acceptable but could be reduced by further gain adjustments.

To show that the modified $\mathbf{K}_{SAS}$ was successful at compensating for any of the six legs being stuck, the simulation was run with each of the six legs stuck. Since the worse response was seen when the back left leg was stuck at $-3\pi/4$, this was the angle used with each of the other five legs. First, the simulation was done using the original SAS and responses of the robot can be seen in Figure 4.9. Once again, it appears that a failure in the back legs results in the greatest effect on the roll and pitch angles of the robot, while the front legs affect the yaw angle the most. The angular steady state errors, particularly in roll and pitch, created by the stuck legs are larger than 10 degrees and are not acceptable.

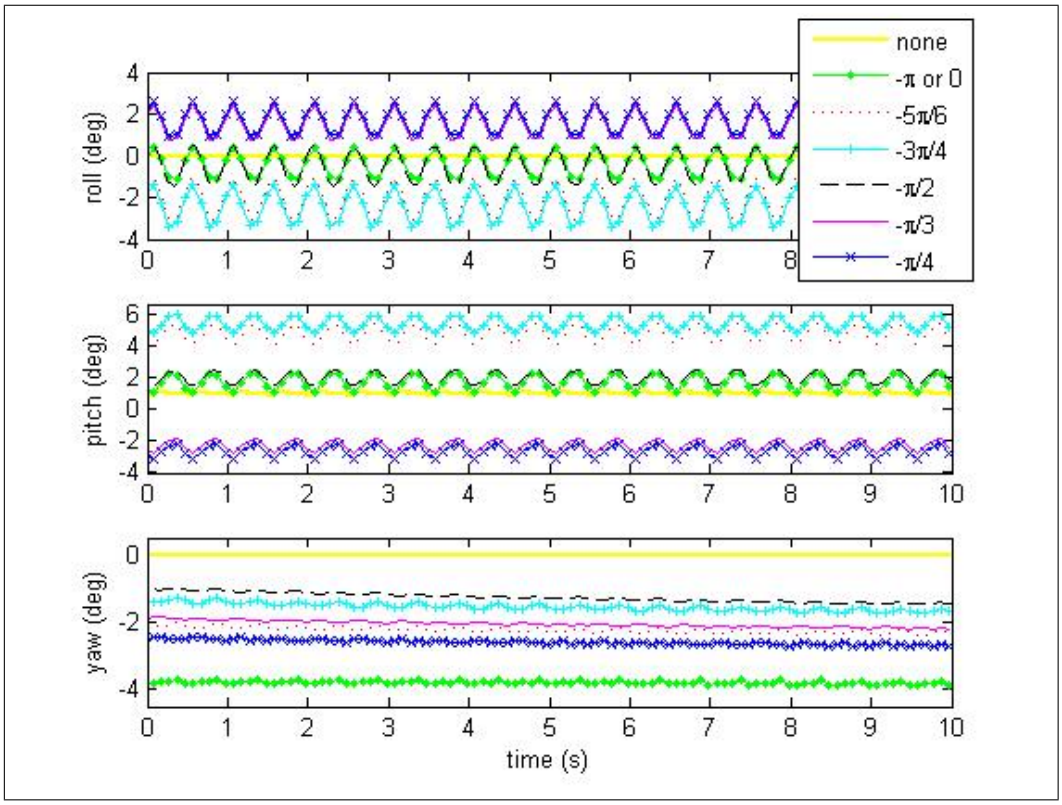


FIGURE 4.8. Response of Aqua to the back left leg stuck at various angles using the new SAS

Next the modified SAS was implemented in the simulation with each of the legs stuck at -3π/4rad. Figure 4.10 shows the response of the robot in these simulations and this illustrates that the new SAS can successfully compensate for failures in any of the six flippers. The angles were all significantly reduced and in most cases the errors are well below 10 degrees. This would allow the robot to swim in a more level and controllable fashion. These results demonstrate that the modified SAS does not need to take into account which leg was stuck, nor at which angle the leg was stuck at.

Finally disturbances were applied to the system to determine if the modified SAS is able to compensate for them. Figure 4.11 shows the response of AQUA to the same 0.7N·m moment disturbances seen in Section 4.2. The disturbances in the (a)roll, (b)pitch, and (c)yaw disturbances were applied at t = 10s and lasted for 5

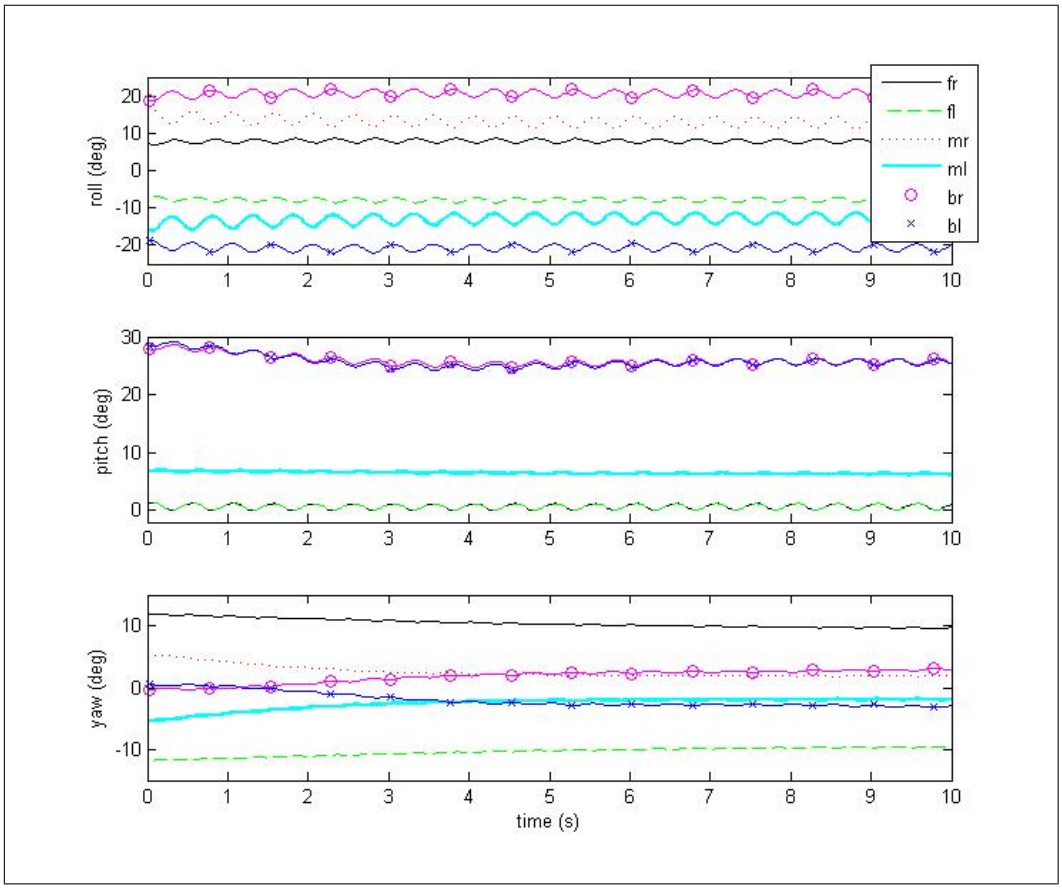


FIGURE 4.9. Response of Aqua to various legs stuck at -3π/4 rad and using the original SAS.

seconds. The figure shows that after the disturbances were applied, the robot was once again able to return to the same angles as it was at before the disturbances were applied.

4.3.1 *Experimental Results*

Similar to the previous section on the missing leg, the modified SAS was implemented on the physical robot in a swimming pool. One set of trials were done with the back left flipper fixed at -π/2 rad and another was done with the flipper fixed at -3π/4 rad. The data from the IMU was logged along with the gains used. As before, the gain on the yaw angle was $K_\psi = 0$ for all the trials, since the measured yaw angle was not reliable. The parameters of the paddle oscillations were the same

as in the previous section, where the period of oscillation was 0.4s and the amplitude was 0.5rad. This corresponded to a forward velocity of approximately 0.55m/s as AQUA swam back and forth along the length of the pool (approximately 25 meters). Disturbances were given by the pilot with the use of a game pad but only when a steady state could be reached by the robot. In some cases this was not possible and since the robot is a fairly delicate piece of equipment, the pilot occasionally gave roll, pitch or yaw commands to keep AQUA from rotating too much.

Figure 4.12 displays some of the results obtained in the trials where the back left leg was fixed at -$\pi/2$ rad. The solid line is the response of the robot without the use of any SAS. Only a roll disturbance was applied to the robot in this set of data at approximately $t = 8$s for 1.5 seconds. When the disturbance was removed the

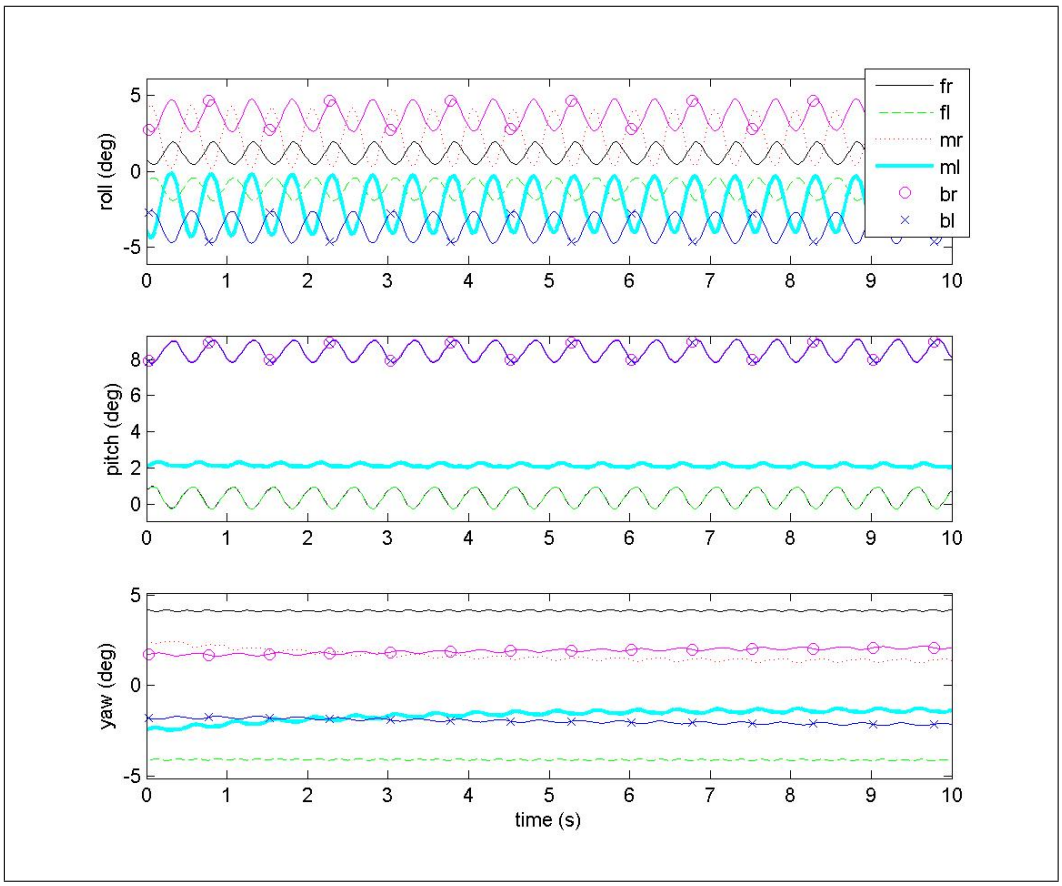


FIGURE 4.10. Response of Aqua to various legs stuck at -$3\pi/4$ rad and using the new SAS.

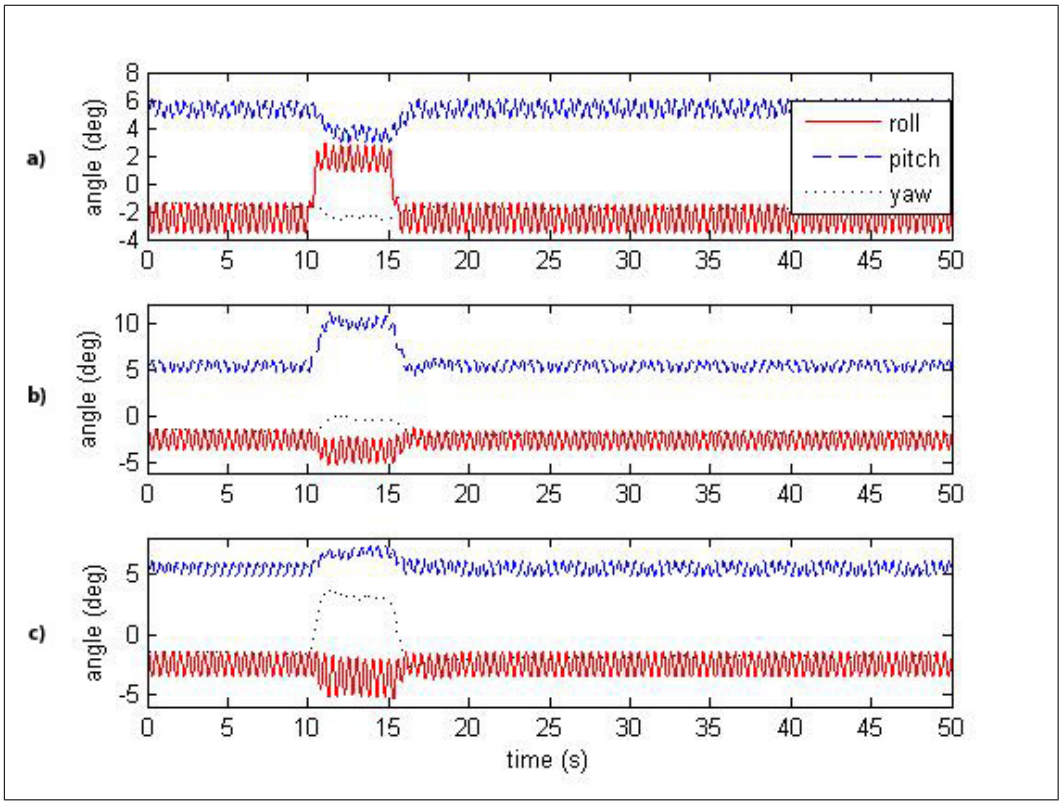


FIGURE 4.11. Response of AQUA to disturbances with the back left leg stuck at -3π/4 rad.

roll angle remained the same as at the moment the disturbance was removed. The pilot attempted to return the roll angle to zero at $t = 10.5$s by giving a corrective impulse in the opposite direction. This command was applied for about one second, after which the roll angle maintained the same angle as when the disturbance was removed. However this steady state angle is 40 degrees which is very large. The pitch angle within this trial goes from approximately -25 degrees to 30 degrees, which is not desirable at all.

The dashed line represents the response of the robot with the use of the original SAS from Chapter 2. Within this trial, roll disturbances were applied at $t = 8$s and 12.5s for one second each. After each disturbance was removed the roll angle returned to zero within one second. Furthermore, the roll angle between disturbances remained at ± 5 degrees, which is an acceptable error and is a large improvement compared to

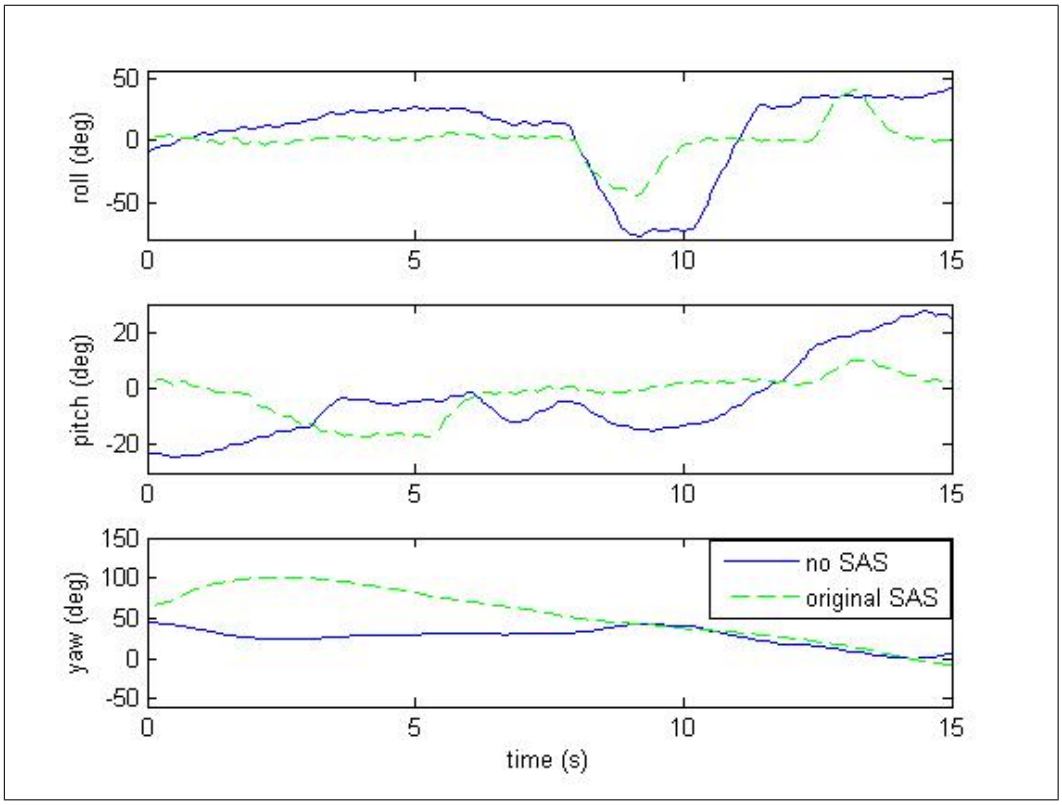


FIGURE 4.12. Experimental response of the robot with the back leg leg fixed at -$\pi/2$ rad.

the trial without a SAS. The pitch angle was improved as well but the results are not as satisfactory as with the roll angle. At $t = 2$s a pitch disturbance was applied for approximately 2 seconds, after which the pitch angle remained at the same angle as when the disturbance was removed. A corrective input was applied at $t = 5.5$s for one second, which brought the pitch angle back close to zero degree. After this, the pitch angle remained fairly close to zero with small deviations when the roll disturbance was applied, showing that changes to the roll angle has an effect on the pitch angle.

The modified SAS was not used with this scenario because at the time of the experiments, it appeared that the behaviour of the robot with the flipper fixed at -$\pi/2$ rad was very similar to that observed with the missing flipper. As AQUA swam, it appeared that the original SAS was sufficient to compensate for the fixed flipper and thus it was presumed that the higher gains of the modified SAS were not needed.

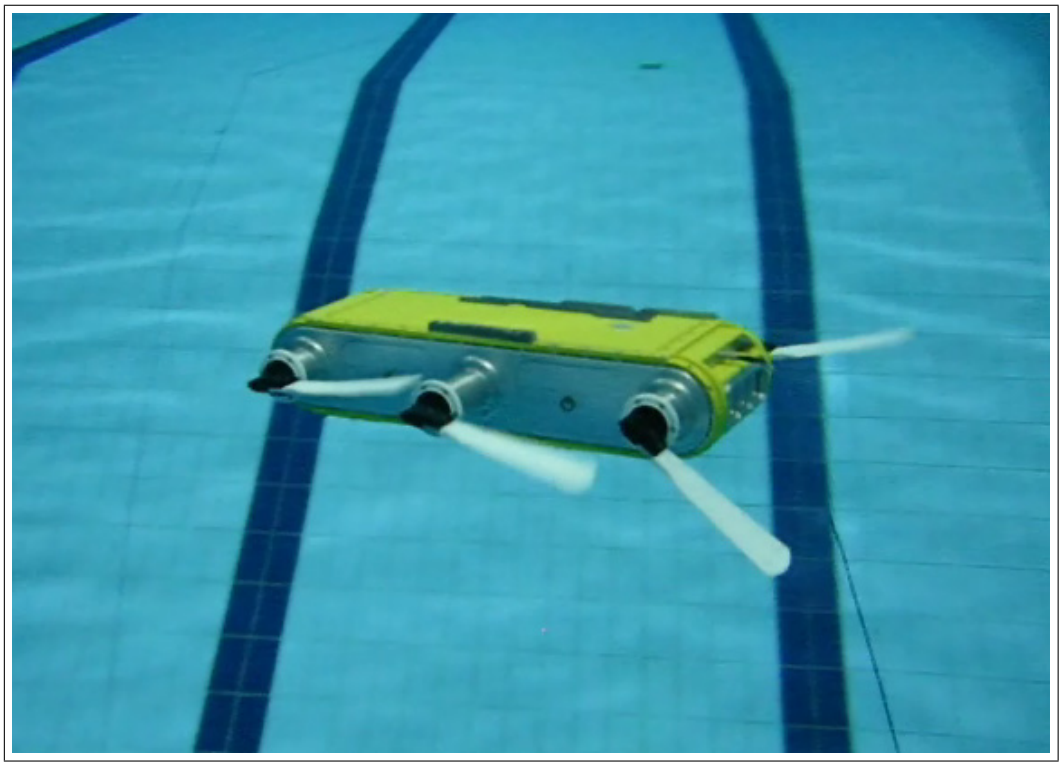


FIGURE 4.13. AQUA during the experiment with the back left leg fixed at -3π/4 rad

Another set of trials were preformed with the back left leg fixed at -3π/4 rad, pictured in Figure 4.13, and the results of a few of these trials are shown in Figure 4.14. The solid line represents the response of the robot without the use of any SAS. In this trial no disturbances were given to the robot since it could not reach any steady state. At $t = 1.5$s the roll angle was becoming too large and the pilot gave a one second corrective command to return the angle to a value close to zero. However, the robot drifted away from zero immediately after the command was removed and continued to drift until $t = 8$s, at which point, the pilot gave another one second corrective command to return the roll angle to zero again. The angle began to drift yet again and a final roll corrective command was given at $t = 15$s. The behaviour of the pitch angle was also erratic and varied between -20 and 45 degrees.

The response of the robot is greatly improved with the use of the original SAS from Chapter 2, which is represented by the dashed line. Roll disturbances were

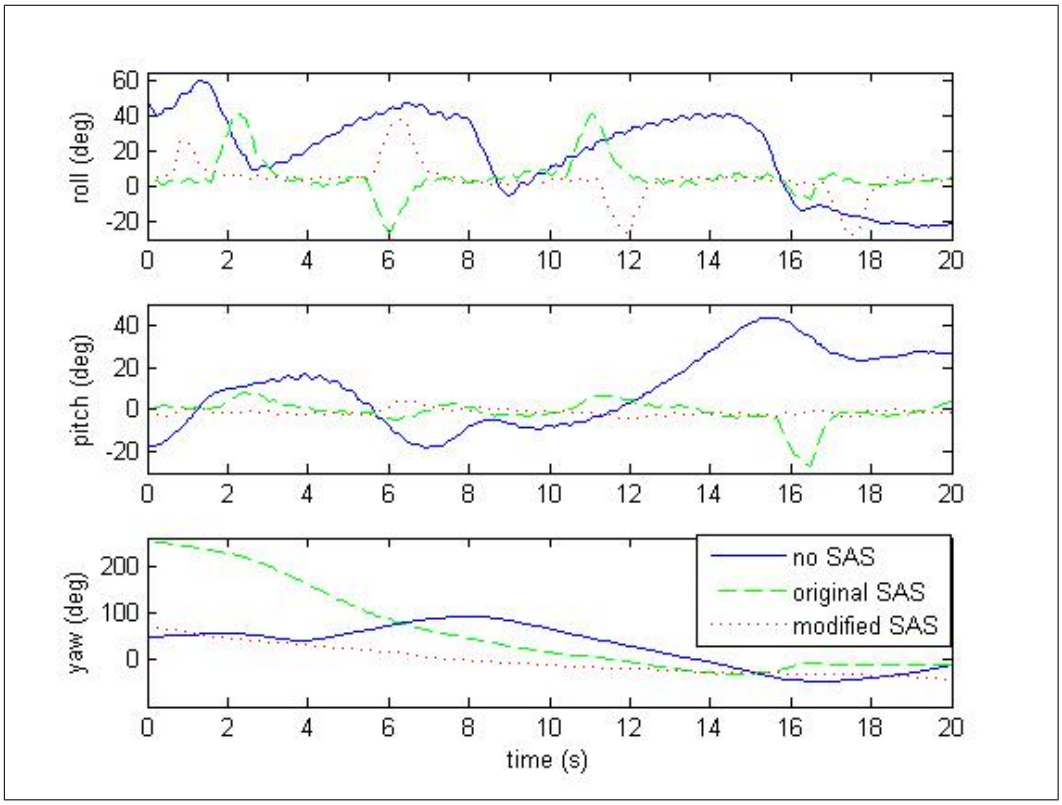


FIGURE 4.14. Experimental response of the robot with the back left leg fixed at -3π/4 rad

applied at $t = 1.5$s, 5.5s, and 10.5s for approximately one second each and the robot was able to recover from the disturbance within a second after the disturbance was removed. AQUA was also able to recover from a pitch disturbance which was applied for one second at $t = 15.5$s. There were smaller disturbances in the pitch angle whenever a roll disturbance was applied, which indicates that the pitch angle is strongly affected by changes to the roll angle. The converse is also seen where a small disturbance in the roll angle occurred when the pitch disturbance was applied.

Finally, the modified SAS from Section 3 was implemented on AQUA but the K_ϕ and K_θ used was slightly lower than what was calculated. In the figure the robot's response is represented by the dotted line with $K_q = 1$N·s/rad, $K_\phi = 8$N/rad and $K_\theta = 12$N/rad. Here, roll disturbances were applied at $t = 0.5$s, 5.5s, 11s and 17s with each one lasting between 0.5 to 1 second. The figure clearly shows that the robot was

able to recover within a second after each disturbance was applied. The roll response on the robot with the modified SAS is similar to the response with the original SAS. However, a difference can be seen in the pitch response. With the original SAS, the pitch angle varied by about ±8 degrees but with the modified SAS, the response was smoother with the angle varying by approximately ±5 degrees.

To summarize, a list of the gains used in this section are shown in Table 3.1. When these gains were implemented, the a/b factor that was introduced in Chapter 3 was included.

TABLE 4.1. Angular and angular velocity gains used in this section

SAS version	K_p N·s/rad)	K_q (N·s/rad)	K_r (N·s/rad)	K_ϕ (N/rad)	K_θ (N/rad)	K_ψ (N/rad)
original in simulation	0	1	6	1	4	8
modified in simulation	0	1	6	9.5	12.5	20
original in experiment	0	1	0	4	8	0
modified in experiment	0	1	0	8	12	0

CHAPTER 5

Conclusions and Recommendations for Future Work

5.1 Conclusions

In this work, the objective was to design a system that stabilizes and controls an underwater vehicle that uses oscillating paddles to propel itself. A stability augmentation system was first designed to reduce the effect of disturbances and perturbations acting on the robot. This was then followed by the design of an autopilot to work in combination with the SAS. Finally, the SAS was modified to compensate for certain leg failures that may occur on the robot during operation.

The issue of stability was first examined by linearizing the robotic system by using numerical differentiation by finite difference in Chapter 2. Linear representations were derived for three different steady state forward velocities. The **A** and **B** state matrices were validated through MATLAB simulations and were then used to create a stability augmentation system. By studying the eigenvalues of the system, it was found that the system is naturally unstable. The gain matrices were derived for each of the steady state forward velocities using linear control design theory. Unfortunately, when the gain matrices were implemented in the non-linear model, the response of the robot did not coincide with what was observed with the linear model. Further refinements of the gain matrices were done through trial and error using the non-linear model.

Although, there were different responses between the linear and non-linear model, the linear model was a useful starting point which gave information on the stability thresholds of the robot.

Since the forward velocity varies during the normal operation of the robot, a gain scheduler was designed that combined the three gain matrices. The gains for any given forward velocity was found by taking the linear interpolation between the three gain matrices. It was shown in simulation that gain scheduling was successful at reducing perturbations to zero, particularly when the velocity of the robot was changing. However, the system could not be implemented on the physical robot since the measurement of the velocity is not yet reliable.

Successful performance of the SAS was demonstrated in a set of experiments at a steady state forward velocity of about $V = 0.49$m/s. The SAS was evaluated quantitatively and qualitatively in both a swimming pool and in sea water. Quantitatively, the results showed that the SAS was capable of reducing the effects of external disturbances acting on the robot. The robot was able to return to roll and pitch angles that were very close to zero within a few seconds after each disturbance was applied. Qualitatively, the inclusion of the SAS to the robotic system was an improvement to the controllability of the vehicle. Specifically, it made it easier to conduct coral transects in the ocean by keeping the robot steady as it recorded video.

The design of an autopilot to work in conjunction with the SAS was presented in Chapter 3. The initial design was a proportional controller to be used at $V = 0.49$m/s with gains determined through trial and error. As one of the angles in the robot's orientation moved away from zero, the command from the SAS grew as it tried to return the angle to zero. In order to overcome these commands from the SAS, the autopilot had very large gains. Unfortunately, the attempt to implement the autopilot on the actual robot was not successful, resulting in large oscillations in the robot behaviour. This was partially due to the difference in mapping force commands from the controller to paddle movements between the simulation and the actual robot. Furthermore, since the gains were very large, small changes in angular error would

result in large changes in the force commands. Therefore, the autopilot was redesigned to include integral gain in order to reduce the proportional gains and thereby reduce the amplitude of the oscillations in robot's behaviour. Tests in simulation showed promising results but the resulting autopilot was not tested on the physical vehicle, due to our limited access to the test facility.

The final area of focus was on the robot's fault tolerance, which was addressed in Chapter 4. Two different failures were investigated: the loss of a flipper and a stuck flipper due to a motor failure. It was found that the original SAS designed in Chapter 2 was robust enough to compensate for the case of a lost flipper. Although there were angular errors associated with the loss of a flipper, the error was deemed to be small enough to be acceptable. This was also verified experimentally, where AQUA was able to swim in a level fashion and was able to recover from perturbations applied to it.

The case of the fixed flipper could not be compensated by the original SAS and therefore it was modified. This was done by evaluating the drag force created by the fixed flipper and modifying the SAS such that the remaining working flippers could produce forces to counteract the additional drag. This was done by increasing the gains such that the force commands were sufficient to overcome the additional drag at some acceptable angular error. The modified SAS was shown to work both in simulation and in experiments, by keeping the vehicle leveled despite the stuck leg and reducing the effects of external perturbations. With this approach, it was not necessary to know which leg had failed, nor was it necessary to know at what angle the failed flipper was stuck at.

5.2 Recommendations for Future Work

The following are some recommendations for future work on the control system of AQUA.

- Verify experimentally that the SAS will work for forward velocities other than 0.5m/s.

- In all the experiments, the yaw angle was not fed back because the measurements from the compass were not reliable. However, the yaw rate can be consistently obtained from the IMU. A method to compute the yaw angle based on the other data from the IMU could allow the yaw angle to be used in the feedback for the SAS and autopilot.

- In Chapter 2 a gain scheduler was designed but was not implemented on the vehicle. This was because the measurement of translational velocity and position from the IMU is not reliable due to the drift that occurs in the measurements. It would be advantageous to design a system or filter to allow for more accurate measurements of the translational states.

- Recently a depth sensor has been added to the robot and should be included in the SAS.

- It is intuitive to give a desired trajectory in terms of x, y, and z rather than angles or velocities. Therefore, it would be beneficial to be able to measure the position of the robot at any given time. This could be done by extending the system from measuring translational velocities to position.

- The design of the autopilot in Chapter 3 was not successfully implemented on the actual robot. Further pool trials are needed to assess the performance of the controllers.

- Design a system to correctly identify that a leg is stuck on the robot. This could be based on the method mentioned in Section 4.1 where the average angular errors are monitored.

REFERENCES

[1] A. P. Aguiar and A. M. Pascoal. Global stabilization of an underactuated autonomous underwater vehicle via logic-based switching. *Proceedings of IEEE Conference on Decision and Control*, 3:3267–3272, 2002.

[2] P. E. ao and A. Pascoal. 3d path following for autonomous underwater vehicle. *Proceedings of IEEE Conference on Decision and Control*, pages 2977–2982, 2000.

[3] D. Barrett, M. Grosenbaugh, and M. Triantafyllou. The optimal control of a flexible hull robotic undersea vehicle propelled by an oscillaing foil. *Proceedings of the IEEE Symposium on Autonomous Underwater Vehicle Technology*, pages 1–9, 1996.

[4] M. Buehler, U. Saranli, and D. Koditschek. *Single actuator per leg robotic hexapod*. McGill University, The Regents of the University of Michigan:USA, US Patent: 6,481,513, 2002.

[5] S. Choi and J. Yuh. Experimental study on a learning control system with bound estimation for underwater robots. *Proceedings for the IEEE International Conference on Robotics and Automation*, pages 2160–2165, 1996.

[6] V. Djapic, J. A. Farrell, P. Miller, and R. Arrieta. Advanced non-linear control algorithms applied to design highly maneuverable autonomous underwater vehicles (auvs). *Proceedings of the International Symposium on Unmanned Untethered Submersible Technology*, 2007.

[7] K. Do, Z. Jiang, J.Pan, and H. Nijmeijer. A global output-feedback controller for stabilization and tracking of underactuated odin: A spherical underwater vehicle. *Automatica*, 40:117–124, 2004.

[8] G. Dudek, P. Giguere, and J. Sattar. Sensor-based behavior control for an autonomous underwater vehicle. In *IEEE International Symposium on Experimental Robotics*, pages 312–317, 2006.

[9] G. Dudek, J. Sattar, and A. Xu. A visual language for robot control and programming: A human-interface study. In *IEEE International Conference on Robotics and Automation (ICRA07)*, pages 2507–2513, Rome, Lazio, Italy, 2007.

[10] F. E. Fish. Balancing requirements for stability and maneuverability in cetaceans. *Integrative and Comparative Biology*, 42:85–93, 2002.

[11] T. I. Fossen and J. Balchen. Modelling and non-linear self-tuning robust trajectory control of an autonomous underwater vehicle. *Modelling, Identification and Control*, 9(4):165–177, 1988.

[12] R. R. Fullmer, R. W. Gunderson, and G. Olsen. The preliminary design of a stability augmentation system for a remotely piloted vehicle in hover. *IEEE Conference on Control Applications*, 2:825–826, 1992.

[13] C. Georgiades. Simulation and control of an underwater hexapod robot. Master's thesis, McGill University, 2005.

[14] C. Georgiades, A. German, A. Hogue, H. Liu, C. Prahacs, A. Ripsman, R. Sim, L. Torres, P. Zhang, M. Buehler, G. Dudek, M. Jenkins, and E. Milios. Aqua: An aquatic walking robot. *IEEE/RSJ International Conference on Robotics and Systems, IROS*, 4:3525–3531, 2004.

[15] P. Giguere, G. Dudek, and C. Prahacs. Characterization and modeling of rotational responses for an oscillating foil underwater robot. *Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems*, pages 3000–3005, 2006.

[16] J. Guo and F.-C. Chiu. Maneuverability of a flat-streamlined underwater vehicle. *Proceedings of IEEE International Conference on Robotics & Automation*, pages 897–902, 2001.

[17] S. Hsu, C. Mailey, E. Eade, and J. Janét. Autonomous control of a horizontally configured undulatory flap propelled vehicle. *Proceedings of the IEEE International Conference on Robotics & Automation*, pages 2194–2199, 2003.

[18] P. Kiriazov, E. Kreuzer, and F. C. Pinto. Robust feedback stabilization of underwater robotic vehicles. *Robotics and Autonomous Systems*, 21:415–423, 1997.

[19] M. V. Kumar, P. Sampath, S. Suresh, S. N. Omkar, and R. Ganguli. Design of a stability augmentation system for a helicopter using lqr control and ads-33 handling qualities specifications. *Aircraft Engineering and Aerospace Technology*, 80(2):111–123, 2008.

[20] R. Lea and S. Merry. A comparative study of control techniques for an underwater flight vehicle. *International Journal of Systems Science*, 30(9):947–964, 1999.

[21] N. E. Leonard. Control synthesis and adaptation for underactuated autonomous underwater vehicle. *IEEE Journal of Oceanic Engineering*, 20(3):211–220, 1995.

[22] N. E. Leonard. Periodic forcing, dynamics and control of underactuated spacecraft and underwater vehicles. *Proceedings of the Conference on Decision & Control*, pages 3980–3985, 1995.

[23] N. E. Leonard. Stabilization of steady motions of an underwater vehicle. *Proceedings of the Conference on Decision and Control*, pages 961–966, 1996.

[24] N. E. Leonard. Stability of a bottom-heavy underwater vehicle. *Automatica*, 33(3):331–346, 1997.

[25] S. Licht, F. Hover, and M. Triantafyllou. Maneuvering "finnegan the roboturtle" through force vectoring with oscillating foils. *Proceedings of the International Symposium on Unmanned Untethered Submersible Technology*, 2007.

[26] N. Mišković and M. Barišić. Fault detection and localization on underwater vehicle propulsion systems using principle component analysis. *Proceedings of the IEEE International Symposium on Industrial Electronics*, 4:1721–1727, 2005.

[27] Y. Nakamura and S. Savant. Nonlinear tracking control of autonomous underwater vehicles. *Proceedings of IEEE International Conference on Robotics and Automation*, pages A4–A9, 1992.

[28] M. Narasimhan, H. Dong, R. Mittal, and S. N. Singh. Optimal yaw regulation and trajectory control of biorobotic auv using mechanical fins based on cfd parametrization. *Journal of Fluids Engineering*, 128:687–698, 2006.

[29] L. Ni and C. R. Fuller. Control reconfiguration based on hierarchical fault detection and indentification for unmanned underwater vehicles. *Journal of Vibration and Control*, 9:735–748, 2003.

[30] A. P. Oliva. An altitude hold autopilot design to work with an observer-based stability augmentation control law. *Proceedings of IEEE Conference on Control Applications*, 1:353–358, 1994.

[31] A. Orrick, M. McDermott, D. M. Barnett, E. L. Nelson, and G. N. Williams. Failure detection in an autonomous underwater vehicle. *Proceedings of the IEEE Symposium on Autonomous Underwater Vehicle Technology*, pages 377–382, 1994.

[32] K. Pettersen and O. Egeland. Time-varying exponential stabilization of the position and attitude of an underactuated autonomous underwater vehicle. *IEEE Transactions on Automatic Control*, 44(1):112–115, 1999.

[33] G. J. Rae and S. E. Dunn. On-line damage detection for autonomous underwater vehicles. *Proceedings of the IEEE Symposium on Autonomous Underwater Vehicle Technology*, pages 383–392, 1994.

[34] J. H. Rife and S. M. Rock. Design and validation of a robotic control law for observation of deep-ocean jellyfish. *IEEE Transactions on Robotics*, 22(2):282–291, 2006.

[35] U. Saranli, M. Buehler, and D. Koditschek. Rhex: A simple and highly mobile hexapod robot. *The International Journal of Robotics Research*, 20:616–631, 2001.

[36] J. Sattar. A visual servoing system for an amphibious legged robot. Master's thesis, McGill University, 2005.

[37] J. Sattar and G. Dudek. Where is your dive buddy: tracking humans underwater using spatio-temporal features. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS07)*, San Diego, California, USA, October 2007.

[38] H. Sayyaadi, T. Ura, and T. Fujii. Collision avoidance controller for the auv systems using stochastic real value reinforcement learning method. *Proceedings of the 39th SICE Annual Conference*, 2000.

[39] D. A. Smallwood and L. L. Whitcomb. Model-based dynamic positioning of underwater robotic vehicles: Theory and experiment. *IEEE Journal of Oceanic Engineering*, 29(1):169–186, 2004.

[40] D. Weihs. Stability versus maneuverability in aquatic locomotion. *Integrative and Comparative Biology*, 42:127–134, 2002.

[41] K. Yang, J. Yuh, and S. Choi. Fault-tolerant system design of an autonomous underwater vehicle odin: an experimental study. *International Journal of Systems Science*, 30(9):1011–1019, 1999.

[42] J. Yuh. Design and control of autonomous underwater robots: A survey. *Autonomous Robots*, 8(1):7–24, 2000.

Document Log:

Manuscript Version 0

Typeset by $\mathcal{AMS}$-LaTeX — 18 January 2009

Olivia Min Yee Chiu

Center for Intelligent Machines, McGill University, 3480 University St., Montréal (Québec) H3A 2A7, Canada, *Tel.* : (514) 398-8054

E-mail address: olivia@cim.mcgill.ca

Typeset by $\mathcal{AMS}$-LaTeX