Modeling and Control of a Biomimetic Underwater Vehicle

Nicolas Plamondon

Doctor of Philosophy

Department of Mechanical Engineering

McGill University

Montreal,Quebec

2010-06-01

A thesis submitted to McGill University
in partial fullfillment of the requirements of the degree of
Doctor of Philosophy

Your file Votre référence
ISBN: 978-0-494-72697-6

Our file Notre référence
ISBN: 978-0-494-72697-6

ACKNOWLEDGEMENTS

I would not be where I am without the help and assistance from many people. First, I want to thank my supervisor Meyer Nahon. I am very grateful for his insight, advice, patience, constructive criticism and experience. I would also like to thank my co-supervisor Gregory Dudek for his help and his contagious optimism.

I would like to thank my fellow *MLL* lab-mates who enjoyed the windowless experience with me. Christopher Prahacs, *Wise and Powerful*, for his advice and optimistic presence in the lab. Marco Masciola for the *Helicopter Video*. Michele Faragalli for the never ending lunch and snack breaks and for his colourful encounter with *Jameson*. Finally, I would like to thank Adam Harmat and Olivia Chiu for all the good times together.

I also want to thank the people I worked with during my numerous trips to Barbados. Philippe Giguère for his help on the robot code and for making sure that my stuff was working. Junaed Sattar for his devotion to the project and his special fish feeding technique. Yogesh Girdhar for the funny conversations and parties. Thanks also to all the others: J-P Lobos, Alec Mills, Anqi Xu, Florian Shkurti, Ioannis Rekleitis and Katherine Turgeon.

J'aimerais aussi remercier ma famille pour leur support au cours de mes études. Je tiens aussi à remercier mes amis de longue date Alexis, PO, Julien, Simon, Luc et Justin.

I thank NSERC, FQRNT, MEDA and Professor Meyer Nahon for providing financial support throughout my studies. Thanks also to everyone I could not include.

ABSTRACT

Biomimetic underwater vehicles have recently become a focus of academic research. They offer possibly better maneuverability and higher efficiency than conventional thrusters and the potential for amphibious capability. The focus of this thesis is the modeling and control of biomimetic underwater vehicles, and the specific platform used for experimentation is the Aqua underwater vehicle.

The study begins with the development of a dynamics model for the Aqua underwater vehicle. An existing model was updated to account for the flexibility of the oscillating paddles. The model was validated experimentally and linearized to enable the design of more advanced controllers. Based on the paddle model, we also develop a reverse model/mapping that determines the paddle motion needed to produce a desired thrust.

We design four classes of trajectory tracking controllers: PID, model-based controllers, adaptive controller and Floquet controller. The controllers were first tested in the dynamics simulation to tune the control gains and assess their performance. We found that the adaptive and Floquet controllers were superior to the other two. The adaptive controller was more accurate to track a changing trajectory while the Floquet was better in set point tracking. The controllers were then tested experimentally on the actual Aqua vehicle. The simulation results were partially corroborated but the controllers were not as accurate. The adaptive controller gave better performance in pitch while the Floquet controller was more accurate in roll.

An optimization of a U-turn was performed to improve the performance of the

vehicle in coral reef inspection. Three design variables were chosen for the optimization: the turn radius, the speed and the bank angle. The turn radius and speed had the most impact on the performance while the bank angle had little effect.

RÉSUMÉ

Les véhicules sous-marins biomimétiques sont récemment devenus un domaine de recherche populaire dans les milieux académiques. Ils offrent entres autres la possibilité d'une meilleure maneuvrabilité, une plus grande efficacité et des capacités amphibies. Cette thèse se concentre sur la modélisation et les commandes de véhicules sous-marins biomimétiques. Le véhicule sous-marin Aqua est utilisé comme plateforme d'expérimentation.

Nos recherches ont commencé par le développement d'un modèle dynamique pour le véhicule sous-marin Aqua. Le modèle fut validé expérimentalement puis linéarisé afin de permettre la conception de commandes avancées. Un second modèle, dit inverse, permettant de déterminer le mouvement d'une nageoire en fonction de la poussée désirée fut par la suite développé.

Nous avons conçu quatre classes de commandes: PID, commandes prédictives, commandes adaptatives et Floquet. Ces commandes furent d'abord testées à l'aide d'une simulation informatique afin d'ajuster les gains et d'évaluer la performance des commandes. Nous avons découvert que les commandes adaptives et Floquet étaient supérieures aux deux autres. Les commandes adaptatives étaient plus précises lorsque la trajectoire à suivre variait alors que les commandes de Floquet étaient plus performantes pour suivre un point fixe. Les commandes furent par la suite testées au cours d'une scéance expérimentale en utilisant le véhicule sous-marin Aqua. Les résultats

expérimentaux corroborèrent partiellement ceux de la simulation, mais les commandes n'étaient pas aussi précises. Les commandes adaptatives donnèrent une performance supérieure en tanguage alors que les commandes de Floquet furent supérieures en roulis.

Une optimisation d'un *U-turn* fut effectuée afin d'améliorer la performance du véhicule lors des inspections de récifs de coraux. Trois paramètres d'optimisation ont été sélectionnés: le rayon du tournant, la vitesse et l'angle d'inclinaison. Le rayon du tournant et la vitesse semblaient les deux paramètres ayant le plus d'effet sur la performance alors que l'angle d'inclinaison n'avait presqu'aucun effet.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF PARAMETERS

A	Amplitude of oscillation [*radian*]
A_f	Inlet area of the paddle [m^2]
$\mathbf{A}(t)$	State matrix
$\bar{\mathbf{A}}$	Constant part of $\mathbf{A}$
$\grave{\mathbf{A}}(t)$	Oscillating part of $\mathbf{A}$
$\hat{\mathbf{A}}$	Estimate of $\mathbf{A}$
$\mathbf{b}(n_2)$	Buoyancy force [N]
$\mathbf{B}(t)$	Input matrix
$\hat{\mathbf{B}}$	Estimate of $\mathbf{B}$
$\mathbf{C(v)}$	Matrix of Coriolis terms
C_L	Lift coefficient
C_D	Drag coefficient
$\mathbf{C}(t)$	Output matrix
$\mathbf{D(v)}$	Matrix of hydrodynamics terms
D	Trackline separation [m]
$\mathbf{D}$	Drag force [N]
$\mathbf{e}_a$	Acceleration error vector
$\mathbf{e}_v$	Velocity error vector

$\mathbf{e}_s$	Position error vector
E	Modulus of elasticity [Pa]
E_{RMS}	RMS error of the position of the vehicle
f_{xi} and f_{zi}	Paddle force in the x- and z-direction [N]
$\mathbf{f}$	Vector of forces acting on the vehicle
F_x, F_y and F_z	Total external force components along the x-, y and z-axes [N]
$\mathbf{g}(n_2)$	Gravitational force [N]
$\mathbf{G}(t)$	Controllability matrix
I	Moment of inertia of the paddle [m^4]
I_{xx}, I_{yy} and I_{zz}	Moments of inertia of the robot about the x-, y- and z axes [Nms^2]
$\mathbf{J}$ and $\mathbf{F}(t)$	Floquet factors
$\mathbf{K}_d$	Derivative gain matrix
$\mathbf{K}_p$	Proportional gain matrix
$\mathbf{K}_I$	Integral gain matrix
$\mathbf{K}_s$	Stabilizing control gain matrix
l	Paddle length [m]
$\mathbf{L}$	Lift force [N]
$L_{\dot{p}}, M_{\dot{q}}$ and $N_{\dot{r}}$	Hydrodynamic derivatives [Nms^2]
L_{p^2}, M_{q^2} and N_{r^2}	Drag factors [Nms^2]
m	Mass of the vehicle [kg]

M	Bending moment acting on the paddle [Nm]
$\mathbf{M}$	Inertia matrix
M_x, M_y and M_z	Total external moment components about the x-, y- and z- axes [Nm]
N	Maximum number of generations
P	Period of oscillation $[s]$
p, q and r	Angular velocity components about the body x-, y- and z-axes [radian/s]
R	Turn radius [m]
R_{in}	Desired turn radius in the optimization [m]
s	Shrinking parameter
S	Total distance traveled during the optimization [m]
T	Thrust produced by the paddle [N]
t_s and $t_{s_{min}}$	Settling time, Minimum settling time [s]
U	Flow velocity [m/s]
u, v and w	Components of the vehicle mass center velocity along the body x-, y- and z-axes $[m/s]$
u_{in}	Desired velocity in the optimization [m/s]
$\mathbf{u}$	Control input
v_{in}	Inflow velocity [m/s]
V	Volume of water displaced by the paddle [m^3] or Lyapunov function

$w(x)$	Paddle width at a distance x from the hip [m]
W_A and W_P	Weighting factors on the amplitude and period of oscillation
$\mathbf{x}(t)$	State vector
$\hat{\mathbf{x}}(t)$	Estimate of state vector $\mathbf{x}(t)$
$X_{\dot{u}}, Y_{\dot{v}}$ and $Z_{\dot{w}}$	Hydrodynamic derivatives [kg]
X_{u^2}, Y_{v^2} and Z_{w^2}	Drag factors [$\mathrm{Nm^{-2}s^2}$]
α	Angle of attack [radian]
β	Direction of flow impinging on the paddle [radian]
ϵ	Estimation error
γ, $\dot{\gamma}$	Angular position and angular velocity of the paddle [radian, radian/s]
γ_d, $\dot{\gamma}_d$	Desired angular position and velocity of the paddle [radian, radian/s]
$\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$	Adaptive gain matrices
λ	Paddle offset [radian]
δ	Phase offset of the paddles [radian]
$\boldsymbol{\eta}(t)$	Modal variable
$\boldsymbol{\eta}_d(t)$	Desired value of the modal variable
ω_i	Poincaré exponents
ϕ, θ and ψ	Euler angles (roll,pitch,yaw) [radian]
Θ_i	Eigenvalues of the monodromy matrix

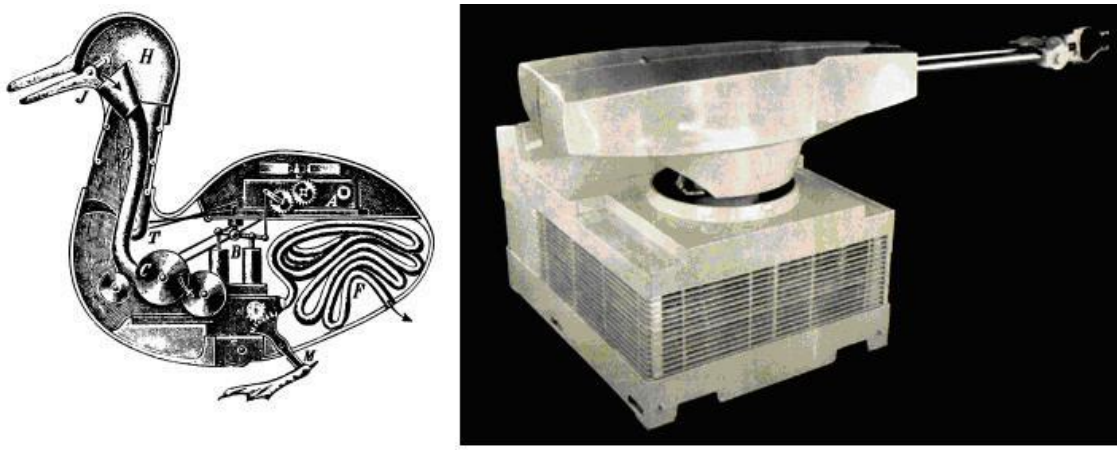
ϕ_{in}	Desired bank angle in the optimization [radian]
$\Phi(t, t_0)$	State-transition matrix
ρ	Water density [kg/m^3]
σ	Standard deviation
τ	Commanded torque to the motors [Nm]
ζ	Deflection angle of the paddle [radian]

CHAPTER 1

Introduction

Robots have been the subject of human fascinations for centuries. The first programmable robot was designed and built in 1206, more than 800 hundred years ago [1]. Leonardo da Vinci tried to build a mechanical knight in the late 15th century but was unsuccessful. Other robots, such as *Digesting Duck* (Figure 1–1a) [2] and the *Karakuri Toys*, were designed over the following centuries but their purpose was purely recreational. Over the past century, robotics has evolved to perform the tasks usually done by humans. The first commercial robot called *Unimate* (Figure 1–1b) and designed by the *Unimation Company* was first installed on a General Motor assembly line in 1961. Its task was to transport die castings and weld them on an auto body, a task dangerous for human due to exhaust gases. Over the years, robot manipulators have mostly replaced humans for simple repetitive tasks.

More recently, there has been a growing interest in mobile robotics. Mobile robots have the advantage of being able to move in their environment to perform their task. We can classify mobile robots in three classes according to the environment in which they operate: land robot, aerial robots, usually known as unmanned aerial vehicles (UAV) and underwater robots also known as autonomous underwater



(a) (b)

Figure 1–1: *(a) Digesting Duck* and (b) *Unimate*

vehicles (AUV). They are used for several applications ranging from cleaning the floor (Figure 1–2a) to bombing missions (Figure 1–2b). Another subclass of mobile robots is gaining more attention: biomimetic robots. These robots embody interesting biological features of living creatures to outperform conventional robots. One advantage of underwater biomimetic robots is that they can be made amphibious. That will enable both land and sea operations using the same robot. This could reduce the deployment time for sea operations since the robot could be deploy from land and walk in the water. With a conventional robot, the robot needs to be brought in the water manually. These robots can be further separated into remotely operated and autonomous ones. The former are controlled by a human pilot while the latter perform their duties autnomously. One of the keys for the success of autonomous robots is accurate control. The controller refers to the subsystem that transforms user inputs into robot actions.

Figure 1–2: (a) Roomba cleaning robot by iRobot and (b) MQ-9 Reaper hunter killer by US military

Over the years, many control techniques have been developed to control mobile robots. Some depend only on the error signals while others are more complex and require knowledge about the dynamics of the system. This knowledge is obtained by constructing a dynamics model for the robot. The model usually takes as input the generalized forces acting on the robot and outputs the motion of the robot. It can be obtained either by curve fitting, or by using physics principles. The accuracy of some controllers depends directly on how accurate the model is. A category of controllers called *adaptive* are designed specifically to deal with model uncertainties. Moreover, dynamics models can be used to simulate the system of interest thus allowing pre-experiment preparation.

In this thesis, we study the modeling and control of underwater biomimetic vehicles. We validate a dynamics model for the Aqua underwater vehicle and develop several controllers that are then tested experimentally. We use a simple control technique, namely PID, as well as more advanced techniques such as model identification

adaptive control (MIAC) and Floquet based controllers. We believe that for complex systems such as underwater biomimetic vehicles, advanced control techniques may offer significant advantages over conventional techniques.

1.1 Aqua Underwater Vehicle

The vehicle or platform used in this research is the Aqua underwater vehicle [3] depicted in Figure 1–3. It is an evolution of several other vehicles but its main inspiration is the RHex hexapod walking robot [4]. RHex is a robot with compliant semi-circular shaped legs, each of which is actuated by only one motor. It is capable of walking over rough terrain that few other robots can negotiate. Aqua retains most of RHex's walking capabilities while also being amphibious due to its waterproof shell. RHex is capable of running at a speed of 2.7 m/s [5], climbing slopes over 40°, bounding [6], flipping [7] and bipedal running using only its two rear legs [8].

Because RHex routinely encountered hazards such as rain or mud it became obvious that its aluminum frame and Lexan cover were not sufficient for outdoor experiments in an uncontrolled environment. This motivated the design and construction of three successive robot designs: Shelley-RHex, Rugged-RHex, and AQUA [3]. Shelley had a waterproof shell made out of carbon fiber, which enabled amphibious operations: walking on land and swimming at the water surface.

Rugged-RHex can walk on land and swim on the water surface using semi-circular legs, or it can swim underwater using oscillating flexible fins. The fins produce the propulsive and control forces. A fiber optic tether was required to transmit the robots video signal to the vehicle operator and the operator's control commands to the robot.

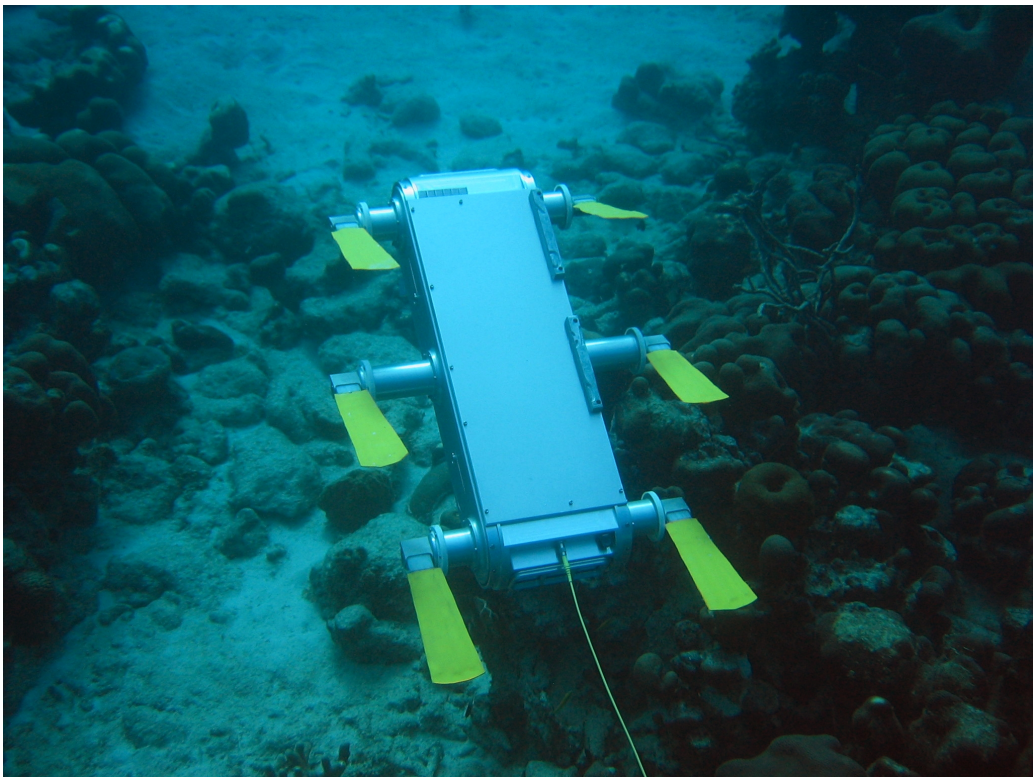


Figure 1–3: Aqua underwater vehicle. Credit: Barbados 2004 team

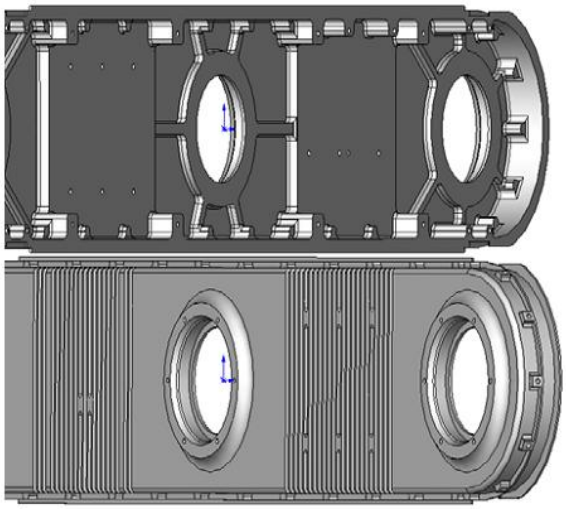


Figure 1–4: Aqua side frame with internal reinforcement

1.1.1 Physical design

Aqua, shown in Figure 1–3 is composed of six hips and one box-shaped body. The dimensions of the body are $66 \times 21 \times 13$ cm with semi-circular ends. The two front and back hips are 8 cm in length while the two middle ones are 13.5 cm in length. The body and hips are made mostly of anodized aluminum and the surfaces that were most at risk of scratching were powder coated [3]. The total mass of the body and hips including batteries is approximately 12.39 kg but this can vary depending on which batteries are used. Aqua was designed to withstand pressures up to 36.6 m depth, thanks to the reinforced side structure shown in Figure 1–4. It has been tested in water up to a depth of 39.6 m without failure. Aqua also has windows at its front and rear, allowing the use of cameras.

1.1.2 Computer and Communication

The Aqua underwater vehicle is equipped with two PC/104 form-factor single-board computers. The first computer, that runs the basic robot code, is a 500 MHz *Cool RoadRunner-LX800* manufactured by LiPPERT Embedded Computers GmbH that uses the *neutrino* real-time operating system from QNX. The second computer card is an ADL-855 from Advanced Digital Logic with a 1400 MHz Pentium-M processor that uses the Linux operating system [9]. Details about the software architecture can be found in [10].

A tether is required to transmit visual data because underwater wireless communication is limited to low bandwidths (e.g. 19200 baud for ultrasonic modem). Since all power is provided by the two onboard batteries, the tether does not carry any power. A single 3 mm optical fibre is used because of its light weight and high transmission capability. The longest fibre optic that we have available for our experiments is 200 m in length.

1.1.3 Sensors

The relative leg position is measured using optical incremental rotary encoders attached to each leg motor shaft. A MSI-P400 quadrature decoder card manufactured by Microcomputer Systems Inc. translates these two quadrature output signals from each encoder into numerical values of angle and rate. The absolute position of each leg is determined by calibration at the beginning of an experiment.

Although there is no sensor to measure the leg motor electrical current, that current is estimated using an accurate motor model [11]. This model computes the current based on the leg angular velocity and the voltage supplied to the motor.

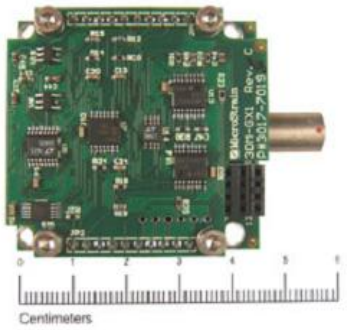
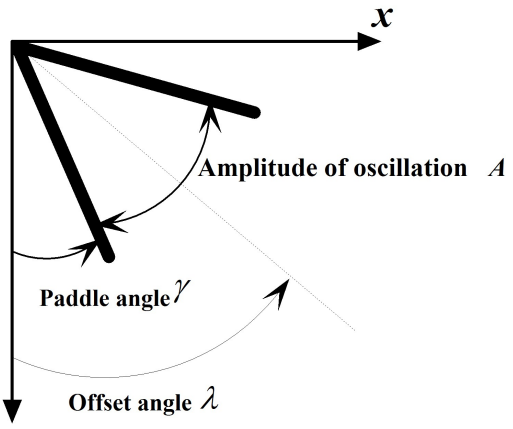


Figure 1–5: IMU 3DM-GX1 from Microstrain

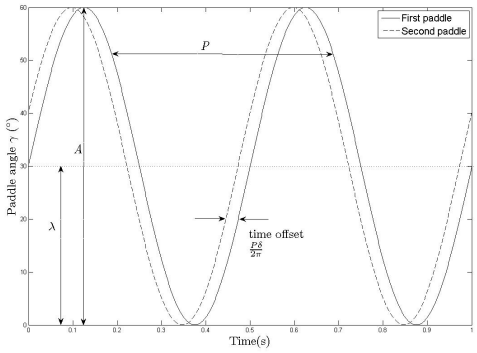
Using the physical parameters of the motor, it is then possible to estimate the motor torque.

A 3-axis inertial measurement unit (IMU) manufactured by Microstrain is used to provide kinematic measurements. It includes three Micro-Electro-Mechanical Systems(MEMS) acceleration sensors,three MEMS rate gyroscopes and three magnetometers. They are oriented orthogonally to provide sensing in all three directions (x,y and z). The accelerometers give the linear acceleration of the vehicle and the rate gyroscopes give the angular velocities. The magnometers are not used because of electromagnetic interference of the leg motors. Therefore, Euler angles are estimated from the rate gyros using filtering techniques [9].

A 1024x768 IIDC-compliant Dragon Fly Firewire camera from PointGrey is installed at the front of the vehicle. It is equipped with a fish-eye lens that compensates for the reduced field-of-view due to the refraction index difference between air and water [9]. In this work, this camera is used to help navigate the vehicle. It can also be used for visual-servoing or to record video footage.



(a) Amplitude A, offset λ and paddle position γ for a generic oscillation



(b) Paddle parameters for two different paddles. $\lambda = 30°$, $P = 30s$, $A = 120°$ and $\delta = 20°$ or 0.028s

Figure 1–6: Parameters pertaining to eq. (1.1)

1.1.4 Propulsion

The use of flapping foils or paddles to produce thrust for underwater application is becoming more popular. Several reasons such as high impulse force and possibly higher efficiency motivates this interest. As will be discussed in Section 1.2.2 many systems employ flapping paddles to propel an underwater vehicle [12–16]. The basic principle is simple: the oscillation of the paddle about an axis creates a thrust along the axis, and this will be analyzed in details in Section 2.

The propulsion forces on Aqua are generated by 6 oscillating paddles. For a symmetric paddle motion, the net thrust produced by the paddle is directed along the average paddle position. The motion of the paddles follows a sinusoidal trajectory:

$$\gamma = \frac{A}{2} \sin\left(\frac{2\pi}{P}t + \delta\right) + \lambda \tag{1.1}$$

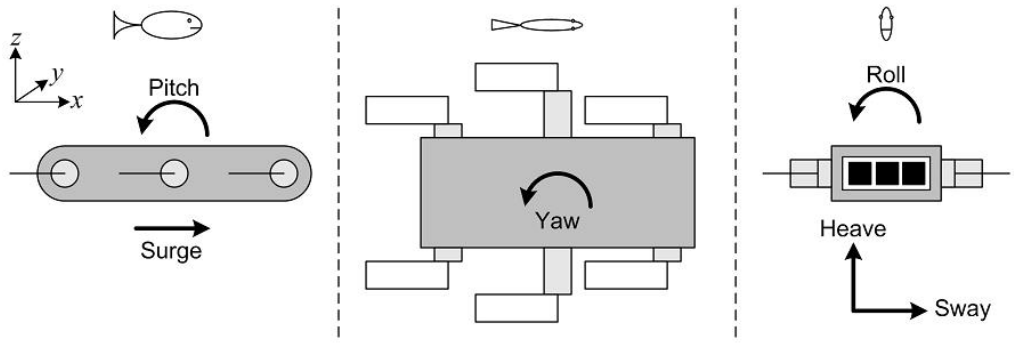


Figure 1–7: The six degrees of freedom of Aqua

where γ is the paddle angle, A is the amplitude of oscillation, P is the period of oscillation, λ is the offset of oscillation and δ is the phase offset between the oscillation of the paddles. Figure 1–6 shows these quantities graphically. Figure 1–6(a) shows the amplitude of oscillation A, offset of oscillation λ and paddle angle γ for a generic oscillation. Figure 1–6(b) shows the paddle angle γ for two paddles with phase offset $\delta = 20°$, period $P = 0.5s$, amplitude $A = 120°$ and offset angle $\lambda = 30°$. Usually, a swimming gait corresponds to a particular combination of constant phase offsets. The *middle-off* gait is the most commonly used on Aqua. The phase offset is zero for all four corner paddles while the offset is 180° for the two middle paddles. This gait has the advantage of exhibiting lower parasitic oscillations of the vehicle body, due the periodic paddle thrust produced.

The period of oscillation represents the time to complete one full oscillation. It is usually sets to a value between 0.2 and 0.6 s. The amplitude of oscillation A is the total angle swept by the paddle during one oscillation. It is the main parameter used to alter the thrust produced by each paddle. On the vehicle GUI, increasing the speed of the vehicle implies increasing A. The offset of oscillation λ determines the

direction along which the thrust is produced. With the current paddle configuration, it is possible to change the offset angle of the paddles to produce a net force in five of the six degrees of freedom: roll, pitch, yaw, heave and surge. Figure 1–7 shows a description of each degree of freedom. A pitching motion is obtained by using an offset angle for the front and rear flippers in opposite directions. For a rolling moment, the offset angle of the right and left paddles are set in opposite directions. The yawing motion is obtained by increasing or decreasing the thrust on one side of the vehicle. Changing the thrust can be done by either changing the period or amplitude of oscillation of the paddles. Finally, heave is produced by changing the offset angle of all paddles by the same amount.

1.1.5 Experimental Setup

Figure 1–8 shows the general setup 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 setup used during an experiment [17]. The vehicle is connected to the Operator Control Unit(OCU) through a optical fiber. The OCU is then connected to a laptop with a serial cable. There is a monitor on top of the OCU that displays the images captured by the camera onboard Aqua. All the information related to the state of the vehicle is transmitted to the laptop.

The pilot sees the actual and desired state of the vehicle on the GUI shown in Figure 1–9(a). The pilot controls the vehicle using a gamepad as shown in Figure 1–9(b). The commands given by the human pilot are also displayed on the GUI. The experimental setup will be revisited in Section 4.4.

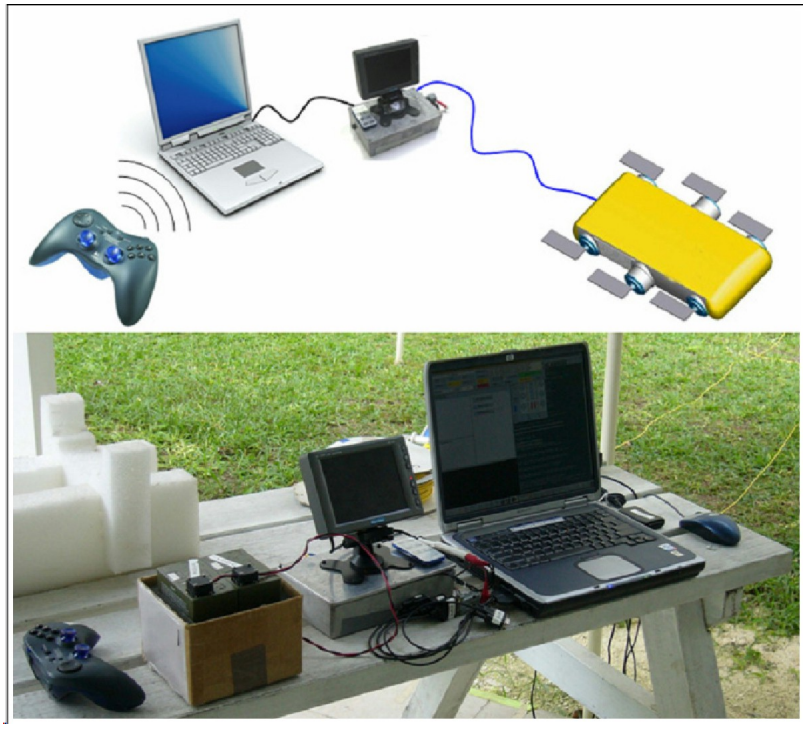
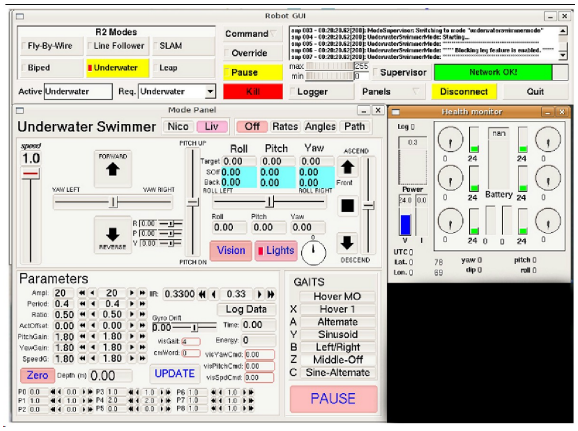


Figure 1–8: General experimental setup with Aqua. Credit: Olivia Chiu



(a) Graphical User Intergace(GUI) used on Aqua.

(b) Gamepad used on Aqua.

Figure 1–9: GUI and gamepad used on Aqua. Credit: Olivia Chiu

1.1.6 Aqua Applications

The Aqua underwater vehicle has already used successfully for a multitude of underwater applications including person tracking, human-robot interaction, system organization, modeling and image recovery. An algorithm to track human divers was developed and implemented on Aqua by Sattar et al. [18, 19]. Their algorithm detected the periodic motion of the legs of a human diver. This allows Aqua to assist divers in their work by following them autonomously. In terms of system organization, Dudek et al. presented behaviors and interaction modes for the Aqua underwater vehicle [20]. With their work, the vehicle can now be controlled via visual servoing using chromatic targets. They use markers provided by the ARTag toolkit to communicate with the vehicle and program it during operation. Furthermore, they developed an hovering gait for the vehicle. An empirical model of the vehicle was also developed by Giguere et al. [21]. They characterized the rotational response of the vehicle to relate the paddle motion to the rotational velocity of Aqua. This model is accurate but it is not based on physics principles and excludes the linear degrees of freedom. Finally, Aqua was used to collect underwater images and develop an algorithm that corrects the colors in the images [22]. With this algorithm, the images collected underwater were closer to those that would be collected under full spectrum illumination.

1.2 Literature Review

This section reviews the work done by other researchers on subjects pertaining to this thesis. It is partitioned into the six major subjects that are discussed in this thesis: flapping foil propulsion, Vehicle modeling and design, control of conventional

underwater vehicle, control of biomimetic underwater vehicle, time-periodic systems and path optimization.

1.2.1 Flapping Foil Propulsion

Many scientists have studied flapping foils as a mean to produce thrust. One of the key advantage of biomimetic propulsion is better maneuvrability and possibly better efficiency. Triantafyllou et al. produced a review of the research on flapping foils [12]. Mittal et al. developped a numerical simulation to examine the performance of a flapping foil [23] that was to be used on a biomimetic autnomous underwater vehicle (BAUV). Schouveiler et al. studied the performance of a two degrees of freedom flapping foil. [24]. The fins moved in the heave and pitch direction with the Reynold number held constant for each experiment. He also investigated asymmetric paddle motion and found this to be a good method to produce side forces. Licht et al. discussed the design and construction of an oscillating foil [15], and estimated the performance of the foil. According to their evaluation, a robot equipped with these foils could achieve a speed of 1 m/s. Harper et al. presented a model in which springs are used to transmit forces to an oscillating foil [25]. They presented the model as a set of ordinary differential equations, making it attractive for control design. The springs are used to improve the energy efficiency of the system. Guglielmini et al. described the vortex structure of a two degrees of freedom foil [26]. They used vortex theory to quantify the thrust produced by the paddle. Although this method will probably give the most accurate results, it is not suitable for incorporation into a controller because it too computationally intensive. Yamamoto et al. performed a feasibility study on the use of oscillating fins to produce thrust

on a surface ship [27]. They studied the efficiency and power output of paddles with different shapes and found that a rectangular paddle with 62.5% of its length flexible gives the best efficiency. However, they also found that a paddle with 62.5% of its length rigid gave the most propulsive power. Mollendorf et al. studied the deflection of thin elastic beam subjected to fluid mechanical forces [28]. They compared their deflection results with measurements taken from underwater video of fins worn by a diver. Their model gave a good approximation of the deflection of a flexible fin, but it is computationally demanding. Lu et al. studied the performance and vortex shedding of an oscillating foil [29]. They studied factors such as frequency and amplitude of the oscillation. Schnipper et al. presented an experimental study of the vortex wakes produced by an oscillating foil [30]. They found that it could have up to 16 vortices per oscillation.

1.2.2 Biomimetic Vehicle Modeling and Design

There are still few underwater vehicles that use biomimetic propulsion system although this number is increasing. Most of them are still at the design and construction phase. Yu et al. developed a four-joint robotic fish and Triantafyllou et al. designed an eight-joint robotic fish [13, 14]. Deng et al. developed a microrobot that uses single oscillating tail fin to produce thrust [31]. The control force was provided by two independent side fins. Guo et al. also developed a microrobot but used two tail fins to provide propulsive force and included a buoyancy adjuster [32]. Kemp et al. built a vehicle named Madeleine that uses flexible four fins to generate both propulsive and control forces [16]. Madeleine has the same general structure as Aqua and its main physical difference is the number of paddles. Long et al. have found

that the front paddles significantly reduced the efficiency of the rear paddles [33]. As a result, the Madeleine achieves the same maximum speed with either 2 or 4 paddles. However, swimming with only 2 paddles is more energy efficient. Licht et al. designed and built a robot that mimics the locomotion of a turtle [34]. The vehicle is trimmed unstable in pitch and yaw to make it more maneuverable. They investigated different paddle motions and different turning strategies and found that a smaller turning radius could be achieved using a banked turn rather than a level yawing turn.

1.2.3 Control of Conventional Underwater Vehicles

Much of research has been done on the control of conventional underwater robots. Yoerger et al. and Xu et al. used sliding mode theory to develop robust trajectory tracking controllers for an underwater robot [35,36]. These controllers had the advantage of dealing directly with nonlinearities and being robust to an imprecise model. However, they also showed that the performance of the controllers is greatly improved by a more accurate model. Smallwood et al. compared the ability of several controllers to track a prescribed trajectory [37]. Their controllers were tested on a conventional underwater robot as well as in simulations. They found that the model-based controllers were capable of providing exact trajectory tracking. The PD controller was able to provide velocity tracking but failed to track the position accurately, though the position error remained bounded. Furthermore, they found that increasing the PD gains improved the tracking performance.

Feng et al. designed a robust controller based on H_∞ theory [38]. They used two separate controllers for longitudinal and lateral degrees of freedom. The controllers

were tested on a nonlinear simulation of the vehicle and the results showed that the desired behavior could be obtained. Yuh described two controllers, nonlinear and adaptive, used for the control of an underwater robot [39]. He concluded that the nonlinear controllers provide good performance if the model parameters are well known whereas the adaptive controller does not require good knowledge of the model. Caccia et al. discussed the design, implementation and testing of a general purpose control system for unmanned underwater vehicle [40]. They used a Lyapunov based guidance system and a PI controller implemented in a two-layered hierarchical architecture for closed loop control. Their approach was tested experimentally and demonstrated good tracking results. Kwon et al. and Lee at al. designed autopilots for an underwater vehicle using fuzzy logic [41, 42]. One of the main features of their controller was a collision avoidance system. Spangelo et al. also studied collision avoidance, but using optimal control [43].

Yuh et al. and Zhao et al. developed a self-adjusting controller for an underwater robot [44, 45]. They presented the theory for an adaptive *plus* disturbance obserber (ADOB) and tested it experimentally. They obtained good tracking results but could not get zero error. Ritonja et al. developed a simple adaptive controller for stability augmentation [46]. Deng at al. designed a model output following control system using a command generator tracker (CGT) for a system with time delays [47]. The controller was tested in a dynamics simulation and they obtained good results.

Learning algorithms can also be used to control underwater vehicles. For example, Coates et al. developed a learning control system for an helicopter [48, 49]. They were able to perform extreme acrobatic maneuvers with their control system.

They also found that their controller performed better than their expert pilot. Also, Jang et al. designed fuzzy logic controllers for a mobile robot [50]. The controller could handle bounded disturbances and unstructured unmodelled dynamics. The online tuning parameter algorithm required no off-line learning but guaranteed small tracking errors. Although, these learning techniques have not been implemented on underwater applications, they could likely be adapted to provide good performance for underwater vehicle.

1.2.4 Control of Biomimetic Underwater Vehicles

Some researchers have considered the guidance and control of biomimetic vehicles. Guo et al. developed a waypoint tracking controller, for a vehicle with oscillating tail fin propulsion [51], based on hierarchical local and global controllers so as to mimic fish behavior. The performance of the controller was evaluated in simulation with good results. They also discussed the effect of model uncertainties and disturbances on the control performance. Geder et al. developed a fuzzy logic PID controller to control the trajectory of a vehicle with two pectoral fins [52]. Naik et al. studied the motion control of a fish-like robot in the yaw plane [53]. The motion of the vehicle was controlled by altering the motion of pectoral-like fins. They used an adaptive control law and obtained good tracking results. Singh et al. studied the motion control of a fish-like robot in the diving plane [54]. The control force was obtained by altering the offset angle of the pectoral-like fins. Yu et al. studied control of a fish-like robot using fuzzy logic and point-to-point control algorithms [55, 56]. They found those algorithms to be effective and reliable when tested for different maneuvers and scenarios. Dong et al. designed a networked controller

for a biomimetic robot fish [57]. Their robot had no onboard sensors and all the information was provided by an external camera. However, the robot was not able to track the target accurately.

1.2.5 Time-periodic systems

Aqua has the particularity of using oscillating fins to produce its thrust. A direct consequence of this propulsion system is that the thrust is not constant but time-periodic. As a result, we can treat Aqua as a time-periodic system. Floquet control theory pertains to the control of time-periodic systems. Many researchers have studied the theory of time-periodic systems and some have developed methods to use Floquet theory to design controllers for these systems. Calico et al. developed a method based on Floquet theory allowing a determination of the location of the poles of the system [58]. Montagnier et al. discussed different approaches to find the Floquet factors [59], putting an emphasis on finding real Floquet factors. Montagnier et al. also studied various techniques to develop controllers using Floquet theory [60]. However, his research was theoretical and did not find applications. Acho et al. used a H_∞ controller to design tracking controllers for a time-periodic system [61]. He obtained good results but his technique did not deal with the time-periodicity of the system directly. Brockett discussed the linear theory pertaining to the approach to obtain the Floquet factors [62]. Finally Cai et al. developed an efficient method to compute the state-transition matrix [63]. This matrix is important in the design of Floquet controllers.

1.2.6 Path Optimization

Other researchers have studied path optimization for underwater vehicles. Petres et al. studied a path planning algorithm for an underwater robot using a fast marching algorithm [64, 65]. They used the water current and vehicle turn radius as variables. Kanakakis et al. discussed path planning and navigation for autonomous underwater vehicles [66]. The optimal path was found using a genetic algorithm and controllers based on fuzzy logic. The optimizer took into consideration the physical limitations of the vehicle and its maneuvring characteristics. Khanmohammadi et al. studied the path optimization in a horizontal plane for a thruster-based underwater vehicle [67]. An energy performance index was minimized. A conjugate gradient method, a genetic algorithm (GA) and particle swarm optimization (PSO) methods were applied to solve the problem. They found that the GA and PSO gave better solutions. Shamir studied the overtaking of a slower vehicle [68]. Karush-Kuhn-Tucker and Lagrange methods were considered but were found to be extremely difficult to use for this problem. Instead, the optimization software Lingo was used. Guo et al. studied path planning optimization for underwater robots using particle swarm optimization (PSO) [69]. They concluded that PSO was a good technique for that application. Lambert et al. studied the optimization of a U-turn maneuver for a towed underwater vehicle [70]. The algorithm used to perform the optimization was Powell's Conjugate Direction method. Kim at al. studied the optimization of a robot fish velocity using a genetic algorithm [71]. The objective was to find the best motion of the vehicle to maximize the speed.

1.3 Claim of Originality

A summary of the main contributions of this thesis is as follows:

- Development of advanced trajectory tracking controllers for a biomimetic underwater vehicle including a model-based controller, an adaptive controller and a controller based on Floquet theory.
- Experimental study of the performance of the trajectory tracking controllers for a biomimetic vehicle.
- Optimization of a U-turn maneuver to improve the performance of coral reef inspection.
- Development of a flexible paddle model that computes the thrust produced by an oscillating paddle. This was followed by the development of a reverse paddle model that determines the paddle motion to produce a desired thrust.
- Experimental validation of the dynamics model of the Aqua biomimetic underwater vehicle.
- Linearization of the time periodic nonlinear vehicle model using a novel technique of average response to perturbations.

1.4 Thesis Motivations and Organization

The overall objective of this thesis is to develop controllers for biomimetic autonomous underwater vehicles (BAUV). There has been little research done on the control of flapping foil vehicle and this research aims to establish a base for the control of these vehicles. We developed four classes of controllers to provide trajectory tracking capabilities for the Aqua underwater vehicle. The results of this thesis will probide a better knowledge of which control techniques to use for a BAUV.

In Chapter 2, we describe the development of a dynamics model for the Aqua underwater vehicle. We first discuss a body model that predicts the motion of the vehicle based on its physical parameters and the forces and moments applied to it. This is followed by the development of a forward paddle model that predicts the thrust produced by the paddles based on the paddle motion. These two models are implemented into a computer simulation programmed in MATLAB Simulink. These models are also used in the controller development. They are validated experimentally and their accuracy is proven. Finally, the vehicle dynamics model is linearized to for application with controllers that require a linear model.

Chapter 3 covers the theory behind the different controllers developed in this thesis. We begin by defining the input to the controllers and then describe each class of controllers. The first class is PID and this is the only class of controllers studied that does not require a dynamics model. The second class is model-based controllers in which the force is based on the dynamics model of the vehicle. The third class is based on adaptive control: controllers that update themselves based on the model and the error signals. The fourth and final class is Floquet based controllers that use the theory of time-periodic systems to deal explicitly with the oscillating thrust produced by the paddles.

In Chapter 4, the controllers are tested in the dynamics simulation and experimentally. We start by discussing how the theory presented in Chapter 3 was applied to the Aqua underwater vehicle. We then introduce the trajectories that were tested

in the simulation and in the experiment. The simulation results are shown first because the simulation was used to tune the control gains. Finally, we present the results of the experiment conducted in open water.

An optimization of a high performance maneuver, a U-turn, is discussed in Chapter 5. The objectives of the optimization are discussed followed by a description of the design variables used in the optimization. We then describe the controller that was used during the optimization. This is followed by a discussion of the performance metric and of the optimization constraints. The optimization technique, a genetic algorithm, used to solve our problem is also described in this chapter. Finally, the optimization results are presented and analyzed.

In chapter 6, the conclusions drawn in the thesis are revisited and recommendations for future work are presented.

CHAPTER 2

Modeling

Before developing controllers for our underwater vehicle, it is necessary to first study the dynamics of the system. An accurate dynamics model will give information about the natural stability of the system and it could be used as a testbed for controllers. As was mentioned in Section 1.2, many researchers have studied the dynamics of underwater vehicles [13, 15, 16, 32, 72–74] and of oscillating fins [12, 14, 15, 25–27, 31, 33, 73, 74]. These models can provide useful insight but they are not directly applicable to our vehicle. Another issue is that most authors use empirical formulas to predict thrust generated by fins and it is impossible to adapt their model for other fins.

This chapter discusses the modeling of the Aqua underwater vehicle. The model can be separated into two parts: the body model and the forward paddle model. The body model determines the external forces applied to the body of the vehicle and calculates the resulting velocities and positions. The forward paddle model uses the paddle motion to determine the forces and moments produced by the oscillating paddles. The body model is based on the approach proposed by Fossen in which the hydrodynamics derivative are computed using empirical values for a bluff rectangular

body [72], and adapt this to our vehicle, as described in Section 2.1. The paddle model is based on the approach by Georgiades in which the lift and drag forces produced by the oscillating paddle are computed at every instant and transformed into thrust and torque [74]. A limitation of that model was that it was assumed that the paddle was fully rigid when in reality, a part of the paddle is flexible. The forward paddle model described in Section 2.2 takes into account the deformation of the paddle as it oscillates through the water. The forward paddle model was then used to create an inverse paddle model that takes the desired thrust as input and ouputs the required paddle motion, as discussed in Section 2.5. The models were implemented in MATLAB Simulink to create a dynamics simulation (Section 2.3) that was then validated in Section 2.4. Finally, in in Section 2.6, the nonlinear model was linearized to facilitate the design and analysis of controllers.

The models developed in this chapter can be used in the design of model-based controllers for the Aqua underwater vehicle. Moreover, the dynamics simulation will later be used to tune the controllers' gains and as a testbed for the controllers.

2.1 Body Model

As was mentioned previously, the dynamics model of the whole vehicle is composed of the paddle model and of the body model. In this section, we describe the body model for the Aqua underwater vehicle, which allows us is to compute the position and velocity of the vehicle from the forces exerted on it. The body model is based on the work of Georgiades et al. [73, 74] and Fossen [72].

The robot has six degrees of freedom, and we consider two relevant reference frames of interest. The first one, R_V, is the robot frame and has its origin at the

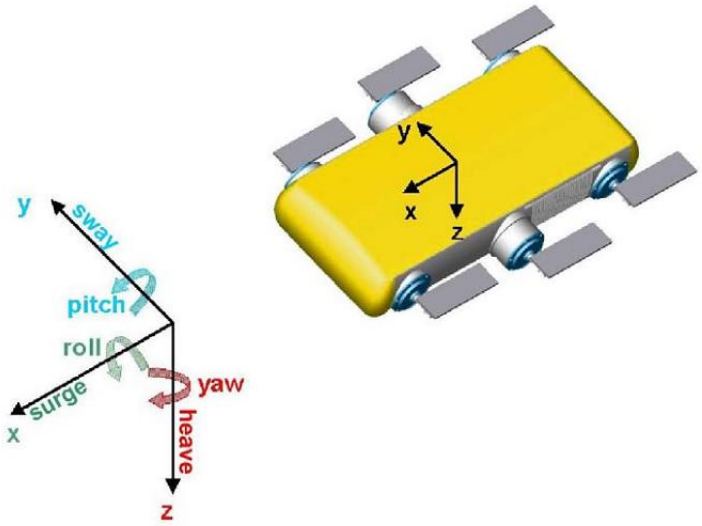


Figure 2–1: Six degrees of freedom of the vehicle. Credit: Christine Georgiades

centre of mass of the robot. As shown in Figure 2–1, the x-axis is pointing toward the front of the vehicle, the z-axis toward the center of the Earth and the y-axis follows the right-hand rule convention. The second one, R_I, is the inertial coordinate frame and has its origin at a fixed arbitrary point on the water surface. Euler angles ($\phi,\ \theta,\ \psi$) are the angles between the R_I and R_V coordinate frames, where ϕ is the roll angle, θ the pitch angle and ψ the yaw angle [72]. The motion of the robot in the 6 degrees of freedom can then be described by the following vectors and matrices:

$$\mathbf{n}_1 = \begin{bmatrix} X & Y & Z \end{bmatrix}^T \qquad \mathbf{n}_2 = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^T$$
$$\mathbf{s} = \begin{bmatrix} X & Y & Z & \phi & \theta & \psi \end{bmatrix}^T \tag{2.1}$$

$$\mathbf{v}_1 = \begin{bmatrix} u & v & w \end{bmatrix}^T \qquad \mathbf{v}_2 = \begin{bmatrix} p & q & r \end{bmatrix}^T$$
$$\mathbf{v} = \begin{bmatrix} u & v & w & p & q & r \end{bmatrix}^T \tag{2.2}$$

The position defined in (2.1) is expressed using components in R_I while the velocity of the vehicle defined in (2.2) is expressed as components in R_V. They are related by a transformation matrix:

$$\dot{\mathbf{s}} = \mathbf{J}(\mathbf{n}_2)\mathbf{v} = \begin{bmatrix} \mathbf{J}_1(\mathbf{n}_2) & 0 \\ 0 & \mathbf{J}_2(\mathbf{n}_2) \end{bmatrix} \mathbf{v} \tag{2.3}$$

$$\mathbf{J}_1(\mathbf{n}_2) = \begin{bmatrix} \cos(\psi)\cos(\theta) & -\sin(\psi)\cos(\phi)+\cos(\psi)\sin(\theta)\sin(\phi) & \sin(\psi)\sin(\phi)+\cos(\phi)\cos(\psi)\sin(\theta) \\ \sin(\psi)\cos(\theta) & -\cos(\psi)\cos(\phi)+\sin(\psi)\sin(\theta)\sin(\phi) & -\cos\psi)\sin(\phi)+\cos(\phi)\sin(\psi)\sin(\theta) \\ -\sin(\theta) & \cos(\theta)\sin(\phi) & \cos(\theta)\cos(\phi) \end{bmatrix} \tag{2.4}$$

$$\mathbf{J}_2(\mathbf{n}_2) = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \frac{\sin(\phi)}{\cos(\theta)} & \frac{\cos(\phi)}{\cos(\theta)} \end{bmatrix} \tag{2.5}$$

The dynamics model takes into account the Coriolis forces, the hydrodynamic forces and the inertia:

$$\mathbf{M}\dot{\mathbf{v}} + \mathbf{C}(\mathbf{v})\mathbf{v} + \mathbf{D}(\mathbf{v})\mathbf{v} + \mathbf{g}(\mathbf{n}_2) + \mathbf{b}(\mathbf{n}_2) = \mathbf{f} \tag{2.6}$$

where $\mathbf{f} = \begin{bmatrix} F_x & F_y & F_z & M_x & M_y & M_z \end{bmatrix}^T$ is the vector of forces and moments produced by the paddles in the six degrees of freedom, $\mathbf{M}$ is the 6 x 6 mass matrix including added mass, $\mathbf{C}(\mathbf{v})$ is the 6 x 6 Coriolis matrix, $\mathbf{D}(\mathbf{v})$ is the 6 x 6 hydrodynamic matrix, $\mathbf{g}$ is the gravitational force vector, $\mathbf{b}$ is the buoyancy force vector and $\mathbf{v}$ and $\mathbf{n}_2$ are defined in (2.2). In the simulation, it is assumed that the centre of gravity is coincident with the centre of buoyancy. As a result, the buoyancy and gravity forces cancel each other. In practice, they are never exactly coincident because the mass distribution changes depending on which batteries, set of paddles and other pieces of equipment are installed. Since the robot is immersed in water, the Coriolis and mass matrices include a rigid body and an added mass component. The rigid body part can be understood as the mass of the robot in a vacuum, while the added mass part models added inertia due to the motion of the robot through the fluid. According to Fossen [72], assuming that there are three planes of symmetry and that the vehicle is moving at low speed, the mass matrix, including the rigid-body and added mass, is diagonal. The hydrodynamic matrix is also a diagonal matrix. However, the Coriolis matrix $\mathbf{C}(\mathbf{v})$ has off-diagonal terms and is responsible for the coupling between the 6 degrees of freedom. Moreover, the Coriolis and hydrodynamic matrices contain the velocity vector. As a result, these two terms are responsible for the nonlinearity of the system. The parameters in the different matrices were obtained using empirical results for a solid rectangular prism [72].

$$\mathbf{M} = \begin{bmatrix} m - X_{\dot{u}} & 0 & 0 & 0 & 0 & 0 \\ 0 & m - Y_{\dot{v}} & 0 & 0 & 0 & 0 \\ 0 & 0 & m - Z_{\dot{w}} & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{xx} - L_{\dot{p}} & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{yy} - M_{\dot{q}} & 0 \\ 0 & 0 & 0 & 0 & 0 & I_{zz} - N_{\dot{r}} \end{bmatrix} \tag{2.7}$$

$$\mathbf{D(v)} = \begin{bmatrix} -X_{u^2} |u| & 0 & 0 & 0 & 0 & 0 \\ 0 & -Y_{v^2} |v| & 0 & 0 & 0 & 0 \\ 0 & 0 & -Z_{w^2} |w| & 0 & 0 & 0 \\ 0 & 0 & 0 & -L_{p^2} |p| & 0 & 0 \\ 0 & 0 & 0 & 0 & -M_{q^2} |q| & 0 \\ 0 & 0 & 0 & 0 & 0 & -N_{r^2} |r| \end{bmatrix} \tag{2.8}$$

$$\mathbf{C(v)} = \begin{bmatrix} 0 & 0 & 0 & 0 & (m-Z_{\dot{w}})w & -(m-Y_{\dot{v}})v \\ 0 & 0 & 0 & -(m-Z_{\dot{w}})w & 0 & (m-X_{\dot{u}})u \\ 0 & 0 & 0 & (m-Y_{\dot{v}})v & -(m-X_{\dot{u}})u & 0 \\ 0 & (m-Z_{\dot{w}})w & -(m-Y_{\dot{v}})v & 0 & (I_{zz}-N_{\dot{r}})r & -(I_{yy}-M_{\dot{q}})q \\ -(m-Z_{\dot{w}})w & 0 & (m-X_{\dot{u}}) & -(I_{zz}-N_{\dot{r}})r & 0 & (I_{xx}-L_{\dot{p}})p \\ (m-Y_{\dot{v}})v & -(m-X_{\dot{u}})u & (I_{yy}-M_{\dot{q}})q & 0 & -(I_{xx}-L_{\dot{p}})p & 0 \end{bmatrix} \tag{2.9}$$

where m is the mass of the vehicle, I_{xx}, I_{yy} and I_{zz} are the moments of inertia of the vehicle about the x, y and z axes. $X_{\dot{u}}$, $Y_{\dot{v}}$, $Z_{\dot{w}}$, $L_{\dot{p}}$, $M_{\dot{q}}$ and $N_{\dot{r}}$ are hydrodynamics

derivatives. The moments of inertia and the mass of the vehicle were evaluated using CAD drawing of the vehicle. The hydrodynamics derivatives were obtained using strip theory for a rectangular box based on experimental data for submerged body [72, 73]. The form of (2.8) implies that the velocity in one degree of freedom creates hydrodynamic force only in that degree of freedom. This was used for simplicity to facilitate the calculation of the hydrodynamic forces by Georgiades [73]. In reality, the hydrodynamic forces are determined by the magnitude of the total velocity vector in three degrees of freedom.

Using (2.6), with the motion of the robot known, we only need an expression for $\mathbf{f}$, the forces and moments produced by the paddles in order to determine the acceleration of the robot, by solving (2.6) for $\dot{\mathbf{v}}$ to obtain $\dot{\mathbf{v}} =$
$\mathbf{M}^{-1}\left[\mathbf{f} - \mathbf{C}(\mathbf{v})\mathbf{v} - \mathbf{D}(\mathbf{v})\mathbf{v} - \mathbf{g}(\mathbf{n}_2) - \mathbf{b}(\mathbf{n}_2)\right]$.

2.2 Forward Paddle Model

As was mentioned in Section 1.2.1, one of the key advantage of flexible paddles over rigid ones is the increase in efficiency. This section describes the forward paddle model used in the dynamics simulation to predict the force produced by an oscillating flexible paddle. It computes the thrust based on a known paddle motion defined by the amplitude, period and oscillation offset. The forward model for our flexible paddle is based on work done by Georgiades et al. on the modeling of a rigid non-tapered oscillating paddle [73, 74]. However, the original model was unusable for the flexible tapered fins used on Aqua. In our model, we consider a tapered paddle of length l with the narrower and wider widths being w_1 and w_2 respectively as shown in Figure 2–2. The axis of rotation is along the hip, at the narrower end of the paddle.

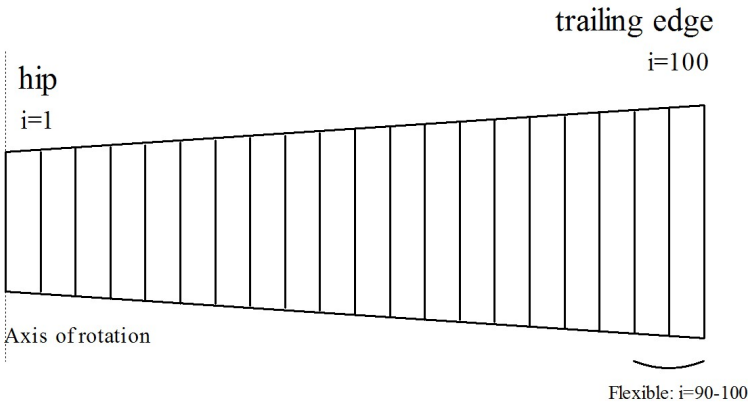


Figure 2–2: Top view of the flexible paddle, showing its discretization

The paddle is composed of two parts, each with different mechanical properties. The first extends from the hip up to 90% of the length and can be considered rigid. The remaining 10% is flexible and deflects as it oscillates through water. To account for the rigid and flexible parts, and to maintain some generality, the paddle was separated into 100 elements of equal length, the first 90 are rigid and the remaining 10 being flexible.

Figure 2–3 shows the forces, the flow velocity and the relevant angles on the paddle. Figure 2–3a shows the three forces that bend the paddle and Figure 2–3b shows the forces that will produce the net thrust. The thrust line is the line about which the paddle oscillates and it determines in which direction the net thrust will be produced. The paddle was modeled as a cantilevered beam attached at the hip joint. From Figure 2–3b, we see that the flow velocity U is made of two components. The normal velocity is due to the rotation of the paddle. The inflow velocity, which is due partly to the water entrained by the paddle motion, and partly due to the vehicle motion, is discussed in detail in Sections 2.2.1 and 2.5.3. As shown in Figure

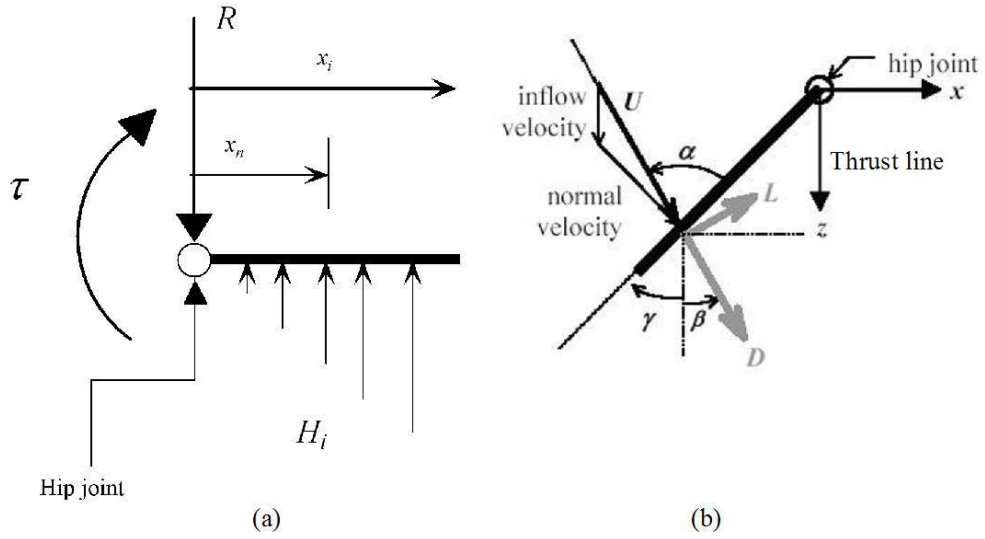


Figure 2–3: Forces, flow velocity and angle on the paddle

2–3a, three main forces act on the paddle: the motor torque (τ), the distributed hydrodynamic force perpendicular to the paddle (H_i) and the reaction force R acting at the hip joint. The hydrodynamic force is a combination of lift and drag forces. From Figure 2–3b, H_i is:

$$H_i = L_i \cos(\alpha_i) + D_i \sin(\alpha_i) \tag{2.10}$$

where L_i is the lift force, D_i the drag force and α_i the angle of attack shown on Figure 2–3b. The subscript i represents the i'th element. L_i and D_i are obtained as discussed below. Then by assuming that the paddle inertia is negligible, we can write the bending moment equation:

$$M_i = -\tau + \sum_{j=1}^{100} H_j x_i + \sum_{n=1}^{i} H_n(x_i - x_n) \qquad x_i \geq x_n \tag{2.11}$$

where x_i is the position of the i'th segment and M_i is the bending moment at the i'th segment. x_i and x_n can be understood better using Figure 2–3a: x_i is the position of the segment where we are calculating the bending moment and x_n is the position of the force that is creating this moment. We know from visual observation that the deflection angle of the paddle remains small during operation. Therefore, we can use elastic bending theory to relate bending moment to paddle deflection:

$$M_i = EI \left(\frac{\partial^2 y}{\partial x^2} \right)_i \tag{2.12}$$

where E is the modulus of elasticity, I is mass moment of inertia of the paddle at a point x along it, while y is the deflection at that point. Equation (2.12) is integrated to obtain the first derivative of y with respect to x. We obtain the boundary conditions by considering the junction between the rigid and flexible part, at which point there is no deflection and the slope is zero. We can thus find the slope of the paddle at any point as:

$$\begin{gathered} \zeta_i = \tan^{-1}(\tfrac{\partial x}{\partial y})_i = \\ \tan^{-1}(\tfrac{1}{2EI}[(2\tau x_i - R x_i^2) + \textstyle\sum_{n=1}^{i} H_n(x_n - x_i)^2 \\ -2\tau x_{90} + R x_{90}^2 - \textstyle\sum_{n=1}^{90} H_n(x_n - x_{90})^2]) \end{gathered} \tag{2.13}$$

where ζ_i is the local deflection angle and x_{90} is the x-location of the juncture of the rigid and flexible parts of the paddle. In the rigid paddle model developed by Georgiades et al., it was assumed that the angle between the paddle and the thrust

line was constant along the paddle and equal to the angular position γ [74]. However, in the case of a flexible paddle, this is only true for the rigid part of the paddle. For the flexible part of the paddle, the angle of attack takes the following form:

$$\alpha_i = \beta - (\gamma + \zeta_i) \tag{2.14}$$

where β is the angle between the thrust line and the flow and γ is the angle between the paddle and the thrust line as shown in Figure 2–3b. In Figure 2–3b, the thrust line is coincident with the z-axis. Once we know the angle of attack at any location along the paddle, the lift and drag forces can be deduced using the usual equations:

$$\begin{aligned} L_i &= 0.5\rho S_i C_{Li} U_i^2 \\ D_i &= 0.5\rho S_i C_{Di} U_i^2 \end{aligned} \tag{2.15}$$

where ρ is the density of the surrounding fluid, S_i is the surface area of the element, U_i is the local velocity and C_{Li} and C_{Di} are the local lift and drag coefficient. The local velocity U_i is comprised of the velocity of the paddle and the inflow velocity. The local lift and drag coefficient are sinusoidal functions of the angle of attack, as described by Georgiades et al. [73, 74] and Healey et al. [75]. The results by Healey were verified experimentally for a propeller and his model was found to be accurate. Furthermore, Georgiades applied this method to a rigid oscillating paddle and found good correspondence to experimental tests. Then by using (2.14), (2.15) and simple geometry, we can obtain the thrust equation:

$$T = \sum_i [D_i \cos(\beta_i) - L_i \sin(\beta_i)] \tag{2.16}$$

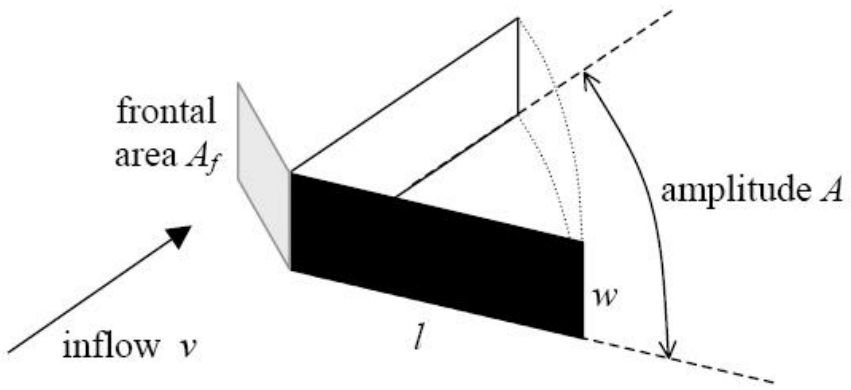


Figure 2–4: Diagram showing the frontal area of the paddle. Credit: Christine Georgiades

where T is the total thrust produced by the paddle. Since β_i is time-periodic, we can easily deduce from (2.16) that the thrust will be time-periodic.

2.2.1 Inflow Velocity

In theory, a paddle performing a symmetric oscillation in a fluid with zero inflow velocity produces no thrust [74]. However, as observed in experiments, an oscillating paddle does produce thrust even in a tank of stagnant water, implying that the paddle produces its own inflow [74]. Therefore, a method to evaluate the inflow velocity needs to be found. During the stroke or recovery phase (half-period), a volume of fluid is displaced by the paddle and fluid in front of the paddle replaces it, causing inflow. The fluid comes in through an area denoted as the inlet area A_f. In [73], this was called frontal area because it was presumed that flow entered from the *front* of the paddle as shown in Figure 2–4. Mathematically, this can be written as:

$$\frac{2V}{P} = A_f v_{in} \tag{2.17}$$

where V is the volume of water displaced by the paddle, P is the period for a complete oscillation and v_{in} is the inflow velocity. The volume of water displaced by the paddle in one half-period is:

$$V = \int_0^A \int_0^l w(x) x dx d\theta = \left(\frac{w_1}{6} + \frac{w_2}{3}\right) l^2 A \tag{2.18}$$

where A is the amplitude of oscillation, $w(x)$ is the width of the paddle at distance x from the hip and θ is angle swept by the paddle.

In the present work, the inlet area for the inflow was different than that used in [73] and was instead defined so as to obtain agreement between simulation and experimental data. The following form was deduced in order to match the simulated average thrust and the corresponding experimentally obtained thrust for different oscillation amplitudes and periods:

$$A_f = (\frac{w_1 + w_2}{2})\frac{l}{2} A \tag{2.19}$$

This represents the area swept by the paddle at half-length. Using (2.16), (2.18) and (2.19), an expression for the inflow velocity can be obtained:

$$v_{in} = \frac{4l}{3P} \frac{w_1 + 2w_2}{w_1 + w_2} \tag{2.20}$$

As can be seen in equation (2.20), the inflow velocity depends only on the period of oscillation and on the geometry of the paddle, but not on the amplitude

of oscillation. Moreover, we have to keep in mind that (2.20) represents an average inflow velocity. In the case of our oscillating paddle, the inflow velocity is likely to be periodic with an average equal to (2.20).

More accurate models of the inflow velocity could be derived using computational fluid dynamics(CFD) or vorticity-based methods. However, the model presented in this section has the advantage of being simple. Moreover, it would be useful to measure the inflow velocity experimentally in the future.

2.3 Dynamics Simulation

This section discusses the dynamics simulation of the Aqua underwater vehicle that is comprised of the body model (Section 2.1) and the forward paddle model (Section 2.2). There are three main uses for this simulation: to help validate the models discussed in Sections 2.1 and 2.2, to develop controllers for the vehicle and to act as a testbed for future experiments. The simulation was programmed using the MATLAB Simulink software environment and was based on the dynamics simulation developed by Georgiades [73]. A snapshot of the Simulink screen can be seen in Figure 2–5. The inputs to the simulation are the paddle amplitude, period and offset of oscillation as well as the initial state of the vehicle. The initial state refers to the initial velocity and position of the vehicle in their respective coordinate frame. The simulation outputs the state of the vehicle, the paddle angles and the paddle forces.

The first step in the simulation is to use the paddle parameters (amplitude, period and offset) to compute the paddle trajectory in a continuous form, $\gamma = \frac{A}{2}\sin\left(\frac{2\pi}{P}\right) + \lambda$. More information about this step can be found at [73]. The model

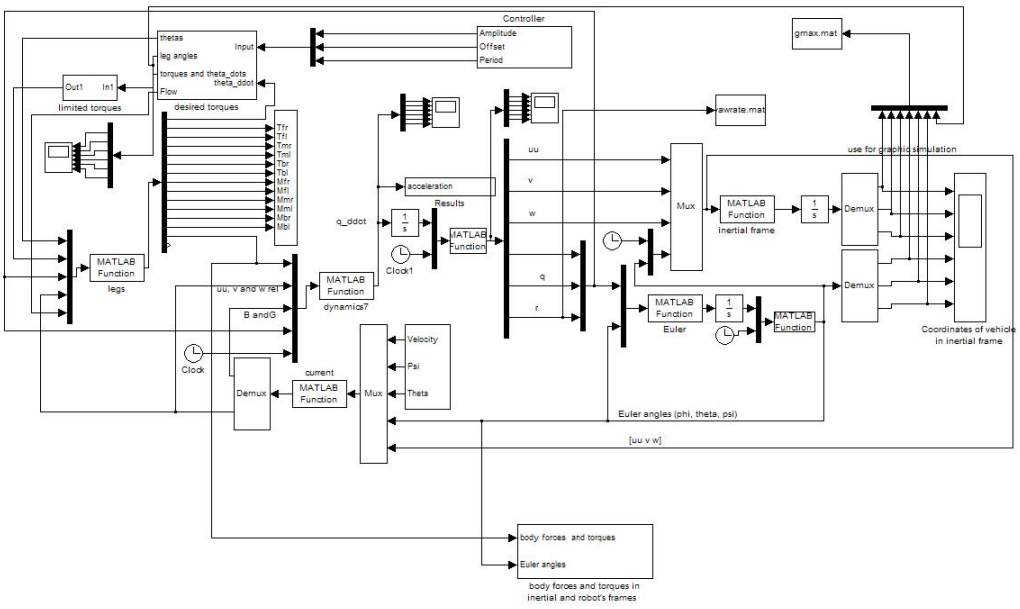


Figure 2–5: The MATLAB Simulink simulation of the vehicle

described in Section 2.2 is then used to calculates the paddle forces and moments ($\mathbf{f}$) in the six degrees of freedom. Finally, the force $\mathbf{f}$ serves as the input for (2.6) that outputs the acceleration of the vehicle. The velocity and position of the vehicle can then be obtained by integration of the acceleration.

2.4 Model Validation

This section describes the experimented performed to validate the paddle and vehicle model. The objective was to confirm that the models were accurate in predicting the behavior of the actual system. The first part of the section discusses the validation of the paddle model where we evaluate the thrust produced by paddle. In the second part, the body and paddle models were combined to form the vehicle model. The vehicle model was validated by comparing the motion of the vehicle in

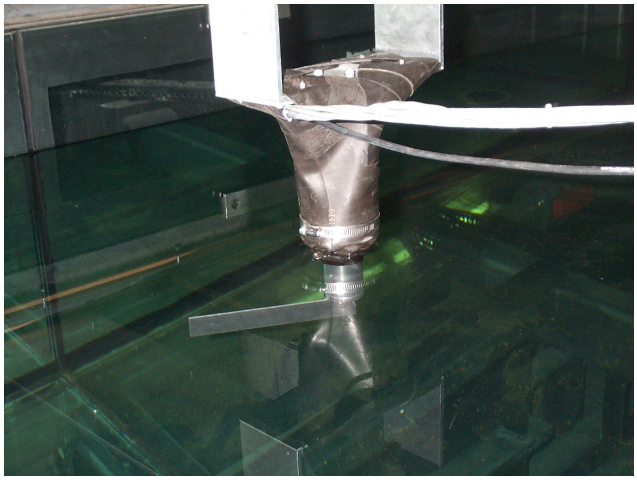


Figure 2–6: Picture of the thrust measuring setup in a stagnant water tank. Credit: Christine Georgiades

simulation and experiment, for the same input paddle motion. Proper validation of the simulation will give us confidence that we can use our dynamics model to design controllers and as a general testbed for the vehicle.

2.4.1 Validation of the paddle model

The experimental data used to validate the paddle model was obtained from experiments conducted by Georgiades [73, 74]. The testbed was a stagnant water tank having a length of 6m, a width of 1.5m and a depth of 1.2m as shown in Figure 2–6. A force/torque sensor located at the paddle hip recorded the force produced by the paddle and an encoder recorded the paddle motion. The only inflow into the paddle was due to the paddle motion. The experiment was performed with a single paddle.

The experimental results obtained in the stagnant water tank were compared to those obtained in a simulation based on the model described earlier in Section 2.2. The simulation was developed using MATLAB Simulink. The paddle parameters

$P(s)$	$A(^\circ)$
0.4	7.3, 9.1, 10.4, 12.3
0.6	14.8, 20.3
0.8	25.3, 32.2, 38.6, 44.2
1.0	31.3, 40.6, 47.6, 55.9

Table 2–1: Amplitude and period of oscillation used in the forward model validation

used in the simulation were l=0.2m, w_1=0.04m, w_2=0.07m, $C_{Dmax} = 1.11$, $C_{Lmax} = 1.2$ and EI=0.05Nm^2. C_{Dmax} and C_{Lmax} are the maximum values for the lift and drag coefficient. They are used to calculate the instantaneous lift and drag coefficient used in (2.15) to compute the lift and drag force. It assumes that the lift and drag coefficients are continuous functions of the angle of attack [73,75] The last parameter, EI, was obtained experimentally by applying loads to the paddle and measuring its deflection. The values of P and A that were used in the experiment are tabulated in Table 2–1.

As shown in Figures 2–7 and 2–8, the simulation results closely matched the experiment. Figure 2–7 shows that the average thrust produced by the paddle increases with the amplitude of oscillation for a given period, and that thrust increases with frequency of oscillation. The average thrust is obtained by computing the time-averaged thrust once steady-state is reached. We also see in Figure 2–8 that, although the experiment slightly overshoots the simulation, the instantaneous thrust is very similar. From this, we can conclude that the flexible paddle model accurately predicts the thrust produced by an oscillating paddle, and that the model can be used with confidence in the controller development. The next step is to find a relation that relates the paddle motion to the thrust. This relationship will be used in

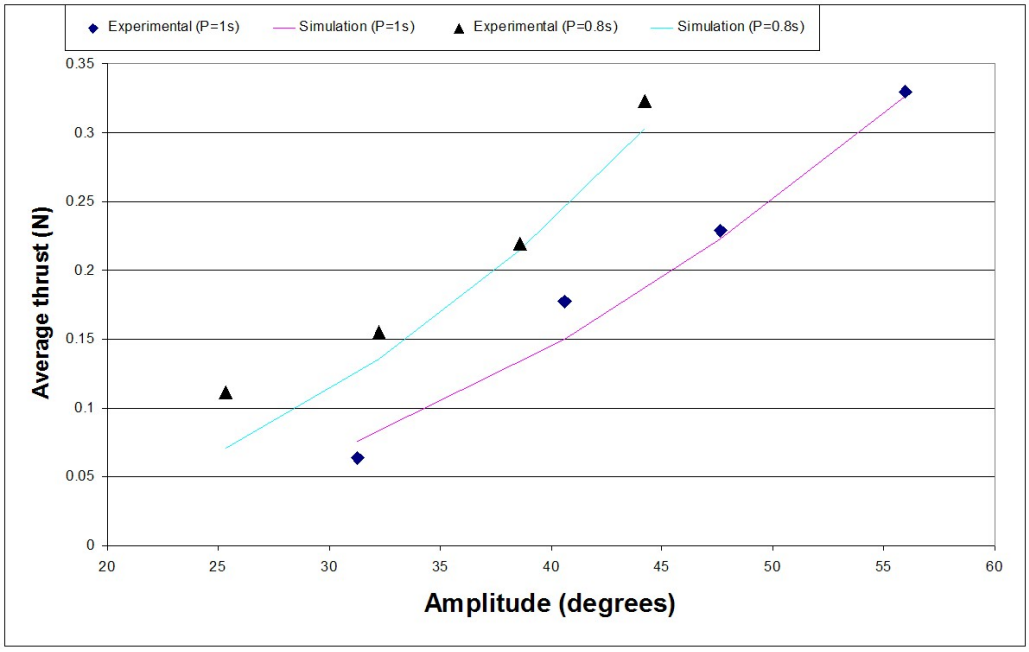


Figure 2–7: Comparison of experimental and simulated average thrust

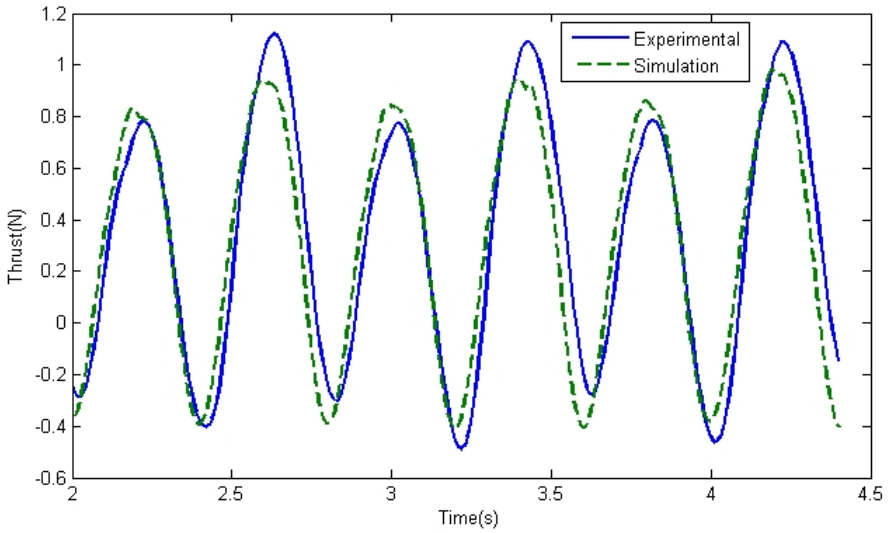


Figure 2–8: Comparison of experimental and simulated instantaneous thrust for A= 30° and P=0.4s

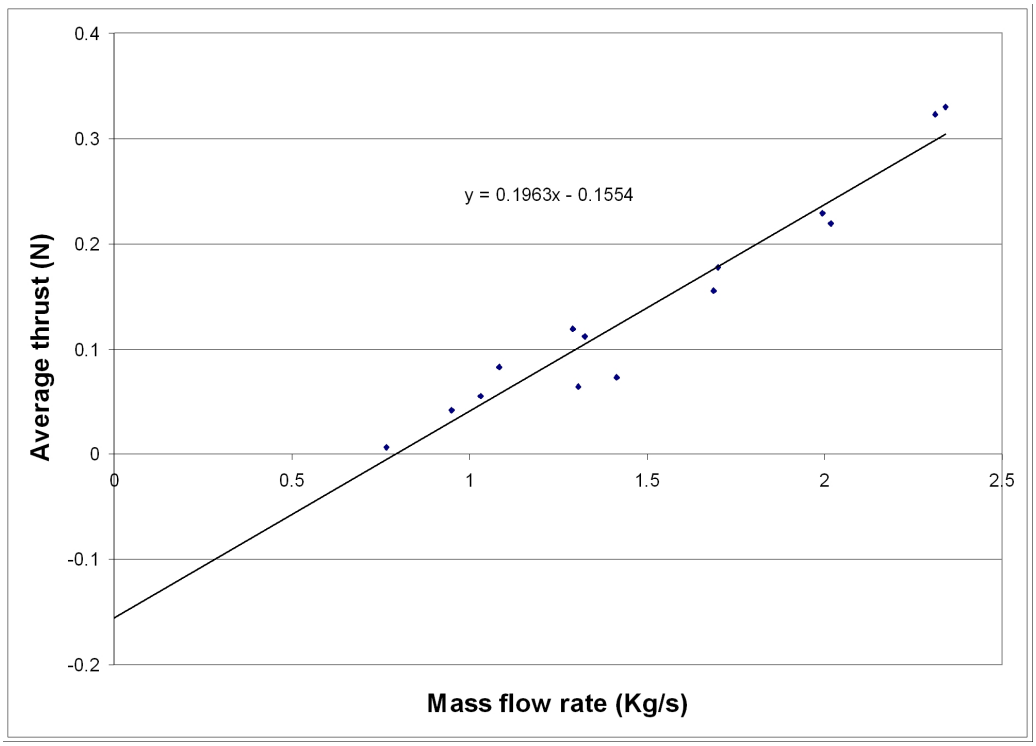


Figure 2–9: Average thrust as a function of the mass flow rate

Section 2.5 to create an inverse model that takes the thrust and finds the appropriate paddle motion.

We know from momentum theory for traditional propellers that the thrust produced is proportional to the mass flow rate of fluid being accelerated. Therefore, we plotted the thrust as a function of the mass flow rate, with the mass flow rate defined as $\rho A_f v_{in}$. The data points shown in Figure 2–9 include experiments performed at 4 different oscillation frequencies and 14 different oscillation amplitudes. As shown in Figure 2–9, this appears to show a linear relationship between the two variables. A linear regression gives us the thrust as a linear function of the mass flow rate:

$$T = K_1 \rho A_f v_{in} + K_2 = 0.1963 \rho A_f v_{in} - 0.1554 \tag{2.21}$$

where T is the thrust produced by the oscillating paddle in Newton and ρ is the density of the fluid. It is important to note that equation (2.21) is valid only for a certain range of mass flow rate and that the inflow velocity v_{in} is along the thrust line. From Figure 2–9, this range is from 0.75 kg/s to 2.35kg/s. Beyond that range, we have no guarantee that (2.21) gives an accurate evaluation of the thrust. By combining (2.19), (2.20) and (2.21), we obtain an expression for thrust that depends only on paddle parameters:

$$T = 0.1963 \frac{(w_1 + 2w_2)\, l^2}{3} \rho \frac{A}{P} - 0.1554 \tag{2.22}$$

It is important to note that this model is not perfect and does not capture paddle interference or relative flow angle. However, it has been found to be accurate for our purpose.

2.4.2 Validation of the vehicle model

The vehicle model is used to develop controllers and to assess the performance of the vehicle during maneuvers. In order to use the model with confidence, it must be validated experimentally. The objective of the validation is to validate the model components, body and paddles, as well as the overall model. Ideally, the onboard sensors would record all the vehicle states of the vehicle and it would be straightforward to compare them with those in the simulation. However, the vehicle has limited sensing capabilities and some states cannot be measured, and we had to develop indirect methods to evaluate them.

Constant Speed Validation

The forward speed is the principal motion we control on Aqua and it is therefore important that the simulation closely match the actual vehicle. In this section, we present the results of the constant speed validation experiment, in which the objective is to compare experimental speed data with our simulation results. The experiment assumes that we have steady speed, steady drag force and a steady paddle thrust. The forward speed depends on the paddle amplitude and period of oscillation.

Aqua does not have onboard sensor to directly measure its speed. In principle, the IMU's accelerometer signal could be integrated to obtain the speed but the measurements are noisy and any bias introduces drift in the integrated signal. Therefore, we chose to measure the speed by external observation. The vehicle was placed in the water at one end of a pool of 25m length and then driven in straight line toward the other end of the pool. Using a stop watch, the time taken to travel between two points was measured. The average velocity was then defined as the distance between the points divided by the time taken to travel the distance. We ensured that the vehicle was no longer accelerating between those two points. Therefore, this experiment only validates those parts of the model that affect the steady forward motion of the vehicle (i.e., not the mass/inertia characteristics). The measurement was repeated 3 times for each paddle motion (amplitude, period), and an average was used to calculate the speed, thereby reducing the measurement uncertainty. The average velocity varied by about 3% between the three repetitions. The differences can be due to a number of factors such as water turbulence or inaccuracies in manual stopwatch triggering. The accuracy could be improved by improving the sensing

Amplitude(deg)	Period(s)	Simulation(m/s)	Experiment(m/s)
25	0.4	0.6	0.58
21	0.4	0.55	0.54
21	0.3	0.66	0.65
18	0.3	0.62	0.61
22	0.3	0.67	0.67
20	0.2	0.82	0.80

Table 2–2: Experimental and simulated speed for different paddle motion

capabilities of the vehicle and enabling accurate speed measurements. This would remove the need from external speed measurement techniques.

Table 2–2 shows the vehicle speed obtained in the experiment and in the simulation for different combinations of paddle amplitude and period of oscillation. The first thing to notice is that the simulation closely matches the experiment. Moreover, the velocity increases with amplitude of oscillation and decreases with period of oscillation. This was expected since the velocity is directly related to the force produced by the paddles, and the force had been shown to vary similarly in Section 2.4.1 and in (2.22).

As was mentioned previously, we do not have an onboard sensor to measure the speed of the vehicle, and therefore cannot measure its speed during operation. The data presented in Table 2–2 could be used to approximate the speed of the vehicle at steady-state by inferring from it an equation relating the paddle motion to the speed. To do this, we referred to the flexible paddle model that predicts the thrust produced by the oscillating paddles. We found in Section 2.4.1 that the thrust was proportional to the ratio of amplitude over period. Moreover, at steady-state, we can assume that the only longitudinal forces acting on the vehicle are the paddle

forces and the drag force, and that these are equal and opposite. The drag force is proportional to the square of the velocity. In mathematical term, this gives:

$$F_{drag} \propto u^2, \quad F_{paddle} \propto \frac{A}{P}, \quad F_{paddle} = F_{drag} \tag{2.23}$$

Since these are the only forces acting on the vehicle in steady forward cruise, (2.23) suggests that the velocity of the vehicle should be proportional to the square root of the ratio of amplitude over period. Figure 2–10 shows the vehicle velocity as a function of $\left(\frac{A}{P}\right)^{1/2}$ and we can see that there is indeed a linear relationship between the two variables. By performing a linear regression on the data we can obtain equation for the speed of the vehicle as a function of the paddle motion:

$$\begin{aligned} u_{sim} &= (0.7345\sqrt{\tfrac{A}{P}} - 0.155)\tfrac{m}{s} \\ u_{exp} &= (0.7254\sqrt{\tfrac{A}{P}} - 0.155)\tfrac{m}{s} \end{aligned} \tag{2.24}$$

where u_{sim} and u_{exp} are the vehicle velocities from the simulation and experiment. The relationships in (2.24) are only valid for the range of period and amplitude presented in Table 2–2. Clearly, the velocity should be zero when $\sqrt{\frac{A}{P}}$ approaches zero but (2.24) contradicts that. This is because for smaller $\frac{A}{P}$, we do not have any data and the relation presented in (2.24) is likely no longer valid.

From the results presented in Table 2–2 and Figure 2–10, we can conclude that the dynamics model closely matches the experiment under these conditions. Moreover, (2.24) can be used to deduce the steady-state speed of the vehicle during operation.

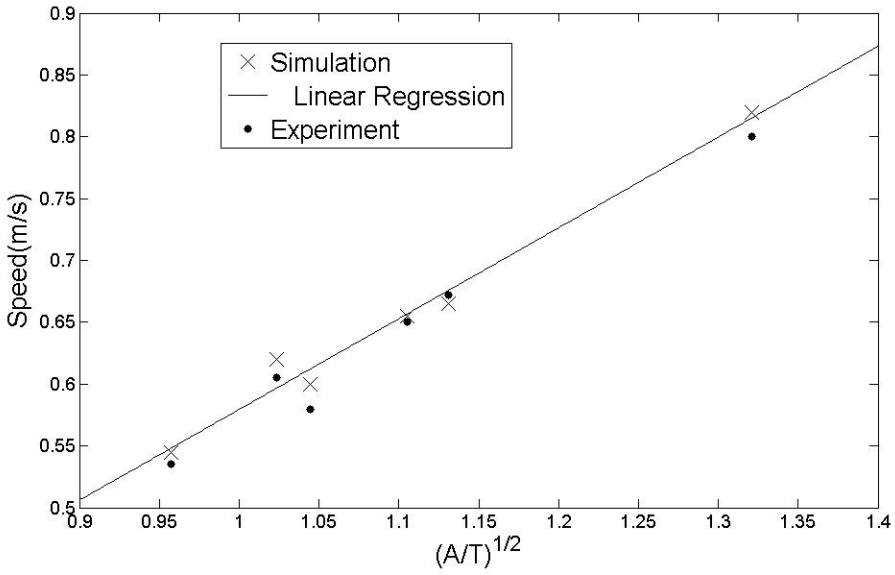


Figure 2–10: Experimental and simulated speed of the vehicle in function of the amplitude over period square

Roll Validation

In this section compare the simulation and experimental results in roll maneuvers. A good match will imply accuracy of the paddle model, the roll inertia and the drag properties of the vehicle. Moreover, the accuracy of the simulation depends on the accuracy of the paddle model described in Section 2.2. The paddle model was never validated for a rotation maneuver and this could therefore negatively affect the validation in the roll motion.

Aqua has accurate sensors to measure the roll rate and roll angle. The vehicle was run in pool tests and the pilot performed roll maneuvers while the pitch and yaw motions were left uncontrolled. We recorded the roll angle, roll rate and the paddle motion as well as the commanded roll from the pilot's joystick.

In order to compare the experimental data to the simulation, we ran the simulation using the paddle motion recorded during the experiment, so that the input is the same in both cases.

We placed priority on comparing the roll rate rather than the roll angle for two reasons. First, the roll rate is part of the equation of motion of the vehicle and it directly affects the dynamics of the system, while the roll angle only affects the dynamics indirectly through the kinematics. Second, the roll angle measurement is less accurate than the roll rate measurement because it is obtained by fusing the low frequency accelerometer measurement with the integral of the roll rate measurement (i.e. the roll angle is not measured directly). Therefore, any bias in roll rate would be integrated and would result in a growing roll angle error. However, the opposite behavior can be observed in Figure 2–12 and we see that the simulation is drifting while the experiment returns to zero. The general motion is similar but there is an offset between the simulated and experimental roll angle. One reason to explain this phenomenon could be explained by the slight asymmetries of the vehicle. In the experiment this is compensated by the pilot who tried to keep the vehicle level before starting the maneuvers. The simulation currently assumes that the vehicle is perfectly symmetric. Therefore, the pilot compensation has the effect of creating a drift in the simulation. To confirm that, we changed the lift coefficient by 10% and the length of the paddle by 2% and we observed a change in the motion of the vehicle while performing a maneuver.

Figure 2–11 shows the results of the roll validation experiments. We can see the simulation closely matches the experiment in the four trials shown on the figure. Even

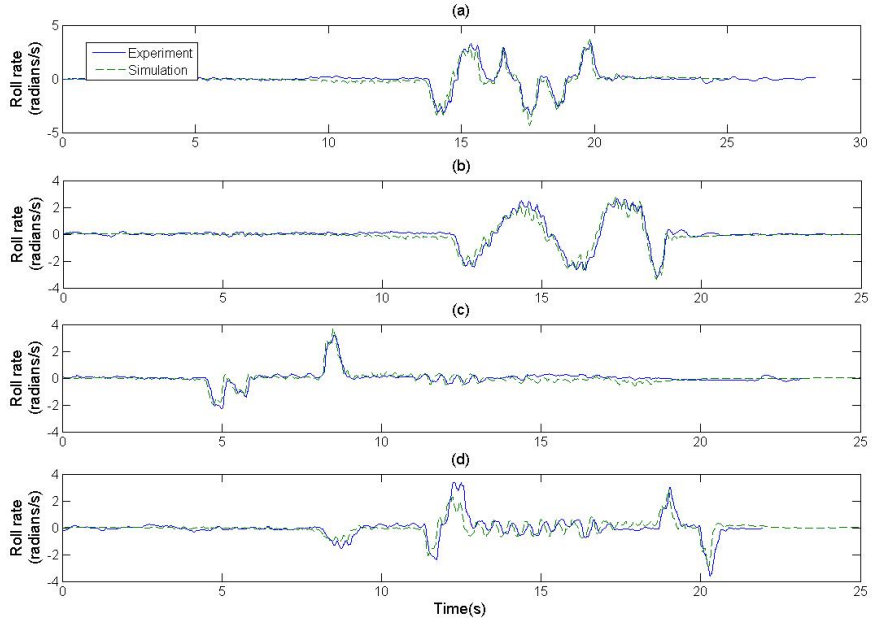


Figure 2–11: Experimental and simulated roll rate for 4 different roll trajectories.

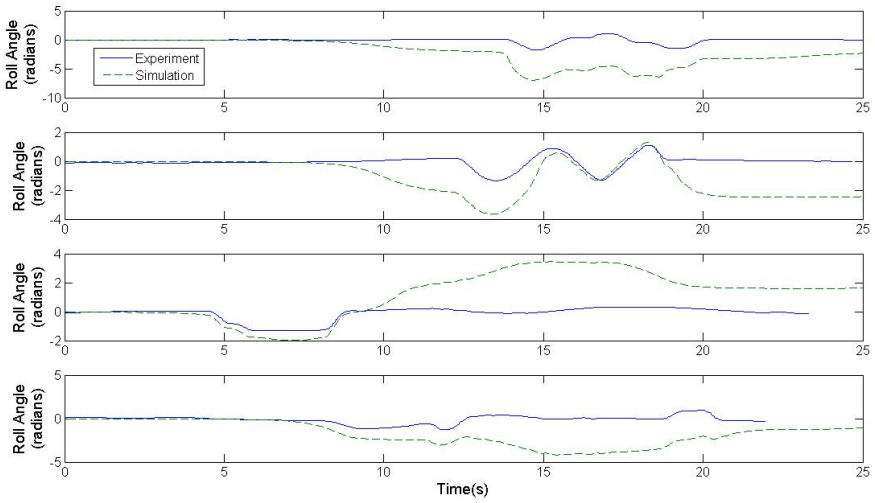


Figure 2–12: Experimental and simulated roll angle for 4 different roll trajectories.

in the presence of sharp change in roll rates, the simulation follows the experimental data. However, the match is not perfect there are small discrepancies between the experiment and simulation. It is also important to note that we do not possess all the initial information about the system. The forward velocity has an impact on the thrust produced by the paddles and that affects the roll motion. In the previous section, we saw that we were accurate in terms of steady-state velocity. However, while performing a maneuver, the velocity will change and we do not know how accurate our model is in this case. Moreover, in the simulation, the vehicle is assumed to be in an ideal environment with no disturbance, while in the pool, the vehicle may be subject to disturbances that are unpredictable. Moreover, some differences may be due to unsteady effects such as unsteady hydrodynamic or interference effects. Based on the results shown in Figure 2–11, we can conclude that the model demonstrates good correspondence to te real vehicle in roll maneuvers.

Pitch Validation

In this section we present a comparison of the simulation and experimental results in pitch maneuvers. As was the case for the roll validation, a good match depends on the accuracy of the paddle model, the pitch inertia and the drag properties of the vehicle.

As was the case for the roll motion, the onboard sensors provide all the necessary information to perform the validation. The vehicle was tested in pool trials and the driver performed pitch maneuvers while the roll and yaw motions were left uncontrolled. Again, the experimentally measured paddle motion was used as the input to the simulation.

In all three maneuvers the vehicle was controlled by a human pilot. In the first two maneuvers, the vehicle was initially left uncontrolled. Then, the pilot sent open loop control signals to make the vehicle pitch. These pitching commands were sent between $t = 6s$ and $t = 19s$ for the first maneuver and between $t = 10s$ and $t = 20s$ for the second maneuver. In the third maneuver, no control signal was sent from the pilot to the vehicle.

We placed the priority on comparing the pitch rate rather than the pitch angle for similar reasons to those mentioned in the roll validation section. The pitch rate is shown in Figure 2–13. In Figure 2–13a, the match is acceptable but in the two others, there is almost no match at all.

It was not possible to use large values of pitch rate because we were concerned about hitting the bottom of the pool if the pitch angle was allowed to grow too much. Moreover, climbing to the surface is also a problem since the simulation does not model the water surface. At the surface, the vehicle cannot continue pitching up, even if the paddle motion implies that it should.

In Figure 2–13, we can particularly observe the presence of pitch rate oscillations that are larger in the experiment than in the simulation. This oscillation is due to the vehicle's reaction to the motor torque. The differences in oscillation amplitude present in Figure 2–13 are likely due to a difference in this torque.

Figure 2–14 shows the match in pitch angle. We can see that there is not a good match between the simulation and experiment. This is due to the fact that any mismatch in pitch rate was integrated and grew over time. Moreover, disturbances

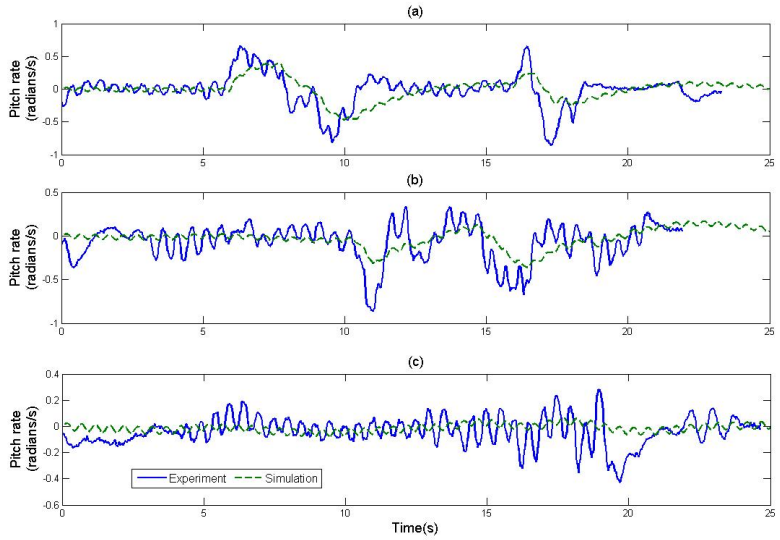


Figure 2–13: Experimental and simulated pitch rate for 3 different pitch trajectories.

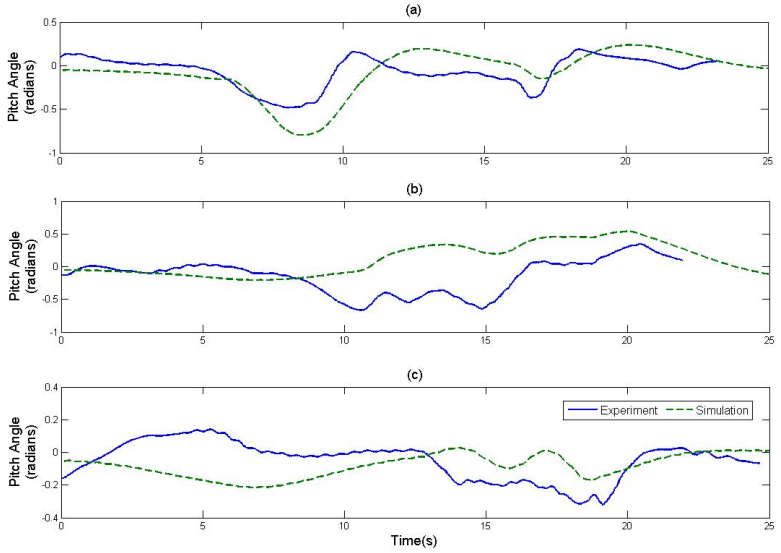


Figure 2–14: Experimental and simulated pitch angle for 3 different pitch trajectories.

in the experiment could not be accounted precisely in the simulation and caused differences between the two.

We investigated whether the match in pitch could be improved by modifying the simulation parameters pertaining to the pitch motion. These include the pitch moment of inertia, the added mass coefficient pertaining to the pitch motion, the drag coefficient and the motor torque. The pitch moment of inertia was initially obtained from the CAD model of the vehicle. However, components have been moved and it is possible that there is an error in our value. The added mass and drag coefficients were obtained based on a rectangular box shape vehicle. Since Aqua has rounded shape ends, it is likely that the drag and added mass coefficients would be lower than for a rectangular box.

Figure 2–15 and 2–16 shows the results of the pitch validation experiment using the modified parameters. The moment of inertia was left unchanged because it had little impact. The drag coefficient of the paddle was reduced by 10% and the motor torque was adjusted to match the one deduced from experimental results. Previously, the motor torque was evaluated by the simulation with a PD controller: $\tau = k_{d\tau}\left(\dot{\gamma}_d - \dot{\gamma}\right) + k_{p\tau}\left(\gamma_d - \gamma\right)$. By varying the motor torque gains, $k_{d\tau}$ and $k_{p\tau}$, we can adjust the torque produced by the motors. That torque was different from the one calculated from the measured motor current. The current is computed using the motor model from the manufacturer and the angular velocity of the paddle [11]. We chose the motor torque gains so that the experimental and simulated motor torque match.

	I_{yy}	$M_{\dot{q}}$	M_{q^2}	$k_{d\tau}$	$k_{p\tau}$
Old parameters	0.124 kgm^2	-1.19 Nms^2	0.673 Nms^2	2 Nms	1 Nm
New parameters	0.150 kgm^2	-1.25 Nms^2	0.800 Nms^2	5 Nms	5 Nm

Table 2–3: Old and new parameters used in the dynamics model

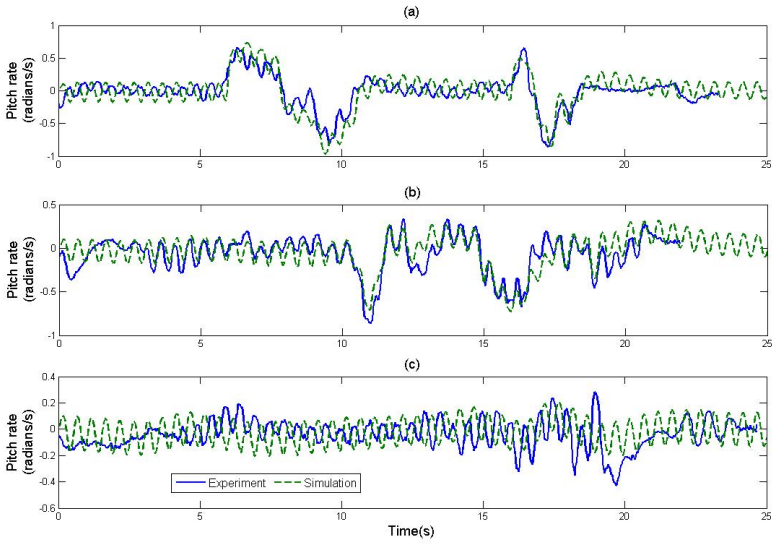


Figure 2–15: Experimental and simulated pitch rate for 3 different pitch trajectories with modified parameters.

We can see that the match is substantially better, particularly in terms of the paddle oscillations. As was mentioned previously, it is difficult to obtain a perfect match since there are uncertainties and disturbances that cannot be accounted for in the simulation. Moreover, the moment arm of the paddles is longer for pitch than for the other degrees of freedom, so that small errors in paddle thrust will yield larger errors in pitch than in roll. Based on Figure 2–15, we can conclude that our dynamics model is reasonably accurate in the pitch motion and therefore, the new parameters will be retained.

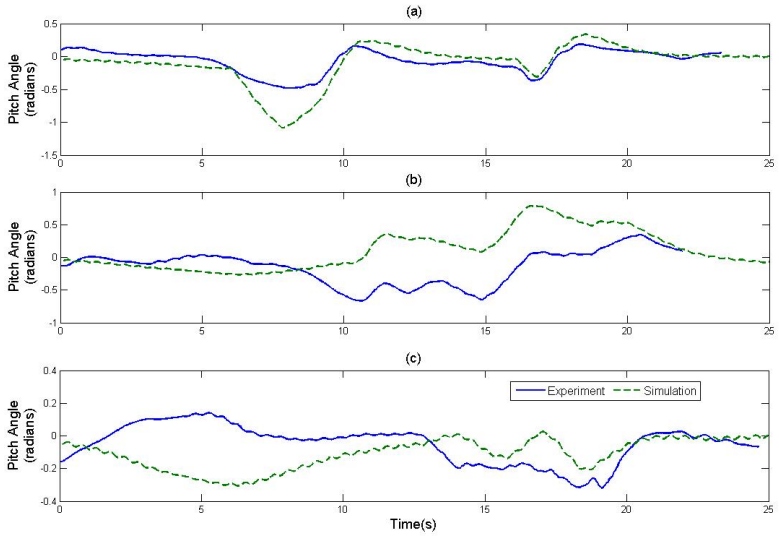


Figure 2–16: Experimental and simulated pitch angle for 3 different pitch trajectories with modified parameters.

Yaw Validation

The objective is now to compare simulation and experimental results in yaw maneuvers to evaluate the accuracy of the simulation in this degree of freedom. A good match depends on the accuracy of the paddle model, the yaw inertia and the drag properties of the vehicle.

Unlike roll and pitch, we do not have an accurate sensor to measure the yaw motion of the vehicle. The IMU and the compass give inaccurate measurements due to electromagnetic interference from the electronics around them. However, we have observed that if the vehicle remains level, we get an error of 5° for a turn of 360°. As will be shown, keeping the vehicle level is almost impossible without any user input and therefore the results are affected.

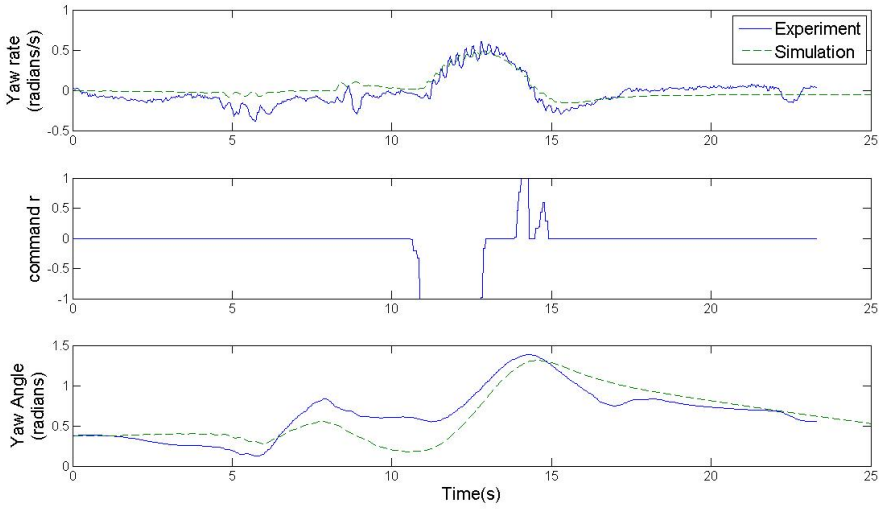


Figure 2–17: Yaw rate, commanded yaw input and yaw angle for the first yaw maneuver

As was done for the previous validations, we ran the simulation using the paddle motion recorded during the experiment, so that the input is the same in both cases.

Figure 2–17 to 2–19 show the results of the yaw validation for a pure yaw maneuver. In all cases, the roll and pitch motion was kept to a minimum but we were unable to keep them at zero for the whole maneuver. As we can see, there is a good match in yaw rate for all three maneuvers. The match is best in the presence of a yaw command. With no commanded yaw, the simulation tends to have a near-zero yaw rate while the experiment shows some yaw-rate perturbations. This yaw motion in the experiment is likely due to water disturbances or other unmodeled features such as asymmetric paddles.

In terms of yaw angle, the results are quite different, in part because in the simulation, the yaw angle is obtained by integrating the yaw rate. Since the yaw rate does not match well in the absence of a commanded yaw motion, its integration

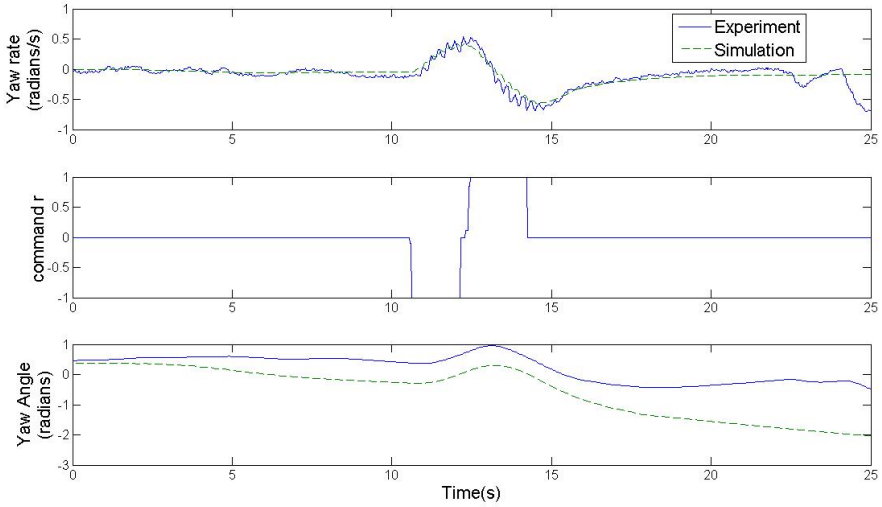


Figure 2–18: Yaw rate, commanded yaw input and yaw angle for the second yaw maneuver

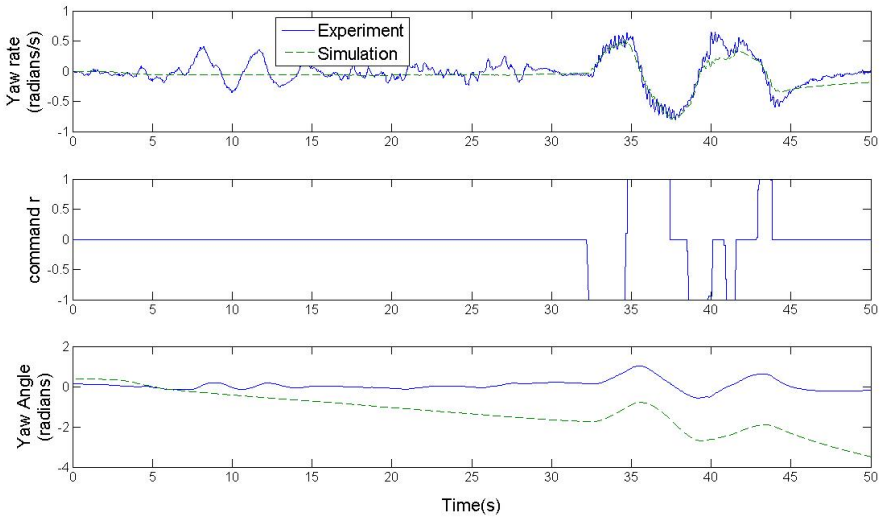


Figure 2–19: Yaw rate, commanded yaw input and yaw angle for the third yaw maneuver

creates a growing error. For example, in Figure 2–18 and 2–19, we can see that the yaw angle has the same shape, but that due to a difference in yaw rate, there is a growing gap in yaw angle. This difference is particularly visible in Figure 2–18 between $t = 5$s and $t = 10$s and in Figure 2–19 between $t = 7.5$s and $t = 30$s. Given our inability to model all possible disturbances and other circumstances we are satisfied with these results and we can conclude that the dynamics model is accurate in yaw.

2.5 Reverse Model

Vehicle controllers typically output required forces and moments that must be applied to the vehicle in order to follow a prescribed trajectory. However, the only variables that can be changed on Aqua are the amplitude, period and offset of paddle oscillation. Generally, the amplitude and period of oscillation determine the output thrust, while the offset angle determines the direction in which the thrust is produced. A mapping is therefore needed to determine the paddle motion that should be used to accomplish the forces and moments specified by the controller. This mapping will be used in the simulation as well as on the real robot.

2.5.1 Force Distribution

The controller, the design of which will be discussed in Section 3, outputs the desired forces and moments in the six degrees of freedom. These are assembled in the vector $\mathbf{f}$. In order to determine the paddle motion, we first need to know what forces each paddle must produce. On the Aqua robot, each of the six paddles can generate a force in the x and z-direction. Since $\mathbf{f}$ has six components and there

are twelve component paddle forces, this problem admits many solutions. Quadratic programming [76] is used to formulate this problem as a minimization problem:

$$\min_{\mathbf{f}_p} \frac{\mathbf{f}_p^T \mathbf{H} \mathbf{f}_p}{2}$$
$$\text{subject to } \mathbf{A}\mathbf{f}_p = \mathbf{f} \tag{2.25}$$
$$\mathbf{f}_p = [f_{x1} \quad \cdots \quad f_{x6} \quad f_{z1} \quad \cdots \quad f_{z6}]^T$$

where $\mathbf{H}$ in this case is an identity weighting matrix, and $\mathbf{A}$ is a 6×12 matrix relating $\mathbf{f}_p$ to $\mathbf{f}$. The vector of paddle forces, $\mathbf{f}_p$, consists of f_{xi} and f_{zi} which are the force required by the i'th paddle in the x and z direction, respectively. It is important to note that, with the present paddle configuration, it is impossible to generate paddle forces in the y-direction. In order to accommodate the generation of lateral forces, a desired y-force is transformed into a yaw moment command. By doing this, the vehicle will change its heading so that the nose of the vehicle points toward the direction of the y-force. In mathematical terms, this means that the force $\mathbf{f}$ in (2.25) can be written as:

$$\mathbf{f} = [\ F_x \quad 0 \quad F_z \quad M_x \quad M_y \quad M_z + kF_y\ ]^T \tag{2.26}$$

where F_x, F_y, F_z, M_x, M_y, and M_z are the desired forces and moments calculated by the controller and k is a length weighting factor that allows the user to choose a balance between the vehicle's response to lateral force commands and yaw moment commands. For the Aqua vehicle, $k = 1$ m was found to give a good balance of performance.

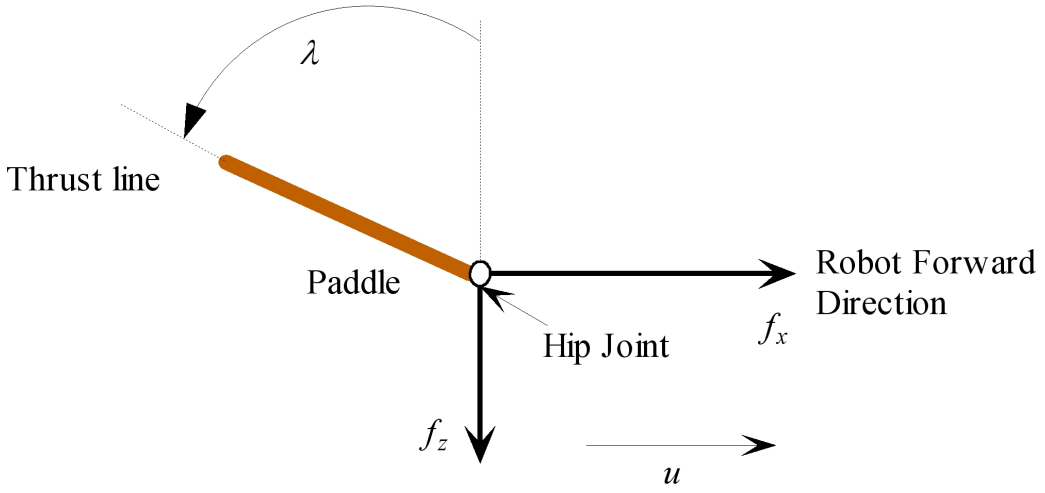


Figure 2–20: Paddle force and offset angle

2.5.2 Offset Angle

Once f_{xi} and f_{zi} are known for each paddle, we can proceed to determine the necessary paddle motion. From Figure 2–20, the required thrust (T) produced by the paddle and its offset angle (λ_i) are easily deduced:

$$T_i = \sqrt{f_{xi}^2 + f_{zi}^2} \qquad \lambda_i = \tan^{-1}(\tfrac{f_x}{f_z}) \tag{2.27}$$

The force T acts along the thrust line that was described in Section 2.2 and λ_i is the angle between the negative z-axis of R_V and the thrust line. The thrust line can be understood as the direction in which a net thrust is produced and it is coincident with the line about which the paddle oscillates.

2.5.3 Paddle Motion

The paddle thrust T must now be transformed into an amplitude and period of oscillation using the relation described by (2.22). However, before doing this, we must modify (2.22) to account for the fact that the inflow is affected by the vehicle motion. We assume that the flow velocity at paddle i, v_i, is composed of two components: the velocity of the paddle at the centre of pressure and the inflow caused by the moving paddle. A key assumption here is that the flow velocity is the sum of these two components. This velocity v_i replaces v_{in} that was used in (2.21). Therefore, we need to find the component of the velocity along the thrust line. We use equation (2.20) to express the inflow velocity caused by the paddle as a function of the period of oscillation:

$$\begin{aligned} v_i &= -U_i \bullet [\ \sin(\lambda) \quad 0 \quad \cos(\lambda)\ ] + v_{in} \\ &= -U_i \bullet \left[[\ \sin(\lambda) \quad 0 \quad \cos(\lambda)\ ] \right] + \tfrac{4l}{3P_i} \left(\tfrac{w_1+2w_2}{w_1+w_2} \right) = u + \tfrac{4l}{3P_i} \left(\tfrac{w_1+2w_2}{w_1+w_2} \right) \end{aligned} \tag{2.28}$$

where U_i is the velocity of paddle i at the hip. The velocity of the paddle includes the effect of the translational and rotational motion of the vehicle, as well as the motion of the paddle relative to the vehicle. The dot product operation in the first term of (2.28) retains only the component of the paddle velocity that is parallel to the thrust line. Combining (2.19), (2.21) and (2.28) together, we can obtain an expression for the forward thrust T that depends only on the velocity, amplitude and period of oscillation:

$$T = 0.0654\rho l^2 \left(w_1 + w_2\right) \frac{A_i}{P_i} - 0.049\rho l \left(w_1 + w_2\right) A_i u - 0.1554 \tag{2.29}$$

From (2.29), we can see that many combinations of A and P will provide the desired force. We formulate this problem as a minimization in which we try to remain near desirable values of A and P:

$$\min_{A,P} \left\{ z = W_A (A - A_0)^2 + W_P (P - P_0)^2 \right\} \tag{2.30}$$

where W_A and W_P are the weights placed on amplitude and period respectively and A_0 and P_0 are desirable values of the paddle oscillation amplitude and period. For the present work, the values A_0=0.4rad and P_0=0.5s were selected because they were found to provide good vehicle performance (speed, efficiency, maneuverability) in our experiments. This period and amplitude of oscillation are routinely used when operating the vehicle manually. Solving equation (2.30) with (2.29) as constraint, we are left with a system of two equations in two unknowns, A and P:

$$\begin{gathered} 490 W_A (A - A_0)\left(\tfrac{F+0.1554}{A}\right) \tfrac{(w_1+2w_2)}{\left(4l(w_1+2w_2) - 3uP(w_1+w_2)\right)^2} + 2W_P (P - P_0) = 0 \\ A - 81.5 \tfrac{F+0.1554}{(w_1+w_2)A} \tfrac{(w_1+2w_2)}{4l(w_1+2w_2) - 3uP(w_1+w_2)} = 0 \end{gathered} \tag{2.31}$$

Equation (2.31) is solved as a set of two equations and two unknowns using the MATLAB/MAPLE function *solve.m* which solves the system of equations in closed form for polynomial of order 4 or less, which is the case here. If u is not 0, we typically obtain four solutions. We select the most appropriate solution by selecting

the one in which A and P are purely real and positive. In all the tests we have performed, these two criteria were sufficient to obtain a single solution. Two of the four solutions are complex conjugates, one has a negative A and/or P and the last one is the valid solution. In the case where no valid solution can be found, the function would output A_0 and P_0.

2.6 Linearization

The dynamics model described in Sections 2.1 and 2.2 is nonlinear. It is convenient for a computer simulation but for some controller design, a linear model is necessary. In this work, an adaptive controller (Section 3.4) and a Floquet controller (3.5) requires a linear model.

The general nonlinear model of a general moving system can be written as:

$$\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x},\mathbf{f}) \tag{2.32}$$

where $\mathbf{f}$ is the net propulsive force coming from the paddles and $\mathbf{x}$ is the state vector as defined by (2.1) and (2.2). $\mathbf{f}$ is a force oscillating with a frequency twice the frequency of oscillation of the paddle. The nonlinear model was described in Section 2.1 and includes the inertial force, the hydrodynamic drag, the Coriolis forces, gravity and buoyancy. The linear model would then take the following form:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{f} \tag{2.33}$$

where $\mathbf{A}$ is 12 by 12 matrix and $\mathbf{B}$ is 12 by 6 matrix. $\mathbf{A}$ is defined as:

$$\mathbf{A} = \frac{\partial \mathbf{F}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial F_1}{\partial x_1} & \frac{\partial F_1}{\partial x_2} & \cdots & \frac{\partial F_1}{\partial x_{12}} \\ \frac{\partial F_2}{\partial x_1} & \frac{\partial F_2}{\partial x_2} & \cdots & \frac{\partial F_2}{\partial x_{12}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial F_{12}}{\partial x_1} & \frac{\partial F_{12}}{\partial x_2} & \cdots & \frac{\partial F_{12}}{\partial x_{12}} \end{bmatrix} \tag{2.34}$$

There are several methods that can be used to linearize a nonlinear time-invariant system. A direct and common ways to obtain a linear model is numerical differentiation by finite difference, which requires that the system start at an equilibrium point. The system is then disturbed from that equilibrium by perturbing one of the state variables. The resulting response is obtained and from these, we calculate:

$$A_{ij} = \frac{\Delta \dot{x}_i}{\Delta x_j} \tag{2.35}$$

where $\Delta \dot{x}_i$ is the rate of change of the i'th state for a given change Δx_j of the j'th state. Because Aqua has oscillating thrust as shown in Figure 2–21, it will never reach an equilibrium point and will instead oscillate around a steady-state value.

With thrust as in Figure 2–21, the method used for nonlinear time-invariant system described above cannot be used, and a new method was designed to account for the unsteady equilibrium. Several issues arise when trying to determine an adequate method. First, the depending on the *position* of the paddle, the response to a disturbance will be different. It is therefore important to disturb the system over a full period of oscillation to get every possible paddle configuration. Second, the disturbance must be large enough to be distinguished from the steady-state. Finally,

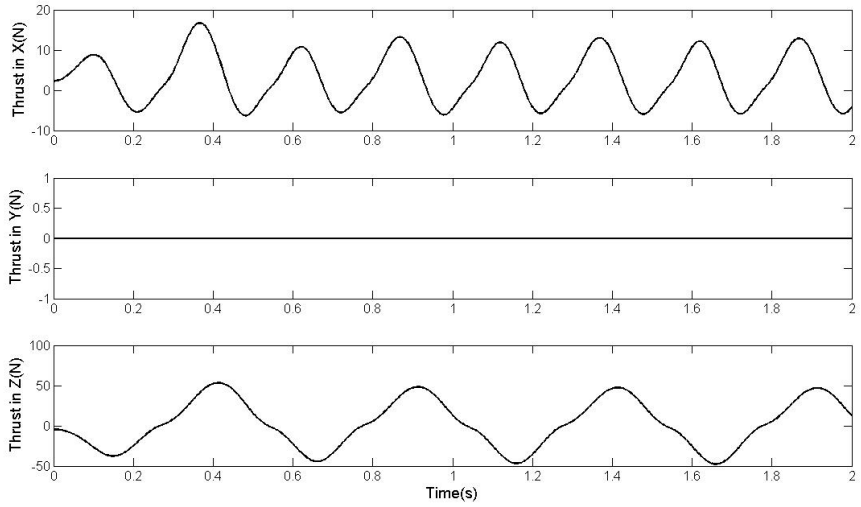


Figure 2–21: Thrust acting on centre of gravity of the robot

the dynamics of the robot changes drastically with velocity and a single operating point will not model the robot accurately.

A disturbance was applied at regular interval over one period of oscillation as can be seen in Figure 2–22. For the specific case shown in Figure 2–22, the nominal steady-state condition is $u = 0.16$m/s with all other velocities equal to 0. A disturbance of 0.016m/s (10% of the nominal speed) is applied to u, 20 times over the paddle cycle. This value was chosen to be large enough to elicit a response, and small enough that Aqua would return to its nominal speed between each excitation. The response of the system in the six degrees of freedom, for the disturbance shown in Figure 2–22, is shown in Figure 2–23. This corresponds to the first column of the state matrix $\mathbf{A}$. We can see that for a disturbance in u as shown in Figure 2–22, there is a response in surge, heave and pitch motion but not in the other three degrees of

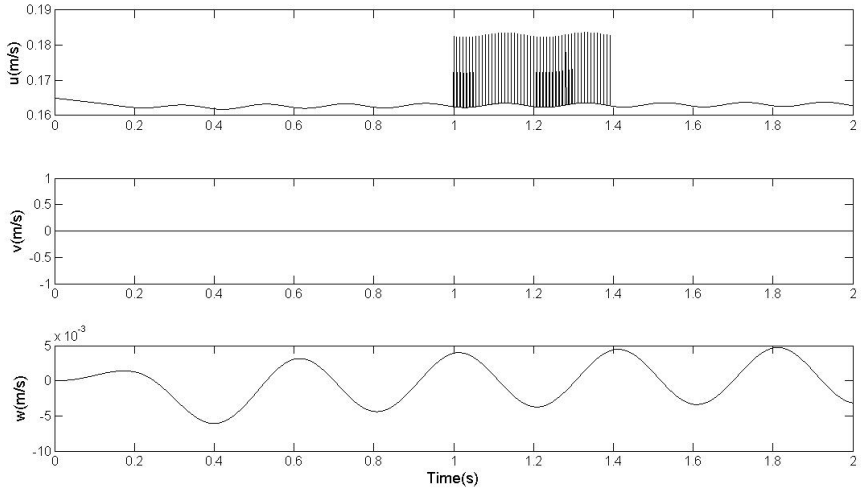


Figure 2–22: Disturbance in the surge velocity

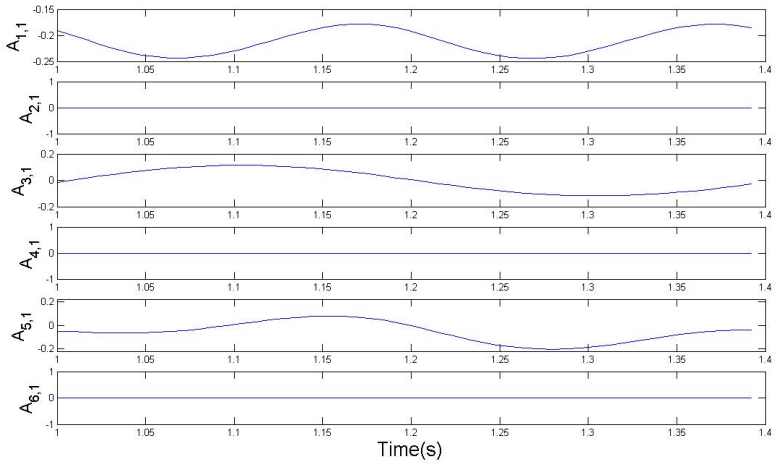


Figure 2–23: Response to a disturbance of 0.016 m/s in u

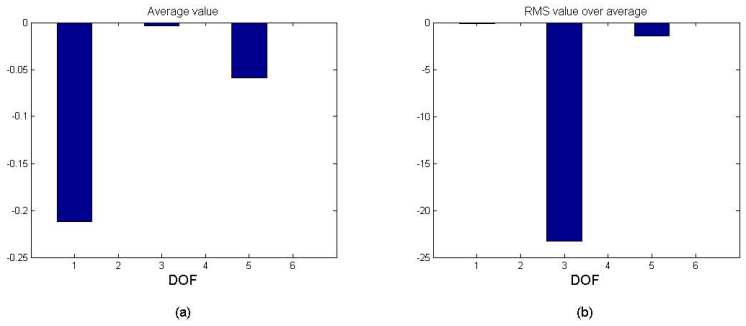


Figure 2–24: (a) Average value of Figure 2–23 (b) RMS over average value of Figure 2–23

freedom as shown in Figure 2–23. The average entries of matrix **A** can be computed from the average response to a disturbance:

$$\bar{A}_{ij} = \frac{\text{average}(\Delta \dot{x}_i)}{\Delta x_j} \tag{2.36}$$

This average given by (2.36) is only one part of the total value of A_{ij}. This total value of A_{ij} is composed of two parts: one constant and one oscillating, $A_{ij} = \bar{A}_{ij} + \grave{A}_{ij}$. These two parts are clear from Figure 2–23. The RMS value of the matrix entries was calculated to evaluate the periodic variation of the dynamics of the robot. Figure 2–24 shows the average value and RMS value of the first column of state-matrix **A** for the disturbance shown in Figure 2–23. We found that the diagonal elements had the largest average values of all entries. Moreover, the constant part is more important than its RMS counterpart except in the case where the average is close to 0, such as $A_{3,1}$ shown in Figure 2–23 and Figure 2–24.

CHAPTER 3

Controllers

The objective of the controllers described in this chapter is to provide trajectory tracking capabilities for the underwater vehicle Aqua. As was mentioned in Section 1.2, many researchers have studied the control of underwater vehicles. Many of them have studied different traditional techniques to control conventional underwater vehicle [35, 36]. Others have tried more advanced methods such as model-based control or robust control [37–39]. Recently, more work has been done on the control of biomimetic underwater vehicle [51–53, 77]. These control techniques worked well on conventional vehicles and we will investigate their performance on a biomimetic vehicle. Moreover, we will apply a control technique known as Floquet that pertains only to time-periodic vehicle, which has not been attempted previously.

The controllers discussed here will output the force vector $\mathbf{f}$ that appeared in the vehicle equation of motion (2.6). This force will have to be generated by the paddles and the mapping from force to paddle motion is done using the method discussed in Section 2.5. Several types of controllers with different properties/features were developed: PID, model-based, adaptive and Floquet. PID controllers are simpler and do not require any knowledge of the system as their output depends only on the

desired and actual state of the system. This controller is discussed in Section 3.2. The model-based controllers that will be discussed in Section 3.3 use the dynamics model developed in Section 2.1 to improve the performance of the PID. Two types of model-based controllers are designed: linearizing and nonlinear. Adaptive controllers are similar to model-based controllers in the sense that they use a model of the system to produce the control input. However, the model is adjusted in real time to account for uncertainties. This controller will be covered in Section 3.4. Finally, the Floquet controller is different from the others because it is the only one that deals explicitely with the time-periodic nature of the vehicle. Section 3.5 discusses the design of a controller using Floquet theory.

The work done in this chapter will be implemented in the dynamics simulation and on the actual vehicle to assess the performance of the different controllers. The process that was used to develop the model-based, Floquet and adaptive controllers could be used for other vehicles if a dynamics model is known. However, the Floquet method can only be applied to time-periodic systems, though most biomimetic vehicles fit this classification.

3.1 Controller input

In this section, we describe the input signals that are sent to the controller, the desired and actual trajectory:

$$\mathbf{v}_d = \begin{bmatrix} u_d & v_d & w_d & p_d & q_d & r_d \end{bmatrix} \tag{3.1}$$

$$\mathbf{s}_d = \begin{bmatrix} x_d & y_d & z_d & \phi_d & \theta_d & \psi_d \end{bmatrix} \tag{3.2}$$

$$\mathbf{v} = \begin{bmatrix} u & v & w & p & q & r \end{bmatrix} \tag{3.3}$$

$$\mathbf{s} = \begin{bmatrix} x & y & z & \phi & \theta & \psi \end{bmatrix} \tag{3.4}$$

where $\mathbf{s}_d$ is the desired position of the vehicle, $\mathbf{v}_d$ is the desired velocity of the vehicle, $\mathbf{v}$ is the actual velocity and $\mathbf{s}$ is the actual position of the vehicle. The actual velocity and position represent the state of the vehicle and are assumed to be provided by sensors on the robot. For the time being, it is assumed that all the entries in (3.3) and (3.4) are precisely sensed; though this assumption will be revisited later. The desired velocity and position are defined by the user and represent the target to track. Each entry of these four vectors represent one degree of freedom.

3.2 PID Controller

We began the controller design with a PID control law because of its simple and well understood nature. A PID controller does not require any knowledge of the system dynamics and its control signal depends only on the error signals. Moreover, it is easy to implement on the actual vehicle and it is a good basis of comparison with more complex controllers. The simple PID controller takes the form:

$$\mathbf{f} = \mathbf{K}_d\mathbf{e}_v + \mathbf{J}_1^{-1}(\mathbf{n}_2)\mathbf{K}_p\mathbf{e}_s + \mathbf{J}_1^{-1}(\mathbf{n}_2)\mathbf{K}_I \int \mathbf{e}_s dt \tag{3.5}$$

$$\mathbf{e}_v = \mathbf{v}_d - \mathbf{v} \tag{3.6}$$

$$\mathbf{e}_s = \mathbf{s}_d - \mathbf{s} \tag{3.7}$$

where $\mathbf{K}_d$, $\mathbf{K}_I$ and $\mathbf{K}_p$ are 6-by-6 diagonal matrices with positive entries, $\mathbf{e}_v$ is the velocity error vector, $\mathbf{e}_s$ is the error in position and $\mathbf{J}_1^{-1}(\mathbf{n}_2)$ is the transformation matrix from the inertial frame to the body frame. Each non-zero entry of a gain matrix $\mathbf{K}$ will affect only one degree of freedom. The equation of motion of the robot is obtained by substituting (3.5) into (2.6), yielding:

$$\begin{aligned} &\mathbf{M}\dot{\mathbf{v}} + \mathbf{C}(\mathbf{v})\mathbf{v} + \mathbf{D}(\mathbf{v})\mathbf{v} + \mathbf{g}(\mathbf{n}_2) + \mathbf{b}(\mathbf{n}_2) \\ &-\mathbf{K}_d\mathbf{e}_v - \mathbf{J}_1^{-1}(\mathbf{n}_2)\mathbf{K}_p\mathbf{e}_s - \mathbf{J}_1^{-1}(\mathbf{n}_2)\mathbf{K}_I \int \mathbf{e}_s dt = \mathbf{0} \end{aligned} \tag{3.8}$$

The PID controller can be separated into two sub-cases: with an integral gain and without. Without any integral gain, we can see from (3.8) that unless $\mathbf{v}_d$ is a zero vector, the controller will not achieve perfect trajectory tracking for which both $\mathbf{e}_v$ and $\mathbf{e}_s$ are $\mathbf{0}$. Equation (3.5) shows that, if $\mathbf{e}_v = \mathbf{0}$ and $\mathbf{e}_s = \mathbf{0}$, the controller would not send any force command and the robot would slow down due to dissipation, causing $\mathbf{e}_v$ to change. With an integral gain, we can expect the controller to provide better trajectory tracking capabilities.

3.3 Model-based controllers

3.3.1 Model-based linearizing controller(MBL)

This controller is more complex than the PID controller and requires a knowledge of the system dynamics, which was discussed in the modeling section of this thesis. The intent of this linearizing controller is to cancel all nonlinear terms of the equation (2.6) of motion by including them in the controller equation. Using this method, we obtain a linear system in which the degrees of freedom are decoupled. The controller takes the form:

$$\begin{aligned}\mathbf{f} &= \mathbf{M}\dot{\mathbf{v}}_d + \mathbf{C}(\mathbf{v})\mathbf{v} + \mathbf{D}(\mathbf{v})\mathbf{v} + \mathbf{g}(\mathbf{n}_2) + \mathbf{b}(\mathbf{n}_2) \\ &+\mathbf{K}_d\mathbf{e}_v + \mathbf{J}_1^{-1}(\mathbf{n}_2)\mathbf{K}_p\mathbf{e}_s + \mathbf{J}_1^{-1}(\mathbf{n}_2)\mathbf{K}_I \int \mathbf{e}_s dt \end{aligned} \tag{3.9}$$

The first five terms of (3.9) are called the model-based terms. They cancel the Coriolis, Hydrodynamic, gravity and buoyancy force of the equation of motion. The PID terms are added to account for uncertainties in the model and to improve the response. In (3.9), it is presumed that the dynamics model is precisely known, so that the quantities $\mathbf{M}$, $\mathbf{D(v)}$, $\mathbf{C(v)}$, $\mathbf{g}$ and $\mathbf{b}$ are precise representations of the true quantities. By substituting (3.9) into (2.6), we then obtain the equation of motion :

$$\mathbf{M}\mathbf{e}_a + \mathbf{K}_d\mathbf{e}_v + \mathbf{J}_1^{-1}(\mathbf{n}_2)\mathbf{K}_p\mathbf{e}_s + \mathbf{J}_1^{-1}(\mathbf{n}_2)\mathbf{K}_I \int \mathbf{e}_s dt = 0 \tag{3.10}$$

$$\mathbf{e}_a = \dot{\mathbf{v}}_d - \dot{\mathbf{v}} \tag{3.11}$$

The system described in (3.10) is linear and time-invariant. It is either a fourth or third order system depending on whether there is an integral gain or not. Moreover, since $\mathbf{M}$, $\mathbf{K}_d$, $\mathbf{K}_P$ and $\mathbf{K}_I$ are all positive diagonal matrices, the system is completely decoupled and each degree of freedom can be treated independently. Therefore, the dynamic system is asymptotically stable. By properly tuning the gains, the response of (3.10) can be set as desired. Moreover, we can see that there is an equilibrium point at $(\mathbf{e}_v, \mathbf{e}_s) = (\mathbf{0}, \mathbf{0})$. However, since the controller force defined by (3.9) will never be achieved exactly because of the oscillating paddles, perfect cancellation of the terms, as shown by (3.10) will never be possible at every instant and will only be possible on average. Therefore, (3.10) will not be truely decoupled in practice. Also, in practice, model uncertainties would also contribute to imperfect coupling.

3.3.2 Model-based nonlinear controller(MBNL)

The purpose of the nonlinear controller is to provide the ideal force that would be required to achieve trajectory tracking. This controller is similar to the MBL controller except that the MBNL uses the *desired* trajectory while the MBL uses the *actual* trajectory to calculate the control force. We expect this controller to provide a better performance since its control force is precisely that needed to track the trajectory. To account for the uncertainties in the model, proportional, integral and derivative gains are also added. The nonlinear controller takes the following form:

$$\mathbf{f} = \mathbf{M}\dot{\mathbf{v}}_d + \mathbf{C}(\mathbf{v}_d)\mathbf{v}_d + \mathbf{D}(\mathbf{v}_d)\mathbf{v}_d + \mathbf{g}(\mathbf{n}_2) + \mathbf{b}(\mathbf{n}_2)$$
$$+\mathbf{K}_d\mathbf{e}_v + \mathbf{J}_1^{-1}(\mathbf{n}_2)\mathbf{K}_p\mathbf{e}_s + \mathbf{J}_1^{-1}(\mathbf{n}_2)\mathbf{K}_I \int \mathbf{e}_s dt \tag{3.12}$$

We can see in (3.12) that the first five terms represent the ideal force to achieve trajectory tracking. In the MBL discussed in Section 3.3.1, the first five terms represented the forces to cancel the nonlinear terms in the equation of motion of the vehicle. Substituting (3.12) into (2.6), the equation of the dynamical system becomes:

$$\mathbf{M}\mathbf{e}_a + (\mathbf{C}(\mathbf{v}_d) - \mathbf{C}(\mathbf{v})\mathbf{v}) + (\mathbf{D}(\mathbf{v}_d) - \mathbf{D}(\mathbf{v})\mathbf{v})$$
$$+\mathbf{K}_d\mathbf{e}_v + \mathbf{J}_1^{-1}(\mathbf{n}_2)\mathbf{K}_p\mathbf{e}_s + \mathbf{J}_1^{-1}(\mathbf{n}_2)\mathbf{K}_I \int \mathbf{e}_s dt \tag{3.13}$$

This system is highly nonlinear and coupled. However, we can still notice the same equilibrium point as for the linearizing controller: $(\mathbf{e}_v, \mathbf{e}_s) = (\mathbf{0}, \mathbf{0})$. We also notice from (3.12) that, as the actual velocity reaches the desired velocity, (3.13) will reduce to (3.10).

3.4 Adaptive controller

Adaptive controllers are types of controllers in which the control law is adjusted to account for the fact that the system parameters might change with time or be poorly known. The design of auto-pilots for high performance aircraft was the main motivation for research on adaptive control because aircraft operates over a wide

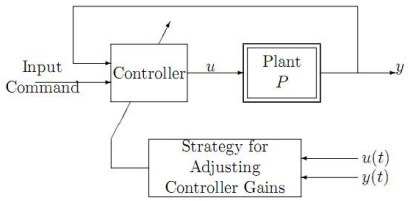


Figure 3–1: Controller structure for adaptive control

range of altitude, weight and speed [78]. Figure 3–1 show the general block diagram used in the implementation of adaptive controllers. The usual feedback structure is present with the addition of an updating mechanism that adapts the controller based on the current state and input.

It is important to note that an adaptive controller is different from a robust controller. A robust controller guarantees that, if the uncertainties remain within given bounds, the control law does not need to be modified to satisfy the performance requirements. In adaptive control, the control law is modified to account for the uncertainties. There exist several approaches to adaptive control such as gain scheduling, adaptive pole placement and model reference adaptive control (MRAC) but we will concentrate on only one approach: model identification adaptive control (MIAC). We decided to apply this method because the adaptive law will output an approximation of the system matrices $\mathbf{A}$ and $\mathbf{B}$ which could be compared to those obtained in the linearization. As a result, the MIAC will provide tracking capabilities as well as information about the dynamics of the system.

3.4.1 MIAC Theory

The objective of MIAC is to estimate the unknown state-space matrices $\mathbf{A}(t)$ and $\mathbf{B}(t)$ and to adjust the control law accordingly. Therefore, the form of the

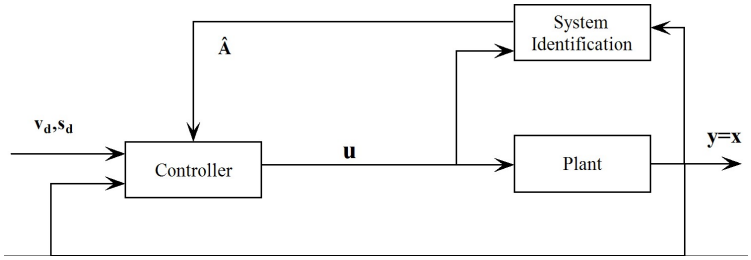


Figure 3–2: controller structure for MIAC

controller will be similar to that for the MBNL controller (Section 3.3.2). However, in the case of the MIAC, the model-based part of the controller will be updated in real time. Moreover, a linear representation of the system dynamics will be used within the controller, rather than a nonlinear one. The general structure of the MIAC controller is shown in Figure 3–2. The adaptive mechanism is determined by the block "system identification" that takes the state of the system and the controller output as inputs. The plant parameters are updated in the identification process. Then, using these parameters, the control law is adjusted.

The rest of this section will now describe the plant model that was used and the process to obtain the system identification and adjustment mechanism. The dynamics of the plant or system is assumed to be governed by a set of linear differential equations:

$$\dot{\mathbf{x}}(t) = \mathbf{A}(t)\mathbf{x}(t) + \mathbf{B}(t)\mathbf{u}(t) \tag{3.14}$$

where $\mathbf{A}(t)$, $\mathbf{B}(t)$ are the state-space matrices, $\mathbf{x}(t)$ is the state vector and $\mathbf{u}(t)$ is the input vector. We assume that $\mathbf{A}(t)$ and $\mathbf{B}(t)$ are unknown or uncertain and that $\mathbf{x}(t)$ and $\mathbf{u}(t)$ are accurately measured. Furthermore, at all times, the eigenvalues

of matrix $\mathbf{A}(t)$ must be to the left of the imaginary axis for this theory to hold (i.e. matrix $\mathbf{A}(t)$ needs to be stable). The objective of the system identification process is to generate state-space matrices as close as possible to $\mathbf{A}(t)$ and $\mathbf{B}(t)$. In order to do this, a parallel model is formed:

$$\dot{\hat{\mathbf{x}}} = \hat{\mathbf{A}}(t)\hat{\mathbf{x}}(t) + \hat{\mathbf{B}}(t)\mathbf{u}(t) \tag{3.15}$$

where $\hat{\mathbf{A}}(t)$, $\hat{\mathbf{B}}(t)$ are estimates of matrices $\mathbf{A}(t)$, $\mathbf{B}(t)$ and $\hat{\mathbf{x}}(t)$ is the estimate of vector $\mathbf{x}(t)$. The input vector $\mathbf{u}(t)$ is the same for both systems and comes from the controller. Then, the estimation error is defined as:

$$\boldsymbol{\epsilon} = \mathbf{x}(t) - \hat{\mathbf{x}}(t) \tag{3.16}$$

The objective is now to find an adaptive law so that $\hat{\mathbf{A}}(t)$ and $\hat{\mathbf{B}}(t)$ converge toward $\mathbf{A}(t)$ and $\mathbf{B}(t)$. First, we find the rate of change of the estimation error by subtracting (3.15) from (3.14):

$$\dot{\boldsymbol{\epsilon}} = \mathbf{A}\boldsymbol{\epsilon} - \tilde{\mathbf{A}}\hat{\mathbf{x}} - \tilde{\mathbf{B}}\mathbf{u} \tag{3.17}$$

where $\tilde{\mathbf{A}} = \hat{\mathbf{A}} - \mathbf{A}$ and $\tilde{\mathbf{B}} = \hat{\mathbf{B}} - \mathbf{B}$. Note that (3.17) has an equilibrium point(i.e. $\dot{\boldsymbol{\epsilon}} = \mathbf{0}$) when all estimated matrices and vector match their exact counterpart. Therefore, the objective of our adaptive law is to get:

$$\dot{\tilde{\mathbf{A}}} = \dot{\hat{\mathbf{A}}} = \mathbf{0}$$
$$\dot{\tilde{\mathbf{B}}} = \dot{\hat{\mathbf{B}}} = \mathbf{0}$$
$$\dot{\boldsymbol{\epsilon}} = \mathbf{0} \tag{3.18}$$

To meet our objective stated in (3.18), we assume that $\dot{\hat{\mathbf{A}}}$ and $\dot{\hat{\mathbf{B}}}$ have the following structure:

$$\dot{\hat{\mathbf{A}}} = \mathbf{F}_1(\boldsymbol{\epsilon}, \mathbf{x}, \hat{\mathbf{x}}, \mathbf{u})$$
$$\dot{\hat{\mathbf{B}}} = \mathbf{F}_2(\boldsymbol{\epsilon}, \mathbf{x}, \hat{\mathbf{x}}, \mathbf{u}) \tag{3.19}$$

where $\mathbf{F}_1$ and $\mathbf{F}_2$ are functions of known signals and they will be chosen so that (3.18) is satisfied. Note that this assumption simply states that $\dot{\hat{\mathbf{A}}}$ and $\dot{\hat{\mathbf{B}}}$ depend on all signals of the system. We will now use a Lyapunov technique to determine appropriate functions $\mathbf{F}_1$ and $\mathbf{F}_2$. The objective is to find a Lyapunov function that would estimate the energy of the system. Then, by proving that the energy decreases with time, we prove that the system errors will converge to zero [62]. Ioannou et al. [78] suggested a Lyapunov function candidate to solve this problem:

$$V(\boldsymbol{\epsilon}, \tilde{\mathbf{A}}, \tilde{\mathbf{B}}) = \boldsymbol{\epsilon}^T \mathbf{P} \boldsymbol{\epsilon} + \text{tr}\left(\frac{\tilde{\mathbf{A}}^T \mathbf{P} \tilde{\mathbf{A}}}{\boldsymbol{\Gamma_1}}\right) + \text{tr}\left(\frac{\tilde{\mathbf{B}}^T \mathbf{P} \tilde{\mathbf{B}}}{\boldsymbol{\Gamma_2}}\right) \tag{3.20}$$

where V is the Lyapunov function, $\boldsymbol{\Gamma_1}$, $\boldsymbol{\Gamma_2}$ are positive constants diagonal matrices and $\mathbf{P} = \mathbf{P}^T$ is chosen to satisfy the following Lyapunov equation:

$$\mathbf{PA} + \mathbf{A}^T\mathbf{P} = -\mathbf{I} \tag{3.21}$$

It is important to note that the existence of $\mathbf{P}$ is guaranteed by the stability of $\mathbf{A}$. According to Lyapunov stability theory, the system will be stable if the rate of change of the Lyapunov function candidate (3.20) is always negative. This function being an evaluation of the energy of the system, this stability criterion simply implies that the energy of the system will decrease with time. The time derivative of (3.20) is given by:

$$\dot{V} = \dot{\boldsymbol{\epsilon}}^T\mathbf{P}\boldsymbol{\epsilon} + \boldsymbol{\epsilon}^T\mathbf{P}\dot{\boldsymbol{\epsilon}} + \text{tr}\left(\frac{\dot{\tilde{\mathbf{A}}}\mathbf{P}\tilde{\mathbf{A}}}{\boldsymbol{\Gamma_1}} + \frac{\tilde{\mathbf{A}}\mathbf{P}\dot{\tilde{\mathbf{A}}}}{\boldsymbol{\Gamma_1}}\right) + \text{tr}\left(\frac{\dot{\tilde{\mathbf{B}}}\mathbf{P}\tilde{\mathbf{B}}}{\boldsymbol{\Gamma_2}} + \frac{\tilde{\mathbf{B}}\mathbf{P}\dot{\tilde{\mathbf{B}}}}{\boldsymbol{\Gamma_2}}\right) \tag{3.22}$$

This equation can be rearranged by replacing $\dot{\boldsymbol{\epsilon}}, \dot{\tilde{\mathbf{A}}}, \dot{\tilde{\mathbf{B}}}$ in (3.22) with the expressions given by (3.17) and (3.19):

$$\dot{V} = \boldsymbol{\epsilon}^T\left(\mathbf{PA} + \mathbf{A}^T\mathbf{P}\right)\boldsymbol{\epsilon} - 2\boldsymbol{\epsilon}^T\mathbf{P}\tilde{\mathbf{A}}\hat{\mathbf{x}} - 2\boldsymbol{\epsilon}^T\mathbf{P}\tilde{\mathbf{B}}\mathbf{u} + \text{tr}\left(2\frac{\tilde{\mathbf{A}}^T\mathbf{P}F_1}{\boldsymbol{\Gamma_1}} + 2\frac{\tilde{\mathbf{B}}^T\mathbf{P}F_2}{\boldsymbol{\Gamma_2}}\right) \tag{3.23}$$

Then, using the result in (3.21) and using some matrix properties discussed in [62, 78], we can reduce (3.23) to:

$$\dot{V} = -\boldsymbol{\epsilon}^T\boldsymbol{\epsilon} + 2\text{tr}\left(\frac{\tilde{\mathbf{A}}^T\mathbf{P}F_1}{\boldsymbol{\Gamma_1}} + \frac{\tilde{\mathbf{B}}^T\mathbf{P}F_2}{\boldsymbol{\Gamma_2}} - \tilde{\mathbf{A}}^T\mathbf{P}\boldsymbol{\epsilon}\hat{\mathbf{x}}^T - \tilde{\mathbf{B}}^T\mathbf{P}\boldsymbol{\epsilon}\mathbf{u}^T\right) \tag{3.24}$$

Our objective is now to find functions F_1 and F_2 that will guarantee that (3.24) is negative. Although there are many possibilites, one simple solution that makes the second term of (3.24) vanish is:

$$F_1 = \dot{\hat{\mathbf{A}}} = \mathbf{\Gamma_1} \boldsymbol{\epsilon} \hat{\mathbf{x}}^T \quad \text{and} \quad F_2 = \dot{\hat{\mathbf{B}}} = \mathbf{\Gamma_2} \boldsymbol{\epsilon} \mathbf{u}^T \tag{3.25}$$

In (3.25), $\boldsymbol{\epsilon}, \hat{\mathbf{x}}$ and $\mathbf{u}$ are all measured quantities. Therefore, only $\mathbf{\Gamma_1}$ and $\mathbf{\Gamma_2}$, the adaptive gain matrices, can be adjusted to improve the convergence of F_1 and F_2. Eq. (3.25) constitutes the adaptive control law used to update the estimated state matrices $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ so that they converge to their exact values. The "system identification" process in Figure 3–2 thus consists of integrating eq. (3.15) to find $\hat{\mathbf{x}}$; finding $\boldsymbol{\epsilon}$ using (3.16); and finding $\dot{\hat{\mathbf{A}}}$ and $\dot{\hat{\mathbf{B}}}$ from (3.25); and integrating these to find $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$.

3.5 Floquet Controller

Floquet theory is a branch of the theory of ordinary differential equation that allows the solution of time-periodic problems. It is named after Gaston Floquet, a French mathematician, and its main result is a coordinate change that transforms a time-periodic system into a linear time-invariant system. It can also be used to design controllers for time-periodic systems. Aqua uses oscillating paddles instead of thrusters to propel itself through water. Due to this particular propulsion method, the thrust is periodic with a pre-determined constant period. In the reverse model presented in Section 2.5, the period is allowed to vary. We typically select the weights in (2.30) to keep the period constant. However, if it is necessary to modify

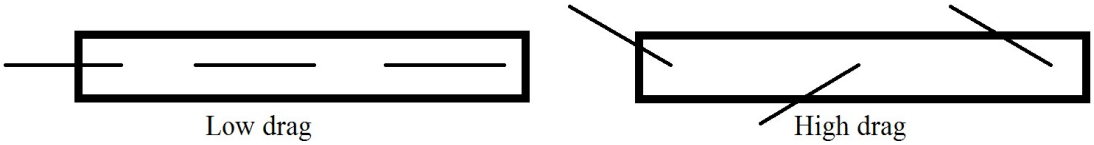


Figure 3–3: Side view of the vehicle with different paddle configurations. a-b

the period to produce more thrust, we can design multiple Floquet controllers with different period and gain scheduled. This was not necessary in this work. Moreover, as it can be seen in Figure 3–3 the hydrodynamic profile of the vehicle changes as the paddles move, which changes the response of the vehicle to control inputs and disturbances. In Figure 3–3a, if the vehicle is moving horizontally the paddles are all aligned with the vehicle motion and therefore result in very low drag. On the other hand, on Figure 3–3b, the paddles are oblique to the flow and generate more drag. This variation in paddle orientation is periodic in nature due to the periodic paddle motion.

As a result of the periodic nature of the system, Floquet-Lyapunov theory seems a good candidate for the development of controllers for the Aqua vehicle. A linear model for the vehicle can be represented as follows:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}(t)\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) &= \mathbf{C}(t)\mathbf{x}(t) \end{aligned} \tag{3.26}$$

where $\mathbf{A}(t)$, $\mathbf{B}(t)$ and $\mathbf{C}(t)$ are the time-periodic state-space time-periodic linear matrices with period P, $\mathbf{x}(t)$ is the state vector and $\mathbf{u}(t)$ is the input vector. The linear model presented in (3.26) is valid for all systems and not specific to our vehicle. Aqua dynamics are nonlinear and must be linearized in order to use Floquet theory. It is important to note that matrix $\mathbf{A}$ depends on time and on velocity. During the

linearization process discussed in Section 2.6, the vehicle was linearized for different velocities. As a result, matrix $\mathbf{A}(t)$ presented in (3.26) is exact only for the particular velocity used in the linearization process. In the case where the velocity of the vehicle was to change significantly during the operation, gain scheduling should be used. The solution to the homogenous equation, with no input $\mathbf{u}(t)$ is given by:

$$\mathbf{x}(t) = \mathbf{\Phi}(t, t_0)\mathbf{x}(t_0) \tag{3.27}$$

where $\mathbf{\Phi}(t, t_0)$ is the state-transition matrix and must satisfy the following differential equation:

$$\frac{d\mathbf{\Phi}(t, t_0)}{dt} = \mathbf{A}(t)\Phi(t, t_0) \tag{3.28}$$

The results presented in (3.26)-(3.28) are analogous to those of conventional linear theory, with the exception that $\mathbf{A}(t)$, $\mathbf{B}(t)$ and $\mathbf{\Phi}(t, t_0)$ are periodic instead of constant. The main result of Floquet theory is that the state-transition matrix can be factored into two matrices $\mathbf{F}$ and $\mathbf{J}$:

$$\mathbf{\Phi}(t, t_0) = \mathbf{F}(t)e^{\mathbf{J}t}\mathbf{F}^{-1}(t_0) \tag{3.29}$$

where $\mathbf{J}$ is a constant matrix and $\mathbf{F}(t)$ a time-varying matrix. These two matrices are often called the Floquet factors. There are several methods to obtain the matrices $\mathbf{F}(t)$ and $\mathbf{J}$ and it is important to note that there are many possible solutions to this problem. However, the eigenvalues of $\mathbf{J}$, called the Poincaré exponents (ω_i), are unique. They give information about the stability of the system as the eigenvalues of

the state matrix **A** would do for a time-invariant system. Solving a Floquet problem for all time requires determining the constant matrix **J** and the time-periodic matrix $\mathbf{F}(t)$ for one full period. We now discuss two methods for obtaining the Floquet factors.

3.5.1 Floquet factors using the Eigenvalues and Eigenvectors

This section describes a technique to compute the Floquet factors from the state-transition matrix found in Section 3.5.2. It is derived from direct observation of (3.29) and its similarity to an eigenvalue problem. In the current form of the equation, there are two unknowns (**J** and $\mathbf{F}(t)$) and it is therefore impossible to solve the eigenvalue problem. We evaluate (3.29) after one period in order to circumvent this problem. If we assume that $\mathbf{F}(t)$ is periodic with period P then $\mathbf{F}(P + t_0) = \mathbf{F}(t_0)$, and evaluation of (3.29) at t=P gives us:

$$\mathbf{\Phi}(P, 0) = \mathbf{F}(0)e^{\mathbf{J}P}\mathbf{F}^{-1}(0) \tag{3.30}$$

where $\mathbf{\Phi}(P, 0)$, the state-transition matrix after one full period, is called the monodromy matrix. Because the system is time-periodic we have replaced t_0 by 0 in (3.30) without loss of generality. The monodromy matrix is obtained using the results shown in the previous section, Section 3.5.2.

From (3.30), we can see that $\mathbf{F}(0)$ is the eigenvector matrix of the monodromy matrix. Moreover, **J** will be diagonal with the Poincaré exponents as its entries:

$$\mathbf{J} = \begin{bmatrix} \omega_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \omega_n \end{bmatrix} \tag{3.31}$$

where each ω_i represents one Poincaré exponent of the system. They are the time-periodic equivalents of the eigenvalues for time-invariant system. The Poincaré exponents are related to the eigenvalues of the monodromy matrix in the following way:

$$\omega_i = \frac{\ln(\Theta_i)}{P} \tag{3.32}$$

where Θ_i represents the i-th eigenvalues of the monodromy matrix. Since $\mathbf{F}(t)$ is time-periodic and therefore bounded, the stability of the system depends solely on the Poincaré exponents and those are now known from (3.32). At this stage, we have all the knowledge to assess the stability of the system, but not enough to design controllers. As it will be shown in a later section (Section 3.5.3), we now need to compute matrix $\mathbf{F}(t)$ for all time in order to design controllers. With the matrix $\mathbf{J}$ known, $\mathbf{F}(t)$ can be obtained for all time by rearranging (3.29):

$$\mathbf{F}(t) = \mathbf{\Phi}(t, 0)\mathbf{F}(0)e^{-\mathbf{J}P} \tag{3.33}$$

It is important to note that there is no guarantee that the Floquet factors will be real using this method. Real Floquet factors are desirable because the control gain matrix is a function of the Floquet factors. An appropriate rearrangement exists that makes both matrices real and leaves the previous formula unaltered [58]. The

result is that the Floquet factors are real and eq. (3.29)-(3.30) still hold. For every complex conjugate pair $\omega_{i1,2} = a \pm bi$, a transformation is applied:

$$\mathbf{J} = \begin{bmatrix} \omega_{i1} & 0 \\ 0 & \omega_{i2} \end{bmatrix} = \begin{bmatrix} a+bi & 0 \\ 0 & a-bi \end{bmatrix} = \begin{bmatrix} a & b \\ -b & a \end{bmatrix} \tag{3.34}$$

With the transformation shown in (3.34) applied to every complex conjugate pair, matrix **J** becomes real. However, it is not diagonal anymore, but at will be discussed later, this does not affect the final result.

3.5.2 Computation of state-transition matrix

The state-transition matrix is used to obtain the general solution of a linear dynamics system governed by eq. (3.26). The state-transition matrix allows calculation of the state of the system at time t from the state at time t_0 as shown by (3.27). In our case, the state-transition matrix is used to obtain the Floquet factors in Section 3.5.1. There are several methods that have been developed to find an analytical solution to a problem of the form presented in (3.26). In the case of a constant matrix **A**, the state-transition matrix is simply:

$$\Phi(t, 0) = e^{\mathbf{A}t} \tag{3.35}$$

Equation (3.35) does not usually apply for time-varying system but it does hold only if the following commutativity condition is satisfied [62]:

$$\mathbf{A}(t) \int_0^t \mathbf{A}(\sigma) d\sigma = \int_0^t \mathbf{A}(\sigma) d\sigma \mathbf{A}(t) \tag{3.36}$$

where σ is a variable of integration. When (3.36) is not satisfied, a Peano-Baker series can be used to find the state-transition matrix of the system [62]:

$$\Phi(t,0) = \mathbf{I} + \int_0^t \mathbf{A}(\sigma_1)d\sigma_1 + \int_0^t \mathbf{A}(\sigma_1)\int_0^{\sigma_1} \mathbf{A}(\sigma_2)d\sigma_2 d\sigma_1 \cdots \tag{3.37}$$

It is obvious that even though (3.37) can provide an analytical solution, it will be computationally intensive and complicated. In most cases, it is simpler to find a numerical approximation to the state-transition matrix rather than finding an exact solution. Many algorithms exist to do this. They usually rely on finding the solution $\mathbf{x}(t)$ and deducing the state-transition matrix from it. There are two main computational approaches to this problem: n-pass and single-pass schemes [63].

Using the n-pass approach, eq. (3.26) with $\mathbf{u}(t) = \mathbf{0}$ is solved n times for n initial conditions consisting of the n columns of the identity matrix. As a result, for each pass, we compute one column of the state-transition matrix. For example, the first column of $\Phi(t,0)$ would be deduced from the solution $\mathbf{x}(t)$ for $\mathbf{x}_0(t) = \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix}^T$:

$$\begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{bmatrix} = \begin{bmatrix} \Phi_{11}(t) & \cdots \\ \Phi_{21}(t) & \cdots \\ \vdots & \\ & \ddots \\ \Phi_{n1}(t) & \cdots \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ \vdots 0 \\ 0 \end{bmatrix} \Rightarrow \begin{bmatrix} \Phi_{11}(t) \\ \Phi_{21}(t) \\ \vdots \\ \Phi_{n1}(t) \end{bmatrix} = \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{bmatrix} \tag{3.38}$$

Using this approach, (3.26) needs to be solved several times and this significantly increases the computing time of the algorithm. Cai et al. developed a method to calculate $\mathbf{x}(t)$ that allows finding the state-transition matrix in a single pass. Effectively, their method solves a system of the form:

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{A}_0\mathbf{x}(t) + (\mathbf{A}(t) - \mathbf{A}_0)\mathbf{x}(t) \tag{3.39}$$

where $\mathbf{A}_0$ is the constant part of $\mathbf{A}(t)$ and $(\mathbf{A}(t) - \mathbf{A}_0)$ can be expressed as:

$$(\mathbf{A}(t) - \mathbf{A}_0) = \sum(\mathbf{D}_c\sin(ct) + \mathbf{B}_c\cos(ct)) \tag{3.40}$$

where $\mathbf{D}_c$ and $\mathbf{B}_c$ are constant coefficient matrices and c is the frequencies of the system. In our case, the system oscillates at a single frequency (the paddle frequency) but in other situations, there could be more than one frequency, hence the summation sign. Then, after some manipulation, Cai et al. developed an equation to find the vector $\mathbf{x}(t)$ from the previous time step:

$$\mathbf{x}_{k+1} = \mathbf{H}_{k+1}\mathbf{x}_k \tag{3.41}$$

where $\mathbf{H}_{k+1}$ is a matrix that updates the state $\mathbf{x}_k$ into $\mathbf{x}_{k+1}$. At this point, it is important to note that $\mathbf{H}_{k+1}$ is in fact a state-transition matrix from one state to the next. Also, it is important to note that (3.41) is discrete and not continuous. Moreover, remembering the composition rule [62], we can find any state $\mathbf{x}_k$ as a function of the initial state. For example:

$$\mathbf{x}_2 = \mathbf{H}_2\mathbf{x}_1 = \mathbf{H}_2\mathbf{H}_1\mathbf{x}_0 \tag{3.42}$$

As a result, the state-transition matrix can be obtained from $\mathbf{H}$ at any time t:

$$\Phi(n\Delta t, 0) = \tag{3.43}$$
$$\Phi(n\Delta t, (n-1)\Delta t)\Phi((n-1)\Delta t, (n-2)\Delta t \ldots \Phi(\Delta t, 0) = \prod_{n=1}^{k} \mathbf{H}_{k-n+1}$$

Using this method, the state-transition matrix can be obtained using a single pass. The monodromy matrix can then found from (3.44):

$$N = \frac{T}{\Delta t} \rightarrow \Phi(T, 0) = \prod_{n=1}^{N} \mathbf{H}_{k-n+1} \tag{3.44}$$

In equations (3.41) to (3.44), the discrete form was preferred to the continuous form because the algorithm developped by Cai et al. uses it. By choosing a small enough step size, the discrete state-transition matrix given by (3.44) can be considered continuous.

3.5.3 Control law

In the previous sections, we have shown how to obtain the state-transition matrix and the Floquet factors for a system in the form of (3.26). We now have all the tools to design control laws for our time-periodic system. We chose to use $\mathbf{F}(t)$ and $\mathbf{J}$ found using the first method with appropriate rearrangement to obtain real Floquet factors because $\mathbf{F}(t)$ is P-periodic like $\mathbf{A}(t)$ and because $\mathbf{J}$ has the advantage of being

in a quasi-diagonal form. We first introduce another variable that we call the modal variable:

$$\boldsymbol{\eta}(t) = \mathbf{F}(t)^{-1}\mathbf{x}(t) \tag{3.45}$$

As we see from (3.45), the modal variable is simply a transformation of the state variable $\mathbf{x}(t)$. We can then rewrite (3.26) in term of the modal variable:

$$\dot{\boldsymbol{\eta}}(t) = \mathbf{J}\boldsymbol{\eta}(t) + \mathbf{F}(t)^{-1}\mathbf{B}(t)\mathbf{u}(t) \tag{3.46}$$

This means that the matrix $\mathbf{F}(t)$ reduces the time-periodic system of (3.26) into a constant-coefficient system. Moreover, as was mentioned in Section 3.5.1 the stability of the system depends only on the eigenvalues of $\mathbf{J}$, the Poincaré exponents. We will use a simple proportional feedback for our control law:

$$\mathbf{u}(t) = \mathbf{K}(t)(\boldsymbol{\eta}_d(t) - \boldsymbol{\eta}(t)) \tag{3.47}$$

where the subscript d signifies that this is a desired value. Using this control law, the system described by (3.46) becomes:

$$\dot{\boldsymbol{\eta}}(t) = \left[\mathbf{J} - \mathbf{F}^{-1}(t)\mathbf{B}(t)\mathbf{K}(t)\right]\boldsymbol{\eta}(t) + \mathbf{F}^{-1}(t)\mathbf{B}(t)\mathbf{K}(t)\boldsymbol{\eta}_d(t) \tag{3.48}$$

The first part of (3.48) is the controlled natural response of the system. The eigenvalues of $\mathbf{J} - \mathbf{F}^{-1}(t)\mathbf{B}(t)\mathbf{k}(t)$ are different from those of $\mathbf{J}$ in (3.46) and are determined by the gain matrix $\mathbf{K}(t)$. By adjusting the gain matrix, the eigenvalues can be selected to improve the response of the system. The second part of (3.48)

determines how the system will track a desired trajectory. Now, we can define the controllability matrix:

$$\mathbf{G}(t) = \mathbf{F}^{-1}(t)\mathbf{B}(t) \tag{3.49}$$

In our case, since we only have direct control over the velocity states, the controllability matrix will have a maximum rank of 6. Then, the matrix of Poincaré exponents of the controlled system is given by:

$$\mathbf{J}' = \mathbf{J} - \mathbf{G}(t)\mathbf{K}(t) \tag{3.50}$$

The control problem here becomes finding the appropriate gain matrix $\mathbf{K}(t)$ that will place the poles at the desired location. It is also important to note that since $\mathbf{G}(t)$ has a rank of 6, only 6 eigenvalues can be selected independently. Calico et al. proposed a method to determine which eigenvalues to modify [58]. The $\mathbf{J}$ matrix is partitioned into modes to control and modes to ignore: $\mathbf{J}_c$ and $\mathbf{J}_u$. A similar partition is done for $\mathbf{F}(t)$, $\boldsymbol{\eta}(t)$ and $\mathbf{G}(t)$. Then, the first part of (3.48) becomes [58]:

$$\begin{bmatrix} \dot{\boldsymbol{\eta}}_c(t) \\ \dot{\boldsymbol{\eta}}_i(t) \end{bmatrix} = \begin{bmatrix} \mathbf{J}_c + \mathbf{G}_c(t)\mathbf{K}_c(t) & \mathbf{G}_c(t)\mathbf{K}_i(t) \\ \mathbf{G}_i(t)\mathbf{K}_c(t) & \mathbf{J}_i + \mathbf{G}_i(t)\mathbf{K}_i(t) \end{bmatrix} \begin{bmatrix} \boldsymbol{\eta}_c(t) \\ \boldsymbol{\eta}_i(t) \end{bmatrix} \tag{3.51}$$

By setting $\mathbf{K}_i(t)$ to $\mathbf{0}$, the controlled modes can be decoupled from the uncontrolled ones. Furthermore, this leaves the eigenvalues of the ignored modes unchanged. Rewriting (3.50) for the controlled modes:

$$\mathbf{J}'_c = \mathbf{J}_c + \mathbf{G}_c(t)\mathbf{K}_c(t) \tag{3.52}$$

where $\mathbf{J}'_c$ is the desired eigenvalues matrix. With $\mathbf{G}_c(t)$ being a square matrix, we can find the required control gains to place the poles at the desired location:

$$\mathbf{K}_c(t) = \mathbf{G}_c^{-1}(t)\left[\mathbf{J}'_c - \mathbf{J}_c\right] \tag{3.53}$$

This is valid only if $\mathbf{G}_c(t)$is of full rank. Then, the full gain matrix is given by:

$$\mathbf{K}(t) = \begin{bmatrix} \mathbf{K}_c(t) & \mathbf{0} \end{bmatrix} \tag{3.54}$$

Finally, the gain matrix in the $\mathbf{x}(t)$ domain can be obtained by post-multiplying (3.53) by $\mathbf{F}^{-1}(t)$.

CHAPTER 4

Results

Chapter 3 described five controllers that were developed to provide trajectory tracking capabilities to the underwater vehicle Aqua. The form of the controllers was presented as well as the resulting equation of motion of the vehicle. The controllers output the forces to be provided by the paddles. A method to transform that force into desired paddle motion was also described in Section 2.5.

This chapter describes the evaluation of these trajectory tracking controllers on Aqua. Section 4.1 discusses how the controllers were implemented in the simulation and on the actual vehicle. The trajectories that were tested are described in Section 4.2. The trajectories are first tested on the dynamics simulation to tune the PID gains in Section 4.3. Finally, an experiment to test the controllers in a dynamics environment is described in Section 4.4. With good results in these maneuvers, it would then be possible to test the performance of the vehicle in high performance maneuvers.

4.1 Implementation

This section describes how the controllers were implemented in the dynamics simulation and on the actual vehicle. The process was different in the simulation

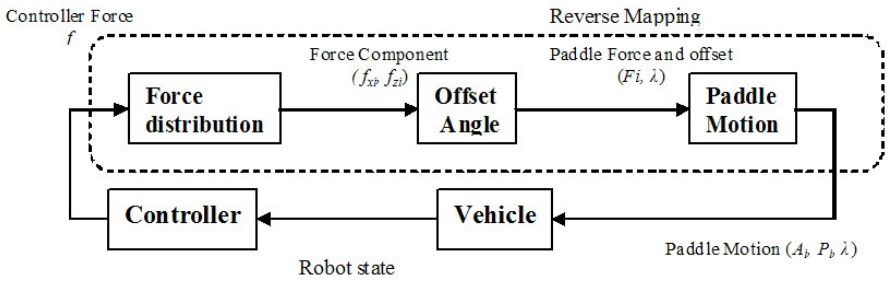


Figure 4–1: Control loop used in the simulation

and in the experiment because we do not have access to the same sensor information and the programming language is different in the two environments. We also discuss how the adaptive and Floquet theory was applied to the vehicle.

4.1.1 Implementation in the Simulation

The simulation uses the MATLAB programming language. As a result, the controller was defined as a callable function where the input was defined as in Section 3.1. The five controllers were programmed within that function and the user could select which controller to use. Figure 4–1 shows the control loop used in the simulation. The controller takes the state of the vehicle as input and outputs a force. Then, the reverse mapping developed in Section 2.5 is used to determine the appropriate paddle motion (A,P,λ).

4.1.2 Implementation on the vehicle

The vehicle is controlled via a standard laptop with a *Linux* operating system. The user controls the vehicle with a *gamepad* that has 2 joysticks. The left joystick controls the pitch and roll motion while the right one controls the yaw motion. There is a graphical user interface(GUI) on the laptop where the user can see the joystick

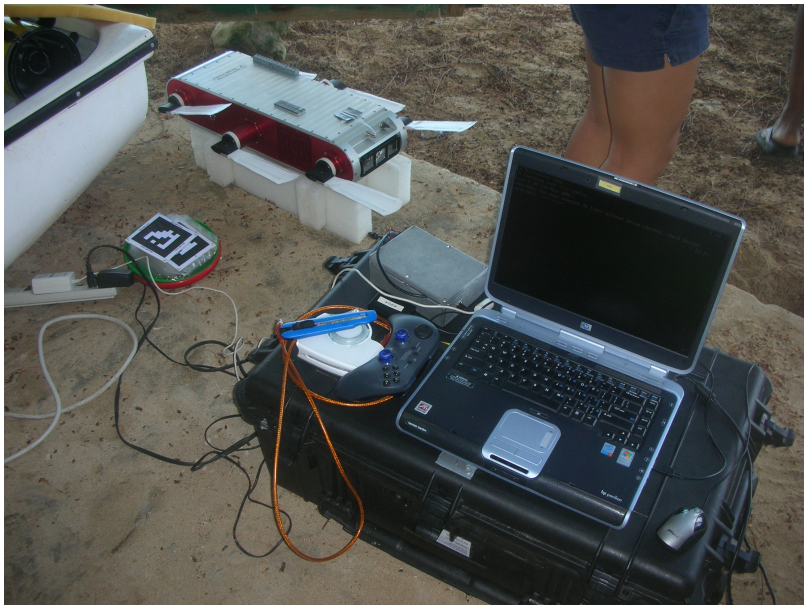


Figure 4–2: Experimental setup

commands as well as the Euler angles. The GUI also gives information about the energy consumption of the vehicle or any failure occuring. The video from the camera at the front of the vehicle is displayed on a separate monitor. On the GUI, the user also has the option to choose which swimming gait to use and to turn on/off the controller. The setup used during a shore experiment is shown in Figure 4–2. It is similar to the one used during boat experiments.

The vehicle control software uses the QNix programming language. The controller was programmed as an object that was called from the GUI. The input of the controller object is the sampling rate, the Euler angles, Euler rates and the depth of the vehicle. For this experiment, the desired trajectory was defined within the controller itself. In the future, the desired trajectory will be an external input. The user could decide which controller to use directly from the GUI. The reference trajectory

was also selectable using an option on the GUI. The controllers were programmed using the expressions discussed in Section 3. The only difference was for the controllers that rely on states that were not available for measurement on the vehicle. This was the case for the MBL and MBNL. In these two cases, the unavailable states were the surge, sway and heave velocity. They were replaced by their estimated value for the maneuver based on the amplitude and period of oscillation:

$$\begin{bmatrix} u & v & w \end{bmatrix} \frac{m}{s} = \begin{bmatrix} 0.5 & 0 & 0 \end{bmatrix} \frac{m}{s} \tag{4.1}$$

It is important to note that, except for the control gains that can be changed online, all other changes in the controllers require recompiling the code. Due to the low speed CPU onboard the vehicle, recompiling is time consuming and cannot be done during an experimental session. Making changes in the code is only possible between sessions.

4.1.3 MIAC: Application

In this section, we describe how the MIAC approach was applied to Aqua. This includes how we initialize matrices $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$, the controller form, the stabilization of the system and the tuning of the adaptive gain matrices $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$.

The initial values for the system matrices $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$, denoted as $\hat{\mathbf{A}}_0$ and $\hat{\mathbf{B}}_0$, are obtained from the linearized model. This initial guess is used as the starting point for the adaptive law. It is important to note that since the system is highly nonlinear, it cannot be modeled perfectly using the linear differential equation given by (3.14). However, using the adaptive law described in (3.25), $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ will be adjusted in real time to account for nonlinearities.

As was mentioned in Section 3.4.1, the linear system must be stable for the theory to hold. However, even though the nonlinear model is stable, the linear model is unstable in yaw and a stabilizing controller $\mathbf{u}_s = \mathbf{K}_s\mathbf{x}$ needs to be added. This controller acts as a stability augmentation system(SAS) and the adaptive controller is based on the resulting augmented system. The augmented state matrix takes this form:

$$\mathbf{A}' = (\mathbf{A} + \mathbf{B}\mathbf{K}_s) \tag{4.2}$$

where $\mathbf{K}_s$ is a stabilizing control gain matrix. The yaw rate is the only unstable degree of freedom and therefore, we chose to add a gain only for the yaw rate (the (6,6) element of $\mathbf{K}_s$. This way, all other degrees of freedom are left unaltered and the effect of the adaptive law will be more apparent. Note that this stabilizing controller is applied to the nonlinear model.

Next, the adaptive gain matrices $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$ must be determined in order to solve (3.25). Instead of using a trial and error procedure that could be time consuming, we chose to use an offline optimization algorithm. The simulation was run using a predetermined sinusoidal input vector $\mathbf{u}$. We constructed the input vector so that it had components in all 6 degrees of freedom so that all states of the vehicle were excited. Then, the objective of the optimizer was to find matrices $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$ so that the norm of the error vector ϵ converged to 0 as quickly as possible with a maximum overshoot of 5%. This structure is shown in Figure 4–3. The input $\mathbf{u}$ is fed to both the linear model(considered perfect for this optimization, eq.(3.14), but with the state matrix of eq. (4.2) and to the uncertain model eq. (3.15). Then,

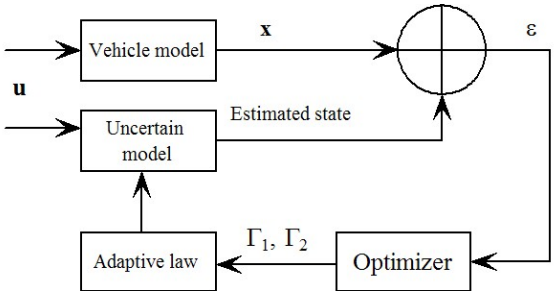


Figure 4–3: Block diagram of the adaptive gain optimization

the error vector $\boldsymbol{\epsilon}$ was sent to the optimizer which then adjusted the adaptive gains accordingly.

A gradient based method was used to solve the problem using the *fminunc* function available in MATLAB. The dynamics simulation was set as a callable function with the adaptive gain matrices as input, and *fminunc* then outputs the optimal $\boldsymbol{\Gamma}_1$ and $\boldsymbol{\Gamma}_2$. Figure 4–4 shows how the (4,4) entry of matrix $\hat{\mathbf{A}}$ changed with time in our simulation for different values of the roll adaptive gain. This entry of $\hat{\mathbf{A}}$ is the main one pertaining to the roll rate. We see that for a small $\boldsymbol{\Gamma}_{roll}$, it takes a long time to settle and when the gain is too large, the overshoot becomes more significant. The optimization found that the optimal gain was $\Gamma_{roll} = 262$. We also note that it takes around 2 seconds to settle. It is also important to note that this process is done offline and that the adaptive gain matrices will remain constant during the simulation and experiments.

We will now describe the control law used, which is similar to that used in the model-based controller discussed in [79, 80]. Matrices $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ were partitioned into upper and lower part. The upper part relates to the velocity states and the lower part

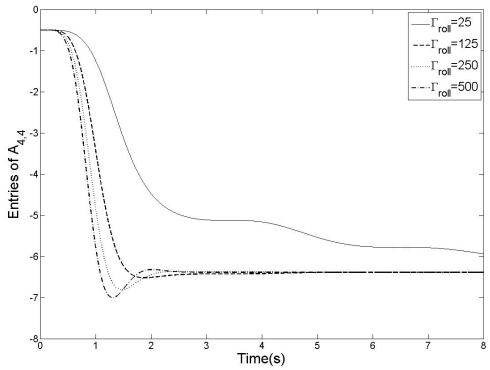


Figure 4–4: Entry of $\mathbf{A}_{4,4}$ with time for different adaptive gain

the displacement states. The state vector $\mathbf{x}$ was also partitioned into the velocity vector $\mathbf{v}$ and the position vector $\mathbf{s}$.

$$\hat{\mathbf{A}} = \begin{bmatrix} \hat{\mathbf{A}}_{top}^{6\times6} & \mathbf{0} \\ \hat{\mathbf{A}}_{bot}^{6\times6} & \mathbf{0} \end{bmatrix} \quad \mathbf{x} = \begin{bmatrix} \mathbf{v}^{6\times1} \\ \mathbf{s}^{6\times1} \end{bmatrix} \quad \hat{\mathbf{B}} = \begin{bmatrix} \hat{\mathbf{B}}_{top}^{6\times6} \\ \hat{\mathbf{B}}_{bot}^{6\times6} \end{bmatrix} \tag{4.3}$$

We note that the right side of $\hat{\mathbf{A}}$ is $\mathbf{0}$ because $\dot{\mathbf{x}}$ does not depend $\mathbf{s}$ and that $\hat{\mathbf{A}}_{bot}$ is the identity matrix. The partitioning was done because the objective of the adaptive part of the controller is to cancel the Coriolis and hydrodynamic drag forces of (2.6). By combining (2.6), (3.15) and (4.3) and rearranging the equations we obtain a relationship between the Coriolis and hydrodynamic drag matrices and the linear state-space matrices:

$$\hat{\mathbf{B}}_{top}^{-1}\hat{\mathbf{A}}_{top} = -\left(\mathbf{C}+\mathbf{D}\right) \tag{4.4}$$

where $\mathbf{C}$ and $\mathbf{D}$ are the linearized forms of $\mathbf{C(v)}$ and $\mathbf{D(v)}$. Then, we can define the control force generated by the adaptive controller:

$$\mathbf{f} = -\hat{\mathbf{B}}_{top}^{-1}\hat{\mathbf{A}}_{top}\mathbf{v} + \mathbf{K}_d\mathbf{e}_v + \mathbf{J}_1^{-1}(\mathbf{n}_2)\mathbf{K}_p\mathbf{e}_s + \mathbf{J}_1^{-1}(\mathbf{n}_2)\mathbf{K}_I \int \mathbf{e}_s dt \tag{4.5}$$

We can then obtain the equation of motion of the vehicle using this controller by substituting (4.5) into (2.6):

$$\begin{aligned} &\mathbf{M}\dot{\mathbf{v}} + \mathbf{C}(\mathbf{v})\mathbf{v} + \mathbf{D}(\mathbf{v})\mathbf{v} + \mathbf{g}(\mathbf{n}_2) + \mathbf{b}(\mathbf{n}_2) \\ &+\hat{\mathbf{B}}_{top}^{-1}\hat{\mathbf{A}}_{top}\mathbf{v} - \mathbf{K}_d\mathbf{e}_v - \mathbf{J}_1^{-1}(\mathbf{n}_2)\mathbf{K}_p\mathbf{e}_s - \mathbf{J}_1^{-1}(\mathbf{n}_2)\mathbf{K}_I \int \mathbf{e}_s dt = \mathbf{0} \end{aligned} \tag{4.6}$$

In the ideal case, the buoyancy and gravity forces would cancel each other since the centre of gravity is coincident with the centre of buoyancy and the model-based part of (4.5) represented by $\hat{\mathbf{B}}_{top}^{-1}\hat{\mathbf{A}}_{top}\mathbf{v}$ would cancel the Coriolis and hydrodynamics terms. The system would be decoupled and its acceleration would depend only on the PID terms. However, the cancelation will not be perfect because of model uncertainties and time-periodic thrust. However, we expect that since the state matrices $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ are updated in real time, the model uncertainties would not be a factor and that the cancelation would be significantly better than for the non-adaptive model-based controller of [79, 80].

4.1.4 Floquet Control Laws: Application

In this section, we describe how the theory discussed in Section 3.5 was applied to the Aqua vehicle. The state-transition matrix was obtained using the single-pass approach developed by Cai et al. [63].

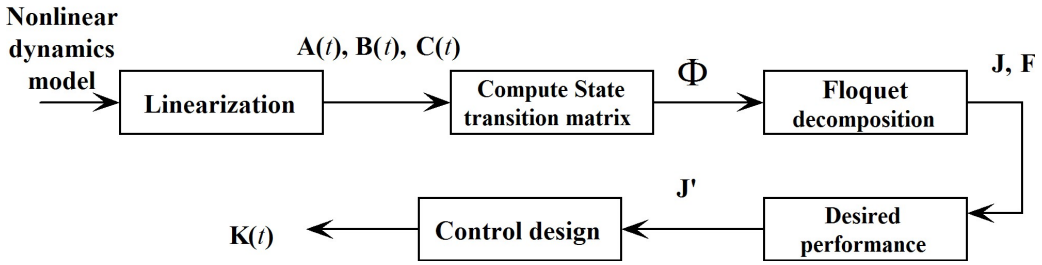


Figure 4–5: Flowchart of the process to obtain the gain matrix using Floquet theory

The complete process to go from the nonlinear dynamics model to the gain matrix is presented in Figure 4–5. The model is first linearized using the technique described in Section 2.6. This technique has the advantage of giving $\mathbf{A}(t)$ as a sum of a constant term and a sinusoidal term, which allows using (3.39)-(3.40) directly. The velocity of the vehicle, states $x_1(t) - x_6(t)$, was held constant in order to linearize the system, implying that the state matrices become less accurate as we deviate from the linearization velocity. Moreover, matrices $\mathbf{D}_c$ and $\mathbf{B}_c$ in (3.40) of Section 3.5.3 depend on the period of oscillation of the paddles, meaning that those matrices are only valid for that particular period. We chose to use a large value for the period weighting factor W_p in (2.30) from the reverse mapping. This had the effect of keeping the period almost constant when transforming the desired thrust into paddle motion. Another option would be to design several Floquet controllers and use gain scheduling in between but that approach was not used. Then, the state-transition matrix is computed with results from Section 3.5.2 while the Floquet factors are obtained using the procedure detailed in Section 3.5.3. Finally, gain matrix can be obtained from the Floquet factors to achieve a desired performance.

The rearrangement of (3.34) was made so that both matrices were composed of real entries. The constant $\mathbf{J}$ matrix is almost diagonal:

$$\mathbf{J} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -0.204 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.787 & 0.398 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.398 & -1.787 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.344 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.383 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -6.40 \end{bmatrix} \tag{4.7}$$

Matrix $\mathbf{J}$ was initially diagonal but some of its entries were complex which is undesirable. Equation (3.34) was used to eliminate the complex part of $\mathbf{J}$ but as a result, $\mathbf{J}$ has some off-diagonal elements. However, this has no negative impact on the effectiveness of this technique. The second factor, $\mathbf{F}(t)$, is time-varying and periodic with the same period P. However, as we can see from (3.45) and (3.49) the inverse of $\mathbf{F}(t)$ is in fact more relevant for use in the controller. Similar to $\mathbf{A}(t)$, it is composed of a constant term and of a sinusoidal term:

$$\mathbf{F}^{-1}(t) = \mathbf{a}_F + \mathbf{K}_F \cos(\frac{2\pi}{P}t) \tag{4.8}$$

where $\mathbf{a}_F$ and $\mathbf{K}_F$ are constant matrices of the same size as $\mathbf{F}^{-1}(t)$:

$$\mathbf{a}_F = \begin{bmatrix} 4.72 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -0.217 & 0 & 0 & 0 & 0.872 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.57 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.06 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1.85 & 0 & 0.442 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 8.53 & 0 & 0 & 0 & -0.02 & 0 & 0 & 0 & 0 & 0 & 1 \\ 4.83 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 6.97 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -5.10 & 0 & 0 & 0 & 1.63 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 4.44 & 0 & 0 & 0 & 1.42 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1.00 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.9}$$

$$\mathbf{K}_F = \begin{bmatrix} 0.68 & 0 & -0.0949 & 0 & 0.21 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0043 & 0 & 0.0014 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.0027 & 0 & -2.1785 & 0 & 0.1410 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.0015 & 0 & 0.0012 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.0015 & 0 & -2.5689 & 0 & 0.1684 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.003 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0068 & 0 & -0.0869 & 0 & 1.20 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0064 & 0 & 0.0102 & 0 & 0.0042 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0291 & 0 & 0.0048 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0057 & 0 & 0.27 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0092 & 0 & 0.0023 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -0.0053 & 0 & -0.0064 & 0 & -0.33 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.10}$$

These two matrices were obtained using a simple curve fitting procedure. $\mathbf{F}^{-1}(t)$ is known and each entry is taken individually and decomposed into its constant or average value and its oscillating term. The first thing to notice from these matrices is that the constant term is generally more dominant than the sinusoidal term. Eq. (4.8) gives us $\mathbf{F}^{-1}(t)$ at any time. However, it would be more convenient in practice to have this matrix as a function of the paddle position since the periodicity of the system comes from the oscillating paddles. Noting that the paddle angle is sinusoidal, we can obtain an equation for $\cos(\frac{2\pi}{P}t)$ as a function of the paddle angle (γ):

$$\cos(\frac{2\pi}{P}t) = \frac{\gamma(t) - \lambda}{A} \tag{4.11}$$

where A is the amplitude of oscillation of the paddle and λ is the paddle offset angle. Using (4.11) is convenient because the paddle angle is measured on the actual robot. Then, we can finally combine (4.8) and (4.11) to obtain an expression that depends only on paddle position $\gamma(t)$:

$$\mathbf{F}^{-1}(t) = \mathbf{a}_F + \mathbf{K}_F \frac{\gamma(t) - \lambda}{A} \tag{4.12}$$

With both Floquet factors known, the next step is to determine an appropriate $\mathbf{J}_c'$ for our system. By altering the pole locations, we can achieve different performance. Our objective was to obtain a critically damped system and therefore the fastest response. Table 4–1 is used to appropriately tune the eigenvalues to obtain the desired performance. It is based on $\mathbf{F}^{-1}(t)$ because it relates $\mathbf{x}(t)$ to $\eta(t)$. Since $\mathbf{a}_F$ represents the average value of $\mathbf{F}^{-1}(t)$, Table 4–1 is obtained from inspection of $\mathbf{a}_F$. For example, in the first row of $\mathbf{a}_F$, the first and seventh entries have significantly higher values than the others. This implies that $\boldsymbol{\eta}_1$ will mostly affect $x_1(t)$ and $x_7(t)$. A user who would like to control $x_1(t)$ and $x_7(t)$ will place more importance on η_1. As we can see, $x_3(t)$ and $x_6(t)$ each appear three times in the bottom row of Table 4–1, which means that their tuning is not straightforward. Moreover, as was mentioned in the previous section, we cannot control all 12 degrees of freedom independently because our controllability matrix $\mathbf{G}(t)$ is only of rank 6. However, since some states of the modal variable depend on multiple states of the vehicle, we can actually control more than 6 states, though not independently. The most obvious choice for the

$\eta(t)$	1	2	3	4	5	6	7	8	9	10	11	12
$\mathbf{x}(t)$	1, 7	6, 8	3, 9	10	3, 5, 11	2, 12	1	5	3	2, 6	2, 6	4

Table 4–1: Relationship between the original states and of the modal variable states

controlled states are $\eta_1(t) - \eta_6(t)$ since this will allow control of 11 of the 12 states of the vehicle. The only uncontrolled state is the fourth one, the roll rate. However,the roll motion is controlled by x_{10}, the roll angle. Another option would be to control $\eta_{12}(t)$ instead of $\eta_2(t)$. In this case, yaw rate and y-position are left uncontrolled in order to control roll rate.

4.2 Reference trajectories

This section describes the trajectories and maneuvers that were defined to test the controllers. The objective was to cover a wide range of motion to have a good assessment of the performance. Some maneuvers will be used only in the simulation because the vehicle can only sense and feed back a limited number of states as discussed earlier in Section 1.1. The states readily available for feed back are the Euler angles and rates. However, since the results presented in the validation section were excellent, we are confident that good performance in the simulation would imply good performance in the experiment. The maneuvers are defined in surge, roll and pitch. The surge maneuvers are tested only in the simulation because sensing is not available for feed back on the actual vehicle. We did not use maneuvers in the sway and heave degrees of freedom because they are seldom use on the actual vehicle and because sensing is innaccurate. The sensing in yaw is more accurate but not reliable enough for feed back. As a result, only roll and pitch motion were tested experimentally.

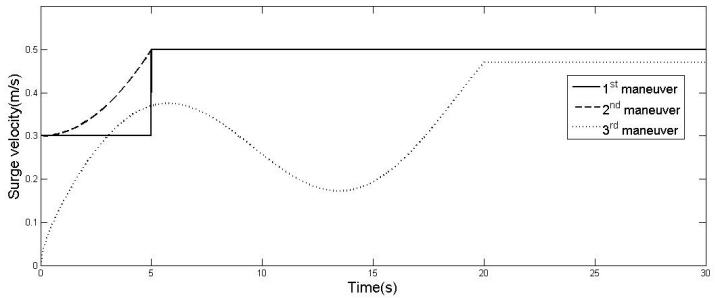


Figure 4–6: Surge trajectories

4.2.1 Surge trajectory

The goal of the surge trajectory is to investigate how the vehicle can track a velocity in the x-direction. Figure 4–6 shows the three trajectories that were defined in the surge motion. The first maneuver is a step from 0.3 m/s to 0.5 m/s to determine whether the controllers can track a constant velocity. The second one is made of two parts: a parabolic acceleration followed by a steady velocity. The third is a constantly varying surge velocity where the vehicle starts from rest and then accelerates and decelerates. Mathematically, it is composed of a sine and square root function. Taken together, these three maneuvers cover the basic range of application for Aqua since the speed is mostly constant in all experiments.

4.2.2 Roll trajectory

The purpose of the roll trajectories is to test the performance of the controllers in roll angle tracking. Many applications require that Aqua remains level or at a particular roll angle to take video footage. Moreover, the roll motion is often used in high performance maneuvers such as a bank turn. Therefore, it is crucial to have good control in roll. We defined three maneuvers that cover the most common roll

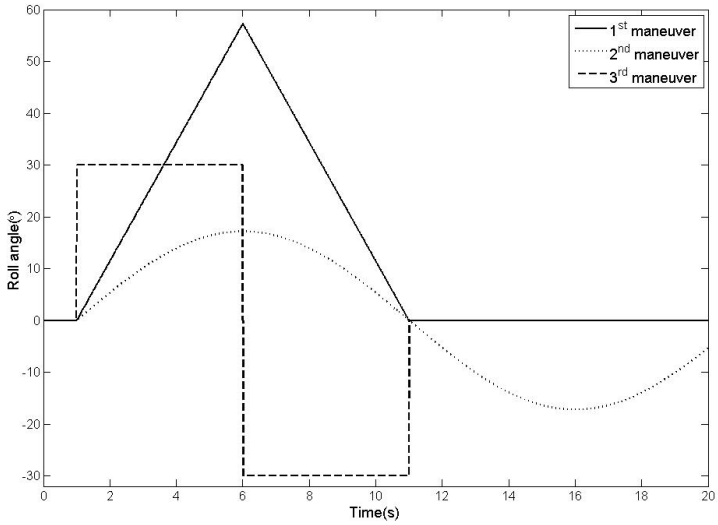


Figure 4–7: Roll trajectories

motions as shown in Figure 4–7. The first maneuver is composed of two ramps, one with a positive slope and one with a negative slope. This will test whether the vehicle can track a linearly varying roll angle. The second is a sinusoidal trajectory. The third one is a step roll where the vehicle has to make a sharp change in roll angle. The third maneuver will provide information about the settling time in roll.

4.2.3 Pitch trajectory

The purpose of the pitch trajectories is to test the performance of the controllers in pitch angle tracking. The pitch angle is used to control the depth of the vehicle and also in bank turns. Figure 4–8 shows the two maneuvers that were defined in the pitch motion. The first maneuver consists of pitch pulses where the vehicle changes its pitch orientation abruptly. This maneuver could be used to avoid an obstacle for example. The second one is a sinusoidal motion. The frequency is higher than

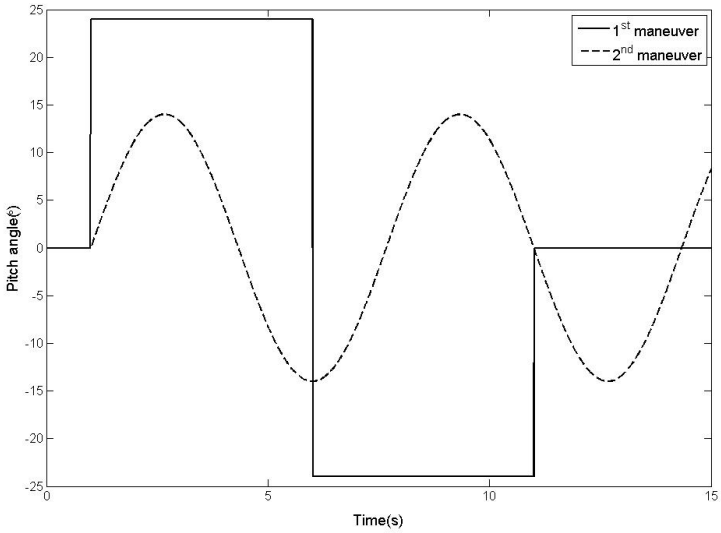


Figure 4–8: Pitch trajectories

for the roll trajectories in order to avoid the vehicle going too deep or going to the surface, both of which would negatively affect the results.

4.3 Simulation Results

This section presents the simulation results in which the vehicle had to track different trajectories. It is seperated into surge, roll and pitch subsections. The PID control gains were tuned by trial and error so as to obtain good performance in the simulation and are presented in Table 4–2. Note that the PID gains were tuned for the sinusoidal trajectories in surge (Figure 4–10), roll (Figure 4–14) and pitch (Figure 4–17). These same PID gains were used in every controller that contains a PID element.

	x	y	z	Roll	Pitch	Yaw
K_d	12 Nsm^{-1}	12 Nsm^{-1}	12 Nsm^{-1}	1 $(Nms)rad^{-1}$	1 $(Nms)rad^{-1}$	0.35 $(Nms)rad^{-1}$
K_p	12 Nm^{-1}	12 Nm^{-1}	12 Nm^{-1}	4 $(Nm)rad^{-1}$	6 $(Nm)rad^{-1}$	0.35 $(Nm)rad^{-1}$
K_I	1 $N(sm)^{-1}$	1 $N(sm)^{-1}$	6 $N(sm)^{-1}$	0.5 $(Nm)(srad)^{-1}$	1 $(Nm)(srad)^{-1}$	0.35 $(Nm)(srad)^{-1}$

Table 4–2: PID control gains used in the simulation

4.3.1 Surge simulation results

The complete six degrees of freedom simulation was run where the vehicle had to follow the three surge trajectories described in Section 4.2.

Figures 4–9, 4–10 and 4–11 show the simulation results for the three surge maneuvers and Table 4–3 shows the settling time to reach a constant speed in Figure 4–9. The settling time was evaluated using the MATLAB function *lsiminfo* that determines the time the vehicle takes to remain within 2% of the desired value.

The first thing to notice in these three figures is the absence of the MBL controller. The MBL gave very poor results and will be treated separately later in this section. We can see that for the step surge tracking shown in Figure 4–9, each controller gives different results with the Floquet and adaptive controllers outperforming the other two. Table 4–3 clearly shows that the Floquet controller gives the best performance in terms of settling time. Both the MBNL and Floquet have the same initial acceleration but the MBNL overshoots the desired speed. The PID controller is much slower than the other three but we can see that it will eventually tracks the desired speed with a small error. The adaptive controller gives good performance except for the first 3-4 seconds during which the adaptive model is not very accurate. Once it settles, the controller becomes accurate, as we can see from the small settling time. More discussion about the evolution of $\hat{\mathbf{A}}$ will be given in Section 4.3.5 while the initialization of $\hat{\mathbf{A}}$ was discussed in Section 4.1.3.

Controller	Settling time(s)
PID	15
MBNL	15
Adaptive	4.5
Floquet	1.4

Table 4–3: Settling time for the constant speed maneuver

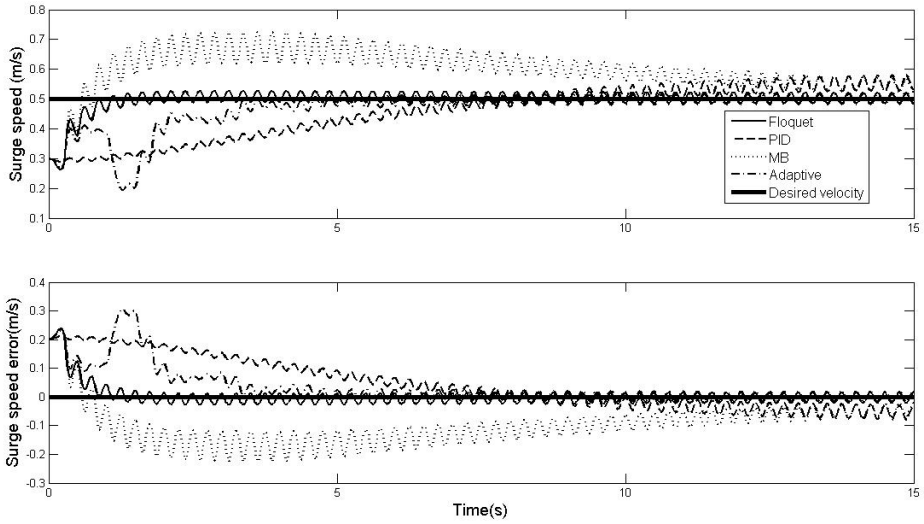


Figure 4–9: Simulation of the first surge maneuver

Figure 4–10 shows clearly that the Floquet and adaptive controllers outperform the MBNL and PID controller. The tracking error with the adaptive or Floquet controller did not exceed 0.05m/s while the error with the PID controller reached 0.2 m/s and the error with MBNL reached 0.15 m/s. As was the case for the constant speed maneuver, the PID gives the worst performance. Relative to the other controllers, the PID controller has a large error for most of the maneuver. Even after 15 seconds, it does not seem to settle. The MBNL is quite accurate during the acceleration phase but again overshoot once the desired speed becomes constant. We can see that the tracking error remains close to 0 for the entire maneuver with the adaptive controller. While the Floquet has some small tracking error during the acceleration, the adaptive follows it accurately. This is because in the case of a continuous acceleration, the adaptive law has time to adjust itself and provide an accurate approximation of $\hat{\mathbf{A}}$. In the case of the transition to constant speed, the change in speed was abrupt and it takes a few seconds for the adaptive adjustment to take place. The adaptive controller slightly overperforms the Floquet controller in the case of an accelerating vehicle. Once the desired speed becomes constant, both demonstrate similar performance. The MBNL is accurate during the acceleration but does not track the constant speed very well. It is also important to note that the other degrees of freedom remains at zero as desired.

We can see in Figure 4–11 that, in the case of a more complex maneuver, the Floquet and adaptive controllers are again superior to the MBNL and PID controllers. However, the PID controller gave better performance for this particular maneuver, mainly because the gains were tuned specifically for this maneuver. However, even

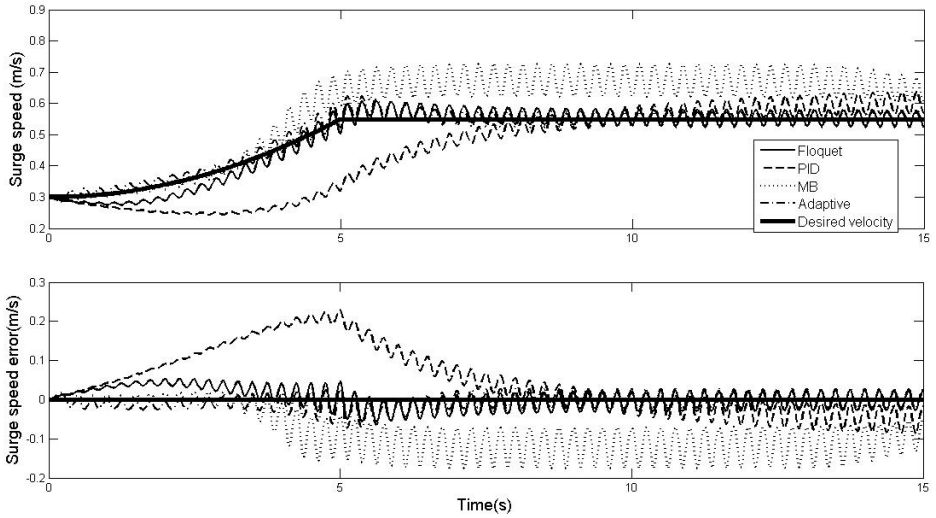


Figure 4–10: Simulation of the second surge maneuver

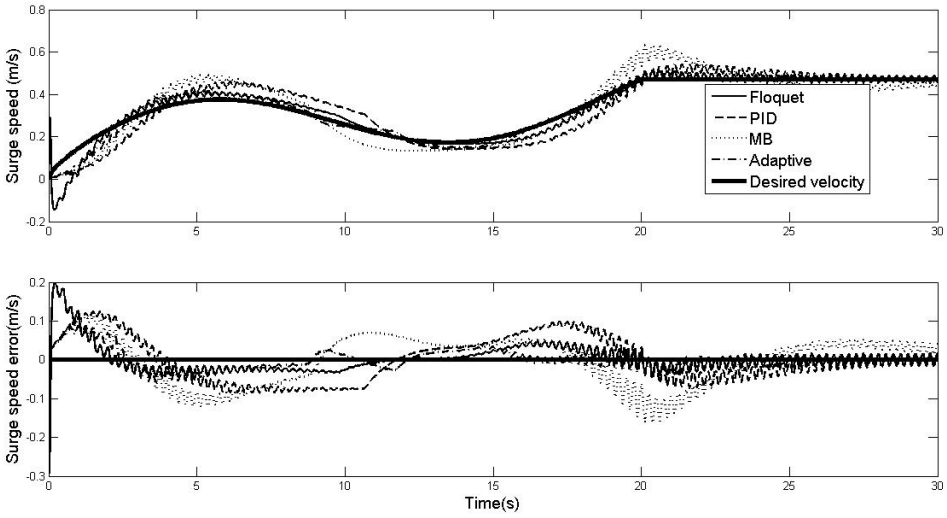


Figure 4–11: Simulation of the third surge maneuver

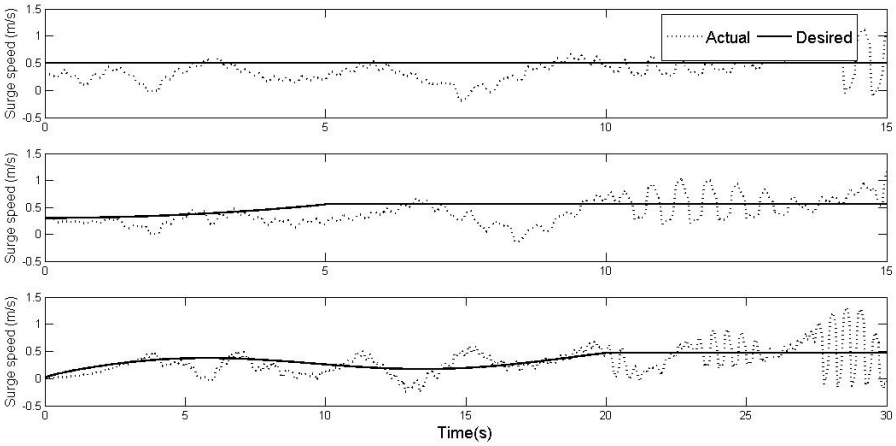


Figure 4–12: Simulation of all three maneuvers using the MBL controller

with this tailor-made tuning, the PID is not as accurate as the Floquet or adaptive controllers. Both these controllers give similar performance but the adaptive controller settles faster. We can also see that none of the controllers achieve perfect tracking.

The MBL controller was treated seperately because of its poor performance as can be observed in Figure 4–12. If its results had been plotted alongside the other controllers, the plots would have been harder to read. The MBL fails to accelerate or even to maintain the speed constant. It does not reach any clear steady-state speed, though on average, it remains around the desired speed. We can also observe on Figure 4–12 that vehicle does not reach an equilibrium. It is difficult to explain why the MBL performed poorly. The PID gains were set to the same values for all controllers, and so, the poor performance came from the model-based part of the controller. The two terms which differed between the MBL and the MBNL controllers were the damping and Coriolis matrix as shown in (3.9) and (3.12). We

can see that the MBL uses the actual velocity of the vehicle instead of the desired one in the calculation of the damping and Coriolis forces. Therefore, any undesired motion could be accentuated by the MBL. For example, an undesired yaw motion will command a positive yaw moment from the damping force because of its form:

$$N_{MBL} = N_{PID} + I_{zz}\dot{r} + \mathbf{C}_{\Psi}(\mathbf{v})\mathbf{v} + \mathbf{D}_{\Psi}(\mathbf{v})\mathbf{v} \tag{4.13}$$

where N_{MBL} and N_{PID} the desired yaw moment as defined by the MBL and PID, $\mathbf{C}_{\Psi}$ is the Coriolis force in the yaw direction and $\mathbf{D}_{\Psi}$ is the damping coefficient in the yaw direction. In the case of an undesired positive yaw rate, the last term will still command a positive yaw moment which will have the effect of accentuating the undesired motion. Because of this poor performance, we have decided not to use the MBL controller in the future simulation and experiments.

Based on the results presented in Figure 4–9, 4–10, 4–11 and Table 4–3 we can conclude that the Floquet and adaptive controllers are superior to the PID and MBNL for surge speed tracking. The Floquet controller appears to be better for constant speed tracking while adaptive handles the acceleration better.

4.3.2 Roll simulation results

Once again, the six degrees of freedom simulation was run where the vehicle had to follow the three roll trajectories described in Section 4.2. The controllers used the same gains as for the surge maneuvers presented in Table 4–2. Figures 4–13, 4–14 and 4–15 show how the vehicle was able to track the three roll maneuvers. The first thing to notice is that all controllers provide good roll tracking performance. In the first maneuver, shown in Figure 4–13, we can see that the Floquet controller gives

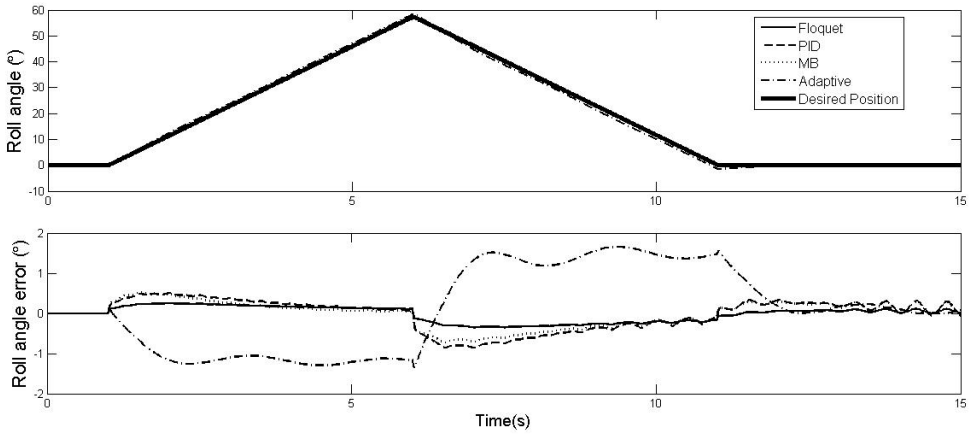


Figure 4–13: Simulation of the first roll maneuver

the best performance while the adaptive controller gives the worst. The good result from Floquet was expected since it deals directly with the time-periodic feature of Aqua. The PID and MB controllers give similar performance with a small advantage to the model-based one. Furthermore, if we zoom at the end of the maneuver, we could see that the adaptive controller has the smallest steady-state error.

In the case of the second maneuver displayed in Figure 4–14, we see that all controllers give similar performance. The tracking error is insignificant in all cases. We can explain the improvement of the adaptive controller by the fact that the desired roll rate was changing gradually and the adaptive controller had more time to adjust. For the third maneuver shown in Figure 4–15, the adaptive controller was the only controller that did not give an overshoot, while the three other controllers had a small overshoot which is most visible at $t \approx 10s$. However, all controllers give good results and the error remains small.

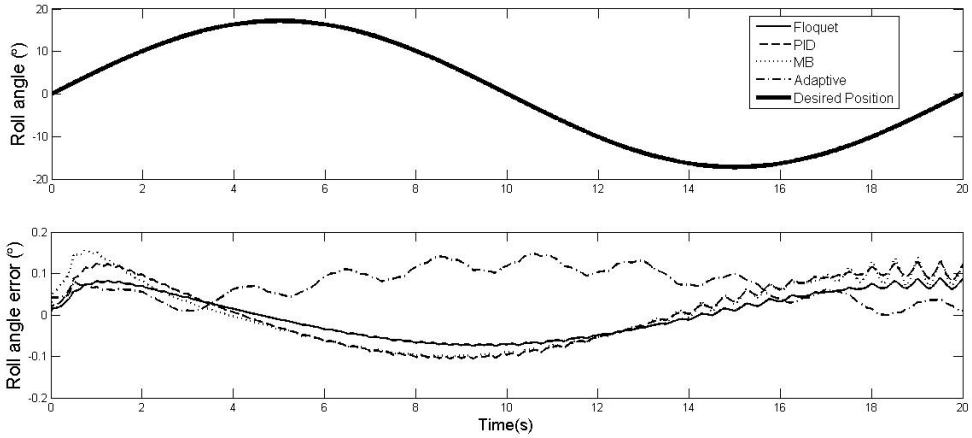


Figure 4–14: Simulation of the second roll maneuver

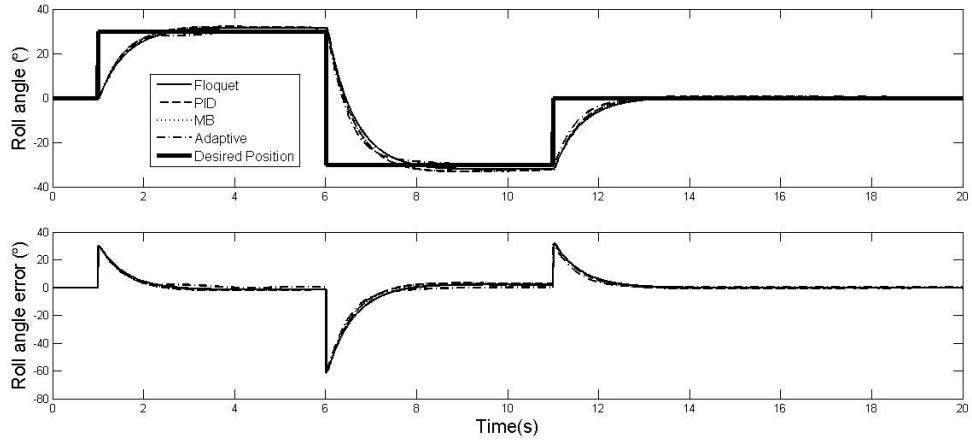


Figure 4–15: Simulation of the third roll maneuver

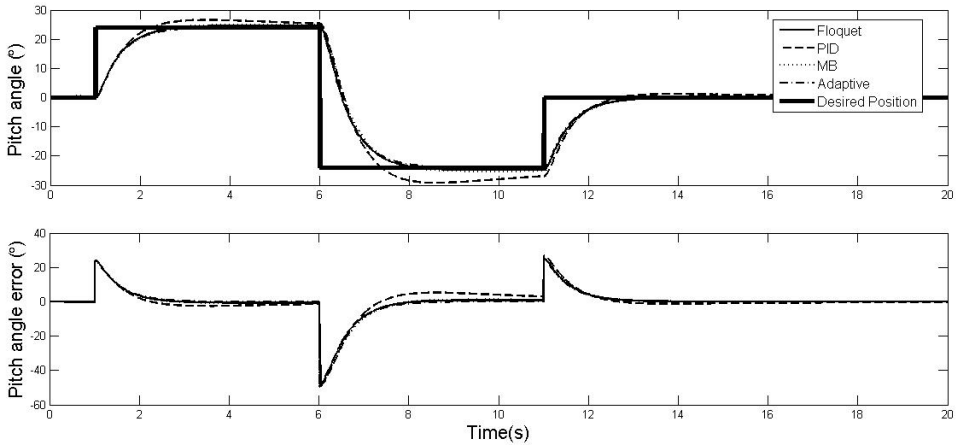


Figure 4–16: Simulation of the first pitch maneuver

4.3.3 Pitch simulation results

Once again, the six degrees of freedom simulation was run where the vehicle had to follow the two pitch trajectories described in Section 4.2. The controllers used the same gains as for the surge maneuvers presented in Table 4–2. Figures 4–16 and 4–17 show how the vehicle tracked the 2 pitch maneuvers. We clearly notice in both figures that the PID controller gives the worst performance. It has a large overshoot and takes more time than the other controllers to settle. This is probably due to the fact that the integral gain is higher in pitch than it was in roll. The PID pitch gains were tuned using the maneuver shown in Figure 4–17 and the integral gain would have been lower if tuned using the first pitch maneuver. The two other controllers that used the PID gains (MBNL and adaptive) perform better because they had other terms to counter balance the integral effect.

In the second pitch maneuver shown in Figure 4–17, the PID controller is still the worst but gives better performance than for the first maneuver. The adaptive

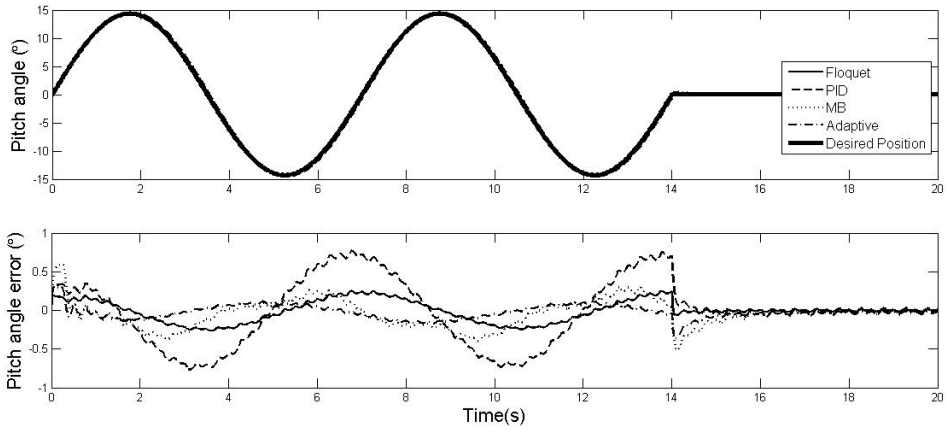


Figure 4–17: Simulation of the second pitch maneuver

controller is slightly more accurate than the others, but only by a very small margin. The Floquet and MB controllers give good performance with similar error. Based on the results shown in Figure 4–16 and 4–17, we can conclude that all controllers can track pitch accurately with a small advantage for the adaptive controller. Moreover, we note that the PID controller gives the worst performance.

4.3.4 Surge simulation with disturbances

In the previous simulations, it was assumed that there were no external flow/current acting on the vehicle. In this section, we have simulated an horizontal current of $0.2m/s$ acting at angle of 45°. The disturbance was simulated by a pulse starting at $t = 5s$ and ending at $t = 10s$. We compared the motion of the vehicle with and without a controller to evaluate the performance of the controller with disturbances. The objective of the controller was to keep the velocity constant at $u = 0.455m/s$. Figure 4–18 shows the controlled and uncontrolled motion of the vehicle under disturbance. The controlled motion shown in Figure 4–18 was obtained using the Floquet

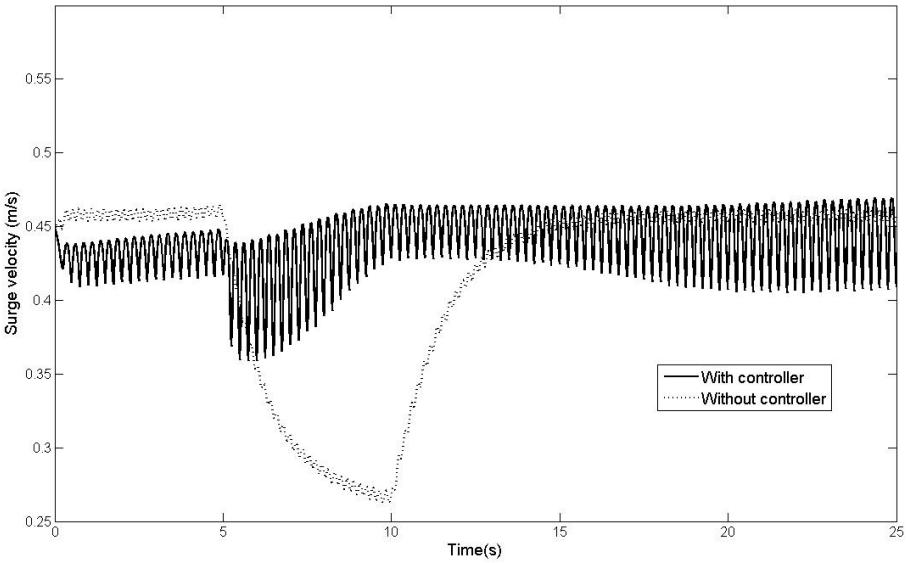


Figure 4–18: Surge simulation with a disturbance applied between $t = 5s$ and $t = 10s$.

controller. The adaptive controller also gave good performance but is not shown to keep the figure understandable and clear. We clearly see that the speed of the uncontrolled vehicle decreases drastically until the current stops at $t = 10s$. On the other hand, the speed of the controlled vehicle decreases when the disturbance starts at $t = 5s$ but the controller compensates for it and the speed remains constant afterward. From this, we can conclude that the Floquet controller is robust to flow disturbance.

4.3.5 Evolution of $\hat{\mathbf{A}}$ in the simulation

As mentioned in Section 3.4.1, one of the advantage of the MIAC adative controller was that it provides an approximation of state-space matrix $\mathbf{A}$. In this section, we present a comparison between $\mathbf{A}$ obtained in Section 2.6 and the estimated by

Maneuver	$\hat{\mathbf{A}}_{1,1}$	$\mathbf{A}_{1,1}$	$\hat{\mathbf{A}}_{4,4}$	$\mathbf{A}_{4,4}$	$\hat{\mathbf{A}}_{5,5}$	$\mathbf{A}_{5,5}$
Surge (Figure 4–9)	-0.834	-0.8726	-5.38	-5.54	-9.62	-9.264
Roll (Figure 4–15)	-0.824	-0.8726	-5.489	-5.54	-9.22	-9.264
Pitch (Figure 4–16)	-0.830	-0.8726	-5.520	-5.54	-9.37	-9.264

Table 4–4: Comparison of $\hat{\mathbf{A}}$ and $\mathbf{A}$ during the simulation

the MIAC update algorithm. We compare the most relevant elements of the matrix for each maneuver: $\mathbf{A}_{1,1}$ for the surge maneuver, $\mathbf{A}_{4,4}$ for the roll maneuver and $\mathbf{A}_{5,5}$ for the pitch maneuver. The results are tabulated in Table 4–4. We used the step maneuvers in surge, roll and pitch as the reference maneuvers. They were chosen because they match the linearization conditions and are therefore suitable for direct comparison: steady-state surge speed of 0.5 m/s with zero roll rate and pitch rate. The step maneuvers remain closest to these conditions than the other maneuvers. The values presented in Table 4–4 were evaluated at the end of the maneuver.

We can see from Table 4–4 that $\hat{\mathbf{A}}$ generally matches $\mathbf{A}$ with good accuracy for all maneuvers. However, $\hat{\mathbf{A}}_{5,5}$ takes significantly more time to converge than the two other entries of $\hat{\mathbf{A}}$ shown in the Table. Increasing the adaptive gain pertaining to pitch did not improve the convergence speed. One cause of this behavior is that there are two entries of $\mathbf{A}$ that strongly affect the pitch motion: $\mathbf{A}_{5,5}$ and $\mathbf{A}_{5,3}$. This means that in the pitch motion these two entries can offset each other and reduce the speed of convergence:

$$\dot{\hat{\mathbf{A}}}_{5,3} = \mathbf{\Gamma}_{heave} \boldsymbol{\epsilon}_3 \mathbf{x}_3 \tag{4.14a}$$

$$\dot{\hat{\mathbf{A}}}_{5,5} = \mathbf{\Gamma}_{pitch} \left(\boldsymbol{\epsilon}_5 \mathbf{x}_3 + \boldsymbol{\epsilon}_5 \mathbf{x}_5 \right) \tag{4.14b}$$

$$\dot{\boldsymbol{\epsilon}}_3 = \mathbf{A}_{3,3}\boldsymbol{\epsilon}_3 + \left(\hat{\mathbf{A}}_{3,3} - \mathbf{A}_{3,3}\right)\hat{w} \tag{4.14c}$$

$$\dot{\boldsymbol{\epsilon}}_5 = \mathbf{A}_{5,3}\boldsymbol{\epsilon}_3 + \mathbf{A}_{5,5}\boldsymbol{\epsilon}_5 + \left(\hat{\mathbf{A}}_{5,3} - \mathbf{A}_{5,3}\right)\hat{w} + \left(\hat{\mathbf{A}}_{5,5} - \mathbf{A}_{5,5}\right)\hat{q} \tag{4.14d}$$

where $\dot{\boldsymbol{\epsilon}}_5$ is the derivative of the pitch rate error and $\dot{\boldsymbol{\epsilon}}_3$ is the derivative of the heave velocity error. We can see from (4.14b) that $\dot{\hat{\mathbf{A}}}_{5,5}$ will converge to zero when $\boldsymbol{\epsilon}_5$ converges to zero. However, the convergence of $\boldsymbol{\epsilon}_5$ depends on $\mathbf{A}_{5,3}$ and that entry also takes a long time to converge to its final value. In our case, the desired value for pitch rate and heave velocity is zero which means that $\hat{q}$ and $\hat{w}$ are small. If $\mathbf{A}_{5,3}\boldsymbol{\epsilon}_3 \approx -\mathbf{A}_{5,5}\boldsymbol{\epsilon}_5$ in (4.14d), then $\dot{\boldsymbol{\epsilon}}_5$ will be small and as a result the convergence time will be large.

Figure 4–19 shows how $\hat{\mathbf{A}}_{1,1}$ adapts itself and eventually converges to its final value. We can see in Figure 4–19a that it takes around 5 seconds to settle to a steady-state value. Figure 4–19b shows how $\hat{\mathbf{A}}_{1,1}$ changes over a 2 second period once it has converged to its final value. We notice that it oscillates at twice the paddle frequency which means that the controller reacts to Aqua's time-periodic thrust. Therefore, even though the vehicle reaches its desired speed, the controller keeps updating $A_{1,1}$ because of the oscillating thrust. The frequency of oscillation observed on Figure 4–19b is twice the paddle frequency because there are two thrust cycles per paddle cycle.

4.4 Experimental Results

This section presents the results of the trajectory tracking experiments performed on the Aqua underwater vehicle. The experiments were done in the Caribbean Sea in January 2009 and January 2010. Aqua sensors can only measure the Euler

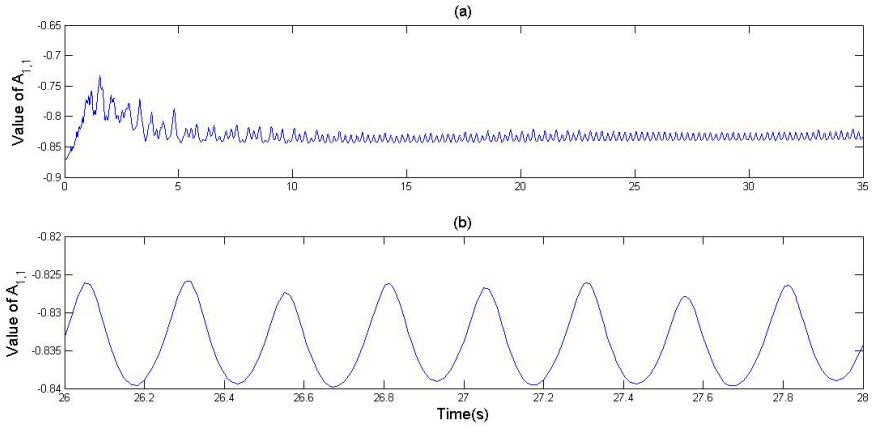


Figure 4–19: (a) Value of $A_{1,1}$ with time during the maneuver shown in Figure 4–9 (b) Zoom of $A_{1,1}$ between t=26s and t=28s

	Roll	Pitch	Yaw
K_d	3 $(Nms)rad^{-1}$	3 $(Nms)rad^{-1}$	0.5 $(Nms)rad^{-1}$
K_p	5 $(Nm)rad^{-1}$	6 $(Nm)rad^{-1}$	0.5 $(Nm)rad^{-1}$

Table 4–5: PID control gains used in the experiment

angles accurately. The sensing in the translational degrees of freedom is inaccurate and cannot be used for feedback purposes. Sensing in yaw is more accurate than in the transaltional degrees of freedom but still unreliable for feedback purposes. Moreover, we were not able to use the integral gains on the robot and it was set to zero for all controllers. Therefore only the pitch and roll maneuvers were tested during the experiments and those maneuvers are described in Section 4.2. The PD gains had to be retuned for the experiments because it was clear from our first test that the gains obtained in the simulation could be improved. The new gains are presented in Table 4–5. Note here that the translational degrees of freedom were left uncontrolled. Moreover, since we had trouble in the simulation with the MBL, this controller was not tested in the most recent experimental session in January 2010.

Figure 4–20: Boat used in the experiment in Barbados

The experiment was performed from a boat located around 100 meters from the shore. The boat used during the experiment is shown in Figure 4–20 and the experimental setup is similar to the shore on as shown on Figure 4–2 in Section 4.1.2. More information about the experimental setup was given in Section 1.1.5. From that distance, the effect of the surf was considered negligible. The vehicle was placed at a depth of one meter by a diver and released. Once released, the vehicle started to move in straight line with the controller inactive for four seconds to allow the vehicle to reach steady-state. Then, the controller was switched on and the vehicle started to track the prescribed trajectories. The speed was uncontrolled but the period and amplitude of oscillation of the paddles were set at the values that give a speed of 0.5 m/s when going in straight line. The controllers change the amplitude of oscillation to produce more thrust, but due to the high weight put on period in

Controller	Settling time(s)
PD	4.5
MBNL	N/A
Adaptive	1.9
Floquet	1.6

Table 4–6: Experimental settling time for the maneuver shown in Figure 4–23

(2.30) of Section 2.5, the period of oscillation is almost constant. The roll and pitch motion was controlled by the controllers.

4.4.1 Roll experiment results

This section presents the experimental results for the three roll maneuvers. Figures 4–21, 4–22, 4–23 and Table 4–6 shows how the vehicle was able to track the maneuvers. From Figure 4–21, we can see that the Floquet controller gives the best performance in tracking a steadily increasing roll angle. The adaptive controller gives good performance, especially when the roll rate is positive, and gives more error when the desired roll angle starts to decrease. When the vehicle reaches the maximum roll angle, the desired speed suddenly changes from positive to negative which has the effect of changing the state matrix $\mathbf{A}$. Therefore, the adaptive model $\hat{\mathbf{A}}$ becomes inaccurate and some time is required to make the update. As a result the controller does not initially output enough force to make the sharp change in roll motion. The PD controller gives reasonable performance while the MBNL follows the general motion, but with an error of about 25%.

In the case of Figure 4–22, all controllers track the trajectory accurately. The performance of the adaptive and Floquet controller is marginally better. There are more significant differences in the case of the maneuver shown in Figure 4–23. Table 4–6 shows the settling time pertaining to the maneuver shown in Figure 4–23.

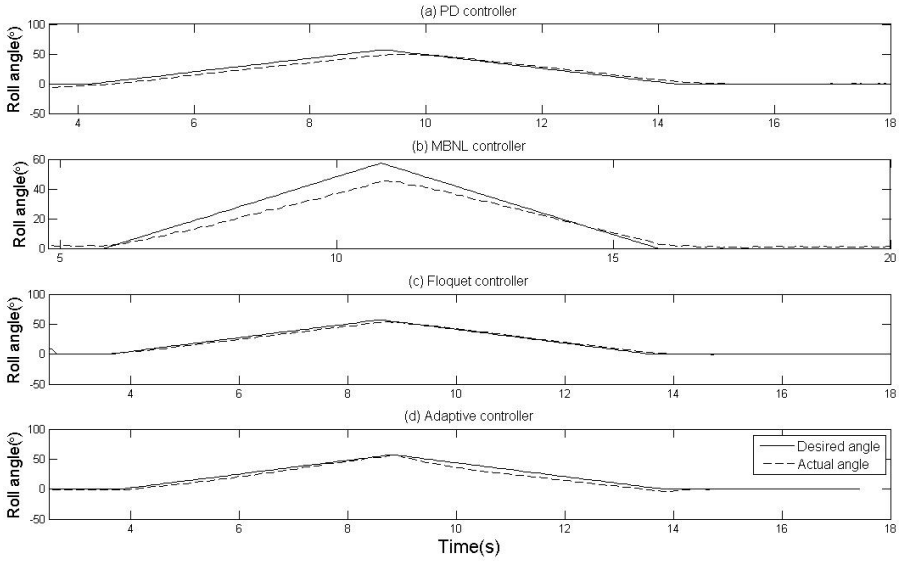


Figure 4–21: Experimental results of the first roll maneuver.

The settling time was obtained as in the simulation using the MATLAB function *lsiminfo*. Again, the MBNL controller gives the worst performance and does not reach the desired roll angle within a reasonable time. The PD controller tracks the desired roll angle but the settling time is significantly larger than that of the Floquet or adaptive controller. The adaptive and Floquet controllers give good performance with small settling time. We can also notice the presence of oscillations in the case of the Floquet controller. This oscillation is due to high control gains in roll which has the effect of increasing the oscillations.

Based on the results presented in Figures 4–21, 4–22, 4–23 and Table 4–6, we can conclude that the adaptive and Floquet controllers outperform the MBNL and PD controllers in roll tracking. Moreover, we see that in the case of a constant roll

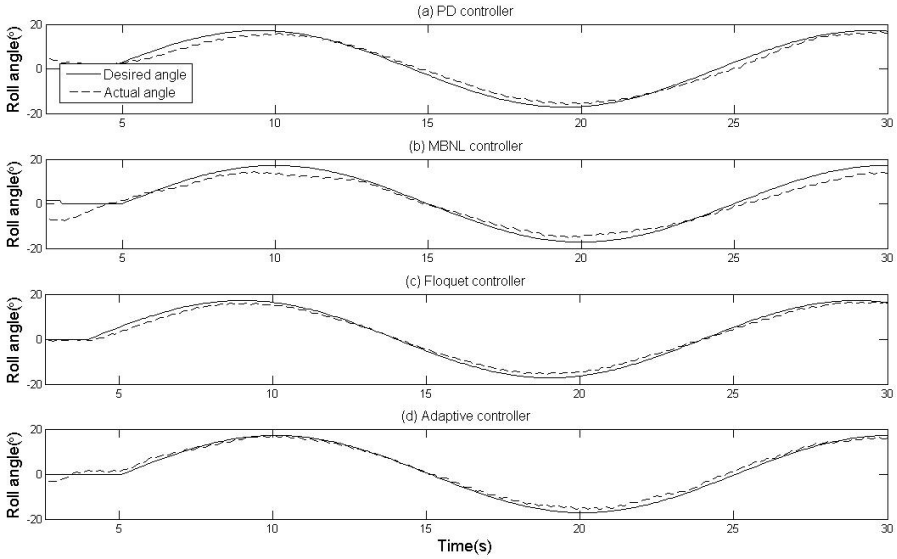


Figure 4–22: Experimental results of the second roll maneuver.

rate, the Floquet controller gives the best performance. In the case of a step change in roll angle, the adaptive controller has the advantage.

4.4.2 Pitch experiment results

This section presents the experimental results for the two pitch maneuvers. Figures 4–24 and 4–25 show how the vehicle tracks two pitch maneuvers. We can clearly see from both figures that the performance in pitch tracking is not as good as that in roll tracking, which had already been suggested by the simulation results. However, the simulation indicated a better performance than what was obtained in the experiment.

We can see in Figure 4–24 that only the adaptive controller was able to reach the desired pitch angle within the required time of 5 seconds. The adaptive controller even overshoots the desired pitch angle when it goes to $\theta_d = -28.6°$. This poor

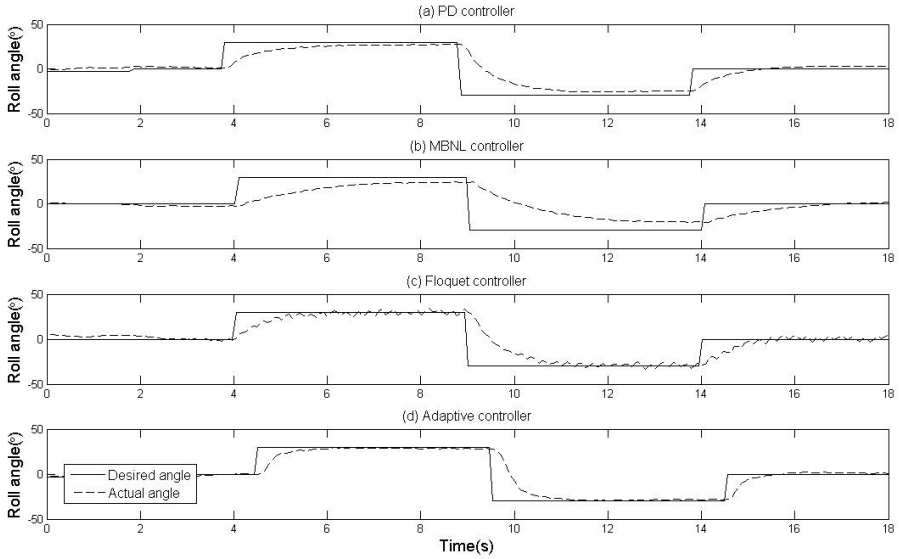


Figure 4–23: Experimental results of the third roll maneuver.

performance of the PD and Floquet controllers is probably due to a high derivative gain relative to the proportional gain. In a maneuver as shown in Figure 4–24, the desired roll rate is zero and therefore, the derivative part of the PD controller tries to prevent the vehicle from rotating and negates the effect of the proportional gain that tries to move the vehicle to its desired pitch angle.

The performance of the different controllers in tracking a sinusoidal pitch maneuver as shown in Figure 4–25 is also worse than expected. First, we see that for all four controllers, the vehicle significantly lags the desired roll angle. We could not reduce the frequency of the maneuver in order not to hit the sea bottom or reach the surface. We see that the Floquet controller has the smallest lag but it does not reach the maximum pitch angle. The adaptive controller shows more lag but after three complete oscillations, it reaches the maximum pitch angle. This implies that

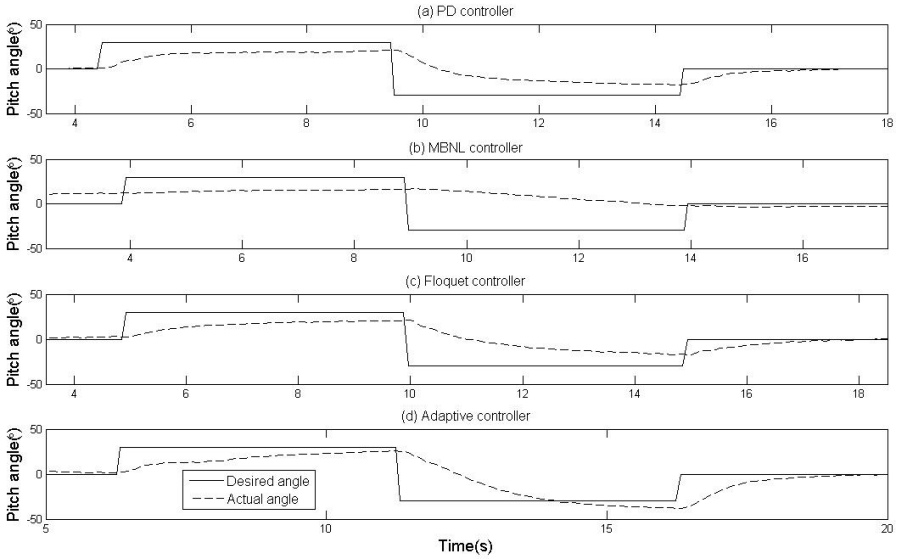


Figure 4–24: Experimental results of the first pitch maneuver.

it takes around 3 cycles to adapt itself. However, we do not have enough data to see if the adaptive controller would eventually reduce the time lag.

Based on these results, we can conclude that the adaptive controller gives a significantly better performance than the three others in pitch tracking. The MBNL gives the worst performance and the results suggest that there might be an implementation mistake since previous experiments suggested good performance from the MBNL [80].

4.4.3 Evolution of $\hat{\mathbf{A}}$ in the experiment

In this section, we present a comparison between $\mathbf{A}$ obtained in Section 2.6 and the one estimated by the MIAC update algorithm. We compare the most relevant elements of the matrix for each maneuver: $\mathbf{A}_{4,4}$ for the roll maneuver and $\mathbf{A}_{5,5}$ for the pitch maneuver. Once again, we used the step maneuvers in roll and pitch as the

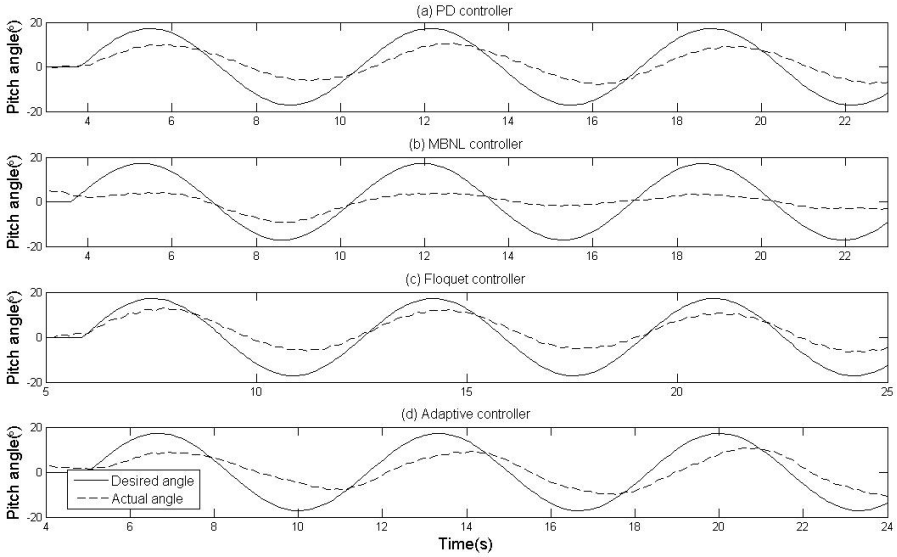


Figure 4–25: Experimental results of the second pitch maneuver.

reference maneuvers, because they are the maneuvers that remain the closest to the linearization conditions: steady-state surge speed of 0.5 m/s and zero roll rate and pitch rate. The results are tabulated in Table 4–7. We see that the match between $\hat{\mathbf{A}}$ and $\mathbf{A}$ is not as good as it was during the simulation. One cause for this is that the forward speed of the vehicle was likely not 0.5 m/s because during the maneuver, some of the thrust is used to perform the maneuver instead of keeping the speed constant. By repeating the linearization process for different values of the forward speed, we found that at $u = 0.32m/s$ there is a better match: $\mathbf{A}_{4,4} = -5.96$ and $\mathbf{A}_{5,5} = -5.64$. In the simulation, we found that the velocity of the vehicle dropped to $u = 0.338m/s$ during this maneuver before going back to $0.5m/s$. In the experiment, the surge velocity is uncontrolled so it did not reaccelerate to $u = 0.5m/s$. Without

$\hat{\mathbf{A}}_{4,4}$	$\mathbf{A}_{4,4}$	$\hat{\mathbf{A}}_{5,5}$	$\mathbf{A}_{5,5}$
-6.46	-5.54	-5.90	-9.26

Table 4–7: Comparison of $\hat{\mathbf{A}}$ and $\mathbf{A}$ during the experiment

any accurate measurement of the forward speed u, comparing the model obtained through linearization and the one obtained with the MIAC is of questionable.

Figure 4–26 shows how the value of $\hat{\mathbf{A}}_{4,4}$ and $\hat{\mathbf{A}}_{5,5}$ were updated. The maneuver shown in Figure 4–23 was used to generate Figure 4–26a and the maneuver in Figure 4–24 was used to generate Figure 4–26b. Both maneuvers were tested twice. The first thing to notice is that both trials converge to approximately the same the same value. We also notice from Figure 4–26a that $\hat{\mathbf{A}}_{4,4}$ becomes positive around t=10s. This is probably due to a too high adaptive roll gain. By reducing the value of Γ_{roll}, the curves presented in Figure 4–26 would probably be smoother. The same is true for $\hat{\mathbf{A}}_{5,5}$.

Although it is not visible on Figure 4–26, we have observed the presence of an oscillation with frequency equal to twice the paddles' frequency of oscillation. This phenomenon was also be observed in the simulation. This oscillation, albeit small, is the reaction of the adaptive law to the time-periodic thrust provided by the paddles.

4.5 Summary of the Results

This chapter presented the simulation and experimental results for the four trajectory tracking controllers developed in Chapter 3. These controllers were implemented in a dynamics simulation and on the Aqua underwater vehicle. They were then tested to assess and compare their performance. Trajectories were defined in surge, roll and pitch degrees of freedom. We did not evaluate sway, heave or yaw

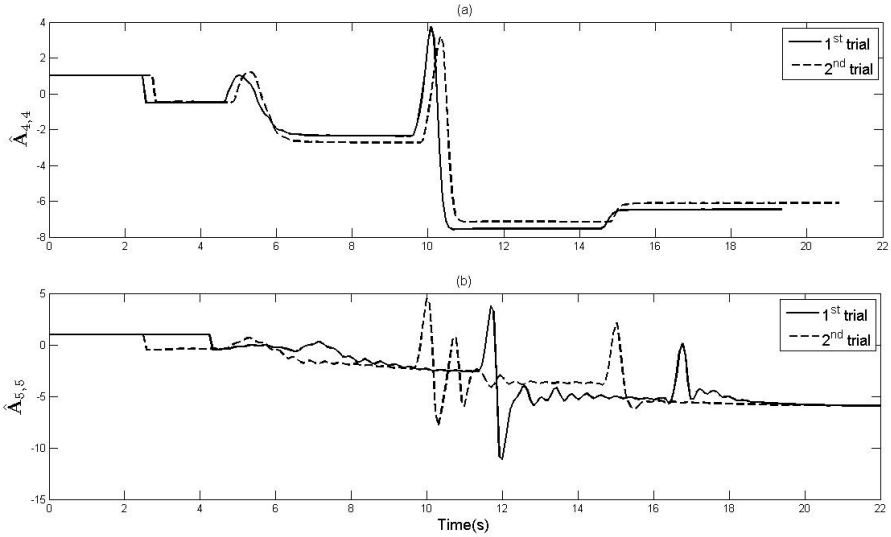


Figure 4–26: Variation of $\hat{\mathbf{A}}_{4,4}$ and $\hat{\mathbf{A}}_{5,5}$ with time.

tracking because the vehicle was not equipped with reliable feedback sensing in these degrees of freedom. The tracking of a yaw trajectory was tested in the simulation during the optimization discussed in Chapter 5.

The simulation results suggested that in surge speed tracking, the Floquet controller was the best for constant speed tracking and the adaptive controller was the best for acceleration. In roll motion, every controller gave a good performance but the Floquet controller was better by a small margin. In pitch, the adaptive controller was the most accurate of all controllers. In the experiment, the Floquet and adaptive controllers were the most accurate. The Floquet controller gave the best performance in all three roll maneuvers. In the third roll maneuver, the adaptive controller gave similar performance as the Floquet controller. For pitch tracking, the

adaptive controller was the most accurate, but no controller was capable of tracking the sinusoidal pitch motion accurately. We also saw that there is a good match between the simulation and the experiment results. The controllers perform better in the simulation but in general, the results are similar. Moreover, the Floquet and adaptive controllers were superior in both the simulation and the experiment.

Each of these controllers would perform differently depending on the type of mission. We could expect that when the trajectory and mission conditions are known in advance, the Floquet controller would be the most appropriate. It gave very good performance for all maneuvers but it is not as robust to disturbances as the adaptive controller is. The adaptive controller could be used for missions when there are several unknown conditions and a lot of maneuvring. This type of mission would take advantage of the adaptive capability of this controller.

CHAPTER 5

Maneuver Optimization

With a validated dynamics model and effective controllers available for Aqua, it is now possible to study particular applications for the vehicle. In this chapter we use the vehicle model and associated controllers to optimize a high performance maneuver. As was mentioned in Section 1.2, other researchers have studied high performance maneuver and path planning for underwater vehicle [67,68,70,77]. Their work provides good starting point for our optimization but cannot be applied directly to Aqua because of the differences in vehicle.

A U-turn was chosen for this optimization because of its many practical applications. The optimization approach discussed in this chapter could be used to optimize other high performance maneuvers for underwater vehicles. Moreover, the results of this optimization can be used in real life application such as coral reef inspection.

The chapter is organized as follow. The details of the maneuver are presented in Section 5.1. Three design variables were chosen to characterize the maneuver: desired speed, turn radius and bank angle. This is discussed in Section 5.2. The controller used in for this work is presented in Section 5.3. Then, the objective function and the constraints are described in Sections 5.4 and 5.5. A genetic algorithm was chosen to

perform the optimization, and the motivation for that decision is discussed in Section 5.6. Finally, the results of the optimization are presented in Section 5.7.

5.1 Objective and motivations

One of the many applications of Aqua is coral reef inspection. In order to inspect a larger area many video segments are recorded and combined to get an overall image of the reef. In order to get an accurate image of the area, the vehicle must obtain images using a systematic procedure [81, 82]. This is accomplished by following parallel track line as shown in Figure 5–1, where the area being inspected would be located under the vehicle path. This is known as a Boustrophedon pattern. At the end of each track line, the vehicle has to perform a 180 degrees turn in order to reach the next track line, a distance D from it. During the turn, the vehicle does not perform any useful work, since the data collected during transient motion is usually of no value. For a large area with several track lines, a significant amount of time might be lost. Therefore, it would be extremely useful to minimize that wasted time.

In this chapter, the objective is to minimize the time to transition from one track line to the next by determining the best path to perform the maneuver. The maneuver is assumed to be composed of 3 parts that can be seen in Figure 5–1. The first part is a quarter circle of radius R. Then the vehicle follows a straight trajectory perpendicular to the track line for a distance $d = D - 2R$ and finally another quarter circle of radius R to complete the turn. The maximum turn radius is half the distance between the track lines. The total distance (S) that the vehicle has to travel is a function of R and D:

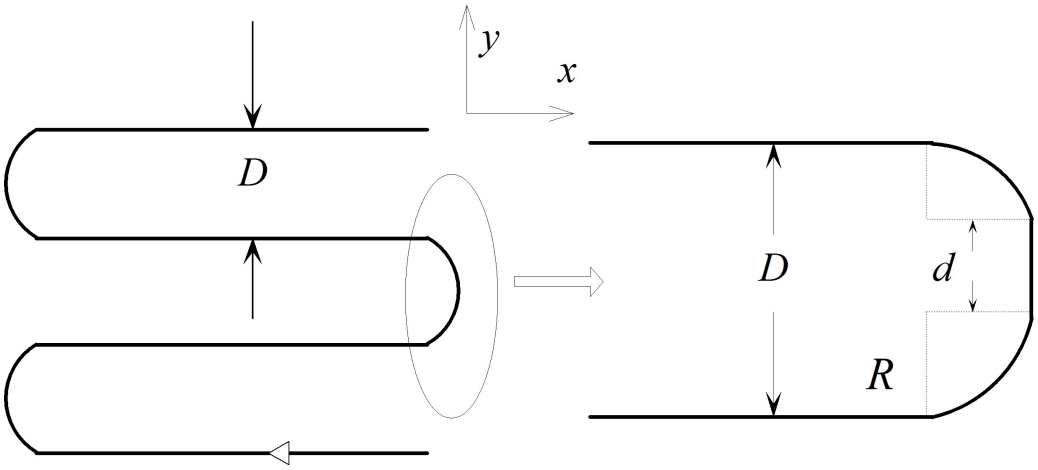


Figure 5–1: View from above of the ptimization trajectory

$$S = D + (\pi - 2)R \tag{5.1}$$

From (5.1), we can see that the total distance increases linearly with turn radius for a given track line separation. We can also notice that for $R = \frac{D}{2}$, $S = \pi R$. With perfect tracking and constant speed, we would expect that by minimizing S, we would also minimize the time. However, since perfect tracking is not guaranteed, and the speed does not stay constant, we cannot assume that the smallest S gives the smallest time.

5.2 Design variables

This section discusses the design variables or parameters that are used in the optimization. Three design variables were used in our optimization problem, the turn radius (R_{in}), the desired speed (u_{in}) and the desired bank angle (ϕ_{in}). These values were used to generate the desired trajectory of the vehicle, defined by x_d, y_d,

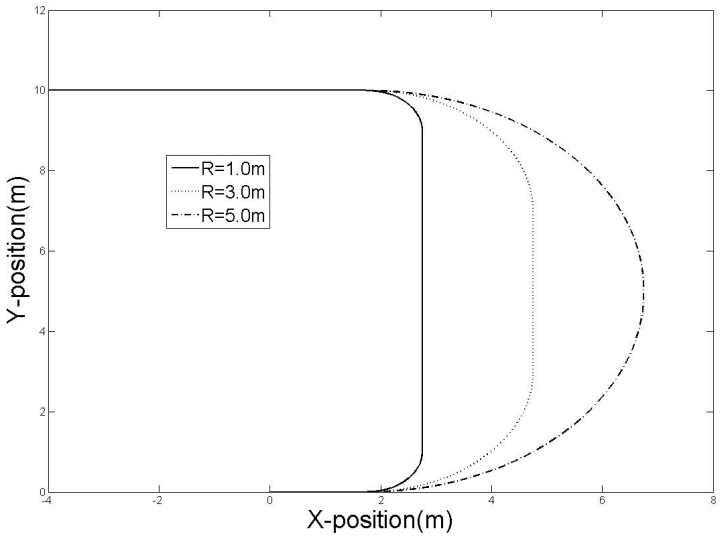


Figure 5–2: Path of the vehicle for different turn radius for $D = 10$m

u_d and ϕ_d—respectively the longitudinal position, the lateral position, the forward speed and the bank angle of the vehicle. The desired values of the vertical position and pitch angle were always zero, while the desired heading angle was such that the vehicle was always oriented tangent to the desired path in the x-y plane.

The turn radius is the only design variable that directly affects the path taken by the vehicle to go from one track line to the other. This can be observed in Figure 5–2 where we see how the turn radius affects the desired trajectory of the vehicle for a trackline separation of 10m. We expect that for a larger R, the vehicle will be able to maintain a higher speed than with a small R, and this may compensate for the fact that the path S is larger.

The desired speed of the vehicle, u_d, can be viewed as the speed setpoint during the turn maneuver. It is a desired speed because, during the turns, the vehicle usually

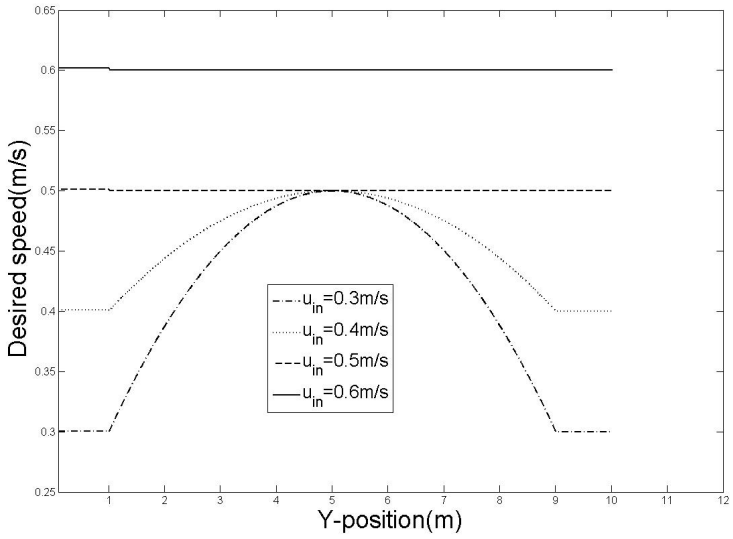


Figure 5–3: Desired speed of the vehicle,u_d, as a function of its y-position for $D = 10$m and $R_{in} = 1$m

loses some of its forward speed. We expect that by reducing the speed of the vehicle during the turn, it may be possible to perform sharper turns.

Figure 5–3 shows the desired speed of the vehicle,u_d, as a function of the y-position for four different values of u_{in}. The vehicle is turning for $0 < y < R$ and $(D - R) < y < R$. During those portions of the maneuver, the desired speed of the vehicle is set to the input desired speed u_{in}. However, for $R < y < (D - R)$, the vehicle is not turning and the desired speed can be increased without sacrificing the turning capabilities of the vehicle. If the desired speed remained at u_{in} for the straight part of the turn, this would significantly reduce the performance of the vehicle. Therefore, as shown in Figure 5–3, the desired speed is increased during the straight part before decreasing for the second quarter-circle turn. Note that for u_{in} $\geq$ 0.5m/s, the desired speed remains constant for the entire maneuver. Moreover,

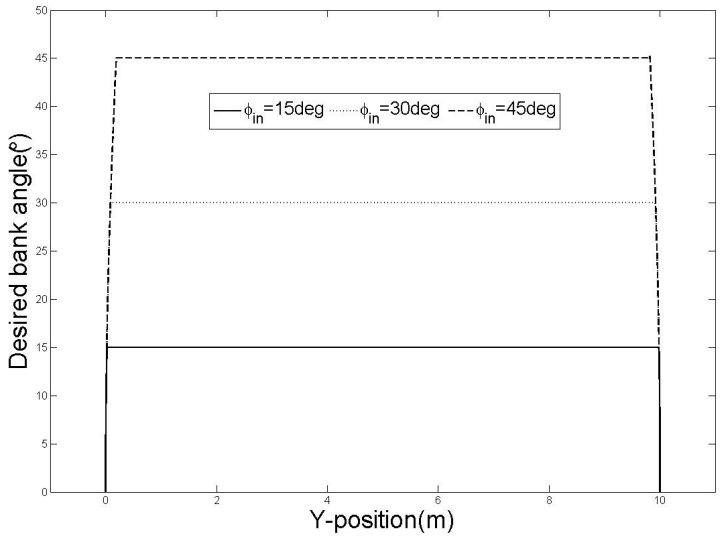


Figure 5–4: Desired bank angle of the vehicle,ϕ_d as a function of its y-position for $D = 10$m and $R_{in} = 1$m

the desired speed is set to u_{in} before the start of the maneuver so that it normally enters the maneuver at the proper desired speed.

Figure 5–4 shows the desired bank angle of the vehicle,ϕ_d, as a function of the y-position for three different value of ϕ_{in}. The desired bank angle increases at a maximum rate of $45.8°/s$ until it reaches its final value. The desired bank angle then remains at ϕ_{in} until the end of the second quarter-circle turn, at which point it returns to zero. The desired bank angle does not return to zero between the two quarter-circle turns in order to maintain a constant speed. The paddles produce both the propulsive and the control forces. When rolling, some of the thrust is used to roll the vehicle instead of propelling it forward, resulting in a loss of speed. As a result, it is more efficient for the vehicle to remain banked during the straight part of the turn, as there is no performance penalty for doing so.

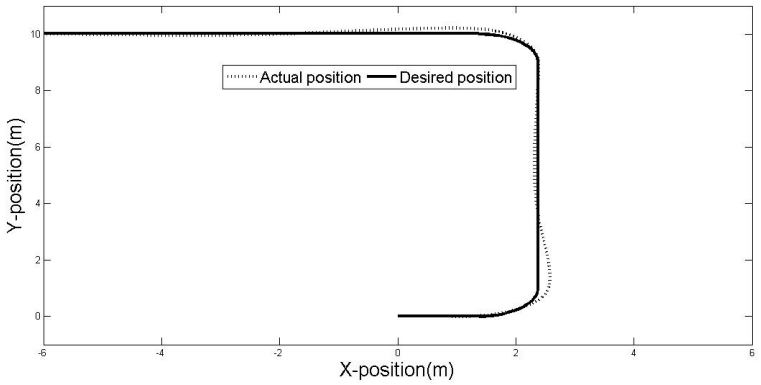


Figure 5–5: Desired and actual position of the vehicle for $D = 10$m and $R_{in} = 1$m.

5.3 Control of the vehicle

As was mentioned in Section 3.1, the dynamics simulation takes the desired trajectory, such as those shown in Figs. 5–2, 5–3 and 5–4, as input. The objective of the vehicle controller is to make the vehicle track the desired trajectory as closely as possible. We chose to use a PID controller because it was simple and gave good results in the evaluations of Chapter 4. This controller was described in detail in Section 3.2 and tested in Chapter 4. It was tuned specifically for this maneuver and gave the same performance as the more advanced controllers in this specific maneuver.

Figures 5–5 and 5–6 show how the PID controller tracks the desired trajectories for a turn with $D = 10m$, $R_{in} = 1m$, $u_{in} = 0.4m/s$ and $\phi_{in} = 0°$. The vehicle tracks the desired path reasonably well. Although there is some overshoot at the end of each quarter-circle turn, the steady-state error is negligible. It is important to note that the vehicle tracks the desired yaw angle accurately and that the overshoot is due to the sideslip velocity. This velocity is induced by the Coriolis force and since we

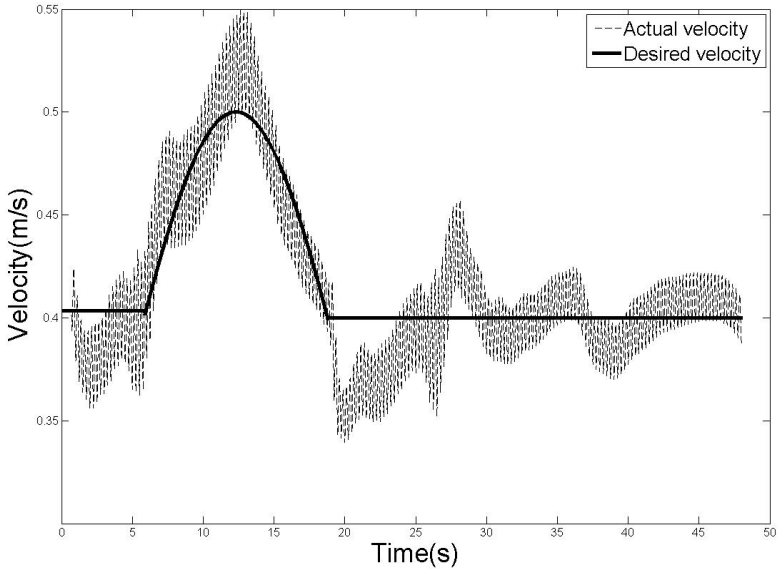


Figure 5–6: Desired and actual forward speed of the vehicle for $D = 10$, $R_{in} = 1$m and $u_{in} = 0.4$m/s.

have no control in the lateral direction, it cannot be corrected directly. The vehicle tracks the desired speed reasonably well. Since the propulsive force is generated by oscillating paddles, the velocity also oscillates throughout the maneuver. We can also see that the vehicle loses speed at the beginning of each quarter-turn but regains it relatively quickly.

5.4 Objective function

The objective or fitness function is a metric of the quality of the maneuver being optimized. In our specific situation, this function was defined as the time needed to complete the maneuver. It is evaluated by running the MATLAB Simulink simulation and recording the time history of the vehicle position. To do this, we had to define a criterion to determine when the maneuver is considered completed. The criterion for

turn completion was defined based on the RMS error of the position of the vehicle, which is written as:

$$E_{RMS} = \frac{1}{n}\sqrt{\sum_{i=0}^{n-1}\left(y\left(N_5 - i\right) - D\right)^2} \tag{5.2}$$

where $y(j)$ is the lateral position of the vehicle at instant j and N_5 is the time at which the vehicle reached the position $x = -5m$ (see Figure 5–2). This position was chosen because at that point, the tracking error was essentially zero in all cases, meaning that the maneuver was complete. We begin evaluating E_{RMS} at $x(N_5)$=-5m and back up along the trajectory (increasing values of n). The time to complete the maneuver,t_s , called the settling time, is defined as the instant n at which E_{RMS} first exceeds 1.5% of the desired track line.

5.5 Constraints

During the optimization search, to find the optimal values of R_{in}, u_{in} and ϕ_{in}, constraints are imposed on the allowable range of values that these design variables can have. The purpose of these constraints is to guarantee that the solution to the optimization problem remains within the range of the vehicle capabilities. Moreover, it reduces the search space of the algorithm which reduces the computing time. The three design variables were bounded as follows:

$$
\begin{aligned}
0.05 \leq R_{in} &\leq \frac{D}{2} \\
0 \leq \phi_{in} &\leq \frac{\pi}{2} \\
0.3 \leq u_{in} &\leq 0.6
\end{aligned} \tag{5.3}
$$

The lower bound on R_{in} represents the smallest turn that the vehicle was able to make during our experimental tests, while the upper bound represents a perfect semicircular trajectory. The bounds on the desired bank angle represent a pure yaw turn and pure bank turn. For $\phi_{in} = 0°$, the vehicle will turn while staying perfectly level. For $\phi_{in} = 90°$, the vehicle will turn by performing a pitching motion. Finally, the bounds on the desired speed were based on the desirable operating range of the vehicle.

5.6 Genetic algorithm

There exists many algorithms to solve optimization problems, and this section describes the rationales for our choice of algorithm. The following criteria were considered to choose among the many existing algorithms:

- capable of solving multivariable problems
- capable of handling constrained problems
- robust to local minima
- readily available in MATLAB
- no analytical gradient required

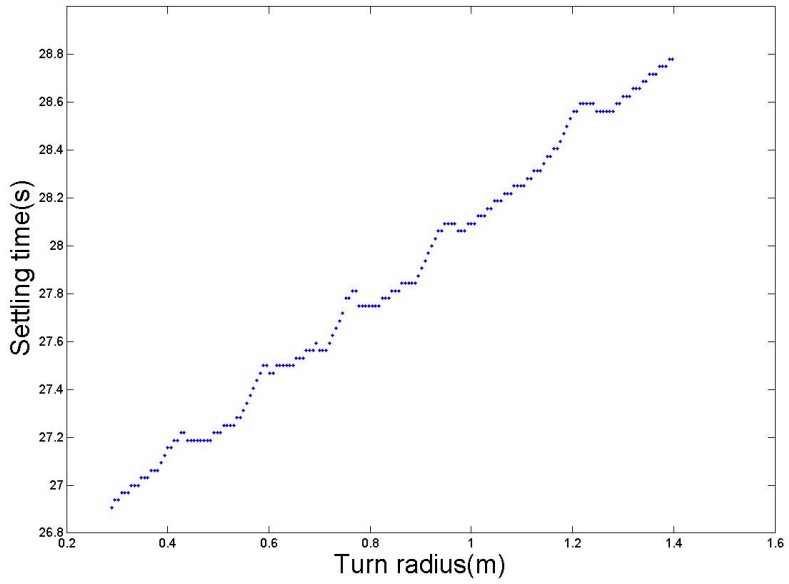


Figure 5–7: settling time as a function of turn radius(R_{in}) for $D = 10$m, $u_{in} = 0.4m/s$ and $\phi_{in} = 0°$

The two most readily available algorithms in MATLAB are a steepest descent method and a genetic algorithm. Both algorithms are able to handle constrained multivariable problems and do not require an analytical gradient. However, the genetic algorithm can better handle local minima. Figure 5–7 is a plot of the settling time as a function of the turn radius for a track line separation of $D = 10m$. It was obtained by running the simulation for different turn radii with $u_{in} = 0.4m/s$, $\phi_{in} = 0°$ and recording the settling time. There are many local minima along the objective function and a gradient-based method could easily become trapped in one of them. Based on this, we chose to use the genetic algorithm method to optimize the maneuver.

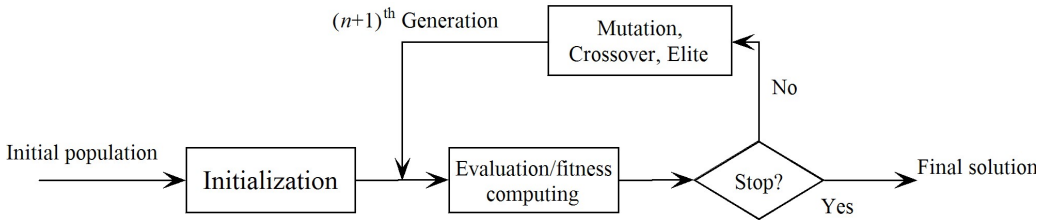


Figure 5–8: Genetic algorithm structure

A genetic algorithm is a search technique to find an approximate solution to an optimization problem [83]. The structure of the genetic algorithm is shown in Figure 5–8. It begins with an initial population that initializes the search process. The fitness of the n-th generation is evaluated in the block "evaluation/fitness computing". This operation is performed by running the simulation with the appropriate R_{in}, u_{in} and ϕ_{in} and determining the corresponding t_s. If the generation satisfies the convergence criterion, the algorithm stops and outputs the solution. Otherwise, a new generation is created. The n+1 generation is divided into elite, crossover and mutated members. The elites are the best members of the previous generation, and they survive from one generation to the next. The crossovers are obtained by mixing the genes (design variables) of elite members and the mutated are obtained randomly based on the existing members of the previous generation. In our case, the genes correspond to the design variables, i.e., turn radius, desired speed and desired bank angle. This process continues until a solution is found.

The genetic algorithm is implemented using the MATLAB function ga.m. This function takes as input the fitness function, the number of variables, the constraints, the initial population, the stopping criteria, the number of elite, crossover and mutated members, the maximum number of generations and the tolerance. These inputs

were set at the beginning of the optimization and could not be modified while the optimization was running. It outputs the optimal set of design variables as well as the value of the fitness function for that optimal set. The function ga.m finds the local minimum of the fitness function subject to the constraints described in (5.3). The algorithm evaluates the fitness function for each member (set of genes) of the population. Then, a new population is created based on the performance of the previous population. The process is repeated until the stopping criterion is met.

To allow the optimizer to evaluate the objective function, the MATLAB Simulink simulation of the vehicle was set up as a callable function. The design variables were the inputs to the function while the settling time to complete the maneuver was the output. The user specifies the initial population of design variables for the optimizer. In our case, the initial population was set to random numbers within the constraints defined in Section 5.5.

We considered single-variable optimizations and multi-variable optimizations. In the single-variable case, only one design variable was allowed to vary while the others were held fixed. In the multi-variable optimizations, multiple design variables were varied simultaneously. In the single variable problem, we consider a population size of eight individuals. The optimizer evaluates the objective function for each individual and then keeps the best one (the elite member) for the next generation. Since the genes have only a single design variable, crossover members cannot be created. Thus, the seven other members of the new generation are mutated members. The mutants are generated by the MATLAB function mutationadaptfeasible.m. This function randomly generates directions that are adaptive with respect to the last

successful or unsuccessful generation. The feasible region is bounded by the constraints and inequality constraints. A step length is chosen along each direction so that linear constraints and bounds are satisfied. The process of propagating populations is repeated until no improvements are observed after 10 generations. At that point, the optimizer switches to a gradient-based method to refine the solution. The gradient-based method uses the MATLAB function fmincon.m, based on a Sequential Quadratic Programming algorithm.

For the multivariable optimization, we use a population of 12 individuals instead of eight. The reasoning behind that change is that since there are two additional design variables, the search space is larger. Three elite members that are retained from one generation to the next. Of the remaining nine members of the population, six are mutants, while the three others are crossover members. The crossovers are obtained by mixing the design variables of the elite members. A different approach was used to create the mutants for the multivariable case. The genes of the offspring are obtained as the average of the genes of randomly-chosen parents, and a Gaussian-distributed variation is then added. The standard deviation of the distribution decreases with each generation:

$$\sigma_n = \sigma_{n-1}\left(1 - s\frac{n-1}{N}\right) \tag{5.4}$$

where σ_n is the standard deviation at generation n, s is a shrinking input parameter that determines how quickly the standard deviation decreases and N is the maximum number of generation. As before, each mutant must meet the bounds shown in (5.3).

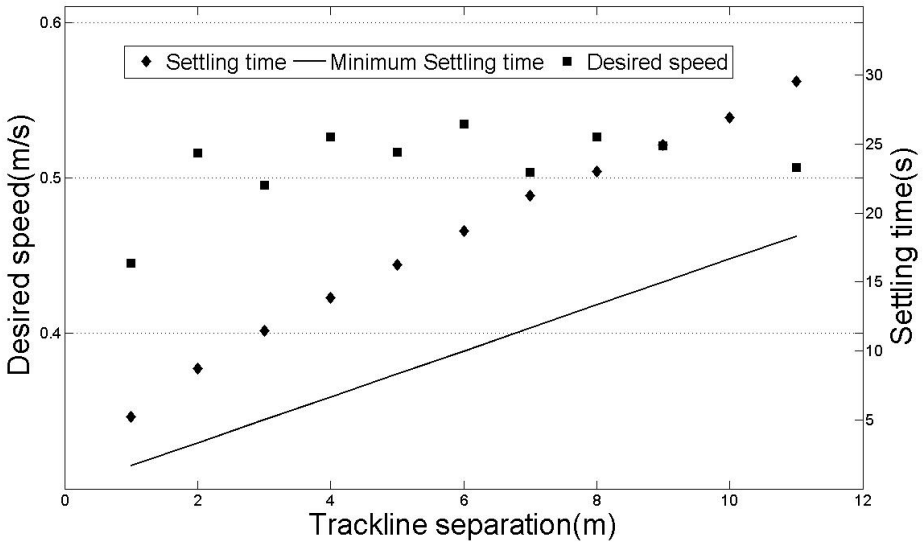


Figure 5–9: Optimum settling time t_s and optimal desired speed u_{in} for different track line separations D. The turn radius was set to $R_{in} = 0.25m$ and the desired bank angle at $\phi_{in} = 0°$.

5.7 Optimization Results

We now consider the results of the optimization described in the preceding section. This section is separated into two parts: single variable optimization and multivariable optimization.

5.7.1 Single variable optimization

The algorithm was run for a range of trackline separations, from $D = 1m$ to $D = 11m$. A single design variable was allowed to vary in each case and the two other variables were held constant. When they were held constant, the desired velocity of the vehicle was fixed at 0.5 m/s, the desired turn radius at 0.25m and the desired bank angle at 0°.

Figures 5–9(speed), 5–10(turn radius) and 5–11(bank angle) show the results of the three single variable optimizations. In addition to the value of the design variable

and settling time, the minimum settling time is also plotted. The minimum settling time is a function of the trackline separation only and is defined as the minimum time it would take to go from one trackline to the other if there were no physical limitations. It is obtained by assuming a turn radius of $0m$ in (5.1) and that the vehicle can go as fast as the upper bound stated in (5.3):

$$T_{s_{minimum}} = \frac{D}{0.6} \tag{5.5}$$

In Figure 5–9, we find that the optimal speed is somewhat lower for the smallest track line separation but then remains relatively constant at around 0.5 m/s for all the others. This implies that for small track line separation, it is better to go slower to perform the turn. Beyond that, increasing the speed does not improve the performance. We also see that the settling time increases almost linearly with track line separation. This is expected since the speed is almost constant and from Eq. (5.1) the total distance increases linearly with track line separation, implying that the trajectories are similar but with different track line separation. From Figure 5–9, we see that the optimal speed never reaches the upper bound of $u_{in} = 0.6$m/s. At this upper bound, the overshoot in lateral position is large and this negatively affects the settling time.

In Figure 5–10, we see that the settling time again increases with track line separation. The only exception is from 6.0 to 7.0m where the settling time decreased. There is also a large change in the optimal turn radius between those two track line separations. This is explained by the loss of velocity when the vehicle makes a sharp turn. Because of its design, the propulsive and control forces are coupled on Aqua

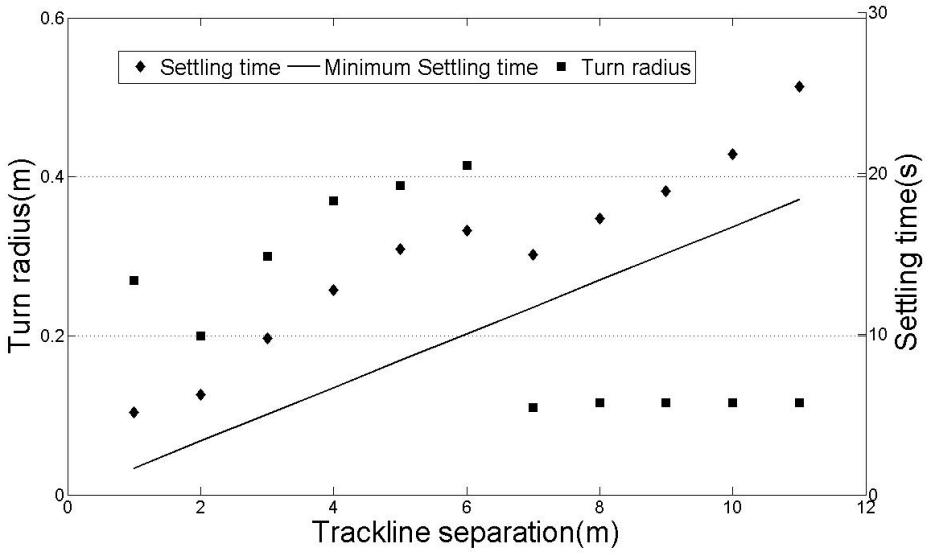


Figure 5–10: Optimum settling time t_s and optimal turn radius R_{in} for different track line separations D. The desired speed was set to $u_{in} = 0.5m/s$ and the desired bank angle at $\phi_{in} = 0°$.

and, as a result, changing the heading affects the speed. Even if the desired velocity remains at 0.5m/s, the vehicle does not have the capability to turn quickly while maintaining a constant forward speed. For large D, the loss of speed during the turn is not as significant because there is more distance to re-accelerate. However, for a smaller D, the vehicle does not have time to accelerate to its desired speed before it needs to turn again. In that case, the results suggest that it is better to take a wider turn that requires less control effort. It is important to note that these results were obtained for a constant desired speed of 0.5m/s. The results would be different for a different speed. For track line larger than 7.0m, the optimal turn radius remains constant at around 0.11m. We expect that it would remain at that value of R for all $D > 7.0$m.

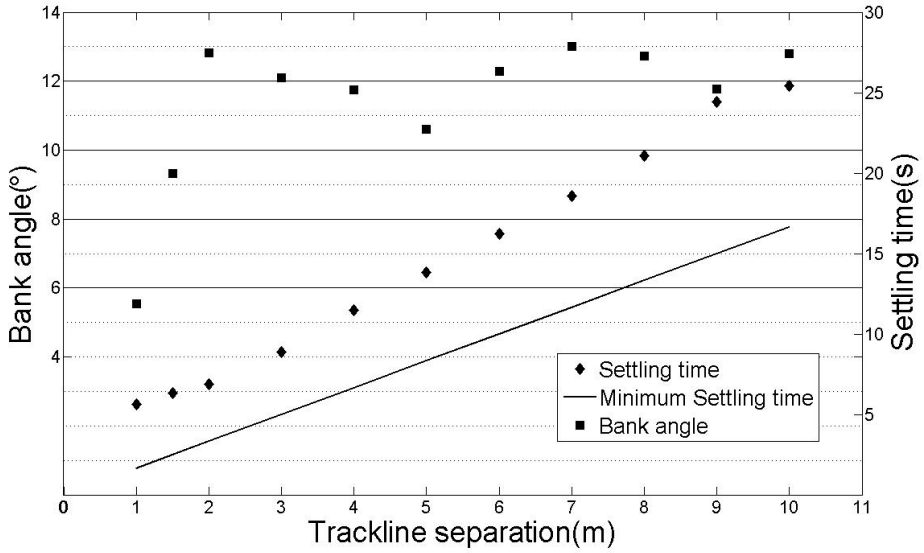


Figure 5–11: Optimum settling time t_s and optimal bank angle ϕ_{in} for different track line separations. The desired speed was set to $u_{in} = 0.5m/s$ and the turn radius to $R_{in} = 0.25m$.

In Figure 5–11, we see that the optimal bank angle is never zero, implying that banking improves the vehicle performance for this maneuver. The optimal bank angle initially starts low and then levels out at 12°. Moreover, we find from looking at the detailed optimization data that the optimal bank angle reduces the settling time by about 15% relative to the pure yawing turn for $2m \leq D \leq 7m$, which is significant.

It is difficult to explain why the optimal bank angle is not higher for small track line separation as one might expect. We find that the bank angle has less impact on the settling time for these cases. The difference in settling time between a bank angle of 6° and 12° for D=1m is only 0.2s. Moreover, increasing the bank angle to 45° only increases the settling time by 0.8s relative to the optimal bank angle.

For track line separations between 8m and 10m, the bank angle does not change the performance significantly. This was expected since the turn radius remains constant for all track line separations. Thus, as D increases, the straight part of the maneuver becomes dominant and that part of the maneuver is unaffected by the bank angle. Therefore, from these results, we conclude that a bank angle can reduce the settling time for $2m \leq D \leq 7m$, while outside that range, the improvement is less significant.

5.7.2 Multivariable optimization

The algorithm was again run for a range of trackline separations using the desired speed u_{in} and turn radius R_{in} as the design variables. We chose this pair of variables because they are the easiest to control and because we expect them to have the most effect on the performance. In Eq. (5.4), the initial standard deviation (σ_0) was set to half the range of the design parameters, the shrinking parameter (s) was set to 1 and the maximum number of generations (N) to 75. With s set to 1, the standard deviation decreased linearly to 0 as it reached the maximum number of generation N.

Figure 5–12 and 5–13 shows the results for a two-variable optimization with desired speed and turn radius as the design variables. The first thing we notice in Figure 5–12 is that for large values of $D \geq 7m$, the optimal speed and turn radius are the same as those found in the single variable optimization. Moreover, for $4m \leq D \leq 7m$ the optimal turn radius is almost the same, while the optimal desired speed changes. This implies that it is better to take a shorter path (i.e., a small turn radius) at slower speed than to maintain a higher speed and take a longer path.

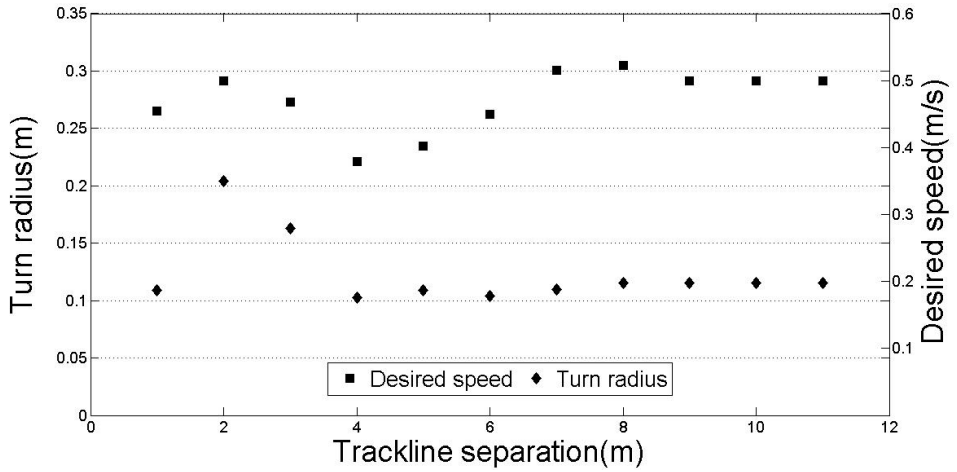


Figure 5–12: Optimal turn radius and desired speed for different track line.

Trackline separation(m)	1	2	3	4	5	6	7	8	9	10	11
2 variables(s)	4.97	6.25	8.50	10.37	11.45	12.52	14.12	16.87	18.90	21.17	25.40
Single variable(s)	5.15	6.25	9.75	12.75	15.27	16.42	14.92	17.18	18.90	21.17	25.40
Difference (%)	3.52	0.00	14.71	22.89	33.36	31.14	5.66	1.78	0.00	0.00	0.00

Table 5–1: Settling time for the two-variable optimization and for the best result obtained in the single variable optimization

Track line separations between 2 and 3 meters are the exception to this rule. In those two cases, the optimal turn radius is higher and the desired speed remains high. It means that it is better to take a longer path at higher speed. We also notice that both the turn radius and desired speed decrease for $2m \leq D \leq 4m$. After $D = 4m$, it becomes better to take a shorter path at a lower speed.

Figure 5–13 and Table 5–1 show the settling time obtained in the two-variable optimization shown in Figure 5–12 and compares it to the best time obtained in Figures 5–9 or 5–10. The optimal settling time is computed using (5.5). It shows that the settling time obtained with the two-variable optimization is always less or equal to the best settling time obtained in the single variable optimizations. For large track line separations, the results are similar because we were already using the

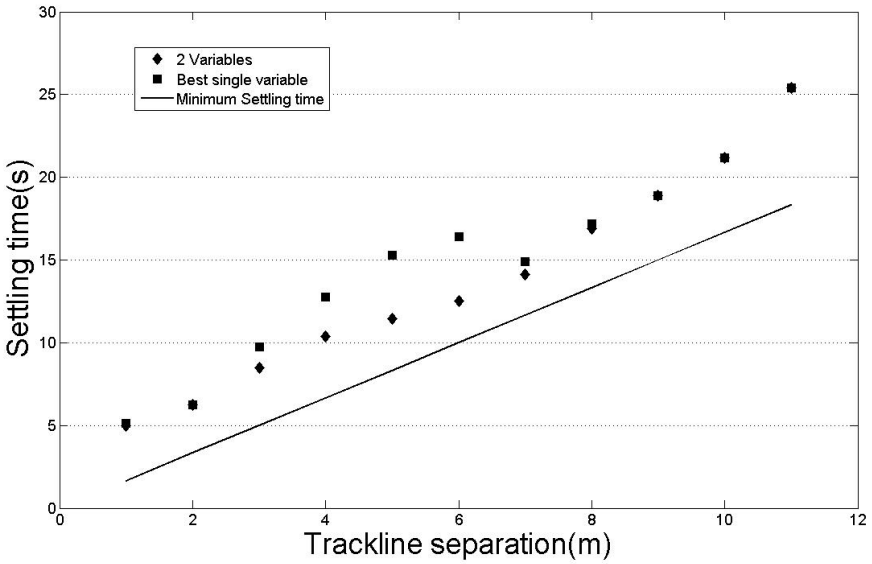


Figure 5–13: Settling time for the two-variable optimization and for the best result obtained in the single variable optimization.

Trackline separation(m)	1	2	3	4	5	6	7	8	9	10
Optimal Settling time(s)	1.67	3.33	5.00	6.67	8.33	10.00	11.67	13.33	15.00	16.67
Settling time(s)	4.97	5.75	8.50	10.37	11.45	12.52	14.12	16.87	18.90	21.17
Turn radius(m)	0.109	0.103	0.163	0.103	0.109	0.110	0.110	0.110	0.106	0.106
Desired speed(m/s)	0.455	0.464	0.468	0.380	0.402	0.455	0.5156	0.5156	0.5156	0.5156
Bank angle(°)	0.00	36.24	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table 5–2: Results of the three variables optimization

optimal desired velocity in the single variable optimization. The difference is more pronounced for $4m \leq D \leq 6m$. This is due to the fact that, for those track line separations, the optimal turn radius and desired speed obtained in the two-variable optimization are significantly different from those obtained in the single variable optimizations. In all cases, the optimal desired speed decreased while the optimal turn radius became constant at around 0.11m. That turn radius is the same as that obtained for $D \geq 7$ in the single variable optimization with turn radius as the design variable.

The logical next step is to perform an optimization with all three design variables. As was the case for the 2 variables optimization, the initial standard deviation (σ_0) was set to half the range of the design parameters, the shrinking parameter (s) was set to 1 and the maximum number of generations (N) to 75. Table 5–2 shows the results of the three-variable optimization. We can see that the settling time is the same as for the two-variable optimization except in the case where $D = 2m$. For that trackline separation, there was a small improvement in settling time by using a non-zero bank angle. Moreover, we see that the desired speed and turn radius both decrease relative to the two-variable optimization (Figure 5–12. These results imply that for all trackline separation, it is better to take the shortest possible path and adjust the speed and bank angle accordingly. Figure 5–14 shows how the bank angle affects the settling time for D=6m, u=0.45m/s and R=0.104m. The turn radius and desired speed are the optimal values found in both the two-variable and three-variable optimization. We see that a non-zero bank angle does not improve the performance of the maneuver when we use the optimal turn radius and desired speed. This implies that the best turn is done by using a turn radius of about 0.109m and adjusting the speed of the vehicle without any bank angle. You can also see on Figure 5–14 that once you reach a bank angle of 11°, increasing the bank angle does not affect the settling time anymore.

5.8 Summary of the Optimization

In this chapter, we found that the optimal turn radius is around $R_{in} = 0.11m$ for all D except for $D = 3m$. The optimal speed u_{in} is not constant and is varying for $D \leq 6m$. For $D \geq 7m$, the optimal speed remains constant at $u_{in} = 0.52m/s$.

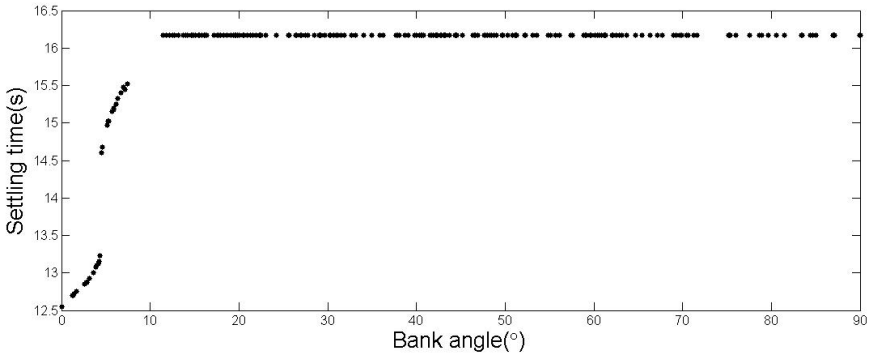


Figure 5–14: Settling time as a function of the bank angle for D=6m, $R_{in} = 0.110m$ and $u_{in} = 0.455m/s$

The optimal bank angle was zero for all D except for $D = 2m$ where $\phi_{in} = 36°$. In general, this implies that the best turn is accomplished by taking a small R_{in} and adjusting the speed u_{in} and bank angle ϕ_{in} according to Table 5–2. By using these optimal parameters, we were able to significantly reduce the settling time t_s in comparison to a semi-circular turn with the same speed. For example, for $D = 1m$ the settling time can be reduced by 30.4% and for $D = 10m$ by 31.6%.

These results could be used for a coral reef inspection. The parameters presented in Table 5–2 would be stored in the vehicle memory and used when changing trackline. Before changing trackline, the vehicle would compute the new trajectory based on the optimal parameters found in the optmization. A different trajectory would be computed for each trackline change. The approach used in this chapter could be applied to other maneuvers and vehicles. The method presented could be adjusted by changing the design parameters that would be particular to the new maneuver or to special features of the vehicle.

CHAPTER 6

Conclusions and Recommendations for Future Work

6.1 Conclusions

The main goal of this research was to develop trajectory tracking controllers for a biomimetic autonomous underwater vehicle, using the Aqua vehicle as a test case. A dynamics model of Aqua was developed to allow the design and evaluation of advanced controllers. The controllers were tested in a computer simulation and experimentally. Finally, an optimization of a high performance maneuver was performed to accelerate the coral reef inspection procedure.

The study began with the development of a dynamics model for the Aqua underwater vehicle. The dynamics model of the vehicle was validated experimentally and we concluded that it was accurate in average surge velocity, roll, pitch and yaw. A method based on an optimization to evaluate the thrust produced by an oscillating paddle was proposed and validated. We found that the thrust produced by an oscillating paddle was proportional to the ratio of amplitude over period of oscillation $\frac{A}{P}$. We also found that the flexible paddle was more efficient and could produce more

thrust than the rigid one. A reverse mapping that transforms the desired paddle thrust into paddle motion was proposed. It was based on an optimization to minimize the total thrust needed by all six paddles. A technique to linearize time-periodic systems based on differentiation by finite difference was proposed. We found that the main diagonal of the state matrix **A** were the most important entries. We also found that the constant part of the dynamics was more important than the time-periodic part. The linear model was validated against the nonlinear one and was found to be accurate.

With an accurate dynamics model available, we proceed to the design of trajectory tracking controllers for Aqua. Four classes of controllers were developed: PID, model-based controller, adaptive controller and Floquet controller. The PID controller is the simplest and depends only on the error signals. Two model-based controllers were developed: linearizing and nonlinear. An adaptive controller that updates its internal model for modeling uncertainties, changing vehicle dynamics or disturbances, was developed. A Floquet controller was designed using the theory of time-periodic systems. The controllers were tested in a computer simulation and experimentally to assess their performance. We found that the Floquet and adaptive controllers gave the best overall performance of all controllers in both the simulation and the experiment. We obtained the best settling time in roll with the Floquet controller, while the adaptive controller was a close second. Overall, we concluded that the Floquet controller was more accurate in roll while the adaptive controller performed better in pitch. Furthermore, we found that the adaptive law converged with acceptable accuracy.

An optimization of a U-turn maneuver was performed to improve the vehicle performance in coral reef inspection. The objective was to minimize the time taken to complete the U-turn. Three design variables were chosen for the optimization: the turn radius, the speed and the bank angle. We found that the turn radius R_{in} and speed u_{in} have the most impact on the performance, while the bank angle has a lesser impact. We also found that it is always better to take a short path and adjust the speed of the vehicle to minimize the time to complete the maneuver. Furthermore, we found that the optimization results are significantly better than for a semi-circular trajectory. The settling time was reduced by more than 30% in general.

6.2 Recommendations for Future Work

Based on the research presented in this thesis, several avenues for future research can be suggested:

- One of the key similarities of all biomimetic underwater vehicles (BAUV) is the presence of oscillating thrust. This thrust is often characterized by the period and amplitude of oscillation of the fin which creates a situation where there is no unique mapping between thrust and paddle motion. The optimization-based mapping that was developed for Aqua could be applied to other BAUVs with similar thrust characteristics.
- One part of the mapping from control forces to paddle motion was the allocation of the thrust to the individual paddles which constitutes an underdetermined problem. We used an optimization that minimized the sum of the thrust used by the paddles. It would be interesting to investigate other methods for paddle

thrust allocation. Moreover, the technique we used could be applied to other redundant systems.

- The validated dynamics simulation of the vehicle could be used for application other than controller testing, such as swimming gait optimization. Other gaits such as walking or hovering gait could also be implemented in the simulation and optimized.
- The control techniques tested on Aqua could be implemented on other BAUVs. Furthermore, the dynamics model of Aqua could be adapted to other vehicle since it is based on physics equations rather than on curve fitting.
- The inflow velocity should be measured in future thrust measuring experiments for the flexible paddle, thus obviating the need to estimate it, as we are currently doing in Section 2.2.1. This would likely increase the accuracy of the forward flexible paddle model.
- The reverse paddle model should take into account the fact that changing the orientation of the paddle will produce a force perpendicular to the offset line. At the moment, the reverse model assumed that the paddle will produce a thrust only along the offset line. This perpendicular force can have a negative effect, but it can also be used to produce quick rolling and pitching motion.
- The dynamics model was not validated in all possible degrees of freedom. Acceleration in surge should be validated. However, this requires reliable sensors to measure the speed of the vehicle and/or its position. If those sensors were on-board and could be used during vehicle operation, they would also enable

the use of controllers in all degrees of freedom, which is not possible at the moment.

- The linearization discussed in Section 2.6 was performed for only three steady-state conditions. For conditions between those, gain scheduling was used for the Floquet controller. It would be useful to linearize the vehicle for more steady-state conditions as this might improve the performance of the Floquet controller.
- We have found that adaptive control was a good class of controllers to control BAUVs. However, among all techniques within adaptive control, only MIAC was evaluated on Aqua. Other techniques such as model reference adaptive control or command generator tracker could prove to give better performance than the MIAC presented in this thesis. These other controllers should be evaluated.
- Experiments in a controlled environment should be done before testing in an uncontrolled environment. This would give us more information to tune the gains and make appropriate modification to the robot code. Moreover, we would be able to compare the performance of the controllers with and without disturbances.
- To further improve the turn maneuver optimization results, the variation of desired speed and desired bank angle during the maneuver could be improved, perhaps by accelerating more quickly in the straight part of the maneuver. Moreover, it would be interesting to see if the results would differ if the population size and maximum number of iterations were increased.

- The results of the optimization should be validated experimentally. This would require accurate measurement of the position of the vehicle, which is not possible at the moment.
- A more accurate hydrodynamic model describing the interference between the front and back paddles should be investigated. This interference is likely significant but was neglected in the current work.

REFERENCES

[1] C. Fowler, "The museum of music: a history of musical instruments," *Music Educators Journal*, vol. 54, no. 2, pp. 45–49, 2005.

[2] J. Riskin, "The defecating duck, or, the ambiguous origins of artificial life," *Critical Inquiry*, vol. 29, no. 4, pp. 599–633, 2003.

[3] C. Prahacs, A. Saunders, M. Smith, D. McMordie, and M. Buehler, "Towards legged amphibious mobile robotics," *Journal of Engineering Design and Innovation*, vol. 1P, no. 01P3, 2005.

[4] U. Saranli, M. Buehler, and D. Koditschek, "Rhex:a simple and highly mobile hexapod robot," *International Journal of Robotics Research*, vol. 20, no. 1, pp. 616–631, 2001.

[5] J. D. Weingarten, G. A. D. Lopes, M. Buehler, R. E. Groff, and D. E. Koditschek, "Automated gait adaptation for legged robots," in *Proc. of IEEE International Conference On Robotics and Automation (ICRA)*, pp. 2153–2158, 2004.

[6] D. Campbell and M. Buehler, "Preliminary bounding experiments in a dynamic hexapod," *Experimental Robotics VIII*, pp. 612–621, 2003.

[7] U. Saranli and D. Koditschek, "Back flips with a hexapedal robot," *Proc. of IEEE International Conference On Robotics and Automation (ICRA)*, vol. 3, pp. 128–134, 2001.

[8] N. Neville and M. Buehler, "Towards bipedal running of a six legged robot," in *12th Yale Workshop on Adaptive and Learning Systems*, 2003.

[9] P. Giguere, *Unsupervised Learning for Mobile Robot Terrain Classification*. Doctor of Philosophy, McGill University, 2009.

[10] J. Sattar and G. Dudek, "A vision-based control and interaction framework for a legged underwater robot," in *Proc. of Canadian Conference on Robot Vision*, (Kelowna, BC), 2009.

[11] D. McMordie, *Towards pronking with a hexaped robot*. Master of Engineering, McGill University, 2002.

[12] M. Triantafyllou, A. Techet, and F. Hover, "Review of experimental work in biomimetic foils," *IEEE Journal of Ocean Engineering*, vol. 29, no. 3, pp. 585–594, July 2004.

[13] M. S. Triantafyllou and G. S. Triantafyllou, "An efficient swimming machine," *Scientific American*, vol. 272, pp. 64–70, March 1995.

[14] J. Yu, M. Tan, S. Wang, and E. Chen, "Development of a biomimetic robotic fish and its control algorithm," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 34, no. 4, pp. 1798–1810, August 2004.

[15] S. Licht, V. Polidoro, M. Flores, F. Hover, and M. Triantafyllou, "Design and projected performance of a flapping foil AUV," *IEEE Journal of Ocean Engineering*, vol. 29, no. 3, pp. 786–794, July 2004.

[16] M. Kemp, B. Hobson, and J. Long, "Madeleine: An agile AUV propelled by flexible fins," in *Proc. of 14th UUST*, (Durham, NH), 2005.

[17] O. Chiu, M. Nahon, and N. Plamondon, "Stability augmentation for a hexapod underwater vehicle," in *Proc. of the 15th UUST*, 2007.

[18] 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.

[19] J. Sattar and G. Dudek, "Underwater human-robot interaction via biological motion identification," in *Proceedings of Robotics:Science and Systems V, RSS, In Press*, (Seattle, WA, USA), July 2009.

[20] J. Sattar, P. Giguere, and G. Dudek, "Sensor-based behavior control for an autonomous underwater vehicle," *International Journal of Robotics Research*, vol. 28, no. 6, 2009.

[21] P. Giguere, G. Dudek, and C. Prahacs, "Characterization and modeling of rotational responses for an oscillating foil underwater robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3000–3005, Oct. 2006.

[22] G. Dudek, M. Jenkin, C. Prahacs, A. Hogue, J. Sattar, P. Giguere, A. German, H. Liu, S. Saunderson, A. Ripsman, S. Simhon, L. Torres-Mendez, E. Milios, P. Zhang, and I. Rekleitis, "A visually guided swimming robot," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, (Edmonton, Canada), 2005.

[23] R. Mittal, I. Akhtar, and M. Bozkurttas, "Towards a conceptual model of a bio-robotic AUV: Pectoral fin hydrodynamics," in *Proc. of the 13th UUST*, (Durham, NH), 2003.

[24] L. Schouveiler, F. Hover, and M. Triantafyllou, "Performance of flapping foil propulsion," *Journal of Fluid and Structures*, vol. 20, no. 7, pp. 949–959, 2005.

[25] K. Harper, M. Berkemeier, and S. Grace, "Modeling the dynamics of spring-driven oscillating-foil propulsion," *IEEE Journal of Ocean Engineering*, vol. 23, no. 3, pp. 285–296, July 1998.

[26] L. Guglielmini, P. Blondeaux, and G. Vittori, "A simple model of propulsive oscillating foils," *Ocean Engineering*, vol. 31, no. 7, pp. 883–899, May 2004.

[27] I. Yamamoto, Y. Terada, T. Nagamatu, and Y. Imaizumi, "Propulsion system with flexible/rigid oscillating fin," *IEEE Journal of Ocean Engineering*, vol. 20, no. 1, pp. 23–30, January 1995.

[28] J. Mollendorf, J. Felske, S. Samimy, and D. Pendergast, "A fluid/solid model for predicting slender body deflection in a moving fluid," *Journal of applied mechanics*, vol. 70, no. 3, pp. 346–350, 2003.

[29] X. Lu, J.-M. Yang, and X. Yin, "Propulsive performance and vortex shedding of a foil in flapping flight," *Acta Mechanica*, vol. 165, no. 3-4, pp. 189–206, 2003.

[30] T. Schnipper, A. Andersen, and T. Bohr, "Vortex wakes of a flapping foil," *Journal of Fluid Mechanics*, vol. 633, pp. 411–423, 2009.

[31] X. Deng and S. Avadhanula, "Biomimetic micro underwater vehicle with oscillating fin propulsion: System design and force measurement," in *IEEE International Conference on Robotics and Automation*, (Spain), August 2004.

[32] S. Guo, T. Fukuda, and K. Asaka, "A new type of fish-like underwater microrobot," *IEEE transactions on Mechatronics*, vol. 9, no. 8, pp. 64–70, March 1993.

[33] J. L. Jr., J. Schumacher, N. Livingston, and M. Kemp, "Four flippers or two? tetrapodal swimming with an aquatic robot," *Bioinspiration and Biomimetics*, vol. 1, no. 1, pp. 20–29, 2006.

[34] S. Licht, F. Hover, and M. S. Triantafyllou, "Maneuvering finnegan the roboturtle," in *Proc. of 15th UUST*, (Durham, NH), 2007.

[35] D. Yoerger and J.-J. E. Slotine, "Robust trajectory control of underwater vehicles," *IEEE Journal of Ocean Engineering*, vol. 10, no. 4, pp. 462–470, October 1985.

[36] H. Xu, M. Mirmirani, P. Ioannou, and H. Boussalis, "Robust adaptive sliding control of linearizable systems," in *Proc. of the 2001 American Control Conference*, June 2001.

[37] D. Smallwood and L. Whitcomb, "Model based dynamic positioning of underwater robotic vehicles: Theory and experiment," *IEEE Journal of Ocean Engineering*, vol. 29, no. 1, pp. 169–186, January 2004.

[38] Z. Feng and R. Allen, "H- autopilot design for an autonomous underwater vehicle," in *In Proc. Of the IEEE International Conference on Control Applications*, (Glasgow, UK), September 2002.

[39] J. Yuh, "Control of underwater robotic vehicles," in *In Proc. Of IEEE International Conference on Intelligent Robots and Systems*, (Yokohama, Japan), 1993.

[40] M. Caccia and G. Veruggio, "Guidance and control of a reconfigurable unmanned underwater vehicle," *Control Engineering Practice*, vol. 8, pp. 21–37, 2000.

[41] K.-Y. Kwon, J. Cho, B.-S. Yoo, and J. Joh, "Autonomous navigation of a underwater robot using fuzzy logic," in *Proc. of Annual Meeting of the North American Fuzzy Information Processing Society*, 2005.

[42] S.-M. Lee, K.-Y. Kwon, and J. Joh, "A fuzzy logic for autonomous navigation of marine vehicles satisfying COLREG guidelines," *International Journal of Control, Automation and Systems*, vol. 2, no. 2, pp. 171–181, 2004.

[43] I. Spangelo and O. Egeland, "Trajectory planning and collision avoidance for underwater vehicles using optimal control," *IEEE Journal of Ocean Engineering*, vol. 19, no. 4, pp. 502–511, October 1994.

[44] J. Yuh, E. West, and P. Lee, "An autonomous underwater vehicle control with a non-regressor based algorithm," in *Proc. of the 2001 IEEE International Conference on Robotic and Automation*, (Seoul, South Korea), 2001.

[45] S. Zhao and J. Yuh, "Experimental study on advanced underwater robot control," *IEEE Transactions on Robotics*, vol. 21, no. 4, pp. 695–703, 2005.

[46] J. Ritonja, D. Dolinar, and B. Grcar, "Simple adaptive control for stability improvement," in *Proc. of the 2001 IEEE International Conference on Control Applications*, (Mexico City, Mexico), 2001.

[47] M. Deng, I. Mizumoto, Z. Iwai, and S. Shah, "Model output following control based on CGT approach for plants with time delays," *International Journal of Systems Science*, vol. 30, no. 1, pp. 69–75, 1999.

[48] A. Coates, P. Abbeel, and A. Y. Ng, "Apprenticeship learning for helicopter control," *Communications of the ACM*, vol. 52, 2009.

[49] A. Coates, P. Abbeel, and A. Y. Ng, "Learning for control from multiple demonstrations," *Proc. of the Twenty-fifth International Conference on Machine Learning*, 2008.

[50] J. Jang and H. Chung, "Neuro-fuzzy network control for a mobile robot," *Proc. of the American Control Conference*, pp. 2928–2933, 2009.

[51] J. Guo and Y.-J. Joeng, "Guidance and control of a biomimetic autonomous underwater vehicle using body-fin propulsion," in *Proceedings of the Institution of Mechanical Engineers Part M: Journal of Engineering for the Maritime Environment*, vol. 218, May 2004.

[52] J. Geder, J. Palmisano, R. Ramamurti, W. Sandberg, and B. Ratna, "Fuzzy logic PID based control design and performance for a pectoral fin propelled

unmanned underwater vehicle," in *Proc. Of the 2008 International Conference on Control, Automation and Systems*, (Seoul, South Korea), 2008.

[53] M. Naik and S. Singh, "Oscillatory adaptive yaw-plane control of biorobotic autonomous underwater vehicles using pectoral-like fins," *Applied Bionics and Biomechanics*, vol. 4, no. 4, pp. 137–147, December 2007.

[54] S. Singh, A. Simha, and R. Mittal, "Biorobotic AUV maneuvering by pectoral fins: inverse control design based on CFD parameterization," *IEEE Journal of Oceanic Engineering*, vol. 29, no. 3, pp. 777–785, July 2004.

[55] J. Yu, S. Wang, and M. Tan, "Basic motion control of a free-swimming biomimetic robot fish," in *Proc. of the 42nd IEEE Conference on Decision and Control*, (Maui, HI), December 2003.

[56] J. Yu, E. Chen, S. Wang, and M. Tan, "Study of motion control algorithms for a free-swimming biomimetic robot fish," in *Proc. of the 2003 IEEE International Symposium on Intelligent Control*, (Houston, TX), October 2003.

[57] X. Dong, S. Wang, Z. Cao, and M. Tan, "Design of a networked control system for biomimetic robot fish," in *Proc. of the 2008 IEEE International Conference on Networking, Sensing and Control*, 2008.

[58] R. Calico and W. Wiesel, "Control of time-periodic systems," *Journal of Guidance*, vol. 7, no. 6, Nov-Dec 1984.

[59] P. Montagnier, C. Paige, and R. J. Spiteri, "Real floquet factors of linear time-periodic systems," *Systems and Control Letters*, vol. 50, no. 4, pp. 251–262, Nov 15, 2003.

[60] P. Montagnier, R. J. Spiteri, and J. Angeles, "The control of linear time-periodic systems using floquet-lyapunov theory," *International Journal of Control*, vol. 17, no. 5, pp. 472–490, March 20, 2004.

[61] L. Acho, Y. Orlov, and V. Solis, "Nonlinear measurement feedback h_{infty} control of time-periodic systems with application to tracking control of robot manipulators," *International Journal of Control*, vol. 74, no. 2, pp. 190–198, 2001.

[62] R. Brockett, *Finite Dimensional Linear Systems*. New York, NY: Wiley, 1970.

[63] Z. Cai, Y. Gu, and W. Zhong, "New approach of computing floquet transition matrix," *Computers and Structures*, vol. 79, no. 6, pp. 631–635, February 2001.

[64] C. Petres, Y. Pailhas, P. Patron, Y. Petillot, J. Evans, and D. Lane, "Path planning for autonomous underwater vehicles," *IEEE Transactions on Robotics*, vol. 23, no. 2, pp. 331–341, April 2007.

[65] C. Petres, Y. Pailhas, Y. Petillot, and D. Lane, "Underwater path planning using fast marching algorithms," *Oceans-Europe*, vol. 2, pp. 814–819, 2005.

[66] V. Kanakakis and N. Tsourveloudis, "Evolutionary path planning and navigation of autonomous underwater vehicles," in *Proc. of the 2007 Mediterranean Conference on Control and Automation*, (Athens, Greece), July 2007.

[67] S. Khanmohammadi, G. Alizadeh, J. Jassbi, and M. Pourmahmood, "A new artificial intelligence approach for 2d path planning for underwater vehicles avoiding static and energized obstacles," in *Proc. of IEEE Congress on Evolutionary Computation*, 2008.

[68] T. Shamir, "How should an autonomous vehicle overtake a slower moving vehicle: design and analysis of an optimal trajectory," *IEEE Transactions on Automatic Control*, vol. 45, no. 4, April 2004.

[69] S. Guo and B. Gao, "Path planning optimization of underwater microrobots in 3-D space by PSO approach," in *Proc. of the 2009 IEEE International Conference on Robotics and Biomimetics*, (Guilin, China), December 2009.

[70] C. Lambert, M. Nahon, B. Buckham, and M. Seto, "Dynamics and control of towed underwater vehicle system, partii: model validation and turn maneuver optimization," *Ocean Engineering*, vol. 30, no. 4, March 2003.

[71] H.-S. Kim, B.-R. Lee, T.-Q. Vo, and Q.-B. Truong, "A study on optimization of fish robot velocity using genetic algorithm," in *Proc. of the International Conference on Smart Manufacturing Application*, (Gyeonggi-Do, Korea), April 2008.

[72] T. Fossen, *Guidance and Control of Ocean Vehicles*. United Kingdom: John Wiley and Sons Ltd., 1994.

[73] C. Georgiades, *Simulation and Control of an Underwater Hexapod robot*. M.Eng. Thesis, McGill University, 2005.

[74] C. Georgiades, M. Nahon, and M. Buehler, "Simulation of an underwater hexapod robot," *Ocean Engineering*, vol. 36, no. 1, pp. 39–47, 2009.

[75] A. Healey, S. Rock, S. Cody, D. Miles, and J. Brown, "Toward an improved understanding of thruster dynamics for underwater vehicles," *IEEE Journal of Ocean Engineering*, vol. 20, no. 4, 1995.

[76] P. Gill, W. Murray, and M. Wright, *Practical Optimization*. San Diego: Academic Press, 1989.

[77] J. Guo, "Optimal measurement strategies for target tracking by a biomimetic underwater vehicle," *Ocean Engineering*, vol. 35, no. 5-6, pp. 473–483, 2008.

[78] P. Ioannou and J. Sun, *Robust Adaptive Control*. New York, NY: Prentice Hall, 1996.

[79] N. Plamondon and M. Nahon, "Control of an underwater robot using flexible oscillating paddles," in *Proc. of 15th UUST*, (Durham, NH), 2007.

[80] N. Plamondon and M. Nahon, "Trajectory tracking controller for an underwater hexapod vehicle," in *Proc. of Oceans Conference 2008*, (Quebec City, Qc), 2008.

[81] S. Fleischer, H. Wang, S. Rock, and M. Lee, "Video mosaicking along arbitrary vehicle paths," in *Proc. of the 1996 Symposium on Autonomous Underwater Vehicle Technology*, 1996.

[82] R. Marks, S. Rock, and M. Lee, "Real-time video mosaicking of the ocean floor," *IEEE Journal of Ocean Engineering*, vol. 20, no. 3, 1995.

[83] M. D. Vose, *The Simple Genetic Algorithm: Foundations and Theory*. Cambridge, MA: MIT Press, 1999.